Natural CICS Interface Functionality

This chapter describes the functionality of the Natural CICS interface.

It covers the following topics:

Natural CICS Interface

The Natural CICS Interface is implemented in command level Assembler, thus allowing Natural to be compatible with the CICS Multiple Region Option and the debugging facility CEDF.

The Natural CICS Interface controls session initialization, roll-in restart (in pseudo-conversational mode), terminal I/O, database access, ABEND processing, Natural local buffer pool calls and the loading, linking to and releasing of external subroutines. In addition, all roll I/O operations are made from the Natural CICS Interface.

Natural Nucleus under CICS

The Natural nucleus is a combination of the reentrant Natural module and various support routines, which are delivered as source programs requiring site-dependent assemblies and as load modules.

The CICS-related components of the Natural nucleus are:

- the Natural CICS interface module `NCISTART`

  This is the entry routine, which in particular prepares the Natural CICS Interface Language Environment (LE) linkage; see *Natural CICS Interface and IBM Language Environment (LE)*. The module is CICS version dependent.

- the Natural CICS interface module `NCIROOT`

  This module holds all CICS related logic as CICS services and CICS control block access. The module is CICS version dependent.

- the Natural CICS parameter module `NCIPARM`

  This module holds Natural CICS Interface runtime and system environment generation options. The module is not CICS version dependent, although some of the parameters should be set depending on the CICS version.

- the Natural CICS interface object-only part `NCINUC`

  This module holds service routines called by the Natural nucleus and Natural CICS Interface system control logic. These routines are independent of CICS and CICS version and are dealing with CICS by calling CICS service routines in `NCIROOT` and `NCISTART`.

- the Natural CICS interface module `NCIXCALL`

  This module is a seperate program in CICS, that is, it is not linked to the Natural nucleus, as it is invoked via `EXEC CICS LINK` from 3GL programs called by Natural; see *Natural 3GL CALLNAT Interface* in the *Operations* documentation. The module is CICS version dependent.

System Control under CICS

Natural features specific to CICS include the organization of dynamic storage in threads and the additional capability of handling these threads so that the Natural CICS System Control Program can more efficiently handle dynamic storage.

The Natural CICS System Control Program was initially developed to overcome the 64 KB GETMAIN limit under CICS. It provides complete storage allocation and management functions, including roll file I/O operations and relocation functions for pseudo-conversational users.

In order to enhance the pseudo-conversational processing capabilities of Natural with CICS, the System Control Program uses threads, a contiguous amount of storage which is set up for each user. This structure allows Natural to manage dynamic storage with minimal CICS involvement.

A complete understanding of system control can be attained from the following discussion of its structure and operation. Ensure that you understand this mechanism before starting the installation procedure of Natural under CICS.

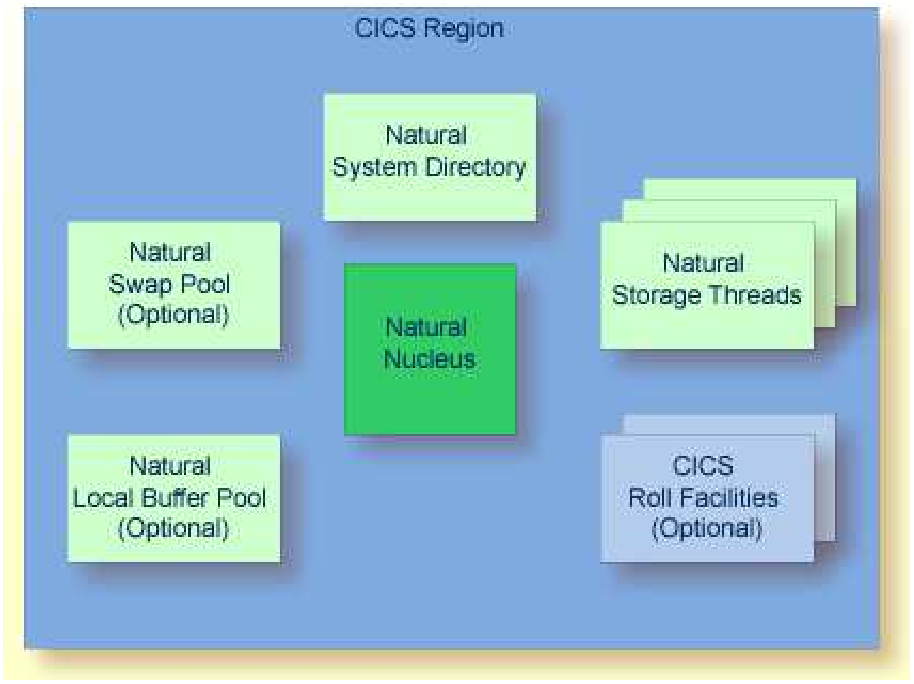
OSCOR/GETVIS - Natural Components in CICS Dynamic or Operating System Storage

Scenario 1:

Single CICS Region

The diagram below shows the components of the Natural system that reside in CICS dynamic storage. The components are explained under the following headings:

- *Natural Storage Threads under CICS*
- *Natural Local Buffer Pool under CICS*
- *Natural Swap Pool under CICS*
- *Natural Roll Facilities*

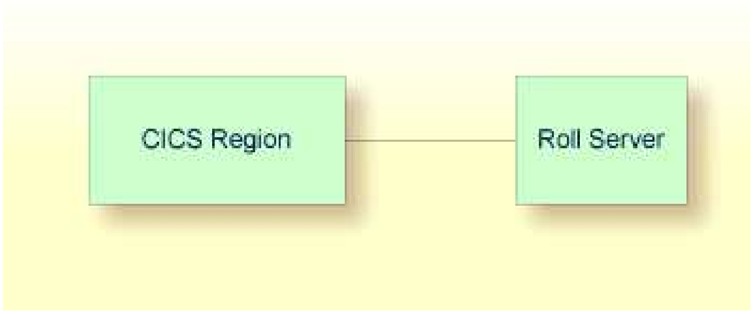

CICS Region
Natural System Directory
Natural Swap Pool (Optional)
Natural Nucleus
Natural Storage Threads
Natural Local Buffer Pool (Optional)
CICS Roll Facilities (Optional)

Scenario 1 applies when running Natural locally in a single CICS application region under z/OS or z/VSE.

Note:
Note Concerning z/OS Systems: Additional scenarios are possible. The following three diagrams show combinations of z/OS systems, CICS regions, the *Natural Roll Server* and the *Natural Authorized Services Manager*.

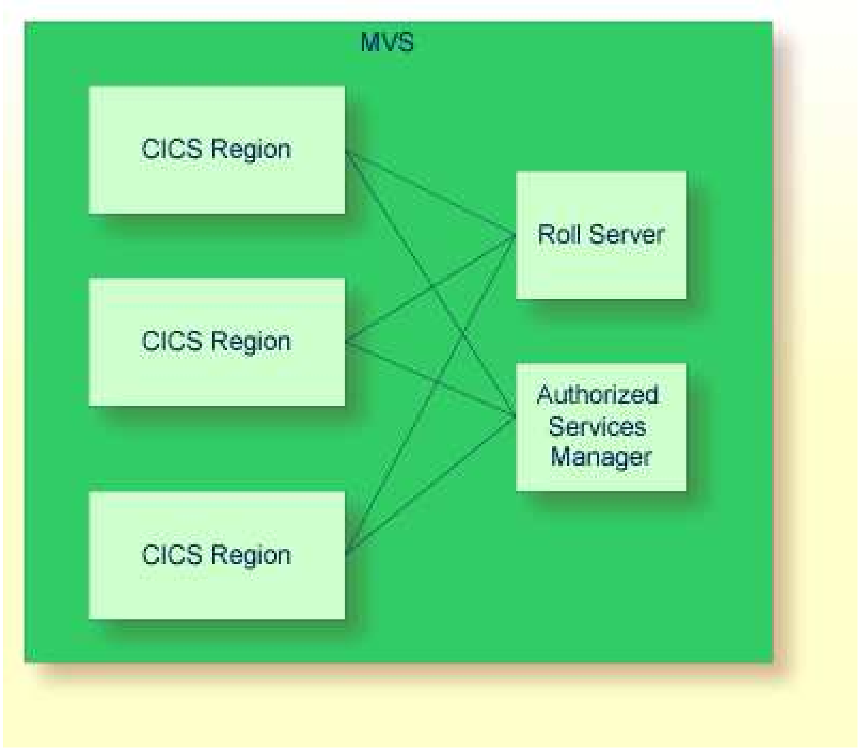
Scenario 2:

Single z/OS With Single CICS Region, Single Roll Server



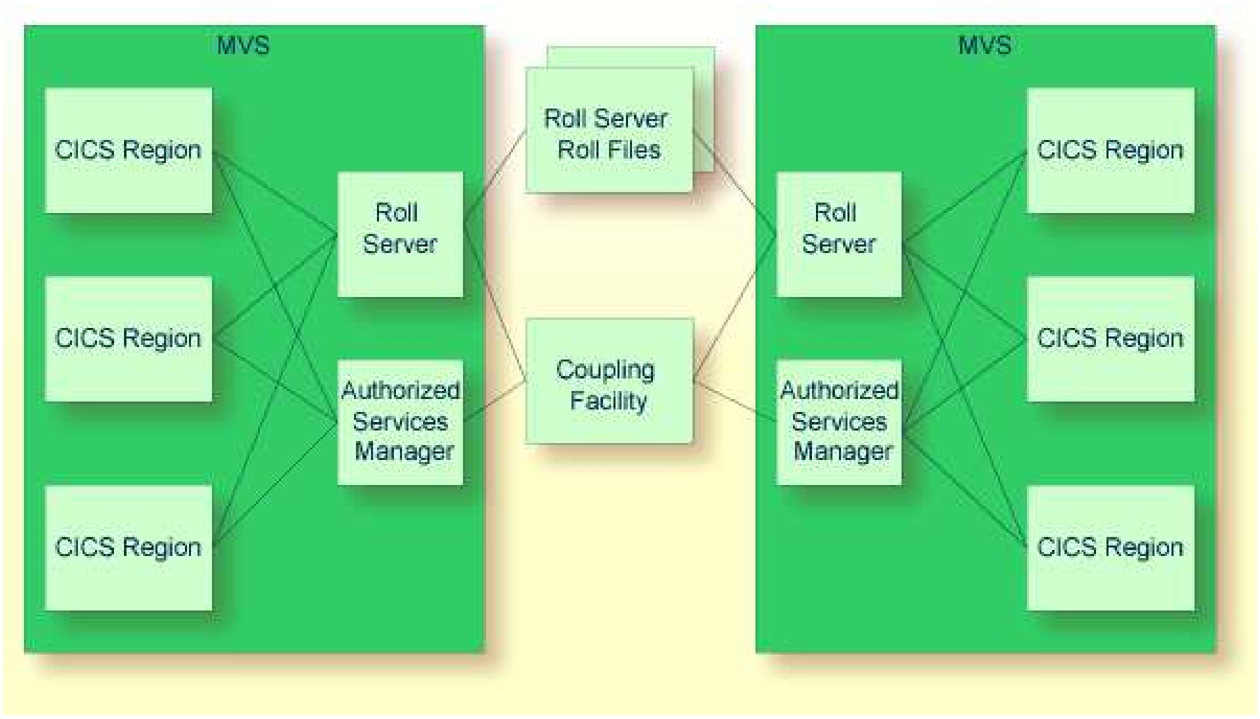
Scenario 3:

Single z/OS With Multiple CICS Regions, Single Roll Server and (Optional) Authorized Services Manager



Scenario 4:

Multiple z/OS With Multiple CICS Regions, Multiple Roll Servers/Authorized Services Managers



Parameter Settings Required for the Above Scenarios

Module	Scenario 1	Scenario 2	Scenario 3	Scenario 4
`NTBPI (BPI)`	`TYPE=SWAP,SIZE=`*`nnn`*	n/a	n/a	n/a
`NCMDIR CICSPLX`	`NO`	`NO`	`YES/MODE`	`YES/MODE`
`NCMDIR SIPSERV`	`NO`	`NO/YES`	`yes`	`yes`
`NCMDIR ROLLSRV`	`NO`	`yes`	`yes`	`yes`
Roll Server CF structure name	n/a	none	none	*`name`*
Authorized Services Manager/SIP	n/a	n/a	SIP slot number/size	XCF group name/CF structure name

The Natural CICS interface requires a SIP slot size of 256 bytes.

Note:
For the scenarios 2, 3 and 4, the very first Natural session initializing the NCI environment must have the `SUBSID` parameter set to the value of the corresponding *Roll Server* and/or *Authorized Services Manager*.

Natural Storage Threads under CICS

A thread is a contiguous storage area from where Natural requests all its required storage. It can either be storage shared by several Natural users or, in 31-bit mode environments, CICS user storage above the 16 MB line dedicated to a specific task.

Each storage thread can be seen as the "address space" for a Natural user. Each memory allocation request issued by the Natural nucleus is transferred to the system control program to be satisfied from the storage thread.

Storage threads are allocated when the Natural CICS interface is initialized. They are allocated in a CICS region or partition, in which case they are permanent (shared) threads or they are allocated during the start of a Natural CICS task, in which case they are exclusive threads (task-dependent user storage).

The technique of storage threads was implemented with Natural for the following reasons:

- To overcome the 64 KB limitation of CICS for user storage in non-31-bit mode systems.
- To be able to optimize rolling (formerly, each piece of user storage had to be written to the roll medium; now, as there is a contiguous storage area, this area is compressed by making the relevant portions contiguous to each other before rolling out).
- The Natural CICS interface tries to satisfy all `GETMAIN` requests of a Natural session from its thread. This is faster than `GETMAIN` requests by means of CICS service calls. This is particularly true for CICS command level calls, as the CICS `EXEC` Interface Program (`EIP`) is involved, too.

A thread is released by the owning task with every screen I/O. This is true for both conversational and pseudo-conversational tasks. When a session is resumed, its storage is rolled into a thread again, unless its storage is still there; that is, no other task used the thread in between.

The Natural thread selection algorithm balances thread usage to minimize roll I/Os. This means that the more threads there are, the better is the chance of finding the old data thus preventing a roll-in. However, the more threads there are, the more paging the operating system must perform to keep all threads efficiently in real storage.

Threads are grouped together depending on their size and their type; that is, whether they have been pre-allocated as permanently shared storage or via a `GETMAIN` request. The decision on which kind of thread group to use, is controlled by the CICS transaction code at session initialization time. All storage threads belonging to the same group have the same size.

The thread should be defined as small as possible; see also the *Buffer Usage Statistics* function of the Natural utility `SYSTP` in the Natural *Utilities* documentation. However, the thread must still be large enough to hold the session with the largest sizes.

If you have separate Natural development and production environments, the rule is to have more smaller threads in the production environment (to serve production requests as soon as possible) and fewer larger threads in the development environment (as Natural programmers normally need larger Natural sizes and have longer "think times").

The very first Natural session allocates all permanent (shared) threads.

Natural Roll Facilities under CICS

As permanent storage threads are shared by several users and as larger threads allocated via `GETMAIN` should not be kept for too much time, a Natural task releases its thread with each terminal I/O. Previously, however, the user data have to be saved to be able to restart the Natural session after the terminal I/O has been performed.

Session data can be saved by using

- the Natural Roll Server with its local roll buffer and roll files;
- the CICS Roll Facilities;
- the Natural swap pool.

See also the various component scenarios. For more information, see *Roll Server* in the Natural *Operations* documentation.

CICS Roll Facilities

CICS Roll Facilities are local CICS storage facilities. They can be either CICS main or auxiliary temporary storage or VSAM relative record datasets (RRDS) which the user has previously defined to CICS. These files allow Natural to store a user's compressed dynamic storage when a roll-out occurs.

When a swap pool is used, the CICS roll facilities only serve as backup for the swap pool. The choice of the roll medium is of greater importance when no swap pool is used, since it affects Natural performance and throughput.

Every CICS service request causes CICS system overhead. So, the larger the `CISIZE`/record size for the roll facility is, the less CPU overhead occurs due to fewer CICS service calls to roll a Natural session. On the other hand, larger `CISIZE`/record size also means more VSAM buffer space allocated for the roll facility.

See *Performance Considerations* for further information on roll facilities.

Warning:
When using the Roll Server, the swap pool and the CICS Roll Facilities are not available.

Natural Local Buffer Pool under CICS

The Natural local buffer pool contains all Natural modules during execution and copies of Natural modules once they have been loaded from the Adabas or VSAM system file.

The local buffer pool must be large enough to minimize the number of Natural program loads. However, if the local buffer pool is too large, this means wasted storage and may introduce paging overhead.

The local buffer pool is allocated as `GETMAIN` storage, that is, `EXEC CICS GETMAIN SHARED` with all CICS Transaction Server versions or a `GETVIS` request with CICS/VSE in z/VSE. Sufficient storage must be available in the partition or in the relevant CICS DSA.

A local buffer pool is optional, as Natural can also run with a global buffer pool, which can be shared with other Natural environments like *Natural in Batch Mode* (z/OS and z/VSE) or *Natural under TSO* or *Natural under IMS* (under z/OS only).

Natural Swap Pool under CICS

The Natural swap pool offers the possibility to "swap" a compressed Natural session from the thread into a main storage area instead of doing expensive roll I/Os.

The swap pool is allocated as `GETMAIN` storage, that is, `EXEC CICS GETMAIN SHARED` with all CICS Transaction Server versions or a `GETVIS` request with CICS/VSE in z/VSE. Sufficient storage must be available in the partition or in the relevant CICS DSA.

The options for the swap management are set in the Natural CICS source module `NCISCPCB` and by using the Natural profile parameter `BPI`.

The size, name and cache size of the swap pool are specified using profile parameter `BPI` or the corresponding macro `NTBPI` in the Natural parameter module `NATPARM`, that is, the `NTBPI` or `BPI` settings in effect for the Natural session initializing the NCI environment are taken.

For further details on the swap pool, see *Natural Swap Pool* in the Natural *Operations* documentation and *Using the Natural Swap Pool under CICS*.

Note concerning z/OS Systems

The swap pool can only be used when running Natural under CICS locally in a single CICS region. However, even in such a scenario, you should consider using the Roll Server instead, because it runs asynchronously to the CICS region and because it can provide more roll buffers in its data space than the swap pool. When using the Roll Server, the swap pool and the Roll Facilities are not available under CICS.

Natural CICS Interface System Control Records in CICS Temporary Storage

The Natural CICS Interface remembers its permanent `GETMAIN`ed storages, that is, storages acquired via `EXEC CICS GETMAIN SHARED` or operating system `GETMAIN` requests for `OSCOR`/`GETVIS` storage, in NCI system control records in CICS main temporary storage.

These system control records are kept for two reasons:

1. System recovery:

   As all NCI related storages are chained of the NCI system directory, the system control records can be used to re-construct storage chains in case of storage corruptions.

2. Clean up old NCI system after CICS `NEWCOPY` of NCI system directory module:

   At NCI system environment initialization, NCI checks for existing system control records, and, if found, NCI frees the associated permanent storages prior to the installation of the new environment.

The CICS temporary storage queue names of these control records are *prefixX*CR, where *prefix* is the common prefix for Natural CICS components (see `NCIPARM` generation parameter `PREFIX`) and *X* is a hexadecimal value, namely

x'01'	for the main system control record holding information about NCI system directory extension, shared threads (TYPE=SHR), and secondary SIR blocks (see NCMDIR generation parameter USERS).
x'02'	for the parms system control record holding information about the NCI shared profile parameters retrieved via file input (see NCIPARM generation parameter PRMDEST).
x'03'	for the pools system control record holding information about all local pools belonging toe the NCI environment including a potential swap pool.

Important:
As the NCI system control records describe a local NCI environment, these CICS MAIN temporary storage queues must be kept also in the CICS AOR. This is particularly true when running Natural in CICSPlex.

NCIDIREX - System Directory Module Name Exit Interface

The name of the Natural CICS interface system directory module is *prefix* CB by default (see PREFIX parameter of NCMPRM macro) unless specified explicitly via the DIRNAME parameter of the NCIPRM macro.

The NCIDIREX exit interface is to set/modify the name of the Natural CICS interface system directory module at run-time. This makes it possible to use the same NCI driver/ NCIPARM, but use different NCI environments (thread groups/thread sizes, etc) by accessing different system directory modules, depending for example on CICS system ID, transaction ID.

The first 5 characters of the directory module name are also used as part of CICS temporary storage queue names related to the relevant NCI environment. So when running more than one Natural CICS environment in a CICS region, the relevant system directory module names must be different in the first 5 characters.

The NCIDIREX interface exit is called using standard linkage conventions (Registers 13, 14, 15 and 1) but in addition with Registers 4 and 5 holding CICS EIB and EISTG addresses to enable the exit to call CICS services.

Source module XNCIDIRX contains a sample system directory module name exit.

NCITIDEX - Terminal ID Exit Interface

The 4-character CICS terminal ID which is unique per CICS region is used by the Natural CICS interface as part of the session key (SIP server, roll server, CICS temporary storage queues). For compatibility with Natural, the Natural CICS interface uses an 8-character field. This NCI terminal ID can be made unique over several CICS regions by appending the CICS system ID to the CICS terminal ID (see UNITID parameter of NCMPRM macro).

Alternatively, the NCITIDEX terminal ID exit interface can be used to set that NCI terminal ID. It should be noted that for CICS purposes (for example, temporary storage queue names, etc) just the first four characters of the NCI terminal ID are taken. Therefore these 4-character strings must be unique.

The NCITIDEX exit interface is particularly interesting for session managers under CICS in order to distinguish multiple Natural sessions running at the same physical terminal.

The terminal ID set by a `NCITIDEX` exit is used "externally" by the Natural CICS interface and is the default for the Natural system variable `*INIT-ID` for "internal" Natural use. (The `*INIT-ID` system variable can subsequently be modified by the `NCIUIDEX` / `NATUEX1` user ID exit interface.)

The `NCITIDEX` interface exit is called by using standard linkage conventions (Registers 13, 14, 15 and 1), but in addition by using the Registers 4 and 5 holding CICS EIB and EISTG addresses to enable the exit to call CICS services.

Source module `XNCITIDX` contains a sample terminal ID exit.

Restrictions

Certain Natural CICS interface functions cannot work if the first four characters of the logical terminal ID do not match the physical terminal.

As a consequence,

- you cannot send a message to a logical terminal by way of message switching,
- you cannot use the `SYSTP` utility or NEP to flush a session at a logical terminal.

NCIUIDEX - User ID Exit Interface

Natural provides the `NATUEX1` user exit interface to determine whether or not a user is authorized to use Natural and to set various Natural system variables.

Whenever a Natural user session is started, the `NATUEX1` interface exit is called using standard linkage conventions (Registers 13, 14, 15 and 1).

In a CICS environment, the standard linkage conventions are not sufficient in order to issue CICS service calls and to obtain addressability of CICS control blocks.

Therefore the Natural CICS interface delivers the load module `NCIUEX1` as a `NATUEX1` interface exit in a CICS environment. This module just sets up addressability in CICS and calls the `NCIUIDEX` interface exit by using standard linkage conventions (Registers 13, 14, 15 and 1), but in addition by passing CICS related addresses in other registers: R4 (`EIB`), R5 (`EISTG`), R6 (`TCTTE`).

Thus, if you want to issue requests requiring addressability of the CICS environment, the `NCIUIDEX` user ID exit interface should be used rather than the standard `NATUEX1` interface.

Source module `XNCIUIDX` contains a sample user ID exit.

Important:
With each installation of a new CICS release, the `NCIUIDEX` interface exit must be reassembled and linked.

Natural CICS Interface Debugging Facilities

The following topics are covered:

- Using the TPF Parameter
- Using Asynchronous Natural Sessions

Using the TPF Parameter

The dynamic parameter `TPF=(TPF1,TPF2,TPF3,TPF4,TPF5,TPF6,TPF7,TPF8)` can be set for driver-specific options by specifying "1" for the corresponding option.

Supported options are:

`TPF1`	Invoke Adabas linkage module via `EXEC CICS LINK` with Adabas parameter in TWA and CICS COMMAREA rather than via DCI. Enables debugging of Adabas-related problems via CEDF.
`TPF2`	Dump the whole Natural swap pool. With this parameter setting, the entire Natural swap pool is included in a CICS transaction dump.
`TPF3`	Dump the whole Natural buffer pool. With this parameter setting, the entire Natural buffer pool is included in a CICS transaction dump. **Note:** Usually the Natural buffer pool is not required in a dump, as all objects from the buffer pool relevant to a session are dumped anyway; so this option may only be required in the case of a buffer pool problem.
`TPF4`	Dump the whole EDITOR buffer pool. With this parameter setting, the EDITOR buffer pool is included in a CICS transaction dump.
`TPF6`	Handle terminal I/O errors by NCI. With this parameter setting, NCI will not pass control back to Natural for terminal I/O errors, but will handle it by itself, which results in one of the error messages `NT06` - `NT13`.
`TPF7`	Force abend in case of NCI system errors. With this parameter setting, a program check is forced in case of `NS`*`xx`*, `NI`*`xx`*, `NR`*`xx`* or `NUS`*`nnnn`* error messages. This is particularly helpful when a debugging tool intercepting abends is active. Then the error can be analyzed directly online.

When specifying `0` (which can also be omitted), the corresponding option is not set, for example:

`TPF=(0,0,0,1)` which is equivalent to `TPF=(,,,1)`

Using Asynchronous Natural Sessions

If the first 5 characters in the dynamic parameter string for starting Natural are `ASYN,`, the Natural CICS interface will always setup an asynchronous Natural session, regardless of whether the session is terminal-bound or not.

This may be helpful for testing purposes, particularly with EDF or with other debugging tools installed.

Natural CICS Interface CICS TWA Usage

The Natural transactions are all defined with a TWA size of 128 bytes, although the Natural CICS Interface just uses the first 88 bytes of the CICS transaction work area (TWA) for Natural processing of the following functions:

- on calling Adabas for the Adabas parameter list (up to 32 bytes), the Natural CICS Interface saves the TWA contents before calling Adabas and restores it after the Adabas call.
- on calling external programs for the parameter list address pointers (up to 20 bytes, see the Natural `CALL` statement), the Natural CICS Interface saves the TWA contents before calling the external program and restores the TWA call portion after the external program call.
- on invoking a back-end program for the termination message and potential termination data (80 bytes, see *Back-End Program Calling Conventions* in the Natural *Operations* documentation).
- on returning control to a "LINK" front-end caller for the termination message and potential termination data at session end and the termination message area fully reset to low-value at Natural dialog step end respectively, that is, 80 bytes at session and dialog step end.
- for passing LE information at CICS task start (up to 88 bytes, just at start of task).

User programs (front-end, back-end, called external programs) can also take advantage of the CICS TWA to communicate besides Natural, but they should not use the TWA portion used by Natural; for such cases, it is higly recommended to increase the TWA size of the Natural transactions and use TWA portions outside the first 128 bytes.