Aug 29/79 All S/A's
- D. McBride
- O. Townsend
- C. Gibbs
- B. Zado
- M. Strain
- E. Hanson, H.O.

The following working paper describes the use of the Common Area feature available in IMS/90 Release 6.

SPERRY UNIVAC
Philadelphia Software Development Center

Working Paper
Data Management Systems

Working Paper #6964-__*-__060__ Revision #__2__
(*D=Database, U=Utilities, A=Applications)

Title: Common Area

Abstract:

This working paper describes a method to implement Common Area feature for IMS 90.

Author: R. Vohra

Date: 09/27/78

Distribution:

J. J. Sweeney	C. Casals	J. Merman
R. D. Pratt	M. Lombardo	R. Winkelman
M. N. Theodore	K. Koenig	P. Reichmuth

SPERRY UNIVAC COMPUTER SYSTEMS

1.0 General Description

Several installations have expressed a need for main storage resident common area. This common area can be used to keep vital information which can be accessed by all the terminals. Records in this area will be lockable and recoverable. Criteria for the support of common area is efficiency, flexibility, and ease of use. We are proposing, that common area be a main storage resident ISAM file. It is a multi-record file. User can issue GET, GETUP or PUT on any record of this file as he currently does for an ISAM file. IMS 90 record locking procedure will hold for this file.

2.0 To implement the common area feature the following components of IMS 90 are affected:

a. Configurator
b. Startup
c. On-line
d. Shutdown

a) Configurator

Common area is main storage resident, variable blocked or fix blocked ISAM file. Any number of files can be main storage resident. Beside supporting the current options in the FILE section, two additional keyword parameters are supported for common area file.

- CAFILE = {YES / NO} specifies that the file is a common area file. Default is NO. If CAFILE=YES is specified, then configurator will set the bit X'80' in the flag byte ZF#FFLG2 of the File Control Table (FCT).

- CUPDATE = {YES / NO}

This option will allow the user to specify whether updating of the disk images is necessary. Default is NO. If YES option is selected, a bit in the FCT (bit X'40' byte ZF#FFLG2) is set.

- TRACE = NO option will be valid for common area files.

- If LOCK = UP is specified, then no before images are written to the AUDIT file. I.E. no on-line recovery takes place in case of any error in the user action program.

b) Startup

During startup procedure of IMS 90, common data area file will be loaded in main storage. For error diagnostics, refer to technical documentation by P. Reichmuth.

c) On-Line Processing

User is allowed to issue GET, GETUP, and PUT IMS 90 function calls for a common area file. All the applicable error codes are posted in the PIB.

- On-line and off-line recovery procedure is valid for common area files if user chooses recovery options.

- Common area file requests are handled by the common data area modules (ZF#CSAS for S/T and ZF#CSAM for M/T). The general request processor module will pass control to the appropriate common data area module.

- No changes are required to on-line recovery, off-line recovery or warm restart.

d) Shutdown

During shutdown, common data area files will be updated using the data in main storage.

1. TITLE

ZF#CSAS. This module handles all requests to a CSA*-file in memory for single-thread IMS 90.

*CSA = Common-Storage-Area

1.1 Purpose

Several installations have expressed a need for main storage resident common area. This common area can be used to keep vital information which can be accessed by all the terminals. Records in this area will be lockable and recoverable. Criteria for the support of common area is efficiency, flexibility, and ease of use. We are proposing, that common area be a main storage resident ISAM file. It is a multi-record file. User can issue GET, GETUP or PUT on any record of this file as he currently does for an ISAM file. IMS 90 record locking procedure will hold for this file.

DESIGN DESCRIPTION FOR COMMON DATA AREA

General

The following modules manipulate the requests to the CSA files which are in memory.

ZB#STRTN	during Startup processing
ZF#GEN/ZF#CSAS	during On-Line
ZB#SHUTN	during Shutdown Processing

Before the module ZF#GEN branches into ZF#CSAS, the FCTs are searched and bit Ø of ZF#FFLG2 is checked if it is on (X'8Ø').

If there is an occurrence, control is given to the start-label of ZF#CSAS with a BALR instruction. Following the BALR, a half-word value indicates from where control came.

Values of the Halfword

Value	Set by/for
H'Ø'	On-Line GETUP/PUT/GET
H'4'	On-Line Termination
H'8'	On-Line Cancel

Figure 2.1

2.2 STARTUP

2.2.1 Moment of Loading

Because the original CSA-files could be affected by a restart facility, loading of the records into memory starts immediately before Startup branches to Phase 2 of IMS 90.

Due to this reason, the high address of the storage pool has to be carried to this point (restoring of register 4 into ZB#MEHI at line 449Ø).

2.2.2 Memory Allocation for CSA

The item ZT#HPIBA of the THCB contains the first free location in the storage pool. This will be the Start Access of CSA. To allow IMS to load subsequent nonresident Action Programs, 8K are kept free at the top of storage pool. During the load of the CSA-records, this area is used as I/O-area for the CSA-files.

The CSA Start-Address (contents of ZT#HPIBA) will be stored into a new added fullword in the GEN-module (ZF#CSADR). ZT#HPIBA in the THCB will be updated after successful CDA-load.

2. DESIGN DESCRIPTION FOR COMMON DATA AREA

2.1 General

The following modules manipulate the requests to the CSA files which are in memory.

ZB#STRTN	during Startup processing
ZF#GEN/ZF#CSAS	during On-Line
ZB#SHUTN	during Shutdown Processing

Before the module ZF#GEN branches into ZF#CSAS, the FCTs are searched and bit Ø of ZF#FFLG2 is checked if it is on (X'8Ø').

If there is an occurrence, control is given to the start-label of ZF#CSAS with a BALR instruction. Following the BALR, a half-word value indicates from where control came.

Values of the Halfword

Value	Set by/for
H'Ø'	On-Line GETUP/PUT/GET
H'4'	On-Line Termination
H'8'	On-Line Cancel

Figure 2.1

2.2 STARTUP

2.2.1 Moment of Loading

Because the original CSA-files could be affected by a restart facility, loading of the records into memory starts immediately before Startup branches to Phase 2 of IMS 90.

Due to this reason, the high address of the storage pool has to be carried to this point (restoring of register 4 into ZB#MEHI at line 449Ø).

2.2.2 Memory Allocation for CSA

The item ZT#HPIBA of the THCB contains the first free location in the storage pool. This will be the Start Access of CSA. To allow IMS to load subsequent nonresident Action Programs, 8K are kept free at the top of storage pool. During the load of the CSA-records, this area is used as I/O-area for the CSA-files.

The CSA Start-Address (contents of ZT#HPIBA) will be stored into a new added fullword in the GEN-module (ZF#CSADR). ZT#HPIBA in the THCB will be updated after successful CDA-load.

2.2.3 Memory Layout before CSA-Loading

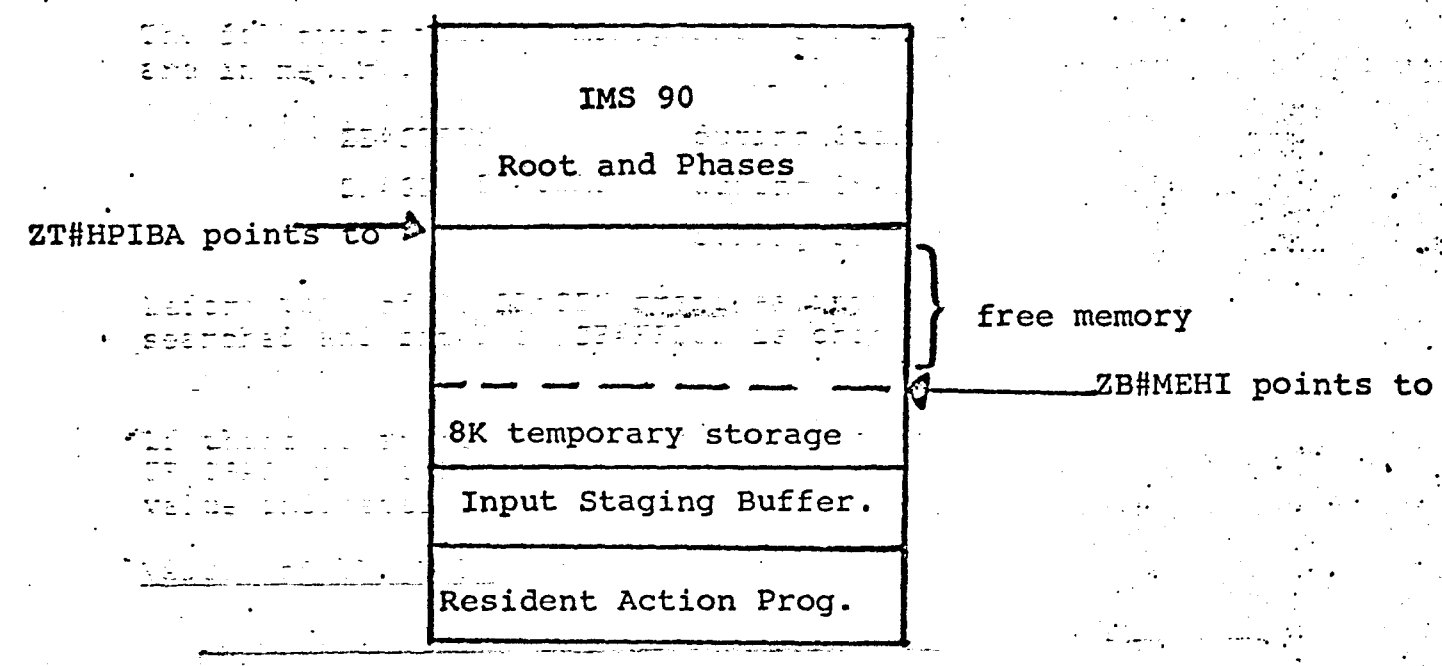


Figure 2.2

2.2.4 Memory Layout after CSA-Loading

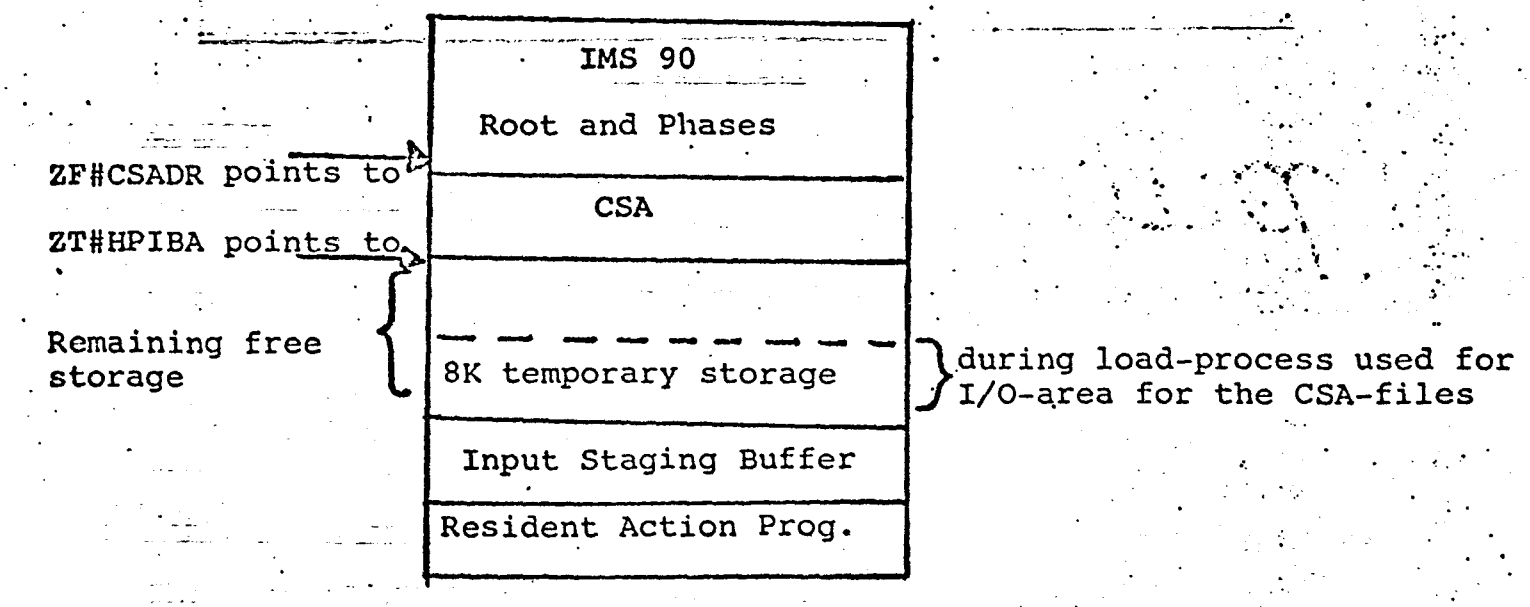


Figure 2.3

2.2.5 Loading Procedure

All FCT's are sequentially searched for the X'8Ø' value in the ZF#FFLG2-byte.

If such an FCT exists, the corresponding file is read sequentially into memory. The format of the memory CSA-file is shown in Figure 2.4.

2.2.6 Format of the CSA in Memory

Length	Name	Function	Values
1	ZF#DCSFT	Filetype	*
7	ZF#DCSFN	Filename	Copy of ZF#FNAME
4	ZF#DCSFL	File length in bytes	
4	ZF#DCSIX	Index Table Address	**
8		Identification	Copy of ZC#TTRID Nonzero = locked
(2)		Record-length (if variable)	
V		Key and Data of Record	
1		End of current file	X'FF'
1		End of CSA	X'FF'

per file per record

Figure 2.4

* See figure 2.5. ** See figure 2.6.

2.2.7 Filetype Byte in the CSA

	LOCK = TR	LOCK = UP	CUPDATE = YES	CUPDATE = NO	Filetype-Value
TRACE = YES	X		X		ØØ
TRACE = YES	X			X	2Ø
TRACE = YES		X	X		4Ø
TRACE = YES		X		X	6Ø
TRACE = NO	X		X		8Ø
TRACE = NO	X			X	AØ
TRACE = NO		X	X		CØ
TRACE = NO		X		X	EØ

Figure 2.5

Filetype-Byte:	Bit Ø	Ø	TRACE = YES
		1	TRACE = NO
	Bit 1	Ø	LOCK = TR (Recovery wanted)
		1	LOCK = UP (No recovery wanted)
	Bit 2	Ø	CUPDATE = YES
		1	CUPDATE = NO
	Bits 3-6	always Ø	
	Bit 7	Ø	Fixed length records
		1	Variable length records

2.2.8 Failures

There are two possibilities a failure can occur during load:

- I/O Error on a specific CSA-file
- not enough memory available to hold all records of a file in memory

For both cases, the already used memory for that file is released and bit Ø of ZF#FFLG2 in the corresponding FCT is reset. (The CSA-file will be a normal file for the future processing).

A warning message is issued for the appropriate error and processing continues.

```
W  CSA  LOAD I/O-ERROR  FOR  FILE  *--------*

W  CSA  LOAD ERROR  FOR  FILE      *--------*   MEMORY EXHAUSTED
```

2.2.9 Restriction on the I/O-Area-Size of CSA-Files

Due to the use of the "dummy-reserved" 8KB block as I/O-area for the CSA-files, blocks may not exceed this value.

Filetype-Byte:	Bit Ø	Ø	TRACE = YES
		1	TRACE = NO
	Bit 1	Ø	LOCK = TR (Recovery wanted)
		1	LOCK = UP (No recovery wanted)
	Bit 2	Ø	CUPDATE = YES
		1	CUPDATE = NO
	Bits 3-6	always Ø	
	Bit 7	Ø	Fixed length records
		1	Variable length records

2.2.8 Failures

There are two possibilities a failure can occur during load:

- I/O Error on a specific CSA-file
- not enough memory available to hold all records of a file in memory

For both cases, the already used memory for that file is released and bit Ø of ZF#FFLG2 in the corresponding FCT is reset. (The CSA-file will be a normal file for the future processing).

A warning message is issued for the appropriate error and processing continues.

```
W  CSA  LOAD I/O-ERROR  FOR  FILE  *--------*

W  CSA  LOAD ERROR  FOR  FILE      *--------*   MEMORY EXHAUSTED
```

2.2.9 Restriction on the I/O-Area-Size of CSA-Files

Due to the use of the "dummy-reserved" 8KB block as I/O-area for the CSA-files, blocks may not exceed this value.

.3 On-Line

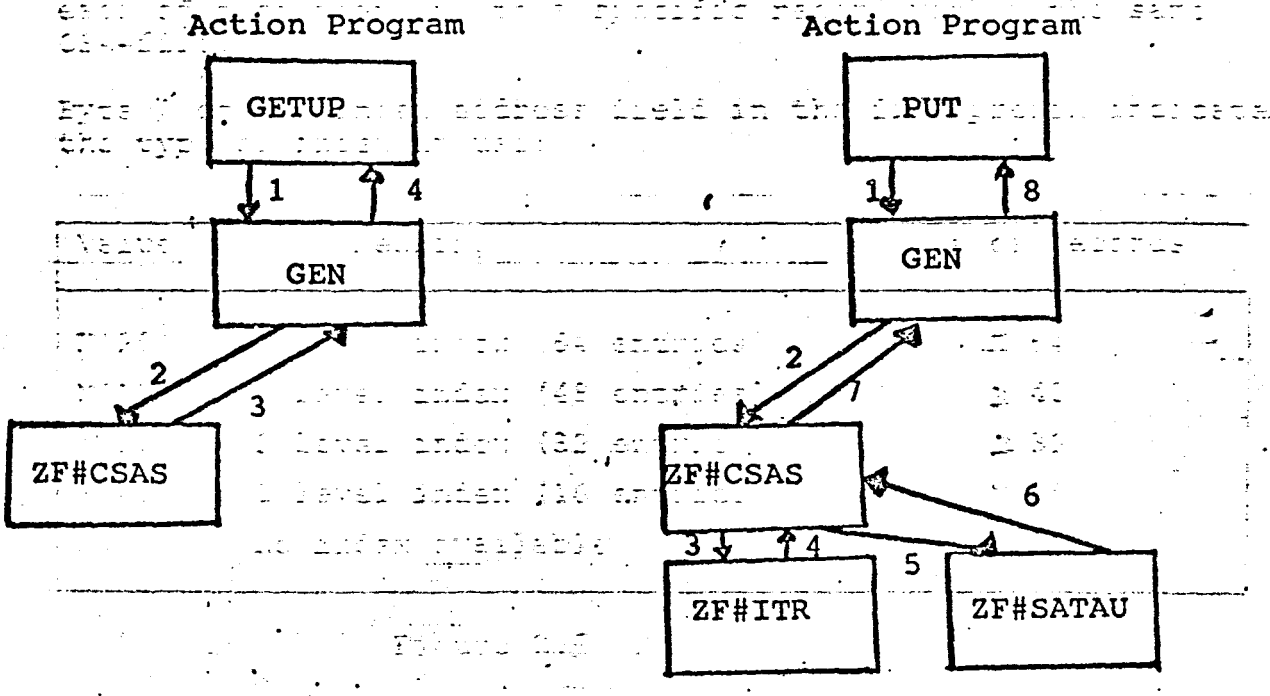
.3.1 General

ZF#CSAS is invoked by three kinds of requests:

- normal requests like GET, GETUP or PUT from user action programs
- Rollback functions (GETUP and PUT) requested by the ZS#ROL action program
- upon termination of an action/transaction

2.3.2 Normal Requests for CSA

a) Control flow for an update



GETUP

1. The action program issues a GETUP for a CSA file. Control is passed to the GEN module.

2. GEN passes control to ZF#CSAS, which locates and retrieves the specific record to the user's work area and manipulates the lock for the record.

3-4 Control returns to the action program.

PUT

1. The action program issues a PUT for a CSA file. Control is passed to the GEN module.

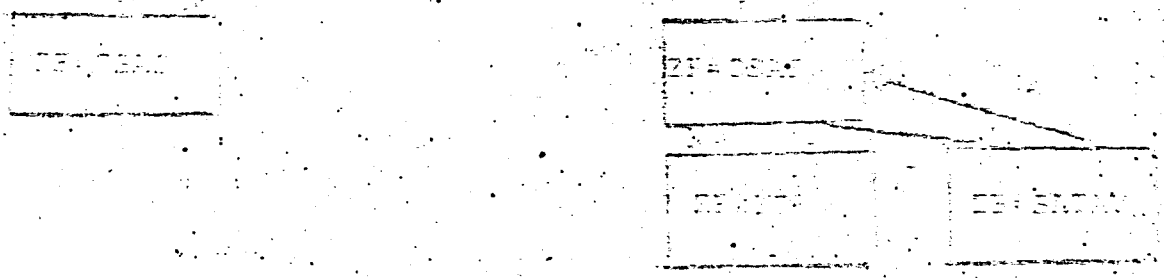
2. GEN passes control to ZF#CSAS, which locates and retrieves the specific record from the user's work area to the CSA.

3-4 Depending on the filetype-byte in the CSA, ZF#CSAS activates ZF#ITR to write before and after images to the trace file.

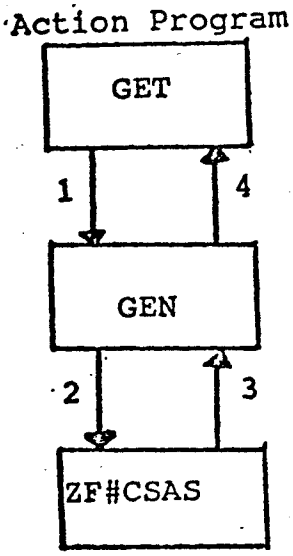
5-6 And/or before images to the audit file with the ZF#SATAU module.

When control returns to ZF#CSAS and the CUPDATE=YES option was specified, the specific record is written back to disk. Finally, the main-storage-record will be changed and, if lock=up is specified, the lock released.

7-8 Control returns to the action program.



b) Control flow for a GET



1. The action program issues a GET for a CSA file. Control is passed to GEN.

2. GEN passes control to ZF#CSAS. The specific record is read out of the CSA and written into the specified user work area.

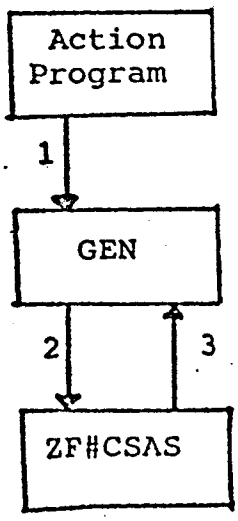
3-4 Control returns to the action program.

2.3.3 Rollback Functions from ZS#ROL

If bit 6 of ZC#TST1 in the terminal control table (TCT) (X'Ø2'= Rollback in Process) is set, it can be determined if the request comes from ZU#ROL.

The same control flow applies as for a normal update request. (see 2.3.2), except that no before/after images are written to the audit/trace-file.

2.3.4 Termination of an Action/Transaction



1. An action terminates. (e.g. - PC, abnormal or normal termination). Control is passed to the GEN module.

2. GEN activates ZF#CSAS to release all pending locks for this transaction.

3. Control is returned to GEN.

2.3.5 Record-Lock-Mechanism in Memory

As described in Figure 2.4, each record in memory is prefixed with an 8 byte identifier, which tells from whom the record was locked. Every time a CSA-record is updated by a GETUP-Request, the specific record in memory is locked:

- the field ZC#TTRID of the TCT is copied into the identification field of the CSA-record.

During a GETUP, the record is not read from disk, although CUPDATE=YES is specified. If a user issues a GETUP on a locked record, the status and detailed status-codes in the PIB are set to 3, respectively 18.

The locks are released after a successful PUT or at Action/Transaction Termination by clearing the identifier-field in the CSA to zeros.

2.4 Shutdown

2.4.1 General

When the system comes to Shutdown, all records of every CSA memory file with CUPDATE=NO parameter specified are written back to disk sequentially. This is done by searching the FCT's for the ZZ#FFLG2 X'80' value.

2.4.2 Failures

Because of a failure, two types of actions can occur:

- termination of processing for the current file
- termination of processing for all subsequent files

a) Termination for current file

- If an I/O-error occurs:

  W CSA UNLOAD I/O ERROR FOR FILE *_____*

- If a key in the CSA does not match with the key on disk:

  W CSA UNLOAD INVALID KEY FOR FILE *_____*

- If there are more records in memory than on disk: or vice versa

  W CSA UNLOAD INVALID # OF RECORDS FOR FILE *_____*

b) Termination for all subsequent files

- If a filename of a CSA-file in memory is inexistent in the FCT's

or

- If the unload-processing cannot detect an end of CSA-indicator:

W CSA UNLOAD DATA IN MEMORY DESTROYED

4. INTERFACES

As described in 2.1., CSA functions are used in 3 different modules. The following entry and exit conditions apply:

4.1 Entry Conditions

Information	Startup	On-Line	Term/Cancel	Shutdown
Address of THCB	ZQ#THCB	RB$	R8$	RB$
Address of TCT	--	RA$	RA$	--
Address of FCT	ZB#SFCTI/ZF#FCTA	R3$	--	--
Address of DTF	--	R4$	--	--
Function Code	--	RØ$	--	--
Param-List-Address	--	R1$	--	--
Address of PIB	--	RC$	--	--
Address first free byte in storage	ZT#HPIBA	--	--	--
Address last free byte in storage	ZB#MEHI	--	--	--
Address I/O area CSA file	Contents ZB#MEHI +1	--	--	--
Address of CSA start	--	ZF#CSADR	ZF#CSADR	ZF#CSADR

4.2 Exit Conditions

After Startup, the fields ZT#HPIBA and ZF#CSADR are adjusted.

The registers remain unchanged except RE$ and RF$.

The return-address is picked up from a table in ZF#CSAS, depending on the displacement value following the BALR instruction in the calling module.

Coding is calling Program

```
BALR RE$,RF$
DC   H'Ø'
BALR RE$,RF$
DC   H'4'
BALR RE$,RF$
DC   H'8'
```

Table in ZF#CSAS (Return-Address)

```
DC V(ZF#GTT9)     ON-LINE
DC V(ZF#GRTN)     TERMINATION
DC V(ZF#GTCN1)    CANCEL
```

4.3 External References

The following modules are activated during processing of ZF#CSAS:

ZF#ITR	Write before/after images to Trace
ZF#SATAU	Write before image to Audit
ZF#GSR (ZF#GEN)	Allocate I/O area from Storage Pool

4.3.1 Entry/Exit Conditions for ZF#ITR

R3$	Address of FCT user file	--	--	--
R4$	Address user DTF	--	--	--
RA$	Address TCT	--	--	--
RB$	Address THCB	--	--	--
RC$	Address PIB	--	--	--
DD$KLE	Must contain key length	--	--	--
DH$SPB	Must contain record size	--	--	--
DH$KLC	Must contain key location	--	--	--
ZT#HRPLA	Must contain parameter list address			
ZT#HFWA	Request-Packet (2 fullwords)	--	--	--
	Byte 1: 83 ISAM before image 8B ISAM after image	--	--	--
	Byte 2: ØØ Update			
	Byte 4-7: Address of key argument			

4.3.2 Entry/Exit Conditions for ZF#SATAU

R3$	Address of FCT of the User File	These registers can be restored from the GEN-S.A.
R4$	Address of the user DTF	
RA$	Address of the user TCT	
RB$	Address of the user THCB	
RC$	Address of the user PIB	

DH$KARG must contain Address of the key argument
ZF#MRN must contain record length of the user record
ZF#REC must contain record address of user record
ZT#HFWA must contain X'Ø3' to indicate ISAM request
ZT#HFWA+4 must contain key argument-address
DD$KLE must contain key length

4.3.3 Entry/Exit Condition for ZF#GSR

R3$: Address FCT

RB$ Address THCB

ZF#FIOS must contain the size of the requested I/O area.

After exit of ZF#GSC:

R6$ contains the address of the allocated I/O area

R7$ is destroyed

4.3.4 System Macros used for CSA Processing

SETL	Start sequential retrieval (Startup/Shutdown)
ESETL	End sequential retrieval (Startup/Shutdown)
GET	Read record sequential (Startup/Shutdown)
PUT	Update record sequential (Shutdown)
READ	Read record randomly (On-Line)
WRITE	Write record randomly (On-Line)
WAITF	Wait on I/O completion

5. DATA REFERENCES

This module does not need any new external data except a new added Flag-Byte in the File Control Table (ZF#FFLG2), which indicates whether a file is a CSA file or not.

ZF#FFLG2 - values:

BIT Ø	Ø = normal file 1 = CSA file
BIT 1	Ø = no disk update 1 = disk update performed

The format of the generated comma-storage-area in main storage is seen in Figure 2.5.

6. STORAGE MAPPING

The coding for the CSA processing is divided in three parts:

Startup-Coding. This coding is part of the ZB#STRTN module and for this only during Startup in memory.

On-Line-Coding. This coding is an own module (ZF#CSAS), which is part of the Root-Phase of IMS/90. The Start-Address of the CSA is stored in this module.

Shutdown-Coding. This coding is part of the ZB#SHUTN module, and for this only during shutdown in memory.

The Common-Storage-Area location is shown in Figure 2.3.

7. AVAILABILITY, RELIABILITY AND MAINTAINABILITY

7.1 Availability

As described in 2.2.8, a failure at the loading procedure does not force IMS to fail too. Processing can continue and it is the operator's decision to continue with that environment or not.

7.2 Reliability

Access to CSA data is done in the very same way as a normal I/O request. (GET/PUT level). The user does not notice the existence of the CSA facility in his action programs.

Just so, the same error conditions occur as with normal requests.

For error messages, refer to 2.2.8 and 2.4.2.

7.3 Maintainability

Very few modules are touched by the CSA option. Beside the Startup and Shutdown processing, all requests are handled in an own module, which is called from ZF#GEN only. Thus, corrections and/or enhancements have to be done in four modules. (ZB#STRTN, ZB#SHUTN, ZF#GEN, ZF#CSAS).

8. IMPLEMENTATION CONSIDERATIONS

8.1 Type of Code

Serially reusable.

8.2 Source Language

Series 90 Assembler.

8.3 Coding Constraints

None.

8.4 System Generation

Will be supplied.

8.5 Conventions and Standards

All labels in the module start with the prefix ZF#CS followed by a 3 digit number.

Register usage:

R2$	Cover
R3$	FCT
R4$	DTF
R5$-R9$	free
RA$	TCT
RB$	THCB
RC$	PIB
RD$	S.A.
RE$+RF$	branching register