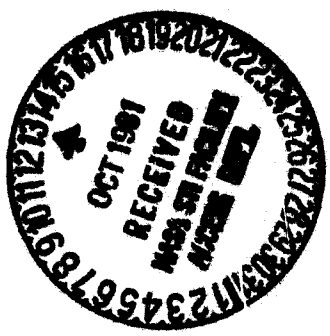
Contract No. NAS8-34279

IGDS/TRAP INTERFACE PROGRAM (ITIP)

DETAILED DESIGN SPECIFICATION (DDS)



September 18, 1981

Prepared for:

Coal Gasification Project Office
George C. Marshall Space Flight Center
National Aeronautics and Space Administration
Marshall Space Flight Center, Alabama 35812

INTERGRAPH
CORPORATION

PREFACE

This report is the Detailed Design Specification (DDS) for a Computer Program Contract End Item (CPCEI) identified as the IGDS/TRAP Interface Program (ITIP). Development of ITIP was performed by Intergraph Corporation as an add-on task under Contract No. NAS8-34279 for the Coal Gasification Project Office of George C. Marshall Space Flight Center (MSFC). The NASA COR for this contract is E. T. Deaton, Jr.

Prepared by:

Steve Jefferys
Wendell Johnson
Robert Lewis
Ralph Rich

Approved by:

John R Conway

J. R. Conway

TABLE OF CONTENTS

Table of Contents

(Continued)

LIST OF ACRONYMS

AST	Asynchronous System Trap
CI	Command Interpreter (IGDS)
CPCEI	Computer Program Contract End Item
DEC	Digital Equipment Corporation
DMRS	Data Management and Retrieval System
EDB	Edge Definition Block
HIPO	Hierarchical Input Processing Output (HIPO)
HOL	Host Language
IGDS	Interactive Graphics Design System
ITIP	IGDS/TRAP Interface Program
NDB	Node Definition Block
PDB	Project Definition Block
PDP	Programmable Data Processor
RAB	Resource Activity Block
RDB	Resource Definition Block
SDD	Software Design Document
TCB	Terminal Control Clock (IGDS)
TRAP	Time-Line and Resources Analysis Program

LIST OF FIGURES

List of Tables

1. SCOPE

This specification establishes the detailed design for a CPCEI identified as the IGDS/TRAP Interface Program (ITIP). This software provides the capability to develop at an Interactive Graphics Design System (IGDS) design station process flow diagrams for use by the NASA Coal Gasification Task Team. In addition, ITIP uses the Data Management and Retrieval System (DMRS) to maintain a data base from which a properly formatted input file to the Time-Line and Resources Analysis Program (TRAP) can be extracted. This set of software resides on the PDP-11/70 in Building 4487, MSFC.

2. APPLICABLE DOCUMENTS

- IGDS8 Application Software Interface Document, 79-076, Rev. B, Intergraph Corporation, November, 1980.
- IGDS Operating Manual (IGDS8), 79-067, Rev. B, Intergraph Corporation, March 1981.
- Graphics Standard Interchange Format, 80-021, Intergraph Corporation, May, 1980.
- DMRS 8.0 HOL Control Block (CB) User's Guide, 79-031, Rev. A, Intergraph Corporation, July 1980.
- DMRS 8.0 HOL User's Guide, 80-124, Intergraph Corporation, April 1980.
- DMRS Data Definition Language (DDL) Compiler, Internal Report No. 79-050, Rev. B, Intergraph Corporation, January 1981.
- DMRS8 Command Language User Guide, 79-065, Rev. B, Intergraph Corporation, February 1981.

3. Requirements

3.1 ITIP Program Configuration

The ITIP is a program which provides the Coal Gasification Task Team with an efficient method for developing network flow diagrams. This network flow diagram consists of subprocess modules (nodes) interconnected by directed links (edges). The ITIP provides software interfaces to the following hardware components:

- IGDS Design Station - Operator interaction with ITIP.
- PDP 11/70 - Host computer for the ITIP Software.
- Honeywell 560 - Host computer for the TRAP Software.
- PDP/Honeywell Interface - Communication link between PDP 11/70 and Honeywell 560.

The functional relationship between the ITIP software components which provide the interfaces between the hardware described above is depicted in Figure 3.1-1. The software components are:

- ITIP Software (described herein)
- IGDS Software (Intergraph product)
- DMRS Software (Intergraph product)
- TRAP Software (NASA product)
- ITIP Data base (described herein)

The main structure of the ITIP software provides the following functions:

- Executive Control
- Initilization functions
- User interaction
- IGDS graphics interface
- DMRS data base interface
- Termination functions

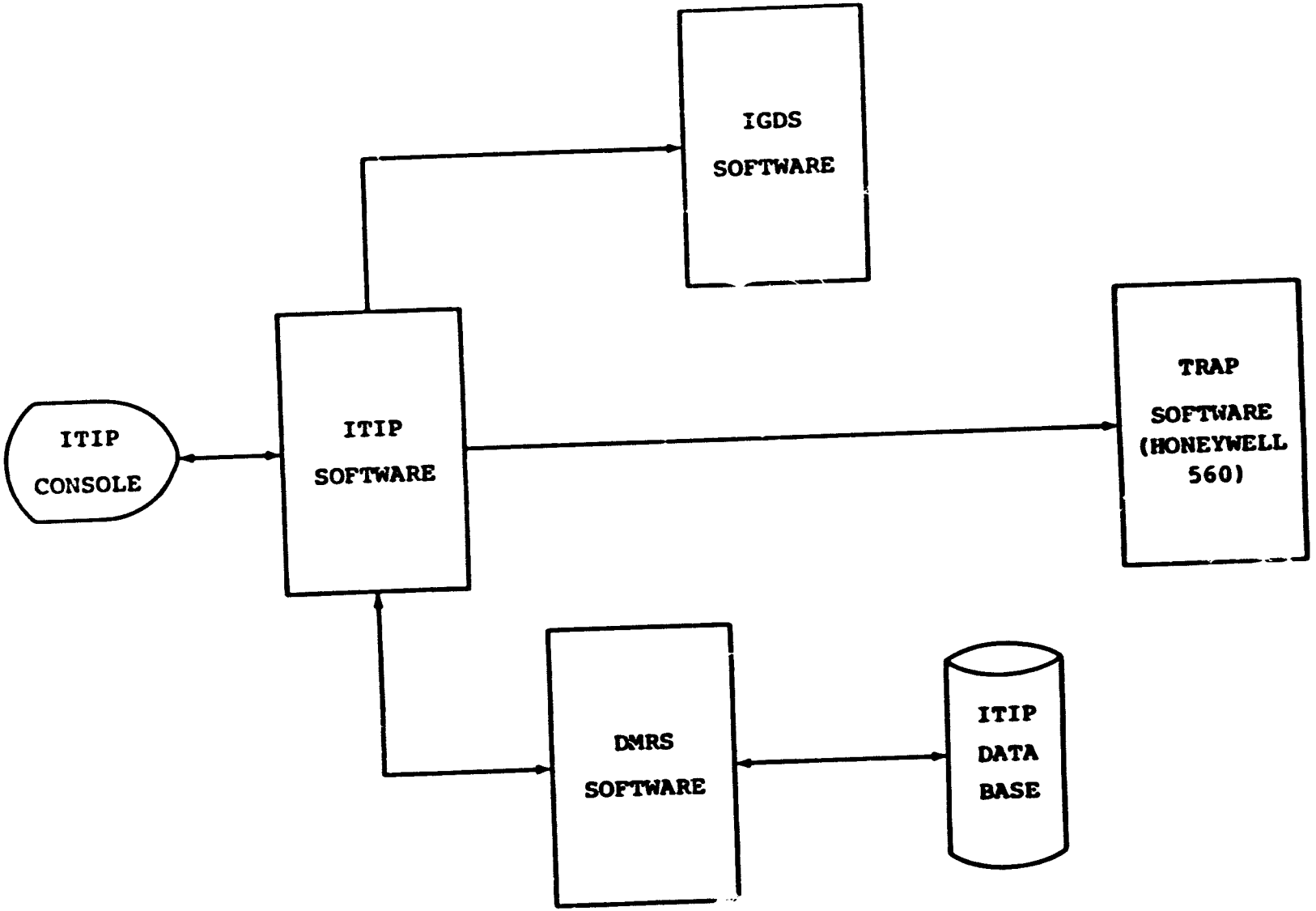


FIGURE 3.1-1 ITIP SOFTWARE SYSTEM OVERVIEW

The modular design of the ITIP software is presented by the structure chart shown in Figure 3.1-2. The organization of the ITIP data base is defined by the following components:

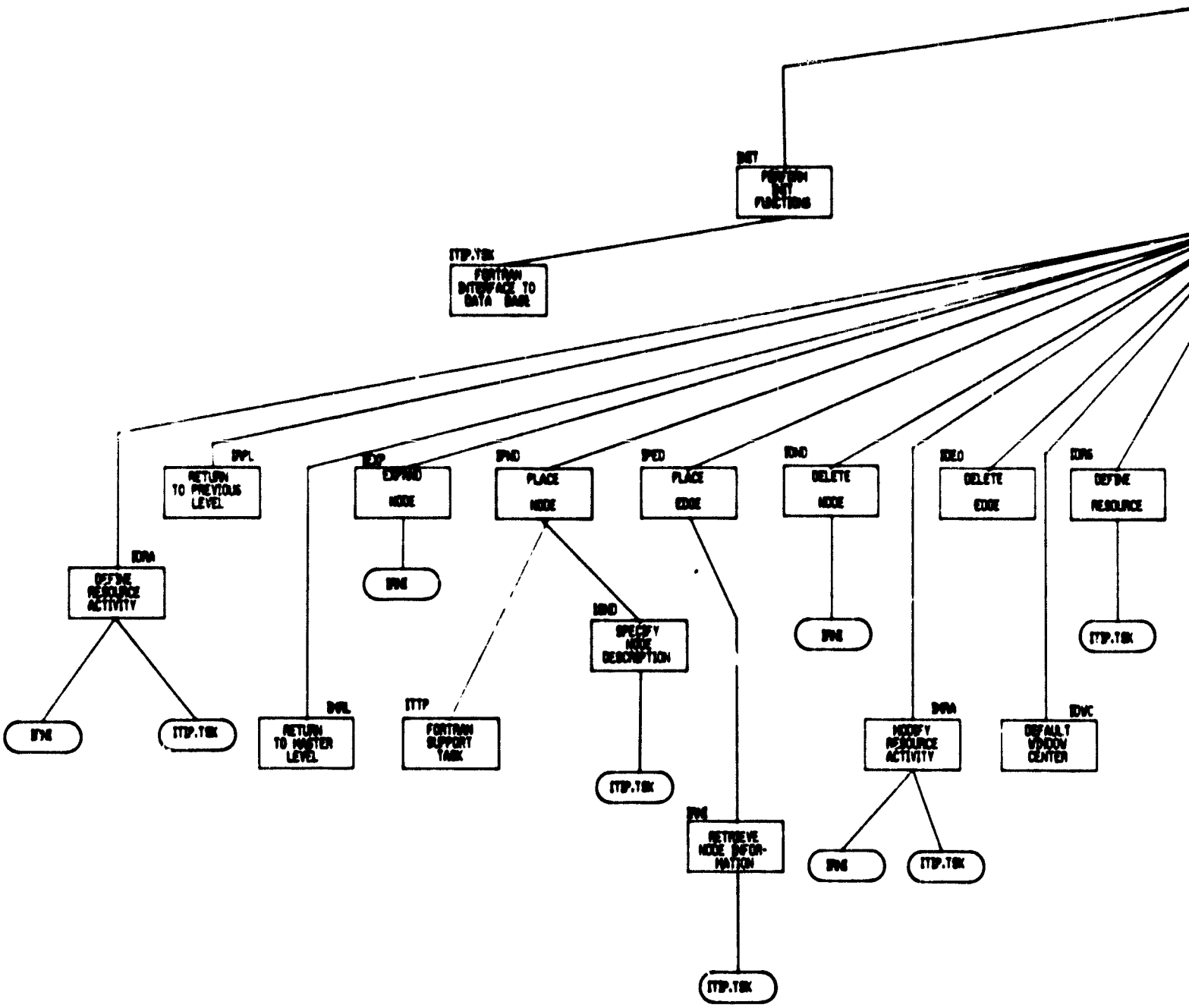
- Project Definition Block (PDB)
- Resource Definition Block (RDB)
- Node Definition Block (NDB)
- Edge Definition Block (EDB)
- Resource Activity Block (RAB)

3.2 ITIP Program Operation

The ITIP program consists of two sets of software. One set of software is written in FORTRAN and is included in the RSX-11M Plus task ITIP. This task provides interaction with the ITIP data base via IGDS/DMRS attribute services software. The other set of software is written in the IGDS Command Interpreter syntax. The modules comprising this set of software are known as user commands. They are processed by the IGDS Command Interpreter to provide software execution of the graphics commands available to the interactive user of the IGDS system.

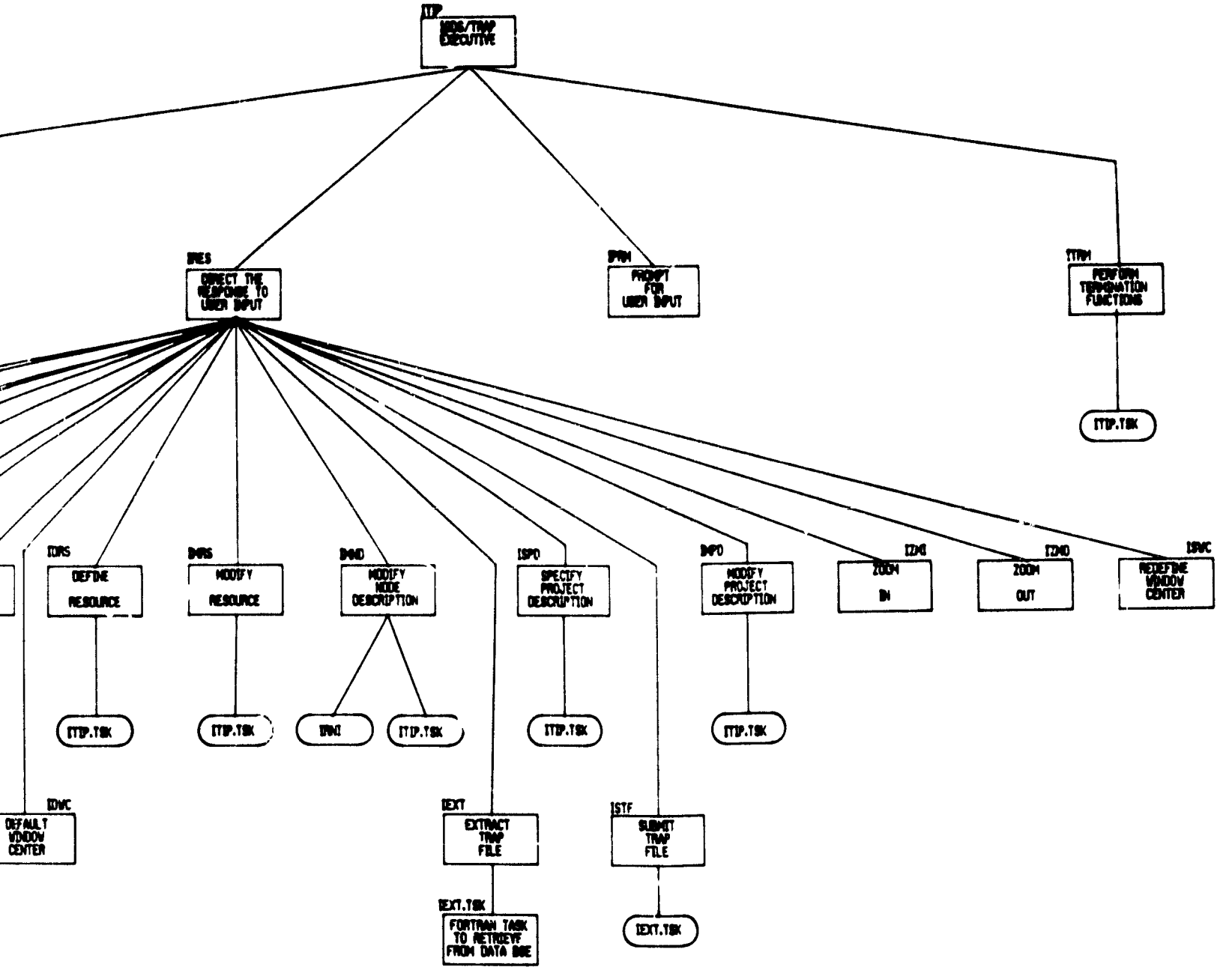
The ITIP is initiated via RSX-11M PLUS Operating System at the request of the user. The user installs the FORTRAN task ITIP by request at the PDP 11/70 console device. The user then logs on to an IGDS design station and requests IGDS execution. Under IGDS control the user then requests execution of the ITIP executive user command.

The ITIP executive user command controls execution of the ITIP software. Initialization of IGDS global registers and initial graphic conditions is then provided and the ITIP data base is attached. The executive user command then directs queries for user input and responses to the user commands. Table 3.2-1 defines the ITIP user options. The index is the internal index used by the software to define the user option. The mnemonic is the two character code input by the user to define the desired option. The ITIP executive user command (ITIP program) is terminated via operator command. Prior to termination the files are closed and IGDS conditions are reset to what they were prior to execution of the ITIP executive user command.



FOLDOUT FRAME

FIGURE 3.1-2 ITIP S



3.1-2 ITIP STRUCTURE CHART

FOLDOUT FRAME 2

INDEX	MNEMONIC	OPTION
1	PN	Place Node
2	PE	Place Edge
3	DN	Delete Node
4	DE	Delete Edge
5	DR	Define Resource
6	MR	Modify Resource
7	UP	UPdate drawing
8	MN	Modify Node description
9	SP	Specify Project description
10	MP	Modify Project description
11	SF	Submit TRAP File
12	ZI	Zoom In
13	ZO	Zoom Out
14	WC	Redefine Window Center
15	DC	Default Window Center
16	EX	EXpand node
17	PL	Return to Previous Level
18	ML	Return to Master Level
19	DA	Define resource Activity
20	MA	Modify resource Activity
21	STOP	Terminate ITIP

Table 3.2-1 ITIP User Options

3.3 ITIP Computer Program Characteristics

This section describes the software modules which comprise the ITIP. These software modules are divided into two classes depending on whether they are written in the IGDS Command Interpretive language or FORTRAN. The ITIP user commands are written in the Command Interpretive language and will be discussed first. The ITIP task is written in FORTRAN, and the modules which comprise this task will then be covered.

3.3.1 User Commands

The environment for the execution of the user commands has the following limitations:

- Predefined set of variable names
- Variables are global
- Subroutines are not supported
- Length of user command limited to 2048 characters

The predefined set of variable names are listed in Table 3.3-1. The fact that these variables are global requires that some be reserved for local usage while others are rigidly defined for use throughout ITIP. The global registers are listed in Table 3.3-2. The registers which are not reserved for local usage must not be used for any purpose other than that which is specifically defined in Table 3.3-2. The fact that subroutines are not supported and the length of a user command is limited presents a problem. The functions required by ITIP cannot be provided in a single user command. This is resolved by the capability to chain user commands together. However, the desirability of developing the software in modular fashion demands the capability to treat some modules as subroutines of others. This is resolved by chaining a user command subroutine back to the command that called it. This is done by setting a global register to a known code prior to calling the subroutine. A check for that code is then provided at the entry point of the calling module to permit branching to the next statement following the call to the subroutine. The global register should then be reset to prevent other subroutines inadvertantly returning to that location.

This section contains a description and Hierarchical Input Processing Output (HIPO) for each user command depicted in the ITIP structure chart shown in Figure 3.1-2. The description of each user command consists of:

- Function of the user command
- Calling sequence
- Modules which call this user command
- Modules called by this user command
- IGDS commands used by this user command
- Local usage of global registers

Each HIPO contains the principal functions performed within the module. Also included with each function are a list of the inputs which may be required to perform the function and a list of the outputs which may be created as a result of performing the function.

Table 3.3-1. User Command Interpreter Variable Names

Name	Size	Description
R0-R31	16 bits	Single-precision integer registers.
ERR	16 bits	Register where the status of each statement executed by the UCI is posted.
NUM	16 bits	Register set to the number of input characters for keyboard input.
I0-I15	32 bits	Double-precision integer registers.
XUR,YUR, ZUR	32 bits	Registers where the X,Y,Z UOR coordinates from a data point are stored.
VNO	16 bits	Register where the view number is stored.
A0-A15	64 bits	Double-precision floating point registers.
KEY	42 characters	Variable where the results of an alphanumeric keyboard input are stored.
MSG	42 characters	Alphanumeric working register.

Table 3.3-2 Global Registers

Register	Usage
R0	Error Code
R1	ITIP Option Index
R2	Return Code (Level 1)
R3	Return Code (Level 2)
R4	Return Code (Level 3)
R5	Not used
R6	Not used
R7	Not used
R8	Not used
R9	Not used
R10	Network Level Index
R11	View Orientation Index
R12	Zoom Index
R13	Next Design Space Index (Node Index)
R14	Current Design Space Index
R15	Previous Design Space Index
R16	Node Parent of Current Design Space
R17	Node Parent of Previous Design Space
R18	Reserved for Local Usage
R19	Reserved for Local Usage
R20	Reserved for Local Usage
R21	Reserved for Local Usage
R22	Reserved for Local Usage
R23	Reserved for Local Usage
R24	Reserved for Local Usage
R25	Reserved for Local Usage
R26	Reserved for Local Usage
R27	Reserved for Local Usage
R28	Reserved for Local Usage
R29	Reserved for Local Usage
R30	Reserved for Local Usage
R31	User Command Communication
I1	Reserved for Local Usage
I2	Reserved for Local Usage
I3	Reserved for Local Usage
I4	Reserved for Local Usage
I5	Reserved for Local Usage
I6	Reserved for Local Usage
I7	Reserved for Local Usage
I8	Reserved for Local Usage
I9	Reserved for Local Usage
I10	Reserved for Local Usage
I11	X-Coordinate of Design Space
I12	Y-Coordinate of Design Space
I13	User Command Communication
I14	User Command Communication
I15	Not Used

3.3.1.1 Executive Module (ITIP)

The purpose of this module is to provide the overall control for the ITIP operation. This module calls module INIT to perform initialization functions, module IPRM to prompt the user for input, module IRES to direct the responses to the user input, and ITRM to provide ITIP termination functions.

Calling sequence: UCM 'QS2:[50,2]ITIP.UCM'

Modules which call ITIP:

None

Modules called by ITIP:

INIT.UCM
IPRM.UCM
IRES.UCM
ITRM.UCM

IGDS commands used by ITIP:

None

Local registers used by ITIP:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ITIP HIPO		
	1	Clear message fields.	2	
	2	Set level one return code and error code.	3	R0, R2
	3	Call a module (INIT) to initialize ITIP variables, IGDS conditions and DMRS data base.	4	
R0	4	If error condition go to step 10. Else continue	10/5	
	5	Set level one return code, option index and error code.	6	R2, R1, R0
	6	Call a module (IPRM) to prompt user for option.	7	
	7	If terminate option selected go to step 10. Else continue.	10/8	
	8	Set level one return and clear level two return.	9	R2, R3
	9	Call a module (IRES) to direct response to user option and go to step 6.	6	
	10	Set level one return.	11	R2
	11	Call a module (ITRM) to perform termination functions.	12	
	12	Display "ITIP TERMINATED" in status field and stop.	E	

3.3.1.2 Initialization Module (INIT)

The functions of this module are to attach to the DMRS data base, to provide initial values for ITIP global registers, and to set IGDS initial conditions.

Calling sequence: UCM 'QS2:[50,2]INIT.UCM'

Modules which call INIT:

ITIP.UCM

Modules called by INIT:

ITIP.TSK

IGDS commands used by INIT:

SNAPLK
GGLOCK
DOUBL1
UPDBTH
TJST7
TXTNLK

Local registers used by INIT:

I1 - X-coordinate

I2 - Y-coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		INIT HIPO		
CONTRL	1	Inhibit command and prompt messages	2	CONTRL
	2	Execute IGDS commands SNAPLK and GGLOCK to turn on snap lock and graphic group locks respectively.	3	
	3	Retrieve cell library and select arrowhead as line terminator.	4	
	4	Initialize design space parameters.		I11, I12, R10, R11, R12
	5	Set data base option code and subcode and call FORTRAN task ITIP to retrieve design space index. If no error is returned go to step 7. Else continue.	7/6	R20, R21, ERR
	6	Initialize design space index, data base subcode, call FORTRAN task ITIP to store design space index, and go to step 8.	8	R13, R20
KEY	7	Set design space index to value returned by ITIP task.	8	R13
	8	Initialize design space and node parent indices.	9	R14, R15, R16, R17
I11, I12	9	Set window origin to master level design space.	10	I1, I2
	10	Execute IGDS command DOUBL1 twice to zoom out the reference screen (left hand) and execute IGDS command UPDBTH to update both screens to display master level.	11	
	11	Define character height and width and line length and spacing.	12	
	12	Execute IGDS commands TJST7 to set text justification to center text and TXTNLK to turn test node lock on.	13	
CONTRL	13	Enable command and prompt messages and return to ITIP user command.	R	CONTRL

3.3.1.3 Terminal Interface Module (IPRM)

The function of this module is to prompt the user for the ITIP option which is to be executed.

Calling sequence: UCM 'QS2:[50,2]IPRM.UCM'

Modules which call IPRM:

ITIP.UCM

Modules called by IPRM:

None

IGDS commands used by IPRM:

None

Local registers used by IPRM:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IPRM HIPO		
	1	Display Command and prompt messages.	2	
	2	Get user input. If key-board entry go to step 4. Else continue.	4/3	KEY
	3	Display message indicating invalid response and go to Step 1.	1	
KEY	4	Test keyboard entry for valid user option. If valid go to step 6. Else continue.	6/5	
	5	Display message indicating invalid mnemonic and go to step 1.	1	
	6	Set option index to reflect user option selected.	7	R1
	7	Return to ITIP user command.	R	

3.3.1.4 Response Direction Module (IRES)

The function of this module is to direct control to the specific user command which provides the IGDS and DMRS capabilities of the user selected ITIP option.

Calling sequence: UCM 'QS2: [50,2]IRES.UCM'

Modules which call IRES:

ITIP.UCM

Modules called by IRES:

IPND.UCM
IPED.UCM
IDND.UCM
IDED.UCM
IDRS.UCM
IMRS.UCM
IMND.UCM
ISPD.UCM
IMPD.UCM
IEXT.UCM
ISTF.UCM
IZMI.UCM
IZMO.UCM
ISWC.UCM
IDWC.UCM
IEXP.UCM
IRPL.UCM
IRML.UCM
IDRA.UCM
IMRA.UCM

IGDS commands used by IRES:

DOUBL1
UPDBTH

Local registers used by IRES:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRES HIPO		
R1	1	If option index is not valid go to step 22. Else continue.	22/2	
R1	2	If option index is 1, set second level return, reset third level return, and call a module (IPND) to place nodes. Go to step 23.	23	R3, R4
R1	3	If option index is 2, set second level return, reset third level return, and call a module (IPED) to place edges. Go to step 23.	23	R3, R4
R1	4	If option index is 3, set second level return, reset third level return, and call a module (IDND) to delete nodes. Go to step 23.	23	R3, R4
R1	5	If option index is 4, set second level return and call a module (IDED) to delete edges. Go to step 23.	23	R3
R1	6	If option index is 5, set second level return and call a module (IDRS) to define a network resource. Go to step 23.	23	R3
R1	7	If option index is 6, set second level return and call a module (IMRS) to modify a network resource Go to step 23.	23	R3

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRES HIPO - Continued		
R1	8	If option index is 7, execute IGDS command UPDBTH to update both screens and go to step 23.	23	
R1	9	If option index is 8, set second level return, reset third level return, and call a module (IMND) to modify the NDB. Go to step 23.	23	R3, R4
R1	10	If option index is 9, set second level return and call a module (ISPD) to specify the PDB. Go to step 23.	23	R3
R1	11	If option index is 10, set second level return and call a module (IMPD) to modify the PDB. Go to step 23.	23	R3
R1	12	If option index is 11, set second level return, call a module (IEXT) to extract TRAP formatted input file, and call a module (ISTF) to submit TRAP file to Honeywell. Go to step 23.	23	R3
R1	13	If option index is 12, set second level return and call a module (IZMI) to zoom in right-hand screen. Go to step 23.	23	R3
R1	14	If option index is 13, set second level return and call a module (IZMO) to zoom out right-hand screen Go to step 23.	23	R3

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRES HIPO - Continued		
R1	15	If option index is 14, set second level return and call a module (ISWC) to redefine the window center. Go to step 23.	23	R3
R1	16	If option index is 15, set second level return and call a module (IDWC) to set default window center. Go to step 23.	23	R3
R1	17	If option index is 16, set second level return, reset third level return, and call a module (IEXP) to expand a node. Go to step 23.	23	R3, R4
R1	18	If option index is 17, set second level return and call a module (IRPL) to return to previous level design space. Go to step 23.	23	R3
R1	19	If option index is 18, set second level return and call a module (IRML) to return to master level design space. Go to step 23.	23	R3
R1	20	If option index is 19, set second level return, reset third level return, and call a module (IDRA) to define a RAB. Go to step 23.	23	R3, R4
R1	21	If option index is 20, set second level return, reset third level return, and	23	R3, R4

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRES HIPO - Continued		
	21	call a module (IMRA) to modify a RAB. Go to step 23.		
	22	Display message indicating option index error and set error index.	23	R0
	23	Return to ITIP user command.	R	

3.3.1.5 Termination Module (ITRM)

The function of this module is to reset the IGDS conditions to what they were prior to the execution of the INIT module.

Calling sequence: UCM 'QS2: [50,2]ITRM.UCM'

Modules which call ITRM:

ITIP.UCM
ITIP.TSK

Modules called by ITRM:

None

IGDS commands used by ITRM:

SNAPLK
GGLOCK
TXTNLK
HALF1

Local registers used by ITRM:

R20

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ITRM HIPO		
	1	Execute IGDS commands SNAPLK, GGLOCK, and TXTNLK to turn on the snap, graphic group, and text node locks, respectively.	2	
	2	Execute the IGDS command HALF1 twice to restore the left-hand screen.	3	
	3	Set data base option code and call FORTRAN task ITIP to store current design space index.	4	R20
	4	Return to ITIP user command.	R	

3.3.1.6 Place Node Module (IPND)

The function of this module is to prompt the user for the center of a node and place an ellipse at that point. The module also prompts for a node description and displays this text in the ellipse. A module ISND is called to prompt for shift factor and node duration and to enter these and other entries in the Node Definition Block (NDB) of the ITIP Data Base. The graphic element (ellipse), text and data base entries constitute placement of a node.

Calling sequence: UCM 'QS2: [50,2]IPND.UCM'

Modules which call IPND:

IRES.UCM

Modules which IPND calls:

ITIP.TSK
ISND.UCM

IGDS commands used by IPND:

PELL1
PTEXTN
PTEXT

Local registers used by IPND:

R20 - Data base option code
R21 - Data base subcode
R29 - Character count
I1 - Ellipse X-coord.
I2 - Ellipse Y-coord.
I3 - Ellipse X-coord.
I4 - Ellipse Y-coord.

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IPND HIPO		
CONTRL	1	Display command and prompt messages and disable message fields.	2	CONTRL
	2	Get user input. If data button go to step 4. If reset button go to step 20. Else continue.	4/20/3	XUR, YUR
	3	Display message indicating invalid response and to to step 1.	1	
XUR, YUR	4	Display message indicating point selected and calculate the coordinate of ellipse.	5	I1, I2, I3, I4
XUR, YUR, I1, I2, I3, I4	5	Execute IGDS Command PELL1 to draw ellipse.	6	
R13	6	Set text node index.	7	R30
XUR, YUR	7	Execute IGDS command PTEXTN to place text node.	8	
	8	Execute IGDS command PTEXT to place text in node.	9	
	9	Display prompt message and get first line of node description.	10	KEY, NUM
NUM, KEY	10	Save number of characters in line, send line to text node, insert blank, and save line.	11	MSG, R29
	11	Display prompt message and get second line of node description.	12	KEY, NUM
NUM	12	If second line is blank, go to step 14. Else continue.	13	
MSG, KEY	13	Append second line to first line to complete node description.	14	MSG
R29, NUM, KEY	14	Update character count, insert line feed in second line and send line to text node.	15	R29, KEY

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IPND HIPO - Continued		
	15	Enable return and call ISND user command to store description in data base and get remaining node description variables.	16	R4
R13	16	Increment design space index, store in KEY buffer, and calculate number of characters in node index.	17	R13, KEY, NUM
	17	Set data base option code and sub-code and call FORTRAN task ITIP.	18	R20, R21
R0	18	If ITIP does not return an error code, go to step 1. Else continue.	1/19	
	19	Display error message.	20	
CONTRL	20	Enable message fields and return to IRES user command.	R	CONTRL

3.3.1.7 Place Edge Module (IPED)

The function of this module is to prompt the user to identify the predecessor and successor nodes of an edge. The graphic element (line with arrowhead) is displayed and the node ID's are entered in the Edge Definition Block (EDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] IPED.UCM'

Modules which call IPED:

IRES.UCM

Modules called by IPED:

IRNI.UCM

IGDS commands used by IPED:

LOCELE
ATCPTO
PLINE
PTERM

Local Registers used by IPED:

R20 - Loop counter
R27 - Predecessor node index
R28 - Successor node index
R30 - Element type
I3 - X Coordinate
I4 - Y Coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IPED HIPO		
CONTRL	1	Display command and prompt messages and disable message fields.	2	CONTRL
	2	Execute IGDS command LOCELE and get data point to locate predecessor node. If data button to to step 4. If reset button go to step 22. Else continue.	4/22/3	XUR, YUR
	3	Display message indicating invalid response and go to step 1.	1	
UELETY	4	Get element type from TCB. If it is an ellipse go to step 6. Else continue.	6/5	R30
	5	Display message indicating that a node was not selected and go to step 2.	2	
XUR, YUR	6	Display message indicating predecessor node selected and save coordinates. Display prompt message for successor node.	7	I1, I2
	7	Reset element type in TCB, execute IGDS command LOCELE and get data point to locate successor node. If data button go to step 9. If reset button go to step 22. Else continue.	9/22/8	UELETY, XUR YUR
	8	Get element type from TCB. If it is an ellipse go to step 10. Else continue.	10/9	
	9	Display message indicating that a node was not selected and go to step 7.	7	
XUR, YUR	10	Display message indicating successor node selected and save coordinates.	11	I3, I4
I1, I2	11	Enable return, restore XUR, YUR coordinates of predecessor node, save calling routine name and call IRNI user command to retrieve node index.	12	R4, XUR, YUR, UCASC R31

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IPED HIPO - Continued		
R31	12	Save predecessor node index, enable return, restore XUR, YUR coordinates of successor node, and call IRNI user command to retrieve node index.	13	R27, R4, XUR, YUR, R31
R31	13	Save successor node index and reset return register.	14	R28, R4
R27, R28	14	Enter predecessor and successor node indices in data base prototype.	15	
I1, I2, I3, I4	15	Execute IGDS command PLINE to place line between predecessor node and successor node.	16	
I3, I4	16	Execute IGDS command PTERM to place arrowhead terminator at successor end of line to complete edge graphic element.	17	
	17	Execute IGDS command ATCPTO to attach prototype data base entries to edge graphic element.	18	
I1, I2, I3, I4	18	Calculate mid point of line, point to line and set loop counter.	19	I1, I2, R20
R20, UELETY	19	Loop checking on element type. If element is a line go to step 20. If line not found go to step 21.	20/21	
	20	Issue accept point, reset and go to step 1.	1	
	21	Display message indicating error attaching prototype and set error code.	22	R0
CONTRL	22	Enable message fields and return to IRES user command.	R	CONTRL

3.3.1.8 Delete Node Module (IDND)

The function of this module is to permit the user to identify a node to be deleted. The graphic element (ellipse) and text are marked for deletion and the linkage to the Node Definition Block (NDB) of the ITIP Data Base deleted.

Calling sequence: UCM 'QS2: [50,2] IDND.UCM'

Modules which call IDND:

IRES.UCM

Modules called by IDND:

IRNI.UCM

IGDS Commands used by IDND:

DLELEM

Local registers used by IDND:

R30 - Element type
I1 - X Coordinate
I2 - Y Coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDND HIPO		
CONTRL	1	Display command and prompt messages and disable message fields and continue.	2	CONTRL
	2	Execute IGDS command LOCELE and continue.	3	
	3	Get data point to locate node. If data button go to step 5. If reset button go to step 11. Else continue.	5/11/4	XUR, YUR
	4	Display message indicating invalid response and go to step 1.	1	
UELETY	5	Get element type from TCB. If it is not on ellipse go to step 7. Else continue.	7/6	R30
XUR, YUR	6	Display message indicating node selected and save coordinates. Go to step 8.	8	I1, I2
	7	Display message indicating node not selected and go to step 1.	1	
I1, I2	8	Save calling routine name, set return code, restore XUR, YUR coordinates of node, and call IRNI user command to retrieve node index.	9	XUR, YUR, R4, UCASC, R31, I13, I14
R31	9	If error returned from IRNI, display message indicating error and go to step 1. Else continue.	1/10	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDND HIPO (Continued)		
I13, I14	10	Execute IGDS Command DLELEM, issue accept point for text node and ellipse then go to step 3.	3	
CONTRL	11	Enable message fields and return to response director.	R	CONTRL

3.3.1.9 Delete Edge Module (IDED)

The function of this module is to permit the user to identify an edge to be deleted. The graphic element (line with arrowhead) is marked for deletion and the linkage to the Edge Description Block (EDB) of the ITIP Data Base deleted.

Calling sequence: UCM 'QS2: [50,2] IDED.UCM'

Modules which call IDED:

IRES.UCM

Modules called by IDED:

None

IGDS Commands used by IDED:

DLELEM

Local registers used by IDED:

R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDED HIPO		
CONTRL	1	Display command and prompt messages, disable message fields and continue.	2	CONTRL
	2	Execute IGDS command DLELEM and get data point to locate edge. If data button go to step 4. If reset button go to step 8. Else Continue.	4/8/3	XUR, YUR
	3	Display message indicating invalid response and go to Step 1.	1	
UELETY	4	Get element type from TCB. If it is either a line or a group, go to step 5. Else go to step 6.	5/6	R30 XUR, YUR
	5	Display a message that edge selected and go to step 7.	7	
	6	Display message indicating that edge not selected and go to step 1.	1	
	7	Issue accept point, reset and go to step 1.	1	
CONTRL	8	Enable message fields and return to response director.	R	CONTRL

3.3.1.10 Define Resource Module (IDRS)

The function of this module is to permit the user to specify values for the entries in the Resource Definition Block (RDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] IDRS.UCM'

Modules which call IDRS:

IRES.UCM

Modules called by IDRS:

ITIP.TSK

IGDS commands used by IDRS:

None

Local registers used by IDRS:

R20 - Data base option code
R21 - Data base sub code
R22 - Data base variable
R23 - Data base variable

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRS HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable message fields.	2	CONTRL
	2	Display prompt for resource ID and get user input. If keyboard entry go to step 4. If reset button go to step 27. Else continue.	4/27/2	KEY, NUM
	3	Display message indicating invalid response and go to step 2.	2	
NUM	4	If number of characters input exceeds 2, display message indicating invalid response and go to step 2. Else continue.	2/5	
KEY, NUM	5	Save resource ID and number of characters.	6	MSG, R22
KEY, NUM	6	Set data base option code and subcode and call FORTRAN task ITIP to check for use of resource ID. If no error is returned go to step 8. Else continue.	8/7	R20, R21, ERR
	7	Display message indicating a resource with this ID has already been defined and go to step 2.	2	
	8	Set data base option code and subcode and call FORTRAN task ITIP to check for available RDB. If no	10/9	R20, R21, ERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRS HIPO (CONTINUED)		
	8	error is returned go to step 10. Else continue.		
	9	Display message indicating that all RDB's are in use and go to step 27.	27	
	10	Display prompt for resource and get input. If keyboard entry, go to step 12. If reset button go to step 27. Else continue.	12/27/1	KEY, NUM
	11	Display message indicating invalid response and go to step 10.	10	
KEY,NUM	12	Save resource description and number of characters.	13	UCASC, R23
	13	Display prompt for constraint level and get input. If keyboard input to go step 15. If reset button go to step 27. Else continue.	15/27/14	KEY,NUM
	14	Display message indicating invalid response and go to step 13.	13	
KEY,ERR	15	Save constraint level and check validity. If invalid go to step 14. Else continue.	14/16	A0
	16	Display prompt for initial quantity of resource and get input. If keyboard input go to step 18. If reset button go to step 27. Else continue.	18/27/17	KEY,NUM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRS HIPO - Continued		
	17	Display message indicating invalid response and go to step 16.	16	
KEY, ERR	18	Save initial quantity and check validity. If invalid go to step 17. Else continue.	17/19	A1
MSG, R22, UCASC, R23, A0, A1	19	Set data base option subcode and call FORTRAN task ITIP to store input values in RDB.	20	R21
	20	Display prompt for resource flags and get input. If keyboard input go to step 22. If reset button go to 27. Else continue.	22/27/21	KEY, NUM
	21	Display message indicating invalid response and go to step 20.	20	
KEY, ERR	22	Save resource flag and check validity. If invalid go to step 21. Else continue.	21/23	R22
	23	Display prompt for resource function and get input. If keyboard input go to step 25. If reset button go to step 27. Else continue.	25/27/24	KEY, NUM
	24	Display message indicating invalid response and go to step 23.	23	
KEY,ERR	25	Save resource function and check validity. If invalid go to step 24. Else continue.	24/26	R23

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRS HIPO - Continued		
R22, R23	26	Set data base option sub-code and call FORTRAN task ITIP to store input values in RDB.	27	R21
CONTRL	27	Enable command and prompt messages and return to ITIP user command.	R	CONTRL

3.3.1.11 Specify Node Description Module (ISND)

The function of this module is to permit the user to specify values for the entries in the Node Definition Block (NDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] ISND.UCM'

Modules which call ISND:

IPND.UCM

Modules called by ISND:

ITIP.TSK

IGDS commands used by ISND:

ATCPTD

Local registers used by ISND:

R20 - Loop counter
R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISND HIPO		
	1	Define node prototype and display command message.	2	
NUM, MSG, R13	2	Enter node description and node index in data base prototype.	3	
	3	Display prompt for shift factor and get user input. If keyboard entry go to step 6. If reset button go to step 5. Else continue.	6/5/4	KEY
	4	Display message indicating invalid response and go to step 3.	3	
	5	Set shift factor to default value.	6	KEY
NUM, KEY	6	Check for valid shift factor. If invalid display message and go to step 2. Else continue.	3/7	
KEY	7	Enter shift factor in data base prototype.	8	
	8	Display prompt for node duration and get user input. If reset button go to step 10. If keyboard entry go to step 11. Else continue.	10/11/9	KEY
	9	Display message indicating invalid response and go to step 8.	8	
	10	Set node duration to default value.	11	
KEY	11	Set ERR register and enter node duration in data base prototype.	12	ERR
ERR	12	If error returned display message and go to step 8. Else continue.	8/13	
R10, R16, XUR, YUR	13	Enter network level, node parent, and coordinates of node center in data base prototype.	14	
	14	Display command message and set loop counter.	15	R20

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISND HIPO - Continued		
R20, I3, I4, UELETY	15	Loop executing IGDS command ATCPTC to attach prototype data base entries to graphic element. If ellipse (node) cannot be found go to step 17. Else continue.	17/16	R30
I3, I4	16	Issue accept point.	17	
	17	Clear command and prompt messages and return to IPND user command.	R	

3.3.1.12 Specify Project Description Module (ISPD)

The function of this module is to permit the user to specify values for the entries in the Project Definition Block (PDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] ISPD.UCM'

Modules which call ISPD:

IRES.UCM

Modules called by ISPD:

ITIP.TSK

IGDS Commands Used by ISPD:

None

Local registers used by ISPD:

R20 - Data base option code
R21 - Data base subcode

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISPD HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable message fields.	2	CONTRL
	2	Display prompt for project title and get user input. If keyboard entry go to step 4. If reset button go to step 7. Else continue.	4/7/3	KEY, NUM
	3	Display message indicating invalid response and go to step 2.	2	
	4	Set data base option code and subcode.	5	R20,R21
NUM	5	If number of characters exceeds 30, set limit to 30.	6	NUM
R20, R21, NUM, KEY	6	Call FORTRAN task ITIP to store project title in PDB.	7	
	7	Display prompt for network ID and get user input. If keyboard input go to step 9. If reset button go to step 11. Else continue.	9/11/8	KEY, NUM
	8	Display message indicating invalid response and go to step 7.	7	
NUM	9	If number of characters in network ID exceeds 4, set limit to 4.	10	NUM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISPD HIPO - Cont.		
NUM, KEY	10	Set data base option sub-code and call FORTRAN task ITIP to store network ID in PDB.	11	R21
	11	Display prompt for network title and get user input. If keyboard input go to step 13. If reset button go to step 15. Else continue.	13/15/12	KEY, NUM
	12	Display message indicating invalid response and go to step 11.	11	
NUM	13	If number of characters in network title exceeds 30, set limit to 30.	14	NUM
NUM, KEY	14	Set data base option sub-code and call FORTRAN task ITIP to store network title in PDB.	15	
	15	Display prompt for start time and get user input. If keyboard input go to step 17. If reset button go to step 19. Else continue.	17/19/16	KEY, NUM
	16	Display message indicating invalid response and go to step 15.	15	
KEY	17	Save start time and check validity. If invalid go to step 16. Else continue	16/18	A0, ERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISPD HIPO - Continued		
NUM, KEY, A0	18	Set data base option subcode and call FORTRAN task ITIP to store start time in PDB.	19	R21
	19	Display prompt for units of time and get user input. If keyboard input go to step 21. If reset button go to step 23. Else continue.	21/23/20	KEY, NUM
	20	Display message indicating invalid response and go to step 19.	19	
NUM	21	If number of characters in units of time exceeds 8, set limit to 8.	22	NUM
NUM, KEY	22	Set data base option subcode and call FORTRAN task ITIP to store units of time in PDB.	23	R21
	23	Display prompt for network duration and get user input. If keyboard input go to step 25. If reset button go to step 27. Else continue.	25/27/24	KEY, NUM
	24	Display message indicating invalid response and go to step 23.	23	
KEY	25	Save network duration and check validity. If invalid go to step 24. Else continue.	24/26	A0, ERR
KEY, NUM, A0	26	Set data base option subcode and call FORTRAN task ITIP to store network duration in PDB.	27	R21

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISPD HIPO - Continued		
CONTRL	27	Enable command and prompt messages and return to IRES user command.	R	CONTRL

3.3.1.13 Modify Resource Module (IMRS)

The function of this module is to permit the user to modify the values of entries in the Resource Definition Block (RDB) of the ITIP Data BAse.

Calling sequence: UCM 'QS2: [50,2] IMRS.UCM'

Modules which call IMRS;

IRES.UCM

Modules called by IMRS:

ITIP.TSK

IGDS Commands used by IMRS:

None

Local registers used by IMRS:

R20 - Data base option code
R21 - Data base subcode
R22 - Data base variable
R23 - Data base variable
R24 - Data base variable

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRS HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable message fields.	2	CONTRL
	2	Display prompt for resource ID and get user input. If keyboard input go to step 4. If reset button go to step 31. Else continue.	4/31/3	KEY, NUM
	3	Display message indicating invalid response and go to step 2.	2	
NUM	4	If number of characters input exceeds 2, then go to step 3. Else continue.	3/5	
NUM, KEY	5	Set data base option code and subcodes and call FORTRAN task ITIP to check for resource ID in RDB. If valid resource ID go to step 7. Else continue.	7/6	R20, R21, R24
	6	Display message indicating resource ID not found and go to step 31.	31	
	7	Set data base option sub-codes and call FORTRAN task ITIP to get resource description from RDB.	8	R21, R22, KEY
KEY	8	Display resource description in status field and get input. If keyboard input go to step 10. If reset button go to step 11. Else continue	10/11/8	KEY, NUM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRS HIPO - CONT.		
	9	Display message indicating invalid response and go to step 7.	7	
KEY, NUM	10	Set data base option subcode and call FORTRAN task ITIP to store resource description in RDB.	11	R21
	11	Set data base option subcodes and call FORTRAN task ITIP to get resource constraint from RDB.	12	R21, R22, KEY
KEY	12	Display resource constraint in status field and get input. If keyboard input go to step 14. If reset button go to step 15. Else continue.	14/15/13	KEY, NUM
	13	Display message indicating invalid response and go to step 11.	11	
KEY	14	Save resource constraint and check validity. If invalid go to step 13. Else continue.	13/15	A1, ERR
KEY, NUM	15	Set data base option subcode and call FORTRAN task ITIP to store resource constraint in RDB.	16	R21
	16	Set data base option subcodes and call FORTRAN task ITIP to get Resource initial quantity from RDB.	17	R21, R22, KEY

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRS HIPO - CONT.		
KEY	17	Display resource initial quantity in status field and get input. If keyboard input go to step 19 If reset button go to step 20. Else continue.	19/20/18	KEY, NUM
	18	Display message indicating invalid response and go to step 16.	16	
KEY	19	Save resource initial quantity and check validity. If invalid go to step 18. Else continue.	18/20	A1, ERR
KEY, NUM	20	Set data base option subcode and call FORTRAN task ITIP to store resource initial quantity in RDB.	21	R1
	21	Set data base option subcodes and call FORTRAN task ITIP to get resource flags from RDB.	22	R21, R22, KEY
KEY	22	Display resource flags in status field and get input. If keyboard input go to step 24. If reset button go to step 25. Else continue.	24/25/23	KEY, NUM
	23	Display message indicating invalid response and go to step 21.	21	
KEY	24	Save resource flags and check validity. If invalid go to step 23. Else continue.	23/25	R24, ERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRS HIPO - CONT.		
KEY, NUM	25	Set data base option sub-code and call FORTRAN task ITIP to store resource flags in RDB.	26	R21
	26	Set data base option sub-codes and call FORTRAN task ITIP to get resource function from RDB.	27	R21, R22 KEY
KEY	27	Display resource function in status field and get input. If keyboard input go to step 29. If reset button go to step 31. Else continue.	29/31/28	KEY, NUM
	28	Display message indicating invalid response and go to step 26.	26	
KEY	29	Save resource function and check validity. If invalid go to step 28. Else continue.	28/30	R23,ERR
KEY, NUM	30	Set data base option sub-code and call FORTRAN task ITIP to store resource function in RDB.	31	R21
CONTRL	31	Enable command and prompt messages and return to IRES user command.	R	CONTRL

3.3.1.14 Modify Node Description Module (IMND)

The function of this module is to permit the user to modify the values of entries in the Node Definition Block (NDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] IMND.UCM'

Modules which call IMND:

IRES.UCM

Modules called by IMND:

IRNI.UCM
ITIP.TSK

IGDS commands used by IMND:

REVPTO
DLELEM
PTEXTN
PTEXT
ATCPTO
DEFPTO

Local registers used by IMND:

R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMND HIPO		
CONTRL	1	Clear error and status fields and disable command and prompt messages.	2	CONTRL
	2	Define review table for NDB and reset element type in TCB.	3	UELETY
	3	Display command and prompt messages.	4	
	4	Execute IGDS command DEFPTO to define data base prototype and get input. If data button go to step 6. If reset button go to step 35. Else continue.	6/35/5	XUR, YUR
	5	Display message indicating invalid response and go to step 3.	3	
UELETY	6	Get element type from TCB. If it is an ellipse go to step 8. Else continue.	8/7	
	7	Display message indicating data for prototype not found and go to step 3.	3	
	8	Execute IGDS command REVPTO to display NDB review table.	9	
	9	Set data base option code and subcode and call FORTRAN task ITIP to retrieve NDB parameters.	10	R20, R21

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMND HIPO - CONT.		
	10	Display CURRENT ATTRIBUTE VALUES in status field and clear error field.	11	
	11	Enable return, save calling routine name, and call a module (IRNI) to retrieve node index and coordinates of center.	12	R4, UCASC R31, I13, I14
R31	12	Test node index. If valid go to step 14. Else continue.	14/13	
	13	Display message indicating that IRNI did not find node and go to step 35.	35	
	14	Display prompt message and get first line of node description. If reset button go to step 26. Else continue.	26/15	KEY, NUM
KEY, NUM	15	Set loop counter, save first line and save number of characters in first line.	16	R28, MSG, R29
I13, I14 R28, UELETY	16	Execute IGDS command DLELEM to delete text node. Loop checking element type for text node. If text node found go to step 18. Else continue.	18/17	R28
	17	Display message indicating text node not found and go to step 35.	35	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMND HIPO - CONT.		
I13, I14, R31	18	Accept text node for deletion and set text node index.	19	
I13, I14	19	Execute IGDS command PTEXTN to place text node.	20	
	20	Execute IGDS command PTEXT to place text in node and send first line to text node.	21	
	21	Display prompt message and get second line of node description.	22	KEY, NUM
NUM	22	If second line is blank, go to step 24. Else continue.	24/23	
MSG, KEY	23	Append second line to first line to complete node description.	24	MSG
NUM, R29, KEY, I13, I14	24	Update character count, insert line feed in second line and send line to text node.	25	NUM, KEY
	25	Change node description attribute in NDB.	26	
	26	Display prompt for shift factor and get user input. If keyboard entry go to step 28. If reset button go to step 30. Else cont	28/30/27	KEY, NUM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMND HIPO - CONTINUED		
	27	Display message indicating invalid response and go to step 26.	26	
NUM, KEY	28	Check for valid shift factor. If invalid display message and go to step 26. Else continue.	26/29	
KEY	29	Enter shift factor in data base prototype.	30	
	30	Display prompt for node duration and get user input. If reset button go to step 34. If keyboard entry go to step 32. Else continue.	34/32/31	KEY
	31	Display message indicating invalid response and go to step 30.	30	
KEY	32	Set ERR register and enter node duration in data base prototype.	33	ERR
ERR	33	If error returned display message and go to step 30. Else continue.	30/34	
XUR, YUR	34	Execute IGDS command ATCPTO to reattach prototype data base entries to graphic element.	35	
CONTRL	35	Enable message fields and return to IRES user command.	R	CONTRL

3.3.1.15 Modify Project Description Module (IMPD)

The function of this module is to permit the user to modify the values of entries in the Project Definition Block (PDB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] IMPD.UCM'

Modules which call IMPD:

IRES.UCM

Modules called by IMPD:

ITIP.TSK

IGDS commands used by IMPD:

None

Local registers used by IMPD:

R20 - Data base option code
R21 - Data base subcode

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMPD HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable messages.	2	CONTRL
	2	Set data base option code and subcode, call FORTRAN task ITIP to retrieve project title from PDB and display project title in status field.	3	R20, R21, KEY, MSG
	3	Display prompt for project title and get user input. If keyboard input, go to step 5. If reset button go to step 7. Else continue.	5/7/4	KEY, NUM
	4	Display message indicating invalid response and go to step 3.	3	
NUM	5	Set data base option code and limit number of characters to 30.	6	R20, NUM
R20, R21, NUM, KEY	6	Call FORTRAN task ITIP to store the project title in PDB.	7	
	7	Set data base option code and subcode, call FORTRAN task ITIP to retrieve network ID from PDB.	8	R20, R21, KEY
KEY	8	Display network ID in status field and set data base option code.	9	MSG, R20

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMPD HIPO - Cont.		
	9	Display prompt for network ID and get user input. If keyboard input go to step 11. If reset button go to step 13. Else continue.	11/13/10	KEY, NUM
	10	Display message indicating invalid response and go to step 9.	7	
NUM	11	If number of characters exceeds 4, set limit to 4.	12	NUM
R20, R21, NUM, KEY	12	Call FORTRAN task ITIP to store network ID in PDB.	13	
	13	Set data base option code and subcode and call FORTRAN task ITIP to retrieve network title from PDB.	14	R20, R21 KEY
KEY	14	Display network title in status field and set data base option code.	15	MSG, R20
	15	Display prompt for network title and get user input. If keyboard input go to step 17. If reset button go to step 19. Else continue.	17/19/16	KEY, NUM
	16	Display message indicating invalid response and go to step 15.	15	
NUM	17	If number of characters exceeds 30, set limit to 30.	18	NUM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMPD HIPO - Cont.		
R20, R21, NUM, KEY	18	Call FORTRAN task ITIP to store network title in PDB.	19	
	19	Set data base option code and subcode and call FORTRAN task ITIP to retrieve start time from PDB.	20	R20, R21 KEY
KEY	20	Display start time in status field and set data base option code.	21	MSG, R20
	21	Display prompt for start time and get user input. If keyboard input go to step 23. If reset button go to step 25. Else continue.	23/25/22	KEY, NUM
	22	Display message indicating invalid response and go to step 21.	21	
KEY	23	Save start time and check validity. If not valid go to step 22. Else continue.	22/24	A0, ERR
R20, R21, A0 NUM, KEY	24	Call FORTRAN task ITIP to store start time in PDB.	25	
	25	Set data base option code and subcode and call FORTRAN task ITIP to retrieve time units from PDB.	26	R20, R21, KEY
	26	Display time units in status field and set data base option code.	27	MSG, R20

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMPD HIPO - Continued		
	27	Display prompt for time units and get user input. If keyboard input go to step 29. If reset button go to step 31. Else continue.	29/31/28	KEY, NUM
	28	Display message indicating invalid response and go to step 27.	27	
NUM	29	If number of characters in time units exceeds 8, set limit to 8.	30	NUM
R20, R21, NUM, KEY	30	Call FORTRAN task ITIP to store time units in PDB.	31	
	31	Set data base option code and subcode and call FORTRAN task ITIP to retrieve network duration from PDB.	32	R20, R21, KEY
KEY	32	Display network duration in status field and set data base option code.	33	MSG, R20
	33	Display prompt for network duration and get user input If keyboard input go to step 35. If reset button go to step 37. Else continue.	35/37/34	KEY, NUM
	34	Display message indicating invalid response and go to step 33.	33	
	35	Save network duration and check validity. If not valid go to step 34. Else continue.	34/36	A0, ERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMPD HIPO - Cont.		
R20, R21, A0, NUM, KEY	36	Call FORTRAN task ITIP to store network duration in PDB.	37	
CONTRL	37	Enable command and prompt messages and return to IRES user command.	R	CONTRL

3.3.1.16 Extract TRAP File Module (IEXT)

The function of this module is to extract a TRAP file.

Calling sequence: UCM 'QS2:[50,2] IEXT.UCM'

Module called by IEXT:

IEXT.TSK

IGDS commands used by IEXT:

None

Local registers used by IEXT:

R20 - FORTRAN task option number

R21 - FORTRAN task subcode.

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IEXT HIPO		
	1	Display the command message.	2	
	2	Set data base option code.	3	R20
R20, R10, R16	3	Call FORTRAN task IEXT to extract the TRAP file.	4	
	4	Get response back from task IEXT.	5	R24
R24	5	If an error occurred, display a message in the error field and set R3 to one so that module ISTF will not be called; else continue.	6	R3
	6	Return to IRES user command.	R	

3.3.1.17 Submit TRAP File Module (ISTF)

The function of this module is to submit a properly formatted Trap Input File to the Honeywell 560 via the PDP/Honeywell Interface.

Calling sequence: UCM 'QS2: [50,2] ISTF.UCM'

Modules which call ISTF:

IRES.UCM

Modules called by ISTF:

ITIP.TSK
IEXT.TSK

IGDS commands used by ISTF:

None

Local registers used by ISTF:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISTF HIPO		
	1	Set the option and suboption and call FORTRAN task ITIP to cause a 4 second delay.	2	R20, R21 R24
	2	Display the command message.	3	
	3	Set the option and call FORTRAN task IEXT to submit the TRAP file.	4	R20, R24
R24	4	If an error occurred display a message; else continue.	5	
	5	Return to IRES user command.	R	

3.3.1.18 Define Resource Activity Module (IDRA)

The function of this module is to permit the user to specify values for the entries in the Resource Activity Block (RAB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2]IDRA.UCM'

Modules which call IDRA:

IRES.UCM

Modules called by IDRA:

ITIP.TSK
IRNI.UCM

IGDS commands used by IDRA:

LOCELE

Local registers used by IDRA:

R20 - Data base option code
R21 - Data base subcode
R22 - Data base variable
R23 - Data base variable
R26 - Intermediate variable
R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRA HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable message	2	CONTRL
	2	Set data base option code and subcode and call FORTRAN task ITIP to check for available RAB.	3	R20, R21, R24
R24	3	If RAB available go to step 5. Else continue.	5/4	
	4	Display message indicating no RAB is available and go to step 24.	24	
	5	Display prompt for resource activity ID and get user input. If keyboard input go to step 7. If reset button go to step 24. Else continue.	7/24/6	KEY, NUM
	6	Display message indicating invalid response and go to step 5.	5	
KEY, NUM	7	Save resource activity ID and number of characters. If number of characters exceeds 2, go to step 6. Else continue.	6/8	MSG, R22
	8	Set default resource activity code, display prompt, and get user input. If keyboard input go to step 10. If reset button go to step 12. Else continue.	10/12/9	R26, KEY, NUM
NUM, KEY	10	If resource activity code is invalid go to step 9. Else continue.	9/11	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRA HIPO - Continued		
KEY	11	Save resource activity code.	12	R26
	12	Set default resource quantity, display prompt, and get user input. If keyboard input go to step 14. If reset button go to step 15. Else continue.	14/15/13	A0, KEY, NUM
	13	Display message indicating invalid response and go to step 12.	12	
KEY	14	Save resource quantity and check validity. If invalid go to step 13. Else continue.	13/15	A0, ERR
	15	Execute IGDS command LOCELE to locate node.	16	
	16	Display prompt and get user input. If data button go to step 18. If reset button go to step 24. Else continue.	18/24/17	XUR, YUR
	17	Display message indicating invalid response and go to step 16.	16	
UELETY	18	Get element type from TCB. If element is ellipse go to step 20. Else continue.	20/19	
	19	Display message indicating element is not a node and go to step 15.	15	
	20	Reset IGDS command LOCELE.	21	
	21	Enable third level return, save calling routine name, and call a module (IRNI) to	22	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDRA HIPO - Cont.		
	21	retrieve node ID from NDB.	22	R4, UCASC, R31
R26	22	Convert resource activity code and set data base option code and subcode.	23	KEY, R20, R21
R20, R21, R22, MSG, A0, KEY, R31	23	Call a FORTRAN task ITIP to store resource activity ID, resource activity code resource quantity, and node ID in RAB.	24	
CONTRL	24	Enable command and prompt messages and return to IRES user command.	R	CONTRL

3.3.1.19 Modify Resource Activity Module (IMRA)

The function of this module is to permit the user to modify the values of entries in Resource Activity Block (RAB) of the ITIP Data Base.

Calling sequence: UCM 'QS2: [50,2] IMRA.UCM'

Modules which call IMRA:

IRES.UCM

Modules called by IMRA:

ITIP.TSK
IRNI.UCM

IGDS commands used by IMRA:

LOCELE

Local registers used by IMRA:

R20 - Data base option code
R21 - Data base subcode
R22 - Data base variable
R25 - Number of characters
R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRA HIPO		
CONTRL	1	Display command message, clear status and error fields, and disable messages.	2	CONTRL
	2	Display prompt for resource activity ID and get user input. If keyboard input to go step 4. If reset button go to step 24. Else continue.	4/24/3	KEY, NUM
	3	Display message indicating invalid response and go to step 2.	2	
KEY, NUM	4	Save resource activity ID and number of characters. If number of characters exceeds 2, go to step 3. Else continue.	3/5	MSG, R25
	5	Execute IGDS command LOCELE to locate node.	6	
	6	Display prompt and get user input. If data button go to step 8. If reset button go to step 24. Else continue.	8/24/7	XUR, YUR
	7	Display message indicating invalid response and go to step 6.	6	
	8	Get element type from TCB. If element is ellipse go to step 10. Else continue.	10/9	
	9	Display message indicating element is not a node and go to step 5.	5	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRA HIPO - Cont.		
	10	Reset IGDS command LOCELE, enable third level return, save calling routine name, and call a module (IRNI) to retrieve node ID from NDB.	11	R4, UCASC, R31
R31	11	Set data base option code and subcode, convert node ID, and set number of characters.	12	R20, R21, KEY, NUM
R20, R21, R25, NUM, MSG, KEY	12	Call FORTRAN task ITIP to search RAB for occurrence of resource activity ID for this node. If occurrence found go to step 14. Else continue.	14/13	R24
	13	Display message indicating that resource activity was not found and go to step 24.	24	
	14	Set data base option sub-codes and call FORTRAN task ITIP to get resource activity code from RAB.	15	R21, R22, KEY
KEY	15	Display resource activity code in status field, display prompt, and get user input. If keyboard input go to step 17. If reset button go to step 19. Else continue.	17/19/16	KEY, NUM
	16	Display message indicating invalid response and go to step 15.	15	
KEY	17	If resource activity code is not valid go to step 16. Else continue.	16/18	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IMRA HIPO - Cont.		
R20	18	Set data base option sub-code and call FORTRAN task ITIP to store resource activity code in RAB.	19	R21
	19	Set data base option code and subcode and call FORTRAN task ITIP to get resource quantity from RAB.	20	R20, R21, KEY
	20	Display resource quantity in status field, display prompt, and get user input. If keyboard input go to step 22. If reset button go to step 24. Else continue.	22/24/21	KEY, NUM
	21	Display message indicating invalid response and go to step 20.	20	
KEY	22	Save resource quantity and check validity. If quantity is not valid go to step 21. Else continue.	21/23	A0, ERR
R20, A0, NUM, KEY	23	Set data base option sub-code and call FORTRAN task ITIP to store resource quantity in RAB.	24	R21
CONTRL	24	Enable command and prompt messages and return to IRES user command.	R	CONTRL

3.3.1.20 Expand Node Module (IEXP)

The function of this module is to expand a node to display the subnodal network which describes it. The design space parameters of the node expansion are set and both screens updated to activate that design space.

Calling sequence: UCM 'QS2: [50,2] IEXP.UCM'

Modules which call IEXP:

IRES.UCM

Modules called by IEXP:

IRNI.UCM

IGDS commands used by IEXP:

LOCELE
UPDBTH
HALF1

Local registers used by IEXP:

R30 - Element type
I1 - X Coordinate
I2 - Y Coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IEXP HIPO		
CONTRL	1	Display command and prompt messages, clear message fields, disable message fields and continue.	2	CONTRL
	2	If return code indicates a return from IRNI, go to step 10. Else continue.	10/3	
	3	Execute IGDS command LOCELE. Get data point to locate node. If data button go to step 5. If reset button go to step 16. Else continue.	5/16/4	XUR, YUR
	4	Display message indicating invalid response and go to step 3.	3	
UELETY	5	Issue accept point. Get element type from TCB. If it is on ellipse go to step 7. Else continue.	7/6	R30
	6	Display a message indicating node not selected and go to step 3.	3	
XUR, YUR	7	Save node coordinates. Issue accept point and reset. Display message indicating node selected and continue.	8	I1, I2
R14, R16	8	Save current design space index in previous design space index. Save current design space node parent in previous design space node parent. Save calling routine name, and continue.	9	R15, R17

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IEXP HIPO - Cont.		
I1, I2	9	Set up return. Restore XUR, YUR coordinates of node. Call IRNI user command to retrieve node index and continue.	10	XUR, YUR, R4
	10	If error returned from IRNI, display message indicating error and go to step 3. Else continue.	11/3	
R31, R14, I11, R10	11	Set current design space index. Set current node parent index. Calculate X-Coordinate of design space set Y-Coordinate of design space. Increment network level index and continue.	12	R14, R16, I11, I12, R10
	12	If zoom index indicates lowest level zoom go to step 14. If zoom index indicates first level zoom go immediately to step 13. Else execute IGDS command HALF1 and set VNO register to right hand screen. Issue accept point with VNO register, reset and continue.	14/13	VNO
	13	Execute IGDS command HALF1 Set VNO register to right hand screen. Issue accept point with VNO register, reset and continue.	14	VNO
I11,I12	14	Reset default in zoom and orientation indices. Set up window origin to current	15	R12, R11, KEY, VNO

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IEXP HIPO - Cont.		
	14	level design space. Set VNO register to right hand screen, issue accept point with VNO register, reset and continue.		
I11, I12, I1, I2	15	Set coordinates of left screen. Set up window origin to current level design space. Set VNO register to left-hand screen, issue accept point with VNO register, reset, execute IGDS command UPDBTH and continue.	16	I1, I2, KEY, VNO
CONTRL	16	Re-enable messages and return to response director.	R	CONTRL

3.3.1.21 Return to Previous Level Module (IRPL)

The function of this module is to reset the design space parameters to the previous level display and update both screens to activate that design space.

Calling sequence: UCM 'QS2: [50,2] IRPL.UCM'

Modules which call IRPL:

IRES.UCM

Modules called by IRPL:

None

IGDS commands used by IRPL:

HALF1
UPDBTH

Local registers used by IRPL:

R30 - Current design space index
I1 - X Coordinate
I2 - Y Coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRPL HIPO		
CONTRL	1	Display command message, clear message fields, disable message fields, and continue.	2	CONTRL
R14, R15, R30, I11	2	If network level index indicates master level go to step 8. Else decrement network level index by 1, save current design space index, reset current design space index, reset previous design space index, adjust X, Y Coordinates to previous level display and continue.	8/3	R10, R30, R14, R15 I11, I12
	3	If orientation index indicates default origin go to step 5. Else set orientation index to default origin and continue.	5/4	R11
R12	4	If zoom index indicates lowest level zoom go to step 5. Else decrement zoom index by 1 and execute IGDS command HALF1, set VNO register to right-hand screen, issue accept point with VNO register, reset and re-execute this step.	5/4	R12, VNO
KEY, I11, I12	5	Reset window origin to master level, set VNO register to right-hand screen, issue accept point with VNO register, reset and continue.	6	KEY, VNO

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRPL HIPO - Cont.		
I11, I12, I1, I2, KEY	6	Set coordinates of left screen. Reset window origin to master level. Set VNO register to left hand screen, issue accept point with VNO register, reset, and continue.	7	I1, I2, KEY, VNO
	7	Execute IGDS command UPDBTH and go to Step 9.	9	
	8	Display message indicating cannot return from master level, set error code and continue.	9	R0
CONTRL	9	Re-enable messages and return to response director.	R	CONTRL

3.3.1.22 Return to Master Level Module (IRML)

The function of this module is to reset the design space parameters to the master level network display and update both screens to activate that design space.

Calling sequence: UCM 'QS2: [50,2] IRML.UCM'

Modules which call IRML:

IRES.UCM

Modules called by IRML:

None

IGDS commands used by IRML:

HALF1
UPDBTH

Local register used by IRML:

I1 - Coordinate of left screen
I2 - Coordinate of left screen

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRML HIPO		
CONTRL	1	Display command message clear message fields, disable message fields, and continue.	2	CONTRL
R14, R16	2	Set network level index to master level, save previous design space index, reset current design space index, save previous node parent, set node parent of master level, adjust X,Y Coordinates of design space and continue.	3	R10, R15, R14, R17, R16, I11, I12
	3	If orientation index indicates default origin go to step 5. Else set orientation index to default origin and continue.	5/4	R11
R12	4	If zoom index indicates lowest level zoom go to step 5. Else decrement zoom index by 1, execute IGDS command HALF1, set VNO register to right-hand screen, issue accept point with VNO register, reset and re-execute this step.	5/4	R12, VNO
I11, I12, KEY	5	Reset window origin to master level. Set VNO register to right-hand screen, issue accept point with VNO register, reset and continue.	6	VNO KEY

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRML HIPO - Cont.		
I11, I12, I1, I2 KEY	6	Set coordinates of left-hand screen. Reset window origin to master level design space. Set VNO register to left-hand screen, issue accept point with VNO register, reset and execute IGDS command UPDBTH.	7	I1, I2, KEY, VNO
CONTRL	7	Re-enable messages and return to response director.	R	CONTRL

3.3.1.23 Redefine Window Center Module (ISWC)

The function of this module is to permit the user to redefine the window center of the current design space.

Calling sequence: UMC 'QS2: [50,2] ISWC.UCM'

Modules which call ISWC:

IRES.UMC

Modules called by ISWC:

None

IGDS commands used by ISWC:

UPDAT1

Local registers used by ISWC:

R30 - Element type
I1 - X Coordinate
I2 - Y Coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISWC HIPO		
	1	Display command and prompt messages and clear message fields and continue.	2	
	2	Get data point to locate window center. If data button go to step 4. If reset button go to step 9. Else continue.	4/9/3	XUR, YUR
	3	Display message indicating invalid response and go to step 1.	1	
XUR, YUR GOXUOR, GOYOUR	4	Reset design space parameters and store. If zoom index indicates zoom at lowest level go to step 7. If zoom index indicates zoom at first level go to step 6. Else continue.	7/6/5	I1, I2
I1, I2	5	Reset orientation index to zoom and window center. Adjust coordinates of design space to zoom at second level then go to step 8.	8	I11, I12, R11
I1, I2	6	Adjust coordinates of design space to zoom at first level. Reset orientation index to zoom and window center. Then go to step 8.	8	I11, I12, R11
I1, I2	7	Reset orientation index to origin redefined via window center. Adjust coordinates of design space to zoom at lowest level and continue.	8	I11, I12, R11

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ISWC HIPO - Cont.		
I11, I12	8	Set coordiantes of window center in data base prototype. Set VNO register to right-hand screen. Issue accept point with VNO register. Execute IGDS command UPDAT1 and continue.	9	KEY, VNO
	9	Return to response director.	R	

3.3.1.24 Default Window Center Module (IDWC)

The function of this module is to permit the user to redefine the window center to the default center of the current design space.

Calling sequence: UMC 'QS2: [50,2] IDWC.UMC'

Modules which call IDWC:

IRES.UCM

Modules called by IDWC:

None

IGDS commands used by IDWC:

WINDC1
UPDAT1

Local registers used by IDWC:

None

C-2

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDWC HIPO		
CONTRL	1	Display command message clear message fields, disable message fields then continue.	2	CONTRL
I11, R14	2	If orientation index indicates window unchanged go to step 8. Else redefine coordinates of window center and continue.	8/3	I11, I12
I11, I12 GOXUOR, GOYOUR	3	Execute IGDS command WINDC1. Adjust coordinates to the current design space. Issue accept point. Set VNO register to right-hand screen, and issue accept point with VNO register.	4	XUR, YUR VNO
R11, I11 I12	4	Reset orientation index. If zoom index indicates zoom at lowest level, go to step 6. If zoom index indicates zoom at first level, go to step 5. Else adjust coordinates to zoom at second level and go to step 7.	6/5/7	R11, I11, I12
I11, I12	5	Adjust coordinates of design space to zoom at first level. Go to step 7.	7	I11, I12
I11, I12	6	Adjust coordinates of design space to zoom at lowest level and continue.	7	I11, I12
	7	Execute IGDS command UPDAT1.	8	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IDWC HIPO - Cont.		
CONTRL	8	Enable message fields and return to response director.	R	CONTRL

3.3.1.25 Zoom in Module (IZMI)

The function of this module is to permit the user to zoom in on the current design space.

Calling sequence: UMC 'QS2: [50,2] IZMI.UMC'

Modules which call IZMI:

IRES.UCM

Modules called by IZMI:

None

IGDS commands used by IZMI:

HALF1
UPDAT1

Local registers used by IZMI:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IZMI HIPO		
	1	Display command message, clear message fields, and continue.	2	
	2	If zoom index indicates zoom out twice go to step 4. If zoom index indicates zoom past default level go to step 9. Else continue.	4/3, 3	
I11, I12	3	Adjust coordinates of design space to zoom in once then go to step 5.	5	I11, I12
I11, I12	4	Adjust coordinates of design space to zoom in twice then continue.	5	I11, I12
R12	5	Decrement zoom index by one. If zoom index not at the lowest level go to step 7. Else continue.	7/6	R12
R11	6	Decrement orientation index and continue.	7	R11
	7	Execute IGDS command HALF1, set VNO register to right-hand screen, issue accept point with VNO register, issue reset command and continue.	8	VNO
	8	Execute IGDS command UPDAT1 and go to step 10.	10	
	9	Display message indicating cannot zoom past this level, reset option index.	10	R1
	10	Return to response director	R	

3.3.1.26 Zoom Out Module (IZMO)

The function of this module is to permit the user to zoom out on the current design space.

Calling sequence: UMC 'QS2: [50,2] IZMO.UMC'

Modules which call IZMO:

IRES.UMC

Modules called by IZMO:

None

IGDS commands used by IZMO:

DOUBL1
UPDAT1

Local registers used by IZMO:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IZMO HIPO		
	1	Display command message, clear message fields and continue.	2	
R12	2	If zoom index indicates zoom out twice go to step 8. Else increment zoom index and continue.	8/3	R12
R11	3	If orientation index indicates window center changed go to step 4. Else reset orientation index to zoom only and go to step 5.	4/5	R11
	4	Reset orientation index to both window and zoom then continue.	5	R11
I11, I12	5	If zoom index indicates zoom out twice go to 6. Else adjust coordinates to zoom out once and go to step 7.	6/7	I11, I12
I11, I12	6	Adjust coordinates of design space to zoom out twice and continue.	7	I11, I12
	7	Execute IGDS command DOUBL1, set VNO register to right-hand screen, issue accept point with VNO register, issue reset command, execute IGDS command UPDAT1, and go to step 9.	9	VNO
	8	Display message indicating cannot zoom more than twice, reset option index and continue.	9	R0

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IZMO HIPO - Cont.		
	9	Return to response director.	R	

3.3.1.27 Return Node Information Module (IRNI)

The function of this module is to retrieve the node ID and coordinates of the node center from the Node Definition Block (NDB) of the data base.

Calling sequence: UMC 'QS2: [50,2] IRNI.UMC'

Modules which call IRNI:

IRES.UCM

Modules called by IRNI:

ITIP.TSK

IGDS commands used by IRNI:

LOCELE

Local registers used by IRNI:

I5 - X Coordinates
I6 - Y Coordinates
R20 - Option number for FORTRAN task
R21 - Suboption
R22 - Entity number
R23 - Occurrence number
R24 - Protection counter
R30 - Element type

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRNI HIPO		
	1	Clear error field and set defaults for return registers.	2	R31, I13, I14
XUR, YUR	2	Set loop counter and X and Y Coordinates for node search.	3	R20, I5, I6
XUR, YUR	3	Execute IGDS command LOCELE to locate ellipse and go to step 5.	5	
I5, I6, R20	4	Give another point and increment loop counter. If loop limit exceeded go to step 11. Else continue.	11/5	R20
UELETY	5	Get element type from TCB. If element is ellipse go to step 7. Else continue.	7/6	R30
I5	6	Reject trial point, increment X coordinate, and go to step 4.	4	I5
BYTLEN	7	Set data base option code and get design buffer index from TCB.	8	R20, R21
R21, DGNBUF	8	Get entity number, occurrence number and protection number from design buffer.	9	R22, R23, R24
	9	Set data base option sub-code.	10	R21
R20, R21, R22, R23, R24	10	Call FORTRAN task ITIP to retrieve mode ID and X and Y coordinates of node center from NDB.	11	R31, I13, I14

INPUT	STEP	PROCESSING	STEP	OUTPUT
		IRNI HIPO - Cont.		
UCASC	11	Return to calling routine	R	

3.3.1.28 Arrowhead Generation Module (ARO)

The function of this module is to create an arrowhead cell and to enter this in the ITIP Cell Library for use in placing edges.

Calling sequence: UMC 'QS2: [50,2] ARO.UMC'

Modules which are ARO:

None

Modules called by ARO:

None

IGDS commands used by ARO:

PLINE
PFENCB
DOCELL
DLFNCC

Local registers used by ARO:

I1 - X coordinate
I2 - Y coordinate
I3 - Y coordinate
I4 - X coordinate
I5 - Y coordinate

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ARO HIPO		
	1	Retrieve cell library ITIP.CEL.	1	
	2	Display command and prompt messages CREATE ARROW and GIVE POINT.	3	
	3	Get input point.	4	XUR, YUR
XUR, YUR	4	Calculate coordinates of arrowhead.	5	I1, I2, I3 I4, I5
I1, I2, I4, I5	5	Execute IGDS command PLINE to draw lower half of arrowhead.	6	
I1, I3, I4, I5	6	Execute IGDS command PLINE to draw upper half of arrowhead.	7	
I1, I2, I3, I4	7	Calculate fence coordinates and execute IGDS command PFENCB to place fence block around the arrow-head.	8	I1, I2
I4, I5	8	Execute IGDS command DOCELL to define cell origin at the point of arrowhead.	9	
	9	Create point cell named ARO containing the arrow-head in the ITIP.CEL library.	10	
I4, I5	10	Execute IGDS command DLFNCC to delete the fence contents (arrowhead).	11	
	11	Display message OPERATION COMPLETE and exit.	E	

3.3.2 FORTRAN Modules

The FORTRAN software which supports the ITIP consists of two tasks ITIP and IEXT. The ITIP task supports the interactive creation of a network and the IEXT task extracts and submits the TRAP file. Both tasks are activated by nodes being sent from user commands. The tasks then respond according to the data received from the user commands and send data back to the user commands.

The ITIP task is initially activated and then remains in execution for the duration of the ITIP session. The task establishes an Asynchronous System Trap (AST) when it is initially executed. The AST will occur each time a node is sent to the ITIP task. The task will respond to the node and then go into a wait state until the next node is received. A node is sent at the end of the ITIP session to cause the task to exit from the system. This task uses a utility package developed by Intergraph called Attribute Services (AS). AS handles the interface to the DMRS for the storage, modification, and retrieval of data from the data base.

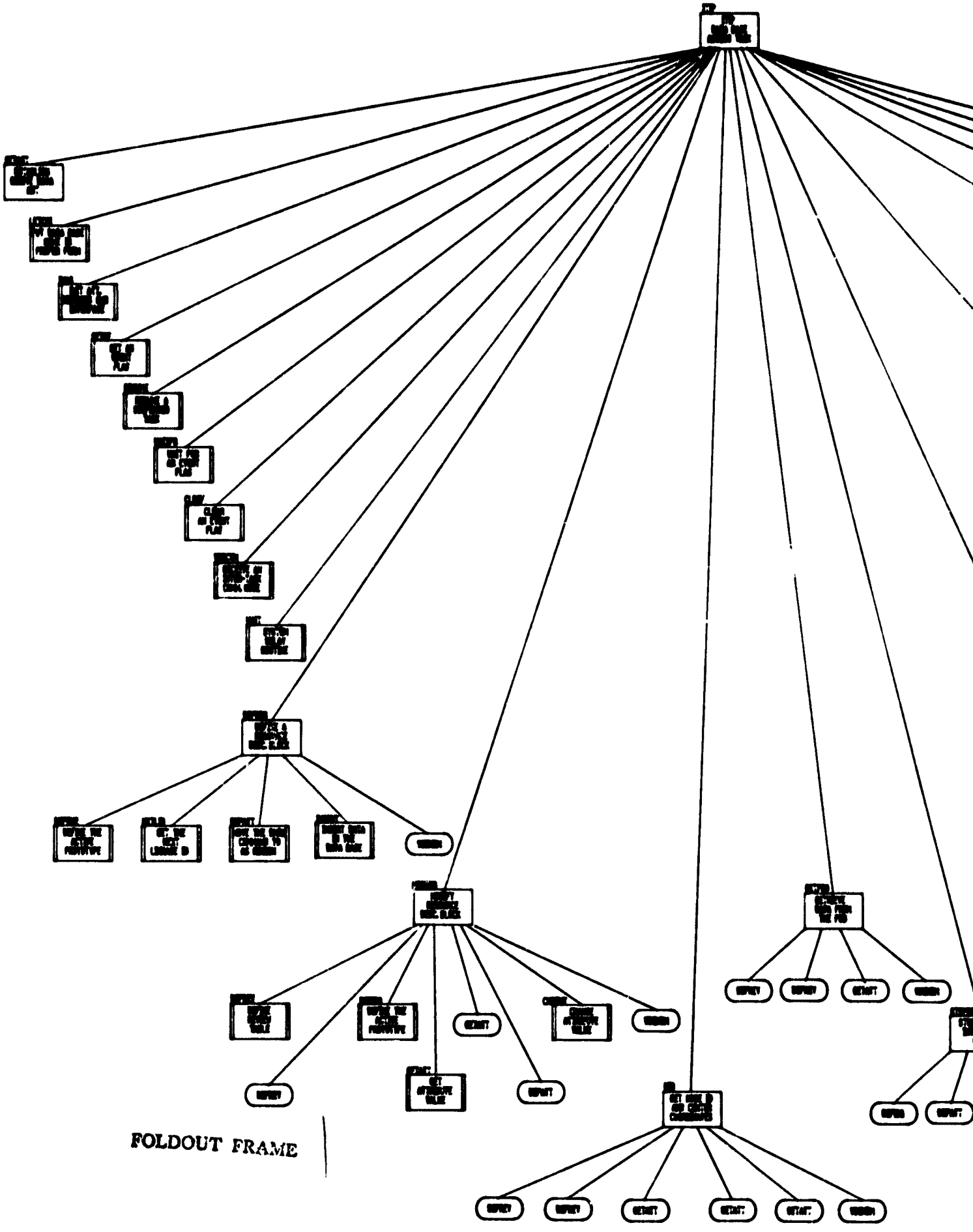
The IEXT task is activated by a user command sending a node to the task. The task responds to the node and then exits the system. This task uses the DMRS Host Language (HOL) to retrieve data from the database.

One limitation of this technique is that the FORTRAN tasks cannot interact directly with the user but must interact indirectly through user commands.

This section contains a description and Hierarchial Input Processing Output (HIPO) for each of the modules in tasks ITIP and IEXT. The structure charts for these two tasks are in figures 3.3.2-1 and 3.3.2-2. The description of each module consists of:

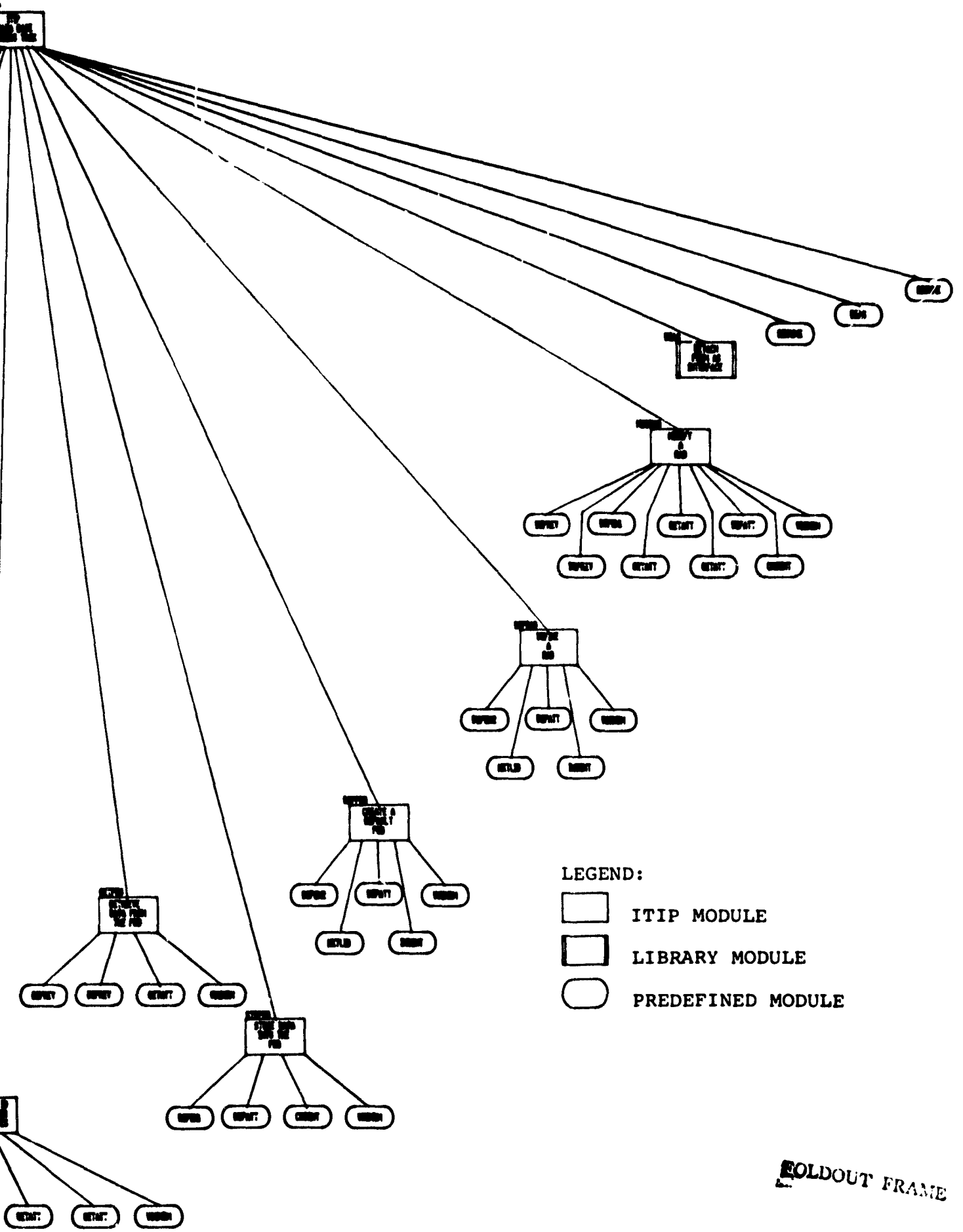
- Function of the module
- Calling sequence
- Modules which call this module
- Modules called by this module

Each HIPO contains the principal functions performed within the module. Also included with each function are a list of the inputs which may be required to perform the function and a list of the outputs which may be created as a result of performing the function.



FOLDOUT FRAME

FIGURE 3.3.2-1 ITIP STRUCTURE CHART (FORTRAN



FOLDOUT FRAME 2

RUCTURE CHART (FORTRAN TASK)

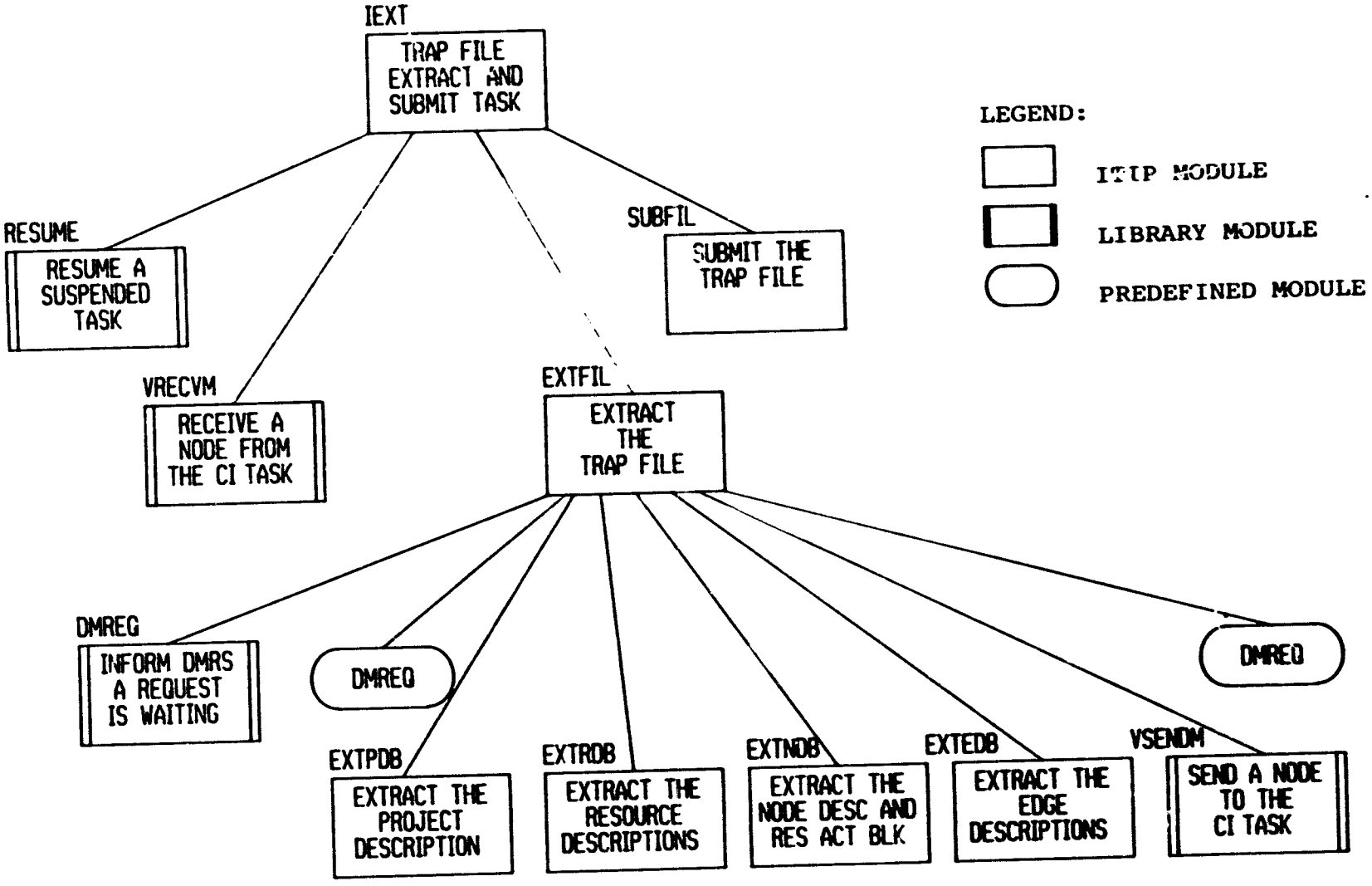


FIGURE 3.3.2-2 IEXT STRUCTURE CHART (FORTRAN TASK)

3.3.2.1 Create a Default PDB Module (DEFPDB)

The function of this module is to store default values into the Project Description Block.

Calling sequence:

Call DEFPDB (ISC, IARG, RARG, ISTRNG, CITASK, IERR)

Modules which call DEFPDB:

ITIP

Modules called by DEFPDB:

DEFFN2, NXTLID, DEFATT, INSENT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		DEFPDB		
	1	Call a module (DEFEN2) to define the PDB as the prototype.	2	IERR
IERR	2	If an error occurred go to step 9.	9/3	
	3	Call a module (NXTLID) to get the next occurrence of the PDB entity.	4	LID,IERR
IERR	4	If an error occurred go to step 9.	9/5	
DEFALT	5	Call a module (DEFATT) to move the defaults for the PDB into the DMRS dynamic region.	6	IERR
IERR	7	If an error occurred go to step 9.	9/8	
LID	8	Call a module (INSENT) to store the default values.	9	IERR
IERR	9	Call a module (VSENDM) to send the return code to the CI task.	R	

3.3.2.2 Define a RAB Module (DEFRAB)

The function of this module is to define a Resource Activity Block (RAB). Two calls are made to this routine. The first determines if any RAB's are available. The second call stores data in the first available RAB.

Calling sequence:

Call DEFRAB (ISC, ISTRNG, IARG, RARG, CITASK, IRC)

Modules which call DEFRAB:

ITIP

Modules called by DEFRAB:

DEFEN2, NXTLID, DEFATT, INSENT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		DEFRAB		
ISC	1	If this is a store request go to step 6; else continue.	6/2	
	2	Call module (DEFEN2) to define the RAB as the active prototype.	3	IRC
IRC	3	If an error occurred go to step 15; else continue.	15/4	
	4	Get the linkage for the next available occurrence of the RAB entity.	5	LID,IRC
	5	Go to step 15.	15	
ISTRNG,IARG	6	Put the resource ID in the DMRS command buffer.	7	IDATA
ISTRNG	7	Put the resource activity code in the DMRS command buffer.	8	IDATA
RARG(1)	8	Put the resource quantity in the DMRS buffer.	9	IDATA
IARG(2)	9	Put the RAB owner in the DMRS buffer.	10	IDATA
RARG(2),IVAL	10	Put the expansion owner in the DMRS buffer.	11	IDATA
ICHAR,IDATA	11	Call a module (DEFATT) to move the DMRS buffer to the DMRS dynamic region.	12	IRC
IRC	13	If an error occurred go to step 15; else continue.	15/14	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		DEFRAB(continued)		
LID	14	Call a module (INSENT) to insert the data into the database.	15	IRC
IRC	15	Call a module (VSENDM) to send the return code to the CI task.	R	

3.3.2.3 Define a RDB Module (DEFRDB)

The function of this module is to allocate a Resource Description Block (RDB) and store data into it.

Calling sequence:

Call DEFRDB (ISC, IRID, IRDESC, IARG, CONST, RQTY, CITASK, IRC)

Modules which call DEFRDB:

ITIP

Modules called by DEFRDB:

DEFEN2, NXTLID, DEFATT, INSENT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		DEFRDB		
ISC	1	If it is desired to store the resouce ID, description, constraint level and initial quantity go to step 7; else continue.	7/2	
ISC	2	If it is desired to store the resource flags and function go to step 11; else continue.	11/3	
	3	Call a module (DEFEN2) to define the RDB as the prototype.	4	IRC
IRC	4	If an error occurred go to step 15; else continue.	15/5	
	5	Call a module (NXTLID) to get the next available occurrence of the prototype.	6	LID,IRC
	6	Go to Step 15.	15	
IRID,IARG(1)	7	Store the resource constraint level in the DMRS command buffer.	8	IDATA
IRDESC, IARG(2)	8	Store the resource description in the DMRS command buffer.	9	IDATA
CONST	9	Store the resource constraint level in the DMRS command buffer.	10	IDATA
RQTY	10	Store the resource initial quantity in the DMRS command buffer and go to step 15.	11	IDATA
IARG	11	Store the resource flags and the resource function in the DMRS command buffer.	12	IDATA

INPUT	STEP	PROCESSING	STEP	OUTPUT
		DEFRDB(continued)		
ICHAR,IDATA	12	Call a module (DEFATT) to put the DMRS command into the DMRS dynamic region.	13	IRC
IRC	13	If an error occurred go to step 15; else continue.	15/14	
LID	14	Call a module (INSENT) to insert the data into the database.	15	
IRC	15	Put the return code in a buffer and send it to the CI task.	R	IOUT

3.3.2.4 Database Access Main Module (ITIP)

This is the main routine for the ITIP program. This module receives nodes from a CI task and then directs processing according to an option number received in the node.

Calling sequence:

None

Modules which call ITIP:

None

Modules called by ITIP:

SETAST, LKTCSI, INAS, SETEF, RESUME, WAITFR, CLREF, VRECVM, WAIT, VSENDM, DEFRDB, MODRDB, NID, RETPDB, STOPDB, DEFPDB, DEFRAB, MODRAB, DEAS

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ITIP		
	1	Call a module (SETAST) to establish a receive data AST.	2	
	2	Call a module (LKTCSI) to put the database name into proper format for attribute services (AS) software.	3	IFILE,IRC
IRC	3	If an error occurred write a message and go to step 24; else continue.	24/4	
IFILE,REGNAM	4	Call a module (INAS) to initialize the AS interface.	5	IRC,REGNAM
IRC	5	If an error occurred write a message and go to step 24; else continue.	24/6	
REVF	6	Call a module (SETEF) to set an event flag which will signal a node is queued to the ITIP task.	7	
	7	Go to step 9.	9	
CITASK	8	Call a module (RESUME) to resume the CI task.	9	
REVF	9	Call a module (WAITFR) to wait for the event flag (REVF), which indicates a node is queued, to be set by the AST service routine.	10	
REVF	10	Call a module to clear the event flag.	11	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ITIP(continued)		
	11	Call a module to receive the node which has been queued to the ITIP task.	12	IOP,ISC, CITASK,RARG, IARG,ISTRNG
IOP,ISC	12	If this is a request to delay call a module (WAIT) to wait the indicated number of seconds (ISC), send a node back to the CI task and go to step 8; else continue.	8/13	
IOP,ISC, IARG,RARG, ISTRNG,CITASK	13	If this is a request to modify a RDB call a module (MODRDB) to make the modification and then go to step 8; else continue.	8/14	IERR
IOP,IARG	14	If this is a request to retrieve a node ID call a module (NID) to get the node ID and center coordinates, then go to step 8; else continue.	8/15	IERR
IOP,ISC, RARG,IARG, ISTRNG,CITASK	15	If this is a request to to get PDB data call a module (RETPDB) to get the desired data and then go to step 8; else continue.	8/16	IERR
IOP	16	If this is not a request to create a default PDB go to step 19; else continue.	19/17	
ISC,IARG, RARG,ISTRNG, CITASK	17	Call a module (DEFPDB) to create the default PDB.	18	IERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		ITIP(continued)		
	18	If an error occurred go to step 24; else go to step 8.	24/8	
IOP	19	If this is not a request to define a RAB go to step 22; else continue.	22/20	
ISC,ISTRNG, IARG,RARG, CITASK	20	Call a module (DEFRAB) to define the RAB.	21	IERR
	21	If an error occurred go to step 24; else go to step 8.	24/8	
IOP,ISC,IARG, RARG,ISTRNG, CITASK	22	If this is a modify RAB request call a module (MODRAB) to modify the RAB and then go to step 8; else continue.	8/23	IERR
IOP	23	If this is an option to stop the ITIP system go to step 25.	25	
	24	Detach from the AS interface, resume the CI task, print a message to indicate an abnormal exit and exit.	S	
	25	Detach from the AS interface, resume to CI task, print a message indicating a normal exit and exit.	S	

3.3.2.5 Modify a RAB Module (MODRAB)

The function of this module is to modify a predefined resource activity block.

Calling sequence:

Call MODRAB (ISC, IARG, RARG, ISTRNG, CITASK, IRC)

Modules which call MODRAB:

ITIP

Modules called by MODRAB:

DEFREV, DEFEN1, GETATT, DEFATT, CHGENT

INPUT	STEP	PROCESSING	STEP	OUTPUT
		MODRAB		
	1	Set the return code to indicate no error, set the default attribute value and length, set the retrieved resource ID and mode ID to blanks.	2	IRC,IOUT, IATT,IATT2
ISC	2	If the request is not to determine if the RAB is defined go to step 10; else continue.	10/3	
	3	Define the initial linkage ID for beginning the search for the RAB and set the return code to indicate RAB not found.	4	LID,IOUT
	5	Call module (DEFREV) to define the review table for the RAB entity.	6	IRC
IRC	6	If an error occurred go to step 17; else continue.	17/7	
ISTRNG,LID	7	Loop through all defined RAB's to search for the particular RAD desired.	8	IOUT
	9	Go to step 17.		
ISC	10	If this is not a request to retrieve data from the RAB go to step 13; else continue.	13/11	
LID,IARG(1)	11	Call a module (GETATT) retrieve the desired attribute value.	12	IOUT,IRC
	12	Go to step 17.	17	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		MODRAB(continued)		
IARG, ISTRNG	13	Move the attribute number and its value to DMRS buffer so that it can be stored.	14	IDAT
IDAT,ICHAR	14	Call a module (DEFATT) to move the DMRS command into DMRS dynamic region.	15	IRC
IRC	15	If an error occurred go to step 17; else continue.	17/16	
LID	16	Call a module (CHGENT) to store the data into the database.	17	IRC
IOUT,IRC	17	Call a module (VSENDM) to send the return code and any requested attribute value to the CI task.	R	

3.3.2.6 Modify a RDB Module (MODRDB)

The function of this module is to modify a Resource Description Block (RDB).

Calling sequence:

Call MODRDB (ISC, IARG, RARG, ISTRNG, CITASK, IRC)

Modules which call MODRDB:

ITIP

Modules called by MODRDB:

DEFREV, DEFEN1, GETATT, DEFATT, CHGENT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		MODRDB		
ISC	1	If it is desired to store data into a particular RDB go to step 11; else continue.	11/2	
ISC	2	If it is desired to retrieve data from a particular RDB go to step 10; else continue.	10/3	
ISC	3	If it is desired to see if a particular RDB exist go to step 5; else continue.	5/4	
	4	Write an error message and go to step 15.	15	
LID	5	Call a module (DEFREV) to define a review table so that attributes can be retrieved.	6	IRC
IRC	6	If an error occurred go to step 15; else continue.	15/7	
ISTRNG	7	Search for the particular RDB.	8	IRC
IRC	8	If it is found set the return code to so indicate; else set the return code to indicate not found.	9	IRC
	9	Go to step 15.	15	
LID,IARG(1)	10	Call a module (GETATT) to retrieve the desired attribute number and value.	15	IOUT,IRC

INPUT	STEP	PROCESSING	STEP	OUTPUT
		MODRDB(continued)		
ISTRNG,IARG	11	Build a DMRS command using the attribute number and value.	12	IDAT,ICHAR
ICHAR,IDAT	12	Call a module (DEFATT) to move the command to the DMRS dynamic region.	13	IRC
IRC	13	If an error occurred go to step 15; else continue	15/14	
LID	14	Call a module (CHGENT) to modify the attribute value.	15	IRC
IRC,IOUT,LEN	15	Call a module (VSENDM) to send the return code and any attribute value requested back to the CI task.	16	IDS
IDS	16	If an error occurred write a message.	17	
	17	Return	R	

3.3.2.7 Retrieve Node ID and Center Module (NID)

The function of this module is to retrieve the node ID and node center coordinates given a DMRS linkage ID.

Calling sequence:

Call NID (CITASK, LID, IRC)

Modules which call NID:

ITIP

Modules called by NID:

DEFREV, GETATT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		NID		
LID(1)	1	If the linkage ID does not indicate a NDB entity go to step 14; else continue.	14/2	
	2	Call a module (DEFREV) to define a review table for the NDB entity.	3	IRC
IRC	3	If an error occurred go to step 14; else continue.	14/4	
LID	4	Call a module (GETATT) to retrieve the node ID.	5	NC,IDATA, IRC
IRC	5	If an error occurred go to step 14; else continue.	14/6	
IDATA,NC	6	Store the node ID in an output buffer.	7	IOUT
LID	7	Call a module (GETATT) to retrieve the node center X coordinate.	8	NC,IDATA,IRC
IRC	8	If an error occurred go to step 14; else continue.	14/9	
IDATA,NC	9	Store the X coordinate in the output buffer.	10	ICPT(1)
LID	10	Call a module (GETATT) to retrieve the node center Y coordinate.	11	NC,IDATA,IRC
IRC	11	If an error occurred go to step 14; else continue.	14/12	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		NID(continued)		
IDATA,NC	12	Store the Y coordiante in the output buffer.	13	ICPT(2)
IOUT	13	Swap the words of the X and Y coordiantes so that they will be in IGDS coordinates.	14	IOUT
IOUT,CITASK	14	Call a module (VSENDM) to send the output buffer to the CI task.	15	IDS
IDS	15	If an error occurred output a message.	16	
	16	Return.	R	

3.3.2.8 Retrieve PDB Data Module (RETPDB)

The function of this module is to retrieve data from the Project Description Block (PDB).

Calling sequence:

Call RETPDB (ISC, IARG, ISTRNG, CITASK, IRC)

Modules which call RETPDB:

ITIP

Modules called by RETPDB:

DEFREV, GETATT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		RETPDB		
	1	Define the linkage ID, default attribute value and attribute length.	2	LID,CHAR, NC
	2	Call a module (DEFREV) to define the review table so that attributes can be retrieved.	3	IRC
IRC	3	If an error occurred go to step 7.	7/4	
LID,ISC	4	Call a module (GETATT) to retrieve the desired attribute value.	5	NC,CHAR,IRC
IRC	5	If an error occurred go to step 7.	7/6	
	6	Set the return code to to indicate the attribute value was retrieved.	7	IERR
IOUT,LEN	7	Call a module (VSENDM) to send the return code and attribute value back to the CI task.	8	IDS
IDS	8	If an error occurred write a message.	9	
	9	Return	R	

3.3.2.9 Establish Receive Data AST Module (SETAST)

The function of this module is to set up a receive data AST. This module also contains the service routine which is called when the AST occurs.

Calling sequence:

Call SETAST

Modules which call SETAST:

ITIP

Modules called by SETAST:

SRDA$, SETF$, ASTX$

INPUT	STEP	PROCESSING	STEP	OUTPUT
		SETAST		
		Entry Point SETAST		
	1	**Call a module (SRDA$C) to establish a receive data AST.**	**R**	
		Entry Point RDAST		
	1	**Call a module (SETF$C) to set event flag one.**	**2**	
	2	**Call a module (ASTX$) to exit the AST service routine.**	**R**	

3.3.2.10 Store PDB Data Module (STOPDB)

The function of this module is to store data values into the Project Description Block (PDB).

Calling sequence:

```
Call STOPDB (ISC, IARG, RARG, ISTRNG, CITASK, IERR)
```

Modules which call STOPDB:

ITIP

Modules called by STOPDB:

DEFEN1, DEFATT, CHGENT, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		STOPDB		
ISC,ISTRNG IARG(1)	1	Build DMRS COMMAND to store the attribute value in the PDB.	2	IBYTE,ICHAR
	2	Set up the linkage for the PDB	3	LID
LID	3	Call a module (DEFEN1) to define the prototype.	4	IERR
ICHAR,IBYTE	4	Call a module (DEFATT) to put the command into the DMRS dynamic region.	5	IERR
LID	5	Call a module (CHGENT) to modify the data in database.	6	IERR
IERR	6	Send the return code to the CI task.	R	IOUT

3.3.2.11 Extract EDB Module (EXTEDB)

The function of this module is to extract the edge description data and write it to the TRA1 file.

Calling sequence:

Call EXTEDB (NPAREN, IPT, IERR)

Modules which call EXTEDB:

EXTFIL

Modules called by EXTEDB:

DMREQ

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTEDB		
CBREQU	1	Create a universe file which contains all EDB's	2	CBNSIZ, CBRETC
CBUSIZ	2	Write the number of EDB's to the TRAP file.	3	
CBUSIZ, CBRETC	3	If there are no EDB's go to step 11; else continue.	11/4	
CBREQU	4	Ready the universe for access.	5	CBRETC
CBRETC	5	If an error occurred go to step 11; else continue.	11/6	
CBREQU	6	Get the first/next entry from the universe file.	7	CBRETC
CBRETC	7	If an error occurred or no more EDB's exist, go to step 11; else continue.	11/8	
	8	Write a dummy edge ID to the TRAP file. (The ID is not used).	9	
CBTXT, CBLNG	9	Get the edge predecessor and successor and write them to the TRAP file.	10	
	10	Go to Step 6.	6	
	11	Set the return code and return	R	IERR

3.3.2.12 Extract TRAP File Control Module (EXTFIL)

The function of this module is to control the extraction of data for the TRAP file.

Calling sequence:

Call EXTFIL (CITASK, NLEV, NPAREN, IERR)

Modules which call EXTFIL:

EXTMN

Modules called by EXTFIL:

DMREQ, EXTPDB, EXTRDB, EXTNDB, EXTEDB, VSENDM

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTFIL		
	1	Open the output TRAP file.	2	
LENPDB,LENRDB LENNDB,LENRAB, LENEDB	2	Write the length of the PDB,RDB,NDB,RAB and EDB to the TRAP file.	3	
LOCRDB	3	Write the pointer to the first RDB to the TRAP file.	4	
CBREQU	4	Set the HOL CB request code and call a module to attach to DMRS.		CBRETC
CBRETC,RTNSUC	5	If an error occurred go to 17; else continue.	17/6	
CBREQU,CBTXT CBLNG	6	Set the HOL CB request code and call a module (DMREQ) to define the database.	7	CBRETC
CBRETC,RTNSUC	7	If an error occurred go to 17; else continue.	17/8	
	8	Set a pointer to indicate the next record to be written in the TRAP file.	9	IPT
NLEV,IPT	10	Call a module (EXTPDB) to to extract the PDB data and write it to the TRAP file.	11	IERR
IERR	11	If an error occurred go to 17; else continue.	17/12	
IPT	12	Call a module (EXTRDB) to extract the RDB data and write it to the TRAP file.	13	IERR

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTFIL(continued)		
IERR	13	If an error occurred go to 17; else continue.	17/14	
NPAREN,IPT	14	Call a module (EXTNDB) to extract the NDB's and their associated RAB's and write the data to the TRAP file.	15	IERR
IERR	15	If an error occurred go to 17; else continue.	17/16	
NPAREN,IPT	16	Call a module (EXTEDB) to extract the EDB data and write it to the TRAP file.	17	IERR
IERR,CITASK, IOUT	17	Store the return code in a buffer and send the buffer to the CI task.	18	IDS
IDS	18	If an error occurred sending data write a message.	19	
CBREQU	19	Close the TRAP file and detach from DMRS.	R	

3.3.2.13 Extract TRAP File Main Module (EXTMN)

The function of this module is to receive a node from the CI task and then call a module which controls the collection of data for the TRAP file.

Calling sequence:

None - This is the main routine.

Modules which call EXTMN:

None

Modules called by EXTMN:

EXTFIL, VRECVM, RESUME

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTMN		
	1	Call a module (VRECVM) to receive a node from the CI task.	2	IOP, CITASK, IARG, IDS
IDS	2	If an error occurred go to step 5; else continue.	5/3	
IARG,CITASK, IOP	3	If the request is to extract a TRAP file call a module (EXTFIL) to do the extraction; else continue.	4	IERR
CITASK, IOP	4	If the request is to submit a TRAP file call a module (SUBFIL) to do so; else continue.	5	IERR
	5	Call a module (RESUME) to resume the CI task and stop.	S	

3.3.2.14 Extract NDB Module (EXTNDB)

The function of this module is to extract the Node Description Blocks and write them to the TRAP file. The Resource Activity Blocks associated with each NDB are also extracted and written to the TRAP file.

Calling sequence:

```
Call EXTNDB (NPAREN, IPT, IERR)
```

Modules which call EXTNDB:

EXTFIL

Modules called by EXTNDB:

DMREQ

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTNDB		
CBREQU	1	Get the total number of RAB's so that the pointer to the first EDB can be calculated.	2	NUMRAB, CBRETC
CBREQU	2	Get the universe of all NDB's.	3	CBUSIZ
IPT, NUMRAB, CBUSIZ	3	Calculate the pointer to the first EDB and save the pointer to the beginning to this NDB.	4	IPT, ISAV
IPT	4	Write the pointer to the first EDB to the TRAP file.	5	
CBUSIZ	5	If no NDB's exist go to step 31; else continue.	31/6	
CBREQU	6	Ready the universe of NDB's for access.	7	CBRETC
CBRETC	7	If an error occurred, go to step 31; else continue.	31/8	
CBREQU	8	Get the first/next entry from the NDB universe [illegible].	9	CBRETC
CBRETC	9	If an error occurred or no more NDR's exist, go to step 31; else continue.	31/10	
CBREQU, CBANUM, CBENUM	10	Get the node ID for this entry.	11	CBTXT, CBLNG, CBRETC, IDLEN, ID
CBRETC	11	If an error occurred, go to step 31; else continue.	31/12	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTNDB - CONTINUED		
CBTXT, CBLNG	12	Get a universe of all RAB's associated with this node.	13	CBRETC, CBUSIZ NUMRAB
CBRETC, CBUSIZ	13	If no RAB's exist reposition the NDB universe file to point to the current entry; else continue.	14	
NUMRAB	14	Calculate a pointer to the next NDB in the TRAP file.	15	NXTNDB, ISAV
ID, IDLEN, NXTNDB	15	Write the pointer and the node ID to the file.	16	
CBREQU CBANUM	16	Get the node description and shift factor and write them to the TRAP file.	17	CBTXT, CBLNG, CBRETC
CBREQU, CBANUM	17	Get the node duration.	18	IHOLD, LEN CBRETC
CBRETC	18	If an error occurred in step 16 or 17, go to step 31; else continue.	31/19	
CBREQU, CBANUM	19	Get the node flags.	20	CBRETC
CBRETC	20	If an error occurred go to step 31; else continue.	31/21	
IDLEN, ID, RVAL, CBTXT, CBLNG	21	Write the node flags and node duration to the TRAP file.	22	
NUMRAB	22	Write the number of RAB's for this NDB to the TRAP file.	23	
NUMRAB	23	If the number of RAB's is zero go to step 8; else continue.	8/24	

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTNDB - CONTINUED		
CBREQU	24	Get the first/next entry from the RAB universe file	25	CBRETC
CBRETC	25	If an error occurred, go to step 31; else continue.	31/26	
CBRETC	26	If no more RAB's exist for this NDB go to step 30; else continue.	30/27	
CDREQU, CBANUM	27	Get the RAB ID, quantity and code and write them to the TRAP file.	28	CBRETC, CBTXT CBLNG
CBRETC	28	If an error occurred go to step 31; else continue.	31/30	
	29	Go to step 24.	24	
	30	Go to step 8.	8	
	31	Set the return code and return.	R	IERR

3.3.2.15 Extract the PDB Module (EXTPDB)

The function of this module is to extract the project description information and write it to the TRAP file.

Calling sequence:

```
Call EXTPDB (NLEV, IPT, IERR)
```

Modules which call EXTPDB:

EXTFIL

Modules called by EXTPDB:

DMREQ

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTPDB		
IPT	1	Adjust the record pointer for the TRAP file to point to the first RDB.	2	IPT
IFPDB,CBTXT CBLNG	2	Get the universe of PDB's from the DMRS database.	3	CBRETC
CBRETC	3	If an error occurred go to step 14; else continue.	14/4	
CBREQU,CBUNIV	4	Ready the universe of PDB's for access.	5	CBRETC
CBRETC	6	If an error occurred go to 14; else continue.	14/7	
CBREQU	7	Get the first entry from the universe file.	8	CBRETC
CBRETC	8	If an error occurred go to step 14; else continue.	14/9	
CBANUM,CBTXT	9	Get the project title, network ID and network description from the database and write them to the TRAP file.	10	CBRETC
NLEV	10	Write the network level to the TRAP file.	11	
CBANUM,CBTXT	11	Get the network start time from the database and write it to the TRAP file.	12	CBRETC
CBANUM,CBTXT	12	Get the time units from the database and write it to the TRAP file.	13	CBRETC

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTPDB(continued)		
CBANUM,CBTXT	13	Get the network duration from the database and write it to the TRAP file.	14	CBRETC
	14	Release and delete the universe file.	R	

3.3.2.16 Extract RDB Module (EXTRDB)

The function of this module is to extract the Resource Description data from the database and write it to the TRAP file.

Calling sequence:

```
Call EXTRDB (IPT, IERR)
```

Modules which call EXTRDB:

EXTFIL

Modules called by EXTRDB:

DMREQ

INPUT	STEP	PROCESSING	STEP	OUTPUT
		EXTRDB		
CBRETC, IFRDB	1	Create a universe which contains all of the RDB's.	2	CBUSIZ
IPT, CBUSIZ	2	Update the TRAP file pointer to point to the first record with node data.	3	IPT
IPT, CBUSIZ	3	Write the pointer and number of RDB's to the TRAP file.	4	
CBUSIZ	4	If the universe is empty, go to step 11; else continue.	11/5	
CBREQU	5	Ready the universe of RDB's for access.	6	CBRETC
	6	If an error occurred, go to step 11; else continue.	11/6	
CBREQU	7	Get the first/next entry from the universe file.	8	CBRETC
CBRETC	8	If an error occurred or all entries have been processed, go to step 11; else continue.	11/8	
CBANUM, CBENUM	9	Get the attribute values for the resource ID, description, flags, constraint level, initial quantity and function and write them to the TRAP file.	10	CBTXT, CBLNG, IFLAGS, IFLG1, IFLG2, IFLG3, LEN
	10	Go to step 7.	7	
	11	Set the return code and return.	R	IERR

3.3.2.17 Submit TRAP File Module (SUBFIL)

The function of this module is to submit the TRAP file across the PDP/SIGMA interface.

Calling sequence:

Call SUBFIL

Modules which call SUBFIL:

EXTMN

Modules called by SUBFIL:

None

INPUT	STEP	PROCESSING	STEP	OUTPUT
		SUBFIL		
	1	Open the TRAP file to be transferred.		
	2	If an error occurred set the return code and go to step 4; else continue.	4/3	IERR
	3	Transmit the file to the SIGMA computer.	4	
CITASK, IERR	4	Call a module to send the return code to the CI task.	R	

3.4 ITIP Data Base Characteristics

Intergraph's Data Management and Retrieval System (DMRS) is used to store the data which is associated with the network flow diagrams generated by IGDS. General information about DMRS can be found by referencing the DMRS documents listed in section 2 of this document. This section will discuss the specific structure of the data base used to support the ITIP. Figure 3.4-1 defines the structure of the data base. This figure shows a subset of the statements that would be necessary to generate the data base using the Data Description Language (DDL) compiler. The following subsections will discuss the five entities which are shown in Figure 3.4-1.

```
1.  PDB     P = 0              DF = 'PDB.ENT'        OCC = 2

            .1  P_TITLE        F=AN(30)              ; PROJECT TITLE
            .2  N_ID           F=AN(4)               ; NETWORK ID
            .3  N_TITLE        F=AN(30)              ; NETWORK TITLE
            .4  N_LEVEL        F=I(16)               ; NETWORK LEVEL
            .5  S_TIME         F=F                   ; NETWORK START TIME
            .6  U_TIME         F=AN(8)               ; TIME UNITS
            .7  N_DURATION     F=F                   ; NETWORK DURATION
            .8  EX_INDEX       F=I(9999)             ; NODE EXPANSION
                                                       INDEX

2.  RDB     P = 1              DF = 'RDB.ENT'        OCC = 50

            .1  R_ID           F=AN(2)               ; RESOURCE ID
            .2  R_DESC         F=AN(24)              ; RESOURCE DESCRIPTION
            .3  R_FLAGS        F=I(10)               ; RESOURCE FLAGS
            .4  R_CONST        F=F                   ; RESOURCE CONSTRAINTS
            .5  R_INIT_QTY     F=F                   ; RESOURCE INITIAL QUANTI
            .6  R_FUNC         F=I(10)               ; FUNCTION INDEX

3.  NDB     P = 1              DF = 'NDB.ENT'        OCC = 2400

            .1  ND_ID          F=I(9999)             ; NODE ID
            .2  ND_DESC        F=AN(24)              ; NODE DESCRIPTION
            .3  ND_SHF_FAC     F=I(10)               ; NODE SHIFT FACTOR
            .4  ND_FLAGS       F=I(10)               ; FLAGS
            .5  ND_DUR         F=F                   ; NODE DURATION
            .6  ND_NUM_RABS    F=I(12)               ; NO. OF RABS FOR NODE
            .7  ND_LEV         F=I(8)                ; NODE LEVEL
            .8  ND_PAREN       F=I(9999)             ; NODES PARENT
            .9  ND_XUR         F=I(2147000000)       ; NODE CENTER XUR
            .10 ND_YUR         F=I(2147000000)       ; NODE CENTER YUR

4.  EDB     P = 1              DF = 'EDB.ENT'        OCC = 1000

            .1  E_ID           F=AN(4)               ; EDGE ID
            .2  E_PRED         F=I(9999)             ; EDGE PREDESESSOR
            .3  E_SUCC         F=I(9999)             ; EDGE SUCCESSOR
            .4  E_PAREN        F=I(9999)             ; PAREN OF THIS EXPANSION

5.  RAB     P = 3              DF = 'RAB.ENT'        OCC = 1000

            .1  RA_ID          F=AN(2)               ; RESOURCE ACTIVITY B ID
            .2  RA_CODE        F=AN(1)               ; RESOURCE ACTIVITY CODE
            .3  RA_QTY         F=F                   ; QUANTITY
            .4  RA_PAREN       F=I(9999)             ; NODE WHICH OWNS RAB
            .5  EX_PAREN       F=I(9999)             ; EXPANSION PARENT
```

FIGURE 3.4-1

3.4.1 Project Description Block (PDB)

The PDB entity has eight attributes. The first is labeled P_TITLE and is the project title. The project title is an ASCII string whose maximum length can be thirty characters. The second attribute is labeled N_ID and is the network identification. The network identification is an ASCII string of not more than four characters. The third attribute is labeled N_TITLE and is the network title. The network title is an ASCII string of thirty characters or less. The fourth attribute is labeled N_LEVEL and is the network level. The network level is an integer with a value from zero to sixteen. The fifth attribute is labeled S_TIME and is the network start time. The start time is a floating point value. The sixth attribute is labeled U_TIME and is the units of time. The units of time is an ASCII string of eight characters or less. The seventh attribute is labeled N_DURATION and is the network duration. The network duration is a floating point value. The eighth attribute is labeled EX_INDEX and is the expansion index. This index points to the next design space to be allocated for a node expansion. The expansion index is an integer in the range of 0 to 9999. These eight attributes collectively define the PDB. Attribute eight is used internally to support ITIP processing. The other attributes are used when a TRAP file is created.

3.4.2 Resource Description Block (RDB)

The RDB entity has six attributes. The first attribute is the resource identification and is labeled R_ID. It is an ASCII string of one or two characters. The second attribute is the resource description and is labeled R_DESC. It is also an ASCII string but the length must be twenty-four or fewer characters. The third attribute is the resource flags variable. This is a composite flag variable represented by an integer in the data base. Three bits in the word are used and are counted from the left. They are, from the left, the sum check flag, value type flag and the constrained resource flag. These flags are defined in the TRAP documentation. The fourth attribute is the resource constraint level and is labeled R_CONST. It is a floating point value. The fifth attribute is the initial quantity of the resource and is labeled R_INIT_QTY. It is also a floating point value. The sixth attribute is the function index and is labeled R_FUNC. It is an integer value from zero to ten. These six attributes define each resource. The above information is stored in the data base for each resource defined by the user.

3.4.3 Node Description Block (NDB)

The NDB entity is defined by ten attributes. The first five are extracted for the TRAP file. The last five are used internally by the ITIP system. The first attribute is the node identification and is labeled ND_ID. It is an integer in the range of 0 to 9999. This is the unique node identification. The second attribute is the node description and is labeled ND_DESC. The node description is an ASCII string of 24 or fewer characters. The third attribute is the shift factor and is labeled ND_SHF_FAC. The shift factor should be an integer and may assume a value of one, two or three. The fourth attribute is the node discrete flags and is labeled ND_FLAGS. The flags variable is as defined in the TRAP documentation. This is a composite flag variable and is an integer. The fifth attribute is the node duration and is labeled ND_DUR. The node duration is a floating point value. The sixth attribute is the number of Resource Activity Blocks associated with the node and is labeled ND_NUM_RABS. It is an integer value in the range of zero to twelve. This attribute is not used and will have a value of -128 for each node defined. The seventh attribute is the node level and is labeled ND_LEV. The node level is an integer value and indicates the network level of the node. The eighth attribute is the node parent or owner and is labeled ND_PAREN. This is also an integer value which contains the ID of the node that is being expanded. The ninth and tenth attributes labeled ND_XUR and ND_YUR are the node X and Y coordinates. These coordinates are in IGDS units. These ten attributes define a node and up to 9999 nodes can be defined in the NDB.

3.4.4 Edge Description Block (EDB)

The EDB entity has four attributes. The first attribute is the edge identification and is labeled E_ID. The edge ID is an ASCII string of four or less characters. The edge ID is not used since attributes two and three uniquely identify the edge. The second attribute is the predecessor node for the edge and is labeled E_PRED. The third attribute is the successor node for the edge and is labeled E_SUCC. The second and third attributes are integer values in the range of 0 to 9999. They are the node identifications for the nodes involved. The fourth attribute is the node identification for the node which owns the expansion that this edge is a part of and is labeled E_PAREN. This is also an integer in the range of 0 to 9999. The fourth attribute is used only by the ITIP system and is not extracted for the TRAP file. These four attributes define an edge and up to 4096 edges can be defined in the EDB.

3.4.5 Resource Activity Block (RAB)

The RAB also called a Resource Activity Descriptor (RAD) is defined by five attributes. The first attribute is the resource identification code and is labeled RA_ID. This is a one or two character ASCII string. The second attribute is the resource activity code and is labeled RA_CODE. This is a single ASCII character and is defined in the TRAP documentation. The third attribute is the resource activity quantity and is labeled RA_QTY. The resource activity quantity is a floating point value. The fourth attribute is the RAB parent and is labeled RA_PAREN. This is an integer in the range of 0 to 9999 and identifies the node with which the RAB is associated. The fourth attribute is the expansion owner or parent and is labeled EX_PAREN. This is an integer value in the range of 0 to 9999. This value identifies the node whose expansion the RAB is a part of. The first three attributes are extracted for the TRAP file while the last two are used internally by ITIP.

APPENDIX A

TASK BUILD COMMAND FILES

Appendix A describes the PDP command files used by the PDP Task Builder to produce the two FORTRAN tasks which support ITIP operation. The command file listed below is used to build the ITIP.TSK task.

```
ITIP, ITIP/-SP=ITIP, NID, RDAST, PETPDB, STOPDB, DEFPDB
                DEFRDB, MODRDB, DEFRAB, MODRAB
                   QS0:  [15,100]UTILITIES/LB
                QS2:  [50,2]ASIS/LB

/
WNDWS=1
VSECT=ASREGN:160000
UNITS=6
ASG=TT0:6
ASG=NL:2:3:4
GBLDEF=RSX11$:1
//
>
```

This command file requires two libraries. The IGDS utilities library is named UTILITIES.OLB and resides in account QS0:[15,100]. The Attribute Services interface library is named ASIS.OLB and resides in account QS2:[50,2].

The command file listed below is used to build the IEXT.TSK task.

```
IEXT=EXTMN, EXTFIL, EXTPDB, EXTRDB, EXTNDB, EXTEDB, SUBFIL
     QS:[14,35]DMRS/LB:FORAST
          QS:[14,35]DMRS/LB

/
UNITS=6
ASG=TT0:6
ASG=SY:4
//
>
```

This command file requires the DMRS library, which is named DMRS.OLB and resides in account QS0:[14,35].

APPENDIX B

ITIP SOFTWARE DELIVERY MEDIUM

Appendix B describes the ITIP software delivery medium. The ITIP software resides in the QS2:[50,2] account on the PDP-11/70 in Building 4487, MSFC. The official delivery of this software is on a magnetic tape which can be used to recreate the QS2:[50,2] account. Table B.1 lists the files which reside on the delivery tape.

The files in Table B.1 are identified by the three-character file extension. The files with extensions of UCM, FTN and MAC are source modules which are described in Section 3 of the document. The COM files define the intermodule COMMON variables used by the FORTRAN source modules. The files with extensions of CMD and OLB identify the command and library files, respectively, which are used to build the FORTRAN tasks. The TSK files are the FORTRAN task files and the CEL file is the cell library. The DDL file is the Data Definition Language input file used to build the ITIP DMRS data base. The SEED.DGN file defines the default IGDS design file conditions.

Table B.1 ITIP Delivery Modules

```
DIRECTORY MT0
18-SEP-81 11:42

ITIP UCM 1          6          18-SEP-81 00:00
IRES UCM 2          9          18-SEP-81 00:00
ISWC UCM 1          5          18-SEP-81 00:00
IZMI UCM 1          5          18-SEP-81 00:00
IZMO UCM 1          5          18-SEP-81 00:00
IRML UCM 1          6          18-SEP-81 00:00
IDWC UCM 1          5          18-SEP-81 00:00
IDED UCM 1          5          18-SEP-81 00:00
IMRA UCM 1          8          18-SEP-81 00:00
IDRS UCM 1          8          18-SEP-81 00:00
ISPO UCM 1          7          18-SEP-81 00:00
ARO UCM 1           5          18-SEP-81 00:00
IMPO UCM 1          6          18-SEP-81 00:00
IMRS UCM 1          9          18-SEP-81 00:00
IMNO UCM 1          5          18-SEP-81 00:00
INIT UCM 1          7          18-SEP-81 00:00
IMPO2 UCM 1         4          18-SEP-81 00:00
IRNI UCM 1          6          18-SEP-81 00:00
IPNO UCM 1          9          18-SEP-81 00:00
IPRM UCM 1          7          18-SEP-81 00:00
IRPL UCM 1          6          18-SEP-81 00:00
IPED UCM 1          9          18-SEP-81 00:00
IDNO UCM 1          6          18-SEP-81 00:00
IMNO2 UCM 1         7          18-SEP-81 00:00
ITRM UCM 1          4          18-SEP-81 00:00
ISNO UCM 1          8          18-SEP-81 00:00
ISTF UCM 1          4          18-SEP-81 00:00
IEXT UCM 1          4          18-SEP-81 00:00
IEXP UCM 1          8          18-SEP-81 00:00
IDRA UCM 1          8          18-SEP-81 00:00
EXTMN FTN 1         4          18-SEP-81 00:00
EXTFIL FTN 1        8          18-SEP-81 00:00
EXTPOB FTN 1        8          18-SEP-81 00:00
EXTROB FTN 1        8          18-SEP-81 00:00
EXTNOB FTN 1        18         18-SEP-81 00:00
EXTEOB FTN 1        7          18-SEP-81 00:00
SUBFIL FTN 1        1          18-SEP-81 00:00
ITIP FTN 1          11         18-SEP-81 00:00
DEFWOB FTN 1        10         18-SEP-81 00:00
MODROB FTN 1        9          18-SEP-81 00:00
NIO FTN 1           7          18-SEP-81 00:00
RETPOB FTN 1        5          18-SEP-81 00:00
STOPOB FTN 1        7          18-SEP-81 00:00
DEFPOB FTN 1        5          18-SEP-81 00:00
DEFRAB FTN 1        9          18-SEP-81 00:00
MODRAB FTN 1        10         18-SEP-81 00:00
SETAST MAC 1        2          18-SEP-81 00:00
RDAST MAC 1         2          18-SEP-81 00:00
ITIP COM 1          1          18-SEP-81 00:00
AS COM 1            1          18-SEP-81 00:00
ASCOM COM 1         1          18-SEP-81 00:00
ITIPBLD CMD 1       1          18-SEP-81 00:00
IEXTBLD CMD 1       1          18-SEP-81 00:00
LOGIN CMD 1         1          18-SEP-81 00:00
ASIS OLB 1          41         18-SEP-81 00:00
ITIP TSK 1          68         18-SEP-81 00:00
IEXT TSK 1          80         18-SEP-81 00:00
ITIP CEL 1          15         18-SEP-81 00:00
COLGAS DDL 1        4          18-SEP-81 00:00
SEED DGN 1          5          18-SEP-81 00:00

TOTAL OF 541 /541  BLOCKS IN 60  FILES
```