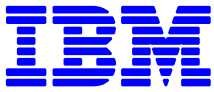
Parallel Sysplex Performance:
XCF Performance Considerations (Version 3.1)



This document can be found on the web, www.ibm.com/support/techdocs
Under the category of "White Papers."
This information was previously published as WSC FLASH10011.
This document is a complete replacement for the Flash.

Version Date: July 21, 2006

Joan Kelley
Kathy Walsh

Overview

Some installations implementing parallel sysplex have seen performance issues due to XCF signaling. These performance issues are generally solved by tuning changes to the XCF transport class definitions, buffer definitions, and signaling paths. This document is intended to review recommended XCF configurations and known performance tuning options.

This document has been updated in Version 3.1 to add information on:

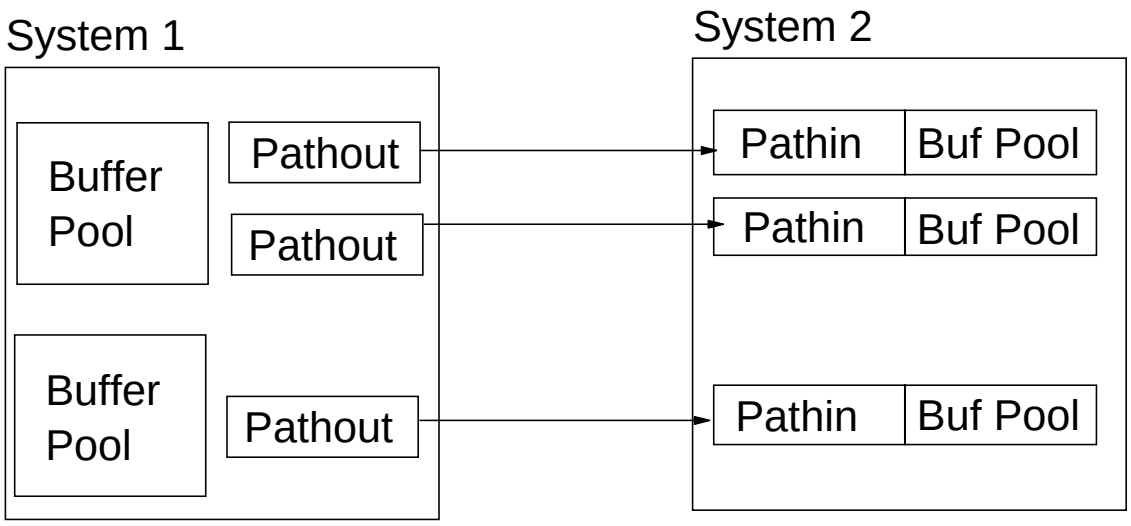
- New MAXMSG recommendations
- New signaling performance information when using FICON CTCs
- New display commands

Tuning XCF

XCF signaling is used to communicate between various members of a sysplex. The user of XCF signaling, usually an MVS component or a subsystem, issue messages to members within the user's group. The content and/or use of these messages are unique to the members of the group.

As XCF messages are generated, they are assigned to a transport class based on group name and/or message size. The messages are copied into a signal buffer from the XCF buffer pool. The messages are sent over outbound paths, (PATHOUT), defined for the appropriate transport class. Messages from other systems are received by inbound paths, (PATHIN). Inbound paths are not directly assigned transport classes, although a correlation can be made about which transport class messages are received via the inbound paths based on the outbound path to which the inbound side is connected.

The following is a diagram which highlights the XCF message traffic.



The key to ensuring good performance for the XCF signaling service is to provide sufficient signaling resources, namely message buffers, message buffer space, and signaling paths, and to control access to those resources with the transport class definitions.

Transport Classes

Transport classes are used to group messages. Using the CLASSDEF parameter in the COUPLExx parmlib member you can assign messages to a transport class based on the group name, the message size, or both.

Each transport class has its own resources which consists of a buffer pool and one or more outbound signaling paths. It is recommended you keep the number of transport classes small. In most cases, it is more efficient to pool the resources and define the transport class based on message size. Some initial product documentation recommended separate transport classes for GRS or RMF. These recommendations are no longer advised. If you do have separate transport classes for specific groups based on early product recommendations you should consider changing these definitions.

Message Buffers

XCF message buffers are managed by correctly selecting the size of the message most frequently sent from specific buffer pools and by specifying an adequate upper limit for the size of the buffer pool.

Message Buffer Size

First let's look at the individual message buffer size definitions. Message buffer size is determined by the CLASSLEN parameter on the CLASSDEF statement in the COUPLExx parmlib member. An example of this specification in the COUPLExx parmlib member is:

```
CLASSDEF CLASS(DEFSMALL) CLASSLEN(956)   GROUP(UNDESIG)
CLASSDEF CLASS(DEFAULT)  CLASSLEN(16316) GROUP(UNDESIG)
```

The CLASSLEN value determines the size of the most frequent message expected in this transport class. If a message could be assigned to more than one transport class, XCF selects the one with the smallest buffer which will hold the message. If the signal is larger than the CLASSLEN for any of the assigned transport classes, XCF has to choose a transport class to expand. XCF assigns the message to the transport class with the largest buffer size and expands the buffer size of this transport class.

Expanding the message buffer entails some overhead. The PATHOUT on the sending side and the PATHIN on the receiving side must be cleared out and expanded to handle the larger buffer size. A new, larger buffer must be obtained on the PATHIN side. If no additional messages of this size are received in a short time period, XCF then contracts the PATHIN, PATHOUT, and buffer sizes. In both of these cases extra XCF internal signals are generated to communicate these changes.

The best way to eliminate the overhead of expanding and contracting the message buffers is to define transport classes based solely on the size of the message buffers. One class with the default length of 956 should handle most of the traffic. A second class can be defined to handle larger messages.

The parameter GROUP(UNDESIG) specifies the messages should be assigned to the transport class based solely on message size. This definition makes all the resources available to all users and provides everyone with peak capacity.

There may be times when you want a separate transport class for a specific group. For instance, if you have a particular XCF user which is consuming a disproportionate amount of XCF resources, you may want to isolate this user to a separate transport class to investigate the user's behavior and protect the other XCF users. Hopefully, after you have diagnosed the problem, you can reassign this user to a transport class based on the length of the messages. XCF can dynamically add and delete transport classes by using the SETXCF command.

You can use an RMF XCF report to determine how well the messages fit:

XCF USAGE BY SYSTEM

REMOTE SYSTEMS

OUTBOUND FROM JB0

TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	BUFFER % SML	BUFFER % **FIT**	BUFFER % BIG	BUFFER % OVR	....
JA0	DEFLARG	16,316	189	98	1	1	100	
	DEFAULT	956	**55,794**	0	**100**	0	0	
JB0	DEFLARG	16,316	176	100	0	0	0	
	DEFAULT	956	**44,156**	0	**100**	0	0	
JC0	DEFLARG	16,316	176	100	0	0	0	
	DEFAULT	956	**34,477**	0	**100**	0	0	....
TOTAL			134,968					

%SML is the % of messages smaller than the buffer length
%FIT is the % of messages which fit the buffer length
%BIG is the % of messages larger than the buffer length

In this example, the majority of the messages fit in the DEFAULT class. A few exceeded the size of the DEFLARG class, but not enough to justify the definition of a new transport class.

Note: XCF has internal buffers of fixed size: 1K, 4K, 8K, ..64K. XCF uses 68 bytes for internal control blocks. So if you specify a length which doesn't fit one of these sizes, XCF will round up to the next largest size. For example, if you specify 1024, it will not fit into the 1K block (1024-68=956), and XCF will round up to the next largest block. If you issue a command, D XCF,CLASSDEF, it will list the CLASSLEN specified in the PARMLIB member, in this example, 1024. The RMF XCF report will show the actual buffer length, in this case 4028.

XCF provides operator commands to dynamically tune transport classes. You can use an operator command, D XCF,CD,CLASS=ALL to get information about the current behavior of the XCF transport classes. The command, which has a single image in scope, returns information regarding message traffic throughout the sysplex. The command returns infomation regarding the size of messages being sent through the transport class to all members of the sysplex and identifies current buffer usage needed to support the load.

The following example shows the data which is returned by the command. In this example the message output has been edited to display a representative sample of the returned data.

```
D XCF,CD,CLASS=ALL
       IXC344I  09.51.37  DISPLAY XCF 282
       TRANSPORT     CLASS        DEFAULT     ASSIGNED
       CLASS         LENGTH       MAXMSG      GROUPS
       DEFAULT         956          2500      UNDESIG
       LARGE         16316          2500      UNDESIG
       SMALL          4028          2500      UNDESIG

       LARGE TRANSPORT CLASS USAGE FOR SYSTEM SYSA
       SUM MAXMSG:       5000    IN USE:        180  NOBUFF: 0
       SEND CNT:     296373  BUFFLEN (SML):  8124
       SEND CNT:         47  BUFFLEN (SML): 12220
       SEND CNT:     282274  BUFFLEN (FIT): 16316
       SEND CNT:         10  BUFFLEN (BIG): 20412
       SEND CNT:         36  BUFFLEN (BIG): 24508
       SEND CNT:          4  BUFFLEN (BIG): 62464
       LARGE TRANSPORT CLASS USAGE FOR SYSTEM SYSB
       SUM MAXMSG:       5000    IN USE:        180  NOBUFF: 0
       SEND CNT:     648621  BUFFLEN (SML):  8124
       SEND CNT:     112985  BUFFLEN (SML): 12220
       SEND CNT:          8  BUFFLEN (FIT): 16316
       SEND CNT:          4  BUFFLEN (BIG): 62464
```

The command output will show information on the defined transport classes, and the message lengths they are processing. It will also show the buffer allocations (MAXMSG). SUM MAXMSG is the sum of the MAXMSG for the transport class and the MAXMSG values for any PATHOUTS to this system. For each transport class it will show the size and number of messages sent via the transport class. To use this command it is necessary to issue the command twice at some set interval. The SEND CNT is a display of internal buckets so it is necessary to build deltas on the SEND CNTs between the two displayed intervals. With this information you can determine how the transport classes are being used and if additional transport classes are needed.

In the example above very few messages are being sent which required a bigger buffer and so the LARGE transport class is sufficient and no changes need to be made. There is significant traffic which could use a smaller buffer (8k vs 16k). Sending a smaller message in a larger buffer does not cause increased XCF overhead but is a less efficient use of buffer space.

Message Buffer Pools

Having determined the optimal size for the individual message buffer, the next thing to do is select an upper limit for the amount of virtual storage to be allocated to the message buffer pool. The message buffer space is virtual storage used by XCF to store the message buffers which are being processed, sent or received.

Most of the virtual storage used for this purpose is backed by fixed storage. The storage to hold LOCAL buffers (for communication within the processor) is DREF (disabled reference) storagehich is backed by central storage. LOCAL buffers are used for messages within groups which are on the same MVS image. The most well known IBM exploiters of local messages are APPC, JES2, JES3, and GRS in Star Mode but any XCF group may choose to take advantage of LOCAL message processing.

XCF only uses the amount of storage it needs to support the load. But to ensure there are no surprises which stress available central storage, the installation can use the MAXMSG parameter to place an upper limit on the amount of storage which can be used for this purpose.

XCF storage is associated with the transport class, the outgoing paths, and the incoming paths, so MAXMSG can be specified on the CLASSDEF, PATHIN and PATHOUT definitions, or more generally on the COUPLE definition. MAXMSG is specified in 1K units. The default values are determined in the following hierarchy:

```
          OUTBOUND                          |                INBOUND
------------------------------------------|-----------------------------------
   PATHOUT - not specified, use           |  PATHIN - not specified, use
     CLASSDEF - not specified, use        |    COUPLE
       COUPLE                             |
```

Prior to z/OS 1.7 the default for MAXMSG is 750. For z/OS 1.7 and later releases the default for MAXMSG is 2000. The value for MAXMSG can be changed dynamically by issuing the SETXCF Modify command. By not specifying the default parameter in the parmlib definition you will automatically get the most current default size as you migrate to newer releases. Significant performance problems have been experienced as a result of insufficient XCF buffers. The default of 750 buffers is too small for most XCF configurations. The recommendation is to set MAXMSG on the COUPLE statement to at least the default of 2000.

For large installation the XCF environment should be reviewed to ensure there are no messages being rejected for lack of buffers. If there are rejected messages then the MAXMSG value should be adjusted upwards until the rejected messages are eliminated. While reviewing any buffer issues it would be worthwhile to ensure an XCF user did not suddenly start issuing many more messages and this is what is causing the stress on message buffers. Identifying and correcting excessive use of XCF should be an ongoing performance tuning objective. Simply adding message buffers without an understanding of the reason for the increased buffer requirement is just postponing a problem.

The total amount of storage used by XCF on a single system is the sum of:

- Sum of MAXMSG for all classes * systems in sysplex
- Sum of MAXMSG for all PATHOUTs
- Sum of MAXMSG for all PATHINs

In this example:

```
                          XCF PATH STATISTICS
-------------------------------------------------------------------------
                OUTBOUND FROM JB0                 INBOUND TO JB0
-----------------------------------        --------------------------
        T FROM/TO                                     T FROM/TO
TO      Y DEVICE, OR      TRANSPORT     ...  FROM     Y DEVICE, OR
SYSTEM  P STRUCTURE       CLASS              SYSTEN   P STRUCTURE
JA0     S IXCPLEX_PATH1   DEFAULT            JA0      S IXCPLEX_PATH1
        C C600 TO C614    DEFSMALL                      C C600 TO C614
        C C601 TO C615    DEFSMALL                      C C601 TO C615
        C C602 TO C616    DEFSMALL                      C C602 TO C616
JB0     S IXCPLEX_PATH1   DEFAULT            JB0      S IXCPLEX_PATH1
        C C600 TO C614    DEFSMALL                      C C600 TO C614
        C C601 TO C615    DEFSMALL                      C C601 TO C615
        C C602 TO C616    DEFSMALL                      C C602 TO C616
```

If a MAXMSG of 2000 was specified on the COUPLE statement and MAXMSG was not specified on the other parameters, the maximum storage which could be used by XCF is 44M:

- 2 classes * 3 systems * 2M = 12M
- 8 PATHOUTs * 2M = 16M
- 8 PATHINs * 2M = 16M

Note: This calculation implies if you add additional transport classes, signaling paths or systems, you will also be increasing the upper limit on the size of the message buffer pool.

Outbound Messages

For the outbound messages to a particular system if the sum of the storage for the CLASSDEF and the PATHOUTs is insufficient, the signal will be rejected. This is reported on the RMF XCF report as REQ REJECT for OUTBOUND requests. In general, any non-zero value in this field suggests some further investigation. The problem is generally resolved by increasing MAXMSG on the CLASSDEF or PATHOUT definition.

```
                          XCF USAGE BY SYSTEM
-------------------------------------------------------------------------
                                                REMOTE SYSTEMS
-------------------------------------------------------------------------
                          OUTBOUND FROM SYSC
-------------------------------------------------------------------------
                                                    ALL
TO          TRANSPORT  BUFFER        REQ          PATHS      REQ
SYSTEM      CLASS      LENGTH        OUT          UNAVAIL    REJECT
K004        DEFAULT       956    126,255   ...         0     1,391
            DEF16K     16,316         28               0         0
SYSA        DEFAULT       956     97,834               0         0
            DEF16K     16,316      3,467               0         0
                                ----------
TOTAL                           227,584
```

Inbound Messages

For the inbound messages from a particular system if the storage for the PATHINs is insufficient the signal will be delayed. This is reported on the RMF XCF report as REQ REJECT for INBOUND requests. If the delay causes signals to back up on the outbound side, eventually an outbound signal could get rejected for lack of buffer space. In this case, you may wish to increase the MAXMSG on the PATHIN definition.

XCF USAGE BY SYSTEM

REMOTE SYSTEMS			LOCAL	
	INBOUND TO SYSC		SYSC	
.....	FROM SYSTEM	REQ IN	**REQ REJECT**	TRANSPORT CLASS	**REQ REJECT**
	K004	117,613	1,373	DEFAULT	0
				DEF16K	0
	SYSA	101,490	0		
	TOTAL	219,103			

Another indicator the storage for PATHINs is insufficient is the BUFFERS UNAVAIL count on the XCF PATH STATISTICS report. If this is high check the AVAIL and BUSY counts: AVAIL counts should be high relative to BUSY counts. High BUSY counts can be caused by an insufficient number of paths or a lack of inbound space. First look at the inbound side to see if there are any REQ REJECTs. If so, increase the PATHIN MAXMSG. Otherwise, it is important to review the capacity of the signaling paths. The methodology for determining this is described later in this document.

Note: The RMF Communications Device report cannot be used to determine if the CTC devices are too busy. XCF CTCs will typically always report high device utilization because of the suspend / resume protocol used by XCF.

Local Messages

Local messages are signals within the same image, so no signaling paths are required. In this case, the message buffer storage used is the CLASSDEF storage plus any storage specified on the LOCALMSG definition. If MAXMSG is not coded on the LOCALMSG statement the additional message buffer storage contributed is none (0 buffers.)

Member Usage Information

An XCF group is a set of related members defined to XCF by a multisystem application. A member is a specific function, or instance, of the application. A member resides on one system and can communicate with other members of the same group across the sysplex.

Communication between group members on different systems occurs over the signaling paths connecting the systems; on the same system, communication between group members occurs through local signaling services.

To prevent multisystem applications from interfering with one another, each XCF group name in the sysplex must be unique. Information is provided in the Appendix on which components own specific group names.

Performance problems can often be traced back to an XCF group which dramatically increases its signaling rate. The increased signaling may cause capacity issues and the result may be a slowdown for both the group with the increased signaling rate as well as all other users of XCF services. The quickest way to determine which XCF group is using more signaling resources is to use the RMF Usage by Member report. The Usage by Member section gives information about messages sent to and from each remote system, broken down by remote group and member, and summarizes messages sent and received by the local system (the local system is the system on which the data was collected) broken down by local group and member. The following is an example of the report.

```
                  XCF USAGE BY MEMBER
-------------------------------------------------------------------
  MEMBERS COMMUNICATING WITH WSC2                MEMBERS ON WSC2
-----------------------------------------    ------------------

                          REQ        REQ
                          FROM       TO                        REQ
GROUP    MEMBER  SYSTEM   WSC2       WSC2      GROUP   MEMBER  OUT
SYSBPX   WSC1    WSC1       488        485     SYSBPX  WSC2    16,200
         WSC3    WSC3     1,073      1,072                   ---------
         WSC4    WSC4    11,832     11,825     TOTAL           16,200
         WSC5    WSC5       672        672
         WSC6    WSC6     2,128      2,117
                         -------    --------
TOTAL                     16,193     16,171
SYSGRS   WSC1    WSC1       719        719     SYSGRS  WSC2    13,773
         WSC3    WSC3     1,209      1,208                  ----------
         WSC4    WSC4     1,284      1,284     TOTAL           13,773
         WSC5    WSC5     1,152      1,152
         WSC6    WSC6     4,014      3,384
                         -------    --------
TOTAL                     8,378      7,747
```

By reviewing this information across time it should be possible to determine which XCF group has changed the rate at which it issues messages. Often the cause of the increased signaling is traced back to a group called IXCLOxxx where xxx is a system generated number. These XCF groups are dynamically created by XES to manage lock contention on either a lock structure or a serialized list structure. If an application using either of these structure types begins to experience increased lock contention one of the byproducts of this increased contention is increased XCF signaling traffic. This is because XES will use XCF services to perform lock negotiation for the structure in question.

By using XCF commands it is possible to identify which XCF signaling group is being used to support structure lock negotiation. This method of identifying the user of an IXCLOxxx structure only works if the RMF data is from the currently running system. If the entire Sysplex has been re-IPLed, or a group has been recycled on ALL members of the sysplex, the displayed information may not match the historical RMF data. If the RMF data being reviewed is from the current system then issue the following command:

```
D XCF,STR,STRNAME=J2CKPT1
     IXC360I  11.33.11  DISPLAY XCF       FRAME  1     F
     STRNAME: J2CKPT1
     STATUS: ALLOCATED
     POLICY SIZE    : 15000 K
     SYSTEM-MANAGED PROCESS LEVEL: 8
     ...
     XCF GRPNAME    : IXCLO006
```

In this example by displaying the lock structure J2CKPT1 you can identify the XCF Group which is going to be used to handle the lock contention. The XCF reports can then be used to understand the impact of the XES created group on XCF signaling resources.

If you also wish to know which XCF group/member is using the structure then issue the XCF command:

```
D XCF,GROUP,IXCLO006,ALL
     IXC333I  11.13.44  DISPLAY XCF 132
     INFORMATION FOR GROUP IXCLO006
     MEMBER NAME:        SYSTEM:     JOB ID:    STATUS:
     M363                SYSA        JES2       ACTIVE
     M370                SYSC        JES2       ACTIVE
     M372                SYSB        JES2       ACTIVE
     M375                SYSD        JES2       ACTIVE
```

This same XCF commands can also be used to help isolate an XCF group which is not behaving as expected, especially if the signaling rate has risen dramatically. By issuing the command D XCF,GROUP,*groupname*,ALL detailed information on the group's use of signaling services will be shown. Below is an example of the type of information you will see displayed.

```
D XCF,GROUP,SYSWLM,ALL
     IXC333I  10.15.04  DISPLAY XCF 103
      ...

     SIGNALLING SERVICE
     MSGO ACCEPTED:     656493  NOBUFFER:          0
     MSGO XFER CNT:     321097   LCL CNT:          0  BUFF LEN:   956
     MSGO XFER CNT:          2   LCL CNT:          0  BUFF LEN:  4028
     MSGO XFER CNT:         12   LCL CNT:          0  BUFF LEN:  8124
     MSGO XFER CNT:     335358   LCL CNT:          0  BUFF LEN: 12220
     MSGO XFER CNT:         24   LCL CNT:          0  BUFF LEN: 16316

     MSGI RECEIVED:     607870  PENDINGQ:          0
     MSGI XFER CNT:     641967  XFERTIME:       2091
```

With this information you can review which transport classes are now being used more heavily by the group and perhaps isolate the group to its own set of transport classes to isolate the increased usage thereby protecting the rest of the signaling environment. Once the cause of the increased signaling is known and corrected the group would no longer be needed. If the increased signaling is for new functions then additional signaling capacity can be provided and the transport classes used to temporarily isolate the group can be deleted. The CNT values above can wrap. So to use this information you must issue the command twice, say several minutes apart, and calculate deltas to see the true rates.

Signaling Paths

XCF signals from each transport class are sent out on the PATHOUT path and received into the system on the PATHIN paths. Tuning is achieved by altering the number or type of paths, or both. To review the XCF path configuration use the RMF XCF Path Statistics report. Two different issues commonly reported to IBM regarding signaling paths are reviewed in this document: no paths defined, and an insufficient number of paths defined.

Number of Paths

1. No paths

In the worst case, there may be NO operational paths for a transport class. This is not fatal. XCF routes the requests to another transport class but there is additional overhead associated with this operation. To determine if this condition exists, look at the RMF XCF Usage by System report. ALL PATHS UNAVAIL should be low or 0. In many cases this is caused by an error in the path definition; in other cases there may be a problem with the physical path.

XCF USAGE BY SYSTEM

					REMOTE SYSTEMS	
			OUTBOUND FROM SD0			
TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	....	**ALL PATHS UNAVAIL**	REQ REJECT
JA0	DEFAULT	16,316	189		0	0
	DEFSMALL	956	55,794		**55,794**	0
JB0	DEFAULT	16,316	176		0	0
	DEFSMALL	956	44,156		0	0
JC0	DEFAULT	16,316	176		0	0
	DEFSMALL	956	34,477	....	0	0
TOTAL			134,968			

In this example, the CTC links to system JA0 had been disconnected.

In the **next** example from the same system notice for system JA0 there were no paths for the transport class DEFSMALL so all the requests were re-driven through the DEFAULT class.

```
                          XCF PATH STATISTICS
--------------------------------------------------------------------------
                            OUTBOUND FROM SD0
--------------------------------------------------------------------------
         T FROM/TO
TO       Y DEVICE, OR      TRANSPORT      REQ   AVG Q
SYSTEM   P STRUCTURE       CLASS          OUT   LNGTH    AVAIL  BUSY RETRY
JA0      S IXCPLEX_PATH1   DEFAULT     56,011    0.16   55,894   117     0
JB0      S IXCPLEX_PATH1   DEFAULT        176    0.00      176     0     0
         C C600 TO C614     DEFSMALL   16,314    0.01   16,297    17     0
         C C601 TO C615     DEFSMALL   15,053    0.01   15,037    16     0
         C C602 TO C616     DEFSMALL   15,136    0.01   15,136    20     0
JC0      S IXCPLEX_PATH1   DEFAULT        176    0.00      176     0     0
         C C600 TO C614     DEFSMALL   11,621    0.01   11,515   106     0
         C C601 TO C615     DEFSMALL   13,086    0.01   12,962   124     0
         C C602 TO C616     DEFSMALL   11,626    0.00   11,526   100     0
```

Is it necessary to correct the 'ALL PATHS UNAVAIL' condition? In most cases it is. In the example above, DEFSMALL was defined to hold small messages (956). Because there is no path, they are being re-driven through the **DEFAULT** class. The DEFAULT class is sending data in large buffers (16,316 bytes). This is certainly not an efficient use of message buffer storage to transfer a 956 byte message in a 16,316 byte buffer. Re-driving large messages through a transport class defined with small messages causes more problems. It causes the buffers in this class to expand and contract with all the extra signaling explained previously. Defining separate classes is done for a purpose. If you don't provide paths for these classes, it negates this purpose.

2. Insufficient number of paths

Signaling paths can be CTC links or Coupling Facility structures. In the example above, the TYP field indicates the connection is a CF structure (S) or a CTC link (C). Since these two types of paths operate in unique ways different methods are used to evaluate their performance.

a. CF structures:
   For CF structures, an insufficient number of PATHOUT links could result in an increase in the AVG Q LNGTH, and high BUSY counts relative to AVAIL counts. Additional paths are obtained by defining more XCF signaling structures in the CFRM policy and making them available for use as PATHOUTs (and/or PATHINs).

   Note: RETRY counts should be low relative to REQ OUT for a transport class. A non zero count indicates a message has failed and was resent. This is usually indicative of a hardware problem.

b. CTCs
   CTCs can be configured in a number of ways. The installation can define CTC's as unidirectional (one PATHOUT or one PATHIN per physical CTC) or bi-directional (one or more PATHOUTs and PATHINs on a physical CTC). Due to the nature of XCF channel programs, a unidirectional path definition can achieve the most efficient use of a CTC thus providing the best XCF response time and message throughput capacity. However, a unidirectional definition will also require using **at least four physical CTCs** to configure for

availability. As will be noted in the capacity planning section below, two paths are usually sufficient for most systems, thus only those customers with very high XCF activity, (requiring >=4 paths), should consider using the unidirectional definition.

Using the AVG Q LEN on the RMF XCF report for CTCs is **not a good indicator** to determine if there are enough CTCs for a particular transport class. In the case of CTCs queued requests are added to the CCW chain which can increase efficiency. To insure capacity for heavier or peak workloads check the channel utilization for the CTCs as reported on an RMF Channel Activity report. In laboratory testing, acceptable XCF message response times were observed even at channel utilization of 70% (or 90% when there were multiple CTCs per transport class). Beyond this threshold, response time degenerated rapidly.

For CTC definitions the recommendation is to not define multiple device unit addresses for the same transport class on the same CTC channel. Each device address would look like an available pathout to XCF. If two signals arrive then XCF may choose to put a signal on each device address since from the software's point of view both addresses appear available. However, once the signals reach the channel subsystem contention will happen. For ESCON devices one signal will use the channel and the other will have to wait. For FICON devices both signals can use the channel but the signal data will be interspersed and overall connect times will potentially include the transfer times of both signals.

The multiple pathouts for the same transport class on the same CTC may cause XCF to not use a pathout on a different CTC or structure which may be available. So the definition may cause signals to queue to a busy CTC when other signaling resources are available.

Message Transfer Time

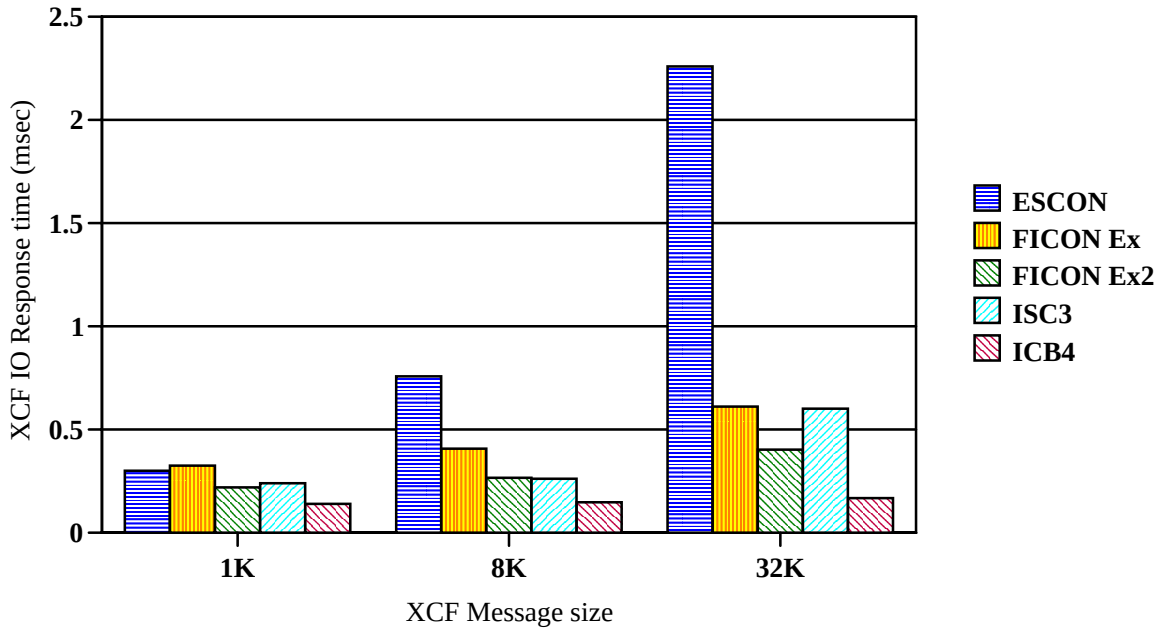
An indicator to use to determine if sufficient capacity exists on the signaling path is the Display XCF command. This command displays the response time of messages on the path as seen by XCF. The MXFER TIME is the mean transfer time in microseconds for up to the last 64 signals received within the last minute. If the MXFER TIME is acceptable, less than 2 milliseconds, (or 2000 microseconds), there is probably enough signaling path capacity. The response times are provided only on PATHINs.

```
D XCF,PI,DEVICE=ALL,STATUS=WORKING
IXC356I  12.02.12  DISPLAY XCF 901
LOCAL DEVICE    REMOTE   PATHIN      REMOTE                  LAST     MXFER
PATHIN          SYSTEM   STATUS      PATHOUT RETRY  MAXMSG  RECORD   TIME
C200            JA0      WORKING     C200       10     500    3496     339
C220            JA0      WORKING     C220       10     500    3640     419
```

RMF has also provided support for the message transfer time and will store the MXFER TIME as observed in the last minute before the end of the RMF interval in the RMF SMF 74 subtype 2 record in a field called R743PIOT. This enables a historical view of the performance of the signaling paths across multiple intervals. This data is in the RMF record but is not formatted on any of the XCF RMF reports.

Type of Signaling Path
A CTC provides a direct path between two systems, while sending a message through a CF is a two step, push-pull process. Thus, depending on message size and the type of CF link, CTCs are sometimes faster than using CF structures.

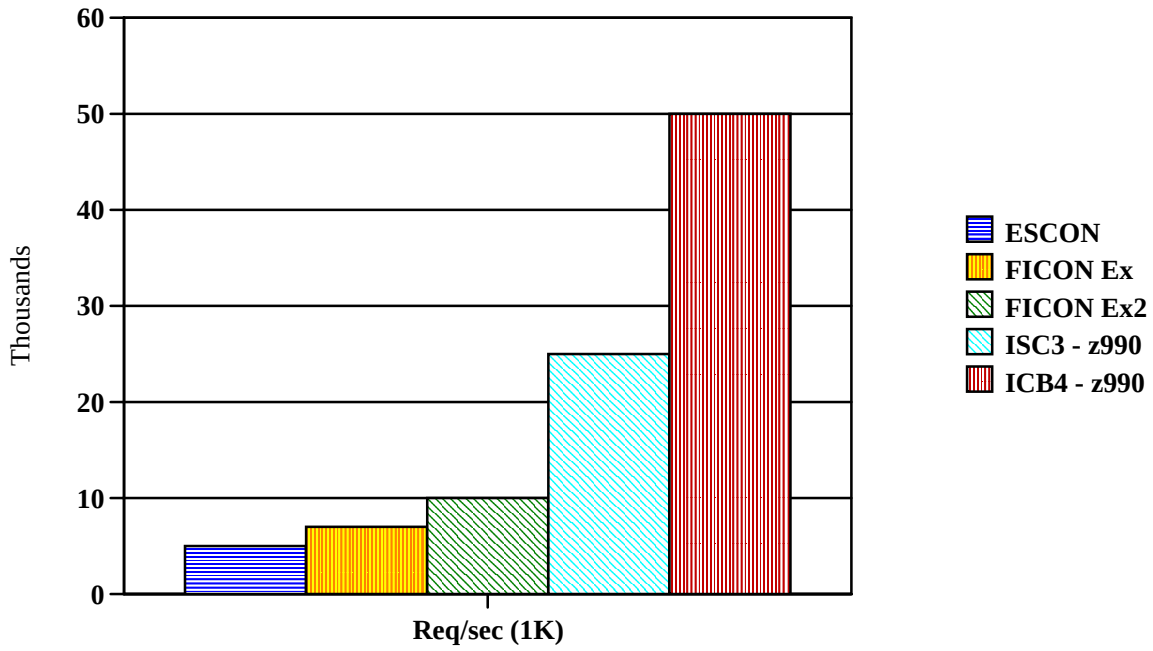
The chart below shows examples of XCF response time, (MXFER TIME), from controlled experiments in a test environment on a z990 CEC. The three types of CTC configurations have 4 pairs of PATHIN and PATHOUT per physical CTC.



A comparison of these examples shows the ESCON CTCs are slightly faster than the FICON Express CTCs for 1K messages but ESCON is slower when compared to FICON Express 2 as well as the newer ISC3 and ICB4 links. Though ESCON CTCs are now slower than newer technology the service times of small (1K) messages on ESCON CTCs are still very acceptable and may meet the requirements of many XCF users.

For larger messages, the newer CTC technology (FICON and FICON Express) is faster than ESCON. FICON Express 2 compares favorably to the ISC3 links. For very large messages ICBs are the fastest option. This results from the higher bandwidth associated with ICB coupling links compared to CTCs.

The capacity of the different types of links is also an important factor to understand when planning your XCF configuration. Below is a chart which describes the request rates for the different technologies when sending 1K messages. This chart shows the point at which the response time doubled for each type of link, indicating its saturation point.



High signaling environments may benefit from using ICB and ISC links due to the increased bandwidth these links afford. Less intense signaling environments may choose to use either structures or CTCs especially if the majority of the message traffic is small messages.

XCF internally times the various signals and gives preference to the faster paths. In the following example, compare the number of requests for DEFSMALL which were sent through the structure to the number which were sent through the CTCs. It should be noted XCF does not attempt to balance the workload across paths; once it finds a fast path, it continues to use it. APAR OW38138 describes changes which improves the path distribution.

```
                              XCF PATH STATISTICS
--------------------------------------------------------------------------

                               OUTBOUND FROM JA0
--------------------------------------------------------------------------
         T FROM/TO
TO       Y DEVICE, OR       TRANSPORT       REQ   AVG Q
SYSTEM   P STRUCTURE        CLASS           OUT   LNGTH   AVAIL  BUSY RETRY
JC0      S IXCPLEX_PATH1    DEFAULT       1,744    0.00   1,176     0     0
         S IXCPLEX_PATH2    DEFSMALL      8,582    0.01   8,362   220     0
         C C600 TO C614     DEFSMALL     20,223    0.01  20,160    63     0
         C C601 TO C615     DEFSMALL     23,248    0.01  23,229    19     0
         C C602 TO C616     DEFSMALL     23,582    0.01  23,568    14     0
```

In many environments, the difference in response time between CTCs and CF structures is indiscernible and using CF structures certainly simplifies management of the configuration.

Capacity Planning

For availability, a minimum of two physical paths must be provided between any two systems. This can be accomplished with two physical CTCs, structures in each of two different CFs, or a combination of CTCs and CF structures.

Most environments will find the rate of XCF traffic can be handled by the two paths which were configured for availability. Only for environments with very high rates of XCF traffic would additional paths be required.

The XCF message rate capacity of a path is affected by many factors:

1. The size of the message
2. How the paths are defined
3. If the path is also used for other (non-XCF) functions such as VTAM, or GRS.

Based on these factors, message rates (XCF IN+OUT), have been observed from 1000/sec to 5000/sec on a CTC, up to 9000/sec via an ICB and up to 4000/sec per HiPerLink. The adage "Your mileage may vary" is certainly true here.

When using CF structures for XCF messaging, there is also a cost in CF CPU utilization which needs to be included in the capacity plan. As an example, running 1000 XCF messages/sec through an R06 CF would utilize approximately 10% of one CF processor. Additionally, if you use CF structures as XCF paths, make sure the structure size is adequate. You can use the CF sizer available on the z/OS website, www.ibm.com/servers/eserver/zseries/cfsizer to obtain an initial estimate for the structure size. If the structure is too small, you will see an increase in the number of REQ REJECT and AVG Q LNGTH, and these events will definitely affect response time.

CTC Configuration Planning

When configuring CTCs for large volumes of XCF traffic some additional configuration planning needs to be done. CTC I/O will use SAP capacity, and large XCF environments can generate I/O rates much higher than traditional DASD and Tape workloads.

The SAP acts as an offload engine for the CPUs. Different processor models have different numbers of SAPs, and, though highly unusual, additional SAP processor may be defined if needed. SAP functions include:

- Execution of ESA/390 I/O operations. The SAP (or SAPs) are part of the I/O subsystem of the CPC and act as Integrated Offload Processor (IOP) engines for the other processors.
- Machine check handling and reset control
- Support functions for Service Call Logical Processor (SCLP)

In high volume XCF environments planning should be done to ensure the CTC configuration is defined so the CTC I/O load is spread across all available SAPs. Information on channel to SAP relationships can be found in the *IOCP User's Guide and ESCON CTC Reference*, GC38-0401-11. With the introduction of the z900 and follow on processors additional performance information on SAP utilization can be found by using an RMF IOQ Activity report which records SAP utilization.

Summary

A basic tenet of parallel sysplex performance is to ensure the XCF signaling environment is well performing. Installations need to ensure sufficient resources have been provided in the areas of message buffers and signaling paths. It is also important when adding processor capacity to a parallel sysplex you remember to include the necessary planning to ensure there is adequate XCF resources to support the anticipated growth.

The rest of this document contains additional information on managing an XCF environment. There is a case study provided which shows a step by step approach for resolving commonly seen XCF performance problems. The case study builds on the information found in the body of this white paper. The Appendix to the document contains important XCF APAR information and provides information on XCF group names and the z/OS component or program product which owns the XCF group. It also will contain a representative example of the Couplexx parameter library for reference. The Appendix also contains information on the z/OS Health Checker and describes XCF related checks which are performed.

XCF when configured for reliability is a robust signaling service. XCF performance data alone may not be sufficient to understand and resolve performance problems. The XCF data should be paired with more specific application information to get the most complete picture of the performance of the XCF signaling environment.

Case Study:

This is a case study which illustrates some of the items discussed.

An application was invoked which was changed to use CF signaling. When the workload was increased XCF delays increased. This was evident from messages like ERB463I which indicated the RMF Sysplex Data Server was not able to communicate with another system because the XCF signaling function was busy.

Looking at RMF Monitor III it showed:

```
                        RMF 1.3.0  XCF Delays
Samples: 120     System: J90   Date: 02/07/97  Time: 13.03.00

              Service    DLY     ------------ Main Delay Path(s)
Jobname   C   Class       %       %  Path    %  Path    %  Path
WLM       S   SYSTEM      87      87  -CF-
*MASTER*  S   SYSTEM      10      10  -CF-
RMFGAT    S   SYSSTC       3       3  -CF-
JESXCF    S   SYSTEM       1       1  C601
```

Comparing the RMF XCF reports to some earlier reports, it was noticed the amount of XCF traffic had quadrupled and the increase was in the class with the larger CLASSLEN (DEFAULT on this system).

In order to protect other XCF users and to investigate what was happening, a decision was made to separate these messages into their own transport class. A new transport class, NEWXCF, was defined using the GROUP keyword to specifically assign messages from the new application to this class. Since it was known the messages were bigger than the transport class with the smaller CLASSLEN (DEFSMALL), using the new XCF display command to look at actual message sizes being sent it was decided the messages would fit into a 8K(-68) buffer. This report was generated:

```
                                    ----- BUFFER -----       ALL
TO      TRANSPORT  BUFFER      REQ   %    %     %    %     PATHS     REQ
SYSTEM  CLASS      LENGTH      OUT   SML  FIT   BIG  OVR   UNAVAIL   REJECT
JA0     DEFAULT    20,412    1,715   90   10    0    0         0       0
        DEFSMALL      956   37,687    0  100    0    0         0       0
        NEWXCF      8,124  103,063    0  100    0    0         0   3,460
JB0     DEFAULT    20,412    2,075   92    8    0    0         0       0
        DEFSMALL      956   38,985    0  100    0    0         0       0
        NEWXCF      8,124  117,727    0  100    0    0         0     195
```

Now all the messages fit, but some are being rejected. This suggests message buffer space for the outbound path is no longer large enough.

The XCF path statistics confirm outbound messages are queuing up.

TO SYSTEM	T/P	DEVICE, OR STRUCTURE	TRANSPORT CLASS	REQ OUT	**AVG Q LNGTH**	AVAIL	BUSY
JA0	S	IXCPLEX_PATH1	DEFAULT	1,715	0.00	1,715	0
	S	IXCPLEX_PATH2	DEFSMALL	486	0.00	486	0
	S	IXCPLEX_PATH3	NEWXCF	103,063	**1.42**	102,818	245
	C	C600 TO C584	DEFSMALL	13,644	0.00	13,644	0
	C	C601 TO C585	DEFSMALL	13,603	0.00	13,603	0
	C	C602 TO C586	DEFSMALL	12,610	0.00	12,610	0
JB0	S	IXCPLEX_PATH1	DEFAULT	2,075	0.00	2,075	0
	S	IXCPLEX_PATH2	DEFSMALL	737	0.00	737	0
	S	IXCPLEX_PATH3	NEWXCF	117,727	**1.26**	117,445	282
	C	C610 TO C584	DEFSMALL	16,391	0.00	16,391	0
	C	C611 TO C585	DEFSMALL	12,131	0.01	12,131	0
	C	C612 TO C586	DEFSMALL	12,294	0.00	12,294	0

Increasing the MAXMSG on the PATHOUT for the NEWXCF transport class from 1000 to 2000 clears up the queuing delays.

				----- BUFFER -----				ALL	
TO SYSTEM	TRANSPORT CLASS	BUFFER LENGTH	REQ OUT	% SML	% FIT	% BIG	% OVR	PATHS UNAVAIL	REQ REJECT
JA0	DEFAULT	20,412	2,420	93	7	0	0	0	0
	DEFSMALL	956	41,215	0	100	0	0	0	0
	VTAMXCF	8,124	133,289	0	100	0	0	0	0
JB0	DEFAULT	20,412	2,362	93	7	0	0	0	0
	DEFSMALL	956	39,302	0	100	0	0	0	0
	VTAMXCF	8,124	143,382	0	100	0	0	0	0

The BUSY conditions are reduced, and more importantly the AVG Q LNGTH has been greatly reduced. Since the pathout with the contention is a coupling facility structure AVG Q LNGTH is an appropriate metric to use when tuning.

TO SYSTEM	T/P	FROM/TO DEVICE, OR STRUCTURE	TRANSPORT CLASS	REQ OUT	**AVG Q LNGTH**	AVAIL	BUSY
JA0	S	IXCPLEX_PATH1	DEFAULT	2,420	0.00	2,420	0
	S	IXCPLEX_PATH2	DEFSMALL	361	0.00	361	0
	S	IXCPLEX_PATH3	NEWXCF	133,289	**0.08**	133,117	2
	C	C600 TO C584	DEFSMALL	12,700	0.00	12,700	0
	C	C601 TO C585	DEFSMALL	16,421	0.00	16,421	0
	C	C602 TO C586	DEFSMALL	14,173	0.00	14,173	0
JB0	S	IXCPLEX_PATH1	DEFAULT	2,362	0.00	2,362	0
	S	IXCPLEX_PATH2	DEFSMALL	1,035	0.00	1,033	2
	S	IXCPLEX_PATH3	NEWXCF	143,382	**0.09**	143,086	296
	C	C610 TO C584	DEFSMALL	12,647	0.00	12,646	1
	C	C611 TO C585	DEFSMALL	15,944	0.00	15,944	0
	C	C612 TO C586	DEFSMALL	12,183	0.00	12,182	1

When determining how to tune the application to limit the number of XCF messages, a DEF8K transport class for UNDESIG messages was created and the NEWXCF class assigned to this application was eliminated.

Note: In this case study, the messages were being queued because the message buffer space was too small. If, instead of REJECTS, there was a high percentage of messages marked as BUSY, then increasing the number of signaling paths would have been appropriate.

Incidentally the path associated with the NEWXCF was a CF structure. The structure was chosen since it was quicker and easier to implement. Since the structure was receiving over 500 req/sec, it was unclear if the structure could handle the traffic. As can be seen from the queue lengths, it was capable of handling this rate.

Special Notices

This publication is intended to help the customer manage an z/OS Parallel Sysplex environment. The information in this publication is not intended as the specification of any programming interfaces provided by z/OS. See the publication section of the IBM programming announcement for the appropriate z/OS release for more information about what publications are considered to be product documentation. Where possible it is recommended to follow-up with product related publications to understand the specific impact of the information documented in this publication.

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Performance data contained in this document was determined in a controlled environment; therefore the results which may be obtained in other operating environments may vary significantly. No commitment as to your ability to obtain comparable results is any way intended or made by this release of information.

Appendix

APARS

The following APARs are directly related to XCF performance and/or RMF reporting of XCF performance:

z/OS 1.4 HIPER XCF APARS

OW51741	OA03359	OA02703	OA06505	OA01597
OA09731	OW54474	OA05330	OW56113	OW57715
OW57730	OA04616	OW54072	OA08462	OA07138
OA09388	OA05229	OW54315	OW55966	OW56630
OA02087	OW55305	OA02585	OW55429	OA08556
OA05232	OA06436	OA10351	OA06620	OW56611
OW53849	OW54542	OA03126	OW56355	OW56369
OA10868	OA05163	OA05391	OA08486	OA10197
OA02080	OW56633	OA05025	OW55992	
OA04344	OA07640	OW54685	OW55711	

z/OS 1.6 HIPER APARS

OA06505	OA04616
OA08556	OA05690
OA06620	OA06436
OA07640	OA08462
OA10197	OA08486
OA09388	
OW56611	
OA10351	
OA09731	
OA10868	

XCF Users

The following is a list of known XCF groups and the products/components which use them.

GROUP	OWNER
AOFSMGRP	AOC
ASFBGRP1	AOC
ATRRRS	* RRS
BBGROUP	CPSM
COFVLFNO	* VLF
DFHIR000	CICS
DSNDB1G	DB2
DXRDBZG	DB2
EJESEJES	EJES
ESCM	ESCOM MGR
EZBTCPCS	BatchPipes
IDAVQUIO	VSAM
IGWXSGIS	VSAM RLS
IRLMGRP1	IRLM
IRRXCF00	RACF
ISTCFS01	VTAM
ISTXCF	VTAM
IXCLOxxx	*# XES
JES2xx	$JES2 MAS
JES3xx	@JES3 Cmplx
POKUTC58	NJE-JES2
SYSATBxx	APPC
SYSBPX	Shared HFS
SYSDAE	* DAE
SYSENF	* ENF
SYSGRS	* GRS
SYSIEFTS	* ALLOCAS
SYSIGW00	DF/SMS - PDSE
SYSIKJBC	* TSO
SYSIOS	* IOS
SYSJES	* JES
SYSMCS	* CONSOLES
SYSMCS2	* CONSOLES
SYSRMF	RMF
SYSWLM	* WLM

```
* denotes MVS component
# one for each lock and serialized list
 structure
```

JES2xx - Local node name
JES3xz - Node name on NJERMT init stmt

Sample COUPLExx PARMLIB member

This PARMLIB member defines two transport classes:

DEFAULT - used for messages <= 956, defined with 4 PATHOUTs:
    1 CF structure named IXCPLEX_DEF1
    3 CTC connections
for each of the 10 systems in the SYSPLEX.

LARGE - used for messages >956, defined with 1 PATHOUT
1 CF structure named IXCPLEX_LRG1.

Since this is an z/OS 1.7 system, the MAXMSG default of 2000 is used for everything except the PATHIN and PATHOUT paths which use structures.

```
CLASSDEF CLASS(LARGE)   CLASSLEN(16316) GROUP(UNDESIG)
CLASSDEF CLASS(DEFAULT) CLASSLEN(956)   GROUP(UNDESIG)

LOCALMSG MAXMSG(2000) CLASS(DEFAULT)

PATHOUT  CLASS(DEFAULT) MAXMSG(2000) STRNAME(IXCPLEX_DEF1)
PATHOUT  CLASS(LARGE)   MAXMSG(3000) STRNAME(IXCPLEX_LRG1)
PATHIN   MAXMSG(2000)   STRNAME(IXCPLEX_LRG1,IXCPLEX_DEF1)

PATHOUT CLASS(DEFAULT)  DEVICE(C400,C410,C580,C590,C600,C610)
PATHOUT CLASS(DEFAULT)  DEVICE(C620,C630,C640,C650)
PATHIN                  DEVICE(C404,C414,C584,C594,C604,C614)
PAHTIN                  DEVICE(C624,C634,C644,C654)

PATHOUT CLASS(DEFAULT)  DEVICE(C401,C411,C581,C591,C601,C611)
PATHOUT CLASS(DEFAULT)  DEVICE(C621,C631,C641,C651)
PATHIN                  DEVICE(C405,C415,C585,C595,C605,C615)
PATHIN                  DEVICE(C625,C635,C645,C655)

PATHOUT CLASS(DEFAULT)  DEVICE(C402,C412,C582,C592,C602,C612)
PATHOUT CLASS(DEFAULT)  DEVICE(C622,C632,C642,C652)
PATHIN                  DEVICE(C406,C416,C586,C596,C606,C616)
PATHIN                  DEVICE(C626,C636,C646,C656)
```

XCF z/OS Health Checker Information

The IBM Health Checker for z/OS is a base z/OS component in z/OS 1.7. Installations can the Health Checker to gather information about their system environment to help identify potential configuration problems before they impact availability or cause outages. Individual products, z/OS components, or ISV software can provide checks which take advantage of the IBM Health Checker for z/OS framework. For additional information about available checks and about the IBM Health Checker for z/OS see IBM Health Checker for z/OS: User's Guide.

z/OS 1.4, 1.5, and 1.6 users can obtain the IBM Health Checker for z/OS from the z/OS download page at: http://ibm.com/servers/eserver/zseries/zos/downloads/

The following is a list of XCF related checks which run under the IBM Health Checker for z/OS with a brief explanation of the function the check performs. For additional information on these check please use the z/OS Health Checker product documentation.

XCF_CDS_SEPARATION	Check that sysplex couple data set and function couple data sets are properly isolated with alternates
XCF_CF_CONNECTIVITY	Checks that the system has connectivity to each CF
XCF_CF_STR_EXCLLIST	Check that each structure is excluded from all structures coded in its exclusion list
XCF_CF_STR_PREFLIST	Check that each structure is allocated according to the preference list in the CFRM policy
XCF_CLEANUP_VALUE	Check that the XCF cleanup interval is set to a reasonable value to hasten the removal of a failed system from the sysplex
XCF_DEFAULT_MAXMSG	For each path check that there is a MAXMSG of at least the indicated minimum value specified by or inherited from the COUPLExx, transport class definition, or path definition
XCF_FDI	Check that the XCF failure detection interval (FDI) equates to the formula "multiplier * SPINTIME + increment"
XCF_SFM_ACTIVE	Check that the status of Sysplex Failure Management (SFM) policy is as recommended
XCF_MAXMSG_NUMBUF_RATIO	Check each inbound signal path and ensure that each can support at least the indicated minimum number of messages from the sending system
XCF_SYSPLEX_CDS_CAPACITY	Check that the maximum number of systems, groups, and members have not at some time reached a threshold determined by the best practice amount of space required for growth of systems, groups, and members
XCF_SIG_CONNECTIVITY	Check that multiple pathin/pathout pairs are in the working state for each system in the sysplex connected to the current system
XCF_SIG_PATH_SEPARATION	Check for single points of failure for paths to all systems which are connected
XCF_SIG_STR_SIZE	Check that there are enough signaling structure entries to support full connectivity in the sysplex
XCF_TCLASS_CLASSLEN	Check that there are at least a certain number of different transport classes with unique class lengths defined
XCF_TCLASS_CONNECTIVITY	Check that all defined transport classes are assigned at least to the indicated number of pathouts (outbound paths)
XCF_TCLASS_HAS_UNDESIG	Check that all transport classes are set up to service the pseudo-group name 'UNDESIG'