THE CPCO FULL CORE PIDAL SYSTEM

SOFTWARE DESCRIPTION

G.A. Baustian
Reactor Engineering
Palisades

REV 0—October 20, 1987 P*Z*INCA*871020
REV 1—May 26, 1988 P*Z*INCA*871020
REV 2—January 19, 1989 P*C08*88003
REV 3—March 27, 1989 P*PID*89001
REV 4—June 05, 1989 P*PID*89002

ABSTRACT

This report provides a software description for the Palisades Incore Detector Algorithm, PIDAL. A detailed description of the full core PIDAL program and methodology is presented.

THE CPCO FULL CORE PIDAL SYSTEM

SOFTWARE DESCRIPTION
REV 4

TABLE OF CONTENTS

1- INTRODUCTION

2- DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.1 Program Section #1-- Input Collection
2.2 Program Section #2-- XTG Theoretical Calculations
2.3 Program Section #3-- Calculation of Detector Powers
2.4 Program Section #4-- Full Core Radial Power Distribution Calculation
2.5 Program Section #5-- Full Core Axial Power Distribution Calculation
2.6 Program Section #6-- Calculations for Monitoring Technical Specification Compliance
2.7 Program Section #7-- Calculation of Exposures, Incore Alarm Limits and Summary Output

3- LIST of TABLES

4- LIST of FIGURES

5- LIST of REFERENCES

6- GLOSSARY

INTRODUCTION

The program PIDAL was developed by Consumers Power Company to calculate incore power distributions for the Palisades reactor based on measured signals from fixed Rhodium incore detectors. PIDAL was designed to produce full core power distributions, with the intent of having the PIDAL program and methodology replace the older 1/8th core INCA model which had been used for the first eight fuel cycles at Palisades.

For purposes of analysis, this report divides the PIDAL main program into seven major sections. Within each major section, a set of specific tasks are performed that together complete a logical step toward the completion of each PIDAL case.

Figure 1.1 shows the major data flow paths and seven sections of the main program as they will be divided for this report.

Section #2 of this report is divided into seven subsections, each of which correspond directly with one of the major program sections of figure 1.1. Within each of the seven subsections, the tasks performed and subroutines used are discussed in detail.

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.1 Program Section #1

Program section #1 controls the input of most of the data used in the PIDAL calculations. The text that follows provides descriptions of the individual tasks performed by the PIDAL main program in program section #1.

The variables read in from the card image hand inputs are listed in Table #2.0.1 with a short description of each given. For a more descriptive discussion of the format of the hand inputs the reader should consult Ref #2 Attachment #1. Some important notes on these inputs are summarized as follows.

Card Image Hand Inputs

The variables BLOCK, DT, TSTART and TEND are primarily used as input for an PIDAL exposure case. As input, they have no significance for a power distribution case. The major difference between a power distribution and an exposure case is that for an exposure case, PIDAL and XTG perform fuel burnup calculations based on the current power distribution and core thermal energy given by variable DT. From the burnup calculations, PIDAL determines fuel cycle and batch average exposures, along with control rod exposures for inserted control rods. For a power distribution case, no burnup calculations are performed. The primary use for a power distribution run is to obtain an up-to-date picture of the core power distribution and update the incore detector high alarm limits.

The optional report flag IPRIO (card 2) is used to produce maps of radial and axial detector locations, assembly numbering schemes and assembly fuel type identification. If IPRIO>0 then Edits #3,4 and 5 are produced.

The flag IED (card 2) is used to notify PIDAL that non-default edits are being requested. If IED=1 PIDAL will read in card 10 for edit processing. If IED=0 then PIDAL generates default edits.

The flag IXPOW notifies PIDAL to use XTG detector powers instead of powers from the incore detectors. See sections #2.4.5 and #2.4.11.

The flag IOUT (card 2) is used to activate the PIDAL statistical analysis routines BDZOIK and RECALC. These routine are used only when a detailed statistical analysis of the PIDAL solution is required. For more information see Reference #1.

The control rod positions and incore detector signals are normally supplied to PIDAL via the PIP inputs. Control rod positions may be hand entered either indiviually or by group by setting RODSIN (card 2) and supplying data on either card 4 or 5. Incore signals may be entered via variable N (card 7) and by supplying data on card type 8. If a detector signal is overlaid, then a corresponding flag in the array IFAIL is set to '+'. Overlaid detector signals are reported by Subroutine OPTION.

2.1 Program Section #1

Card Image Hand Inputs (continued)

Excore detector readings can be input to PIDAL to monitor excore quadrant tilt and axial shape index by setting the flag IXCORE (card 2) and supplying values for the variables XCORE and either XCPR or ASI on card types 3a and 3b. XCPR is used for cycles 5 through 7 and ASI is used for cycles after cycle 7.

The variables IPTAPE, IPCTPW, DTEPAL and COMENT are not used by PIDAL for any calculations but instead serve as part of the case title. IPTAPE is used by the Retrieve support program to fetch the correct PIP information from the UNTRANS data file. These variables are read from card 1.

The variable RUNDTE (card 1) is the PIDAL computer run date and is optional because PIDAL will calculate it at run time using the IBM TIMER routine. If RUNDTE is supplied, then that data becomes the run date for the PIDAL case.

The variables SOLBOR, CALPOW, PSIA, O and TIN are optional hand inputs used to overide reactor operating characteristic values supplied by the PIP. If no value is input for these variables, then the PIP equivalents will be used. These values may be input on card 6. See the discussion for Subroutine DECIDE, Section 2.1.6, for further information on these variables.

The flag IRAD (card 7) is currently unused.

Variable PMIN is the minimum acceptable detector power to be used by PIDAL. If any detector has a power less than PMIN (in MWth) then it will be failed in Subroutine CHPMIN. The PMIN is not supplied on card 7 then it's value defaults to 0.1 MWth.

The uncertainty factor UNUSER (card 7) is used to raise or lower the calculated peaking factor and LHGR values. It can effectively raise or lower incore detector high alarm limits. UNUSER is generally not used.

Variable RUNRQT (card 9) is used only for power distribution cases. If a power distribution case is to mate (restart) with any exposure case other than the last one in the exposure file then RUNRQT must be set equal to the computer run date of the desired exposure case.

The variable HISTAT (card 9) specifies that the current PIDAL case is a BOC run. This forces the PIDAL restart to come from the first record of the exposure file which is generated by the Shuffle support program.

The flag TAPPUN (card 9) is used to produce alarm limit set points. If TAPPUN=1 then alarm limits will be written to FORTRAN unit=40. If TAPPUN=0 then alarm limits will be written only for power distribution cases. If TAPPUN=-1 then no alarm limits will be generated.

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.1 Program Section #1

Discussions of miscellaneous tasks performed within the main program follow.

Initialize Case Data

This section of code initializes several PIDAL variables. This is also the point to which PIDAL returns if it is run using the multiple case option. (Multiple cases are run by supplying consecutive sets of hand input cards and PIP data on FORTRAN units 5 and 14 respectively)

Wprime Library Title

The title record and control rod correction factors are read from the W' library in the main program. The title identifies the current generation of library and is edited by Subroutine PAGE1A. The control rod correction factors are used by Subroutine BXPWR.

Plant Computer (PIP) Data

Input from the Varian Datalogger (PIP) Untrans data file contains plant operating data including incore detector: millivolt signals, sensitivities, backround correction and flux scaling factors. Also included are control rod positions, primary and secondary loop flow conditions, reactor and steam generator powers and an identifying time for the data dump. Table #2.0.2 is a list of all of the PIDAL variables read in from the Untrans data file on FORTRAN unit=14.

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.1 Program Section #1

Cycle Fuel Exposure Calculation

After a restart from the Exposure data file is completed, PIDAL performs a calculation of current cycle core exposure by utilizing the following equation:

$$CUREXP = TOTEP - BOCEXP + DT/(24.0 * FUELM) \quad (MWD/MTU)$$

where:

- TOTEP = Core average exposure from last case (MWD/MTU)
- BOCEXP = Core average exposure at BOC (MWD/MTU)
- DT = Energy distributed for this case (MWhrs) input on card #6
- FUELM = Total fuel mass in core (MTU)

The calculated core exposure value is then used in the HFP ARO critical Boron concentration calculation of Subroutine CBORON and as restart file input ti XTG via Subroutine PREXTG.

Variable	Type	Description	
BLOCK	INTEGER	Identifying PIDAL case number	
IPTAPE	INTEGER	Number of datalogger dump to be used	
IPCTPW	INTEGER	Reactor percent power for current case	
DTEPAL	REAL DTEPAL(4)	Date of burnup or power distribution	
COMENT	REAL COMENT(10)	Free format comment for PIDAL title page	
RUNDTE	INTEGER RUNDTE(2)	Computer run date of the PIDAL case	
IPRIO	INTEGER	Optional PIDAL output report flag	
RODSIN	INTEGER	Hand input control rod position flag	
IXCORE	INTEGER	Hand input excore detector flag	
IOUT	INTEGER	Statistical analysis routine start flag	
IXPOW	INTEGER	XTG detector powers flag	
IED	INTEGER	Optional edit input control flag	
REFROT	INTEGER	Quarter core symmetry control flag	
IEXPO	INTEGER	Not used by IBM version	
XCORE	REAL XCORE(4,2)	Hand input excore detector readings	
XCPR	REAL	Hand input excore power ratio reading	
ASI	REAL ASI(4)	Hand input excore axial shape indices	
H	REAL H(45)	Hand input rod positions by drive index	
RODG	REAL RODG(7)	Hand input rod positions by group	
DT	REAL	Energy in MWth for PIDAL case	
SCALE	REAL	Boronmeter scaling factor	
SOLBOR	REAL	Hand input Boron concentration in ppm	
CALPOW	REAL	Hand input reactor thermal power level	2
PSIA	REAL	Hand input primary pressure in psia	2
O	REAL	Hand input primary flow in 1.0E6 lbm/hr	2
TIN	REAL	Hand input primary inlet temperature	2
N	INTEGER	Number of hand input incore detectors	
IRAD	INTEGER	Unused	
DEV	REAL	Percent deviation for radial ratio check	
UNUSER	REAL	Uncertainty factor for alarm limits	
PMIN	REAL	Minimum operable detector power in MWth	
SIGMA	REAL	Currently not used	
J	INTEGER	Hand input detector string number	
I	INTEGER	Hand input detector axial level number	
E	REAL	Hand input detector signal in mv	
TSTART	INTEGER TSTART(4)	Start date for PIDAL exposure case	3
TEND	INTEGER TEND(4)	End date for PIDAL exposure case	3
RUNRQT	INTEGER RUNRQT(2)	Run date to mate power case with exposure	
HISTAT	REAL	BOC exposure file flag	
TAPPUN	INTEGER	Alarm limit output flag	

Notes:

1--If this value not positive, then PIDAL uses the PIP equivalent
2--If these values zero then PIDAL uses the PIP equivalents
3--For a power distribution case TSTART and TEND are set equal to the datalogger dump date in subroutine GETEXP

Table #2.0.1 Variables read from PIDAL case card image input

Variable	Type	Description	
E	REAL E(45,5)	Incore detector mv signals	1
KSUBB	REAL KSUBB(45,5)	Incore detector background corrections	1
KSUBF	REAL KSUBF(45,5)	Incore detector flux scaling factors	1
KSUBS	REAL KSUBS(45,5)	Incore detector sensitivities	1
MWTSG1	REAL	Steam generator #1 power in MWth	
MWTSG2	REAL	Steam generator #2 power in MWth	
MWTRX	REAL	Reactor calorimetric power in MWth	
TFW1	REAL	Steam generator #1 feedwater temp, F	
TFW2	REAL	Steam generator #2 feedwater temp, F	
FWFLO1	REAL	Steam generator #1 feedwater flow	2
FWFLO2	REAL	Steam generator #2 feedwater flow	2
PSG1	REAL	Steam generator #1 pressure, psia	
PSG2	REAL	Steam generator #2 pressure, psia	
PRX	REAL	Pressurizer pressure, psia	
FLORX	REAL	Reactor flow	2
TAV1	REAL	Primary loop #1 average temperature, F	
TAV2	REAL	Primary loop #2 average temperature, F	
DTEMP1	REAL	Primary loop #1 delta temperature, F	
DTEMP2	REAL	Primary loop #2 delta temperature, F	
MWEG	REAL	Megawatts electric gross, MWe	
MWENET	REAL	Megawatts electric net, MWe	
NI009	REAL	Nuclear power channel 9, psia	
P1TURB	REAL	Turbine stage 1 pressure, psia	
BORON	REAL	Boronmeter Boron concentration, ppm	
STMFL1	REAL	Steam generator #1 steam flow, 1.0E6 lb/hr	
STMFL2	REAL	Steam generator #2 steam flow, 1.0E6 lb/hr	
VAC	REAL	Condenser vacuum, inches Hg.	
CHGFLO	REAL	Charging flow, gpm	
PERIOD	REAL	Reactor period	
SPARE	REAL	Unused	
RODPOS	REAL RODPOS(45)	Control rod positions in inches withdrawn	
TAPETM	INTEGER*2 TAPETM(5)	Date of Varian datalogger data dump	

Notes:

1--These values in PIP notation, 1=top of core

2--These values converted from ADC counts to flows within PIDAL

Table #2.0.2 Variables read from Varian Datalogger UNTRANS data file

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.1.1 SUBROUTINE ZERO

Omitted.

2.1.2 SUBROUTINE RBORON

The purpose of SUBROUTINE RBORON is to read in the variables that are used in the HFP ARO critical Boron concentration calculation described in Section 2.1.16. The values are to be supplied on FORTRAN unit=12. If the allocated data set is empty, then the flag NOBOR is set to 1 causing Subroutine CBORON to be bypassed. If the data is successfully input, then NOBOR is set to 0 and the HFP ARO critical Boron concentration calculation will be performed. Table 2.1.2 describes each variable read in on unit=12.

Variable	Type	Description
BORTND	REAL BORTND(12)	Panvalet version of Boron data file
NXEN	INTEGER	Number of Xenon vs Power data points
NPOW	INTEGER	Number of power defect vs power lines
NCON	INTEGER	Number of rod worth vs position points
NBOR	INTEGER	Number of reciprocal Boron worth points
XENDRO	REAL XENDRO(30)	Equilibrium Xenon worths
XENPOW	REAL XENPOW(30)	Percent power corresponding to XENDRO
POWEXP	REAL POWEXP(10)	Exposure points for power defect lines
POWDEF	REAL POWDEF(10)	100% power defects for POWEXP lines
CONDRO	REAL CONDRO(30)	Group 4 control rod worths
CONDIW	REAL CONDIW(30)	Group 4 control rod positions for CONDRO
REBRWH	REAL REBRWH(10)	Reciprocal Boron worth points
RBWEXP	REAL RBWEXP(10)	Cycle exposures for REBRWH Boron points

Table #2.1.2 Variables read in from Boron data file

2.1.3 SUBROUTINE RODBYG

The purpose of SUBROUTINE RODBYG is to unfold the hand input control rod positions entered by control rod group on card 5 to full core configuration using the conversion table documented in Ref #2 Attachment #2. The unfolded full core control positions are stored in array H. RODBYG is called only if RODSIN (card 2) is set to 1.

2.1.4 SUBROUTINE FLOWS

The purpose of SUBROUTINE FLOWS is to convert the PIP supplied reactor and feedwater flow ADC counts to flow rates with units of lbm/hr. The methodology is as described in Ref #3 pages 2-28 and 2-29. The equations can be written using PIDAL variables as:

$$\text{Reactor Flow, FLORX} = \text{SQRT}\left(\frac{\text{CFLORX}}{6023.} * \frac{\text{DRX200}}{1.E4}\right) * 1518. \text{ (units 1.0E6 lbm/hr)}$$

where:

CFLORX = FLORX - 1506.
reactor flow in counts

DRX200 = -9.60E-6 * CTAV12**2 - 0.1455 * CTAV12 + 9856.
primary coolant density

CTAV12 = 4.0 * (10.0 * TAV12 - 4900.) / 0.66412
average primary temperature in counts

TAV12 = (TAV1 + TAV2) / 2.0
average primary temperature in degrees F

$$\text{Feedwater Flow, FWFLO} = \text{GPMFW} * \left(1. + \frac{0.043}{130.} * (\text{TFW}-430.)\right) * \frac{60.0}{7.48} * \text{DFW}$$

where:

GPMFW = 14.242 * SQRT(CFWFLO / 6023.)
feedwater flow in gallons per minute

CFWFLO = FWFLO - 1506.
feedwater flow in raw counts

$$\text{DFW} = \frac{-2.629E-5 * \text{CTFW}**2 - 3.05E-3 * \text{CTFW} + 6437.}{100.}$$

feedwater density

CTFW = 8.0 * (TFW + 125.) / 0.66412
feedwater temperature in raw counts

TFW = feedwater temperature in degrees F

Note that the feedwater flow calculation is repeated for each of the steam generators. Note that the feedwater flow is calculated with units of 1.0E3 lbm/hr and that comments in some PIDAL versions incorrectly stated the units to be 1.0E5 lbm/hr.

2.1.5 SUBROUTINE RODFDL

The purpose of SUBROUTINE RODFDL is to convert the PIP supplied control rod positions from the plants control rod drive system indexing scheme to the indexing scheme used internally by PIDAL. The conversion table used is documented in Ref #2 Attachment #2.

2.1.6 SUBROUTINE DECIDE

The purpose of SUBROUTINE DECIDE is to determine whether hand input or PIP supplied reactor and PCS parameters are to be used for the PIDAL case.

If hand input values for CALPOW, PSIA, O or TIN are supplied on input card #6, then those values override the PIP supplied equivalents. If any of the above hand inputs are omitted, then the PIP values, corresponding to the omitted parameter, become the default.

Since the hand input values may be set to PIP values in DECIDE, they are first stored in variables ZS, EC, XP, ZO and ET which correspond to SOLBOR, CALPOW, PSIA, O and TIN respectively. These stored values are written on the exposure file in Subroutine PUTEXP for trending purposes. Note that the PCS boron concentration, SOLBOR, must always be hand input.

The PIP values which correspond to CALPOW, PSIA, O and TIN are variables EEC, EEP, EEO and EET. These values are also saved for trending purposes. In addition, EEC is equal to MWTRX, EEP equals PRX, EEO equals FLORX/10 and EET equals the average of the two loop inlet temperatures. The equivalent values were read in from the PIP data in the main program. See Table #2.0.2.

Both the hand input and PIP values for each of the reactor parameters are edited by Subroutine PAGE1A. The edit also indicates which of the input sources will be used by PIDAL for each parameter.

2.1.7 SUBROUTINE INVERT

The purpose of SUBROUTINE INVERT is to convert the axial incore detector signals from PIP notation to PIDAL notation. This is necessary because the PIP numbers detectors axially from top=1 to bottom=5 while PIDAL uses the reverse. Therefore, PIDAL inverts the incore detector signals, detector sensitivities, detector background correction factors and detector flux scaling factors after they are read in. The inversion affects PIDAL variables E, KSUBS, KSUBB and KSUBF.

After inversion, the contents of variable E (the detector millivolt signals) are copied into variable ERAW for future storage on the exposure file.

After the incore detector signals are stored in ERAW, PIDAL adjusts all of the detector signals so that their millivolt signals read as they would have with respect to the reference detector at beginning of life. The adjustment for detector depletion takes the following form:

$$E(j,i) = E(j,i) * \frac{KSUBB(j,i)}{KSUBS(j,i)} \ (mv)$$

depletion and background corrected detector signals

where:

KSUBB(j,i) = detector background sensitivity correction factor for string j and detector level i

KSUBS(j,i) = detector depletion sensitivity correction factor for string j and detector level i

2.1.8 SUBROUTINE DAT

If the computer run date for the PIDAL case is not supplied via variable RUNDTE on hand input card type #1 then the IBM Subroutine TIMER is used to provide the current date and time of the run. Subroutine TIMER is described in Ref #6 section 4.1.4.4 .

The Gregorian date is returned by TIMER and stored in PIDAL variable RUNDTE. RUNDTE is declared as:

```
INTEGER RUNDTE(2)
```

and the date of the computer run is stored using the notation

```
RUNDTE(1) YYMMDD
RUNDTE(2) HHMMSSTTT (TTT is tenths of a second)
```

RUNDTE is also converted to a Julian date which is stored in variables J1 (month), J2 (day), J3 (year), J4 (hours and minutes) and j5 (seconds).

2.1.9 SUBROUTINE GENTIT

The purpose of SUBROUTINE GENTIT is to generate the case title from supplied input data.

The average group 4 control rod position is calculated by finding the average of the rod positions for rods 7, 11, 35 and 39 (in PIDAL notation). The average rod position is used in the case title of a power distribution case and for determining a control rod defect for the HFP ARO critical Boron concentration calculation in Subroutine CBORON.

The case title contains information summarizing the plant operating conditions and computer run dates for each PIDAL run. The title formats for power distribution and exposure cases are slightly different.

For a power distribution case, the title consists of the variables: DTEPAL, IG4ROD, IPCTPW, RUNDTE and R. For an exposure case the variables are: BLOCK, DTEPAL, IDT, IPCTPW, RUNDTE AND R. With the exception of the variable R, all of the title variables are described in Table #2.0.1.

The variable R is a flag used to note when the user has hand input the computer run date of the PIDAL case. If the user hand inputs RUNDTE (or if RUNDTE is supplied via a program which generates input card #1) then the variable R is set to ' '. If RUNDTE is not user supplied then PIDAL generates the computer run date and the variable R is set to 'R'. In a multiple case run, R is typically set for the second through the last case because RUNDTE is only generated by the SAVE program, P411825S, for the first set of input cards. PIDAL sets R in Subroutine DATE.

The case title is constructed by writing the above variables and some unchanging data to the work file on FORTRAN unit=29. The data written is then read back into a single variable TITLE. It is the variable TITLE that is then written to the PIDAL output file (FORTRAN unit=6) at the beginning of each page.

Also included in the case title, only on the first page of the PIDAL output file is the hand input variable COMENT from card #1.

2.1.10 SUBROUTINE PAGE1A (EDIT #1)

The purpose of SUBROUTINE PAGE1A is to write the first half of EDIT #1 to output. Data written by PAGE1A consist of:

1) The Panvalet version numbers of the datasets used by PIDAL which document the source code and input data used for each case. The PIDAL source code, Vendor Wprime Library, Boron data file source code versions are written. If the Boron data file is empty, then it's Panvalet version is not output.

2) The hand input and PIP reactor operating conditions are edited. This edit indicates whether hand supplied or PIP primary values (as determined by Subroutine DECIDE) will be used.

2.1.11 SUBROUTINE PAGE1B (EDIT #1)

The purpose of SUBROUTINE PAGE1B is to write the second half of EDIT #1 to output. Data written by PAGE1B consist of:

1) The secondary plant parameters supplies by the PIP input. At times some of these indicators may be somewhat erroneous due to faulty instrumentation.

2) The date and time identifying the PIP data dump being used by the PIDAL case.

2.1.12 SUBROUTINE VERTAP (Edit #1)

The date and time of a PIP data dump being used by an PIDAL exposure case is checked to be certain that it is representative of the time period being covered by the case. The date of the PIP dump used is stored in variables NODY and TAPETM. The dates of the PIDAL exposure case period are in variables TSTART and TEND. PIDAL verifies that NODY and TAPETM are between TSTART and TEND. If the datalogger dump date is not between TSTART and TEND then a warning message is edited.

2.1.13 SUBROUTINE OPTION (Edit #2)

The purpose of SUBROUTINE OPTION is to report on the use of hand input detector signals and control rod positions.

If any hand overlaid incore detector signals were input to PIDAL, then the detector string and level number, along with the input signal provided, will be written out for each overlay. Up to 40 hand input detector signals may be supplied to PIDAL via card type's #7 and #8. The detector signals are input in PIP notation (1=top 5=bottom) and are reported in PIDAL notation (1=bottom 5=top) by Edit #2. For the cycle 8 version of the code, the allowable number of hand input detector signals was increased to 225. This was to aid in a study on the effects of failing large numbers of incore detectors.

If hand input control rod positions are used for a PIDAL case then a message stating their use is edited. The method of input, either by position or by group, is not identified.

2.1.14 SUBROUTINE ADJROD (Edit #2)

The purpose of SUBROUTINE ADJROD is to verify that inserted control rod groups can be properly modeled given the W' and peaking data supplied in the PIDAL W' library.

Central Rod Correction Factor Calculation

If the central control rod is the only group three rod inserted past the 10.0 cm dead band at the top of the active fuel height then PIDAL calculates a rod correction factor to account for center rod perturbation effects on power shape. The rod correction factor concept is outlined in Ref #4 on page III-22.

The central control rod correction factor is found to be the average of the rod correction factors supplied for octant locations 22, 23 and 26 which are farthest from the core center on the diagonal. The factors for these octant locations are supplied by the fuel vendor and are read in from the Library data file.

Withdrawl of Non-Group 4 Control Rods

Because of limitations to the recent versions of the fuel vendors W' library, only the group 4 and central control rods are modeled by PIDAL. All other control rods are automatically withdrawn. For each control rod that PIDAL withdraws, a warning message is written to the output file giving the original rod position and the new, fully withdrawn position of the rod. At this point in the main program, PIDAL also converts the rods positions from inches from core bottom to centimeters from core bottom. An adjustment for the actual axial position of the poison in the control rod is also made by using the following equation:

$$H(i) = (H(i) + 0.8) * 2.54 \text{ (result in centimeters)}$$

where:

H(i) = rod position of rod i (of 45 rods) using PIDAL rod indexing

0.8 = It is assumed that at zero inches withdrawn the tip of the poison is 0.8 inches up from the bottom of the active fuel

Prior to removal of any inserted non group 4 rod banks, the positions of these banks are stored in variable XROD for use by Subroutine PREXTG. See Section #2.2.5.

It should be noted that the positions of the four part-length control rods are negated by PIDAL for easy identification on the PIDAL rod map produced by Subroutine RODEXP.

2.1.15 SUBROUTINE GETEXP (Edit #2)

The purpose of SUBROUTINE GETEXP is to insure that the current PIDAL case collects proper restart data from the PIDAL exposure data file. GETEXP scans the exposure file in search of the correct mating or previous PIDAL cases output records. The search is based on the BLOCK number and plant operation dates supplied by the user for the current PIDAL case.

For a power distribution run, PIDAL will normally mate with the last exposure case run. It is possible for the user to overide this feature and mate a power distribution case with any previous exposure case via the hand input variable RUNRQT.

For an exposure case, PIDAL again will normally mate with the last exposure case run. PIDAL exposure cases are kept track of via their BLOCK numbers and operation dates. If an exposure case is rerun, i.e. BLOCK number duplicated, then only the most recent case with the duplicate BLOCK will be considered current.

When an exposure case search is initiated, PIDAL either searches for the last occurance of the current BLOCK number minus one, or for the last occurance of the current BLOCK number. If the current case is a re-run then the mate will have the same BLOCK number, otherwise the mates block number must equal the current BLOCK number minus one.

Once a suitable mate based on BLOCK number has been found, the operation date of the mate is checked to insure that the PIDAL cases mate based on real plant operation time. If this check passes, then the mate is good and control returns to the main program. If an error with the BLOCKS or dates on the exposure file is detected then the PIDAL run is halted and a warning message is produced.

Table #2.1.15 contains a list of variables returned to the main program when a mate for the current PIDAL case is found.

Variable	Type	Description
FUEXPB	REAL FUEXPB(204,25)	BOC 3-D exposure distribution
TSTART	INTEGER TSTART(4)	Exposure period starting date
TEND	INTEGER TEND(4)	Exposure period ending date
RUNDTE	INTEGER RUNDTE(2)	Computer run date of current case
MRUNOD	INTEGER MRUNOD(2)	Date of computer run used as exposure base
BLOCK	INTEGER BLOCK	Number of exposure case
DT	REAL DT	Effective full power days of exposure case
CALPOW	REAL CALPOW	Calorimetric power for current case
INPUTD	INTEGER INPUTD(2)	Computer run date of original case if this case was a rerun
F3D25	REAL F3D25(204,25)	Measured 3-D power distribution for case
X	REAL X(80)	Trend data for case
AO	REAL AO	Axial offset for case
ERAW	REAL ERAW(45,5)	Unadjusted detector signals for case
ETHOUR	REAL	Ellapsed time for exposure case, hours
PEFF	REAL	Energy for exposure case in MWhr
TOTEP	REAL	Core average exposure MWD/MTU
FUEXP	REAL FUEXP(204,25)	3-D fuel exposure distribution
RODEXP	REAL RODEXP(45,25)	3-D control rod exposure distribution
KSUBS	REAL KSUBS(45,5)	Detector sensitivities for case
OLDK	REAL OLDK(45,5)	Detector sensitivities for last case
LHGRA	REAL LHGRA(204,25)	All pins nodal linear heat rate
XTG3D	REAL XTG3D(51,12)	XTG 3-D normalized power distribution
EXPLIB	INTEGER	Exposure file FORTRAN unit number

Table #2.1.15 Variables accessed on the exposure file by Subroutines GETEXP and PUTEXP

2.1.16 SUBROUTINE CBORON (Edit #2)

The HFP ARO critical Boron calculation is optional and is done only when data can be input from the Boron data file on FORTRAN unit=12 in SUBROUTINE RBORON. (Section 2.1.2)

The equation used for the HFP ARO critical Boron calulation is:

FUPOBO = BORDIF - BORROD + SOLBOR (ppm for HFP ARO)

where:

BORDIF = (DIFREA + DELXEN) * DELBC
ppm Boron adjustment for power and Xenon defects

BORROD = CONWOR * DELBC
ppm Boron adjustment for inserted control rods

SOLBOR = Core measured ppm Boron concentration either from hand inputs or PIP data

The components of the BORDIF equation can be broken down as follows:

DIFREA = DEFECT * (1 - PERPOW/100)
power defect adjustment (% delta rho)

where:

DEFECT = Interpolation of POWDEF and POWEXP from Boron data file as a function of CUREXP

PERPOW = Reactor percent power for current case

DELXEN = XENCON(1) - XENCON(2)
Xenon defect adjustment (% delta rho)

where:

XENCON(1) = Xenon worth for core at full power.

XENCON(2) = Interpolated Xenon worth for core at current operating power.

These values interpolated from XENPOW and XENDRO Boron data file data as functions of reactor power PERPOW by using Function CONINT.

DELBC = Interpolated core reciprocal Boron worth as a function of exposure CUREXP. Interpolated from REBRWH and RBWEXP data of the Boron data file.

The components of the BORROD equation can be broken down as follows:

CONWOR = Interpolated worth of control rod data CONDIW and CONDRO from Boron data file using Function CONINT. Interpolated as a function of group 4 rod position G4ROD.

DELBC = Interpolated core reciprocal Boron worth as a function of exposure CUREXP. Interpolated from REBRWH and RBWEXP data of the Boron data file.

2.1.16 FUNCTION CONINT

The purpose of FUNCTION CONINT is to interpolate arrays of data for the predicted HFP ARO critical Boron concentration calculation. Specifically, CONINT linearly interpolates control rod and xenon worth data based on group four control rod positions and reactor thermal power levels respectively.

If an input rod position or reactor power level is before the first rod worth or Xenon worth data point then no interpolation is done and the value returned by CONINT is equal to the first data point.

If the input rod position or reactor power level is within the range of rod worth or Xenon worth data available then the following equation is used to perform the interpolation.

$$CONINT = \frac{(X(i) - D)*Y(i-1) + (D - X(i-1))*Y(i)}{X(i) - X(i-1)}$$

where:

D	=	input rod position or reactor power level
X(i-1)	=	rod position or power level less than D
X(i)	=	rod position or power level greater than D
Y(i-1)	=	rod worth or Xenon worth data point corresponding to X(i-1)
Y(i)	=	rod worth or Xenon worth data point corresponding to X(i)
i	=	index position in arrays of rod position or power level point closest to and greater than input point D.

If the input rod position or reactor power level is beyond the range of rod worth or Xenon worth data availiable then the following equations is used for the interpolation.

$$CONINT = \frac{(X(n) - D)*Y(n-1) + (D - X(n-1))*Y(n)}{X(n) - X(n-1)}$$

where:

n	=	index position in arrays of rod position or power level of the last good data point.

All other variables as described above

Note that if a negative rod worth is calculated, then the rod worth is set to be zero.

Variable	Type	Description
N	INTEGER	Number of good data points in arrays X(30) and Y(30)
X	REAL X(30)	Array of abcissa values to be interpolated
Y	REAL Y(30)	Array of ordinate values to be interpolated
D	REAL	X operand used to interpolate between array values for an unknown Y value
CONINT	REAL FUNCTION	Result of interpolation, contains the previously unknown Y value

Table #2.1.16 Variables used in FUNCTION CONINT

2.1.17 SUBROUTINE DETAXL (Edit #3)

The purpose of SUBROUTINE DETAXL is to produce an edit identifying the axial position of the incore detectors within the reactor core. The detector centers are located at 10, 30 ,50 ,70 and 90 percent of the active fuel height. The detector positions are given both in feet and centimeters.

The active fuel height is known to PIDAL via variable HA, which is set to 335.28 cm (132 inches) in the BLOCK DATA section of the code. Note that for recent cycles, the cold fuel height was actually 131.8 inches. It is not known whether axial fuel expansion was taken into account in determining the 132 inch height.

2.1.18 SUBROUTINE CORMAP (Edit #4)

The purpose of SUBROUTINE CORMAP is to produce a radial core map of the reactor core showing the locations of the incore neutron detector strings. This report shows each detector strings radial location in the full core configuration. The detector numbering is also given in Ref #2 Attachment 3.

2.1.19 SUBROUTINE ASSMAP (Edit #5)

The purpose of SUBROUTINE ASSMAP is to edit maps showing the assembly numbering schemes used by PIDAL in full core, one-quarter core and octant core configurations. An octant core map showing the current fuel type loading is also produced from the data supplied to variable FUTYPE in the BLOCK DATA section of the code.

2.1.20 SUBROUTINE RBANK (Edit #6)

The purpose of SUBROUTINE RBANK is to assign inserted control rods to PIDAL control rod bank regions and to verify that the control rod insertion pattern follows the sequence as designated in Section 3.10 of Ref #7.

PIDAL has the capability of accommodating the four control rod groups that make up the Palisades regulating group. This allows for up to five possible rod bank regions designated by the note at the end of Table #2.1.20.

If the Group 4 rods are fully inserted in the core then there will be no rod bank 10 region and BANK(1) will equal 1. If there is an unrodded axial region then BANK(1)=10 and BANK(2)=1.

In the current version of the PIDAL code, only rod bank regions 10 and 1 are supported as all non-group 4 rods are withdrawn from the core. If the vendor Library data file is extended to contain wprime data for all rod banks (and Subroutine ADJROD is modified) no changes need be made to SUBROUTINE RBANK in order to allow for coverage of all five rod bank regions.

Variable	Type	Description
S(1)	REAL S(5)	Average group 4 rod position
S(2)	REAL S(5)	Average group 3 rod position
S(3)	REAL S(5)	Average group 2 rod position
S(4)	REAL S(5)	Average group 1 rod position
S(5)	REAL S(5)	Core active fuel height, 335.28 cm
SNUM(1)	INTEGER SNUM(4)	Rod group number assignment, 4
SNUM(2)	INTEGER SNUM(4)	Rod group number assignment, 3
SNUM(3)	INTEGER SNUM(4)	Rod group number assignment, 2
SNUM(4)	INTEGER SNUM(4)	Rod group number assignment, 1
BANK	INTEGER BANK(5)	Rod bank region numbers, see note 1

Notes:

1—Rod bank region 10 reserved for an unrodded axial segment
Rod bank region 1 is the rodded region for group 4's only
Rod bank region 2 is the rodded region for groups 3+4
Rod bank region 3 is the rodded region for groups 2+3+4
Rod bank region 4 is the rodded region for groups 1+2+3+4

Table #2.1.20 Variables used in SUBROUTINE RBANK

2.2 Program Section #2

Program section #2 controls the subroutines that perform the XTG three dimensional theoretical power distribution calculation. Theoretical coupling coefficients and boundary conditions are derived from the XTG solution.

The XTG program is supplied by Advanced Nuclear Fuels Corporation (ANF), formerly Exxon Nuclear. For all versions of PIDAL, the latest XTG version UJUN88 was incorporated. Sections 2.2.1 through 2.2.7 discuss how the new installation of XTG interfaces with PIDAL.

2.2.1 SUBROUTINE EXPAND

The purpose of Subroutine EXPAND is to collapse the three-dimensional XTG power distribution from 51 bundles by 4 nodes/bundle by 12 axial nodes to 51 bundles by 1 node/bundle by 12 axial nodes. Theoretical detector powers are also calculated.

Collapse XTG Power Distribution

The power distribution returned from XTG is a three-dimensional quarter core 51 bundle by 4 nodes/bundle by 12 axial nodes distribution. The XTG power distribution is stored in variable P. The XTG power distribution is collapsed by computing the average normalized powers for each of the 51 assemblies 12 axial nodes and storing the results in variable XTG3D. Note that the XTG model assumes that axial node 1 is at the top of core and PIDAL assumes the opposite. Therefore, the XTG power distribution is flipped during the collapse process and variables XTG3D and XTGAX assume axial node 1 to be at the bottom of the core.

Calculate Radial and Axial Power Distributions

The radial theoretical power distribution is calculated by summing the axial nodal powers in each assembly and then normalizing the resulting 51 integrated assembly powers by the core average assembly power. The radial power distribution is stored in variable XTGS and is calculated as follows.

```
XTGS(i) = XTGS(i) + XTG3D(i,j)   i=1 to 51, j=1 to 12
                                 integrated assembly powers
SUM = SUM + XTG3D(i,j)*4.        i=1 to 51, j=1 to 12
                                 total core power
SUM = SUM/204.                   core average assembly power

XTGS(i) = XTGS(i)/SUM            normalized theoretical assembly
                                 powers
```

The axial theoretical power distribution is calculated by summing the radial nodal powers in each of the 12 axial levels and then normalizing by the core average level power. The axial power distribution is stored in variable XTGAX and is calculated as described below.

```
XTGAX(i) = XTGAX(i) + XTG3D(i,j)*4.  i=1 to 51, j=1 to 12
                                     total power for level j
SUM = SUM*17.                    average power per axial level,
                                 SUM carried from radial calculation
XTGAX(i) = XTGAX(i)/SUM          normalized theoretical axial powers
```

For both the radial and axial calculations, the multiplier 4 is used because 4 is the number of times a given assembly, i, appears symmetrically in the full core.

2.2.1 SUBROUTINE EXPAND (continued)

Calculation of Theoretical Detector Powers

A theoretical axial power distribution consisting of 51 assemblies by 12 axial nodes was generated from XTG as described earlier in this section. From this power distribution it is desired to calculate the theoretical integrated power seen by each of the five axial incore detectors in all 51 assembly locations.

The theoretical detector powers will be found by utilizing polynomial functions, fitted from the known XTG axial shapes. These functions are of the same form for each assembly location and therefore this discussion will deal with the methodology used on a single assembly basis.

Because the computer has numerical limitations in raising numbers to high powers, the polynomial fitting function is limited to seven terms. Therefore, the core is divided into three sections with a separate polynomial function calculated for each section. Section 1 consists of XTG nodes 1 through 7 and covers detectors 1 and 2. Section 2 consists of XTG nodes 4 through 10 and covers detector 3. Section 3 consists of XTG nodes 12 through 6 and covers detectors 5 and 4. Refer to Figure #2.2.1 for a graphical description of the detector and node locations.

The polynomial equations used to model the axial power shapes are of the form:

$$\text{Power}(x) = A_1 + A_2x + A_3x^2 + A_4x^3 + A_5x^4 + A_6x^5 + A_7x^6$$

Where:

x = Axial height

A = Polynomial coefficients (unknown)

The known integral powers from XTG for each of the 12 axial nodes can be described by an equation of the form:

$$\text{Power}_i = \frac{\int_{x_{i-1}}^{x_i} (A_1 + A_2x + A_3x^2 + A_4x^3 + A_5x^4 + A_6x^5 + A_7x^6)dx}{X_i - X_{i-1}}$$

Where:

i = XTG axial node number

X = Axial height of XTG node i

A = Polynomial coefficients (unknown)

Power_i = Integrated power over XTG node i (known)

SECTION 2

DESCRIPTION OF THE MAIN PROGRAM AND SUBROUTINES

2.2.1 SUBROUTINE EXPAND (CONTINUED)

CALCULATION OF THEORETICAL DETECTOR POWERS

This equation is integrated to yield:

$$\text{Power}_i = \left. \frac{A_i x + A_2 \frac{x^2}{2} + A_3 \frac{x^3}{3} + A_4 \frac{x^4}{4} + A_5 \frac{x^5}{5} + A_6 \frac{x^6}{6} - A_7 \frac{x^7}{7}}{X_i - X_{i-1}} \right|_{X_{i-1}}^{X_i}$$

Which can be reduced to:

$$\text{Power}_i = [A_1(x_i - x_{i-1}) + \frac{A_2}{2}(x_i^2 - x_{i-1}^2) + \frac{A_3}{3}(x_i^3 - x_{i-1}^3) + \frac{A_4}{4}(x_i^4 - x_{i-1}^4) +$$

$$\frac{A_5}{5}(x_i^5 - x_{i-1}^5) + \frac{A_6}{6}(x_i^6 - x_{i-1}^6) + \frac{A_7}{7}(x_i^7 - x_{i-1}^7)]/(X_i - X_{i-1})$$

Since seven axial nodes will be treated at a time there are seven equations for Power, whose resulting values are known from XTG. There are seven values of A which are unknown. This system of seven equations and seven unknowns will be solved by subroutine DGELG, which is described in detail in Reference 9. In matrix notation, the problem can be described as:

$$\vec{P} = \bar{\bar{X}}\,\vec{A}$$

Where

$$\vec{P} = \begin{bmatrix} \text{Power}_1 \\ \text{Power}_2 \\ \text{Power}_3 \\ \text{Power}_4 \\ \text{Power}_5 \\ \text{Power}_6 \\ \text{Power}_7 \end{bmatrix} \qquad \vec{A} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \\ A_7 \end{bmatrix}$$

$$\bar{\bar{X}} = \begin{bmatrix} x_1 - x_0 & \frac{x_1^2 - x_0^2}{2} & \frac{x_1^3 - x_0^3}{3} & \cdots & \frac{x_1^7 - x_0^7}{7} \\ \hline x_1 - x_0 & x_1 - x_0 & x_1 - x_0 & & x_1 - x_0 \\ x_2 - x_1 & \frac{x_2^2 - x_1^2}{2} & \frac{x_2^3 - x_1^3}{3} & \cdots & \frac{x_2^7 - x_1^7}{7} \\ \hline x_2 - x_1 & x_2 - x_1 & x_2 - x_1 & & x_2 - x_1 \\ \vdots & \vdots & \vdots & & \vdots \\ x_7 - x_6 & \frac{x_7^2 - x_6^2}{2} & \frac{x_7^3 - x_6^3}{3} & \cdots & \frac{x_7^7 - x_6^7}{7} \\ \hline x_7 - x_6 & x_7 - x_6 & x_7 - x_6 & & x_7 - x_6 \end{bmatrix}$$

2.2.1 SUBROUTINE EXPAND (continued)

Calculation of Theoretical Detector Powers

The values returned from DGELG are the coefficients $\vec{A}$. With these values known, the detector powers are determined by utilizing the following equation.

$$\text{Power}_j = [A_1 (\text{Top}_j - \text{Bottom}_j) + \frac{A_2}{2} (\text{Top}_j^2 - \text{Bottom}_j^2) +$$

$$\frac{A_3}{3} (\text{Top}_j^3 - \text{Bottom}_j^3) + \frac{A_4}{4} (\text{Top}_j^4 - \text{Bottom}_j^4) +$$

$$\frac{A_5}{5} (\text{Top}_j^5 - \text{Bottom}_j^5) + \frac{A_6}{6} (\text{Top}_j^6 - \text{Bottom}_j^6) +$$

$$\frac{A_7}{7} (\text{Top}_j^7 - \text{Bottom}_j^7)]/(\text{Top}_j - \text{Bottom}_j)$$

From this equation, the power in any arbitrary interval j, bounded by bottom_j and top_j, can be determined.

As stated previously, the theoretical detector power solution method chosen requires that the core be split up into three separate axial region calculations. The first calculation performed is for the bottom core region covering detectors 1 and 2. The solution method, as implemented in PIDAL is as follows.

PIDAL first generates the Power and X matrices as described above and stores them in variables C and A respectively. The Power matrix C contains the XTG normalized power distribution for the bottom seven nodes of the core and all 51 assemblies. The matrix X contains the dependent variable terms that resulted from the integration. These terms are functions of core height and are calculated using:

$$A(i,j) = \frac{X(i+1)^{**}j - X(i)^{**}j}{j*(X(i+1) - X(i))} \qquad \begin{array}{l} i=1 \text{ to } 51 \text{ assemblies} \\ j=1 \text{ to } 7 \text{ function terms} \end{array}$$

Since only seven axial nodes are processed at a time, a relative axial indexing scheme was adopted in order to make the curve fit exact. This scheme is detailed in Figure #2.2.1 and is implemented via the values in X.

2.2.1 SUBROUTINE EXPAND (continued)

Calculation of Theoretical Detector Powers

Subroutine DGELG solves for the solution matrix A which is returned in variable C. The detector power solutions for the bottom two detectors are then obtained by implementing the equations:

$$PDQS(k,1) = PDQS(k,1) + C(j,k)* \frac{TOP(1)^{**}j - BOT(1)^{**}j}{j*(TOP(1) - BOT(1))} \quad \text{detector 1 power}$$

$$PDQS(k,2) = PDQS(k,2) + C(j,k)* \frac{TOP(2)^{**}j - BOT(2)^{**}j}{j*(TOP(2) - BOT(2))} \quad \text{detector 2 power}$$

where:

C(j,k) = coefficient matrix 7 terms for each of 51 assemblies
TOP(1) = top of detector 1 using relative indexing positions described by variable X.
BOT(1) = bottom of detector 1 using relative indexing positions described by variable X.
TOP(2) = top of detector 2 using relative indexing positions described by variable X.
BOT(2) = bottom of detector 2 using relative indexing positions described by variable X.
j = function term loop control, 1 to 7
k = assembly loop control, 1 to 51

The matrix, DGELG and PDQS calculations are then repeated twice, once for the axial region containing detector 3 and then for the axial region containing detectors 4 and 5. After the theorectical detector powers for all five detector levels are calculated, they are stored in variable TDET for later use by PIDAL.

Theoretical Detector Power Normalization

The theoretical detector powers are normalized by calculating an average detector power for each of the five axial detector regions and then dividing each detector power by it's respective level average. The equations for the level averages and normalizations are:

$$AVG(i) = AVG(i) + PDQS(j,i)*SYM(j)/204.$$

where:

AVG(i) = average detector power for axial level i
PDQS(j,i) = theoretical detector power for octant j and axial level i
SYM(j) = number of symmetric locations for octant j in the full core

$$PDQS(j,i) = PDQS(j,i)/AVG(i)$$ result is level normalized

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

After the theoretical detector powers have been calculated, they are edited by incore detector level as EDIT #7. The actual generation of the radial detector power maps is performed by SUBROUTINE QTRMAP. See section 2.4.2.

Variable	Type	Description
P	REAL P(2500)	XTG 1/4 Core 3-D Power Distribution 51 bundles by 4 radial nodes/bundle by 12 axial nodes/bundle
XTGS	REAL XTGS(51)	Normalized Radial XTG power distribution
XTGAX	REAL XTGAX(12)	Normalized Axial XTG power distribution
XTG3D	REAL XTG3D(51,12)	Normalized 3-D XTG power distribution XTG3D(x,1) = bottom of core
C	REAL C(7,51)	Temporary storage
A	REAL A(7,7)	Temporary storage
TOP	REAL TOP(3)	Detector tops using relative indexing
BOT	REAL BOT(3)	Detector bottoms using relative indexing
TDET	REAL TDET(51,5)	Theoretical detector power distribution
PDQS	REAL PDQS(51,5)	Theoretical detector power distribution normalized by axial detector level
AVG	REAL AVG(5)	Axial detector level normalization factors

Table #2.2.1 Variables used in SUBROUTINE EXPAND

2.2.2 SUBROUTINE EXTRAP (Edit #8)

The purpose of Subroutine EXTRAP is to calculate theoretical axial boundary conditions for each assembly in the core. The boundary condition is defined as the real core height as a fraction of the apparent core height. These boundary conditions are used later in PIDAL by the assembly power axial curve fitting routine in Subroutine ZIADA2. It is assumed that the reader is familiar with the theory behind ZIADA2 and is directed to Sections #2.5 and #2.5.1 before reading further in this section.

Subroutine EXTRAP is the controlling subroutine for the boundary condition calculations, as the bulk of the work is done via the coding in Subroutine ERROR which is called by EXTRAP.

The method used in EXTRAP to determine the best boundary condition for a given assembly is a simple iterative process. Several boundary condition values near the top and bottom of the core are assumed. A linear least squares computation is used to fit the known XTG assembly axial power distribution with a Fourier series. There is some squared sum error associated with the computed fit. The process iterates until the boundary condition which results in the best fit (minimizes the square sum error) is found.

The initial boundary condition for an assembly is assumed to be 1.0 (which means that the apparent core height equals the active fuel height) and the square sum error for this guess is calculated by Subroutine ERROR. The error calculation is then repeated by decreasing the boundary condition by steps of .05 (increasing the apparent core height) until a minimum square sum error is found.

After a minimum error is found using the large iteration steps, the boundary condition used for the minimum calculation is increased by 0.04 and the iteration process is repeated using smaller boundary condition decrements of 0.01 until the minimum square sum error is found.

The entire iteration process is repeated for each of the 51 assemblies, after which all of the calculated values are written to the PIDAL output file as EDIT #8. The boundary condition map is generated by Subroutine QTRMAP described by section 2.4.2.

It should be noted that if the boundary condition reaches a values of 0.50 and has not converged, then the iteration process is stopped and the boundary condition is assumed to be 0.50 by PIDAL. This is an unrealistic value and the user should assume something is wrong if a boundary condition of 0.50 is used for any assembly. No warnings are given if this condition arises.

This discussion has not dealt with the actual calculations used in determining the square sum error in a given boundary condition calculation. For a discussion on this subject refer to Section 2.2.3 on Subroutine ERROR.

Variable	Type	Description
POW	REAL POW(51,12)	XTG power distribution copied from array XTG3D.
IOCT	INTEGER IOCT	current assembly counter
NOCT	INTEGER NOCT	current assembly counter
TERR	REAL TERR	square sum error of the axial calculation in Subroutine APFS for last iteration
SSE	REAL SSE	square sum error returned from APFS for current iteration
TBC	REAL TBC	boundary condition used for the current iteration
BCINC	REAL BCINC	boundary condition increment to be used between iterations
PASS2	LOGICAL PASS2	Flag for determining whether to start using smaller iteration steps
BC	REAL BC(51)	Final calculated boundary conditions for all 51 assemblies.

Table #2.2.2 Variables used in SUBROUTINE EXTRAP

2.2.3 SUBROUTINE ERROR

Subroutine ERROR determines the square sum error associated with a linear least squares fit of a fifth order sine function to the XTG axial power shape for a given assembly and boundary condition. This sine function used is the same as developed within the discussion of Subroutine ZIADA2 in Section #2.5.1.

Subroutine ERROR performs three functions. First, ERROR determines the dependancy of the sine function on the current boundary condition for the assembly. This allows a sine series polynomial (consisting of the expanded form of equation #2.5.1f) for the assemblies current boundary condition to be represented by Function FFCT2. A set of normal equations for a linear least squares fit to the sine series polynomial are then set up by Subroutine APLL. Lastly, Subroutine APFS is used to solve the normal equations set up by APLL and the square sum error associated with the least squares fit is determined.

The dependancy of the sine series polynomial on the current boundary condition is given by:

$$Z(i) = TBC * PI * \left(\frac{DX}{132.} - 0.5\right) + HP$$

where:

TBC = boundary condition for the current iteration
PI = 3.141592654
HP = 1.570796327
DX = height in inches of axial node boundary, i
bottom of node 1 — i=1, DX=0.0
top of node 1 — i=2, DX=11.0
bottom of node 2 — i=2, DX=11.0
top of node 2 — i=3, DX=22.0
. .
. .
top of node 12— i=13, DX=132.0

A discussion on the use of Subroutines APLL and APFS follows. For a complete discussion on their use the reader is directed to Ref #9.

Subroutine APLL sets up a matrix of normal equations based on the five mode Fourier sine function described by Subroutine FFCT2. The matrix is then used by Subroutine APFS to determine the error in the current boundary condition iteration.

2.2.3 SUBROUTINE ERROR (continued)

Given the following equation, representing the power shape in an octant location as a function of core height:

$$P(x_i) = C_1 G_1(x_i) + C_2 G_2(x_i) + C_3 G_3(x_i) + C_4 G_4(x_i) + C_5 G_5(x_i)$$

where:

$P(x_i)$ = theoretical power as a function of core height (known)

C = Coefficients of linear combination (unknown)

$G(x_i)$ = five terms of sine series described by Function FFCT2 (known)

It is desired to calculate the set of coefficients C_1 through C_5 such that the square sum error, using a least squares fit of the sine series function is minimized. In matrix notation the problem is written as:

$$\bar{\bar{A}}\vec{C} = \vec{R} \quad \text{and}$$

$$\vec{C} = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \end{bmatrix}$$

$\bar{\bar{A}}$ is a 5 by 5 symmetric matrix with values only on the upper side of the diagonal. It's components are generated using the equation:

$$a_{jk} = \sum_{i=1}^{12} WGT(x_i)G_k(x_i)G_j(x_i) \qquad \begin{array}{l} j=1 \text{ to } 5 \text{ sine series terms} \\ k=1 \text{ to } j \end{array}$$

where:

$WGT(x_i)$ = Fraction of XTG node i which is monitored by a detector

$G(x_i)$ = five terms of FFCT2 sine series function

Pictorially the $\bar{\bar{A}}$ matrix appears as:

$$\begin{array}{lll} \sum_{i=1}^{12} WGT(x_i)G_1(x_i)G_1(x_i) & \sum_{i=1}^{12} WGT(x_i)G_1(x_i)G_2(x_i) \cdots & \sum_{i=1}^{12} WGT(x_i)G_1(x_i)G_5(x_i) \\ & \sum_{i=1}^{12} WGT(x_i)G_2(x_i)G_2(x_i) \cdots & \sum_{i=1}^{12} WGT(x_i)G_2(x_i)G_5(x_i) \\ & & \vdots \\ & & \sum_{i=1}^{12} WGT(x_i)G_5(x_i)G_5(x_i) \end{array}$$

2.2.3 SUBROUTINE ERROR

$\vec{R}$ is a 6 element vector which is generated using the equations:

$$r_j = \sum_{i=1}^{12} WGT(x_i)POW(x_i)G_j(x_i) \qquad j=1 \text{ to } 5 \text{ sine series terms}$$

$$r_6 = \sum_{i=1}^{12} POW(x_i)POW(x_i)$$

where:

$POW(x_i)$ = Power in node i of current assembly

Pictorially the $\vec{R}$ vector appears as:

$$\vec{R} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{bmatrix}$$

The $\underline{A}$ matrix and $\vec{R}$ vector are then combined to form a 6 by 6 matrix with the right-hand-column consisting of the $\vec{R}$ vector. The resulting matrix is then stored columnwise in the one-dimensional array WORK and returned to ERROR by APLL.

Subroutine APFS then solves the WORK matrix for the unknown vector $\vec{C}$ and determines the square sum error by utilizing a least-squares fit of the resulting linear combination. The square sum error is returned to ERROR from APFS in the last element of the WORK array. ERROR returns the square sum error to EXTRAP via the variable SSE.

Variable	Type	Description
Z	REAL Z(13)	argument of the cosine function used in Subroutine FFCT2
TBC	REAL TBC	boundary condition used for the current iteration
FFCT2	EXTERNAL FFCT2	subroutine that contains the integral of the 5th order sine series function
P	REAL P(6)	array containing results of integration of Subroutine FFCT2
WORK	REAL WORK(21)	array returned by APLL containing coefficient matrix for least squares fitting routine APFS
SSE	REAL SSE	square sum error returned from APFS for current iteration

Table #2.2.3 Variables used in SUBROUTINE ERROR

2.2.4 SUBROUTINE FFCT2

Subroutine FFCT2 contains the integrated polynomial sine series function described by equation #2.5.1f. FFCT2 is used internally by Subroutine APLL to define a set of normal equations for a least squares fit of the given polynomial to the XTG data. The arguments of the sine series, which are stored in variable Z, were calculated by Subroutine ERROR.

2.2.5 SUBROUTINE PREXTG

The purpose of Subroutine PREXTG is to generate card image input for XTG to use during run execution. The cards generated control the XTG restart function and define the core operating conditions that the current PIDAL case assumes. The cards generated by PIDAL are the XTG 2-4, 3-1, 3-2, 10-0, 21-0 and -1 cards. For a complete description of the data required on these cards consult Reference #12.

2.2.6 SUBROUTINE AFTXTG

Subroutine AFTXTG is a XTG post-processor which extracts data needed by PIDAL from the XTG storage vector. The data is merely copied from the XTG storage vector and placed into local PIDAL variables. The data extracted include the current case XTG power distribution, xenon, iodine, exposure, moderator density and flux distributions. AFTXTG is essentially a copy routine so no further description is warranted.

2.2.7 SUBROUTINE CHKXTG

The purpose of Subroutine CHKXTG is determine if PIDAL needs to activate XTG to predict the core power distribution. This routine was actually written for online use when PIDAL would be running continuously. Almost without exception, XTG will run with every PIDAL case run with the IBM version. CHKXTG will run XTG if any of the following coditions are met:

1) The current PIDAL run is a single power distribution case.

2) The current PIDAL case is an exposure case, either by itself (single), or part of a depletion run.

3) If reactor power, boron concentration or primary coolant pressure have varied by greater than one percent from the previous case in a multiple case PIDAL run. A rod position check will also be added.

If CHKXTG determines that XTG should run, then a message following Edit #6 is printed stating the reason for running XTG. CHKXTG then calls Subroutine PREXTG before passing control to XTG itself. Upon successful completion of XTG, CHKXTG resumes control and calls AFTXTG, EXPAND and EXTRAP in turn.

2.3 Program Section #3

Program section #3 controls six subroutines that convert the incore detector millivolt signals to detector powers. The conversion is done by utilizing wprimes (W') which are detector signal to power coupling coefficients. Corrections to the detector signals are also made which take into account inserted control rod perturbation effects.

2.3.1 SUBROUTINE OCTEXP (Edit #42)

The purpose of Subroutine OCTEXP is to collapse the full core PIDAL exposure distibution from 204 assemblies by 25 axial nodes to an octant distribution of 28 assemblies by 25 axial nodes.

OCTEXP collapses both the BOC and current exposure distributions in arrays FUEXP and FUEXPB and stores the result in EXPO and EXPOB. The arrays EXPO and EXPOB are then used by Subroutine WCALC for interpolating the W' library.

Alternately, an edit of EXPO and EXPOB can be made (Edit #42) by setting IFLAG=1. IFLAG is a parameter passed into OCTEXP by the subroutine call. PIDAL normally produces this edit after calling Subroutine EXPOZF (section 2.7.1) from the main program.

2.3.2 SUBROUTINE WCALC

The purpose of Subroutine WCALC is to interpolate between the detector signal to power coupling coefficients or wprimes (W') supplied in the fuel vendors library data file. An interpolation of the supplied data, based on the current exposure of the fuel surrounding a detector, is necessary because the library is only complete for a few exposure points throughout a given cycle.

Calculation of Detector Exposures

The total accumulated exposure in the region of each detector is found by taking the average of the exposures in the three axial fuel nodes closest to each detector. These exposures are calculated from the 3-D exposure distribution variable EXPO and are stored in variable BU. The following table shows the relationship between the PIDAL axial fuel nodes and five detector levels. This data is represented graphically in Figure #2.2.1.

Detector Level	PIDAL nodes covered
1 bottom of core	2,3,4
2	7,8,9
3	12,13,14
4	17,18,19
5 top of core	22,23,24

The following equation is used to calculate the exposure in the region of detector 1 for any octant location, i. This equation is applied in a similar fashion to the remainder of the detector levels. Note that these exposures are total accumulated assembly exposures since BOL for the given region.

$$BU(i,1) = \frac{EXPO(i,2) + EXPO(i,3) + EXPO(i,4)}{3}$$

Interpolation of W' Values

The W' values are read in from the fuel vendor library data file by octant location. There is a set of up to 25 W' points for each octant location. After each read, the W' and exposure data in variables Y and X respectively are interpolated based on the bundle exposures BU at each of the five axial detector levels. In this way, W' values are generated for all 28 octant locations by five axial levels. The interpolation is performed by Function XPINT which is detailed in Section 2.3.3.

2.3.2 SUBROUTINE WCALC (continued)

Expand W' Values to 45 Detector Strings

Up to this point, the W' values calculated have been stored by octant location and detector level. For use later in PIDAL it is necessary to expand the W' data to all 45 detector strings. This is done by associating each of the detector strings with their octant location via the array ID and copying the WPRIME values to their respective string locations in variable W.

During this step, the WPRIME values are adjusted by the reference detector sensitivity and a detector power to calorimetric power scaling factor is applied. The following equation is used for the expansion and adjustment:

$$W(K,N) = WPRIME(IDD,N) \frac{CONST}{SCAL}$$

where:

W(K,N) = reference sensitivity and power corrected W' for detector string K and axial level N

WPRIME(IDD,N) = interpolated W' from library for octant IDD and axial level N (MWth/(neutrons/sec))

CONST = constant for scaling detector powers with calorimetric power (neutrons/nv-sec-cm-milliohm) see Ref #10

SCAL = detector reference sensitivity per unit length (amp/nv-cm)

The variables CONST and SCAL are declared in the main program block data section and are derived from the W' discussion on page III-3 of Ref #4. The equations used to calculate their values are given as:

$$CONST = \frac{CALIB}{L * R * 1.0E+3}$$

where:

CALIB = Rhodium activation due to unit Maxwellian flux impinging on the detector sheath, per Rh atom, neutrons/nv-sec

L = incore detector length, cm

R = impedence of load resistor, ohms

1.0E+3 = 1000 milli-ohms/ohm

$$SCAL = \frac{Ks,ref}{L}$$

where:

Ks,ref = detector batch reference sensitivity at BOC, amp/nv

L = incore detector length, cm

2.3.2 SUBROUTINE WCALC

Calculate remaining library exposure

The final calculation in Subroutine WCALC is of the remaining exposure for each octant level of the W' data in the library data file. The equation used is:

$$DELEXP(i,l) = STOPER(i) - BU(i,l)$$

where:

- DELEXP(i,l) = amount of W' data remaining in MWD/MTU for a detector in octant, i, at axial level, l
- STOPER(i) = last W' exposure point in library for octant location, i
- BU(i,l) = current PIDAL exposure for octant, i, axial level, l

The remaining library exposures are used later in PIDAL for determining the limiting number of effective full power days remaining before a new W' library must be installed.

Variable	Type	Description
NPTS	INTEGER NPTS	Number of exposure points for rod bank 10 in library data file. NPTS <= 25
STOPER	REAL STOPER(28)	Maximum exposure in library data file for each octant location
X	REAL X(28,25)	Exposure points for each octant location in W' library. Maximum of 25 for each
IWS	INTEGER IWS	Index number of first W' data point in library that is considered accurate
IWF	INTEGER IWF	Index number of W' data point used to extrapolate back if current exposure is before point IWS in library
FUEXP	REAL FUEXP(28,25)	3-D octant exposure distribution based on the mating exposure case found by GETEXP
BU	REAL BU(28,5)	3-D exposure distribution collapsed to five axial detector regions
WPRIME	REAL WPRIME(28,5)	W' values interpolated from the fuel vendor library based on exposure BU
ID	INTEGER ID(45)	octant locations indexed by the incore detector string number
CONST	REAL CONST	constant for scaling detector powers with reactor calorimetric power
SCAL	REAL SCAL	detector reference sensitivity per unit length
DELEXP	REAL DELEXP(28,5)	remaining exposure in the W' library for each octant location and axial level

Table #2.3.2 Variables used in SUBROUTINE WCALC

2.3.3 FUNCTION XPINT

Function XPINT is used by Subroutine WCALC to interpolate W' data that is read in from the library data file. A linear interpolation is used to determine the correct W' value.

If the exposure, BU, of the current octant location, I, is less than the first good data point, X(I,IWS), then a linear extrapolation back to the desired assembly exposure using the equation:

$$XPINT = \frac{(X(I,IWF) - BU)*Y(IWS) + (BU - X(I,IWS))*Y(IWF)}{X(I,IWF) - X(I,IWS)}$$

If the exposure, BU, of the current octant location, I, is within the range of good exposure data then the linear interpolation uses the equation:

$$XPINT = \frac{(X(I,j) - BU)*Y(j-1) + (BU - X(I,j-1))*Y(j)}{X(I,j) - X(I,j-1)}$$

where:

j = data point subscript in arrays X and Y corresponding to the first exposure value greater than or equal to exposure BU for octant location I

If the exposure, BU, of the current octant location, I, is greater than the last data point X(I,NPTS) then a linear extrapolation beyond the data is done using the equation:

$$XPINT = \frac{(X(I,NPTS) - BU)*Y(NPTS-1) + (BU - X(I,NPTS-1))*Y(NPTS)}{X(I,NPTS) - X(I,NPTS-1)}$$

For all of the above methods of interpolation, the result is returned to WCALC via variable XPINT. The variables for the above equations are defined in Table #2.3.3.

Variable	Type	Description
I	INTEGER I	octant location number of detector whose W' value is being determined
BU	REAL BU	PIDAL exposure of octant location I
NPTS	INTEGER NPTS	number of exposure data points in library for each octant location. NPTS <= 25
X	REAL X(28,25)	Exposure points for each octant location in W' library. Maximum of 25 for each
IWS	INTEGER IWS	Index number of first W' data point in library that is considered accurate
IWF	INTEGER IWF	Index number of W' data point used to extrapolate back if current exposure is before point IWS in library
Y	REAL Y(28)	W' data for octant location I. There are NPTS good data points. The subscript 28 should be a 25.

Table #2.3.3 Variables used in FUNCTION XPINT

2.3.4 SUBROUTINE COFCAL (Edit #9)

The purpose of Subroutine COFCAL is to retrieve and interpolate peaking factor and coupling coefficient data from the Library data file. Peaking factors calculated here include all pins, inner pins, narrow water gap pins and burnup peaking factors. The coupling coefficients calculated here are no longer used by PIDAL and are determined only for reference.

In an attempt to more accurately represent the peaking factor and coupling coefficient data, separate calculations are performed for each of the axial rod bank regions within a given octant location. This is done by determining average exposures for each axial rod bank region within an octant location, and then interpolating the peaking factor and coupling coefficient data based on these exposures.

Convert Library Data File Exposures

The Library data file exposures (in variable X) are in MWD/MTU from BOL. For use in COFCAL it is necessary to convert these exposure values to MWD/MTU since BOC. This is done by subtracting the nodal octant location exposures, at the start of the current cycle, from the current respective octant location nodal exposures. The resulting cycle accumulated exposures are replaced into variable X.

Determine the Nodal Boundaries of Each Rod Band Region

In order to accurately calculate the accumulated exposures in each of the rod bank regions, it is necessary to determine the exact nodal boundaries for each region. Variables IZ1S and IZ2S contain the bottom and top axial nodes numbers, respectively, that are either partially or completely within each rod bank region. The variables RBS and RTS contain the fraction of the bottom and top nodes that are within each rod bank region.

Calculate Octant Exposures for each Rod Bank Region

After the nodal boundaries for each rod bank region have been determined, it is possible to calculate the exposures for each octant location of each rod bank region. This is done by summing up the nodal exposures within each octant location and rod bank region, including fractional nodes, and then dividing by the total number the sums by the number of nodes in the rod bank The following equations are used.

For the bottom fuel node in the rod bank region:

```
BEXPI(i,L) = (EXPO(i,IZ1) - EXPOB(i,IZ1)) * (1.-RB)
```

where:

- BEXPI(i,L) = fractional exposure in octant location i, rod bank L, covered by bottom node IZ1
- RB = fraction of node IZ1 contained in octant location i of rod bank L
- EXPO(i,IZ1) = current exposure since BOL for octant location i and node IZ1
- EXPOB(i,IZ1) = current exposure since BOC for octant location i and node IZ1

2.3.4 SUBROUTINE COFCAL (continued)

Calculate Octant Exposures for each Rod Bank Region

For the interior axial (non-fractional) nodes in a rod bank region:

$$BEXPI(i,L) = BEXPI(i,L) + EXPO(i,j) - EXPOB(i,j) \qquad j=IZ1+1 \text{ to } IZ2$$

where: BEXPI(i,L) = total exposure from bottom node IZ1 up to last node, IZ2, completly in the region L

For the top fuel node, the following equation is used only if the top node is partially in the rod bank region:

$$BEXPI(i,L) = BEXPI(i,L) + (EXPO(i,IZ2+1) - EXPOB(i,IZ2+1)) * RT$$

where: BEXPI(i,L) = total exposure in all nodes of rod bank L and octant i
RT = fraction of the top node IZ2+1 in region L

The average nodal exposure in rod bank region, L, and octant location i is then found using:

$$BEXPI(i,L) = BEXPI(i,L)/(IZ2 - IZ1 + 1 - RB + RT)$$

It should be noted that the use of variable IZ1 and IZ2 is somewhat tricky in the actual coding and the reader is referred to the source code for the details. What is stored in varables IZ1S and IZ2S are the values IZ1+1 and IZ2+1 respectively and this is also what appears on the PIDAL output reports.

Calculation of Peaking Factors and Coupling Coefficients

Once the average rod bank region exposures for each octant location are determined, the fuel vendor library data is interpolated for peaking factors and coupling coefficients. This data is interpolated as functions of octant location and rod bank region exposure. The interpolation is performed in Function XINT which is described in Section 2.3.5.

The following one-pin peaking factors are calculated:

F_ℓ^T = peak pin or all pins peaking factor, maximum ratio of the power in an individual fuel rod to assembly average power

$F_\ell^{\Delta h}$ = inner pin peaking factor, maximum ratio of the power in a fuel rod not on the assemblies periphery to assembly average power

F_ℓ^N = narrow water gap peaking factor, maximum ratio of power in a fuel rod adjacent to the narrow interfuel assembly water gap to assembly average power

F_ℓ^B = burnup peaking factor, maximum ratio of exposure in any one pin in an assembly to assembly average exposure

2.3.4 SUBROUTINE COFCAL (continued)

Output of Peaking Factors and Coupling Coefficients

The peaking factors and coupling coefficients calculated from the fuel vendors library are edited to the PIDAL output file by rod bank region. Along with these values, the axial nodal point limits of each rod bank region are written, as well as the assembly exposures since BOC.

Again note that the coupling coefficients calculated from the fuel vendors library are not used in PIDAL for any further calculations.

Variable	Type	Description
X	REAL(28,25)	exposure data points from library data file converted from MWD since BOL to MWD since BOC
IBNK	LOGICAL IBNK(10)	flags used to determine if error occurred while reading library data file
FUEXP	REAL FUEXP(28,25)	3-D octant exposure distribution based on the mating exposure case found by GETEXP
FUEXPB	REAL FUEXPB(28,25)	3-D octant exposure distribution for core at BOC
BEXPI	REAL BEXPI(28,10)	average rod bank exposure for 28 octants and up to 5 rod bank regions
IZ1S	INTEGER IZ1S(10)	bottom node of rod bank region for up to 5 rod bank regions
IZ2S	INTEGER IZ2S(10)	top node of rod bank region for up to 5 rod bank regions
RBS	REAL RBS(10)	fraction of bottom node within each rod bank region for up to 5 rod bank regions
RTS	REAL RTS(10)	fraction of top node within each rod bank region for up to 5 rod bank regions
R	REAL R(28,10)	all pins peaking factor interpolated from library data file for each octant and rod bank region, F
T	REAL T(28,10)	inner pins peaking factor interpolated from library data file, F
TN	REAL TN(28,10)	narrow water gap pins peaking factor interpolated from library data file, F
PINEXP	REAL PINEXP(28,10)	burnup peaking factor interpolated from library data file, F
DOWN	REAL DOWN(28,10)	fuel vendor coupling coefficient for bottom neighbor assembly
RIGHT	REAL RIGHT(28,10)	fuel vendor coupling coefficient for right neighbor assembly
TOP	REAL TOP(28,10)	fuel vendor coupling coefficient for top neighbor assembly
LEFT	REAL LEFT(28,10)	fuel vendor coupling coefficient for left neighbor assembly

Table #2.3.4 Variables used in SUBROUTINE COFCAL

2.3.5 FUNCTION XINT

Function XINT is used by Subroutine COFCAL to interpolate peaking factors and coupling coefficients read in from the library data file. The method of calculation is a linear interpolation based on exposure and octant location.

If the exposure, BU, of the current octant location, I, is less than the first good data point, X(I,1), then a linear extrapolation back to the desired assembly exposure using the equation:

$$XINT = \frac{(X(I,2) - BU)*Y(1) + (BU - X(I,1))*Y(2)}{X(I,2) - X(I,1)}$$

If the exposure, BU, of the current octant location, I, is within the range of good exposure data then the linear interpolation uses the equation:

$$XINT = \frac{(X(I,j) - BU)*Y(j-1) + (BU - X(I,j-1))*Y(j)}{X(I,j) - X(I,j-1)}$$

where:

j = data point subscript in arrays X and Y corresponding to the first exposure value greater than or equal to exposure BU for octant location I

If the exposure, BU, of the current octant location, I, is greater than the last data point X(I,NPTS) then a linear extrapolation beyond the data is done using the equation:

$$XINT = \frac{(X(I,NPTS) - BU)*Y(NPTS-1) + (BU - X(I,NPTS-1))*Y(NPTS)}{X(I,NPTS) - X(I,NPTS-1)}$$

For all of the above methods of interpolation, the result is returned to COFCAL via variable XINT. The variables for the above equations are defined in Table #2.3.5.

Variable	Type	Description
I	INTEGER I	octant location of peaking factor or coupling coefficient being interpolated
BU	REAL BU	cycle exposure of octant location I for region covered by the current rod bank
X	REAL X(28,25)	exposure data from library data file for octant location I in MWD since BOC
NPTS	INTEGER NPTS	number of good data points in arrays X Y. NPTS <= 25
Y	REAL Y(28)	peaking factor or coupling coefficient data to be interpolated. 28 should be 25

Table #2.3.5 Variables used in FUNCTION XINT

2.3.6 SUBROUTINE BXPWR

The purpose of Subroutine BXPWR is to calculate control rod correction factors for the detector signals. These factors are then applied, along with the W' values calculated in Subroutine WCALC, to the detector signals in order to obtain detector powers.

The concept of correcting the detector signals for control rod perturbations is outlined in Ref #4 page III-22. In short, two rod corrections are accounted for in PIDAL. One is for rod patterns with the nearest rod inserted with the other being for rod patterns with the non-nearest rod inserted. The correction factors for these patterns, assuming the rods to be fully shadowing the detectors, are supplied by the fuel vendor in the Library data file.

In the current versions of the Library data file, the coefficients for the near rod patterns are supplied and read in on variable AA. The values for the non-nearest rod patterns are assumed to be negligible and thus values of zero are supplied for variable BB.

PIDAL also performs a calculation in order to determine the fractional amount that a detector is shadowed by the nearest and non-nearest control rods. This allows for further correction to the data supplied in AA. (and BB if this data were supplied)

A table look-up is performed to find the index number of the nearest control rod to the current detector string via variable INDEX. There is no near rod for strings #1,13 and 45. If a near rod is present, then the fraction of the detector shadowed by the near rod is calculated for each detector in the string and a near rod correction coefficient is calculated using:

$$RODEFF = AA(k) * WP$$

where:

- $AA(k)$ = fully shadowed detector rod correction factor for detector string k from Library data file
- WP = fraction of current detector shadowed by nearest rod to the detector

This process is then repeated for the non-nearest rod to the detector. This results in two rod correction factors, one for near and one for non-near control rods. The detector powers in MWth are then calculated using:

$$P(k,n) = W(k,n)*(1.-AMAX1(BB(k)*WP,RODEFF)*E(k,n)$$

where:

- $W(k,n)$ = W' values calculated in subrouinte WCALC
- $E(k,n)$ = Incore detector millivolt signals from the Varian Datalogger
- $RODEFF$ = near rod correction factor
- $BB(k)*WP$ = non-near rod correction factor

2.3.6 SUBROUTINE BXPWR (continued)

The maximum correction factor, either near or non-near, is used to adjust the detector signals. Since currently BB(k) is zero for all detector strings, the near rod correction factor is the only one applied if any.

One final comment is that a slightly different calculation for the fractional insertion shadowing factor WP is done if a part-length rod is the most inserted rod. Insertion of a part-length rod during power operation is currently a violation of Technical Specifications. Therefore, the part length calculation should never be performed by the code.

Variable	Type	Description
INDEX	INTEGER INDEX(45)	Table of nearest rods to each of 45 detector strings
E	REAL E(45,5)	Incore detector signals, mv
P	REAL P(45,5)	Incore detector powers, MWth
W	REAL W(45,5)	Incore detector W' from WCALC
H	REAL H(45)	Control rod positions, cm withdrawn
H1	REAL H1	Position of near or non-near rod
WP	REAL WP	Fractional insertion shadowing factor
ZM	REAL ZM(45,5)	Incore detector powers, MWth
AA	REAL AA(45)	Near fully shadowed rod correction coefficient from Library data file
BB	REAL BB(45)	Non-near fully shadowed rod correction coefficient from Library data file
ALPHA	REAL ALPHA(5)	Top of axial detector regions from core bottom, cm
BETA	REAL BETA(5)	Bottom of axial detector regions from core bottom, cm

Table #2.3.6 Variables used in SUBROUTINE BXPWR

2.3.7 SUBROUTINE DETSIG (Edit #10)

Subroutine DETSIG is an editing routine that generates the log of detector signals and powers that appears in the PIDAL output report. This report arranges the detectors strings in groups of eighth core or octant symmetric detectors.

For each detector string, the Rhodium depletion and background corrected millivolt detector signals and W' corrected detector powers in MWth are edited for each detector. These values come from the variables E and P respectively. Subroutine DETSIG performs no calculations.

2.3.8 SUBROUTINE CHPMIN (Edit #11)

The purpose of Subroutine CHPMIN is to verify that the incore detector powers returned from BXPWR in array P have powers greater than the hand input value PMIN (card 7). If any detector power is less than PMIN, then that detectors millivolt signal and power, in arrays E and P respectively, are set to zero. In addition, the corresponding flag, IFAIL, for the failed detector is set to '*'.

If any detectors are failed due to low power, then a report containing a list of all of the failed low power detectors by string and axial level is generated under Edit #11.

2.4 Program Section #4

Program section #4 controls the subroutines that calculate the full core power distribution and quarter core tilts for PIDAL. This program section is controlled entirely by Subroutine PIDAL which is in turn called by the main program.

The original methodology used in program section #4 was based on the Combustion Engineering method CECOR which is documented in Reference #11. Over the course of development, several enhancements were made to this methodology. In general, PIDAL departs from CECOR in the way that the intra-assembly coupling is performed. This will be discussed in detail later.

The underlying assumption in the CECOR method is that the power in any assembly K can be given by the equation (not using PIDAL variables) :

$$P(K,j) = \frac{\sum^{m} P(m,j)}{CC(K,j)} \qquad \text{(equation 2.4a)}$$

where:

P(K,j) = power in assembly K, axial level j
P(m,j) = power in assembly m, axial level j. Assembly m is adjacent to assembly K.
CC(K,j) = assembly average coupling coefficient for assembly K, axial level j.

Using algebra, CC(K,j) is defined as:

$$CC(K,j) = \frac{\sum^{m} P(m,j)}{P(K,j)} \qquad \text{(equation 2.4b)}$$

In the Palisades core, there are 204 assemblies (K=204) and thus the dimension on CC is also 204. Obviously, P(K,j) is only known for assemblies which contain an operable incore detector. Therefore, a synthesis method is used to determine powers for each of the uninstrumented locations.

It is necessary for the CC(K,j) to be defined or known for all core locations. The CECOR method precalculates the CC(K,j) from planar depletion studies and fits these values based on burnup. This would be similar to calculating the CC(K,j) based solely on the XTG predicted solution.

The problem with this method is the CC(K,j) are based only on predicted and that the final full core solution is greatly biased by the predicted solution. Therefore, it was determined that the preferred method would be to somehow infer the CC(K,j), based on measurement, as well. The method for doing this will be described in sections 2.4.4 through 2.4.11.

DESCRIPTION of the MAIN PROGRAM and SUBROUTINES

2.4 Program Section #4 (continued)

Once the CC(K,j) are known, the problem consists of determining the P(K,j) for uninstrumented assemblies. Equation 2.4a can be rewritten as:

$$P(K,j) * CC(K,j) = \sum^{m} P(m,j) \qquad \text{(equation 2.4c)}$$

If equation 2.4c is written for each uninstrumented assembly, K, then a set of equations on the order (204-45) for each detector level result. If the P(m,j) which are unknown are subtracted over to the left hand side of each equation and the known P(m,j) remain on the right, then a very sparse but symmetric matrix of equations appears on the left and a known vector appears on the right. Remember (or assume) that the CC(K,j) are known.

Assuming that locations 4 and 9 (using the full core assembly numbering scheme of Edit #5) are instrumented, the first 5 equations would be written for each detector level j, in expanded matrix-vector notation as:

P1CC1	-P2	-0	-0	-0	-0	-0	-0	-0	-0	= P9
-P1	+P2CC2	-P3	-0	-0	-0	-0	-P10	-0	-0	= 0
-0	-P2	+P3CC3	-0	-0	-0	-0	-0	-P11	-0	= P4
-0	-0	-0	+P5CC5	-P6	-0	-0	-0	-0	-0	= P4+P13

Because assemblies 4 and 9 are instrumented, rows and columns 4 and 9 were omitted from the set of equations. The corresponding matrix equation for the above set of equations is written as:

$$\underline{\bar{A}}\ \vec{P} = \vec{S} \qquad \text{(equation 2.4d)}$$

$\underline{\bar{A}}$ is a coefficient matrix consisting of the CC and -1 multipliers to the P(m,j). The unknown P(K,j) and P(m,j) (on the left hand side) are symbolically contained in the vector $\vec{P}$. The vector $\vec{S}$, or known vector, consists of the column of sums on the right hand side. What must be found is the solution to the vector $\vec{P}$.

As eluded to by C.E. in Ref #11, the $\underline{\bar{A}}$ matrix is very sparse and in our case symmetric. Therefore, an efficient way of solving the above set of simultaneous equations should be employed. C.E. claims they use something called the conjugate gradient method, but after scanning their reference, nothing by this name was found. Thus, a trustworthy Gaussian elimination routine for symmetric sparse matrices was employed. The routine was part of the YALE Sparse Matrix Package available in the public domain.

After the vector $\vec{P}$ has been determined, it is recombined with the known values of P(K,j) and the full core radial power solution for each detector level is obtained.

The following sections desribe the subroutines that perform the tasks of determing the CC(K,j) and setting up and solving the simultaneous equations.

2.4.1 Subroutine COUPLE (Edit #12)

The purpose of Subroutine COUPLE is to calculate the assembly average coupling coefficents based only on the XTG calculated detector powers. These coefficients are used by Subroutine QUARTM in the process of calculating coupling coefficients based on a one-quarter core measured/inferred radial power distribution.

The definition of the assembly average coupling coefficient is given as:

$$CUP(i,j) = \frac{\sum^{IBUD} PDQS(IBUD,j)}{PDQS(i,j)}$$

where:

- CUP(i,j) = assembly average coupling coefficient for assembly i, detector level j.
- PDQS(IBUD,j) = theoretical detector power in assembly IBUD which is adjacent to assembly i detector level j.
- PDQS(i,j) = XTG calculated detector power for assembly i, detector level j.
- IBUD = assembly number of bundle adjacent to assembly i. There are four adjacent locations for each assembly i.

After the array CUP has been calculated, it is edited by detector level as EDIT #12. The quarter core coupling coefficient maps are generated by subroutine QTRMAP of section 2.4.2.

2.4.2 Subroutine QTRMAP

Subroutine QTRMAP is an editing routine which can edit quarter core radial power distribution maps of either one or two variables per location. The type of map edited depends on the value of variable IOPT passed into the routine. QTRMAP also has the capability of initiating limited incore detector operability verifications by calling subroutine CHKDEV. See section 2.4.10 for a discussion of CHKDEV.

The function of QTRMAP is determined by the input value of IOPT as follows:

IOPT = 0 No edit. Calculate a deviation between two power distribution maps at each location and perform a check on the deviations via Subroutine CHKDEV.

IOPT = 1 Edit a quarter core map with only one power distribution variable at each location. Perform no checking.

IOPT = 2 Edit a quarter core map with two power distribution variables at each location. Also calculate deviations between the power maps at each location and edit them. But, do not perform checking via subroutine CHKDEV.

IOPT = 3 Edit a quarter core map with two power distribution variables at each location. Also calculate deviations between the power maps at each location and perform incore detector operability verification by calling subroutine CHKDEV.

The following subroutines call QTRMAP and use it for the following options:

IOPT=0 EDTQTR (called by SOLVEQ)

IOPT=1 EXPAND, EXTRAP, COUPLE, CUPDET

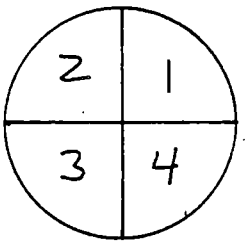
IOPT=2 EDTQTR (called by SOLVEQ)

IOPT=3 EDTQTR (called by SOLVEQ)

2.4.3 Subroutine TILT

The purpose of Subroutine TILT is to calculate the quarter core tilt for each detector level and the whole core based solely on measured incore detector powers. This is done by utilizing all operable two-way symmetric incore sets in the core to calculate six two-way tilt components for each detector level. The two-way tilts are then combined algebraically resulting in values for quadrant power tilt.

Assume a quadrant numbering scheme as follows:



Six intra-quadrant tilts can then be defined between each combination of two quadrants as: tilt 1-2, tilt 1-3, tilt 1-4, tilt 2-3, tilt 2-4 and tilt 3-4.

With tilt defined as the power in the quadrant of interest divided by the sum of the powers in all quadrants of interest, the six intra-quadrant tilts are defined mathematically as:

$$\text{tilt 1-2: } T(1) = \frac{P(1)}{P(1) + P(2)} \qquad \text{tilt 1-3: } T(2) = \frac{P(1)}{P(1) + P(3)}$$

$$\text{tilt 1-4: } T(3) = \frac{P(1)}{P(1) + P(4)} \qquad \text{tilt 2-3: } T(4) = \frac{P(2)}{P(2) + P(3)}$$

$$\text{tilt 2-4: } T(5) = \frac{P(2)}{P(2) + P(4)} \qquad \text{tilt 3-4: } T(6) = \frac{P(3)}{P(3) + P(4)}$$

Using the same definition of tilt, but now expanding it to all four quadrants, quadrant tilt is defined as:

$$\text{quadrant tilt 1: } TQ(1) = \frac{P(1)}{P(1) + P(2) + P(3) + P(4)}$$

$$\text{quadrant tilt 2: } TQ(2) = \frac{P(2)}{P(1) + P(2) + P(3) + P(4)}$$

$$\text{quadrant tilt 3: } TQ(3) = \frac{P(3)}{P(1) + P(2) + P(3) + P(4)}$$

$$\text{quadrant tilt 4: } TQ(4) = \frac{P(4)}{P(1) + P(2) + P(3) + P(4)}$$

2.4.3 Subroutine TILT (continued)

By substituting in for the separate values of power using the six two-way tilts, the quadrant tilts can be redefined as:

$$TQ(1) = \frac{P(1)}{P(1) + P(1)\frac{1 - T(1)}{T(1)} + P(1)\frac{1 - T(2)}{T(2)} + P(1)\frac{1 - T(3)}{T(3)}}$$

$$TQ(2) = \frac{P(2)}{P(2) + P(2)\frac{T(1)}{1 - T(1)} + P(2)\frac{1 - T(4)}{T(4)} + P(2)\frac{1 - T(5)}{T(5)}}$$

$$TQ(3) = \frac{P(3)}{P(3) + P(3)\frac{T(2)}{1 - T(2)} + P(3)\frac{T(4)}{1 - T(4)} + P(3)\frac{1 - T(6)}{T(6)}}$$

$$TQ(4) = \frac{P(4)}{P(4) + P(4)\frac{T(3)}{1 - T(3)} + P(4)\frac{T(5)}{1 - T(5)} + P(4)\frac{T(6)}{1 - T(6)}}$$

The powers P(1), P(2), P(3) and P(4) all cancel out which leaves us with expressions for fractional quadrant power tilts based only on the six intra-quadrant tilts.

The task of subroutine TILT is to determine the six intra-quadrant tilts and then the quadrant power tilt for each incore detector level. The six tilts are found by averaging the tilt values from each of the operable incore detector sets for the current detector level.

There are 22 possible two-way symmetric incore detector combinations. This is assuming one-quarter core rotational symmetry as opposed to one-quarter core reflective symmetry which would only have 15 possible sets. Note that both quarter core rotational and reflective are octant symmetric so the choice is arbitrary for current one-eighth core symmetric loadings. There is an obvious advantage to using rotational symmetry in the event that a quarter core loading pattern in required in the future. See Figures 2.4.3a and 2.4.3b for maps showing the reflective and rotational symmetric incore detector sets.

2.4.3 Subroutine TILT (continued)

TILT can function in either of two modes. The first consists of calculating the quadrant tilt for one of the five detector levels. For this mode input variable J has the value 1 through 5. The other mode calculates the quadrant power tilt for the total core, averaging the five detector level quadrant tilts. This mode is performed when input variable J is set to 6 and may also produce EDITs #27 and #53.

The possible two-way symmetric sets are stored in variable TWO with the corresponding intra-quadrant type (1 through 6) stored in variable TYPE. The average intra-quadrant tilts for each of the six types are calculated and stored in variable T. If any of the intra-quadrant tilts for a given detector level are undefined, i.e. no operable detectors for that tilt, then the tilt calculation for that level is terminated and control is returned to subroutine PIDAL.

If all six tilts are defined for a given detector level, then the quadrant tilt for that level is calculated and stored in variable DETTLT. Note the equations used are idetical to those given above except for the constant four(4) in each numerator which is a normalization factor to make the tilt values normal to unity.

If tilt was being calculated only for a single detector level, i.e. TILT was called by PIDAL while determining individual detector level power distributions then TILT returns to PIDAL at this point. If TILT was being called by PIDAL for the final total core quadrant tilt then the five possible quadrant tilt values (four for each detector level) are averaged axially in order to determine the total core, quarter core tilt. The total quarter core quadrant tilts are stored in variable QTILT and may be edited as EDIT #27.

EDIT #53 will be automatically produced if any of the quarter core tilts for a detector level are undefined. EDIT #53 is generated by Subroutine TLTERR and consists of a table showing all 22 possible symmetric detector combinations and a status by level of how many of these combinations were operable. This edit is also a default edit but may be over-ridden.

2.4.4 Subroutine COLAPS

The purpose of Subroutine COLAPS is to collapse the full core detector powers to an average quadrant either using rotational or reflective symmetry. For current loadings, reflective symmetry is used because one more quarter core assembly location is instrumented than in the rotationally symmetric case. If future loadings are rotationally symmetric, then input variable REFROT in card #2 must be set. See section #2.1.

When performing the collapse to quarter core, COLAPS has the ability to correct for calculated quarter core tilts for each detector level. Whether or not the tilt correction is performed depends on input variable ITILT. If ITILT=0 then no tilt corrections are performed. If ITILT=1 then the tilt for each detectors level and quadrant is divided out. In general, tilt corrections are performed only during the final determination of the quarter core power distribution for a detector level. i.e. after the detectors have been verified for operability.

The collapsed quarter core detector powers are stored in variable DETPOW for later use in the quarter core power distribution solution.

2.4.5 Subroutine XTGPOW

The purpose of Subroutine XTGPOW is to replace all measured detector powers with theoretical detector powers calculated from the XTG solution. This was made possible so that the full core synthesis routines could be verified. By using the theoretical detector powers, the solution returned by full core routine should closely agree with the XTG solution.

XTGPOW is essentially a simple lookup and copy routine. A loop is run over all 45 incore detector strings (and all 5 levels). Within the loop, the theoretical detector powers which were normalized by axial detector level and stored in array TDET by Subroutine EXPAND are copied into the appropriate location in array DETPOW. Subroutine XTGPOW then returns to Subroutine PIDAL.

Note that XTGPOW is called only if input flag IXPOW=1 on input card 2.

2.4.6 Subroutine QUARTM (Edit #13)

The purpose of Subroutine QUARTM is to set up the general form of the one-quarter core coefficient matrix. The matrix is then used in determination of the quarter core radial power distribution for a detector level, from which coupling coefficients, base on this measured/inferred power distribution can be derived.

Subroutine QUARTM sets up the matrix in the same fashion as the full core matrix set up by Subroutine SETUPM of section 2.4.12. The only difference lies in the fact that the detector powers used are those collapsed from full core to quarter core by Subroutine COLAPS. QUARTM was written to be flexible enough to work for either quarter core reflective or rotational symmetry, depending on variable REFROT of input card #2. As an aid to the user, it is possible to edit the general form of the quarter core matrix after set-up by selecting Edit #13. This forces QUARTM to call Subroutine EDMATQ which in turn produces the edit.

As an example, the first eleven terms of the general quarter core matrix assuming reflective symmetry is given below. Remember that this matrix is general and will be corrected for operable instrument locations by Subroutine ALTERQ. See Figures 2.4.6a and 2.4.6b for maps showing the quarter core reflective and rotational assembly numbering schemes.

cc1-2	-1	0	0	0	0	0	0	-1	0	0
-1	cc2-1	-1	0	0	0	0	0	0	-1	0
0	-1	cc3-1	-1	0	0	0	0	0	0	-1
0	0	-1	cc4-1	-1	0	0	0	0	0	0
0	0	0	-1	cc5-1	-1	0	0	0	0	0
0	0	0	0	-1	cc6-1	-1	0	0	0	0
0	0	0	0	0	-1	cc7-1	-1	0	0	0
0	0	0	0	0	0	-1	cc8-1	0	0	0
-1	0	0	0	0	0	0	0	cc9-1	-1	0
0	-1	0	0	0	0	0	0	-1	cc10	-1
0	0	-1	0	0	0	0	0	0	-1	cc11

Note that for the first eight terms, the top couples are with themselves and thus the value of one subtracted off the coupling coefficient. It is advised that the reader write the actual coupling equations for these eleven assemblies in order to verify the process used for generating this matrix.

2.4.7 Subroutine QUARTV (Edit #14)

The purpose of Subroutine QUARTV is to set up the general form of the quarter core solution vector which corresponds to the matrix set up by QUARTM. The method of generating this vector is completely analagous to that of the general full core solution vector created by Subroutine SETUPV, Section 2.4.13.

In short, the value of the solution vector component for a single assembly is the sum of the detector powers which are located in the assemblies which are coupled with it. Therefore, a search is performed for each row of the general matrix and the sum of all detector powers indexed by a -1 couple becomes the solution vector component for that assembly.

Two notes: first, since only the upper triangle of the matrix is stored, because the matrix is symmetric about the diagonal, the solution vector components for two assemblies are indexed by each -1 occurance in the matrix. One corresponding to the row and one to the column of the -1 found. Secondly, remember that this is only the general vector and components for operable instruments are removed by ALTERQ.

As with QUARTM, a satellite routine, EDVECQ exists which will edit the general vector as EDIT #14 if requested.

2.4.8 Subroutine ALTERQ (Edit #15)

The purpose of Subroutine ALTERQ is to adjust the general forms of the quarter core coefficient matrix and solution vectors, set up by routines QUARTM and QUARTV respectively, for instrumented quarter core assembly locations. This routine is analagous to Subroutine ALTER (section 2.4.14) which adjusts the full core matrix and vector for operable detectors.

The main idea is to eliminate rows and columns which correspond to instrumented quarter core assembly locations from the general coefficient matrix. In turn, vector components which correspond to instrumented assemblies (and are actually the right-hand-side of eliminated matrix rows) are also deleted from the vector. At the same time, both the matrix and vector are compressed so that no "place-holders" for instrumented assemblies exist.

With this in mind, if assemblies 1,3,4,7 and 10 were instrumented for the quarter core, the general matrix of Section 2.4.6 would be altered to the following:

cc2-1	0	0	0	0	0
0	cc5-1	-1	0	0	0
0	-1	cc6-1	0	0	0
0	0	0	cc8-1	0	0
0	0	0	0	cc9-1	0
0	0	0	0	0	cc11

After the general matrix and vector have been reduced, they may be edited as Edit #15. Both may be edited if the edit flag is set to 3 or just the matrix (edit flag=1) or just the vector (edit flag=2).

2.4.9 Subroutine SOLVEQ (Edit #16)

The purpose of Subroutine SOLVEQ is to obtain and edit the solutions to the quarter core set of simultaneous equations which were set up by Subroutine ALTERQ. This routine is analagous to Subroutine SOLVES described by section 2.4.15.

SOLVEQ reorders the equations in memory and then calls the simulataneous equation solution routine SDRV. Up to now, the upper triangle of the matrix was stored in the MATRIX array. SDRV only requires storage of the non-zero matrix elements and their respective indices, so within SOLVEQ, a simple storage algorithm is employed in order to prepare the matrix for solution in SDRV.

Upon completion of SDRV, control returns to SOLVEQ which determines if an error occurred during solution of the simultaneous equations. If an error occurrs, a warning message with debug aids is edited.

If the SDRV solution is successful, the solution to the matrix is returned to SOLVEQ in variable VECTOR. VECTOR is then combined with the original known detector powers in DETPOW to form a complete full core radial power distribution.

After the quarter core power distribution for the current detector level has been stored in DETPOW it is edited via Subroutine EDTQTR. EDTQTR also calls EDTMAP which may call CHKDEV to verify incore detecor powers for operability. See sections 2.4.2 and 2.4.10 for discussions of EDTQTR and CHKDEV, respectively.

2.4.10 Subroutine CHKDEV (Edit #17)

CHKDEV is currently the only Subroutine in the PIDAL program which can fail detectors for reasons other than the minimum power check in CHPMIN. CHKDEV checks the deviation between PIDAL and XTG power distributions for each instrumented assembly location. If the deviation is greater than the value DEV (default 20%) input to PIDAL on card 7, then the detector is considered failed.

If a detector is failed, it's millivolt signal (array E) and detector power (array P) are zeroed out. A message under Edit #17 is also issued stating which detectors were failed for each level and the corresponding detector failure mode flag IFAIL is set to either a ">" if the detector was failed for reading high, or a "<" if the detector was failed for reading low.

Because the failed detector had an adverse effect on the PIDAL power distribution, the solution for that level must be re-determined. This is accomplished by setting the flag IDEV=1 and re-running the quarter core solution sequence.

When control returns to PIDAL (through completion of SOLVEQ) the IDEV=1 flag causes the solution for the current level to start over with Subroutine TILT. Under normal conditions, when a detector must be failed, There should only be a maximum of three passes through the quarter core solution sequence for a given detector level.

The first pass obtains the quarter core solution (not tilt corrected) and identifies any detectors which need to be failed. If detectors are failed then a second pass (not tilt corrected) through the quarter core solution is made to verify detectors again. A third pass (tilt corrected) and final pass through the quarter core solution sequence is then made. If no detectors are failed by the first pass then the second non tilt corrected pass is skipped.

2.4.11 Subroutine CUPDET (Edit #18)

Once the tilt-corrected quarter core radial power distribution solution is obtained for a detector level, Subroutine CUPDET is called to re-calculate the intra-assembly coupling coefficients for use in the full core solution. These coupling coefficients are calculated identically to those derived in Subroutine COUPLE (section 2.4.1) except for the fact that these couples are based on the quarter core measured/inferred radial power distribution instead of the XTG calculated distribution.

After the new coupling coefficients are determined and stored in array CUP, they may be output as Edit #18. It is interesting to compare these with the calculated values of Edit #12.

As a final step before going to the full core solution for the detector level, the full core measured detector powers are re-stored in array DETPOW. Incidently, if the input flag IXPOW was set, then XTG powers are placed in DETPOW instead of measured values. This is again useful for debugging and testing.

2.4.12 Subroutine SETUPM (Edit #19)

The purpose of Subroutine SETUPM is to set up the general form of the coefficient matrix A described in section 2.4. The matrix written in SETUPM is referred to as general because it assumes that all assemblies are not instrumented. This allows for later deleting of instrumented rows and columns of the matrix by Subroutine ALTER.

If the general matrix is written out, it can be seen that the assembly average coupling coefficients appear only on the diagonal elements. With the matrix being stored in array MATRIX, the element MATRIX(1,1) gets the coupling term for assembly 1, element MATRIX(2,2) gets the couple for assembly 2 and so on. The assembly average coupling coefficients come from array CUP which was filled in Subroutine CUPDET. Remember that CUP was generated for a quarter core and therefore a lookup is performed to assign the correct coefficients for each assembly.

Another property of the general matrix is that there are only four diagonals which contain values. These diagonals correspond to the four types of neighbors to each assembly: top, bottom, left and right. The diagonal elements consist of values of -1. The method used to fill the diagonal elements of MATRIX is nothing special. Since no good way of doing this was obvious, the diagonals are filled one at a time by taking advantage of knowing what the general form is.

Greater detail on what the general matrix looks like and how it is filled is left up to the coding. The general form can be edited out if desired via Edit #19 and Subroutine EDMAT. This edit is recommended for very infrequent use because large quantities of output are generated. It is highly recommended that the reader write the first 10 equations of the general matrix in order to completely understand this subroutine.

2.4.13 Subroutine SETUPV (Edit #20)

The purpose of Subroutine SETUPV is to set up the general form of the solution vector which corresponds to the S vector of section 2.4. This turns out to be a very simple process after observing the characteristics of the general matrix.

Given an operable detector, the full core assembly number of that detector is determined by a table lookup. This assembly number (between 1 and 204) is assumed to be a column number in the general matrix. A search is then performed for each row in the column corresponding to the detectors location. If a -1 is found in any row, then the detector power (corresponding to the column we are searching down) will appear in the summation of the same row in the general solution vector.

This may be a little difficult to visualize. It is suggested that the reader trace SETUPV for the first 10 assemblies (this covers detectors 1 and 2) and set up the vector. The vector should then be compared to the right-hand-side of the general assembly power equations desribed in section 2.4.

The general vector, which is stored in array VECTOR can be edited via Edit #20 and Subroutine EDVEC. EDVEC, like EDMAT also produces a significant amount of output.

2.4.14 Subroutine ALTER (Edit #21)

The purpose of Subroutine ALTER is to take the general forms of the full core equation matrix and vector and alter them based on which assemblies have operable detectors.

Recall that the final set of simulataneous equations are to be written for only uninstrumented assemblies. Up to this point, the matrix (and vector to a certain extent) are general for all locations. Thus the equations for the instrumented locations must be omitted.

The equations for the instrumented assemblies are omitted by simply compressing the MATRIX and VECTOR arrays down. For example, if string 1 or assembly 4 is operable, then row and column 4 are deleted from the MATRIX and row 4 is deleted from the VECTOR. Actually, all rows and columns are just shifted down so that the assembly 5 equation occupies the equation 4 location, and equation 6 occupies the equation 5 location and so on.

Once the final forms of MATRIX and VECTOR are completed, they may be edited under EDIT #20, using EDMAT and EDVEC. This is done by setting the input flag for Edit #20 to 3 (for both matrix and vector), to 1 (for matrix only) or to 2 (for vector only). It is important that the user realize that the entire 204*204 matrix (and 204*1 vector) are edited, but only the first ROWCOL*ROWCOL (and ROWCOL*1) locations contain valid data. ROWCOL is simply the number of equations or uninstrumented assemblies.

2.4.15 Subroutine SOLVES (Edit #22)

The purpose of Subroutine SOLVES is to obtain and edit the solutions to the set of simultaneous equations which were set up in Subroutine ALTER.

SOLVES reorders the equations in memory and then calls the simulataneous equation solution routine SDRV. Up to now, the upper diagonal of the matrix was stored in the MATRIX array. SDRV only requires storage of the non-zero matrix elements and their respective indexes, so within SOLVES, a simple storage algorithm is employed in order to prepare the matrix for solution in SDRV.

Upon completion of SDRV, control returns to SOLVES which determines if an error occurred during solution of the simultaneous equations. If an error occurrs, a warning message with debug aids is edited.

If the SDRV solution is successful, the solution to the matrix is returned to SOLVES in variable VECTOR. VECTOR is then combined with the original known detector powers in DETPOW to form a complete full core radial power distribution, which is stored in variable POWMAP by detector level.

After the full core power distribution for the current detector level has been stored in POWMAP it is edited via Subroutine EDTLVL. EDTLVL is merely a preprocessor to the actual edit routine EDTMAP.

2.4.16 Subroutine SDRV

Subroutine SDRV is a routine supplied as part of the YALE Sparse Matrix Package. SDRV solves a system of simultaneous linear equations using Gaussian Elimination. SDRV takes advantage of the fact that the coefficient matrix is very sparse and therefore only requires storage of the non-zero matrix elements. For a complete discussion of SDRV, the reader is referred to the source code as the routine is truly self documenting.

2.4.17 Subroutine SAVDEV

Deviations between measured and predicted powers for each full core assembly are calculated in Subroutine EDTMAP. These deviations are passed to Subroutine SAVDEV which in turn saves those deviations, for only the instrumented locations, in variable DETDEV. These are later used by Subroutine HISTOG.

2.5 Program Section #5

Up to this point, the core power distribution is given by a radial power distribution at each of five detector levels. The purpose of program Section #5 is to perform axial curve fitting of the five known powers within each assembly. From the curve fitting, a continuous function for axial power shape within the assembly is defined. The continuous functions can then used to infer the axial power distribution in the regions not covered by incore detectors.

The method employed to perform the axial curve fitting was the same as used by the current CPCo 1/8th core PIDAL system (also the CECOR method). The general idea is to apply a five mode Fourier fit to the axial data for each assembly, resulting in a continuous function describing the axial distribution. In order to do this, two assumptions are made:

1) It is assumed that the axial power distribution shape is adequately approximated by the Fourier sine function. Thus the function may be used for interpolating or extrapolating to compensate for the gross measurements.

2) It is assumed that the power goes to zero near the top and bottom of the active fuel height. This assumption allows us to add the points near the top and bottom to the five known points, effectively improving the curve fit.

From the continuous function, a distribution consisting of 51 axial nodal powers for each assembly is generated. This distribution is also collapsed to 25 axial nodes and even further to a 2-D radial and a 1-D axial power distribution.

The following sections discuss each of the subroutines within program section #5 in detail.

Note: The five mode Fourier functionalization used by PIDAL (and CECOR) is actually the first five terms of a sine series. A Fourier series generally contains both sine and cosine series' for the functionalization.

2.5.1 Subroutine ZIADA2

Subroutine ZIADA2 performs the axial fitting of the five detector powers within an assembly, yielding values for point powers as a function of axial position. For this discussion, point powers are the power at a given axial position as opposed to an integral power covering an axial length.

The underlying assumption is that the axial power distribution within an assembly can be given by the sum of the first few Fourier modes. Written in equation form:

$$P(z) = \sum_{n=1}^{N} a_n \sin(n\pi B z) \qquad \text{(equation 2.5.1a)}$$

where:

- $P(z)$ = point power as a function of core fractional core height
- N = number of Fourier modes (5 in our case)
- a_n = unknown coefficients (5 in our case) which are calculated within ZIADA2
- B = axial boundary condition, the real core height as a fraction of the apparent core height, $B = H/(H + 2\delta)$
- z = fractional core height, h/H (between 0 and 1)
- δ = extrapolation distance, the extra distance past the core edge where the flux apparently goes to zero, $(H/2)(1/B - 1)$
- h = axial position within core in inches
- H = core height (132 inches)

Equation 2.5.1a is not precisely what is used to describe the axial power distribution by PIDAL. Considering Figure #2.5.1 and the definition of z, it should be clear that using 2.5.1a to the describe the axial power shape (and implicitly the flux shape) assumes that the power (and flux) at h=0 and h=H+2δ are zero. In fact, this is not correct. The flux actually goes to zero at h=$-\delta$ and h=H+δ . Therefore, we shall re-define P(z) such that at the point where the flux goes to zero:

$$P(z) = \sum_{n=1}^{N} a_n \sin(n\pi (Bz)') \qquad \text{(equation 2.5.1b)}$$

$(Bz)'$ in this case is an adjusted value for Bz. The wave lengths of the sine functions have been increased to match the core height that has been increased by the extrapolation distance, 2δ. The term, B, serves this purpose. Also, the core height dimension, z, must be biased to be consistent with the larger apparent core height. In effect, the sine function, which was lengthened by B, must be re-centered about the true center of core.

The term $(Bz)'$ will be defined as Bz plus a co-ordinate translation function (for the re-centering). In equation form:

$$(Bz)' = Bz + T$$

The boundary conditions for determining the function T are given by assuming that at h=$-\delta$, $(Bz)'=0$ and at h=H+δ, $(Bz)'=1$. This allows use of the entire positive half of the sine function for describing the power shape and is consistent with the general form of 2.5.1a (B=1) shown by the solid line of Figure #2.5.1.

2.5.1 Subroutine ZIADA2 (continued)

T is determined for $h=-\delta$ as follows:

Known: $z=-\delta/H$, $\sin(n\pi(Bz)') = 0$ and $\delta = (H/2)(1/B - 1)$

Therefore, $n\pi(Bz)' = 0$ and $(Bz)' = 0$ (by taking the arc sine of each side of the above known equation)

Expanding $(Bz)'$, $Bz + T = 0$ and $T = -Bz$

Substituting for z, $T = B(\delta/H)$

Substituting for δ, $T = \frac{B}{H} \frac{H}{2}(1/B - 1)$

Reducing, $T = \frac{B}{2}(1/B - 1) = (1 - B)/2$

The value, T, can then be verified for the other zero flux boundary.

Known: $z=(H+\delta)/H$, $\sin(n\pi(Bz)') = 0$ and $\delta = (H/2)(1/B - 1)$

Therefore, $n\pi(Bz)' = 1$ and $(Bz)' = 1$

Expanding $(Bz)'$, $Bz + T = 1$ and $T = 1 - Bz$

Substituting for z, $$T = 1 - \frac{B(H+\delta)}{H}$$

Rearranging, $$T = 1 - B\left(1 + \frac{\delta}{H}\right)$$

Substituting for δ, $$T = 1 - B\left(1 + \frac{H/2(1/B - 1)}{H}\right)$$

and reducing, $$T = 1 - B - B/2(1/B - 1)$$

$$T = 1 - B - 1/2 + B/2$$

$$T = 1/2(1 - B) \quad \text{(equation 2.5.1c)}$$

2.5.1 Subroutine ZIADAZ (continued)

Equation 2.5.1c is then replaced back into equation 2.5.1b to yield the equation that PIDAL uses for P(z).

$$P(z) = \sum_{n=1}^{5} a_n \sin(n\pi(Bz + 1/2(1 - B))) \text{ or}$$

$$P(z) = \sum_{n=1}^{5} a_n \sin(n\pi B(z - .5) + n\pi/2) \qquad \text{(equation 2.5.1d)}$$

From equation 2.5.1d, it is apparent that the unknowns are the five a_n coefficients. (The boundary conditions are determined from XTG in EXTRAP.) The powers seen by the incore detectors, which are integral powers, lend a convenient method for determining the a_n values.

Defining the integral power seen by an incore, i, as:

$$\text{Power}_i = \int_{z_{i,bottom}}^{z_{i,top}} \sum_{n=1}^{5} a_n \sin(n\pi B(z - .5) + n\pi/2)\,dz \qquad \text{(equation 2.5.1e)}$$

and by defining: $u = n\pi B(z - .5) + n\pi/2$ and $du = n\pi B\,dz$, allows transformation of equation 2.5.1e to:

$$\text{POWER}_i = \int_{z_{i,bottom}}^{z_{i,top}} \sum_{n=1}^{5} a_n \sin(u) \frac{du}{n\pi B}$$

Integrating yields:

$$\text{POWER}_i = \sum_{n=1}^{5} \frac{-a_n}{n\pi B} \cos(u) \Big|_{z_{i,Bottom}}^{z_{i,top}}$$

Substituting for u gives us:

$$\text{Power}_i = \sum_{n=1}^{5} \frac{-a_n}{n\pi B} \cos(n\pi B(z - .5) + n\pi/2) \Big|_{z_{i,bottom}}^{z_{i,top}} \qquad \text{(equation 2.5.1f)}$$

If equation 2.5.1f is written in expanded form for each of the five known axial powers in an assembly, then a system of five equations and five unknowns results. This system of equations can then be solved simultaneously for the a_n values. The resulting a are used in equation 2.5.1d, which can now describe the axial power profile in an assembly.

2.5.1 Subroutine ZIADA2 (continued)

To summarize Subroutine ZIADA2:

1) A loop is performed over each assembly in the core. Within the loop, Equation 2.5.1f is expanded to five terms for each of the five axial powers known for that assembly, resulting in a system of five unknowns (the a_n) and five equations. The system is solved simulataneously and the combining coefficients, a_n, are saved.

2) After the combining coefficients for all 204 assemblies are known, a second loop over the assemblies is performed. In this loop, the combining coefficients are used in equation 2.5.1d in order to determine point powers at 51 equally spaced axial positions in each assembly. When this loop is complete, the full core three-dimensional power distribution consisting of 204 assemblies by 51 axial nodes is known.

2.5.2 Subroutine REDUCE

The purpose of Subroutine REDUCE is to normalize the full core three-dimensional power distribution to the calorimetric core power. This is necessary because the total integrated core power from the detectors does not usually add up to the calorimetric value due to the detector signal to power conversion method.

The full core three-dimensional power distribution consisting of 204 assemblies by 51 axial nodes is stored in variable FULL3D by Subroutine ZIADA2. The first step in REDUCE is to calculate the total integrated core power given by FULL3D. This is done by calculating the integrated power in MWth for each assembly and then adding up all of the assembly powers.

The integrated assembly powers are calculated by utilizing the trapezoidal rule which can be written formally as:

$$\int_a^b f(x)\,dx \simeq \frac{b-a}{2n}\left[f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n)\right]$$

In the code, the trapezoidal rule is implemented for an assembly i by using one-half of the powers in the top and bottom nodes:

```
AXIAL = (FULL3D(i,1) + FULL3D(i,51))/2.
```

and then integrating up assembly, i, using:

```
AXIAL = AXIAL + FULL3D(i,j)  j=2 to 50
```

The AXIAL values for each assembly are summed to generate the total core power:

```
CORPOW = CORPOW + AXIAL
```

In order to satisfy the trapezoidal rule, CORPOW is then divided by 50, which is equal to n given above. Note that the integration limits are from 0 to 1. This is because fractional core heights were used in the calculation of the FULL3D point powers. The reader should realize that the values in FULL3D are point powers per unit length and are thus point linear heat generation rates.

Once the integrated full core detector power has been determined, a normalization factor, RATIO , is calculated.

```
RATIO = CORPOW/CALPOW
```

RATIO is then used to normalize the full core 3-D detector power distribution to the calorimetric core power.

```
FULL3D(i,j) = FULL3D(i,j)/RATIO
```

2.5.2 Subroutine REDUCE (continued)

The maximum value of FULL3D(i,j) after normalization is saved in variable PEAK. PEAK is actually the core nodal LHGR peaking factor.

At this point, REDUCE transforms the point power distribution of FULL3D to 204 by 25 integrated nodal powers. This is also done via the trapeziodal rule. Each integrated nodal power (25 of them) is calculated by integrating over the three nearest point powers defined by FULL3D. The equation used is:

F3D25(i,k) = (FULL3D(i,j) + 2*FULL3D(i,j+1) + FULL3D(i,j+2))/4./25.

where:

F3D25(i,k) = integrated nodal power, assembly i,node k
k = axial node number, 1=bottom to 25=top
FULL3D = 204*51 point powers from ZIADA2
j = (k*2)-1, axial node number, 1=bottom to 51=top

The division by 4 is the division by 2*n from the trapezoidal rule formal definition (n=2). The division by 25 is actually a multiplication by the quantity (b-a) which is the core height of each integral node. Since the total assembly height is 1, (remember the fractional heights in ZIADA2), then each of the 25 integrated nodal powers cover 1/25th of the total core height.

The F3D25 204*25 integral nodal powers are summed up for the entire core in order to determine the total core power as before. This value should be very close to the calorimetric power, but due to numerical errors in the integration procedure, small diferences occur. Therefore, the powers in F3D25 are also normalized to CALPOW as was done above.

2.5.3 Subroutine RADIAL (Edits #23 and #24)

The purpose of Subroutine RDLAXL is to transform the three-dimensional 204 assemblies by 25 axial nodes power distribution (F3D25) calculated in Subroutine REDUCE to two-dimensions radially.

First, an integrated radial power distribution is generated by summing the axial nodal powers in each assembly. This gives a gross indication of the total power generated in each assembly of the core. The equation used is:

FULL2D(i) = FULL2D(i) + F3D25(i,j) j=1 to 25

where:
FULL2D(i) = integrated power for assembly i in MWth
F3D25(i,j) = nodal integral power for assembly i, axial node j

Second, a normalized radial power distribution is generated. This is done by dividing each of the assembly powers from above by the core average assembly power. The equation used is:

FULN2D(i) = FULL2D(i)/CALPOW*204. i=1 to 204
where:
FULN2D(i) = normalized power for assembly i
FULL2D(i) = integrated power for assembly i in MWth

Both the normalized and integrated assembly radial power distributions are edited using Subroutine EDTFUL under Edits #23 and #24, respectively. Edit #23 also contains the XTG normalized assembly radial power distribution and percent deviation between XTG and PIDAL for each assembly.

2.5.4 Subroutine HISTOG (Edit #52)

The purpose of Subroutine HISTOG is to produce a histogram of deviations between measured and predicted detector powers and to perform some simple statistical analysis on these deviations.

The deviations which HISTOG uses are those saved by Subroutine SAVDEV of section 2.4.17. HISTOG generates the histogram of these deviations and also calculates the first four moments for the deviation data: the mean, standard deviation, skewness and peakedness. The equations used are the following:

$$M1 \text{ (mean)} = \frac{\sum_{i=1}^{n} (x_i)}{n}$$

$$M2 \text{ (st. dev.)} = \sqrt{\frac{\sum_{i=1}^{n} (x_i - M_1)^2}{n}}$$

$$M3 \text{ (skewness)} = \frac{\sum_{i=1}^{n} (x_i - M_1)^3}{n (M_2)^3}$$

If the absolute value of M3 is less than 0.5 the distribution may be considered symmetric.

$$M4 \text{ (peakedness)} = \frac{\sum_{i=1}^{n} (x_i - M_1)^4}{n (M_2)^4}$$

M4 < 3 means a curve flatter than the normal distribution
M4 = 3 (roughly) means a normal distribution
M4 > 3 means a curve more peaked than the normal

Note that for all calculations, "n" weighting is used because no attempt is being made to estimate the moments for an entire population of deviations.

2.5.5 Subroutine OFFSET

The purpose of Subroutine OFFSET is to collapse the 3-D full core power distribution radially to a 1-D axial power shape. In addition, the core axial offset (axial shape index or power split) are also determined.

The first step is to collapse the full core solution to an average octant. Only instrumented assemblies are collapsed. The assemblies with inferred powers are initially skipped. After the collpase, a check is made to see if each octant location has values. If not then those locations were not instrumented in the full core. In order to fill these uninstrumented octant locations, all inferred powers symmetric to the uninstrumented location are averaged and used. The result is a 3-D octant power distribution which is stored in variable OCT3D.

OCT3D is then collapsed radially to obtain the average core axial power shape which is stored in variable ACAPS. The following equation is used:

ACAPS(j) = ACAPS(j) + OCT3D(i,j)*SYMOCT(i) i=1 to 28 j=1 to 25

where:

- ACAPS(j) = radially integrated planar power for axial plane j
- OCT3D(i,j) = nodal integral power for octant location i, axial node j.
- SYMOCT(i) = number of symmetric assemblies to octant location i in the full core.

ACAPS(j) = ACAPS(j)*25./CALPOW j = 1 to 25

where:

- ACAPS(j) = normalized planar power for axial plane j

The axial offset is determined by summing the top and bottom half core powers fractions from array ACAPS. The equations used for the summations are:

LOWER = LOWER + ACAPS(i) i=1 to 12
UPPER = UPPER + ACAPS(i) i=14 to 25

where:

- LOWER = total power fraction in the bottom half of the core
- UPPER = total power fraction in the top half of the core
- ACAPS(i) = power fraction for axial plane, i, from Subroutine RDLAXL

The axial offset is the calculated as:

$$AO = \frac{LOWER - UPPER}{LOWER + ACAPS(13) + UPPER}$$

2.6 Program Section #6

The purpose of program section #6 is to compare the core power distribution calculated by PIDAL against limits given by the Palisades Technical Specifications. The subroutines in this section perform the incore peaking factor and linear heat generation rate calculations. High alarm limit factors for the incores along with the allowable power level for operation with only the excore monitoring system are also determined.

Significant revisions to the Palisades Technical Specifications which dealt with core power distribution monitoring were made between EOC 7 and BOC 8. In general, the all pins and narrow water gap pins radial peaking factor monitoring requirements were removed. In addition, the inner pins and narrow water gap pin linear heat generation rate calculation requirements were removed, while the all pins calculations were modified. A change in the method of calculating allowable power level was also made.

The following sections discuss each of the subroutines within section #6 in detail. The discussions of the individual subroutines cover both the cycle 5 through 7 and cycle 8 Technical Specification calculation requirements.

2.6.1 Subroutine CPEAK

The purpose of Subroutine CPEAK is to calculate the assembly and pin peaking factors for the core and to save the maximum values by fuel type. There are four peaking factors calculated by CPEAK: assembly radial, all pins, inner pins and narrow water gap pins peaking factors. From Ref #7 section 1.1, the following definitions apply.

Assembly Radial Peaking Factor

Maximum ratio of individual fuel assembly power to core average assembly power integrated over the total core height, including tilt.

All Pins Peaking Factor (total radial peaking factor)

Maximum product of the ratio of individual assembly power to core average asembly power times the local peaking factor for that assembly integrated over the total core height, including tilt. Local peaking factor is defined as the maximum ratio of the power in an individual fuel rod to assembly average rod power.

Inner Pins Peaking Factor (total interior rod radial peaking factor)

Maximum product of the ratio of individual assembly power to core average assembly power times the highest interior local peaking factor integrated over the total core height. An interior fuel rod is any fuel rod of any assembly that is not on the assembly's periphery.

Narrow Water Gap Pins Peaking Factor

Maximum product of the ratio of individual fuel assembly power to core average fuel assembly power times the highest narrow water gap fuel rod local peaking factor integrated over the total core height including tilt. A narrow water gap fuel rod is a fuel rod adjacent to the narrow interfuel assembly water gap (a gap not containing a control rod).

For versions of PIDAL for Palisades cycle 8 and beyond, only the assembly radial peaking factor and inner pins peaking factors are calculated. Therefore, PIDAL was modified to not perform the all pins and narrow water gap pins peaking factor calculations.

2.6.1 Subroutine CPEAK (continued)

Subroutine CPEAK is called from PIDAL which is looping over each assembly. PIDAL passes the assembly number (between 1 and 204) into CPEAK. Therefore, it should be clear that CPEAK operates on only one assembly at a time.

CPEAK first performs pin peaking factor integrations. A routine is used which determines the appropriate pin peaking factor for the axial nodes based on rod bank insertion.

The reader should recall that pin peaking factors may have been read in from the W' library for several possible rod bank insertion sequences by Subroutine COFCAL. Also, the current rod bank configuration was determined by Subroutine RBANK.

The following equations are used to integrate the nodal peaking factors over the core height:

```
AFRT = AFRT + F3D25(i,j)*R(ioct,m)   j=1 to 25
IFRT = IFRT + F3D25(i,j)*T(ioct,m)   j=1 to 25
NFRT = NFRT + F3D25(i,j)*TN(ioct,m)  j=1 to 25
```

where:

- AFRT = integrated power times the all pins peaking factor for assembly i
- IFRT = integrated power times the inner pins peaking factor for assembly i
- NFRT = integrated power times the narrow water gap pins peaking factor for assembly i
- F3D25 = nodal integral power for assembly i, axial node j from Subroutine REDUCE
- R(ioct,m) = all pins peaking factor for octant location ioct, and rod bank m.
- T(ioct,m) = inner pins peaking factor for octant location ioct, and rod bank m.
- TN(ioct,m) = narrow water gap pins peaking factor for octant location ioct, and rod bank m.
- ioct = octant location (between 1 and 28) for assembly i
- m = rod bank region number for current axial node j

Peak rod powers are calculated from the integrated data using:

- RODKWA(i) = AFRT/NUMROD(ifutyp)*1000.*UNUSER, kilowatts peak rod power, all pins peaking for assembly i
- RODKWI(i) = IFRT/NUMROD(ifutyp)*1000.*UNUSER, kilowatts peak rod power, inner pins peaking for assembly i
- RODKWN(i) = NFRT/NUMROD(ifutyp)*1000.*UNUSER, kilowatts peak rod power, narrow water gaps pins peaking for assembly i

NOTE: All pins and narrow water gap pins calculations performed only for cycles 5,6 and 7.

2.6.1 Subroutine CPEAK (continued)

The peaking factors are then divided by the core average assembly power in order to satisfy the definitions given above.

```
AFRT = AFRT*204./CALPOW*UNUSER
IFRT = IFRT*204./CALPOW*UNUSER
NFRT = NFRT*204./CALPOW*UNUSER
```

where:

AFRT = total radial peaking factor for all pins
IFRT = total radial peaking factor for inner pins
NFRT = total radial peaking factor for narrow water gap pins
CALPOW = total core power in MWth
UNUSER = optional user input uncertainty multiplier supplied via input card #7. (not usually used)

At this point the assembly radial peaking factor is stored as:

```
AFR = FULNZD(i)*UNUSER
```

where:

AFR = assembly radial peaking factor
FULNZD(i) = core normalized power for assembly i calculated in Subroutine RDLAXL
UNUSER = optional user input uncertainty multiplier supplied via input card #7. (not usually used)

A check is then performed for each peaking factor in order to see if the current assembly's values are the maximum for the fuel type. In the case of the all pins peaking factor, the maximum values are stored by fuel type as follows:

FRTA(ifutyp) = maximum total peaking factor for all pins for fuel type ifutyp. Maximum value of AFRT.
IFRTA(ifutyp) = assembly number of the maximum AFRT for fuel type ifutyp.
NUMROD(ifutyp) = number of fuel rods in assembly of fuel type ifutyp.

The analagous variables for inner pins, narrow water gap and assembly radial peaking factors are left for the reader to find in CPEAK.

2.6.2 Subroutine LHGR

The purpose of Subroutine LHGR is to calculate nodal linear heat generation rates and allowed to measured linear heat generation rate ratios per Technical Specification (Ref #7) section 3.23.1. Subroutine LHGR is also called from the PIDAL loop over all 204 assemblies. Like CPEAK, LHGR only operates on one assembly at a time.

LHGR's may be calculated for the three types of pin peaking factors; all, inner, and narrow water gap pins, based on the rod bank dependent peaking factors from the W' library described in the discussion of Subroutine CPEAK.

For Palisades cycle 5,6 and 7 all three types of pin linear heat generation rates were performed. For cycle 8 and beyond, only the all pins LHGR calculations were performed.

The equations used are as follows:

```
LHGRA(i,j) = F3D25(i,j)*R(ioct,m)/RODLEN(ifutyp)*25.*UNUSER*1000.
LHGRI(i,j) = F3D25(i,j)*T(ioct,m)/RODLEN(ifutyp)*25.*UNUSER*1000.
LHGRN(i,j) = F3D25(i,j)*TN(ioct,m)/RODLEN(ifutyp)*25.*UNUSER*1000.
```

where:

- LHGRA(i,j) = peak pin lhgr for assembly i, axial node j for all pins peaking in KW/ft
- LHGRI(i,j) = peak pin lhgr for assembly i, axial node j for inner pins peaking in KW/ft
- LHGRN(i,j) = peak pin lhgr for assembly i, axial node j for narrow water gap pins peaking in KW/ft
- F3D25(i,j) = nodal power in MWth for assembly i, axial node j from Subroutine REDUCE
- R,T,TN = pin peaking factors for octant ioct and rod bank m from Subroutine COFCAL
- RODLEN(ifutyp) = total length in feet of all fuel rods in an assembly of type ifutyp
- ifutyp = fuel type for assembly i
- UNUSER = optional user input uncertainty multiplier supplied via input card #7. (not usually used)
- RODLEN/25. = fuel rod length in the axial region covered by node j

2.6.2 Subroutine LHGR (continued)

Allowed to measured lhgr ratios are calculated in Subroutine LHGR in order to verify compliance with the limits established in Ref *#7*. The equations used come from Ref *#7* section 3.23.1 and are as follows:

```
RATIOA(i,j) = ALOCAL(j)*FBE*AKWFTL(ifutyp)/LHGRA(i,j)
RATIOI(i,j) = ALOCIL(j)*IKWFTL(ifutyp)/LHGRI(i,j)
RATION(i,j) = ALOCIL(j)*NKWFTL(ifutyp)/LHGRN(i,j)
```

where:

RATIOA(i,j) = allowed to measured linear heat generation rate based on all pins peaking

RATIOI(i,j) = allowed to measured linear heat generation rate based on inner pins peaking

RATION(i,j) = allowed to measured linear heat generation rate based on narrow water gap pins peaking

ALOCAL(j) = allowable LHR for axial node j, Ref *#7* Figure 3.23-1

ALOCIL(j) = allowable LHR for axial node j, Ref *#7* Figure 3.23-3

FBE = allowable LHR for peak pin burnup, assembly i, Ref *#7* Figure 3.23-2. See Subroutine GETFBE. For cycle 8 and beyond, FBE is omitted from the equation.

AKWFTL(ifutyp) = allowed LHR in KW/ft for assembly type, ifutyp based on all pins peaking

IKWFTL(ifutyp) = allowed LHR in KW/ft for assembly type, ifutyp based on inner pins peaking

NKWFTL(ifutyp) = allowed LHR in KW/ft for assembly type, ifutyp based on narrow water gap pins peaking

2.6.3 Subroutine GETFBE

The purpose of Subroutine GETFBE is to calculate a burnup dependent linear heat generation rate penalty factor. The penalty factor is applied to the linear heat generation rates based on all pins peaking per Ref #7 section 3.23.1. For cycle 8 calculations and beyond, the burnup penalty was omitted and Subroutine GETFBE was deleted.

The LHGR burnup penalty factor, variable FBE, is calculated based on the peak pin average nodal exposure for the assembly in question. The peak pin average nodal exposure is calculated based on the exposure pin peaking factor supplied to PIDAL in the W' library and the current nodal exposure distribution in the assembly. The equations used are:

PPEXP = PPEXP + FUEXP(i,j)*PINEXP(ioct,m) j=1 to 25

where:

PPEXP = integral peak rod exposure for assembly i, MWD/MTU

FUEXP(i,j) = nodal exposure for assembly i, axial node j, MWD/MTU

PINEXP(ioct,m) = exposure pin peaking factor for octant ioct, rod bank m

The average nodal peak pin exposure is then calculated as:

PPEXP = PPEXP/25. average peak pin nodal exposure in MWD/MTU

Once PPEXP is known, it is passed to the function FBEF. FBEF returns the correct FBE for the exposure PPEXP based on Figure 3.23-2 of Ref #7. Both FBE and PPEXP are then returned by GETFBE to the calling routine.

2.6.4 Subroutine LLHGR

The purpose of Subroutine LLHGR is to find and save the limiting linear heat generation rates by fuel type. The limiting values may be found for each pin peaking type: all, inner, and narrow water gap pins. This subroutine, like CPEAK and LHGR, is called once for each assembly from PIDAL.

The allowed to measured linear heat generation rate arrays are searched, with each nodal value compared with the previously found limiting value for the fuel type. The most limiting value is simply the lowest ratio.

For all pins linear heat generation rates, the following variables are used:

LIMITA(ifutyp) = lowest allowed to measured ratio of all pins LHGR for fuel type ifutyp. Array searched is RATIOA calculated in Subroutine LHGR.

LAI(ifutyp) = assembly number of the peak all pins LHGR for fuel type ifutyp

LAJ(ifutyp) = axial node number of the peak all pins LHGR for fuel type ifutyp

The analogous variables for inner pins and narrow water gap pins limiting linear heat generatiion rates were calculated for cycles 5 through 7 and may be seen in the coding for those versions of Subroutine LLHGR.

2.6.5 Subroutine EDSOME (Edits #25 to #29)

The purpose of Subroutine EDSOME is to edit a variety of power distribution data. There are five edits that can be generated by EDSOME. Each is discussed by edit number as follows:

Edit #25

Edit #25 consists of the integrated detector core power and comparison with the calorimetric power. The variables CORPOW and RATIO, which are the integrated detector power and ratio to calorimetric, respectively, were calculated in Subroutine REDUCE.

Edit #26

Edit #26 consists of the core normalized peak node heat rate PEAKN. PEAKN, derived from the peak node heat rate PEAK calculated in Subroutine REDUCE, is referred to as the core peaking factor and is given by:

```
PEAKN = PEAK/CALPOW*Z04.
```

PEAKN along with it's location is edited.

Also included in Edit #26 is the core total pin peaking factor F(Q). F(Q) is the maximum all pins nodal linear heat generation rate from array LHGRA. It is derived as:

```
TPF = LHGRA(i,j)*TL/CALPOW*1000.
where:
     TPF         = core total peaking factor
     LHGRA(i,j)  = maximum all pins linear heat generation rate
                   in the core (assembly i, axial node j).
     TL          = total rod length of all rods in core, in feet
     CALPOW      = total core power
     1000        = number of killowatts per megawatt
```

TPF is found and edited by Subroutine GETTPF which is not otherwise mentioned by this manual. Also, note that the equation part TL/CALPOW*1000. is the reciprocal of the core average linear heat rate in kw/ft.

Edits #27 and #28

These edits consists of the incore quadrant tilts (from TILT) and may contain excore tilt values if excore readings were input on card type #3. If excore detector readings were input, then excore tilts, and deviations between excore and incore tilts are edited. Also edited is the optional hand input tilt (or peaking factor) uncertainty UNUSER from card type #7. Edit #28 is produced by Subroutine EDTILT.

2.6.5 Subroutine EDSOME (continued)

Edit #29

Edit #29 consists of the incore average core axial power shape found in Subroutine RADIAL and stored in array ACAPS. The normalized axial shape values for each of the 25 core planes are edited. Also edited is a plot showing the PIDAL calculated axial power shape (ACAPS) and XTG power shape in array XTGAX from Subroutine EXPAND. The core average axial power shape is generated via Subroutine PLOTER, which is also not discussed further in this manual.

2.6.6 Subroutine EDPEAK (Edit #30)

The purpose of Subroutine EDPEAK is to edit the calculated assembly and pin peaking factor data calculated in Subroutine CPEAK. For the most part, EDPEAK is an editing routine, but a small amount of limit searching is done which must be explained.

In CPEAK, the maximum assembly peaking factor was saved by fuel type in array FFRA. It is desired to find the most limiting of these values based on allowed to measured assembly peaking factor ratios. Note that the allowed peaking factors may be different for each fuel type.

A loop is executed over the number of fuel types in the core. For each fuel type, the allowed peaking factor is computed per Ref #7 section 3.23.2. For cycles 5 through 7 the calculation was:

```
ALLOW = FRALIM(i)*(1.0 + 0.5*(1.0 - CALPOW/2530.))
```

where:

ALLOW	= allowed assembly radial peaking factor
FRALIM(i)	= assembly radial peaking factor limit for fuel type i declared in BLOCK DATA and specified by Ref #7
CALPOW	= core calorimetric power in MWth

For cycle 8 and beyond:

```
ALLOW = FRALIM(i)*(1.0 + 0.3*(1.0 - CALPOW/2530.))   for CALPOW >= 50%

ALLOW = FRALIM(i)*(1.15)                             for CALPOW <  50%
```

The measured assembly radial peaking factor is then divided into ALLOW resulting in the minimum allow to measured ratio for fuel type i. The minimum allowed to measured ratio and it's corresponding peaking factor, assembly power, assembly number and fuel type are then edited. Also, the minimum ratio and it's corresponding assembly radial peaking factor for the entire core is edited separately.

For cycles 5 through seven, the above procedure was repeated for the three types of pin peaking factors: all, inner and narrow water gap pins. For cycle 8 and beyond, the procedure was repeated for only the inner pins peaking factors.

2.6.7 Subroutine EDLHGR (Edit #31)

The purpose of Subroutine EDLHGR is to edit the calculated peak pin linear heat generation rates calculated and located in Subroutines LHGR and LLHGR, respectively.

EDLHGR is an editing routine only, with no significant calculations performed. In summary, the limiting linear heat generation rate, allowed to measured ratio, and location of the limiting node are edited by fuel type for the appropriate pin linear heat generation rates.

2.6.8 Subroutine CLIMIT (Edit #32)

The purpose of Subroutine CLIMIT is to calculate high alarm limit ratios for the incore detectors. The general idea is to find the limiting linear heat generation rate margins by axial detector level. The individual detector signals will then be multiplied by the appropriate margin in order to define an upper LHGR limit for that detectors signal to reach. The detector signal upper limits are transferred to the plant computer (PIP) which monitors all the detectors for abnormally high signals.

An underlying assumption employed in the determination of the detector high alarm limits (and later in the allowable power level calculation) is that the linear heat generation rate is most limiting at locations of relative axial heat rate peaks. Thus, alarm limits should be set based on linear heat rates at relative axial heat rate peak locations.

Subroutine CLIMIT is actually very simple. A loop is executed over each assembly in which relative power peaks within the assembly are identified. The identification of relative power peaks is performed by Subroutine FINDPK. After an assemblies relative peaks are identified, Subroutine ALARMF checks to see if the peak location coincidently had a limiting alarm limit ratio. If so, the limiting ratio is saved by axial detector level. After completion of the loop, the five limiting high alarm limit ratios (one for each detector level) are edited as EDIT #32.

The reader is directed to the discussions of Subroutines FINDPK and ALARMF for further detail.

2.6.9 Subroutine FINDPK

The purpose of Subroutine FINDPK is to locate relative axial power peaks within a given fuel assembly. Starting at the bottom of core, the all pins linear heat generation rate of a node is compared to the linear heat generation rates of the neighboring nodes. If that node is the site of a power peak, then a logical variable is set which corresponds to the peak nodes axial position. The search then continues up the assembly until top of core is reached.

As mentioned, the assembly number whose axial power peaks are to be determined is passed into FINDPK. The array LHGRA, which contains the axial all pins linear heat generation rate distribution for all 204 assemblies, is searched from bottom to top for the desired assembly.

The locations of the peak nodes in an assembly are returned to the calling routine via the logical array HEREPK. HEREPK has a dimension of 25, with each element corresponding to an axial node within the assembly (1=bottom of core and 25=top). Further description of this Subroutine is left to the coding.

2.6.10 Subroutine ALARMF

The purpose of Subroutine ALARMF is to axially scan an assembly and determine which nodal linear heat generation rates are closest to their limits by detector level. The most limiting values by detector level for all assemblies are then saved for use in calculating high incore alarm set points.

The procedure is to first look at relative linear heat generation rate peaks (determined by Subroutine FINDPK) for the bottom detector level in the assembly. If a relative peak location exists within the bottom level, then the most limiting peak pin (either all, inner or narrow water gap) linear heat generation rate from the location of the relative peak is saved. If no relative linear heat rate peak occurred in the bottom level, then the most limiting of the pin heat rates is determined from the location of the highest nodal all pins linear heat rate within the level.

The most limiting allowed to measured heat rate for the level and assembly is then compared with the values from the other bottom level assembly limiting heat rates in the core, with the most limiting value for the entire core being saved. This procedure is then repeated for the other four detector levels in the core.

The variables used for determing the most limiting peak pin linear heat generation rates are RATIOA, RATIOI and RATION for all, inner and narrow water gap pin types respectively. The peak all pins linear heat generation rates (for the case where no relative peaks occur in a given detector level) are from variable LHGRA. All four arrays were calculated in Subroutine LHGR.

The most limiting linear heat rates for the core by detector level are stored in array ALARM. The locations of the limiting nodes are stored in arrays IALARM and JALARM for the assembly and node numbers, respectively.

For the cycle 8 Technical Specification revisions, Subroutine ALARMF was modified to only consider all pins LHGR margin ratios.

2.6.11 Subroutine GETAPL (Edit #33)

The purpose of Subroutine GETAPL is to determine the allowable power level for continued reactor operation, based on the current core conditions, in the event that the incore monitoring system becomes inoperable. The calculation is performed per Technical Specification 3.11.2 (Ref #7).

The equation used for determining the allowable power level (APL) is given in Ref #7 as:

$$APL = \frac{LHR(z)_{T.S.}}{LHR(z)_{Max} * V(z) * E(z) * 1.02} * \text{Rated Power}$$

where:

$LHR(z)_{T.S.}$ = limiting LHR vs Core Height per Ref #7 Section 3.23.1

$LHR(z)_{Max}$ = measured peak LHR vs Core Height including uncertainties

$V(z)$ = axial variation bounding condition, Ref #7 Figure 3.11-1

$E_p(z)$ = factor to account for the reduction of allowed LHR in the peak rod with increased exposure (Figure 3.23-2) such that:

For fuel rod burnups less than 27.0 GWD/MT

$E_p = 1.0$

For fuel rod burnups greater than 27.0 GWD/MT but less than 33.0 GWD/MT

$E_p = 1.0 + .0064*LHR$

For fuel rod burnups greater than 33.0 GWD/MT

$E_p = 1.0 + .0012*LHR$

where LHR is the measured fuel rod average LHR in KW/FT.

For cycle 8 and beyond, $E_p(z)$ was omitted from the equation.

1.02 = an allowance factor for the effects of upburn. Upburn accounts for variation of the values used in the APL calculation over the 30 days which the APL may be applied.

The calculation is implemented in PIDAL as follows. A loop is executed over all assemblies in the core. For each easembly: the nodal average peak pin (based on all pins peaking) linear heat rate, burnup penalty factor FBE, and burnup penalty ESUBP are calculated. FBE and ESUBP were used only for cycles 5 through 7.

Subroutine FINDPK is then called to identify the location of relative axial all pin linear heat generation rate peaks within the assembly. Subroutine AXLAPL is then called, in order to determine if the allowable power level is limiting, with respect to all other assemblies, at the locations of the relative axial linear heat rate peaks.

Note that the APL calculation for cycles 5 through 7 was performed for all three pin peaking (all, inner and narrow water gap) types. For cycle 8 and beyond, only all pins was considered by AXLAPL. If a core limiting APL value is found, then the value, it's location and pin peaking type are saved. The loop over the remaining assemblies then continues.

2.6.11 Subroutine GETAPL (continued)

After the loop over all assemblies is completed, the limiting APL value is verified to be less than or equal to rated power. If the APL is found to be greater than 100 percent power then it is reset to 100 percent. The limiting APL is then edited along with the location of the limiting node and the type of pin peaking factor applied under Edit #33.

A couple of general comments. The APL equation above was for the cycles 5 through 7 all pins peaking factor case. For these cycles, the E_p term was not included for the inner and narrow water gap pins cases. It is uncertain exactly why this was. Also, FBE only appears in the all pins limit equation and appears to be accounting for the same burnup effect as E_p. Therefore, it is possible that a double hit was taken for burnup effects in the all pins case. The burnup effect accounts for fission gas releases in the fuel pin. It is desired to maintain the internal pin pressure below the moderator pressure.

2.6.12 Subroutine AXLAPL

The purpose of Subroutine AXLAPL is to check all pins linear heat generation rate relative axial locations within an assembly for limiting allowable power level (APL) values per Technical Specification (Ref #7) 3.11.2.

AXLAPL loops from the bottom node to top node for a single assembly. If a relative axial all pins linear heat rate peak (as determined by Subroutine FINDPK) occurred at a given node, then the APL calculation for all, inner and narrow water gap pin linear heat rates is performed. If one of the APL values for the node turns out to be the limiting value so far for the core, then that value, it's location and pin peaking type are saved. Various other paramters used in the APL calculation are also saved depending on pin peaking type. AXLAPL is called once for each assembly in the core by Subroutine GETAPL.

The equations used to determine the allowable power level for each pin peaking type are as follows:

For All Pins Peaking (cycles 5 through 7)

$$APL = \frac{ALOCAL(j)*FBE*AKWFTL(ifutyp)}{LHGRA(i,j)*VZ(j)*ESUBP*1.02} * 100.$$

For Inner Pins Peaking (cycles 5 through 7)

$$APL = \frac{ALOCIL(j)*IKWFTL(ifutyp)}{LHGRI(i,j)*VZ(j)*1.02} * 100.$$

For Narrow Water Gap Pins Peaking (cycles 5 through 7)

$$APL = \frac{ALOCIL(j)*NKWFTL(ifutyp)}{LHGRN(i,j)*VZ(j)*1.02} * 100.$$

For All Pins Peaking (cycle 8 and beyond)

$$APL = \frac{ALOCAL(j)*AKWFTL(ifutyp)}{LHGRA(i,j)*VZ(j)*1.02} * 100.$$

See Table #2.6.13 for a description of the variables used in the previous equations.

Variable	Type	Description
HEREPK	LOGICAL HEREPK(25)	location of relative all pins LHGR peaks by axial node from Subroutine FINDPK
ALOCAL	REAL ALOCAL(25)	allowable LHR as a function of peak power location. Tech Spec Figure 3.23-1
FBE	REAL FBE	allowable LHR as a function of burnup from Subroutine GETFBE. Tech Spec Figure 3.23-2
ALOCIL	REAL ALOCIL(25)	allowable LHR as a function of peak power location. Tech Spec Figure 3.23-3
VZ	REAL VZ(25)	axial variation bounding condition for APL. Tech Spec Figure 3.11-1
ESUBP	REAL ESUBP	LHR reduction factor for peak rod versus exposure. Tech Spec 3.11.2
AKWFTL	REAL AKWFTL(9)	allowed LHR by fuel type for all pins Tech Spec Table 3.23-1. Set in BLOCK DATA
IKWFTL	REAL IKWFTL(9)	allowed LHR by fuel type for inner pins Tech Spec Table 3.23-1. Set in BLOCK DATA
NKWFTL	REAL NKWFTL(9)	allowed LHR by fuel type for narrow water gap pins. Tech Spec Table 3.23-1
LHGRA	REAL LHGRA(204,25)	measured all pins linear heat rate from Subroutine LHGR
LHGRI	REAL LHGRI(204,25)	measured inner pins linear heat rate from Subroutine LHGR
LHGRN	REAL LHGRN(204,25)	measured narrow water gap pins linear heat rate from Subroutine LHGR
APLMIN	REAL APLMIN	minimum allowable power level in percent reactor power for entire core
IAPL	INTEGER IAPL	assembly number of APLMIN location
JAPL	INTEGER JAPL	axial node number of APLMIN location
APLV	REAL APLV	axial variation bounding condition for minimum allowable power level, APLMIN
APLE	REAL APLE	ESUBP for minimum APL all pins only
APLF	REAL APLF	FBE for minimum APL all pins only
APLB	REAL APLB	peak pin exposure for minimum APL all pins
APLLHR	REAL APLLHR	peak pin lhr for minimum APL all pins
APLTST	REAL APLTST	type of peak pin for minimum APL -3=narrow water gap +3=inner 0=all

Table #2.6.13 Variables used in the Allowable Power Level Calculation

2.6.13 Subroutine PEAKKW (Edits #34, #35 and #36)

The purpose of Subroutine PEAKKW is to generate full core radial maps of the axially collapsed assembly peak pin linear heat generation rates in kilowatts. These maps appear under edits #34, #35 and #36 for all, inner and narrow water gap peak pin assembly powers respectively.

No calculations are performed in PEAKKW. The only operation is to determine which of the three edits are required and to call Subroutine EDTRAD which performs the actual editing. The three arrays passed to EDTRAD are RODKWA, RODKWI and RODKWN for all, inner and narrow water gap peak pin powers. The arrays were calculated in Subroutine CPEAK.

2.6.14 Subroutine PKKWFT (Edits #37, #38 and #39)

The purpose of Subroutine PKKWFT is to find and edit the peak axial nodal linear heat generation rates for each assembly. The linear heat rates are found for all, inner and narrow water gap pins and are edited under Edits #37, #38 and #39 respectively.

For each assembly, an axial scan is performed and the peak node for all, inner and narrow water gap pin linear heat rate is found. The arrays searched are LHGRA, LHGRI and LHGRN which were calculated in Subroutine LHGR.

PKKWFT then determines which of the three edits are required and passes the linear heat rate data to Subroutine EDTRAD for editing. No calculations are performed.

2.6.15 Subroutine EDN040 (Edit #40)

The purpose of Subroutine EDN040 is to produce edit #40 which consists of the entire linear heat generation rate and allowed to measured linear heat rate arrays. This edit was created for debugging the Tech Spec linear heat rate calculations. Needless to say, this edits produces lots of output.

The order of editing is as follows:

Array LHGRA	all pins lhgr in kw/ft from Subroutine LHGR
Array RATIOA	all pins allowed to measured linear heat rate ratios
Array LHGRI	inner pins lhgr in kw/ft from Subroutine LHGR
Array RATIOI	inner pins allowed to measured linear heat rate ratios
Array LHGRN	inner pins lhgr in kw/ft from Subroutine LHGR
Array RATION	inner pins allowed to measured linear heat rate ratios

Each of the arrays are edited by assembly (204) and axial node (25), 21 assemblies are edited per page for 30 pages of total output. For cycle 8 and beyond, this edit only produces the all pins maps.

2.7 Program Section #7

The purpose of program section #7 is to calculate the current fuel and control rod exposures, calculate the high incore detector alarm set points and edit various summary data for the PIDAL case.

The following sections discuss each of the subroutines within program section #7 in detail.

2.7.1 Subroutine EXPOZF (Edit #41)

The purpose of Subroutine EXPOZF is to calculate the current nodal fuel exposures. The total core, axially collapsed assembly, and batch average exposures are also calculated, along with the fuel batch power fractions for the current case. The following are the equations utilized.

Core energy for this case

PEFF = DT total energy to be distributed over the core for the current case in MWhr(thermal) from input card #6

DT = DT/(CALPOW*24.) effective full power days for the case based on the input calorimetric power

Batch power fractions

PA(n) = PA(n) + FULL2D(i) i=1 to 204 assemblies
where:
PA(n) = total power in MWth for fuel type n
FULL2D(i) = integrated power for assembly i in MWth from Subroutine RDLAXL
n = fuel type number for assembly i

PA(n) = PA(n)/CALPOW n=1 to number to fuel types in core (9 maximum)
where:
PA(n) = power fraction in batch n
CALPOW = calorimetric total core power in MWth

Nodal Fuel Exposure Distribution

FUEXP(i,j) = FUEXP(i,j) + F3D25(i,j)*DT*25./AMTU(n)
where:
FUEXP(i,j) = nodal fuel exposure from last case updated to current in MWD/MTU
F3D25(i,j) = nodal power in MWth from Subroutine REDUCE
DT = effective full power days for this exposure case calculated above
AMTU(n) = total heavy metal mass for assembly type n in MTU
n = fuel type number for assembly i
25/AMTU = division by the fuel mass in the axial region covered by node j

Axially Collapsed Assembly Average Exposure

EP(i) = EP(i) + FUEXP(i,j) j=1 to 25
where:
EP(i) = integrated exposure for assembly i in MWD/MTU
FUEXP(i,j) = nodal exposure for assembly i and axial node j in MWD/MTU

EP(i) = EP(i)/25. i=1 to 204
where:
EP(i) = nodal average exposure for assembly i

2.7.1 Subroutine EXPOZF (continued)

Batch Average Exposure

EBATCH(n) = EBATCH(n) + EP(i)*AMTU(n) i=1 to 204
where:
EBATCH(n) = total batch energy distributed for fuel type n in MWD
EP(i) = assembly exposure in MWD/MTU
AMTU(n) = mass for fuel type n (assembly i) in MTU

EA(n) = EBATCH(n)/(AMTU(n)*COUNT(n))
where:
EA(n) = batch average exposure for fuel type n in MWD/MTU
COUNT(n) = number of assemblies of fuel type n in the core

Total Core Exposure

ETOTAL = ETOTAL + EBATCH(n) n=1 to number of fuel types (9 maximum)
MASTOT = MASTOT + AMTU(n)*COUNT(n) n=1 to number of fuel types
where:
ETOTAL = total core energy distributed in MWD
MASTOT = total core mass in MTU
EBATCH(n) = total batch energy distributed for fuel type n in MWD
AMTU(n) = mass for fuel type n (assembly i) in MTU
COUNT(n) = number of assemblies of fuel type n in the core

TOTEP = ETOTAL/MASTOT
where:
TOTEP = current core average assembly exposure in MWD/MTU
(not cycle average assembly exposure)

It should be noted that the batch average fuel exposures are further broken down by reload fuel types. i.e. I-FUEL - I + ISP + IGAD. This coding is cycle dependent and will not be discussed herein. See the source listing for a description.

The axially collapsed assembly exposures (array EP) are then edited under Edit #41. This is done by passing the array EP to Subroutine EDFEXP. EDFEXP performs no calculations and will not be discussed further. An edit of the octant collapsed exposure distribtuion (Edit #42) may also be produced via Subroutine OCTEXP. See section 2.3.1.

2.7.2 Subroutine EXPOZR (Edit #43)

The purpose of Subroutine EXPOZR is to calculate the total accumulated exposure distributed to the control rods in MWD. The rod exposures are calculated by determining which fuel nodes were shadowed by rods, and then using the power seen in those nodes to the adjacent control rod exposure calculation. The equation used for calculating rod exposures is:

$$RODEXP(i,L) = RODEXP(i,L) + (F3D25(ROD(1,i),j) + F3D25(ROD(2,i),j) + F3D25(ROD(3,i),j) + F3D25(ROD(4,i),j))*DT/4$$

where:

- RODEXP(i,L) = nodal control rod exposure from last case updated to current in MWD. rod i, axial node L i=1 to 45, L=1 to (Z5-K+1)
- F3D25(ROD(1,i),j) = nodal power in MWth for assembly ROD(1,i) and axial node j, j=K to 25
- ROD(1,i) = number of the assembly to the north-west of rod i
- ROD(2,i) = number of the assembly to the north-east of rod i
- ROD(3,i) = number of the assembly to the south-west of rod i
- ROD(4,i) = number of the assembly to the south-east of rod i
- K = H(i)/HA*25. + 1
- H(i) = rod position in cm withdrawn for rod i
- HA = 335.28 cm (active rod length)

The integral exposure for each control rod is then calculated as:

$$SUMEXP(i) = SUMEXP(i) + RODEXP(i,j) \quad j=1 \text{ to } 25$$

where:

- SUMEXP(i) = integral control rod exposure for rod i in MWD
- RODEXP(i,j) = nodal control rod exposure for rod i, node j

For the above equations, the rods are indexed by PIDAL's internal storage scheme which is different from the plants control rod indexing scheme. See Ref #2 Attachment #2 for the control rod indexing conversion table.

The accumulated control rod exposures may then be edited under Edit #43. Included in the edit are: the PIDAL control rod number, the plants control rod drive index number, the current position of the rod in inches from the core bottom, the integrated control rod exposure, the maximum control rod exposure and the location of the maximum exposure.

It should be noted that this author is not entirely convinced that the rod exposures are correct since some of the rods may have been shuffled or replaced with no modifications made to the PIDAL values. Also note that since the rod positions came from an instantaneous PIP data dump, the rod configuration of the dump may not have been representative of the entire exposure period.

2.7.3 Subroutine CORDAT (Edits #44 and #45)

Subroutine CORDAT is an editing routine which prints out core exposure and power distribution summaries. The summary consists of: total core and cycle average exposure, current core power and axial offset, fuel batch average exposures and powers, and the collapsed octant core PIDAL and XTG power distributions and percent deviations. The power distributions are actually produced by a call to Subroutine OCTANT.

The only calculation performed in CORDAT is the cycle core average exposure, using the following equation:

CYC = TOTEP - BOCEXP

where:

CYC = cycle core average exposure in MWD/MTU

TOTEP = total core average exposure in MWD/MTU from Subroutine EXPOZF

BOCEXP = total core average exposure at beginning of current cycle in MWD/MTU from Subroutine GETEXP

The edits are self-explanatory so no further explanation will be given. Subroutine CHKLIB is also called from CORDAT. CHKLIB determines how much longer the current W' library will be valid. See Section 2.7.5 for further discussion of Subroutine CHKLIB.

2.7.4 Subroutine OCTANT (Edit #46)

Subroutine OCTANT is an editing routine which prints out the axially collapsed PIDAL and XTG normalized power distributions and percent deviations between the two. The PIDAL octant power distribution is generated by collapsing the array FULNZD (from Subroutine RADIAL) from full core to 1/8th core. Two methods were developed for this purpose, but only one was employed in this version of the code. The following discussion describes these methods.

The old method simply collapsed symmetric box powers in the full core to an average octant representation. Theoretically this is a valid approach, but by examining the technique used for determining assembly powers in non-incore instrumented locations, it was realized that the resultant octant distribution was greatly biased by the theoretical power distribution supplied by XTG.

The new method employed consisted of collapsing only instrumented detector locations to an octant core. In other words, if an assembly contained even one operable incore detector, then the box power for that assembly would be used in the octant power calculation. For octant locations which contained no operable incores, then an average of all symmetric full core locations would be used as the power for that octant location. The reader may note that this method results in an octant solution which should be essentially the same as the solution determined by the original 1/8th core PIDAL model. It is the new method which was employed in the current version of the full core model.

The 1/8th core XTG power distribution used by Subroutine OCTANT is from array XTGS (subroutine EXPAND). The percent deviation between PIDAL and XTG octant average locations is determined as:

$$DEV(i) = \frac{XTG(i) - FXY(i)}{FXY(i)} * 100\% \quad i=1 \text{ to } 28$$

where:

DEV(i) = percent relative deviation between XTG and PIDAL for octant location i.
XTG(i) = XTG normalized power for octant location i.
FXY(i) = PIDAL normalized power for octant location i.

From knowing the DEV(i) values for the octant, the root mean square deviation between the two power distributions is calculated as:

$$RMS = \sqrt{\frac{\sum_{i=1}^{28} DEV(i)^2}{28.}}$$

For the cycle 8 version of the code, the RMS without H-stainless steel bundles included in the calculation is also edited. This was due to problems early in cycle 8 with the H-ss Wprimes.

The RMS deviation is saved later on the second record of the PIDAL exposure file. Subroutine OCTANT produces Edit #46.

2.7.5 Subroutine CHKLIB (Edit #47)

The purpose of Subroutine CHKLIB is to determine how many effective full power operating days worth of data remain in the PIDAL W' library. First, the average power seen by each incore detector is found using the following equation:

A(i,L) = (F3D25(i,j-1) + F3D25(i,j) + F3D25(i,j+1))/3.*25 L=1 to 5

where:

A(i,L) = power seen by detector L (1 to 5) of assembly i in MWth
F3D25(i,j) = power for assembly i node j in MWth from Subroutine REDUCE
j = (L*5) - 2
L = detector level

Then, the remaining effective full power days left in the W' library for each detector is calculated as:

EX = DELEXP(IOCT,L)*AMTU(n)/A(i,L)*CALPOW/2530.

where:

EX = effective full power days of W' library data remaining for detector L of assembly i
DELEXP(IOCT,L) = exposure remaining in library for octant location ioct and level L from Subroutine WCALC, MWD/MTU
AMTU(n) = heavy metal mass of assembly i which is of fuel type n, MTU
A(i,L) = power seen by detector L of assembly i in MWth
CALPOW = calorimetric core power in MWth
2530 = rated reactor power in MWth

For the cycle 8 version of the code, the H stainless steel assemblies were not included in the CHKLIB calculations due to problems early in cycle 8 with the H-ss Wprimes.

The minimum value of EX for the entire core is saved and it's location and value are edited as EDIT #47.

2.7.6 Subroutine PUTEXP (Edit #48)

The purpose of Subroutine PUTEXP is to write the records for the current case onto the PIDAL exposure file. For an exposure case, four records are written whereas for a power distribution case, only two records are written.

The first record contains restart identification data for the case such as run date, burnup dates, the case number, reactor power and effective full power days for the case. The second record contains the 3-D 204 assembly by 25 axial nodes power distribution, followed by 80 trend data elements.

The third and fourth records, written only for cases where exposure is applied to the core contain fuel and rod exposure data, PIDAL detector sensitivities and XTG restart data. All XTG data is on the fourth record. The reader should consult Table #2.1.15 for a detailed description of the data on the PIDAL exposure file.

2.7.7 Subroutine EXMON (Edit #49)

The purpose of Subroutine EXMON is to provide an edit of the excore monitoring systems allowable power level and target axial shape with respect to Technical Specification (Ref #7) 3.11.2. This edit (#49) is intended for use by the control room operators and is only produced if excore detector information is input to PIDAL via card type #3.

The allowable power level edited was determined by Subroutine GETAPL. No further explanations are required.

2.7.8 Subroutine CALARM (Edit #50)

The purpose of Subroutine CALARM is to apply the incore detector high alarm limit ratios calculated in Subroutine CLIMIT to the measured detector signals, resulting in new incore high alarm limits. CALARM also calls Subroutine PTAPE which writes the alarm limits to an external device and calls Subroutine TSSUMM which generates a summary of Technical Specification parameters monitored by PIDAL.

The following equations are used for determining the incore detector alarms.

The incore detector readings in flux units (nv) are calculated as:

$$READNV = \frac{E(i,j)}{KSUBF(i,j)}$$

The raw millivolt detector signals are backed out using:

$$EIJ = E(i,j) * \frac{KSUBS(i,j)}{KSUBB(i,j)}$$

The incore detector high alarm in flux units are calculated as:

$$ALMNV = READNV*ALARM(j)$$

Finally, the incore detector high alarm in millivolts is calculated as:

$$ALMMV = EIJ*ALARM(j)$$

Table #2.7.8 contains a list of the variables used above and definitions for each.

If an incores detectors ALMNV value is found to be less than or equal to zero, then that value is set to 870.0 which indicates that the detector is failed. The incore detector high alarm limit values are then converted to units required by the Varian Datalogger (PIP) and stored for later use:

$$ALARM1(jj,i) = ALMNV/.2125 + .5$$

where:

ALARM1(jj,i) = alarm limit in PIP units for detector jj of string i. For the above equations jj is the detector number in PIP notation (1=top) and j is in PIDAL notation (5 = top), j=6-jj. The .5 is for roundoff error.

The calculated alarm limits along with the detector signal data for each detector are edited in PIP notation under edit #50. The equation used for the conversion, comes from Ref #3, Appendix A, INDEX #7. When using the reference equation, note that ALMNV is Y and ALARM1 is X. Also note the one digit decimal positioning. Therefore the reference equation becomes:

$$Y = 0.53125 * 2 * X * 10 = 0.2125 * X$$

2.7.8 Subroutine CALARM (continued)

If an alarm limit output is desired for input to the PIP, then Subroutine PTAPE is called. PTAPE is called if the input variable TAPPUN (card #7) is either set as TAPPUN=1 or if TAPPUN=0 and the case was a power distribution case (DT=0). If TAPPUN is neither one or zero then PTAPE is not called. See Section 2.7.9 for additional discussion of Subroutine PTAPE.

Subroutine TSSUMM is always called to summarize the Technical Specifications monitored by PIDAL. Section 2.9.10 summarizes TSSUMM.

Variable	Type	Description
READNV	REAL READNV	incore detector reading in flux (nv) units
EIJ	REAL EIJ	incore detector reading in millivolts not corrected for sensitivity or background
ALMNV	REAL ALMNV	incore detector high alarm in flux (nv) units
ALMMV	REAL ALMMV	incore detector high alarm in millivolts
E(i,j)	REAL E(45,5)	depletion and background corrected detector millivolt signals from Subroutine INVERT
KSUBF	REAL KSUBF(45,5)	detector flux scaling factor, mv/nv
KSUBS	REAL KSUBF(45,5)	detector sensitivity factor, dimensionless
KSUBB	REAL KSUBF(45,5)	detector background noise correction factor, dimensionless
ALARM(j)	REAL ALARM(5)	incore detector high alarm limit ratios from Subroutine CLIMIT

Table #2.7.8 Variables used in the incore detector high alarm limit calculations

2.7.9 Subroutine PTAPE

The purpose of Subroutine PTAPE is to write incore detector high alarm limit information to an external device. The data is written in a form suitable for input to the PIDAL support program P527102P, which formats the alarms for punching on a punch paper tape that the PIP can read.

PTAPE is strictly an editing routine with no calculations performed. The alarm limit data is edited in a human readable form, eight detectors (not strings) per line. Each line begins with the PIP's starting address for the eight alarm limits that follow. The alarm limit data is preceeded and followed by the PIDAL case run date and the date and time of the PIP data dump that was used as input to the PIDAL case. Further detail on the output format is left to the coding.

2.7.10 Subroutine TSSUMM (Edit #51)

The purpose of Suborutine TSSUMM is to generate a summary of all of the Technical Specification monitoring done by PIDAL. All of the data included in the summary already exists on another edit and only minor calculations are performed. The Technical Specification Summary occurrs as Edit #51. The The following table list all of the variables edited and where they were calculated in the code.

TABLE 2.7.10

Variable	Originating Subroutine	Description
TITLE	GENTIT	PIDAL case title for current case
NDETL	CALARM	number of operable incores by detector level and quadrant
NDETQ	CALARM	number of operable incores by quadrant
NDETC	CALARM	number of operable incores in the core
AO	TILTAO	incore axial offset
EXAO	TSSUMM	incore/excore axial offset deviation, cycles 5-7 EXAO = ABS(2.0*XCPR - AO)
XCPR	card #3a	excore power ratio recorder reading, cycles 5-7
ASI	card #3b	excore axial shape indices, cycle 8 and beyond
ASIDIF	TSSUMM	incore/excore ASI deviations, cycle 8 and beyond ASIDIF(i) = ABS(ASI(i) - AO)
APLMIN	GETAPL	allowable power level for excore LHGR monitoring
AQUAD	TSSUMM	AQUAD(i) = (QTILT(i) - 1.0)*100.0
AZCOR	TSSUMM	AZCOR(i) = (ZCORE(i) - 1.0)*100.0
DIFTLT	TSSUMM	Absolute value of AQUAD(i) - AZCOR(i)
QTILT	TILTAO	Incore tilts for all four quadrants
ZCORE	EDTILT	Excore tilts for all four quadrants
ALLOWR	EDPEAK	allowed peaking factor for limiting assembly radial peaking factor
PEAKR	EDPEAK	measured limiting assembly radial peaking factor
ALLOWA	EDPEAK	allowed peaking factor for limiting all pins total radial peaking factor
PEAKA	EDPEAK	measured limiting all pins total radial peaking factor
ALLOWI	EDPEAK	allowed peaking factor for limiting inner pins total radial peaking factor, cycles 5-7
PEAKI	EDPEAK	measured limiting inner pins total radial peaking factor, cycles 5-7
ALLOWN	EDPEAK	allowed peaking factor for limiting narrow water gap pins total radial peaking factor, cycles 5-7
PEAKN	EDPEAK	measured limiting narrow water gap pins total radial peaking factor, cycles 5-7

Table #2.7.10 Variables appearing on the Technical Specification Summary Sheet, Edit #51.

2.7.11 Subroutine BDZOIK (Edit #A1)

The purpose of Subroutine BDZOIK is to allow PIDAL to re-solve for the full core power distribution by alternately failing each operable incore detector string in the core. Operation of BDZOIK is necessary for determining the uncertainties associated with inferring detector powers in uninstrumented core locations. BDZOIK may be called after a base power distribution has been determined by setting input flag IOUT on input card #2.

BDZOIK closely resembles Subroutine PIDAL in that it obtains the radial power distribution for all five detector levels using the same method as described by Subroutine PIDAL and its satellite routines. BDZOIK does not solve for the fine mesh power distribution but rather just the radial distributions at each level. In addition, BDZOIK along with its satellite routine BDEDIT generates an output file which may be used for statistical analysis' on the effect of failing individual incore detectors. The logic for BDZOIK is described as follows.

BDZOIK first saves the base case PIDAL power distribution for future statistical use. The full core fine mesh power distribution stored in variable F3D25 by subroutine REDUCE is written to a binary output file. Also, variable POWMAP which contained the base case radial power distributions for the five detector levels (stored by subroutine SOLVES) is copied into variable POWOLD.

A loop over all incore strings is then initiated. For each incore string which has all five of its incores operable, all five are failed and the radial power distribution is re-solved for each detector level. Note that only one incore detector string is failed at a time and there may be up to 45 redeterminations of the radial power distributions (one for each possible incore string to be failed). After an incore string has been failed and the power distribution redetermined, the five inferred detector powers for the string which was failed are stored in variable POWNEW. The failed incores are then reset with their original powers (un-failed) and the loop goes on to the next incore string. Along the way, Edit #A1 is produced informing the user of each recalculation of the power distribution and which incores were failed. All of the other PIDAL edits are turned off.

It should be noted that if an incore string is failed which would cause the quadrant tilt calculation routine to fail, (not enough operable symmetric detectors) then that incore is reactivated and the power distribution is not determined for it in a failed state.

After all possible incores have been failed and the power distribution redetermined, then another record to the previous output file is written. This record contains identifying information for the case run and also the original and inferred powers for each of the incore strings failed. Routine BDEDIT is then called to perform some preliminary statistical calculations.

2.7.12 Subroutine BDEDIT (Edits #A2 and #A3)

Subroutine BDEDIT is a post processor to subroutine BDZOIK which calculates the deviations between the original and inferred detector powers for each incore detector BDZOIK failed. The following equations are used.

$$DEV(i,j) = \frac{POWNEW(i,j) - POWOLD(i,j)}{POWOLD(i,j)} * 100\% \qquad i=1,45 \quad j=1,5$$

where:

DEV(i,j) = relative percent deviation between measured and inferred power for detector string i, detector j
POWNEW(i,j) = inferred power for detector string i, detector j
POWOLD(i,j) = measured (original) power for detector string i, detector j

The average deviation and RMS deviation are also calculated as:

$$AVE = \frac{\sum_{i,j} DEV(i,j)}{N}$$

where:

AVE = average inferred detector power deviation
DEV(i,j) = relative percent deviation between measured and inferred power for detector string i, detector j
N = number of detectors for which inferred powers were calculated

$$RMS = \sqrt{\frac{\sum_{i,j} DEV(i,j)^2}{N}}$$

where:

RMS = root mean square inferred detector power deviation
DEV(i,j) = relative percent deviation between measured and inferred power for detector string i, detector j
N = number of detectors for which inferred powers were calculated

Two edits are then produced. Edit #A2 reports the values of all of the inferred detector powers. These may be compared with the measured (original) detector powers given in Edit #Z2 of the base case. Edit #A3 reports the inferred to measured detector power deviations as defined above. The average and RMS deviations are also produced.

2.7.12 Subroutine RECALC

The purpose of Suborutine RECALC is to re-solve the full core power distribution using XTG calculated detector powers. No measured detector powers are used. This option is also for use in performing statistical analyses on the PIDAL methodology and is activated by input variable IOUT on card #2 along with subroutine BDZOIK.

RECALC is identical to PIDAL except that flag IXPOW, which forces PIDAL to use XTG detector powers is automatically set. RECALC calculates the radial power distributions for the five detector levels and also determines the fine mesh axial power distributions based on the XTG detector powers. RECALC produces Edit #A4 to inform the user that XTG detector powers are being used and then runs through the entire PIDAL run sequence, producing all desired edits. Upon completion, RECALC writes the original XTG calculated power distribution and the PIDAL distribution based on XTG detector powers to the statistical output file created by BDZOIK and BDEDIT. In theory these power distributions should be identical and a post processor is used to determine the synthesis errors in the XTG based PIDAL power distribution.

LIST of TABLES

2.0.1 —— Variables read from PIDAL case card image input
2.0.2 —— Variables read from Varian Datalogger UNTRANS data file
2.1.2 —— Variables read in from the Boron data file
2.1.15—— Variables returned from Subroutine GETEXP
2.1.16—— Variables used in Function CONINT
2.1.20—— Variables used in Subroutine RBANK
2.2.1 —— Variables used in Subroutine EXPAND
2.2.2 —— Variables used in Subroutine EXTRAP
2.2.3 —— Variables used in Subroutine ERROR
2.3.2 —— Variables used in Subroutine WCALC
2.3.3 —— Variables used in Function XPINT
2.3.4 —— Variables used in Subroutine COFCAL
2.3.5 —— Variables used in Function XINT
2.3.6 —— Variables used in Subroutine BXPWR
2.6.13—— Variables used in Subroutine AXLAPL
2.7.8 —— Variables used in Subroutine CALARM
2.7.10—— Variables appearing on the Technical Specification Summary sheet (Edit #51).

LIST of FIGURES

1.1 —— Data flow paths and major sections of the PIDAL main program
2.2.1 —— PIDAL and XTG detector and axial node locations
2.4.3a — 1/4 Core Reflective Symmetric Incore Detector Strings
2.4.3b — 1/4 Core Rotational Symmetric Incore Detector Strings
2.4.6a — 1/4 Core Reflective Quadrant Numbering Schemes
2.4.6b — 1/4 Core Rotational Quadrant Numbering Schemes
2.5.1 —— Graphical Representation of Equations Defined to Predict the Axial Power Distribution in the Core.

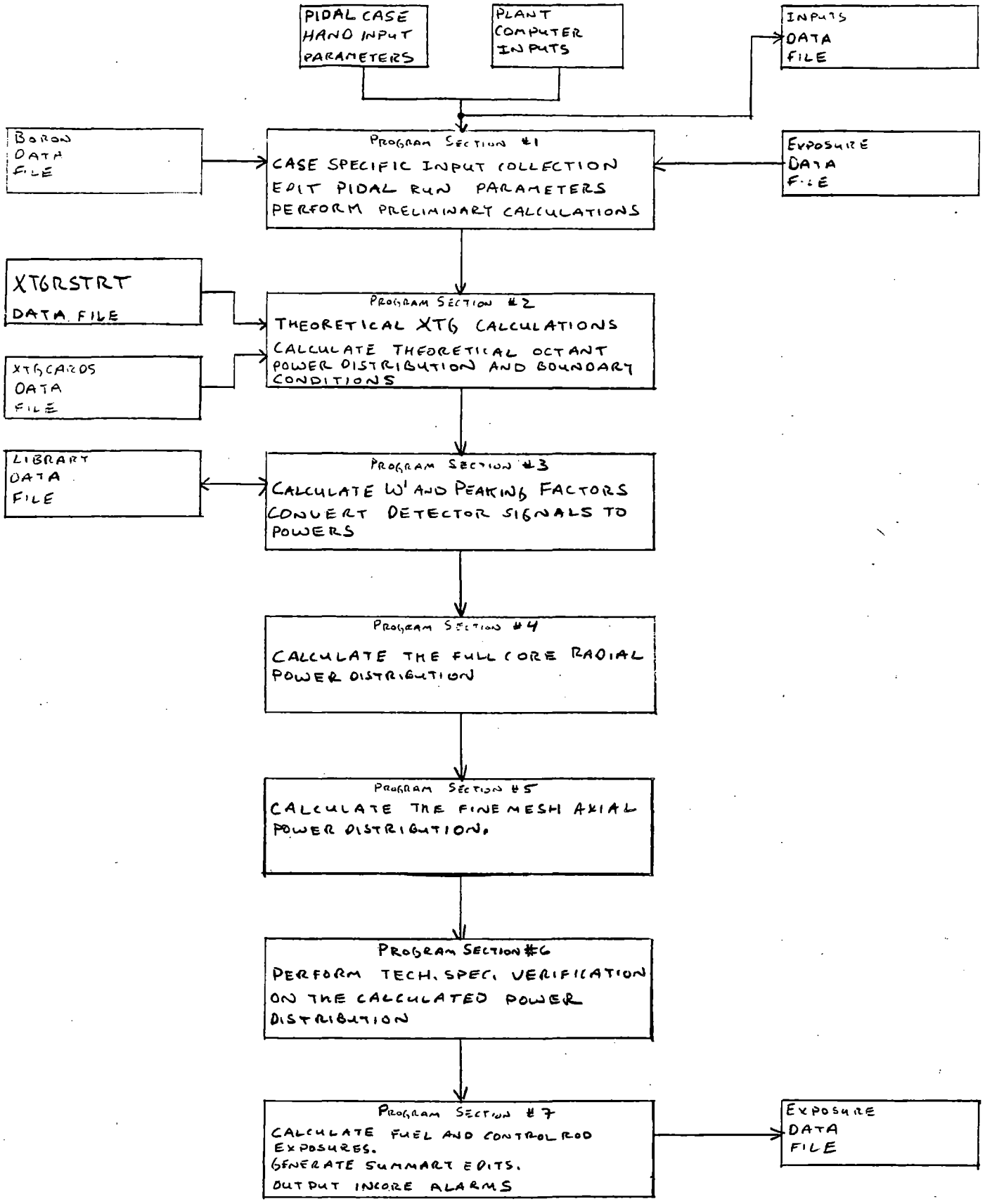


FIGURE # 1.1 DATA FLOW PATHS AND MAJOR SECTIONS OF THE PIDAL MAIN PROGRAM

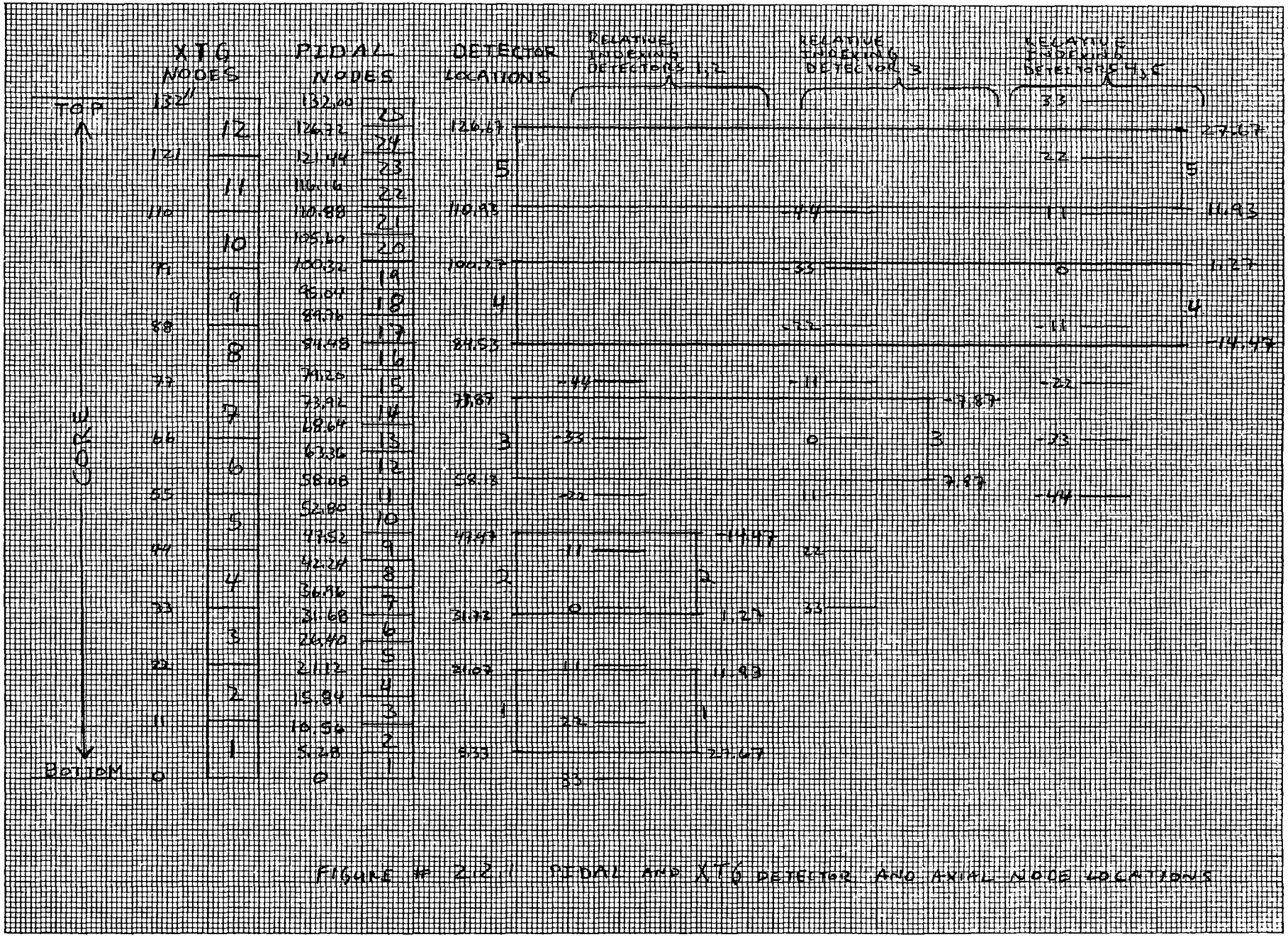


FIGURE # 2.2.1 PIDAL AND XTG DETECTOR AND AXIAL NODE LOCATIONS

BEE M/M

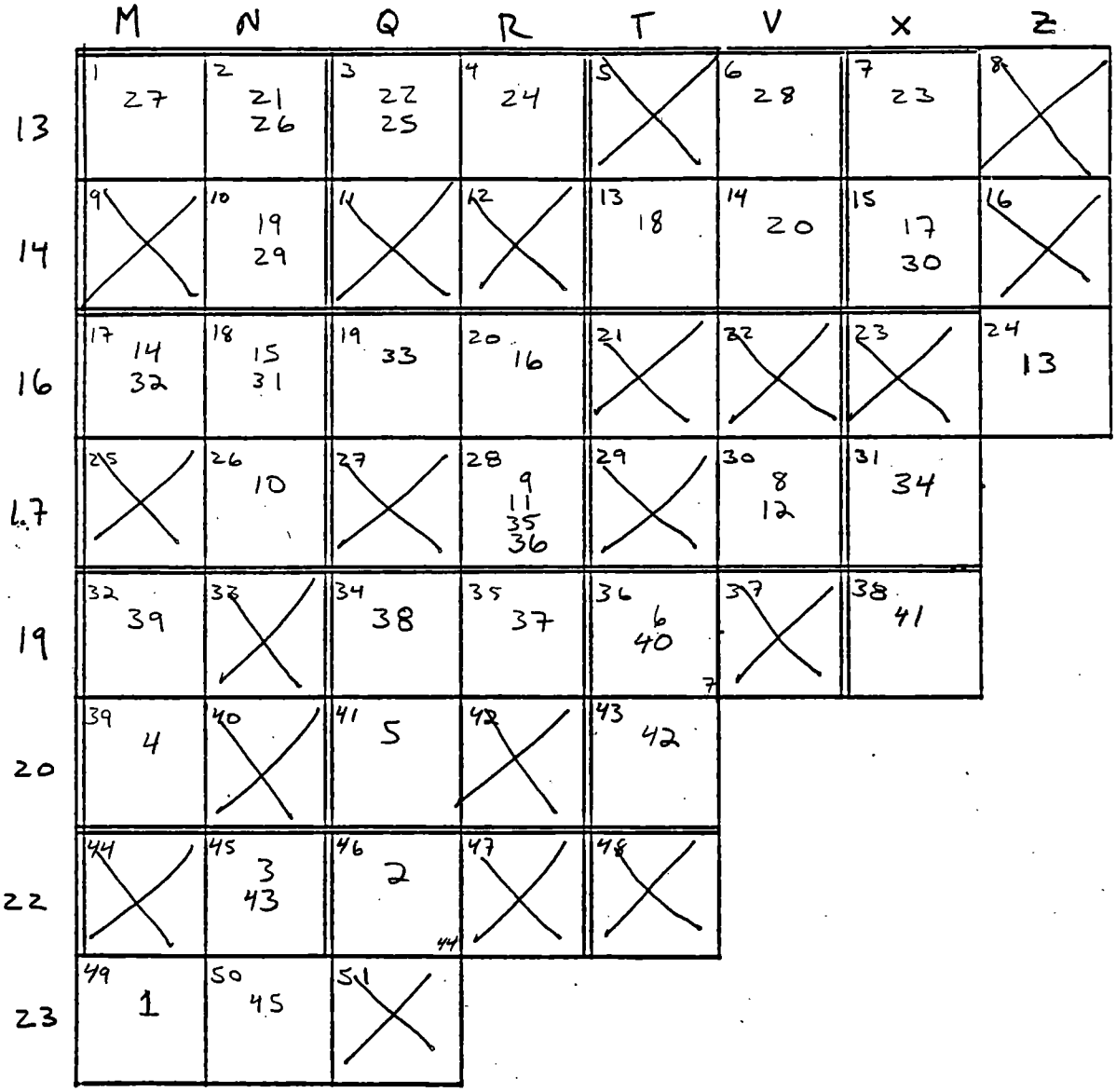
CONSUMERS POWER COMPANY
NUCLEAR ACTIVITIES DEPT.
REACTOR PHYSICS GROUP

SHEET_____ OF_____
DATE_____
BY_____
PLANT - PALISADES

FIGURE 2.4.3a

SUBJECT 1/4 CORE REFLECTIVE SYMMETRIC INCORE STRINGS

20 LOCATIONS UNINSTRUMENTED (39%)
1 4-WAY SET
9 2-WAY SETS



CONSUMERS POWER COMPANY
NUCLEAR ACTIVITIES DEPT.
REACTOR PHYSICS GROUP

SHEET_____ OF_____
DATE_____
BY_____
PLANT - PALISADES

FIGURE 2.4.3b

SUBJECT 1/4 CORE ROTATIONAL SYMMETRIC INCORE STRINGS

21 LOCATION UNINSTRUMENTED (41%)
3 4-WAY SETS
4 2-WAY SETS

1 27	2 X	3 X	4 X	5 X	6 28	7 X	8 1
9 21 26	10 19 29	11 15 31	12 X	13 18	14 X	15 3 17 30 43	16 X
17 14 22 25 32	18 X	19 33	20 X	21 38	22 5	23 X	24 13
25 24	26 10	27 16	28 9 11 35 36	29 37	30 8	31 X	
32 39	33 X	34 X	35 X	36 6 40 7	37 42	38 41	
39 4	40 20	41 X	42 12	43 X			
44 23	45 X	46 2 44	47 34	48 X			
49 X	50 45	51 X					

1

FIGURE 2.4.6a

QUADRANT NUMBERING - 1/4 CORE REFLECTIVE

PALISADES PLANT–REACTOR CORE PLAN

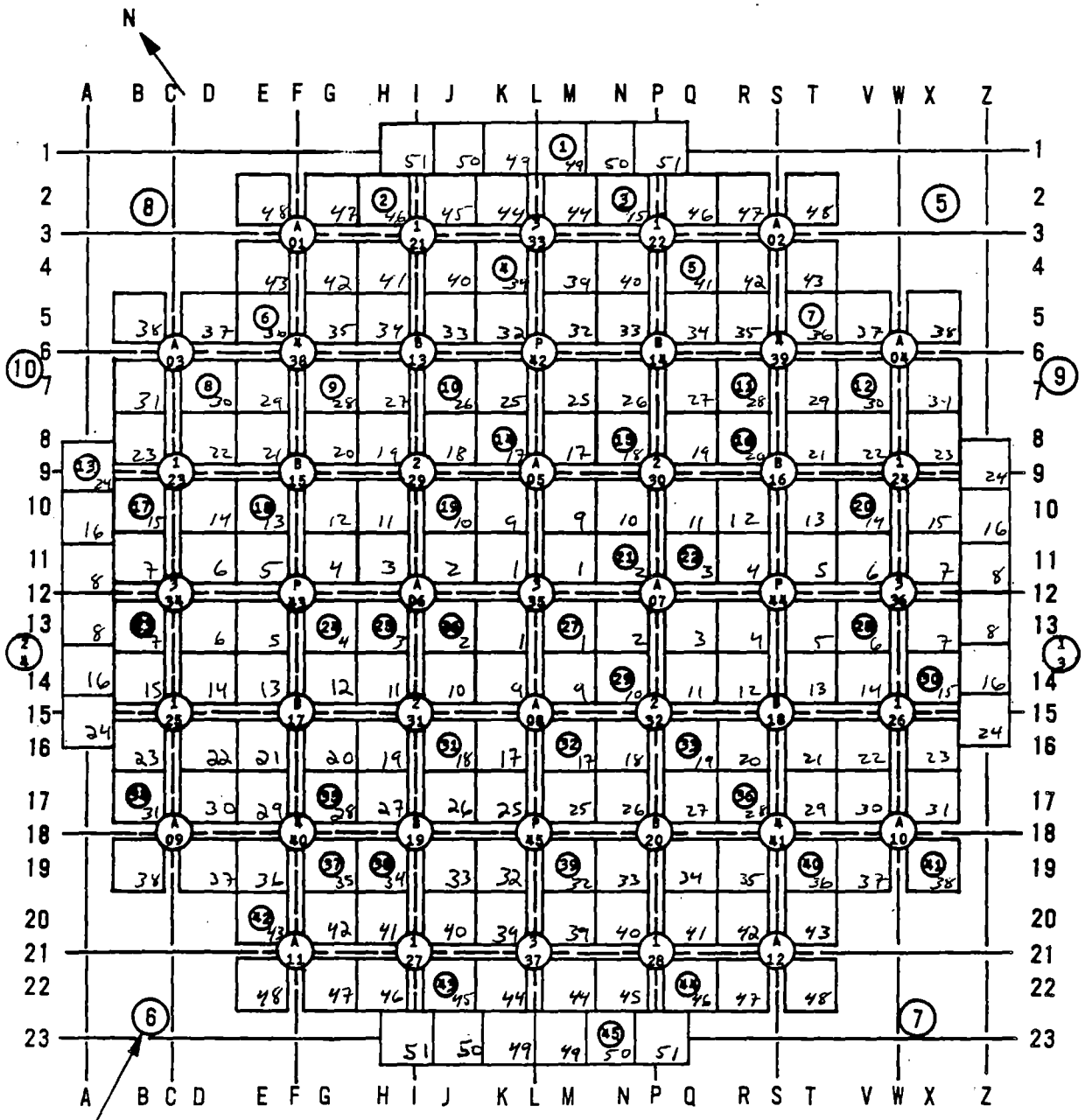
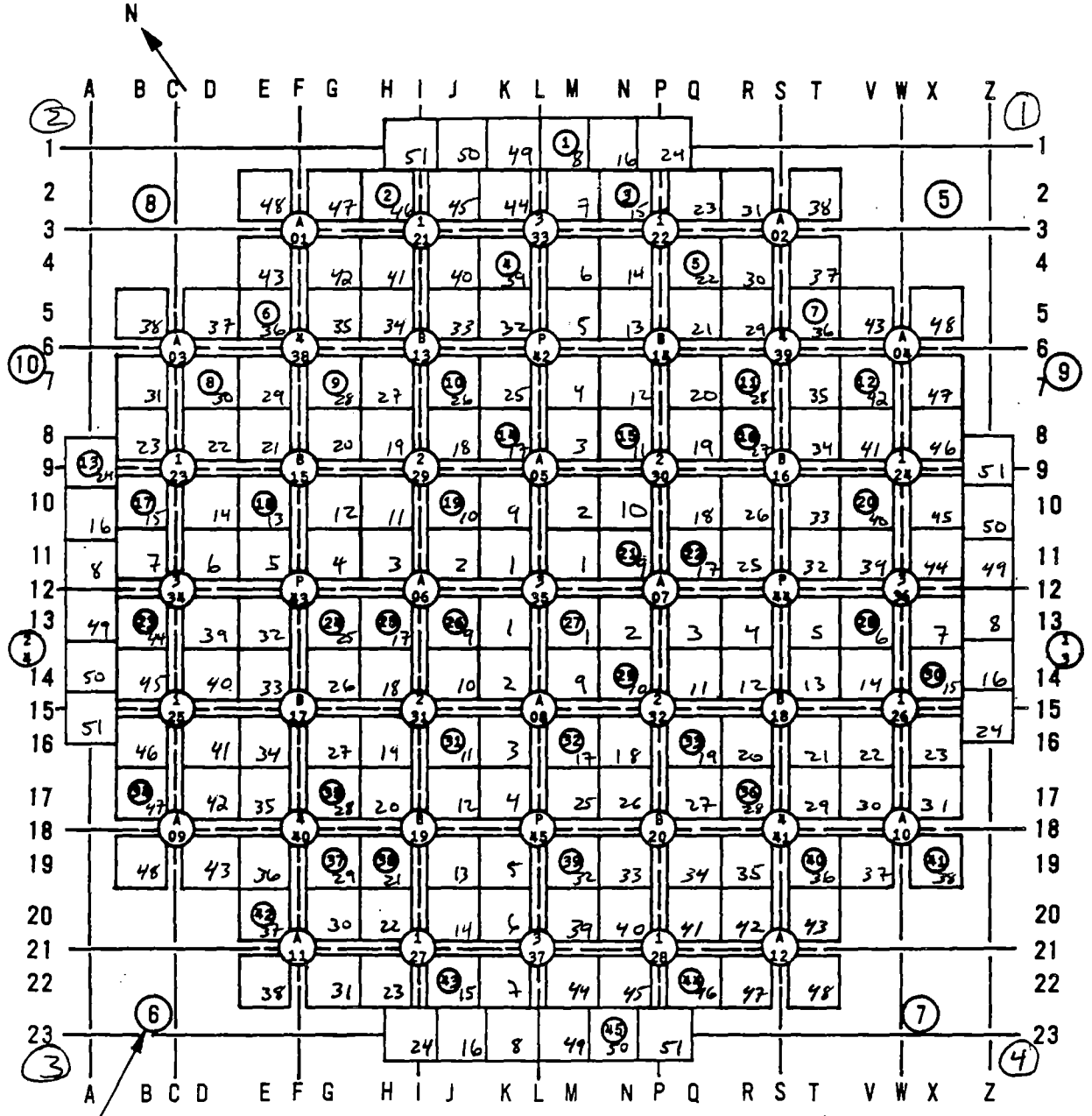


FIGURE 2.4.6b

QUADRANT NUMBERING 3/4 CORE ROTATIONAL

PALISADES PLANT–REACTOR CORE PLAN



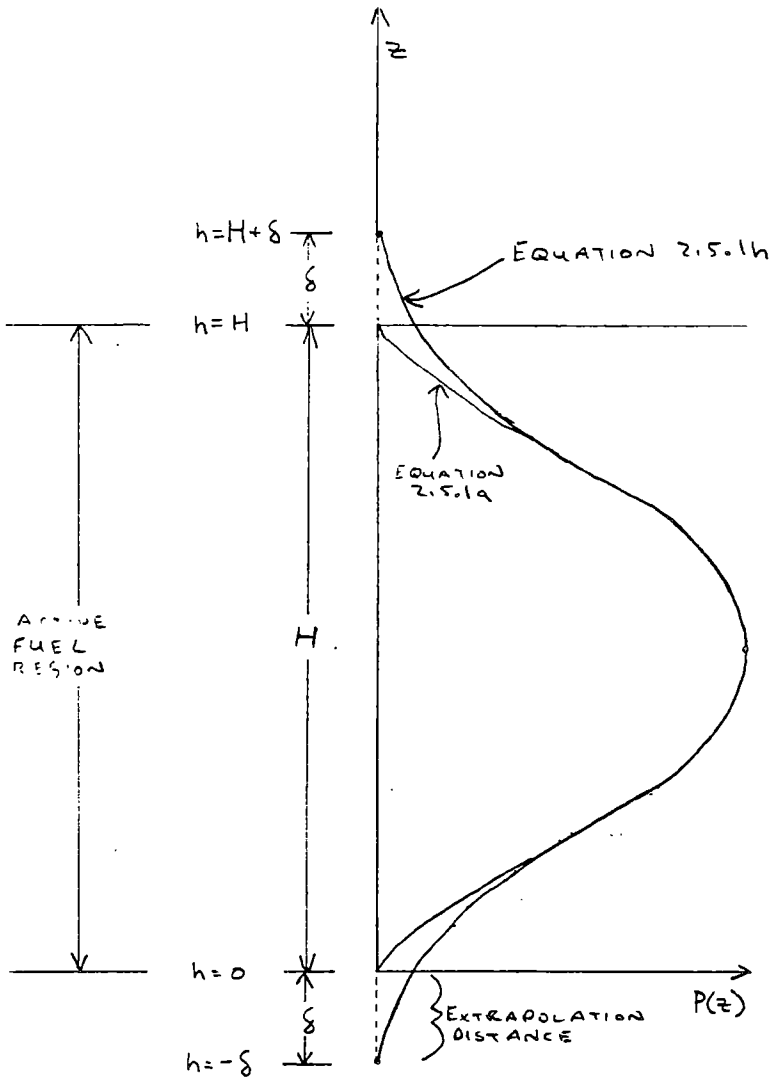


FIGURE #2.5.1 : GRAPHICAL REPRESENTATION OF EQUATIONS DEFINED TO PREDICT THE AXIAL POWER DISTRIBUTION IN THE CORE. NOTE THAT THE CORE FLUX EXTENDS BEYOND THE FUEL REGION WHEN REPRESENTED BY EQUATION 2.5.1h.

LIST of REFERENCES

1 —— P*PID*89002, Calculation of Statistical Uncertainties Associated with the Full Core PIDAL Methodology, Consumers Power Company, Reactor Engineering Department.
2 —— Operational Reactor Physics Section Procedures, Consumers Power Company, Reactor Engineering Department, Procedure ORP-P-01
3 —— Palisades Primary Datalogger Manual, Part 2
4 —— INCA Users Manual, Nuclear Power Department, Combustion Engineering version 1.0
5 —— Operational Reactor Physics Section Procedures, Consumers Power Company, Reactor Engineering Department, Procedure ORP-T-09
6 —— Data Processing Standards Manual, Consumers Power Company, Information Systems Department
7 —— Palisades Technical Specifications, Consumers Power Company
8 —— INCA-XTG Version 3A Users Manual, Exxon Nuclear/CPCo 10/18/84
9 —— IBM System 360 Scientific Subroutine Package Users Manual, 1969
10 —— P*C*03*780320 Cycle 3 Incore Instrumention Data Processor Constants
11 —— CALCULATIONAL VERIFICATION OF THE COMBUSTION ENGINEERING FULL CORE INSTRUMENTATION ANALYSIS SYSTEM CECOR, W.B. TERNEY et al, Combustion Engineering Inc, presented at International Conference On World Nuclear Power, Washington D.C., November 19, 1976.
12 —— XTGPWR UJUN88 Users Manual, Advanced Nuclear Fuels Corporation.

GLOSSARY

INCA - An incore analysis program developed by Combustion Engineering to determine (measure) the power distribution within the Palisades reactor assuming one-eighth or octant core symmetry.

PIDAL - An incore analysis program developed by Consumers Power Company to determine (measure) the power distribution within the Palisades on a full core basis.

XTG - A group and one-half nodal diffusion theory code developed by Advanced Nuclear Fuels Corporation (formerly Exxon Nuclear) for general predictive modeling of pressurized water reactors.

CECOR - An incore analysis program developed by Combustion Engineering to determine (measure) the power distribution within a pressurized water reactor on a full core basis.

PIP - Primary Information Processor. Sometimes called Varian datalogger or datalogger. The primary plant computer installed at Palisades which monitors the incore detector high alarm limits and various other inputs. This machine also generates the incore detector signal and sensitivity dumps which are input to INCA or PIDAL.

Data dump - An output of incore detector signals and sensitivities put out by the PIP for INCA or PIDAL.

BOC - Beginning of Cycle. A new fuel loading or core.

BOL - Beginning of Life. When an instrument or fuel assembly is fresh of new. Unexposed in the reactor.

EOC - End of Cycle. The end point for a fuel cycle, prior to offload.

EOL - End of Life. When an instrument or fuel assembly is spent or ready for discharge.

LHGR - Linear heat generation rate, usually in kilo-watts per foot.

ADC - Analog to digital convertor. ADC counts refer to analog counts which may be converted to an internal, or digital, computer variable.

APL - Allowable power level. See Palisades Tech Spec 3.11.2.

non-nearest- For adjustments to W' values, these could be any control rod not immediately shadowing an incore detector. See pages 63,64 of this report.