Note: Throughout this document, new support is highlighted in green, updates to existing documentation are highlighted in blue, and editorial corrections, notes, and existing support missing from the current documentation are highlighted in magenta.

1 CONTENTS

2 IXLCONN

Publications impacted: z/OS MVS Sysplex Services Guide (SA23-1400)
z/OS MVS Sysplex Services Reference (SA38-0658)

2.1 z/OS MVS Sysplex Services Guide (SA23-1400)

2.1.1 Connecting to a Cache Structure

Under *Sysplex Services for Data Sharing (XES) > Connection Services > Connecting to a Cache Structure,* add the following description of the ASYNCXI keyword after SUPPRESSEVENTS:

. . . .

The following IXLCONN parameters define the attributes of the cache structure or its connector:

. . . .

SUPPRESSEVENTS

. . . .

ASYNCXI
Specifies whether this connection prefers cross-invalidation operations against local caches to

be processed synchronously or asynchronously to the completion of cache operation requests initiated by this connection.

The use of the ASYNCXI keyword with the value of IxlConnAsyncXiYes (1) is meaningful only when a cache structure is allocated in a CFLEVEL=23 or higher coupling facility and the z/OS system where the IXLCONN is issued from supports asynchronous cross-invalidation.

To determine whether the support for asynchronous cross-invalidation processing is available on the system from which you are connecting to a cache structure, issue IXCQUERY REQINFO=FEATURES. QuReqRfAsyncXi, if returned, indicates whether the support is available. If the support is not available and you connect with ASYNCXI(IxlConnAsyncXiYes), the parameter will be ignored.

2.1.2 Using the IXLCONN macro for rebuilds

Under *Sysplex Services for Data Sharing (XES) > Connection Services > Structure Rebuild Processing > Connecting to the New Structure > Using the IXLCONN Macro for Rebuilds,*

Make the following change in this section:

The following keywords are IGNORED for rebuild connect requests (IXLCONN REBUILD or IXLCONN REQTYPE=REBUILDCONNECT):

- **ASYNCXI**
- CFLEVEL
- CONTEXIT
- NOTIFYEXIT
- LISTTRANEXIT.

2.1.3 Specifying Asynchronous or Synchronous Cross-Invalidate Processing

Under *Sysplex Services for Data Sharing (XES) > Connection Services > Connecting to a Coupling Facility Structure >* ***Connecting to a Cache Structure****,* add a new section titled ***"Specifying Asynchronous or Synchronous Cross-Invalidate Processing"*** after *"Suppressing Certain Events for a Connector"*

Specifying Asynchronous or Synchronous Cross-Invalidate Processing

A cache structure and its related services provide sysplex users with data consistency of their shared data. One such cache structure mechanism that is used to facilitate cache data consistency is called "cross-invalidation." Cross-invalidation processing involves setting an indicator in a local cache vector for each of the users of a cache structure data item to indicate whether the locally cached copy of the data is valid or not valid. Cross-invalidates are performed as part of cache requests and may be performed synchronously or asynchronously to the completion of a cache request.

Since users and /or coupling facilities may be remotely located, synchronous cross invalidation signal processing associated with transactions encompassing multiple cache requests may cause a delay in transaction completion. The connector may desire to have cross-invalidations performed asynchronously to a cache request and synchronize the completion of all cross-invalidates for multiple cache requests with a single transactional commit point. The preference for asynchronously completing cross-invalidations is best

suited for performance sensitive transactional processing made up of multiple cache requests that can benefit from reduced coupling facility overhead per cache request and support follow-on transaction commit processing (e.g. using the IXLAXISN service) to ensure the completion of cross-invalidations.

On systems with APAR OA54688 installed or which are at z/OS V2R4 or higher, using the ASYNCXI parameter on the IXLCONN invocation, a cache structure connector may specify their preference as to whether cross-invalidation operations associated with cache requests issued by the connector are to be performed synchronously or asynchronously to the completion of the cache request in the coupling facility. For more information on cross-invalidations and maintaining data consistency, see *"Maintaining Data Consistency."*

When requesting asynchronous cross-invalidations considerable thought must be given to the serialization protocols you implement. Please see section "Serialization considerations when requesting asynchronous cross-invalidations."

2.1.4 Cache Structure Allocation and Connection

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Cache Structure Allocation and Connection,* make the following change:

Characteristics that the user can specify on the IXLCONN macro for the connection to a cache structure include:

- Connection name
- Connection disposition
- Size of the local cache
- Cross-invalidation operation mode processing (synchronous or asynchronous) for local cache cross invalidations generated by commands issued by the connection

With the exception of the size of the local vector, connection characteristics remain fixed for the life of the connection (that is, as long as the user remains connected to the structure).

2.2 z/OS MVS Sysplex Services Reference (SA38-0658)

2.2.1 IXLCONN — Connect to a coupling facility structure

2.2.1.1 Understanding IXLCONN version support

Under *IXLCONN — Connect to a coupling facility structure > Understanding IXLCONN Version Support,* make the following changes:

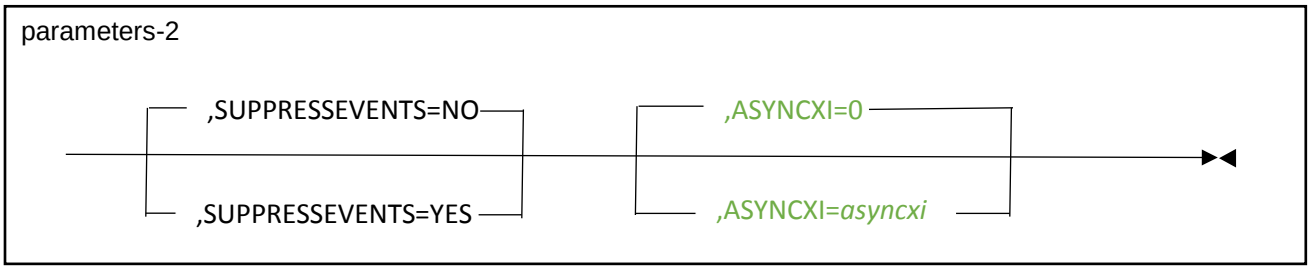
- The following keywords are supported by version 8 and subsequent versions of the IXLCONN macro: ENTRYIDTYPE, KEYTYPE, MINCFLEVEL

- The following keywords are supported by version 9 and subsequent versions of the IXLCONN macro: ASYNCDUPLEX, CRITICAL, FUNCTION, TERMLEVEL and **ASYNCXI.**

2.2.1.2 Syntax Diagram

Under *IXLCONN — Connect to a coupling facility structure* > *Syntax Diagram,* make the following changes:

For parameters-2, add **ASYNCXI** *after* SUPPRESSEVENTS



2.2.1.3 Parameter Descriptions

Under *IXLCONN — Connect to a coupling facility structure* > *Parameter Descriptions,* make the following changes:

Parameters common to all structure types

Under REBUILD

You can specify the following IXLCONN parameters, but the system ignores any changes made to the values in effect from the original invocation of IXLCONN for the structure:

ALLOWALTER
ALLOWAUTO
ALLOWDUPREBLD
ALLOWREBUILD
ASYNCXI
CFLEVEL

Under REQTYPE=REBUILDCONNECT

Update the following Note under REBUILDCONNECT:

The following keywords will be IGNORED for rebuild connect requests:

- **ASYNCXI**
- CFLEVEL
- CONTEXIT
- NOTIFYEXIT
- LISTTRANEXIT

Parameters for TYPE=CACHE

After SUPPRESSEVENTS, *add:*

,ASYNCXI=0
,ASYNCXI=*asyncxi*

Use this input parameter to indicate whether this connection prefers cross-invalidation operations against local caches to be processed synchronously or asynchronously to the completion of cache requests initiated by this connection.

The ASYNCXI keyword only applies for a cache structure. The ASYNCXI keyword is not allowed for structure types list or lock.

A value of 0 (IxlConnAsyncXiNo) specifies that cross-invalidation operations be processed synchronous to the completion of cache requests initiated by this connection.

A value of 1 (IxlConnAsyncXiYes) specifies that cross-invalidation operations be processed asynchronous to the completion of cache operation requests initiated by this connection when supported by the coupling facility where the cache structure is allocated.

An ASYNCXI value of 1 is meaningful only when a cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

Any value other than IxlConnAsyncXiNo (0) or IxlConnAsyncXiYes (1) will have the same behavior as specifying a value of 0 (IxlConnAsyncXiNo).

A connected user can invoke the IXLAXISN service to determine whether asynchronous cross-invalidations associated with its connection have completed.

3 IXLCACHE

Publications impacted: z/OS MVS Sysplex Services Guide (SA23-1400)
z/OS MVS Sysplex Services Reference (SA38-0658)

3.1 z/OS MVS Sysplex Services Guide (SA23-1400)

3.1.1 Information returned inline to IXCQUERY

Under *Sysplex Services for Communication (XCF) > Using the Cross-System Coupling Facility (XCF) > Obtaining XCF Information > Using the IXCQUERY Macro > Information returned inline to IXCQUERY,* make the following addition to existing documentation:

After QuReqRfWriteReadMetrics, add the following:

QuReqRfAsyncXI
Support for asynchronous cross-invalidation is supported on the system.

3.1.2 Handling Resources for a Disconnection

Under *Sysplex Services for Data Sharing (XES) > Connection Services > Disconnecting from a Coupling Facility Structure > Persistent considerations > Handling Resources for a Disconnection,* make the following update to existing documentation:

. . . .

Whether the disconnection is normal or the result of an error, MVS cleans up resources depending on the type of structure (cache, list, or lock) and whether the connection is being made failed-persistent or not.

- Cache structure
  - The local cache vector is released.
  - Cast-out locks held by the terminating connection are reset.
    - For castout locks held by the terminating connection in the read-for-castout state (that is, as the result of a CASTOUT_DATA request), the cast-out lock and cast-out lock state are reset to zero, the change bit for the entry is set to one to overindicate the “changed” state for the entry, and the parity is reset to the null value.
    - For cast-out locks held by the terminating connection in the write-with-castout state (that is, as the result of a WRITE_DATA CHANGED=NO GETCOLOCK=YES request), the entry is deleted from the cache structure along with all data and registered interest.
  - Registered interest in named directory entries is cleaned up.
  - All outstanding asynchronous cross-invalidations will be completed.
    - Reconnecting a failed-persistent connection will ensure structure detach processing has completed and therefore all outstanding asynchronous cross-invalidations will be complete.

3.1.3 Registering Interest in a Data Item

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Maintaining Data Consistency > Registering Interest in a Data Item and Validating Local Copies > Registering Interest in a Data Item,* make the following updates to existing documentation:

When you perform any of the following tasks, you cause the system to register or re-register your interest in a data item and update your local cache vector to indicate that your locally cached copy of the data item is valid:

. . . . .

- Specify a list of data items to register interest in (REQUEST=REG_NAMELIST)

When re-registering interest in a data item and the specified local cache vector entry replaces the previously specified local cache vector entry for the connection and data item, the old local cache vector entry associated with the local cache copy of the data item is invalidated.

3.1.4 Invalidating Local Cache Copies of a Data Item

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Maintaining Data Consistency > Deregistering Interest in a Data Item and Invalidating Local Copies > Invalidating Local Cache Copies of a Data Item,* make the following update to existing documentation:

. . . .

- A user deletes a data item from the cache structure (REQUEST=DELETE_NAME, REQUEST=DELETE_NAMELIST)

. . . .

- The system reclaims directory entry resources for the data item

3.1.5 Overriding the Asynchronous Cross-Invalidate Control Processing

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Maintaining Data Consistency > Deregistering Interest in a Data Item and Invalidating Local Copies*, add a section titled ***Overriding the Asynchronous Cross-Invalidate Processing*** after *Invalidating Local Cache Copies of a Data Item.*

When the system invalidates a connection's local copy of a data item, it sets the local cache vector entry associated with the local copy of the data item to not valid. The process of invalidating a local cache vector entry is referred to as *cross-invalidation*. When a cache structure is allocated in a coupling facility of CFLEVEL=23 or higher, you can specify your preference as to whether the cross-invalidation processing associated with a cache request that generates cross-invalidations is performed synchronously to the completion of cache request (i.e., the cache request does not complete until the cross-invalidation operations complete) or asynchronously to the completion of the cache request. A subsequent invocation of the IXLAXISN service can be used to confirm when the asynchronously initiated cross-invalidations associated with a cache request, which are identified by an asynchronous cross-invalidation sequence number returned by the IXLCACHE service, have completed.

A connector's asynchronous cross-invalidation control preference is specified on the IXLCONN invocation via the ASYNCXI parameter. When the value of the ASYNCXI parameter is IxlConnAsyncXiYes (1), the IXLCACHE AXIOVERRIDE parameter can be used to override the connector preference to cause the coupling facility to perform cross-invalidations synchronously as part of the cache request.

The preference for asynchronously completing cross-invalidations is best suited for performance sensitive transactional processing made up of multiple cache requests that can benefit from reduced coupling facility overhead per cache request yet support follow-on processing (e.g. using the IXLAXISN service) to ensure the completion of cross-invalidations. For cache requests that are less sensitive to performance and throughput and do not warrant the additional overhead of separately synchronizing with cross-invalidation completion, overriding the connector asynchronous cross-invalidation control preference on a per request basis is useful.

In summary, the AXIOVERRIDE keyword is intended for use to override the IXLCONN ASYNCXI=1 specification on a request by request basis. The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a coupling facility of CFLEVEL=23 or higher.

3.1.6 Serialization Considerations When Requesting Asynchronous Cross-Invalidations

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Serializing and Managing Access to Shared Data*, add a section titled ***Serialization considerations when requesting asynchronous cross-invalidations*** after *Updating Data in a Directory-only Cache.*

When a cache structure connector exploiting asynchronous cross-invalidates fails, it is important not to release serialization held by the connector to serialize the cache requests until it is known that all possible asynchronous cross-invalidations have completed. A cache structure exploiter can depend on structure detach processing to know for certain that any asynchronous cross-invalidations have completed for a failed connector. Structure detach processing is an internal XES process that detaches the connector from the cache structure in the coupling facility. Structure detach processing will outwait all asynchronous cross-invalidations that are in progress for that connection. Structure detach is an asynchronous process that is initiated when all peer connectors have acknowledged the disconnected or failed connection event for the failed connector. The only way to know for certain that the detach processing has completed is the ability for the same connector to reconnect to the cache structure. Considering that fact, a possible serialization scheme would call for connections to connect to the cache structure with CONDISP=KEEP allowing the connection to become failed-persistent after a failure. Holding serialization until the failed-persistent connection has reconnected ensures that all asynchronous cross-invalidations for the failed connection have completed.

3.1.7 Defining an Answer Area (ANSAREA)

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Receiving Information from a Request > Defining an Answer Area (ANSAREA),* make the following updates:

. . . .

If you provide an answer area, you must identify it on each IXLCACHE request through the ANSAREA keyword. You must also indicate the length of the answer area through the ANSLEN keyword.

The following are restrictions that apply when you specify an answer area:

- You must provide an answer area if you specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN.
- You must provide an answer area if you specify AXIOVERRIDE=0 (or default to 0) on an IXLCACHE request that can generate asynchronous cross-invalidations *and* specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure. The IXLCACHE requests that can generate asynchronous cross-invalidations are:
  - CROSS_INVAL
  - CROSS_INVALLIST
  - DELETE_NAME
  - DELETE_NAMELIST
  - READ_DATA
  - REG_NAMELIST
  - WRITE_DATA
  - WRITE_DATALIST
- Do not specify the same answer area for more than one request at the same time.

- Re-use or free answer area storage for a request only after you determine that the request is complete, either synchronously, or through the asynchronous specification for MODE on the request.

3.1.8 Specifying the IXLYCAA Level

Under *Sysplex Services for Data Sharing (XES) > Using Cache Services > Receiving Information from a Request > Defining an Answer Area (ANSAREA) > Specifying the IXLYCAA Level,* make the following updates:

The IXLCACHE Answer Area (IXLYCAA) supports several levels of information that IXLCACHE returns. Certain IXLCACHE requests might provide data that was not returned when the IXLCACHE service was first made available. For these request types, you must check the level of the IXLYCAA and ensure that the length of the answer area that you provide is capable of receiving all the data that the IXLCACHE request returns. For example:

- Extended restart tokens might be returned for restarting a request. An extended restart token requires that the level-1 version of IXLYCAA be used and that its length be specified as CAALEVEL1LEN or larger, or

- An asynchronous cross-invalidation sequence number may be returned when cross-invalidates are performed asynchronously to the completion of a request. An asynchronous cross-invalidation sequence number requires that a level-2 version of IXLYCAA be used and that its length be specified as CAALEVEL2LEN or larger.

The IXLYCAA macro provides the following mappings:

CAA

Data area that contains the macro level of the version of the IXLYCAA macro, the offset from the beginning of the mapping to the answer area data and the length of the answer area data. An answer area length of CAALEVEL1LEN (144 bytes) or greater is required when the version (PLISTVER) of the IXLCACHE macro is greater than three (3).

CAA2

Data area that contains the same content as the CAA mapping with additional information such as an asynchronous cross-invalidation sequence number (CaaAsyncXiSeqNum). An answer area length of CAALEVEL2LEN (256 bytes) or greater is required when using a CAA2 mapping.

You must use at least the required minimum answer area size for the request information that may be returned by the IXLCACHE request.

IBM recommends that you use at least the level-2 version of IXLYCAA to accommodate future enhancements to the IXLCACHE service. Note that as the level version of the IXLYCAA mapping increases, so does the corresponding CAALEVELxLEN value increase.

3.2 z/OS MVS Sysplex Services Reference (SA38-0658)

3.2.1 IXLCACHE – Cache Services

3.2.1.1 Understanding IXLCACHE version support

Under *IXLCACHE — Cache Services > Understanding IXLCACHE Version Support,* make the following changes:

The IXLCACHE macro supports versions in the range **0 – 8**

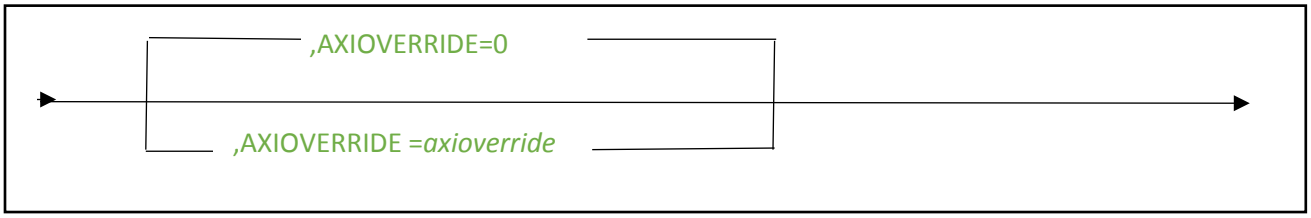
- The following keywords and functions are supported by all versions starting with version 8 and higher of the IXLCACHE macro:

HALTONCHANGE
SUPPCROSSINVAL
AXIOVERRIDE

3.2.2 IXLCACHE REQUEST=CROSS_INVAL

3.2.2.1 Syntax Diagram

Under *IXLCACHE REQUEST=CROSS_INVAL > Syntax Diagram*, add a line in the syntax diagram for AXIOVERRIDE after the line that starts with keyword NAME:



. . . .

NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.2.2 Parameter Descriptions

Under *IXLCACHE REQUEST=CROSS_INVAL > Parameter Descriptions*, make the following changes:

. . .

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*
Use this output parameter to specify an answer area to contain:

- A restart token (CAARESTOKEN) or extended restart token (CAAEXTRESTOKEN) that is returned from a request that exceeds the model-dependent time-out criteria. See the RESTOKEN and EXTRESTOKEN parameters for a description of the restart token

- An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) that is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

  See the IXLAXISN service for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

See the return and reason code descriptions ...

,**ANSLEN**=*anslen*
Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function:

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM)

- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN

- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN

,**AXIOVERRIDE**=0
,**AXIOVERRIDE**=*axioverride*
Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this

request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of the IXLAXISN service to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden.

3.2.2.3 Return and Reason Codes

Under *IXLCACHE REQUEST=CROSS_INVAL > Return and Reason Codes*, make the following changes:

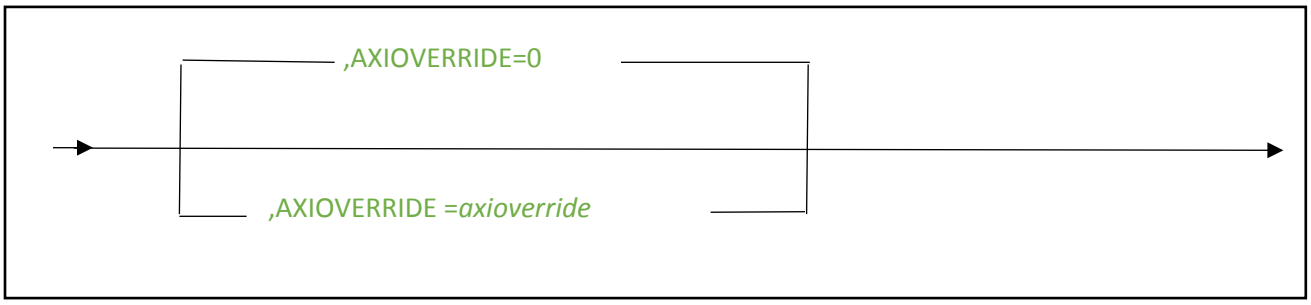
Hexadeci-mal Return Code	Hexadeci-mal Reason Code	Equate Symbol Meaning and Action	
8	Xxxx083D	Equate Symbol: IXLRSNCODEBA-DANSLEN . . . . Action: Increase the size of the answer area provided for the request and rerun your program. Depending on the macro version number or keywords specified on the macro invocation, specify the length of the answer area as follows: • When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of	

		CAALEVEL2LEN • When the value of the macro version number (PLISTVER) is 4 or above, the minimum answer area length is CAALEVEL1LEN • When the value of the macro version number (PLISTVER) is 0 - 3, the minimum answer area length is CAALEVEL0LEN	
8	xxxx08B9	EQUATE SYMBOL: IXLRSNCODENOANSAREA **Meaning:** An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request. **Action:** Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following: • CROSS_INVAL • CROSS_INVALLIST • DELETE_NAME • DELETE_NAMELIST • READ_DATA • REG_NAMELIST • WRITE_DATA • WRITE_DATALIST	

3.2.3 IXLCACHE REQUEST=CROSS_INVALLIST

3.2.3.1 Syntax Diagram

Under *IXLCACHE REQUEST=CROSS_INVALLIST > Syntax Diagram*, add a line in the syntax diagram for AXIOVERRIDE between ERRORACTION and CONTOKEN:



NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.3.2 Parameter Descriptions

Under *IXLCACHE REQUEST=CROSS_INVALLIST > Parameter Descriptions*, make the following changes:

. . .

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*
Use this output parameter to specify an answer area to contain information returned from the request if the request does not complete successfully or if the request initiated asynchronous cross-invalidations. See the return and reason codes descriptions for this request to determine which fields in the answer area are valid for non-zero return codes.

An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the IXLAXISN service for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . .

,**ANSLEN**=*anslen*
Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function:

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required

parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM)

- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN

,**AXIOVERRIDE**=0
,**AXIOVERRIDE**=*axioverride*

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

. . .

3.2.3.3 Return and Reason Codes

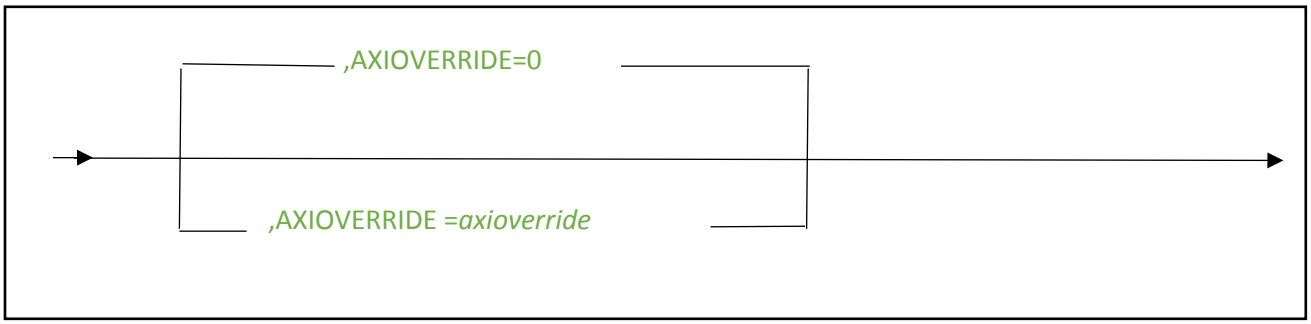
Under *IXLCACHE REQUEST=CROSS_INVALLIST > Return and Reason Codes*, make the following changes:

Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	Xxxx083D	Equate Symbol: IXLRSNCODEBADANSLEN . . . . Action: Increase the size of the answer area provided for the request and rerun your program. Depending on the macro version number or keywords specified on the macro invocation, specify the length of the answer area as follows: • When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN • When the value of the macro version number (PLISTVER) is 4 or above, the minimum answer area length is CAALEVEL1LEN • When the value of the macro version number (PLISTVER) is 0 - 3, the minimum answer area length is CAALEVEL0LEN
8	xxxx08B9	EQUATE SYMBOL: IXLRSNCODENOANSAREA **Meaning:** An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request. **Action:** Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following: • CROSS_INVAL • CROSS_INVALLIST • DELETE_NAME • DELETE_NAMELIST • READ_DATA • REG_NAMELIST • WRITE_DATA • WRITE_DATALIST

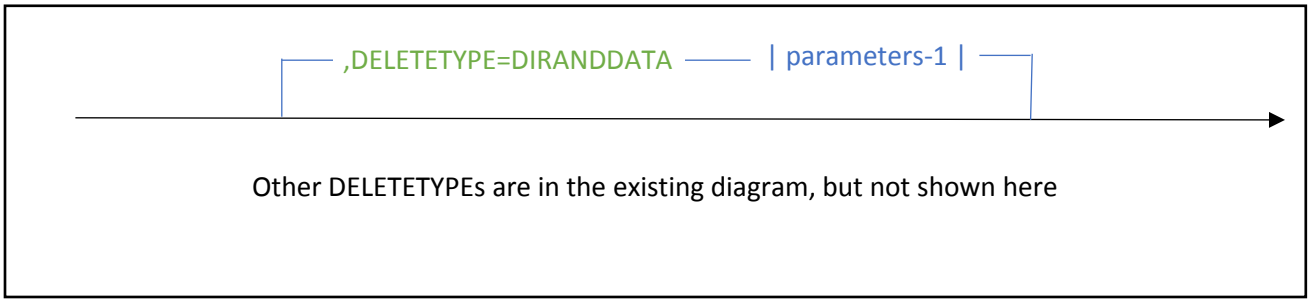
3.2.4 IXLCACHE REQUEST=DELETE_NAME

3.2.4.1 Syntax Diagram

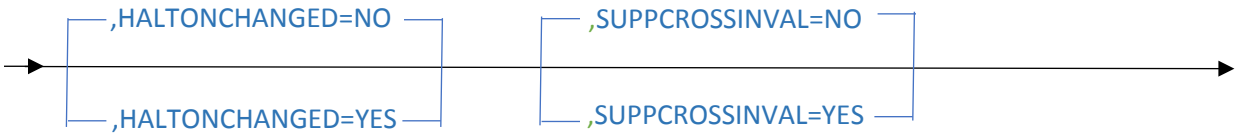
Under *IXLCACHE REQUEST=DELETE_NAME* > *Syntax Diagram*, add AXIOVERRIDE in the syntax diagram before the line that begins with VERSCOMP:



Missing from the existing documented syntax diagram are parameters HALTONCHANGED and SUPPCROSSINVAL which are applicable to DELETETYPE=DIRANDDATA:



Add parameters-1 to the syntax diagrams for DELETE_NAME



NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.4.2 Parameter Descriptions

Under *IXLCACHE REQUEST=DELETE_NAME* > *Parameter Descriptions*, make the following changes:

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*
Use this output parameter to specify an answer area to contain:

- A restart token (CAARESTOKEN) or extended restart token (CAAEXTRESTOKEN) that is returned from a request that exceeds the model-dependent time-out criteria. See the RESTOKEN and EXTRESTOKEN parameters for a description of the restart token
- An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) that is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

  See the IXLAXISN service for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . . . .

,**ANSLEN**=*anslen*
Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function:

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM)
- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN
- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN

,**AXIOVERRIDE**=0
,**AXIOVERRIDE**=*axioverride*
Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates against local caches for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates against local caches for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

3.2.4.3 Return and Reason Codes

Under *IXLCACHE REQUEST=DELETE_NAME > Return and Reason Codes*, make the following changes:

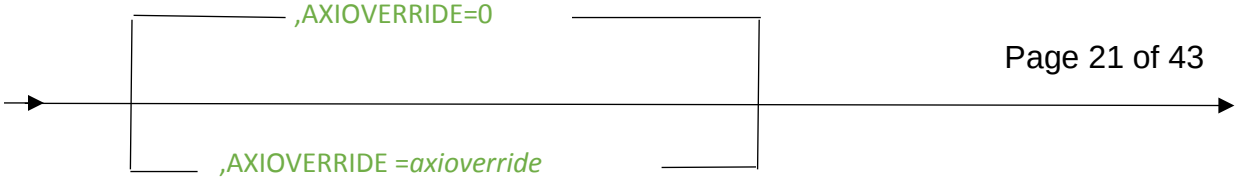
Hexadecimal Return Code	Hexadecimal Reason Code	Equate Symbol Meaning and Action
8	Xxxx083D	Equate Symbol: IXLRSNCODEBADANSLEN . . . . Action: Increase the size of the answer area provided

		for the request and rerun your program. Depending on the macro version number or keywords specified on the macro invocation, specify the length of the answer area as follows: • When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length must be a minimum value of CAALEVEL2LEN • When the value of the macro version number (PLISTVER) is 4 or above, the minimum answer area length is CAALEVEL1LEN • When the value of the macro version number (PLISTVER) is 0 - 3, the minimum answer area length is CAALEVEL0LEN
8	xxxx08B9	EQUATE SYMBOL: IXLRSNCODENOANSAREA **Meaning:** An answer area was not specified when one is required. The requested service determined that conditions exist that require an ANSAREA to complete the request. **Action:** Provide an answer area (ANSAREA) and answer area length (ANSLEN) on the IXLCACHE macro invocation for the request. ANSAREA is required when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure, AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request and the request is one of the following: • CROSS_INVAL • CROSS_INVALLIST • DELETE_NAME • DELETE_NAMELIST • READ_DATA • REG_NAMELIST • WRITE_DATA • WRITE_DATALIST

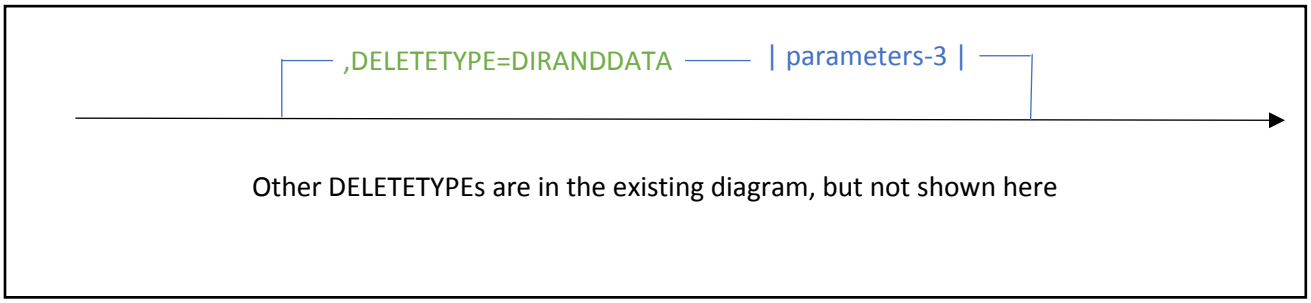
3.2.5 IXLCACHE REQUEST=DELETE_NAMELIST

3.2.5.1 Syntax Diagram

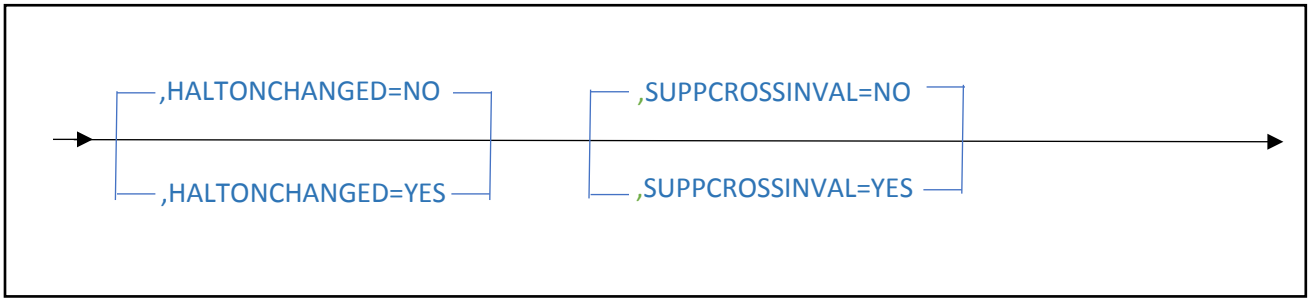
Under *IXLCACHE REQUEST=DELETE_NAMELIST* > *Syntax Diagram*, add AXIOVERRIDE in the syntax diagram between ERRORACTION and CONTOKEN:



Missing from the existing documented syntax diagram are parameters HALTONCHANGED and SUPPCROSSINVAL which are applicable to DELETETYPE=DIRANDDATA:



Add parameters-3 to the syntax diagrams for DELETE_NAMELIST



NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.5.2 Parameter Descriptions

Under *IXLCACHE REQUEST=DELETE_NAMELIST* > *Parameter Descriptions*, make the following changes:

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*

Use this output parameter to specify an answer area to contain information returned from the request. The format of the answer area is described by the IXLYCAA mapping macro.

An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the IXLAXISN service for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . . .

Descriptions of ANSLEN and AXIOVERRIDE are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

3.2.5.3 Return and Reason Codes

Return and reason code changes are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

3.2.6 IXLCACHE REQUEST=READ_DATA

3.2.6.1 Syntax Diagram

Under *IXLCACHE REQUEST=READ_DATA* > *Syntax Diagram*, add AXIOVERRIDE in the syntax diagram before the line that begins with keyword NAME:

. . . .

NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.6.2 Parameter Descriptions

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*

. . . .

The following additional field is returned in the answer area when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility:

- An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) if cross-invalidations were initiated asynchronously to the completion of the request.

  See IXLAXISN for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . . .

,**ANSLEN**=*anslen*

Use either CAALEVEL0LEN, CAALEVEL1LEN or CAALEVEL2LEN of the IXLYCAA mapping macro to determine the minimum size of the answer area. The answer area length must be at least large enough to accommodate the level of the IXLYCAA mapping appropriate to the requested function:

- When the connection specified ASYNCXI=1 on the IXLCONN invocation and AXIOVERRIDE=0 was specified or defaulted to for the IXLCACHE request, the answer area length is a required parameter and must be a minimum value of CAALEVEL2LEN to contain a returned asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM)

- When the value of PLISTVER is 4 or above, the minimum answer area length is CAALEVEL1LEN

- When the value of PLISTVER is 0 - 3, the minimum answer area length is CAALEVEL0LEN

,**AXIOVERRIDE**=0
,**AXIOVERRIDE**=*axioverride*

Use this input parameter to specify whether the asynchronous cross-invalidation control setting of IxlConnAsyncXiYes (1) for the connection identified by CONTOKEN should be overridden for this request. Valid values are 0 (IxlCacheAXiOverrideNo) or 1 (IxlCacheAXiOverrideYes).

The AXIOVERRIDE keyword is meaningful to processing only when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility.

A value of 0 (IxlCacheAXiOverrideNo) indicates that the asynchronous cross-invalidation control for the connection as specified on the IXLCONN invocation should be used for the request. Cross-invalidates for this request will preferentially be initiated asynchronously to the completion of the request when asynchronous cross-invalidations are supported by the coupling facility where the cache structure is allocated.

A value of 1 (IxlCacheAXiOverrideYes) indicates that the ASYNCXI specification of IxlConnAsyncXiYes (1) by the connector on the IXLCONN invocation should be overridden for this request only. Cross-invalidations generated by this request will be processed synchronously to the completion of the request.

Any value other than 0 or 1 for AXIOVERRIDE will have the same behavior as specifying a value of 0 (IxlCacheAXiOverrideNo).

If cross-invalidates for this request were initiated asynchronously to the completion of the request, an asynchronous cross-invalidation sequence number is returned in CAAASYNCXISEQNUM of the cache answer area (ANSAREA).

The asynchronous cross-invalidation sequence number can be used on a subsequent invocation of IXLAXISN to ensure that the asynchronous cross-invalidations associated with this request have completed.

When cross-invalidations are initiated synchronously to the completion of the request or no cross-invalidations occurred for the request, no asynchronous cross-invalidation sequence number is returned.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a one-byte input field that contains the value indicating whether the asynchronous cross-invalidation control setting of the connector should be overridden

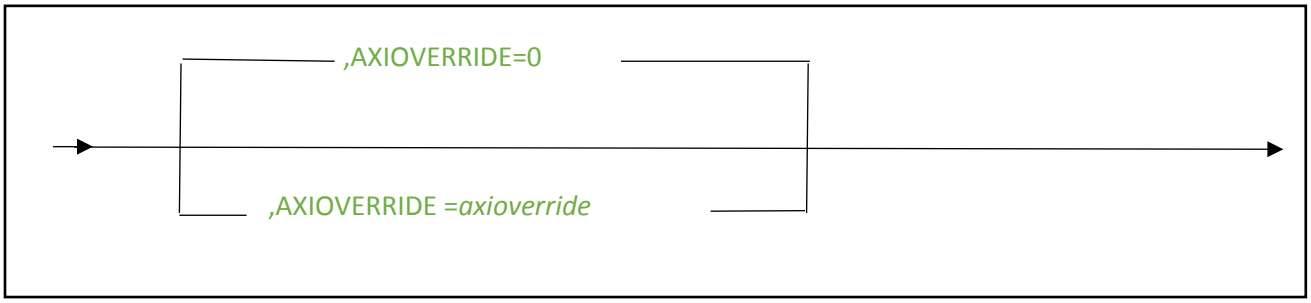
3.2.6.3 Return and Reason Codes

Return and reason code changes are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

3.2.7 IXLCACHE REQUEST=REG_NAMELIST

3.2.7.1 Syntax Diagram

Under *IXLCACHE REQUEST=REG_NAMELIST* > *Syntax Diagram*, add AXIOVERRIDE in the syntax diagram between NSBAREA and CONTOKEN:



NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure

3.2.7.2 Parameter Descriptions

Under *IXLCACHE REQUEST=REG_NAMELIST* > *Parameter Descriptions*, make the following changes:

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*

. . . .

An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) is returned from a request that initiates cross-invalidates of local caches asynchronously to the completion of the request.

See the IXLAXISN service for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . . .

Descriptions of ANSLEN and AXIOVERRIDE are the same as for IXLCACHE REQUEST=CROSS_INVAL-LIST.

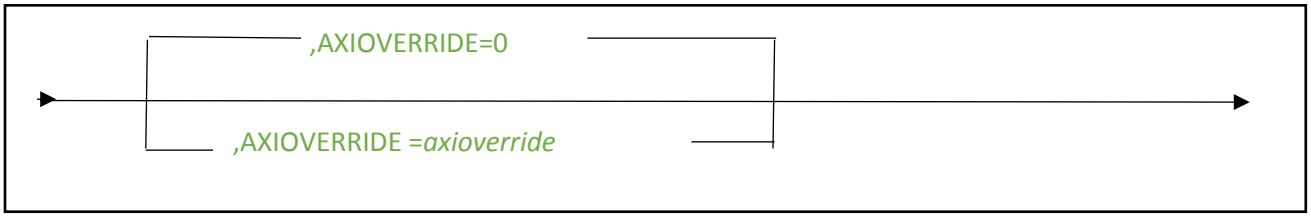
3.2.7.3 Return and Reason Codes

Return and reason code changes are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

3.2.8 IXLCACHE REQUEST=WRITE_DATA

3.2.8.1 Syntax Diagram

Under *IXLCACHE REQUEST=WRITE_DATA* > *Syntax Diagram*, add a line in the syntax diagram for AXIOVERRIDE before the line that starts with keyword NAME:



. . . .

NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.8.2 Parameter Descriptions

Under *IXLCACHE REQUEST=WRITE_DATA* > *Parameter Descriptions*, make the following changes:

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*

. . . .

The following additional field is returned in the answer area when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility:

- An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) if cross-invalidations against local caches for this request were initiated asynchronously to the completion of the request.

  See *IXLAXISN* for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

. . . .

Descriptions of ANSLEN and AXIOVERRIDE are the same as for IXLCACHE REQUEST=CROSS_INVAL-LIST.

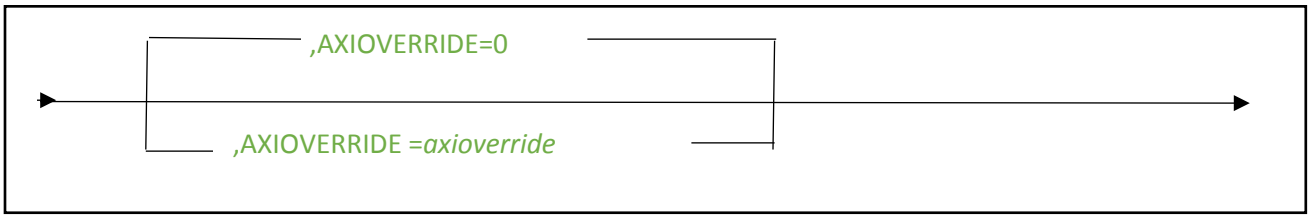
3.2.8.3 Return and Reason Codes

Return and reason code changes are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

3.2.9 IXLCACHE REQUEST=WRITE_DATALIST

3.2.9.1 Syntax Diagram

Under *IXLCACHE REQUEST=WRITE_DATALIST* > *Syntax Diagram*, add a line in the syntax diagram for AX-IOVERRIDE before the line that starts with keyword CONTOKEN:



. . . .

NOTE: You must specify ANSAREA=*ansarea*, ANSLEN=*anslen* if you:

- Specify MODE=SYNCTOKEN or MODE=ASYNCTOKEN, or
- Specify AXIOVERRIDE=0 (or default to 0) and specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure.

3.2.9.2 Parameter Descriptions

Under *IXLCACHE REQUEST=WRITE_DATALIST* > *Parameter Descriptions*, make the following changes:

,**ANSAREA**=NO_ANSAREA
,**ANSAREA**=*ansarea*

. . . .

The following additional field is returned in the answer area when the connection specified ASYNCXI=1 on the IXLCONN invocation when connecting to the cache structure and the cache structure is allocated in a CFLEVEL=23 or higher coupling facility:

- An asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) if cross-invalidations against local caches for this request were initiated asynchronously to the completion of the request.

  See IXLAXISN for a description of how to use the returned CAAASYNCXISEQNUM to determine when cross-invalidates of local caches associated with the request have completed.

Descriptions of ANSLEN and AXIOVERRIDE are the same as for IXLCACHE REQUEST=CROSS_INVAL-LIST.

3.2.9.3 Return and Reason Codes

Return and reason code changes are the same as for IXLCACHE REQUEST=CROSS_INVALLIST.

4 IXLADUPX

Publications impacted: z/OS MVS Sysplex Services Reference (SA38-0658)

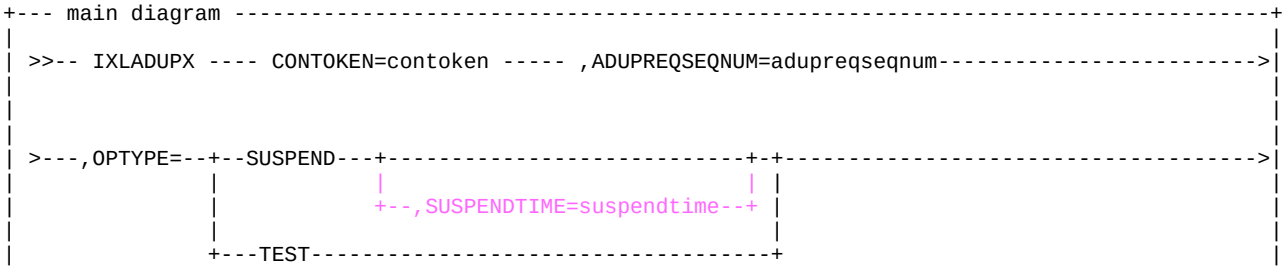
The following information is missing from the current documentation. Missing information is highlighted in magenta.

4.1 z/OS MVS Sysplex Services Reference (SA38-0658)

4.1.1 IXLADUPX

4.1.1.1 Syntax Diagram

Under *IXLADUPX – Synchronize an Asynchronously-Duplexed Structure* > *Syntax Diagram*, modify the Syntax Diagram for IXLADUPX as follows:



The remainder of the syntax diagram is unchanged.

4.1.1.2 Parameter Descriptions

Under *IXLADUPX – Synchronize an Asynchronously-Duplexed Structure > Parameter Descriptions,* add a description of the SUSPENDTIME keyword:

,SUSPENDTIME=suspendtime

Use this output parameter with OPTYPE=SUSPEND to specify a storage area to contain the amount of time, in microseconds, the invoker is suspended waiting for the request to be committed in the secondary structure. If the IXLADUPX request completes successfully without suspending the invoker, the content of this output variable will be zero.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of an 8-character field where the system will put the suspend time.

5 IXLAXISN

Publications impacted: z/OS MVS Sysplex Services Guide (SA23-1400)
z/OS MVS Sysplex Services Reference (SA38-0658)

5.1 z/OS MVS Sysplex Services Guide (SA23-1400)

5.1.1 Using the IXLAXISN macro

Under section “Supplementary List, Lock, and Cache Services”, add a chapter titled *“Using the IXLAXISN macro.”*

The IXLAXISN service provides the capability for a connected cache structure user to determine when asynchronous cross-invalidations associated with cache requests issued by the connector have completed. Asynchronous cross-invalidations for a cache request are identified by the asynchronous cross-invalidation sequence number (CAAASYNCXISEQNUM) returned in the IXLCACHE Answer Area (IXLYCAA) on IXLCACHE requests that generated asynchronous cross-invalidations. The IXLAXISN service supports the following operation types:

OPTYPE=TEST
This option allows you to test whether the cross-invalidations associated with the asynchronous cross-invalidation sequence number (ASYNCXISEQNUM) have completed. The return and reason codes indicate whether the cross-invalidation processing has completed in the coupling facility

This option may be used by invokers that cannot be suspended to poll for cross-invalidation completion status.

OPTYPE=SUSPEND

This option suspends the invoking unit of work until the cross-invalidations associated with ASYNCXISEQNUM have been completed by the coupling facility. Choose this option if the invoking unit of work can be suspended while cross-invalidations are being processed by the coupling facility.

Issuing the IXLAXISN service requires XES asynchronous cross-invalidation support. Use of IXLAXISN on a system that does not have XES asynchronous cross-invalidation support installed will have unpredictable results. Macro IXCYQUAA defines the QuReqRfAsyncXI bit in the QuReqFeatures string that can be used to test for asynchronous cross-invalidation support. Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string. Systems at a z/OS level of V2R2 and above with APAR OA54688 installed or which are at z/OS V2R4 and above support XES asynchronous cross-invalidation.

For more information, see sections *"Connecting to a Cache Structure"* and *"Deregistering Interest in a Data Item and Invalidating Local Copies*".

5.2 z/OS MVS Sysplex Services Reference (SA38-0658)

The following sections form a new chapter describing the IXLAXISN interface introduced by this support.

5.2.1 IXLAXISN

5.2.1.1 Environment

Minimum authorization:	Supervisor State or PKM allowing keys 0-7.
Dispatchable unit mode:	Task or SRB mode.
Cross Memory Mode:	Cross Memory Mode: Any PASN, any HASN, any SASN. The current primary address space must be the same as the primary address space at the time the connection service, IXLCONN, was issued for the structure
AMODE:	31- or 64-bit. If in 64-bit mode, specify SYSSTATE AMODE64=YES before invoking this macro.
ASC mode:	Primary or Access Register. If in Access Register ASC mode, specify SYSSTATE ASCENV=AR before invoking this macro.
Interrupt status:	Enabled or disabled for I/O and external interrupts. When OPTYPE=SUSPEND is specified the caller must be enabled.
Locks	Disabled callers must be legally disabled by holding the CPU lock and cannot hold other disabled locks (invocation by DIE routines

	is not supported). Enabled callers must not hold any locks.
Control parameters	Control parameters must be in the primary address space or, for AR-mode callers, must be in an address/data space that is addressable through a public entry on the caller's dispatchable unit access list (DU-AL). If the caller is running in Access Register ASC mode and specifies a macro parameter using explicit register notation the access register corresponding to the general register must appropriately qualify the general register.

5.2.1.2 Programming Requirements

1. Access Register ASC mode invokers must specify SYSSTATE ASCENV=AR prior to invoking this macro.

2. AMODE 64-bit invokers must specify SYSSTATE AMODE64=YES prior to invoking this macro.

5.2.1.3 Restrictions

- IXLAXISN requires XES asynchronous cross-invalidation support. Use of IXLAXISN on a system that does not have XES asynchronous cross-invalidation support installed will have unpredictable results. Macro IXCYQUAA defines the QuReqRfAsyncXI bit in the QuReqFeatures string that can be used to test for asynchronous cross-invalidation support.
Use IXCQUERY REQINFO=FEATURES to get the QuReqFeatures string

- If the caller is disabled then the parameter list and all storage areas addressed by macro parameters must reside in either fixed or disabled reference storage

- The caller may not be running as a Disabled Interrupt Exit (DIE).

- Callers running in SRB mode should refrain from specifying OPTYPE=SUSPEND under the following circumstances:

    - After the SRB receives a X'47B' abend
    - When running in a suspend exit after invoking SUSPEND

5.2.1.4 Input Register Information

Before issuing the IXLAXISN macro, the caller does not have to place any information into any register or AR unless using it in register notation for a particular parameter, or using it as a base register.

5.2.1.5 *Output Register Information*

When control returns to the caller, the GPRs contain:

REGISTER	CONTENTS
0	Reason
1	Used as work registers by the system
2-13	Unchanged
14	Used as work registers by the system
15	Return code

When control returns to the caller, the ARs contain:

REGISTER	CONTENTS
0-1	Used as work registers by the system
2-13	Unchanged
14-15	Used as work registers by the system

5.2.1.6 *Performance Implications:*

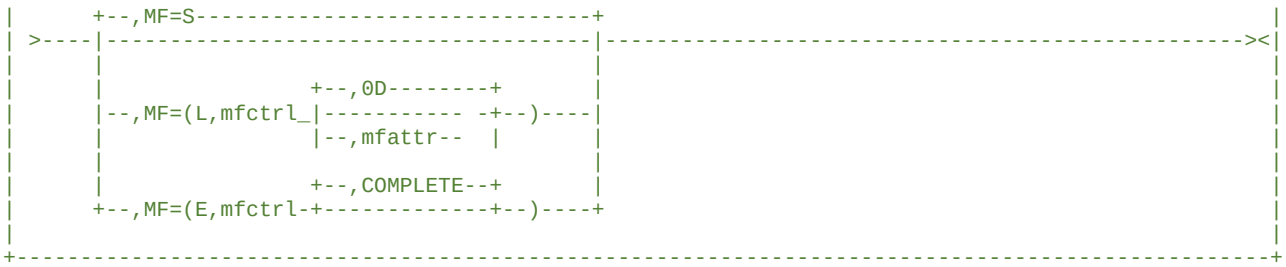
None.

5.2.1.7 *Understanding **IXLAXISN** version support*

The IXLAXISN macro supports version 0 keywords and functions.

Specify the version of the parameter list that you want generated with the PLISTVER keyword. See “Specifying a Macro Version Number” for considerations when specifying the version of the parameter list with PLISTVER.

5.2.1.8 *Syntax Diagram*



5.2.1.9 Parameter Descriptions

The parameter descriptions are listed in alphabetical order. Default values are underlined.

ASYNCXISEQNUM=asyncxiseqnum

Use this input parameter to specify an asynchronous cross-invalidation sequence number returned on a previous IXLCACHE request that generated asynchronous cross-invalidations of local caches. An asynchronous cross-invalidation sequence number is returned in the IXLCACHE Answer Area (IXLYCAA) in data field CAAASYNCXISEQNUM for IXLCACHE requests that generated asynchronous cross-invalidations.

An asynchronous cross-invalidation sequence number identifies cross-invalidations associated with a cache service request issued by the connector.

Cross-invalidations associated with assigned asynchronous cross-invalidation sequence numbers are completed by the coupling facility asynchronously to the completion of the cache request that caused the cross-invalidations of local caches to be generated.

Cross-invalidations with asynchronous cross-invalidation sequence numbers assigned are completed in the ascending order of these sequence numbers. If cross-invalidations identified by ASYNCXISEQNUM have completed, all cross-invalidations for the connector with a lower asynchronous cross-invalidation sequence number are also completed for the connector.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the asynchronous cross-invalidation sequence number.

,CONTOKEN=contoken

Use this input parameter to specify a connect token returned by the IXLCONN service. The CONTOKEN uniquely identifies the user's connection to a cache structure.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a 16-character field that contains the connect token.

,MF=S
,MF=(L,*mfctrl*)
,MF=(L,*mfctrl*,*mfattr*)
,MF=(L,*mfctrl*,*0D*)
,MF=(E,*mfctrl*)
,MF=(E,*mfctrl*,*COMPLETE*)

Use MF=S to specify the standard form of the macro, which builds an in-line parameter list and generates the macro invocation to transfer control to the service.

Use MF=L to specify the list form of the macro. Use the list form together with the execute form of the macro for applications that require reentrant code. The list form defines an area of storage that the execute form uses to store the parameters. Only the PLISTVER parameter can be coded with the list form of the macro.

Use MF=E to specify the execute form of the macro. Use the execute form together with the list form of the macro for applications that require reentrant code. The execute form stores the parameters into the storage area defined by the list form and generates the macro invocation to transfer control to the service.

,mfctrl

Use this output parameter to specify a storage area to contain the parameters.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of the parameter list.

,mfattr

Use this input parameter to specify the name of a 1- to 60-character string that can contain any value that is valid on an assembler DS pseudo-op. You can use this parameter to force boundary alignment of the parameter list. If you do not code *mfattr*, the system provides a value of 0D, which forces the parameter list to a doubleword boundary.

,COMPLETE

Use this input parameter to require that the system check for required parameters and supply defaults for omitted optional parameters.

Note: In the macro expansion you might see some defaults for optional parameters that are not documented here. The ones that are not documented do not have any effect on the macro. For example, if SMILE=*var* were an optional parameter and the default is SMILE=NO_SMILE then it would not be documented. However, if the default was SMILE=:-), then it would be documented because a value would be the default.

,OPTYPE=SUSPEND
,OPTYPE=TEST

Use OPTYPE=SUSPEND to indicate that control should be returned to the invoker when the cross-invalidations associated with ASYNCXISEQNUM have been

completed by the coupling facility. The invoker must be executing in an enabled state to use this option.

Use OPTYPE=TEST to test whether the cross-invalidations associated with ASYNCXISEQNUM have completed. The return and reason codes indicate whether the cross-invalidation processing has completed in the coupling facility.

This option may be used by invokers that cannot be suspended to poll for cross-invalidation completion status.

,PLISTVER=IMPLIED_VERSION
,PLISTVER=MAX
,PLISTVER=plistver

Use this input parameter to specify the version of the macro. See "Understanding IXLAXISN version support" for a description of the options available with PLISTVER.

,RETCODE=retcode

Use this output parameter to specify the location in which the system is to copy the return code from GPR 15.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the return code.

,RETCODE=retcode

Use this output parameter to specify the location in which the system is to copy the reason code from GPR 0.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a fullword to contain the reason code.

,SUSPENDTIME=suspendtime

Use this output parameter with OPTYPE=SUSPEND to specify a storage area to contain the amount of time, in microseconds, the invoker is suspended waiting for the cross-invalidations to complete. If the IXLAXISN request completes successfully without suspending the invoker, the content of this output variable will be zero.

To Code: Specify the RS-type name or address (using a register from 2 to 12) of a 8-character field where the system will put the suspend time.

5.2.1.10 Return and Reason Codes

Return and Reason Codes:

When the IXLAXISN macro returns control to your program:

- GPR 15 (and *retcode*, if you coded RETCODE) contains a return code.
- GPR 0 (and *rsncode*, if you coded RSNCODE) contains a reason code.

Macro IXLYCON provides equate symbols for the return and reason codes. The equate symbols associated with each hexadecimal return code are as follows:

0
IXLRETCODEOK
4
IXLRETCODEWARNING
8
IXLRETCODEPARMERROR
C
IXLRETCODEENVERROR
10
IXLRETCODECOMPERROR

The following table identifies the hexadecimal return and reason codes and the equate symbol associated with each reason code.

Hexadecimal return code	Hexadecimal reason code	Equate symbol meaning and action
0	xxxx0000	**Equate Symbol:** IxlRsnCodeNoCfAccessRequired **Meaning**: Successful completion. The cross-invalidations of local caches associated with ASYNCXISEQNUM have completed. Cross-invalidation completion information was obtained from local XES connector information for cross-invalidations associated with ASYNCXISEQNUM. **Action:** None required.
0	xxxx0001	**Equate Symbol:** IXLRSNCODECFACCESSREQUIRED **Meaning**: Successful completion. The cross-invalidations of local caches associated with ASYNCXISEQNUM have completed. Cross-invalidation completion information was obtained from the coupling facility for cross-invalidations associated with ASYNCXISEQNUM. **Action:** None required.
4	xxxx043A	**Equate Symbol**: IXLRSNCODECROSSINVALSOUTSTANDING

		Meaning: An OPTYPE=TEST operation found that the cross-invalidations associated with ASYNCXISEQNUM are not complete. **Action:** Issue the IXLAXISN macro again with OPTYPE=TEST to determine whether the cross-invalidations associated with ASYNCXISEQNUM are complete. Alternatively, issue the IXLAXISN macro with OPTYPE=SUSPEND in an enabled environment to wait for the cross-invalidations associated with ASYNCXISEQNUM to complete.
4	xxx043C	**Equate Symbol**: IxlRsnCodeBadSeqNumInstance **Meaning:** There are no outstanding asynchronous cross-invalidations associated with the specified ASYNCXISEQNUM for the connection because ASYNCXISEQNUM is not associated with the current instance of the structure. **Action:** None required. However, you might want to ensure that the asynchronous cross-invalidation sequence number is correct.
8	xxxx0801	**Equate Symbol:** IXLRSNCODEBADPARMLIST **Meaning:** Program error. The parameter list for this request is not addressable. **Action:** Verify that: • The parameter list address is uncorrupted. • The parameter list is addressable in the caller's primary address space. • If you are issuing this macro while disabled, the parameter list resides in either page-fixed or DREF storage. • If you are issuing this macro in AR-mode and you specified the parameter list address using explicit register notation, the corresponding access register was updated appropriately. • If you are issuing this macro in AR-mode, SYSSTATE ASCENV=AR must be issued before issuing this macro.
8	xxxx0802	**Equate Symbol:** IXLRSNCODEBADPARMLISTALET **Meaning:** Program error. The parameter list ALET is not valid. **Action:** Ensure that the ALET is zero or that the ALET represents a valid entry on the DU-AL.
8	xxxx0804	**Equate Symbol:** IXLRSNCODEBADVERSION# **Meaning:** Program error. Invalid version number in the parameter list. **Action:** • Verify that your program did not overlay the parameter list storage.

		• Verify that your program was assembled with the correct macro library for the release of MVS your program is running on.
8	xxxx080A	**Equate Symbol:** IXLRSNCODEBADCONTOKEN **Meaning:** Program error. The input contoken is not valid. The contoken may no longer be valid for one of the following reasons: disconnect has occurred, EOT of the connector's task, input contoken is not the contoken returned from IXLCONN, the request was issued outside the connector's address space, or the contoken has been invalidated for rebuild. **Action:** Verify that the CONTOKEN value specified is valid and for the correct structure.
8	xxxx0824	Equate Symbol: IXLRSNCODEWRONGSTRTYPE Meaning: Program error. The connection specified by CONTOKEN is not to a cache structure. Action: Specify the appropriate CONTOKEN value, and rerun your program. The CONTOKEN is returned in the answer area provided on the IXLCONN invocation. The type of structure built is passed on the TYPE parameter of the IXLCONN macro.
8	xxxx0851	**Equate Symbol:** IXLRSNCODENOSUSPENDISABLE **Meaning:** Program error. The request failed because OPTYPE=SUSPEND was specified, but the caller is disabled and cannot be suspended. **Action:** Either specify another OPTYPE value, or become enabled (release the CPU lock), then reissue the request.
8	xxxx08A2	**Equate Symbol**: IxlRsnCodeBadAsyncXiSeqNum **Meaning:** Program error. An invalid asynchronous cross-invalidation sequence number was specified. The value specified in ASYNCXISEQNUM does not represent a valid asynchronous cross-invalidation sequence number returned on a previous IXLCACHE invocation. **Action:** Ensure that you use only an asynchronous cross-invalidation sequence number that is returned on a previous IXLCACHE invocation issued against the current logical version of the cache structure.
8	xxxx08A3	**Equate Symbol**: IXLRSNCODENOASYNCXICONN **Meaning:** Program error**.** The connector did not specify ASYNCXI=IxlConnAsyncXiYes on the IXLCONN invocation when connecting to the cache structure. **Action: E**nsure that the input CONTOKEN identifies a connector that specified ASYNCXI=IxlConnAsyncXiYes on an IXLCONN invocation.
8	xxxx08B6	**Equate Symbol:** IXLRSNCODEBADSUSPENDENV

		Meaning: Program error. The IXLAXISN OPTYPE=SUSPEND operation was issued from a SUSPEND exit routine or from an SRB routine that the system abended with a 47B system completion code. The caller cannot be suspended while running in this environment. **Action:** Either specify another OPTYPE value, or reissue the IXLAXISN macro from an environment that allows the caller to be suspended.
8	xxxx08B7	**Equate Symbol:** IXLRSNCODEBADLOCKS **Meaning:** Program error. The caller's environment does not match the serialization and interrupt status requirements of the IXLAXISN service routine. For example: • Caller is enabled but holds locks. • Caller is disabled but does not hold the CPU lock. • Caller is disabled but holds disabled locks in addition to the CPU lock. **Action:** Reissue the IXLAXISN macro in an environment that matches the requirements of the service macro.
C	xxxx0C06	**Equate Symbol:** IXLRSNCODENOCONN **Meaning:** Environmental error. This system does not have connectivity to the coupling facility that contains the structure. Possible reasons for this are: • The operator issued VARY PATH,OFFLINE. • The operator issued CONFIG CHP,OFFLINE. • Hardware errors to the coupling facility. • Facility or path failure to the coupling facility. **Action:** Begin rebuilding the structure on a different coupling facility, or disconnect from the structure.
C	xxxx0C25	**Equate Symbol:** IXLRSNCODESTRFAILURE **Meaning:** Environmental error. The structure failed prior to completion of the request. **Action:** Either rebuild or disconnect from the structure.
10	xxxx10xx	**Meaning:** XES processing failure. The state of the involved structure and the disposition of the request are unpredictable. **Action:** Contact the IBM support center.

6 IXLYCAA MAPPING MACRO

Publications Impacted: z/OS MVS Data Areas Volume 3 (GA32-0937)

Offsets		Type / Value	Len	Name (DIM)	Description
0	(0)	STRUCTURE	0	CAA	IXLCACHE Answer Area
. . .					
144	(90)	X'90'	0	CAA_LEN	"*-CAA"
0	(0)	STRUCTURE	0	CAA2	IXLCACHE Level 2 Answer Area
0	(0)	CHARACTER	144		Mapped by CAA
144	(90)	STRUCTURE	112	CAA2DATA (0)	Level 2 Answer Area Data
144	(90)	CHARACTER	16	CAAASYNCXISEQNUM	Asynchronous cross-invalidation sequence number. Returned for READ_DATA, WRITE_DATA, CROSS_INVAL, CROSS_INVALLIST, DELETE_NAME, DELETE_NAMELIST, REG_NAMELIST and WRITE_DATALIST requests which initiate cross-invalidates that are executed asynchronously to the command. Valid for connectors that specify ASYNCXI=1 on their IXLCONN invocation.
160	(A0)	CHARACTER	96		Reserved
		X'2'	0	CAALEVEL#	Latest macro level number
		X'0'	0	CAALEVEL0	Macro level number
		X'1'	0	CAALEVEL1	Macro level number
		X'2'	0	CAALEVEL2	Macro level number
		X'60'	0	CAALEVEL0LEN	Length of CAA level 0
		X'90'	0	CAALEVEL1LEN	Length of CAA level 1
		X'100'	0	CAALEVEL2LEN	Length of CAA level 2
		X'100'	0	CAA2_LEN	"*-CAA2"

7 IXLYCON MAPPING MACRO

Publication Impacted: z/OS MVS Data Areas Volume 3 (GA32-0937)

```
OFFSET    OFFSET
  DEC      HEX   TYPE     LENGTH  NAME (DIM)          DESCRIPTION
-------------------------------------------------------------------------------------------------------------------

--------- Constants for use with IXLCONN service -----------

           IXLCONN MonitorStorage constants. Refer to IXLCONN
           for detailed usage description.

  0         (0)      X'0'        0    IXLCONNMONITORSTORAGENO   "0"
  0         (0)      X'1'        0    IXLCONNMONITORSTORAGEYES  "1"

           IXLCONN AsyncXI constants. Refer to IXLCONN
           for detailed usage description.

  0         (0)      X'0'        0    IXLCONNASNCXINO           "0"
```

```
  0         (0)      X'1'        0    IXLCONNASYNCXIYES         "1"


           IXLCONN Monitor constants. Refer to IXLCONN
           for detailed usage description.

  0         (0)     X'0'         0    IXLMONITORDEFAULT         "0"
  0         (0)     X'1'         0    IXLMONITORCFREQRATE       "1"
  0         (0)     X'2'         0    IXLMONITORIXLREBLD        "2"


--------- Constants for use with IXLCACHE service -----------

  0         (0)     X'0'         0    IXLCACHEAXIOVERRIDENO  "0"
  0         (0)     X'0'         0    IXLCACHEAXIOVERRIDEYES "1"

.  .  .  .

-----------------------------------------------------------------
 Reason Codes -- IxlRetCodeOk
 (Note that the reason codes are of the form "xxxxYYYY" where
  "xxxx" is used to contain internal diagnostic information)
-----------------------------------------------------------------

  0         (0)    BITSTRING     0    IXLRSNCODENOCFACCESSREQUIRED
                                        "X'00000000'" The cross-invalidations
                                        of local caches associated with the
                                        AsyncXiSeqNum have completed. Cross-
                                        invalidation completion information was
                                        obtained from local XES connector information
  0         (0)    BITSTRING     0    IXLRSNCODECFACCESSREQUIRED
                                        "X'00000001' The cross-invalidations of
                                         local caches associated with the
                                         AsyncXiSeqNum have completed. Cross-
                                         invalidation completion information was
                                         obtained from the CF
    .  .  .  .

-----------------------------------------------------------------
    Reason Codes -- IxlRetCodeWarning
    (Note that the reason codes are of the form "xxxxYYYY" where
     "xxxx" is used to contain internal diagnostic information)
 ----------------------------------------------------------------

.  .  .  .

  0         (0)    BITSTRING     0     IXLRSNCODECROSSINVALSOUTSTANDING
                                        "X'0000043A'" A IXLAXISN OPTYPE=TEST operation
                                        Found that the cross-invalidations associated
                                        With ASYNCXISEQNUM are not complete.
                                        Cross-invalidation status information was
                                        obtained from the coupling facility for cross-
                                        invalidations associated with ASYNCXISEQNUM.
  0         (0)    BITSTRING     0     IXLRSNCODEBADSEQNUMINSTANCE
                                        "X'0000043C'" An IXLAXISN service request found
                                        no outstanding asynchronous cross-invalidations
                                        associated with the specified ASYNCXISEQNUM for
                                        the connection because ASYNCXISEQNUM is not
                                        associated with the current instance of the
                                        structure.
.  .  .  .


-----------------------------------------------------------------
 Reason Codes -- IxlRetCodeParmError
 (Note that the reason codes are of the form "xxxxYYYY" where
  "xxxx" is used to contain internal diagnostic information)
-----------------------------------------------------------------

.  .  .  .
```

```
0         (0)    BITSTRING     0     IXLRSNCODEBADASYNCXISEQNUM
                                       “X'000008A2'” The IXLAXISN service received an
                                       invalid input asynchronous cross-invalidation
                                       sequence number. The value specified in
                                       ASYNCXISEQNUM does not represent a valid
                                       asynchronous cross-invalidation
                                       sequence number returned on a previous IXLCACHE
                                       invocation.
0         (0)    BITSTRING     0     IXLRSNCODENOASYNCXICONN
                                       “X'000008A3'” The connector did not specify
                                       ASYNCXI=IxlConnAsyncXiYes on the IXLCONN
                                       invocation when connecting to the cache
                                       structure.
0         (0)    BITSTRING     0     IXLRSNCODENOANSAREA
                                       ”X'000008B9'” An answer area was not specified.
                                       When the connect requested asynchronous
                                       cross-invalidate processing, ASYNCXI=YES, and
                                       asynchronous cross-invalidates can occur as a
                                       result of the IXLCACHE request, an ANSAREA
                                       must be specified. The following commands can
                                       cause asynchronous cross-invalidates to
                                       occur: READ_DATA, WRITE_DATA, WRITE_DATALIST,
                                       DELETE_NAME, DELETE_NAMELIST, CROSS_INVAL,
                                       CROSS_INVALLIST and REG_NAMELIST
. . . .

  CFLEVEL constants
-------------------------------------------------------------------------------
0         (0)    X’17’    0 IXLCFLEVEL23       23  “CFLEVEL23”

. . . .
```

Notes to Information Solutions:

- New support is highlighted in **green**, updated support is highlighted in **blue**
- The Cross Reference section is not included in the ELD since the entire Data Areas input is generated automatically.

8 IXCYQUAA MAPPING MACRO

Publications impacted: z/OS MVS Data Area Volume 3 (GA32-0937)
z/OS MVS Sysplex Services Guide (SA23-1400)

8.1 z/OS MVS DATA AREA VOLUME 3 (GA32-0937)

The following field has been added to the IXCYQUAA mapping:

- QuReqRfAsyncXI

Structure QUREQFEATURES

```
OFFSET     OFFSET
   DEC        HEX  TYPE      LENGTH  NAME (DIM)     DESCRIPTION
-------------------------------------------------------------------------------------------------------
...

     0        (0)  STRUCTURE  32    QUREQFEATURES   Data for Query REQINFO=FEATURES

...

     3        (3)  BITSTRING  1     QUREQFEATURES1D
....

                   .... 1...       QUREQRFWRITEREADMETRICS
                                                   "'08'" List and cache write/read
                                                   measurement metrics and IXLMG
                                                   AMDALEVEL=3 supported on this system
                   .... .1..       QUREQRFIXLMGSID "X'04'" IXLMG SID supported on
                                                   this system.
                   .... ..1.       QUREQRFASYNCXI  "'02'" Support for asynchronous
                                                   cross-invalidation available on
                                                   this system
```

8.2 z/OS MVS Sysplex Services Guide (SA23-1400)

In*Sysplex Services for Communication (XCF) > Using the Cross-System Coupling Facility (XCF) > Obtaining XCF Information > Using the IXCQUERY Macro > Information Returned Inline to IXCQUERY*, make the following updates:

The information that IXCQUERY returns when you specify REQINFO=FEATURES is placed in a storage area that you specify with the FEATAREA parameter. The information is mapped by QUREQFEATURES in IXCYQUAA and includes the following:

. . . .

QuReqRfWriteReadMetrics
List and cache structure write/read measurement metrics and IXLMG AMDALEVEL=3 supported on this system

QuReqRfAsyncXi
Support for asynchronous cross-invalidation available on this system.