AD-A031 180   FEDERAL AVIATION ADMINISTRATION WASHINGTON D C SYSTE--ETC F/G 17/7
              CENTRAL FLOW CONTROL COMPUTER PROGRAM SPECIFICATIONS. VOLUME II--ETC(U)
              SEP 76 R BALES, G BEEKER, C BROGLIO, F CASEY
UNCLASSIFIED   FAA-RD-76-157-VOL-3   NL

1 OF 1
AD
A031180

END
DATE
FILMED
11-76

Report No: FAA-RD-76-157, III

AD A031180

CENTRAL FLOW CONTROL COMPUTER PROGRAM SPECIFICATIONS:
VOLUME III
OFF-LINE SUPPORT SUBSYSTEM SPECIFICATION

Central Flow Control Design Team
Federal Aviation Administration

September 1976
Final Report

D D C
OCT 27 1976
A

Document is available to the public through the
National Technical Information Service,
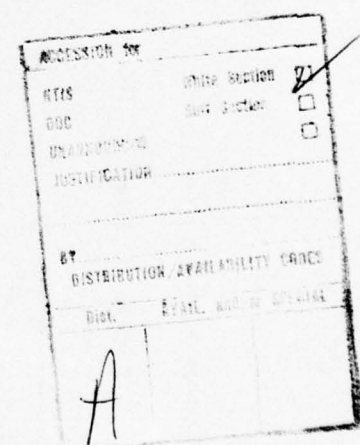Springfield, Virginia 22161.

Prepared for

U.S. DEPARTMENT OF TRANSPORTATION
FEDERAL AVIATION ADMINISTRATION
Systems Research & Development Service
Washington, D.C. 20590

NOTICE

This document is disseminated under the sponsorship of
the Department of Transportation in the interest of infor-
mation exchange. The United States Government assumes no
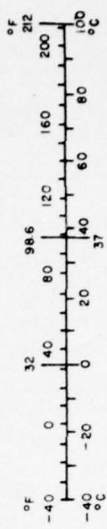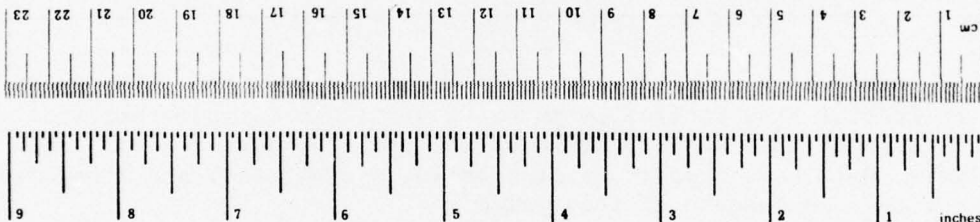liability for its contents or use thereof.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FAA-RD-76-157, III-Vol-3		

4. Title and Subtitle	5. Report Date
Central Flow Control Computer Program Specifications. Volume III. Off-Line Support Subsystem Specification	September 1976
	6. Performing Organization Code
	ARD-102

7. Author(s)	8. Performing Organization Report No.
Central Flow Control Design Team	111-102

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
U.S. Department of Transportation Federal Aviation Administration Systems Research and Development Service Washington, D.C. 20591	
	11. Contract or Grant No.
	111-102
	13. Type of Report and Period Covered

12. Sponsoring Agency Name and Address	
U.S. Department of Transportation Federal Aviation Administration Systems Research and Development Service Washington, D.C. 20591	Final Report.
	14. Sponsoring Agency Code
	ARD-102

15. Supplementary Notes

R. Bales, G. Beeker, C. Broglio, F. Casey Y. Chao   82p.

16. Abstract

This report contains the specifications for the off-line support subsystem of the Central Flow Control Computer Program. This report provides a detailed functional specification of the utility, data recording and analysis, data base support, system build, system test, and management aid software components of the system.

17. Key Words	18. Distribution Statement
Support Software, Functional Specification, Data Recording and Analysis, Software Utilities, Flow Control	Document is available to the public through the National Technical Information Service, Springfield, Virginia 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	87	

Form DOT F 1700.7 (8-72)   Reproduction of completed page authorized

340170

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	*2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in²	square inches	6.5	square centimeters	cm²
ft²	square feet	0.09	square meters	m²
yd²	square yards	0.8	square meters	m²
mi²	square miles	2.6	square kilometers	km²
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft³	cubic feet	0.03	cubic meters	m³
yd³	cubic yards	0.76	cubic meters	m³
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures, Price $2.25, SD Catalog No. C13.10:286.

Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm²	square centimeters	0.16	square inches	in²
m²	square meters	1.2	square yards	yd²
km²	square kilometers	0.4	square miles	mi²
ha	hectares (10,000 m²)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m³	cubic meters	35	cubic feet	ft³
m³	cubic meters	1.3	cubic yards	yd³
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F

°F
−40   0   32   80   98.6   120   160   200   212
−40   −20   0   20   37   40   60   80   100
°C

LIST OF MEMBERS OF THE CENTRAL FLOW CONTROL DESIGN TEAM

	R. Bales	MITRE	T. McCann	AAT-370
	G. Beeker	MITRE	M. McCrory	ARD-141
(1)	C. Broglio, Ph.D.	ARD-102	M. Medeiros	TSC-622
	F. Casey	ARD-141	O. Morganstern	MITRE
	Y. Chao, Ph.D.	FEDSIM	J. Noyes	AAF-630
	R. Davis	FEDSIM	W. Reed	AAF-643
	G. Ellison	ARD-142	J. Richardson	AAT-370
	H. Gabrielli	MITRE	A. Robb	TSC-622
	T. Guyc	BDM	D. Rozzano	ARD-142
	V. Hallows	AAT-510	A. Severino	ARD-142
(2)	T. Hannan	ARD-102	J. Siedsma	AAT-510
	L. Jarze	FEDSIM	R. Watson	MITRE
	G. Kershaw	BDM	H. Whang	TSC-622
	P. MacDonald	TSC-622	G. Wright	FEDSIM
	V. Maglione	TSC-622	R. Wright	TSC-622

(1) Chief Programmer
(2) Back-up Programmer

ACKNOWLEDGEMENT

This specification series was prepared by the members whose names are listed above. In performing this work these members were supported by their respective organizations. Inclusion in the above list, however, does not necessarily imply that either the individual or his organization endorses the report.

CONTRIBUTING ORGANIZATIONS

AAF   - Airway Facilities Service, Federal Aviation Administration
AAT   - Air Traffic Service, Federal Aviation Administration
ARD   - Systems Research and Development Service, Federal Aviation
           Administration
BDM   - The BDM Corporation, Vienna, Virginia
FEDSIM - Federal Computer Performance Evaluation and Simulation Center
MITRE - The MITRE Corporation, McLean, Virginia
TSC   - Transportation Systems Center, Department of Transportation

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF ILLUSTRATIONS

Figures

Tables

1. INTRODUCTION

The CFC support system performs the off-line processing
required in support of the CFC system throughout its
development, implementation, test and maintenance.
The functions provided by the CFC support system
include the following capabilities:

   a. Operational aids

      Provide the capabilities needed to generate
      the operational system and to maintain a
      current data base. In addition, the Data
      Reduction and Analysis (DR&A) capability
      provides the means for automatic analysis
      of past system operations.

   b. Development aids

      Includes the software production tools to
      enhance the efficiency and the timeliness
      of the programming effort. Also, easy to
      use test tools are provided to support a
      comprehensive testing of the system prior
      to cutover and to verify new system
      versions during maintenance.

   c. Control aids

      Provides the tools needed to support the
      overall management of the project during
      the various phases, to maintain control
      over schedules and over the product
      quality, and to enhance communications
      between the various project teams.

The off-line system utilizes the redundant simplex
with its related storage and peripherals.

The CFC support system will utilize many features of
the existing NAS operational support system for the
A3d2 NAS model. Many of the capabilities of the NAS
support system, primarily the utilities group, will
be used to support CFC. Various NAS support programs
may require some modifications before they can be used

1-1

in CFC. In certain areas, for example system build,
the concepts of the NAS support system are retained,
but new programs must be written. These functional
specifications, therefore, have been fashioned after
the NAS functional specification (NAS-MD-347, 348, 349)
and, whenever possible, provide excerpts or adapta-
tions thereof.

1.1 Software

The off-line functions have been divided into five
groups as shown in Figure 1-1. These are:

1. Utilities Group: Includes the operating
system, the compiler, the assembler and similar
processors which are required to facilitate the
production, management and maintenance of pro-
grams and data for both the operational and the
support systems.

2. DR&A Group: Provides the capability to
reduce and analyze the data recorded during
live operation. Comprehensive listings, summaries
and statistical reports are produced to enable
evaluation of past system performance.

3. Static Data Base Group: Assembles the
static data base and the adaptation data that
will be used on-line by the CFC system. The
source data is read, analyzed, updated and
prepared for use in building the operational
system.

4. System Build Group: Provides the capability
to generate and maintain an IPLable disk-resident
CFC system. This system is constructed from
the operational programs, the data bases and
the user defined configuration parameters.

5. System Test Group: Facilitates program
testing, evaluation, training and demonstration.
The system test group provides the mechanism to
drive the operational system with simulated
data without employing the extensive resources
required for a live configuration.

1 UTILITIES GROUP	2 DR&A GROUP	3 STATIC DATA BASE GROUP	4 SYSTEM BUILD GROUP	5 SYSTEM TEST GROUP	6 MANAGEMENT AIDS GROUP

Monitor
Compiler
Assembler
Library
Loader
Sort/Merge
HASP
CRJE
	Reduction
	Sorting
	Reports
		Adaptation
		Static Data
		  Base
			IPLable system
			  Build
			System Update
			Dump/Restore
			ASB
				Interfacility
				Intrafacility
				Automated Unit
				and string test
				drivers
					PERT charts
					Program Analyzer
					HIPO documentation
					Automated Code
					  auditor

FIGURE 1-1 CFC OFF-LINE SUPPORT SYSTEM OVERVIEW

Note: Refer to the appropriate section in this document for a description
of the functions included in each of the groups shown.

1-3

6.   Management Aids Group: Provides the tools
for maintaining control over the progress of
the project during the design, implementation
and maintenance phases. Functions are avail-
able to provide information on the schedules
of the various efforts, the quality of the
code and the structure of the programs.

1.2   Hardware

Table 1-1 lists the hardware requirements for
the off-line support system. Normally, this
hardware is used exclusively by the support system.
Therefore, the support system processing can be
carried out concurrently with the normal processing
of the operational CFC system. However, in case of
a hardware failure in the operational system, any
of the hardware components can be preempted from
the support system. This could result in aborting
whatever processing is going on in the support system
at that time.

1.3   Summary of Support Capabilities

Table 1-2 provides a summary of the CFC support
capabilities. The table also indicates the status
of the support functions in terms of their avail-
ability in the present NAS system. It is important
to note that much of the existing NAS software,
although not directly applicable to the CFC effort,
can and should be utilized for CFC. As an example,
the DR&A requirements (for recorded and reduced data)
are different from those of NAS. However, those por-
tions of the NAS DR&A programs which read and perform
the initial processing of the on-line recorded tapes,
analyze user specifications and prepare the final
reports, can be applied to CFC with a relatively
small programming effort.

1.4   Support Software Programming Requirements

The CFC support software programming effort includes
all the programming requirements as specified in
Attachment A (Programming Requirement) of the RFP.

TABLE 1-1

CENTRAL FLOW CONTROL

OFF-LINE SYSTEM HARDWARE[1]

Number	Element
1	Compute Element (CE)
1	Input/Output Computer Element (IOCE)
1	System Console
4	65K words Storage Elements (1 megabyte)
1	Disk Control Unit
5	Disk Drives
1	Tape Control Unit
8[2]	Tape Drives
1	High Speed Printer
1	Card Reader/Punch
4	Terminals (2 each FOB 10A and Jacksonville)
1	Transmission Control Device

(1)   All the elements used in the off-line system will be available for configuration into the on-line system on demand.

(2)   This is a recommended quantity. However the availability of 8 tape drives is questionable at this time.

TABLE 1-2 SUMMARY OF SUPPORT CAPABILITIES

	Group	Function[1]	Currently Available	Changes Required	New Program
1	Utilities	OS/9020	X		
		JOVIAL compiler	X		
		BAL assembler	X		
		Library Edit	X		
		JOVIAL structured programming listing formatter	X		
		JOVIAL source include processor	X		
		JOVIAL library capability	X		
		CRJE	X		
		HASP	X		
2	DR&A				X
3	Static Data Base	SDBA			X
		SADA			X
4	System Build	Sys Gen			X
		Sys Update			X
		Dump/Restore		X	
		ASB	X		
5	System Test	Inter			X
		Intra			X
		Automated unit and string test drivers			X
6	Management Aids	PERT	X		
		Program Analyzer			X
		HIPO	X		
		Automated Code Auditor			X

Note: 1. Refer to the appropriate section in this document for a description of the indicated functions.

In addition to these general requirements, the
implementation of the CFC support system must
include special recovery capabilities in order
to make provisions for the eventuality that the
support system is aborted due to the preemption
of a hardware component by the operational system
(see Section 1.2). This shall be accomplished by
employing a checkpoint/restart facility in each
support process with a time interval of 30 (SP)
minutes. The recovery recording time, during a
checkpoint operation, shall not exceed 3 (SP)
minutes.

1.5   Document Organization

This document is self contained in the sense that
it describes the complete processes performed on
the off-line system. The six major sections of
this document discuss each of the support sub-
systems, as follows:

        Section 2:   Utilities
        Section 3:   DR&A
        Section 4:   Static Data Base Build
        Section 5:   System Build
        Section 6:   System Test
        Section 7:   Management Aids

2. UTILITIES GROUP

The Utilities Group includes a set of processors which are required to facilitate the preparation, management and maintenance of the operational system components and the support components.

The support system operates under OS/9020, a version of the IBM/360 Operating System tailored for the 9020 computers. OS/9020 provides a set of service programs including a linkage editor, a loader, a sort/merge program and various utilities for handling data and programs (e.g., transfer, copy, print, etc.).

A JOVIAL compiler and a BAL assembler are available under OS/9020. The JOVIAL library capability provides access to commonly used routines (e.g., mathematical, data conversion, etc.).

2.1 Monitor-OS/9020

2.1.1 Introduction

The OS control programs have three major functions: job management, task management and data management. Job management is the processing of communications from the programmer and operator to the control program. Task management includes the following functions:

a. Overlap of central processing unit operation with input/output channel activity

b. Servicing of all hardware interrupts

c. Handling of all supervisor calls (SVCs)

d. Allocation of main storage for programs and data

e. Dynamic loading of programs not in main storage

f. Synchronous overlay supervision

g. Recording of machine malfunctions

h.   Servicing requests for writing CHECKPOINT
records during the execution of a program and
restarting programs at these checkpoints

Data management includes allocating space on direct
access volumes, storing, naming, and cataloging data
sets, and scheduling all I/O operations.

2.1.2   MVT Control Program

The Multiprogramming with a Variable number of Tasks
(MVT) control program configuration is provided
for use in a minimum of four logical SEs. MVT reads
one or more continuous streams of jobs and schedules
the jobs in order of priority. With this configuration
up to 14 jobs can be performed concurrently with
job-step-initiated tasks and operator-initiated tasks
such as readers and output writers. Under MVT the
control program assigns a region from a pool of
available storage to each step as it is initiated.
Upon completion of a job step, the region is returned
to the pool for assignment to other job steps.

2.1.3   Service Programs

Service programs are provided under the MVT Control
Program. The service programs consist of a linkage
editor, a loader, a sort/merge program and a set of
utility programs.

2.1.3.1   Linkage Editor

A linkage editor is provided for combining program
segments that were individually compiled or assembled.
The linkage editor forms a single program that is
ready to be loaded into main storage and executed.
It enables changes to be made in a program without
recompiling the complete program; only those sections
that are changed need to be recompiled.

2.1.3.2   Loader

The loader loads object modules produced by language
translators and load modules produced by linkage
editor into main storage for execution.

2.1.3.3   Sort/Merge Program

The sort/merge program is a generalized program that can be used to sort or merge records in ascending or descending order. The sorting and merging can be performed using magnetic tape and direct access storage devices for input, output, and intermediate storage. Execution of the sort/merge is initiated by control statements in the operating system input stream, or by another program through the use of macro instructions.

2.1.3.4   Utility Programs

These programs are used to:

    a.   Transfer, copy, or merge sets of data from one storage medium or I/O device onto another

    b.   Edit, rearrange, and update programs and data

    c.   Compare, print, or punch data

    d.   Create an input stream

2.1.2   9020 Modifications

2.1.2.1   Problem Areas

Four problems require solutions before either OS control program can be run in 9020 mode using native 9020 equipment such as the PAM. The problems are:

    a.   Physical channel 0 is the only valid specification for a multiplexor channel.

    b.   Storage with a storage key of 0 is unprotected on a 9020.

    c.   1052 consoles, attached to a PAM, require the high order bit of the command code in the CCW to be on.

d.   The high order eight PSW bits on a 9020
and a 360 are identical. On the 9020 bits
16-19 of the PSW are defined as system mask
bits for physical channels 7-A. Consequently,
the SSM (Set System Mask) instruction on the
9020 causes 12, as opposed to eight on the
360s, bits to be set in the PSW. The interrup-
tion code in the PSW is 12 bits on the 9020
opposed to 16 bits on the 360.

2.1.2.2   Modifications

Modifications required to resolve each of the problems
described in Section 2.1.2.1 are explained in the
following paragraphs:

a.   Added Multiplexor Channels

On the 9020, channels 0, 4, and 8 are multi-
plexor channels. To correctly utilize a non-
zero multiplexor channel in OS, patches exist
in the OS input/output supervisor (IOS). The
patches were applied to the Test Channel and
Channel Search Routines in IOS.

b.   Storage Protection

On a 360, OS will assign a zero storage key
for the control program. To provide control
program protection for OS on the 9020, a
storage key of one was assigned to control
program storage. The storage validity check
routines in IOS are modified to assume keys
of zero or one are control program storage.
In OS/MVT the logic in GETMAIN/FREEMAIN
(dynamic storage allocation module) is
appropriately modified. Control program
storage requests are assigned a key of one.
Problem program storage requests are pro-
cessed as if it were a 360. Released storage
is returned to the available storage pool with
a key of one. The technique used in MVT
imposes a procedural restriction such that
the maximum number of job initiators is
reduced from 15 on a 360 machine to 14 on
a 9020.

c.   PAM 1052 Capabilities

The system console I/O modules are simply modified
to assure the presence of the high order bit in
the I/O command code. In addition, the IPL logic
is modified to assure the proper command code.

d.   9020 System Mask Sensitivity

Both the I/O and SVC interrupt handlers in OS
assume bits 16-19 of the old PSW are zero. The
execution of the SSM instruction in OS is numerous
and, consequently, the low order mask bits in
the PSW are unpredictable. The initial logic
in both the I/O and SVC first level interrupt
handlers is modified to clear bits 16-19 of the
old PSW, stored at interrupt time.

2.2   JOVIAL Compiler

The JOVIAL compiler translates JOVIAL source into
Basic Assembly Language (BAL) code for subsequent
processing by the Assembler.

2.2.1   Function

2.2.1.1   BAL Source Generation

During compilation, the compiler checks all JOVIAL
source program statements and issues diagnostic
messages when errors are detected. When library
routines are requested in the source program, the
compiler accesses a disk resident library Procedure
Descriptor Table (PDT) to generate the required
parameter list and library calling sequence.

If a compool is requested, the compiler fetches the
specified compool tables data set from disk.

The compiler is finished when it has generated a
sequential data set of BAL source statements from
the input JOVIAL source. The BAL assembler is
invoked as a separate job step following successful
completion of the JOVIAL compiler.

2.2.1.2 JOVIAL Source Syntax Check

The JOVIAL source syntax check is invoked via the
JOVIAL option, SYNCK. When operating in syntax
check mode, the compiler will terminate after the
first phase, producing a listing with the diagnostic
messages immediately following the statement in error.

2.2.1.3 Compool Generation

When this option is selected, source input to the
compiler consists entirely of data declaration state-
ments. The compiler will generate a new, disk
resident compool consisting of three data sets
(compool tables, compool reserves and compool object
decks).

2.2.2 Input

Programs written in the JOVIAL language are the
primary input to the compiler. In compool generation
mode, input is restricted to JOVIAL data declarations
for arrays, tables and their strings and/or item
definitions (with or without preset constants), and
parameter items. The compiler will be invoked by
standard catalogued procedures of job control language.
One procedure will be provided to invoke the BAL
source generation or syntax checking functions of
the compiler. A second procedure will be used to
generate a compool.

Major options of the compiler are:

     a.   List data references by name

     b.   Punch .XRF cards for LIBRARY calls and/or
compool data references

     c.   Perform source syntax check

     d.   Generate a compool

2.2.3   Output

Compiler output varies according to the function being performed. The following sections itemize the outputs of each function.

2.2.3.1   BAL Source Generation

a.   JOVIAL source listing

b.   BAL source statements and requests for compool segments

c.   Errors and diagnostics

d.   Cross reference listing

e.   Punched deck of XREF cards

2.2.3.2   JOVIAL Source Syntax Check

a.   JOVIAL source listing with diagnostic messages immediately following the statement in error

2.2.3.3   Compool Generation

a.   Listing of compool

b.   Diagnostic messages

c.   Three disk resident data sets

1.   Compool Tables

2.   Compool Reserves

3.   Compool Object Decks

2.3   BAL Assembler

The BAL Assembler translates BAL statements into loadable object decks.

2.3.1 Function

The 9020 Assembler translates BAL input into object decks. An object deck is a loadable module. It includes binary equivalents of the BAL source and linkage information for programs separately assembled that are to be loaded and executed together.

The Assembler is invoked as a separate jobstep by the OS control programs as a result of an // EXEC PGM=BAL card. A catalogued procedure containing the required job control language (JCL) will be provided for calling BAL.

The BAL Assembler accesses the compool reserves data set if requested via a CMPRSV DD statement. FINDS are issued for all compool segments requested via BAL PSEG cards and the appropriate members of the reserves are included in the assembly.

When PUNCHC is requested, the Assembler provides INCLUDE CMPSEG (ZX seqnam) cards to the OS linkage editor. The linkage editor will then include the required compool object decks when constructing an executable load module.

2.3.2 Input

    a. BAL options provided in the PARM of the // EXEC PGM=BAL card which invokes the Assembler

    b. Assembly language source statements in the sequential data set defined via the SYSIN DD statement

    c. Partitioned data set of compool reserves containing one member for each compool segment

2.3.3 Output

    a. Listing of input source, assembled code, and diagnostics

    b. Object decks containing:

        1. Program Text

        2. External Symbol Dictionary (ESD) Information

3.   The Relocation Dictionary (i.e., address constants that must be changed if the program is loaded at a location different than the assembled address)

4.   .LIB cards required to construct a JOVIAL library

c.   Cross-reference listing

d.   Punched cross reference information for compool data and library programs

e.   Object decks and optionally, INCLUDE cards for the OS linkage editor if the LOAD option is requested

2.4   Library Edit

The library edit creates a disk resident library of common subroutine and procedures. Routines may be added, deleted, or replaced in the library.

2.4.1   Function

The library edit is used to generate and maintain a library of routines which are available to JOVIAL and BAL programmers. The library edit program is used to create a new library or to add routines to (or delete routines from) the existing library. To replace a library routine, the old routine must first be deleted and the new routine added. Catalogued procedures of job control language will be provided to allocate and catalog PDT and FSD data sets, delete and uncatalog obsolete PDT and FSD data sets, generate a new library and update an existing library.

2.4.2   Input

Input to the library edit program consists of control cards, object decks for routines being added or replaced and three pre-allocated library data sets. The library edit program will be invoked by a // EXEC LIBEDTN card when a new library is being generated and by a // EXEC LIBEDTR card when an existing library is being updated.

2.4.3   Output

a.   Listing and diagnostic messages. An option
is provided to request ESD and PDT information
for all routines in the library. If omitted,
ESD and PDT information will be printed only for
library routines being added.

b.   A new or updated disk resident library
consisting of the following three data sets:

1.   A sequential PDT data set containing
the information required by JOVIAL to
generate calling sequences to the library
routines.

2.   A sequential ESD data set containing
external symbol dictionary information.

3.   A partitioned data set of library
routine object decks.

2.5   JOVIAL Structured Programming Listing Formatter

2.5.1   Function

The JOVIAL listing formatter provides automatic
indentation of JOVIAL source code which conforms
to the structured programming listings.

2.5.2   Input

Input to the listing formatter consists of JOVIAL
source statements. JOVIAL code may be either structured
or unstructured.

2.5.3   Output

Output from the listing formatter is a printed listing
which conforms to the indentation standards for JOVIAL
structured programming. The indentation conventions
are designed to produce a JOVIAL listing which is easy
to read, maintain and modify.

2.6   JOVIAL Source Include Processor

2.6.1   Function

The JOVIAL Source Include Processor provides the capability to include JOVIAL source statements from other sequential or partitioned data sets into a data set for input to the JOVIAL Compiler.

2.6.2   Input

Input to the JOVIAL Source Include Processor is:

   a.   A data set containing the JOVIAL source program with JOVIAL Include control statements

   b.   Sequential and/or partitioned source data sets to be included into the JOVIAL program. These secondary inputs may contain JOVIAL source and JOVIAL Include control statements.

2.6.3   Output

The JOVIAL Source Include Processor provides two outputs:

   a.   A JOVIAL source data set that will be input to the JOVIAL Compiler

   b.   A sequential message data set which contains a listing of all primary inputs, optionally, all secondary inputs, and all error messages. The listing is formatted according to the structured programming indentation conventions.

2.7   JOVIAL Library Capability

A JOVIAL library will be provided for use by the off-line support systems. The OS/9020 JOVIAL and linkage editor access a disk resident library generated by the OS version of the library edit program. The library routines can be classified according to purpose. They include mathematical, data conversion and movement, data comparison, input/output, unit positioning, and initialization routines. Section 2.7.1 lists these library routines and provides a brief functional description.

The JOVIAL compiler uses tabulated information
generated by the library edit program to determine
the routines needed and their parameter requirements.
It generates an external symbol for each requested
library routine and produces the required parameter
list and calling sequence. Under OS, the automatic
call feature of the linkage editor causes the required
library object decks to be included in the load
module from the partitioned data set of library
object modules furnished via the SYSLIB DD statement.

2.7.1   OS/9020 Library Routines

2.7.1.1   Input Data Conversion

Routine	Function
ASCIN	Translate EBCDIC to ASCII
FLTFL	Convert from EBCDIC floating point to internal floating point
FLTINJ	Convert from EBCDIC floating point to internal floating point
HEXIN	Convert from EBCDIC hexadecimal to internal binary
HOLFL	Access EBCDIC field without alteration
HOLINJ	Direct access of EBCDIC field without alteration
INTFL	Indirect conversion from EBCDIC integer to internal binary integer
INTINJ	Direct conversion from EBCDIC integer to internal binary integer

2.7.1.2 Output Data Conversion

Routine	Function
ASCOUT	Translate ASCII to EBCDIC
FLTOTJ	Convert from internal floating point to EBCDIC floating point
HEXOUT	Convert from internal binary to EBCDIC hexadecimal
HOLOTJ	Move EBCDIC field without alteration
INTOTZ	Convert from binary integer to EBCDIC integer with leading zeros
INTOTJ	Convert from internal binary integer to EBCDIC integer
MIXOTJ	Convert from internal fixed point (scaled to integer) to EBCDIC fixed point

2.7.1.3 Data Conversion and Movement

Routine	Function
MVC	Move data between identically sized character fields
TMTR	Execute TRT plus move and TR
TR	Translate character data from one code to another according to a user-supplied translate table
TRT	Test character data by executing a TRT instruction and returns the address of the first non-zero byte
TSET	Execute a Test and Set instruction and returns a 0 or 1 condition code
TSTR	Execute TRT and TR in sequence

Routine	Function
LEFTJ	Left-justify Hollerith data that are right-justified with leading zeros
MVI	Move an immediate character field into an address
MVM	Move a character field of a specified length from one address to another address
MVT	Return a character field of a specified length from an address as the function output value

2.7.1.4 Data Comparison

Routine	Function
CLC	Compare two equal length strings of characters. The arguments specify the locations of the two fields and their length
CLI	Compare two equal length strings of character. The second field is an argument rather than the address of an argument

2.7.1.5 Mathematical

Routine	Function
ARCOS	Compute angle in radians from its cosine
ARCSIN	Compute angle in radians from its sine
ARCTAN	Compute angle in radians from its tangent
ARCTND	Compute heading in degrees from its X and Y components
ARCTNP	Compute heading in Azimuth Change Pulses (ACPs) from its X and Y components

Routine	Function
ATANFP	Similar to ARCTNP but accept floating point input
COS	Compute cosine for angle expressed in radians
CSINE	Compute sine and cosine for angle expressed in ACPs
EXPON	Compute a number from its natual log
LOGN	Compute the natural log of a number
PANDOM	Generate a pseudo random floating point fraction
RANINT	Generate a pseudo random integer within a provided range
SIN	Compute the sine for an angle expressed in radians
SORT	Compute the square root of a positive number

2.7.1.6 Compiler Called Routines

Routine	Function
EXPFF	Convert floating point base to floating point exponent
EXPFI	Convert floating point base to integral exponent
FIX	Convert floating to fixed
FLTA	Convert fixed A-type to floating
FLTI	Convert fixed integer to floating

2.7.1.7 Data Set Initialization

Routine	Function
OPEN	Logically connect a data set (QSAM)
SHUT	Logically disconnect a data set (QSAM)
SETRET	Set I/O return
DCBQ	Define QSAM Data Control Block

2.7.1.8 I/O Positioning

Routine	Function
CNTRL	Control on-line magnetic tape file positioning

2.7.1.9 I/O Input

Routine	Function
GET	Obtain next logical record (QSAM)
VGETR	Unblock variable blocked input data and return address logical record (QSAM)

2.7.1.10 Data Output

Routine	Function
WTO	Write to operator
WTOR	Write to operator with reply
PUT	Write next logical record (QSAM)

2.7.1.11 Main Storage Supervision

Routine	Function
GETCOR	Allocate main storage
FRECOR	Release allocated main storage

2-16

2.7.1.12 Miscellaneous

Routine	Function
CORE	Core dump in hexadecimal and EBCDIC
FLTFLD	Scan a character sequence to locate individual data fields and store field descriptions in a table for the caller
LOCW	Provide the address of any table or array as a word value
MODULO	Produce an output that is equivalent to input module 14400
TIMDIF	Accept an input time and produce its equivalent with absolute value less than 7200

2.8 Conversational Remote Job Entry (CRJE)

CRJE* operates under OS/9020. It provides Remote Job Entry capability for users at remote keyboard terminals that are connected to the 9020 via communication lines. Users can prepare and enter jobs for background processing under the operating system at the central installation. Remotely submitted jobs are scheduled, initiated, executed, and terminated under the control of the OS job management routines. Thus, a remote CRJE user has the same batch-computing facility that is available for a local user. Through background processing of jobs submitted through CRJE, users have access to the full facilities of OS/9020.

Job input consists of programs and data that are conversationally created and maintained through the facilities of CRJE. Lines of program source statements, data, and job control language are collected

---

* Note: this description of CRJE was largely extracted from IBM form GC30-2012-1.

within the central system as the user types them at
a remote keyboard terminal. There is no need for
keypunching, nor is there the waiting time for
operator handling and card reading. Simple correc-
tion procedures make it easy to get data entered
without typing errors.

Operating on-line with the system greatly reduces
job turnaround time, since data is transmitted
directly between the central processor and the
terminal, with no intermediate devices such as
card readers and printers. To submit a job for
execution, the user just selects the program,
data, and job control statements to be entered
into the OS input stream. As soon as the job
is completed, the user can examine the output
at any terminal.

Data entered from a terminal can also be saved
at the central installation for subsequent use.
Stored data can be easily retrieved for on-line
displaying and modifying as well as be specified
as part of job input to the operating system. A
user can update his stored data by inserting,
replacing, deleting, and changing single lines
or groups of lines. CRJE provides protection
against unauthorized access of stored data as
well as a means for users to share data.

In addition to facilities for job preparation,
job entry, retrieval of job output, and manipula-
tion of programs and data, CRJE provides the
terminal user with information about his data
sets and the status of jobs he has submitted.
There is also a message facility for two-way com-
munication between terminal users and the operator
of the central computer.

2.9   Houston Automatic Spooling Program (HASP)

HASP operates in conjunction with OS/9020. It
performs peripheral functions, not normally
accomplished by OS/9020, by stealing small amounts
of CPU time to operate the printers and the reader/
punch, while operating in direct communications
with OS. HASP also controls the job flow, ordering
of tasks and the spooling of inputs and outputs.

3. DR&A GROUP

The DR&A programs provide the capability to perform off-line processing of data recorded on tape and/or disk during live operation of the CFC system. A variety of DR&A programs shall be available to reduce, sort and analyze the on-line recorded data and to provide clear and concise reports, statistical data listings and plots on past system performance. The DR&A user specifies, through input commands, which components of the DR&A group shall be executed and in what order. The DR&A capability must be easy to use, flexible, easily expandable, efficient and as automatic as possible from the standpoint of operations.

3.1 General

A conceptual flow of the DR&A subsystem is shown in Figure 3-1. The input, processing and output functions are described as follows.

3.1.1 Input

Input to the DR&A subsystem will consist of magnetic tapes and/or disk and user control cards. The tape/disk normally contain raw data recorded during the actual running of the operational system.

In some cases the tape/disk may contain data which was previously processed by other DR&A programs. The user control commands and parameters select the desired program and specify various user options and parameters. Where possible, a missing card input item will be filled in automatically assuming predefined defaults.

3.1.2 Processing

All DR&A processing will be performed under control of the DR&A Executive. This Executive will schedule and initiate all the DR&A subprograms which are needed to successfully carry out a complete DR&A run. The DR&A subprograms will include the following modules:

    a.   Read and process input cards

    b.   read tape/disk recorded data

    c.   Sort/merge/compare the raw data

FIGURE 3-1   DR&A

USER REQUEST CARDS

READ AND PROCESS INPUT CARDS

ON-LINE RECORDED DATA

READ TAPE/DISK RECORDED DATA

DR&A EXEC

SORT/ MERGE/ COMPARE RAW DATA

ANALYZE THE DATA AS NEEDED

PRODUCE OUTPUT REPORTS

OUTPUT REPORTS

3-2
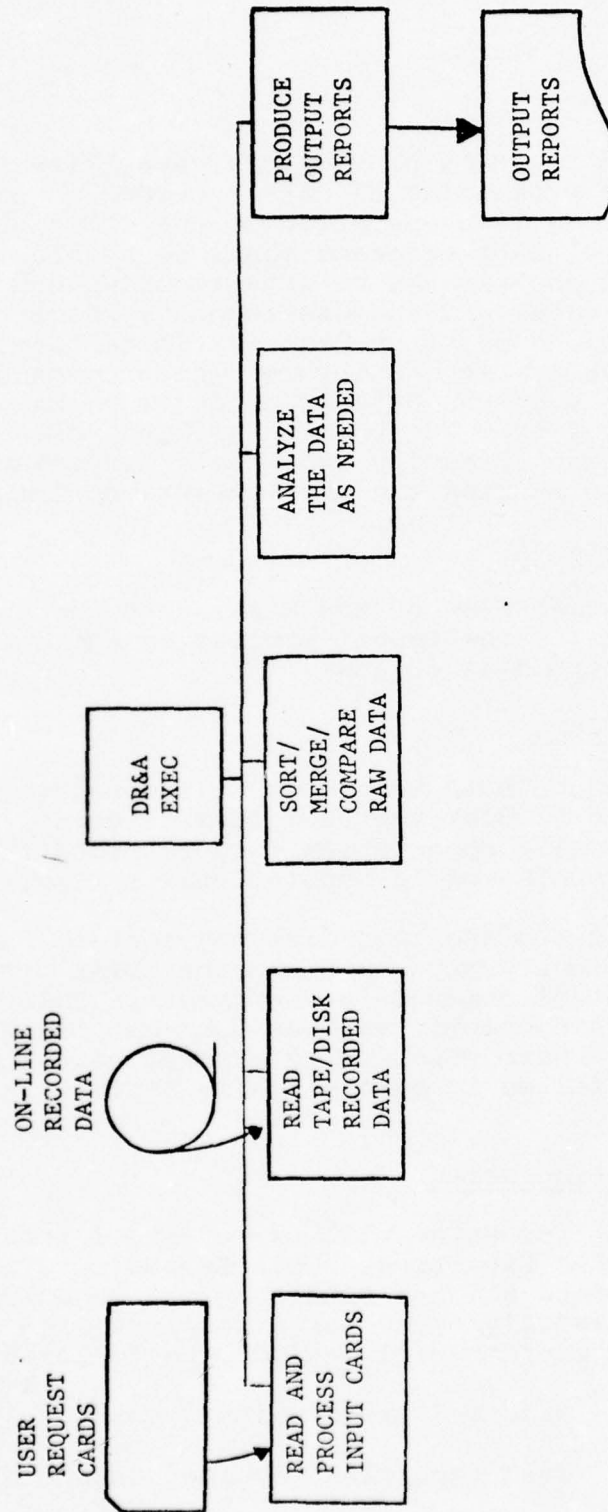
d. Analyze the data according to the required function

e. Generate the output report and print on the output device(s)

3.1.3 Output

The output consists primarily of a printout of the requested DR&A reports. In addition, general information will be provided containing identification of the run requested, the date, the input tape identification, the system version used, and page header and title. The input control cards will also be listed together with all applicable default parameters. Diagnostic messages, and user alerts if any, will be clear and informative.

3.2 DR&A Functions

This section describes some of the output reports that will be produced. These are:

a. Input/Output log

b. Data base analysis

c. Input/Output message summary

d. Response time summary

e. Log comparison function

f. Performance and activity measures

g. General Recording Data Processor

3.2.1 Input/Output Log (IOLOG)

The IOLOG function will provide a listing of all input and output messages which have been recorded on tape during CFC operations. IOLOG will determine, when possible, the message type, source and destination identity, and time. Certain messages will have their contents interpreted into more meaningful information. This information will be displayed in a legible output format.

IOLOG will be able to filter and sort the data according to user selected parameters. The manner in which data is filtered and/or sorted will be any combination of the following:

a.   Start/Stop Time

b.   Message type

c.   Message source

d.   Message destination

e.   Airport/center ID

The user can specify what each report should or should not contain.

3.2.2   Data Base Analysis (DBA)

The DBA function will provide the capability of recreating all dynamic tables at any desired point in time. These tables will be reconstructed from the data recorded on-line and from messages that would effect changes to these tables. After the tables have been reproduced, the user has the option to obtain complete or selective listings of the data.

3.2.3   Input/Output Message Summary (IOSUM)

The IOSUM function will summarize all or selected input and output messages to and from the operational CFC system. Reports are produced in a legible format displaying summaries and statistical information on the volume and distribution of input and output messages.

The selection of the input/output messages is according to the following criteria or any combination thereof, as specified by the user:

a.   Start and/or stop time

b.   Type of message

c.   Message source/destination

d.   Airport/center ID

e.   Type of report desired

The following data will be accumulated to provide, according to the type of report desired, the following statistics: total message counts, average message rate (i.e., messages per parameter time), min, max, standard deviation and histograms of messages in equal time intervals. These statistics can, at the user option, be for all messages or they can be broken down by any combination of the following:

    a.   Message type

    b.   Message source/destination

    c.   Airport/center

3.2.4   Response Time Summary (RTSUM)

The RTSUM calculates and produces summary reports on the amount of time required by CFC to produce an output message in response to an input message.

The program uses the output from the Input/Output Log function as a basis for the analysis. Only successful attempts to correlate input/output messages will be used to compute the following response time statistics:

    a.   Sum

    b.   Mean

    c.   Maximum

    d.   Minimum

    e.   Standard deviation

    f.   Frequency distribution

The above statistics will be produced pertaining to all messages which have been selected by the user. In addition, if the user desires, the statistics can be broken down by any combination of the following:

a.   Type

b.   Source

c.   Time intervals

A matched input/output message report and unmatched
message report may also be produced at the user option.

3.2.5   Log Comparison Function (LOGCOMP)

The LOGCOMP function provides a means of comparing two
sorted edited on-line recorded tapes and providing
a legible formatted output listing noting any dis-
crepancies in the records of the two compared tapes.

Various options are available to the user in order
to suppress reporting of irrelevant errors. Tape
records to be used in the comparison can be selected
or filtered out as specified by:

a.   Time intervals

b.   Records types

c.   Message types

d.   Message source/destination

In addition, "time error" tolerances of the matched
records can be specified.

A printout of all the current records will be optionally
provided. In addition, a summary of total error found
will be printed out containing totals of matches and
no matches.

3.2.6   Performance and Activity Measures (PMAM)

The PMAM function provides the means for analyzing the
operational system hardware and software performance
from the viewpoint of timing, utilization and workload
of the various system components. This capability
can be useful for detecting and eliminating potential
bottlenecks and to support system tuning to improve
performance. The following is a generalized list of
the measures that will be provided:

a.   Utilization of each CE, SE, IOCE, DCU, TCU, I/O devices, etc.

b.   Utilization and frequency of activation of major CFC subprograms

c.   Summary of issued I/O, external and SVC interrupts

d.   Queuing statistics on major system queues

e.   Sizing reports on dynamic buffers in main memory

f.   Sizing reports on storage areas on disk and tape

3.2.7   General Recording Data Processor (GRDP)

The GRDP function provides a general purpose data reduction capability which the user can obtain according to his unique requirements.

The control information provided by the user specifies the data records to be processed from the on-line recorded data. This information will include the identification of the data records to be processed and the time of the recordings of interest.

The records will be identified by one of the following ways:

1.   By a symbolic name or unique identifier

2.   By an identification common to a set of records

3.   By an input code which will indicate all records

In the first case, only those records with the specific name or identifier will be processed; in the second, only records which belong to the set having that identification will be processed; and in the third, all records will be processed. This assumes that all records meet the specified time of interest criteria.

The time of the recordings of interest will be specified by:

    1.   A time interval of recording, or

    2.   Every recording, i.e., no restrictive specification with respect to the time interval.

The output will include a listing of the selected records from the on-line recorded data. In order to reduce the amount of printing, whenever a line of output is identical to the previous line, it will not be printed. Instead, a symbol will be printed which will indicate this fact, and the number of lines which are identical will be indicated. For example, if the block consists of 100 words of which words 1 and 100 are the value 1 and the remaining words have the value 0, then the formatted output might be:

Word	Value
1	1
2	0
.	.
100	1

In addition diagnostic messages will be provided for unprocessed data blocks, if any. These messages will include a dump of the unprocessed records and the reason for the failure to process.

The program will have the capability of processing all types of records encountered in the on-line recorded data.

4. STATIC DATA BASE GROUP

The purpose of the Static Data Base Group is to
assemble the static data base and the adaptation
data that will be used on-line by the CFC system.
The static data base is constructed from the OAG
tapes (Reuben Donnelley tapes) and from other related
data (e.g., airlines, aircraft types, etc.). The
adaptation assembler processes geographical and
system configuration data into a format which can
later be used in building the operational CFC system.
This group of programs reads the source static data
base and the adaptation data, performs error checking,
extracts and rearranges the source inputs, and pro-
duces the object text suitable for future computer
processing. User oriented statistics and summaries
are printed in accordance with user requests. In
addition, a capability is provided to easily update
the static data base and the adaptation data as
needed.

Figure 4-1 is a functional representation of the
two major static data base build processors.

4.1 Static Data Base Assembler

The Static Data Base Assembler (SDBA) is an off-line
program which will generate and update the various
CFC static data base files accessed during on-line
operations.

The kinds of CFC data which are handled by SDBA
include, but are not limited to, the following:

    a.   Air-carrier flight schedules

    b.   Airport data

    c.   Airline codes

    d.   Aircraft types

    e.   ARTCC data

    f.   Zone data

    g.   General aviation data
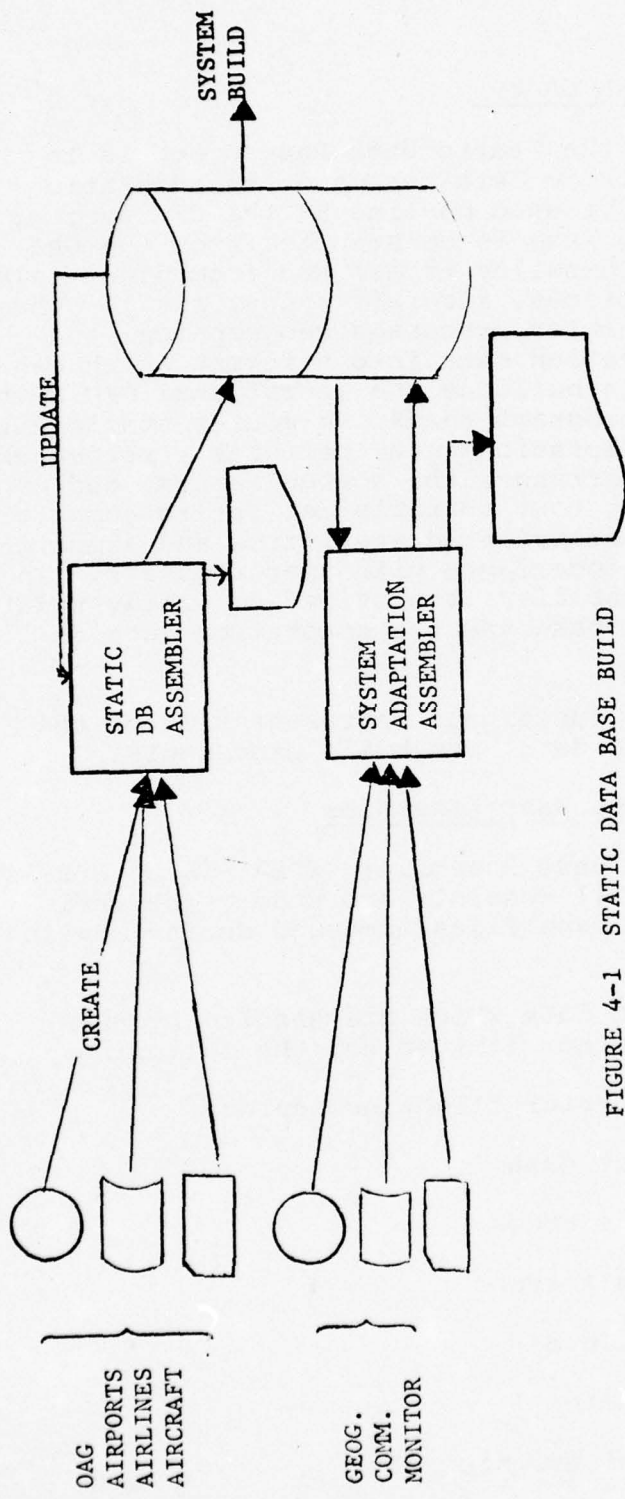
FIGURE 4-1 STATIC DATA BASE BUILD

The primary input to SDBA are the OAG data tapes
(produced by the Reuben Donnelley Corp.). In
addition, necessary data is provided through input
tapes and/or cards. Processing is determined through
user control cards specifying the required SDBA
function and identifying the run parameters. SDBA
checks the legality of the input data and insures
the validity and completeness of the new files. The
user is notified if errors are found.

4.1.1   Program Modes of Operation

There are three modes of operation (Assemble, Update
and List) of the SDBA program. The Assemble mode is
used initially to create a new data base. The Update
mode provides the capability to add, change and
delete information stored on the data files. The
List mode produces a listing of the data base accord-
ing to user options.

4.1.2   Input

The SDBA program accepts one or more of the following
types of input data depending upon the mode of opera-
tion specified on the program control card:

      a.   program control cards

      b.   ADD cards

      c.   CHG cards

      d.   DEL cards

      e.   tapes (e.g., OAG tapes) and/or cards con-
           taining the data to be assembled

Where card input is required, the user will have the
option to input card images from magnetic tape.

4.1.2.1   Program Control Cards

The Program Control Cards contain the following data
fields:

Field 1:   indicates the mode of operation - AS
          (assemble), UP (update), or LI (list)

Field 2:   identifies the user, the date and contains
          a concise description of the run

Field 3:   specifies the requested outputs (e.g., type
          of listings, output device, etc.).

4.1.2.2   ADD Cards

The ADD card is used in the Assemble or Update mode
to enter data (e.g., flight plans) into the static
data base. The following data fields will be pro-
vided by the user:

Field 1:   the characters "ADD" along with any optional
          user identification

Field 2:   the file to which the addition is made

Field 3:   the data to be added (may extend over
          several cards)

4.1.2.3   CHG (Change) Cards

The CHG card is used in the Assemble or the Update
mode. The purpose of this card is to modify the
data contained in the data base. The following
fields are included:

Field 1:   the characters "CHG" along with any optional
          user identification

Field 2:   the file in which the modification will be
          made

Field 3:   the data which is to be changed

Field 4:   the new data

4.1.2.4   DEL (Delete) Cards

The DEL card is used in the Assemble or the Update
mode. The purpose of this card is to delete data
from the data base. The following fields are included:

Field 1: the characters "DEL" along with any optional
user identification

Field 2: the file in which the deletion will be made

Field 3: the data which is to be deleted

4.1.2.5 Tape/Card Input

In the Assemble mode the input data from which the
data base will be generated consist of tapes (e.g.,
OAG tapes) or disk files and, optionally, cards (e.g.,
zone data). Any data read from cards can also be
prestored on tape and the tape may be read in during
the Assemble mode instead of cards.

4.1.3 File Organization

(to be specified)

4.1.4 Output

The SDBA program produces the following types of
output:

      a. error messages (if any)

      b. listings

      c. new or updated data base

4.1.4.1 Error Messages

Validity checking will be performed on all input
data. If an error is found, a self explanatory
error message will be printed out.

During the processing of the Assemble, Update and
the List mode the program will report on all error
conditions such as: unreadable data, missing data,
ambiguities between the data base contents and user
requested updates, etc.

4.1.4.2   Listings

The output listings will contain, according to the
user option, a printout of the user input cards and
a printout of data extracted from the data base.

In the Assemble and the Update mode, the user can
request a listings of any newly created file(s).
The List mode is for the purpose of obtaining such
a listing from an existing data base on tape or disk.
The produced printout will be in a readable form.   It
will also include the identification of the data base
used.

4.1.4.3   New Data Base

The major output from the Assemble and the Update
mode is the newly created data base.   The new data
base will be produced on tape or disk and will be
in a form ready to be integrated within the CFC
system during the system build process.   A header
record on the tape or disk will identify the new
data base version.

4.2   System Adaptation Assembler

The System Adaptation Assembler (SADA) is an off-line
program to adapt CFC to the system configuration
requirements.   The user input to SADA specifies, in
a simple and clearly readable form, the environment
under which CFC will operate.   SADA processes this
data, checks it, and produces the system adaptation
data base.   This data base is in object module form,
and is readily available to be used by the system
build processors to generate the operational on-line
CFC system.   SADA also provides the capability to
modify and update an existing adaptation data base.
As an option, SADA provides various output listings
with user oriented information.

4.2.1   Adaptation Data

The adaptation data is divided into the following
subsets:

a. Geographical adaptation

b. Communications adaptation

c. Monitor adaptation

4.2.1.1 Geographical Adaptation (GA)

The GA tables contain geographical data on the following:

a. airports

b. ARTCCs

c. fixes

d. airways

e. zone structure

4.2.1.2 Communications Adaptation (CA)

The CA tables contain data related to the communication network linking the CFC computer, the ARTCC computers, and the SCC facility. The following data is needed:

a. line connections

b. line characteristics (speed, capacity, etc.)

c. buffer storage

d. number of retransmissions

e. I/O devices

4.2.1.3   Monitor Adaptation (MA)

The MA tables contain data used by the monitor per-
taining to the operational on-line CFC system. These
data include but are not limited to:

    a.   on-line recorded data

    b.   recovery recordings

    c.   program priorities

    d.   timing data

    e.   logical device numbers .

    f.   output message routing

    g.   security locks and authorized passwords

    h.   startup/startover tasks

4.2.2   Program Modes of Operation

SADA operates under three user-defined modes: Create,
Update and List. The Create mode is used to generate
a new output master text file. The Update mode
specifies that certain selective changes are to be
incorporated in the master text files. The master
text file is used by the system build processor to
generate the operational CFC system. The List mode
specifies that a listing is to be prepared of some
or all of the adaptation data.

4.2.3   General Function

In the Create and the Update modes, SADA will accept
symbolic source input defining the adaptation data.
Adaptation input will be divided into logical groups
of data. SADA will process each input group, check-
ing for its validity and correct format, and perform
initial input data processing. After all input data
has been successfully validated and initial data
processing completed, SADA will begin generation of
intermediate work files until all input data has been
processed. Output generation will then be performed
with the generation of object form text files on disk
or tape to be used in system build.

In the List mode the program scans the previously
prepared SADA files for the required information
and produces the necessary listing in logically
structured and readable format.

User control cards must be provided to specify the
required mode and various run parameters.

4.2.4   Input

The input consists of user-specified run parameters,
adaptation data to be assembled and old adaptation
master files to be updated.

4.2.4.1   Run Parameters

The run parameters card or tape input includes the
following information:

     a.   mode (Create, Update or List)

     b.   run identification

     c.   requested listing of input data

     d.   requested listing of output data

4.2.4.2   Master Adaptation File

These data are contained on disk or tape, and are used
in the Update and List modes.

4.2.4.3   Adaptation Data

This data is provided in source form on tape or
cards, and is used either to build a new master
file or to update an existing master file.   The
adaptation data is grouped and sorted in order to
reduce SADA processing time.   The groups of data
are:

     a.   geographical data

     b.   communications data

     c.   monitor data

Each group is preceded by a header card (or record) identifying its contents.

4.2.4.4   Data Updates

The data updates cards or tape entries specify additions (ADD header), deletions (DEL header) or changes (CHG header) to the master file records.

4.2.5   Adaptation Master File Organization

(to be specified)

4.2.6   Output

A SADA run may produce the following outputs:

    a.   New master file

    b.   Listings

    c.   Error messages

4.2.6.1   Master File

The new master file is produced on tape or disk and is readily useable by the system build processor in generating the operational CFC system.

4.2.6.2   Listings

The following complete or partial listings in logically structured format are produced according to user specified options.

    a.   Input data listings

    b.   Listings of the adaptation master file

    c.   Listings of the changed adaptation data
         (in the Update mode)

4.2.6.3　Error Diagnostics

This output will provide a list of error diagnostics which will identify, to the extent possible, the specific data in error, the error number and a concise description of the error.

5.   SYSTEM BUILD GROUP

The System Build Group provides the capability to
generate and maintain an IPLable disk-resident CFC
system.   User options and parameters are furnished to
permit selection of hardware, software, storage
areas, and the overall system environment.   The
preassembled static data base is incorporated during
the system build process in the disk-resident CFC
system.   As an option, a system build run will pro-
duce printouts describing the newly built system
configuration, cross reference listings and statistical
data.   A system update capability shall also be
provided to facilitate an efficient updating of the
disk-resident system and data base.   In addition,
an off-line program shall be provided to create
transportable and backup copies of the disk-resident
system.   The system can thus be "dumped" on magnetic
tape or disk and subsequently "restored".

5.1   CFC System Generation

Support programs will be required to generate an
IPLable disk-resident CFC operational system.
A functional description of the system build pro-
cessor is shown in Figure 5-1.

5.1.1   Input

Input to the system generation program will include:

a.   data sets on tape or disk containing the
complete set of CFC operational programs in
object deck form

b.   9020 system library routines

c.   adaptation file containing geographical
and system configuration data

d.   a set of user specified system parameters
defining the hardware, software, storage areas
and the overall system environment.

e.   static data base file(s) which have been
previously checked and processed by the off-line
static data base assembler program.

5-1

FIGURE 5-1 SYSTEM BUILD

CFC
OBJECT
PROGRAMS

9020
SYSTEM
LIBRARY

ADAPTATION
MASTER
FILE

STATIC
DB
MASTER

USER
SPECIFIED
PARAMETERS

SYSTEM
BUILD
PROCESSOR

IPLable
disk-resident
CFC system

listings
errors
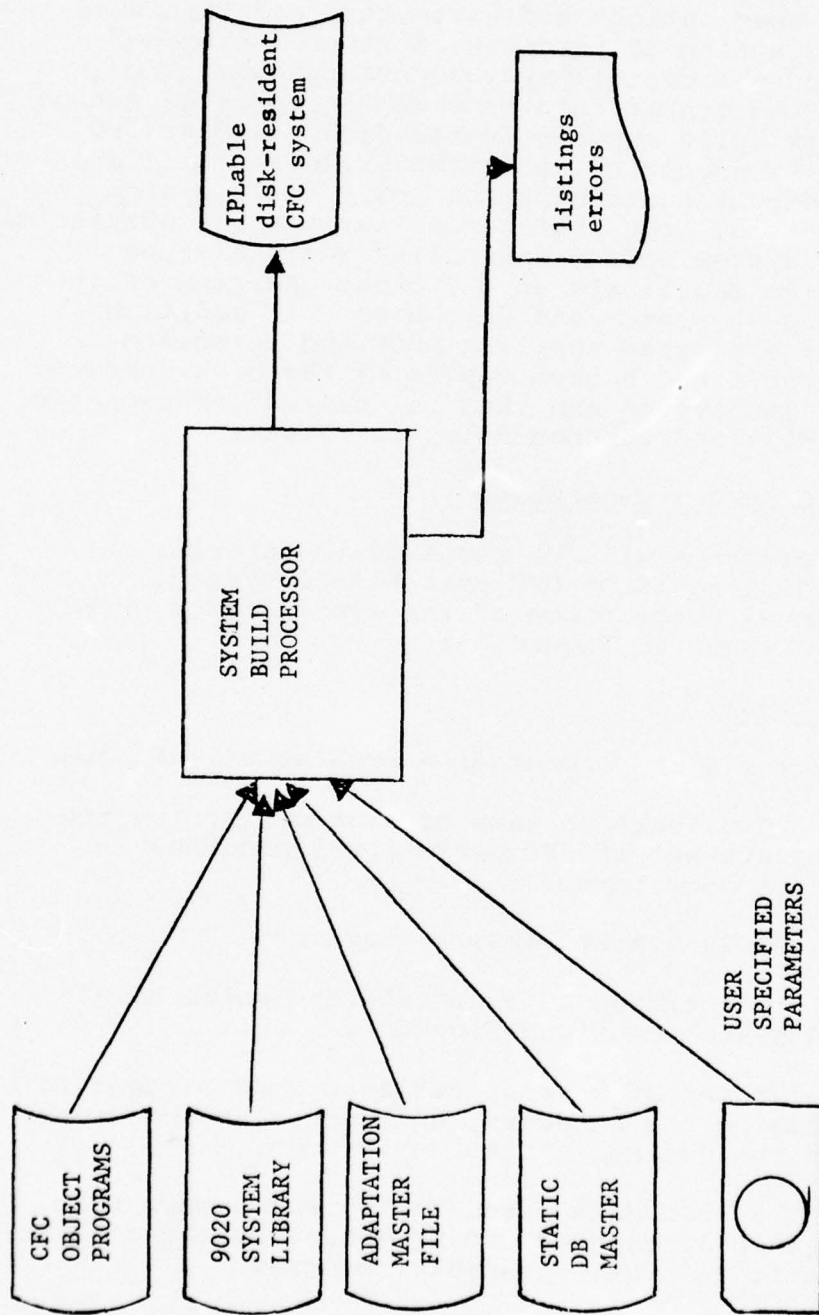
5.1.2 Function

The system generation program will perform the following functions.

    a.   construct and write disk IPL record

    b.   read and transfer onto disk the core-resident and disk-resident data sets in core-image form

    c.   allocate disk space for recovery recording data

    d.   construct the necessary linkages between programs and storage areas and between programs and hardware devices

    e.   set storage protect keys

5.1.3 Program Element (PE) Stringing

The PE stringing capability is required to accomplish the stringing (or linking) of appropriate software modules of the CFC applications programs. For this purpose, a PE string is defined as a set of applications software which is required to process one CFC input message (e.g., LIFP, RS, etc.). A PE string may consist of all or some of the CFC applications processes (i.e., Input, Retrieve, Transform, Update, Output).

The system generation program, based on user-defined input specifications, gathers the program modules to be included in each PE, sets up the linkages between these modules, and builds a CFC PE for each CFC message type. In addition, a table is constructed, for subsequent use by the on-line monitor. This table provides the monitor with the mapping and control information needed for on-line loading and scheduling of the PEs.

5.1.4 Output

The output from the system generation program will be:

a.   an IPLable disk resident CFC operational
system

b.   listings containing storage allocation
map, cross reference data and system diagnostics

5.2   System Update Capability

The System Update capability will be provided to
facilitate an efficient updating of the disk-resident
system and data base. This program will make it
possible to affect changes in the CFC disk-resident
system without requiring a complete system generation
run. The changes that can be implemented through
the System Update capability include, but are not
limited to the following:

a.   Insert program patches in the operational
system

b.   Make changes to the adaptation data base
or replace by a new adaptation data base

c.   Make changes to the static data base or
replace by a new static data base

d.   Update system parameters

A system update run will produce a new IPLable CFC
disk-resident system. The output listings will
contain a record of the changes that were implemented
and diagnostic messages (if any).

5.3   Disk Dump/Restore Capability

Support programming must be provided to create
transportable copies and backup copies of the
disk-resident CFC system. The system can be
"dumped" on disk or tape and subsequently
"restored" on a disk which can be used to
IPL an operational system.

5.4   Automated System Build (ASB)*

The ASB function is currently available and under
control of OS/9020 at NAFEC. ASB eliminates,
whenever possible, the need for manual inter-
vention required in a system build process. The
following sections discuss in summary the major
capabilities of ASB.

---

*   This description of ASB was largely extracted from
draft NCP #3847 (ARD-141).

5.4.1   TRIO Generation

A TRIO is defined to consist of four members:
compressed source module, object module, com-
pressed language translator listing module, and
a cross reference of tables, items and subroutines
used by the program. The TRIO Generator produces
a new TRIO of a program to be served on permanent
storage for subsequent use in System Build. In
the Permanent Update Mode, the TRIO Generator updates
the master disk data base. Optionally, output list-
ings and punched cards can be produced.

5.4.2   Compool Analyzer

The Compool is a Jovial entity used to define
most of the tables referenced by the source
modules. If a Compool is changed, any module
that references a changed table must be recom-
piled or reassembled. The Compool Analyzer
examines the old and new Compools to determine
which tables or items were modified. A list is
then produced of all the programs which reference
these tables or items. The inventory data set
is updated to reflect the requirement to recompile
(or reassemble) these programs. The Compool
Analyzer also produces a cross reference listing
of Compool items vs. programs.

5.4.3   Library Analyzer

The function of the Library Analyzer is to detect
changes which may result from an update of the
9020 Library. Such an update may require a
recompilation of Library routines and/or user
developed application programs. The Library
Analyzer checks the new and the old Library
master file against the inventory and the cross-
reference data sets of the application programs.
In addition, the Library Analyzer invokes the
proper programs needed to perform any necessary
recompliations of Library routines and to update
the Library. Output from the Library Analyzer
consist of:

a. A listing of Library users which require recompilations due to Library changes.

b. The necessary inputs to the programs needed to perform any recompliations and Library updates.

c. A summary of all programs recompiled due to the Library Analyzer and the reason thereof.

d. A diagnostic listings (e.g., missing items, recursive calls, etc.).

e. A tree structure listings showing the interdependencies between the Library routines.

f. A set of compile listings, TRIOs and an updated Library.

5.4.4 Miscellaneous Functions

Other functions of ASB are:

- Create and maintain a system inventory containing information identifying the system and the programs (e.g., Library, Compool and build ID, data created, TRIO size, responsible organization, programmer ID, etc.).

- Maintain a system catalogue to provide the location of user-created TRIOs.

- Enable only authorized users to be accepted for a build process.

- Move tape resident TRIOs to disk.

- Validate a TRIO prior to inclusion in the build process.

- Scratch unneeded TRIO data sets.

- Enable recovery from an aborted system build process resuming from the last break point before the abend occurred.

5-6

6.   SYSTEM TEST GROUP

The primary function of the System Test Group is to
facilitate program testing, evaluation, and shake-
down.   Secondary functions include training and
demonstration.   The simulation capability enables the
operation of the CFC system in a controlled environ-
ment without requiring actual data, flight plans,
site communications, controllers, etc.   This is
accomplished through the provision of test drivers
which produce the stimuli necessary for proper opera-
tion of the CFC system.   These test drivers generate
the inputs which would normally be sent to the CFC
system, and simulate the CFC to NAS and the NAS to
CFC communication interfacing.   The user of the simula-
tion function is provided with the capability to
produce off-line any desired scenario for subsequent
testing on the CFC system.   In addition, simulated
events can also be created on-line during a system
test run.

This section specifies the functional requirements
for the System Test function of the CFC Operational
Support System.   This function is responsible for
accepting keyboard, card, or tape inputs and generat-
ing tape outputs with supporting printing/punching.
The generated tape is used as input to the CFC
operational program to simulate live inputs to
the CFC system.

6.1   Interfacility Test Support (INTEF)

6.1.1   Introduction

The primary function of this capability is to
facilitate program testing and shakedown (secondary
functions include demonstration and training).
The use of the simulation tape provides for con-
trolled program testing without requiring actual
data, flight plans, site communications, controllers,
etc.   A simulation tape is a source of simulated
real-time inputs to the CFC operational programs.

The INTER function processes user's source data on either cards or tape, and produces a tape containing each type of simulated data in a format compatible with the CFC operational program requirements.

A functional description of the INTER facility is shown in Figure 6-1.

6.1.2   Simulation Tape Generation

6.1.2.1   Modes of Operation

The INTER function will consist of five major subfunctions. Each subfunction will be differentiated by a control card which provides the parameters for that subfunction. The subfunctions will be capable of being executed in any desired order. The subfunctions are:

     1.   Simulation of NAS inputs

     2.   Simulation of controller inputs

     3.   Rerun of operational scenario

     4.   Printing of simulation tapes

     5.   Merging of simulation tapes

All subfunctions will check the associated input data for validity. Any error detected will be printed with an applicable error diagnostic. The erroneous data will not be written on the simulation tape. Processing will continue.

6.1.3   NAS Inputs Tape Format and Generation

The symbolic data will be processed and stored on a tape. The generation of the tape is discussed below.

6.1.3.1   Tape Content

The simulation tape will contain the following data:

     a.   Identification record

     b.   Time records (indicates start of real-time data)

     c.   "End of Job" record

NAS TO CFC MSGS (FP, DM, RS)
CONTROLLER MSGS

CFC INPUTS

CFC

interface

NORMAL CFC OUTPUT

ACK, REJ SUPPRESSED

SIM TAPE

SIM TAPE GEN.

NAS INPUTS
CONTROLLER INPUTS
PRINT SIM TAPE
MERGE SIM TAPES

OFF-LINE SYSTEM

ON-LINE SYSTEM
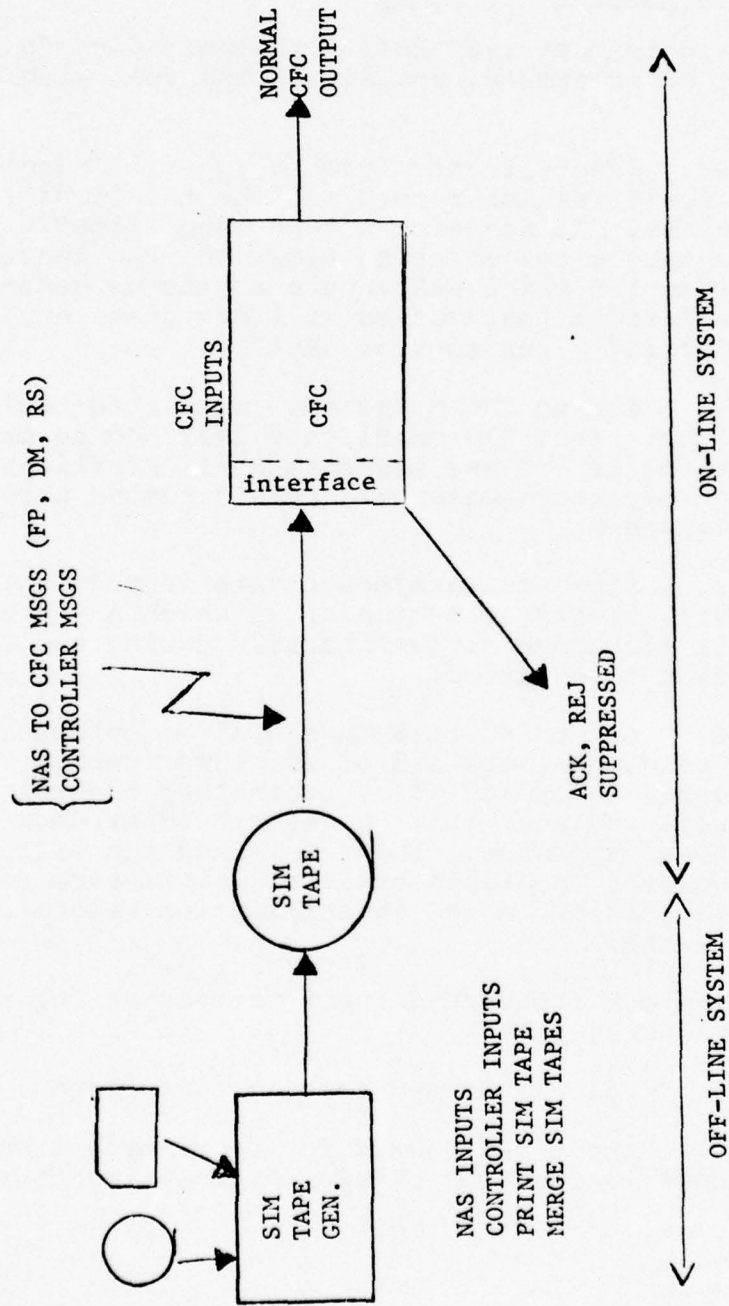
FIGURE 6-1 SYSTEM TEST SUPPORT INTERFACILITY

6-3

NAS inputs may be arranged in any order consistent
with the requirements for the simulation tape read
function.

6.1.3.2   Tape Generation

The program must read Hollerith punched cards, either
direct or prestored, containing the following symbolic
data:

> a.   Identification record (control card). The
> identification record will be the first symbolic
> entry. It contains a tape identification symbol,
> a tape clock starting time, and the length of
> time for which NAS inputs are to be generated.
> Additional parameters will designate any other
> necessary run control data.
>
> b.   NAS to CFC messages. Simulated real-time
> inputs from NAS on flights destined to pacing
> airports. These messages are: FP-flight plans,
> DM-departure messages, and RS-remove strip
> messages.
>
> c.   Site activate/deactivate indication. This
> will specify the time(s) at which any site
> is to become active/inactive during the simula-
> tion time period.
>
> d.   An end of data card will be optionally
> included at the end of the input deck. This
> card, an end-of-file, or another control card,
> will indicate that the entire input deck has
> been processed. The simulation run will,
> however, continue until the final time record
> (specified in the identification record) is
> reached.

The tape generation/NAS input processing includes
the following tasks:

> a.   Hollerith card read-in conversion.
>
> b.   Input data check for legality and complete-
> ness, and output of appropriate diagnostics.

6-4

c.   Sorting of data into ascending time
sequence.

d.   Transformation of all correct input data
into the correct formats for the operational
program(s), and writing of data on the simula-
tion tape.

e.   Generation of requested data listings.

f.   Generation of optional auxiliary outputs.

6.1.4   NAS Input Processing

The NAS input simulation subfunction will generate
NAS to CFC input messages. The following NAS messages
will be formulated and written on the simulated tape:

a.   Flight Plan message (FP)

b.   Departure message (DM)

c.   Remove Strip message (RS)

6.1.5   Controller Input Tape Generation

The controller input simulation subfunction will
generate a simulation tape containing the CFC con-
troller messages. The controller messages will be
stored on a separate tape containing nothing but
controller messages. The format of this tape will
be consistent with the requirements of the simulation
tape read function.

The inputs will be transformed into an appropriate format
and then written on the simulation tape.

Format error checking will be performed. An error
will not terminate further processing; however,
erroneous input will not be included on the simula-
tion tape.

Tape generation/controller input processing includes
the following tasks:

a.   Hollerith card and/or tape read-in

b.   Input data check for legality and com-
pleteness; output of appropriate diagnostics

c.   Sorting of data into ascending time sequence

d.   Transformation of all correct input data
into the correct format for the operational
program(s), and writing of data on the simula-
tion tape as a time record

e.   Generation of optionally requested data
listings

The program must read Hollerith punched cards, either
direct or prestored containing the following symbolic
data:

a.   Identification record (control card): The
identification record will be the first symbolic
entry. It contains a tape identification symbol
and a tape clock starting time. Additional
parameters will designate any other necessary
run control data.

b.   CFC controller input messages. Any of the
controller messages (e.g., CAPS, LIFP, etc.)
can be simulated.

c.   An end-of-data command to signal the end
of the simulated controller input messages.

6.1.6   Rerun of Operational Scenario

The program shall be capable of reading the system log
tape from the on-line system and generating a simulation
tape directly from it.

6.1.7   Printing

All simulation printing will be optional, and is
divided into two categories:

1.   Input data printing

2.   Simulation tape printing

6.1.7.1   Input Data Printing

The parameters contained on the NAS input and
controller input function control cards will
designate the desired input data printing.

6-6

Output listings will include the listings of the
input card deck in unsorted order or in sorted
order according to the user request.

6.1.7.2   Simulation Tape Printing

The printing of simulation tapes will be a subfunction
of the simulation. Selectivity of both the amount and
the contents of the data to be printed will be
available. The parameters designating the amount
of printing will be contained on the subfunction
control card. Similarly, parameters specifying the
contents to be printed will be designated on data
cards. Content selectivity will include:

    a.   Specific messages

    b.   Specific NAS sites

6.1.8   Simulation Tape Merging

The simulation will have the ability to merge any two
or more simulation tapes. Selectivity of the data to be
merged from either tape will be available. The
parameters designating the tapes to be merged will
be contained on the subfunction control card. Similarly,
the parameters designating the data to be deleted will
be contained on data cards.

6.1.9   CFC-Simulation Tape Interface (CSI)

CSI is a special software interface which will be
required in the operational CFC system. This inter-
face will facilitate the testing of the CFC system
with the INTER simulation tape. CSI will perform
the following functions:

    a.   Read the simulated messages off the simula-
        tion tape, interpret them, and place the
        information in the appropriate storage
        buffer(s) for subsequent action by the
        CFC system.

    c.   Perform the handshaking functions normally
        required in accordance with the communica-
        tions protocol. The ACK and REJ messages
        produced by CFC will be received by CSI,
        analyzed and discarded. Similarly, CSI
        will generate simulated ACK/REJ input mes-
        sages if and when these are required as
        input to CFC.

6.2   Intrafacility Test Support (INTRA)

6.2.1   Introduction

The INTRA capability will be used for more extensive testing of the Central Flow Control System (CFC) than is possible with INTER (see Section 6.1). INTRA will aid the user in preparing simulated input to CFC from NAS sites without actually linking CFC to the NAS sites. This simulated input will be sent to NAS over the standard NAS/CFC interface channel hardware.

In addition to providing data from simulated NAS centers, INTRA will print diagnostic listings and input message listings. A functional description of the Intrafacility Test function is shown in Figure 6-2.

6.2.2   System Environment

INTRA will be run on the off-line CE.

System configuration will include the off-line CE and the operational CFC duplex system in a computer-to-computer communication link. The off-line system peripheral equipment (e.g., tape, keyboard) will be used to generate real time NAS messages and to record the CFC to NAS messages (e.g., acknowledgement, rejection).

6.2.3   System Operation

Communication between the operational system that is under test and the INTRA system residing in the off-line facility, for NAS-to-CFC and CFC-to-NAS simulation, will be performed using the standard interface hardware equipment.

6.2.3.1   NAS-to-CFC Communications

The user will be able to build a simulation tape with messages from any specified NAS center without any knowledge of the actual communications protocol between NAS and CFC.
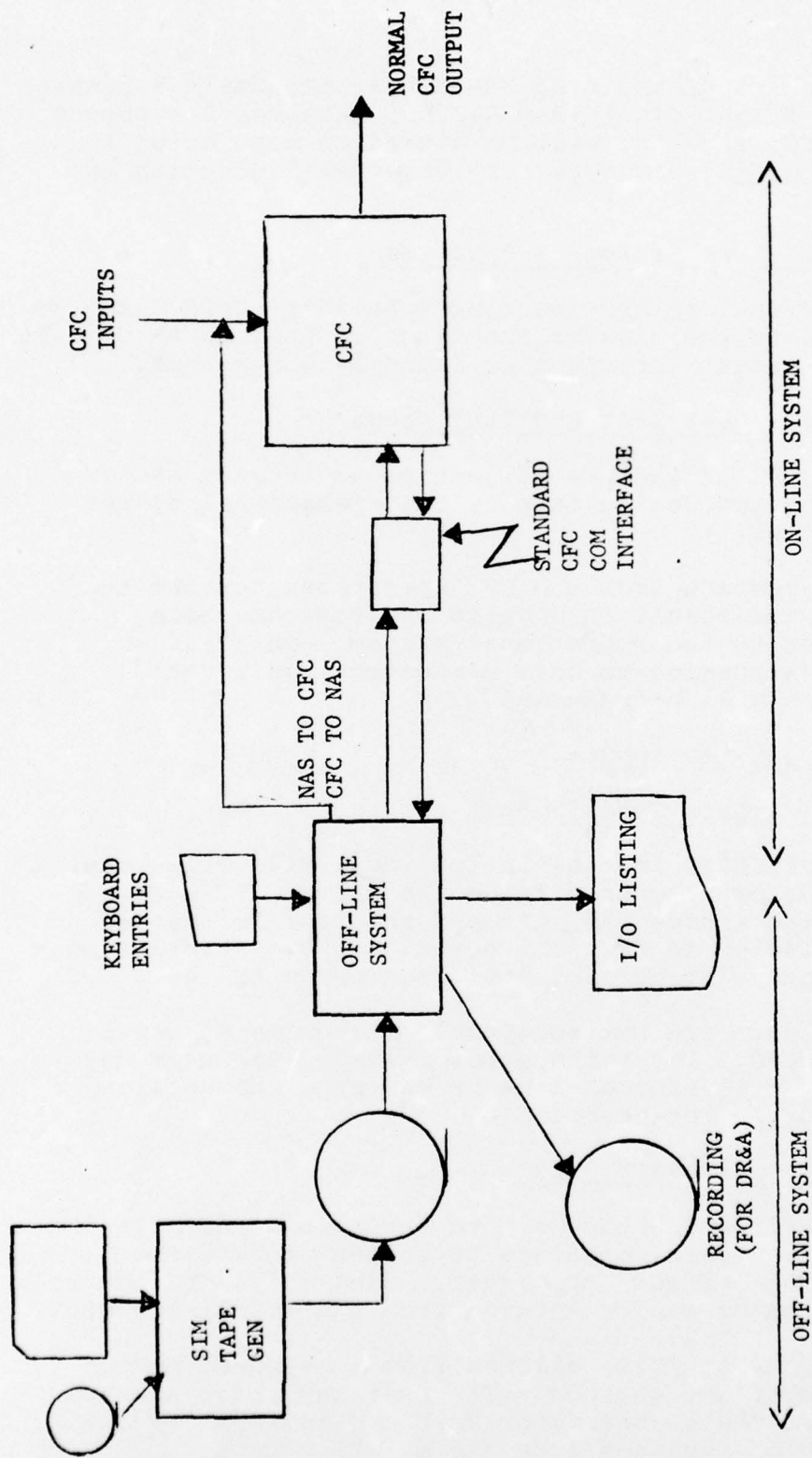
6-8

FIGURE 6-2 SYSTEM TEST SUPPORT INTRAFACILITY

6-9

When an INTRA-simulated NAS center transmits a message
(e.g., flight plan) from NAS and receives a response
from CFC, an entry will be stored on tape or disk
of the off-line system for subsequent reduction and
analysis.

6.2.3.2   Generation of Responses

To test the message checking functions of CFC and the
message retransmission functions of NAS, INTRA will be
able to issue incorrect or incomplete messages.

6.2.3.3   Data Test and Test Message

INTRA will be capable of sending any test messages
from the simulation tape to the operational system
under test.

A test message from the CFC operational system to
INTRA will result in a valid NAS response being
returned to the operational system. On a random
basis (according to user parameters) an invalid
response will be returned.

6.2.4   System Inputs

6.2.4.1   Card Tape Input

A normal INTRA initialization input file will consist
of cards or tape specifying the NAS to CFC message
data, the source NAS site and the time for message
transmission to CFC.   In addition CFC controller mes-
sages can also be simulated from cards or tape.

If the data are not acceptable, error messages will
be printed. After the error message, the user may
correct his error on-line by entering the entire
message via the keyboard.

6.2.4.2   Keyboard Input

INTRA will allow the user to enter, via the off-line
system keyboard, messages to be sent to CFC from
any INTRA-defined NAS center. Similarly, CFC control-
ler messages can be entered from the off-line system.

The inputs to INTRA allowed from a keyboard may be
entered by two methods. The user may enter a message
live via the keyboard, or keyboard entries may invoke
reading of messages from a simulation tape.

6.2.4.3   On-Line Recordings

The on-line recorded data tapes can also be used in order to send simulated messages to the CFC system during an INTRA operation.

6.2.5   System Output

INTRA will record on tape (and optionally list on the off-line printer) each message transmitted to CFC and each response from CFC as it is received. The identification of the NAS center from which the message was sent and the transmission time, will be contained in the message. INTRA will list the input messages as a default option.

The INTRA tape recorded on the off-line system during an INTRA test will be subsequently reducable by a special data reduction program running on the off-line system. The output from this program will provide, according to user specified parameters, complete or selective listings of the messages that are contained on tape and statistics on the volume and frequency of these messages.

Error messages generated by INTRA will be sent to the off-line system printer. An error message will consist of the test time (i.e., HH:MM:SS), a description of the error, and the message that caused the error.

6.3   Automated Unit and String Test Drivers

6.3.1   Introduction

The purpose of the Automated unit and string test drivers in the Central Flow Control System shall be to insure that each subprogram performs as designed before integrating it into the total program. The test program shall be so designed that it will be capable of testing a string of subprograms as well as each individual unit within the string. These tests will be conducted while varying the input parameters.

6.3.2   System

Automated unit and string test drivers will be
disk resident programs run on the off-line system
under the off-line monitor. No communication with
the on-line Central Flow Control (CFC) system
will be required for these programs; however,
the CFC compool will be used as input.

6.3.3   Operational Concept

Operationally the unit or string undergoing test
will be treated as if it were the total operational
(on-line) program. This will require that enough
of the "on-line" system environment be loaded with
the test driver to accommodate the concept.

Input to unit or string tests, as a minimum, will
consist of:

  1.   On-line program environmental data

  2.   Test control data

  3.   Utility Programs (i.e., library routines,
        printout, etc.)

  4.   Any functions which must be simulated to
        facilitate testing

  5.   The unit or string undergoing testing

During operation of the test driver programs it
will be possible to vary the pertinent parameters
using at least three methods:

  1.   Pre-stored as test control data

  2.   Through pre-loaded input devices (e.g.,
        card reader)

  3.   Under programmer control on-line using
        such devices as controlled stops to
        allow manual input.

6-12

Output will be completely under control of "Test Control Data" input. The minimum options available will be:

1. Magnetic tape recording

2. Disc Recording

3. On-line print-out

The data recorded by the above methods will be under programmer control through the use of test control data.

6.3.4 Program Load

It will be possible to load the unit or string to be tested and the required support data from a minimum of the following:

1. Magnetic tape

2. Disc

3. Card

4. Any available input device (e.g., input/ output typewriter)

5. Any combination of the above.

7.   MANAGEMENT AIDS GROUP

The Management Aids Group includes a set of
tools which will be used in the overall control
of the CFC program development test and mainten-
ance. The CFC system represents a major invest-
ment in time and money. Proper and timely control
over this effort will help reduce the project cost,
improve software reliability and enhance communica-
tions between the various project teams.

7.1   Program Evaluation Review Technique (PERT)

PERT is currently being installed at NAFEC under
OS/9020. It is planned that PERT will be used
as a management planning and control tool through-
out the duration of the CFC system development.
The purpose of PERT is to help management to:

    .   logically plan the project

    .   measure current status

    .   forecast future progress and problems

    .   demonstrate the effect of changes on
         established schedules

The PERT system is a tool which helps in visualizing
the progress of the individual project activities
from proposal to completion. The continued use
of PERT throughout the project will inform manage-
ment of current or potential future problems which
would impact the projected completion date of either
the project as a whole or its activities.

7.1.1   Input

PERT accepts user inputs defining all the events
and activities in the PERT network. Also provided
by the user are various control parameters outlining
the optional reports, the title and the date.

7.1.2 Processing

The PERT computer programs process the input data
giving accurate information on activity status
within the given project. Concise information about
time, resource, and responsibility are contained in
the various reports generated by the system.
Within limits PERT will detect various errors in
the input data, such as sequence errors, duplications,
events and/or activities which are not logically
connected to the PERT network, or loops (events
which prohibit the PERT network from reaching
the end event).

7.1.3 Output

When processing is complete the reports requested
by the user are printed out. The reports provide
a variety of useful information. The reports
produce listings of events and/or activities sorted
by various keys such as preceding or succeeding
events, criticality, expected data, responsible
organization or allowed date. In addition, error
reports are produced, if applicable.

7.2 Program Analyzer

7.2.1 Description

This system is intended to reduce the cost of assuring
that software systems written in the JOVIAL language
are comprehensively tested. It is comprised of a
series of tools which provide

- A means of measuring the effectiveness
  of both individual and cumulative
  software test cases

- A capability to facilitate the construction
  of test data that will thoroughly exercise
  the software

- An analysis of retesting requirements
  following software modifications

---

* The description of Program Analyzer was largely
extracted from Report RADC-TR-76-20, February 1976.
The RADC report describe the JOVIAL Automatic Verifica-
tion System (JAVS).

7-2

These tools are intended to be applied during program
testing to aid in the exercising of untested program
paths and the identification of test cases appropriate
to improvement of testing coverage. All of these
features are provided by analysis of program struc-
tures, instrumentation of the system through inser-
tion of appropriate software probes to measure testing
coverage, and comprehensive reports which pinpoint
paths in the program structure that remain to be
exercised. In addition, guidance is provided for the
generation of testcases that will assure coverage
of the untested portions. This function can be thought
of as a partner during the testing process, supplying
a wide variety of automated aids to the user in
comprehensive testing activities.

A functional overview is shown in Figure 7-1.

7.3 Hierarchy Plus Input-Process-Output (HIPO) *

7.3.1 HIPO Description

HIPO is a design aid and a documentation technique.
It describes the function of a system from the general
to the detail level, thereby providing a logical
extension of a top-down development effort. It
graphically shows what a system or program does and
what data it uses and creates.

- As a design aid, HIPO can help define
  the functional structure and identify
  the scope of the programming assignment.

- As a documentation tool, HIPO can make
  a significant portion of the documentation
  of a system a byproduct of the develop-
  ment process and can reduce the time
  needed for locating, understanding and
  altering segments of code.

---

* The description of HIPO in this section was largely
  extracted from IBM manual GC20-1851-0.

```
┌──────────────┐
│ YOUR JOVIAL  │◄─────────
│ SOURCE       │
└──────────────┘
```
JOVIAL source code is input for processing and analysis. A special form of comment (optional) inserted by the user directs the Analyzer processing for program performance analysis.

```
┌──────────────┐
│ SOURCE TEXT  │◄─────────
│ ANALYSIS,    │
│ STRUCTURAL   │
│ ANALYSIS     │
└──────────────┘
```
The Analyzer analyzes the code and generates a directed graph of the control structure. (STEP1) The possible flows through the program are determined. All pertinent data is stored in (STEP2) a data base for later use. Additional or changed source code causes an existing data base to be updated.

```
┌──────────────┐
│ MODULE       │◄─────────
│ INSTRUMENTATION │
└──────────────┘
```
The Analyzer automatically inserts software (STEP3) probes into the source code to intercept and record program flow during execution. A second type of instrumentation is used to record statistics on program performance according to directives inserted by. the user in the source.

```
┌──────────────────┐
│ MODULE TESTING   │◄─────────
│ ASSISTANCE AND   │
│ SEGMENT ANALYSIS,│
│ RETESTING GUIDANCE│
│ AND ANALYSIS     │
└──────────────────┘
```
The Analyzer provides a variety of services (STEP4) which assist in the establishing of a (STEP5) continuing testing strategy both at the detailed level within a module and at the module interdependence level. Test cases are constructed by the user.

```
┌──────────────┐
│ TEST EXECUTION,│◄─────────
│ PROGRAM      │
│ PERFORMANCE  │
└──────────────┘
```
Program execution provides normal computa- (TEST) tional results as well as outputs from the instrumentation. Structural flow output is recorded for later post-test analysis. User-directed program performance output is interspersed with normal program output.

```
┌──────────────┐
│ TESTING      │◄─────────
│ EFFECTIVENESS│
│ MEASUREMENT  │
└──────────────┘
```
The Analyzer includes detailed post-test (STEP6) analysis facilities which provide measures of testing thoroughness, both individually and cumulatively for a set of test cases.

```
      ◇
    TEST GOALS ◄─────────
    ACHIEVED
      ?
      │
     YES
```
The results are examined by the user to determine if test goals have been met and testing is completed.
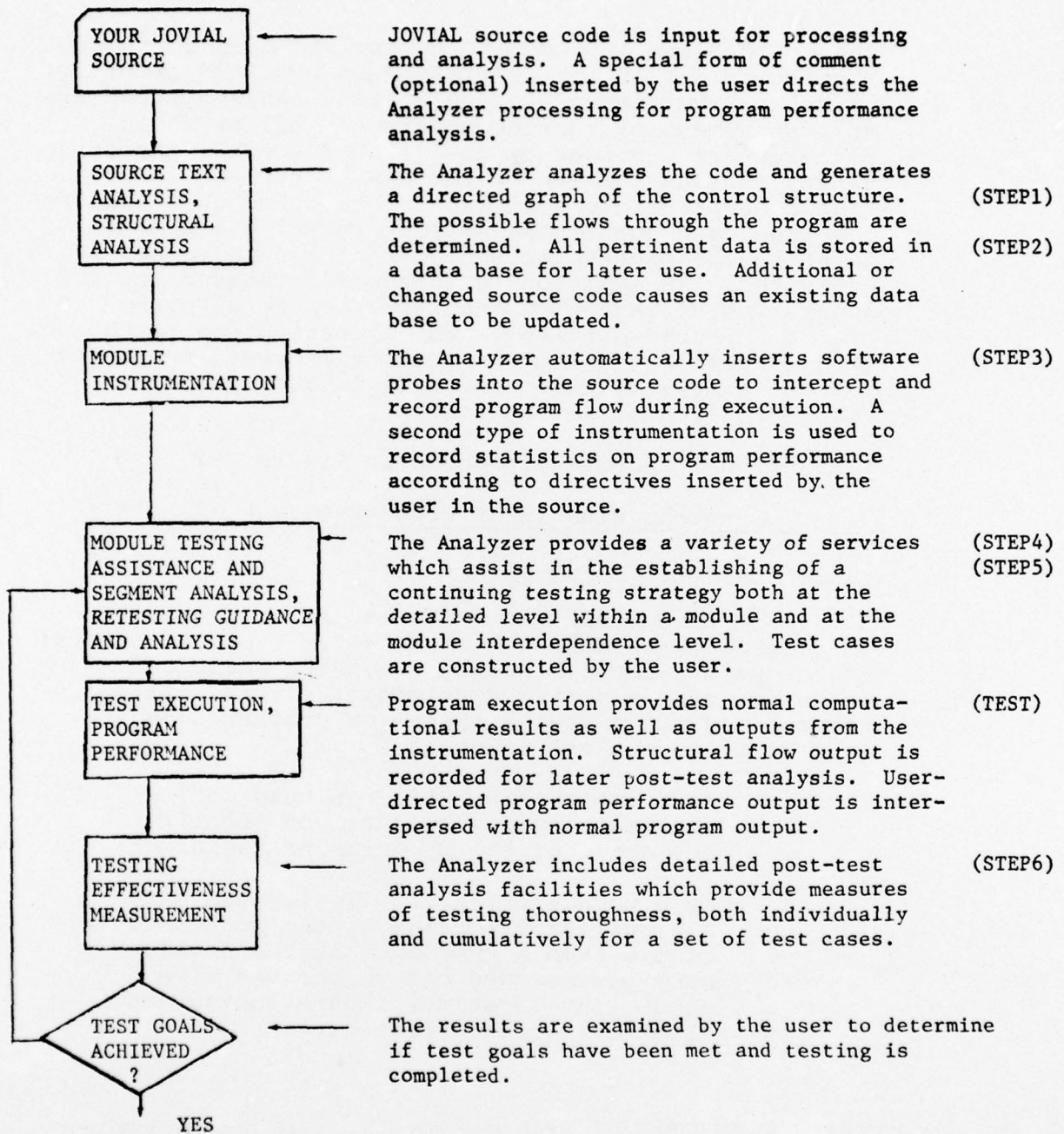
FIGURE 7-1 AN OVERVIEW OF THE PROGRAM ANALYZER
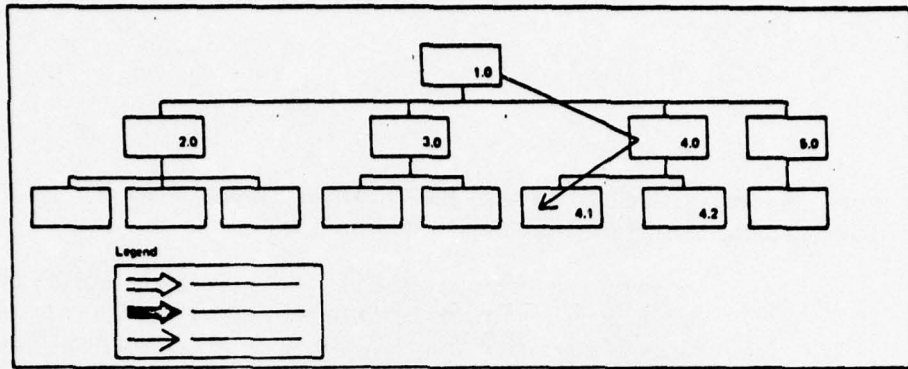
Source: RADC-TR-76-20

The major objectives of HIPO as a design and documentation technique are to:

- Provide a structure by which the functions of a system can be understood. The diagrams are organized in a hierarchy structure much like an organization chart, where each diagram at any level is a subset of the level above it.

- State the functions to be accomplished by the program rather than specify the program statements to be used to perform the functions.

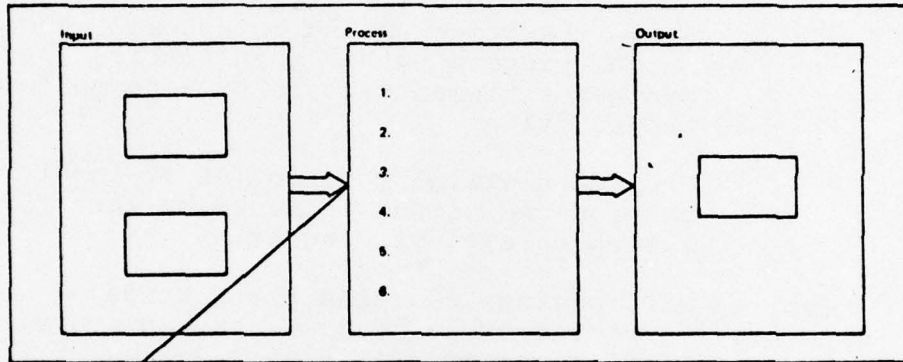- Provide a visual description of input to be used and output produced by each function for each level of diagram.

A typical HIPO package contains three kinds of diagrams: a visual table of contents, an overview and detail HIPO diagrams (see Figure 7-2).

1. Visual table of contents - This diagram contains the names and identification numbers of all the overview and detail HIPO diagrams in the package and shows the structure of the diagram package and relationship of the functions in a hierarchical fashion.

2. Overview diagrams - High-level HIPO diagrams, called overview diagrams, describe the major functions and reference the detail diagrams needed to expand the functions to sufficient detail. The overview diagrams provide, in general terms, the inputs, processes, and outputs.

3. Detail diagrams - Lower-level HIPO diagrams contain the fundamental elements of the package. They describe the specific functions, show specific input and output items, and refer to other detail diagrams.
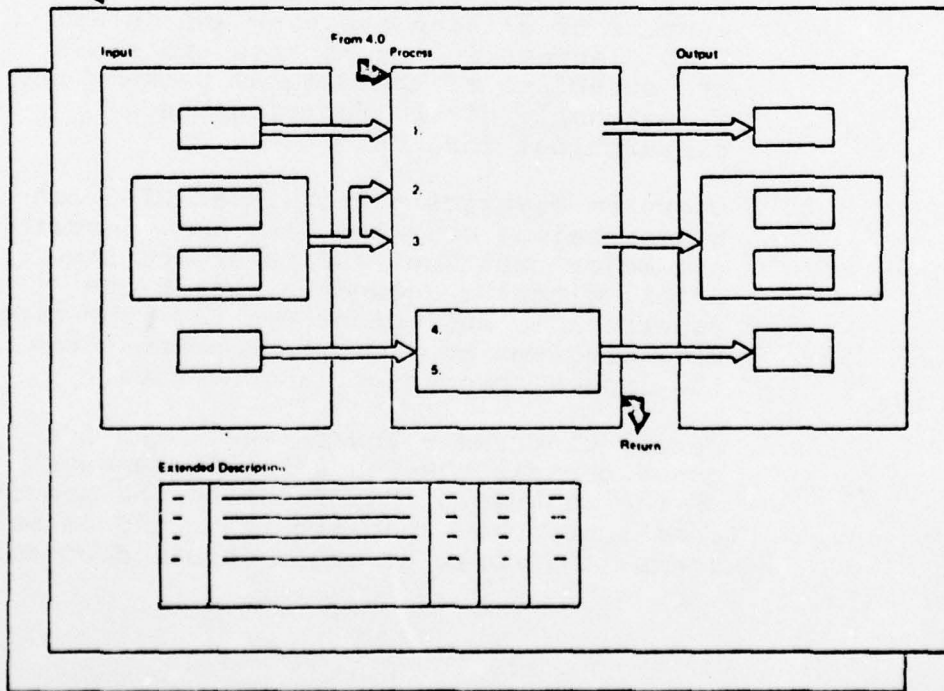
FIGURE 7-2   A TYPICAL HIPO PACKAGE

7-6

7.3.2 HIPO Application in CFC

In the CFC development effort, the complete set
of HIPO documentation and diagrams will be assembled
and maintained as part of the CFC support library.
The librarian is responsible for entering all new
information into the development support library
and to maintain an up-to-date HIPO documentation
package. As CFC operational programs are updated
and incorporated, within the system build (see
Section 5) assembly comprising of all the CFC
operational programs, the appropriate HIPO diagrams
will be concurrently updated. This will insure
that the documentation is current. It will also
help reduce the possibility of discrepancies between
interfacing modules caused by failure to coordinate
changes involving several program modules.

7.4 Automated Code Auditor

7.4.1 Introduction

The constantly increasing cost of producing opera-
tionally acceptable computer programs has resulted
in a directly proportional increase in the concern
of systems managers with code integrity.

In large scale systems, such as the Central Flow
Control System, employing many thousands of lines
of computer code, there is a gap between achieving
diagnostic free compilation and operationally accept-
able programs which, traditionally, has been filled
by programmer inspection. This has resulted in a
situation where it is not uncommon, in large
systems, to find large amounts of code which is
never exercised by the program as well as large
amounts whose only purpose is to correct errors
produced by other code.

It is this gap which the automated code auditor is
intended to fill.

7.4.2   System

Automated code auditor will be a disk resident
program utilizing the "off-line" system.   The
program will test subprograms as units and strings
as well as testing the total program.   This will
result in an appreciable variation in the off-line
system resources required for any specific run of
the program.   No direct interface with the on-line
system will be required; however, some elements
of the on-line system, such as compool, will be
used for the automated code auditor.

7.4.3   Operational Concept

The automated code auditor will begin with a
diagnostic free program element which has success-
fully passed unit and string testing.   It will be
designed to audit the program for code integrity
by flagging such things as code which cannot be
accessed, code which merely corrects other code
induced errors, etc.

Input to the automated code auditor will consist of:

1.   On-line program environmental data

2.   Test control data including simulated
     flight plans, if required

3.   Any required simulated functions

4.   The program element undergoing test

Output will include programmer defined items as well
as diagnostic messages pertaining to detected coding
faults.   The minimum options for output will be:

1.   Magnetic tape recording

2.   Disc Recording

3.   On-line print-out

7-8

7.4.4   Program Load

It will be possible to load the program element
to be tested and the required support data from
a minimum of the following:

1.   Magnetic tape

2.   Disc

3.   Card

4.   Any available input device (e.g., input/
      output typewriter)

5.   Any combination of the above