Getting the most from APPC: A network tuning guide

March 10, 1994 9:15 a.m.

John Brady

(919) 254-4167
Internet: john_brady@vnet.ibm.com

APPC Market Enablement

Contents

How to contact IBM's APPC Market Enablement Group 1
What is APPC Market Enablement? 1

Why you should read this tutorial 3

Disclaimer 7

Why is tuning necessary? 9
SNA has a history of change 9

How this tutorial is organized 11
General Tuning guidelines section 12

Basic concepts 13
Terms 13
Concept: Data with DLC and SNA headers 15

General Tuning Guidelines 17
SNA Guidelines 17
Use APPC(LU6.2): the most advanced SNA LU 17
Use modes wisely 17
Request Units(RUs) 18
Trade offs 19
Pacing windows 20
Mode Consistency and Propagation 21
Consistency 21
Propagation 22
The cost of inconsistency 22
Data Link Control(DLC) Recommendations 23
DLC adapter card selection 23
DLC frame size 23
DLC pacing 24
Trade offs 24

Platform Specific Recommendations 27
Communications Manager/2 (CM/2) tuning recommendations 27
Increasing RU Size 27
Session pacing window value 27
Personal Computer(PC) LAN adapter cards 27
DLC frame size 28
DLC pacing 28
Networking Services/DOS (NS/DOS) tuning recommendations 29
RU size 29
RU pacing window size 29
PC LAN adapter cards 29

DLC frame size . . . 30
DLC pacing . . . 30
SNA Services/6000 v1.2 tuning recommendations . . . 32
DLC frame size . . . 32
DLC pacing . . . 32
LAN adapter queue depth . . . 32
RU size . . . 32
RU pacing window size . . . 32
Optimal memory buffer settings . . . 33
AS/400 tuning recommendations . . . 34
AS/400 performance tools . . . 34
RU size . . . 34
RU pacing . . . 34
LAN adapters(IOPs) . . . 34
Token Ring adapters . . . 35
Ethernet Lan adapters . . . 35
DLC Frame size: MAXFRAME parameter . . . 35
DLC pacing: LANMAXOUT and LANACKFRQ parameters . . . 35
NCP tuning recommendations . . . 36
RU Size: MAXBFRU and UNITSZ parameters . . . 36
BFRS . . . 36
SLODOWN . . . 37
DELAY . . . 37
NCP Token Ring lan adapters . . . 37
DLC frame size . . . 37
RCVBUFC . . . 37
MAXTSL . . . 38
DLC Pacing . . . 38
T2TIMER . . . 38
MAXOUT . . . 39
VTAM tuning recommendations . . . 40
RU size . . . 40
IOBUF . . . 40
Matching parameters across platforms for better performance . . . 42
VTAM with CICS 3.3 . . . 42
SENDSIZE and RECEIVESIZE . . . 42
Automatic session setup . . . 42

Appendix A. Test lab results - microcomputer and mainframe network . . . 45
Methodology . . . 45
The benchmarks . . . 45
Memory to memory data transfer . . . 45
What was measured . . . 45
The network . . . 46
Microcomputer and Mainframe test network . . . 46
Highlights . . . 46
Details of the data transfer and hardware used . . . 47
Base and tuned parameters . . . 49

Appendix B. "APPC Optimal Performance Configuration on OS/2 and DOS" 55
Test Environment . . . 55
Microcomputer test network . . . 55
Highlights . . . 56
OS/2 . . . 56
DOS . . . 56

Software 56
OS/2 56
DOS 56
Transactions 56
File Transfer Transaction 56
Inquiry Transaction 57
Benchmarks 57
OS/2 57
DOS 57
Configuration for Optimal Performance 57
Definition of Parameters 58
Parameter Settings 59
Table 1 - Tuned Values for APPC Parameters on OS/2 59
Table 2 - Tuned Values for APPC Parameters on DOS 60
Performance Results 61
OS/2 61
DOS 61
Conclusion 62

Appendix C. Determining pacing window sizes 63

Appendix D. Replacing a mode table in VTAM 65
Compiling 65
Reloading 65

Appendix E. Additional performance related publications 67

Appendix F. Bibliography 69

How to contact IBM's APPC Market Enablement Group

What is APPC Market Enablement?

APPC Market Enablement is a new department within the Architecture and Telecommunications group of the Networking Systems division. Our mission: to provide you with the information and support you need to market and use APPC, APPN, and CPI-C successfully. We provide a wide variety of different services to help you get the most from promoting and using APPC.

Publications and Forums: To spread the good news about APPC, we use a variety of media:

- Our articles appear regularly in IBM journals. We encourage the publication of features and updates on APPC in the trade press as well.
- We produce and distribute a comprehensive catalog of APPC development tools offered by IBM and other vendors.
- We also provide information on APPC through internal and external forums:
  - CompuServe
  - IBM Information Network

Consulting: Our team members have the skills and expertise to answer your questions and concerns about APPC, APPN, and CPI-C. We regularly provide consulting services on APPC to IBM marketing representatives and developers. We provide similar services directly to customers by special arrangement only.

How Can We Help You?

Please contact us and let us know how we can help you.

```
IBM APPC Market Enablement
P.O. Box 12195
Department E42, Building 673
Research Triangle Park, North Carolina USA  27709

IBM VNET:  APPCMRKT at RALVM6
Internet:  appcmrkt@ralvm6.vnet.ibm.com
CompuServe: GO APPC

Fax:  919-254-6050
```

Why you should read this tutorial

This tutorial is for network administrators who want better performance from their SNA networks and want to know what can be done to improve it. Administrators will need to read this document if:

- They want to know the latest tuning techniques available for APPC(LU6.2).
- They know there is more horsepower in their network(s) and want to leverage that power for their users.
- They have never before considered tuning APPC data transport for higher throughput.

Consider these benefits:

1. **You will have a good foundation in what makes SNA networks perform well or poorly.**
2. **You will be able to tune your network equipment to give your users better bulk data performance. A brief description of the improvements I made follows. As you can see, tuning really pays off.**

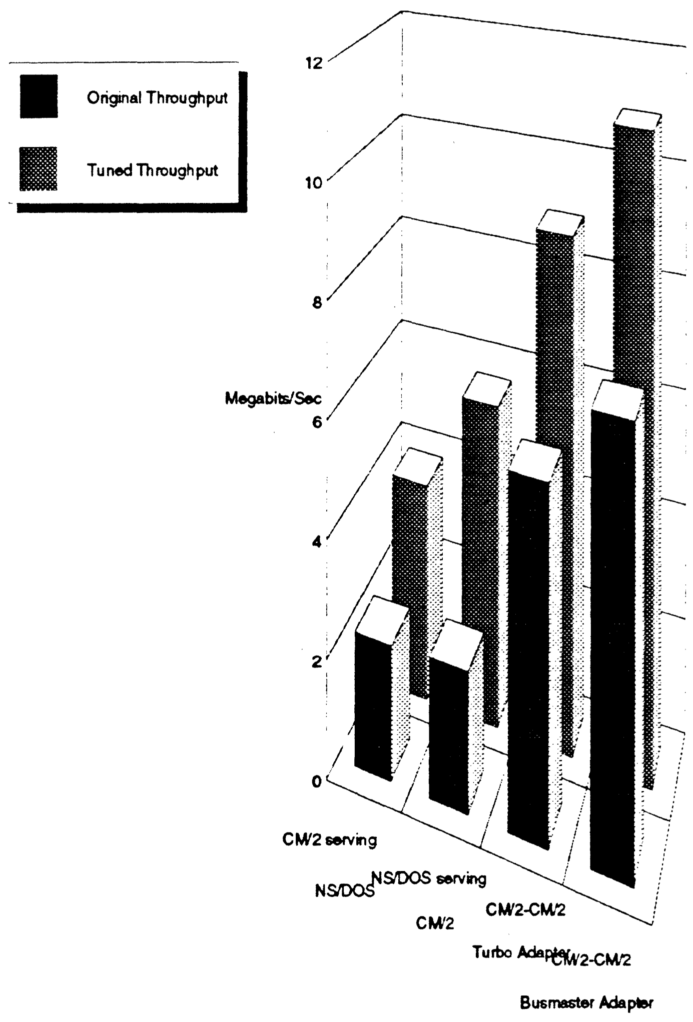

Figure 1. Affects of tuning a PC network

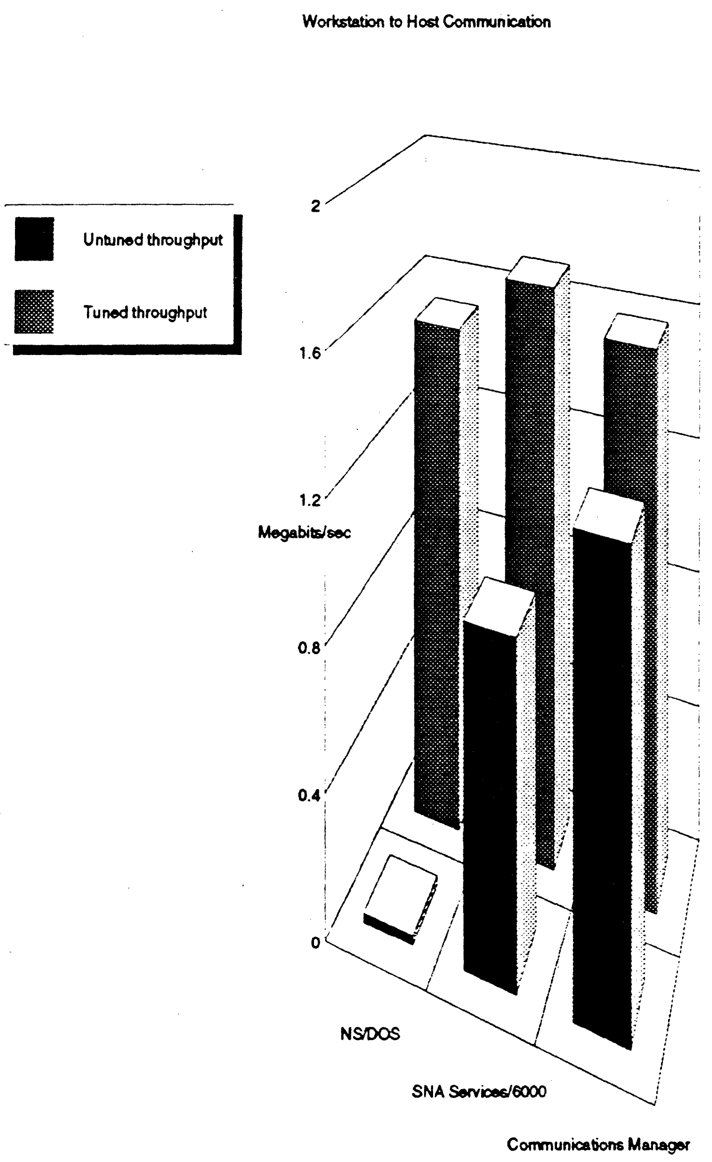

Figure 2. Affects of tuning a Host and PC network

Of course, network administrators are those who have the privilege to change network parameters. If network tuning is something you are not authorized to do, you should pass this good information along to someone who is authorized to make the changes I recommend.

Disclaimer

Since your data transfer applications and network will likely be different from the test networks I used to investigate performance, your results will not be the same as mine. These results are a subset of the whole performance picture, and it is important to note that the conclusions are based only on single data session throughput. The recommendations, therefore, may exceed what you can configure if your machine has to support many sessions/links/users concurrently. Some of the values you use may have to be less than what I recommend because of trade-offs between memory/hardware and the number of sessions, links, and conversations your machine needs to support.

Further perspective on what the numbers reported here mean to you is in "Methodology" on page 45.

Why is tuning necessary?

Simply stated, tuning is setting the parameters of the network software to best match and utilize physical resources. As physical and logical characteristics of networks change, so must the parameters that allow for efficient use of both. Tuning your network allows you to maximize your investment in SNA network software and hardware.

SNA has a history of change

SNA products have evolved for over twenty five years, and so have the networks which are built around them. This evolution comes from new applications, putting new demands on the network, new hardware expanding capacity, and new system software implementing advanced architectures.

As these networks and products have evolved, there have invariably been compromises: New network products must interoperate with older versions of many other products, but must also provide enhanced function and greater capacity. Users also had to make migration choices: network managers had to enable new high demand applications while maintaining response times for existing users and applications.

In response to all of these forces, SNA architecture developed a plethora of options to allow interoperability and evolution. Network parameters chosen for a new set of applications or users must be evaluated before new users are added to a network domain. In addition, the continuing increase of end user software and hardware capacity means that existing network connections and pathways must be evaluated for efficiency on a continuing basis.

The outcome of all of these forces is networks where tuning and planning are necessary.

How this tutorial is organized

"Basic concepts" on page 13 will explain basic terms and concepts used later in the paper.

Then, there are three main sections to this paper:

1. The first is "General Tuning Guidelines" on page 17. It covers some general rules and guidelines to follow when tuning **any** SNA network.
2. "Platform Specific Recommendations" on page 27 covers how to tune the products I tested.
3. In Appendix A, "Test lab results - microcomputer and mainframe network" on page 45 and Appendix B, ""APPC Optimal Performance Configuration on OS/2 and DOS"" on page 55, you'll find the detailed results of the tuning studies this paper is based on. Those tests were conducted at IBM labs in Research Triangle Park, NC. In that section, you'll find:
   a. The data transfer response times both before and after tuning.
   b. The parameter settings in the products both before and after tuning.
   c. Details of the networks. Use this section to help determine whether or not your tuning activities have fully realized your equipment's capabilities. If you have similar equipment to these networks, and your data throughput is orders of magnitude less than what I achieved, there is still a bottleneck somewhere in your network. More tuning, possibly not related to your network, should be done.

The data presented in Appendix A, "Test lab results - microcomputer and mainframe network" on page 45 was from tuning end station SNA products(NS/DOS, Communications Manager, and SNA Services/6000) communicating with a host(VTAM 4.1 for MVS).

The test results presented in Appendix B, ""APPC Optimal Performance Configuration on OS/2 and DOS"" on page 55 were originally reported in "APPC Optimal Performance Configuration for OS/2 and DOS" by Jill Bodine (called "the old report" hereafter). The old report is superseded by this report because of its inclusion here.

General Tuning guidelines section

The topics in this section apply to all SNA networks, due to the nature of the SNA network protocol.

Each topic is preceded by a

Recommendation

Which may be followed by an...

Exception

to the rule.

Platform Specific Recommendations section

These topics cover actual SNA software settings in real products. Parameter settings I recommend will appear as

Recommendations

In some instances, I recommend performing small experiments with performance and changing product parameters. Always use tools available with the products to ensure that the parameters are changing as desired. Protocol negotiations with other computers can result in network parameters less than what you have configured. This downward negotiation can lead to poor performance.

There is a very useful (and free!) tool offered by IBM's APPC Market Enablement Team which can be used to quantify performance gains in SNA networks. It is called **APING**. It is an APPC version of the popular SOCKETS **PING** tool. I suggest you install APING on all of your machines running SNA protocols. It will then be very easy to tell if and how much your tuning is affecting response time.

Basic concepts

Terms

Here are some terms you will see in this paper.

- **Acknowledgement**

  A message sent from one machine to another confirming receipt of data.

- **API**

  Application Programming Interface. A set of commands geared to perform a specialized set of tasks. Examples are APPC, CPIC, SOCKETS, and PostScript. This paper deals mainly with APPC(also known as LU6.2).

- **SNA**

  Systems Network Architecture. A network protocol used to transfer data from one computer to another.

- **Window**

  A quantity, or count, of data frames which machines remember they have sent. Each window of data transmitted requires an acknowledgement to be sent in return.

- **Pacing**

  A mechanism used to control the flow of data in SNA. Pacing allows large, fast machines to seamlessly operate with smaller, less capable, machines. Pacing is unidirectional. That means that a machine can have a different send window than its receive window.

- **Data Link Control (DLC)**

  The communications link protocol used to transmit data between two physically linked machines.

- **Request/Response Unit (RU)**

  A unit of SNA data which carries user and network control data. They are used to carry user application data from one machine to another and can vary greatly in size.

- **Path Information Unit (PIU)**

  An RU or RU segment with SNA routing information added.

- **Conwinner**

  Contention winner. A session for which an LU has won the right to speak first on any conversation mapped over it.

- **Conloser**

  Contention loser. A session for which an LU lost the right to speak first on any conversation mapped over it. The LU is second to speak on these sessions.

- **Negotiation**

A process used by two machines establishing a links sessions and conversations. Values like RU sizes, DLC frame sizes, pacing windows, etc. are decided upon during negotiation.

Concept: Data with DLC and SNA headers

The following figure illustrates the various routing information added to user data (RUs) for transmission.

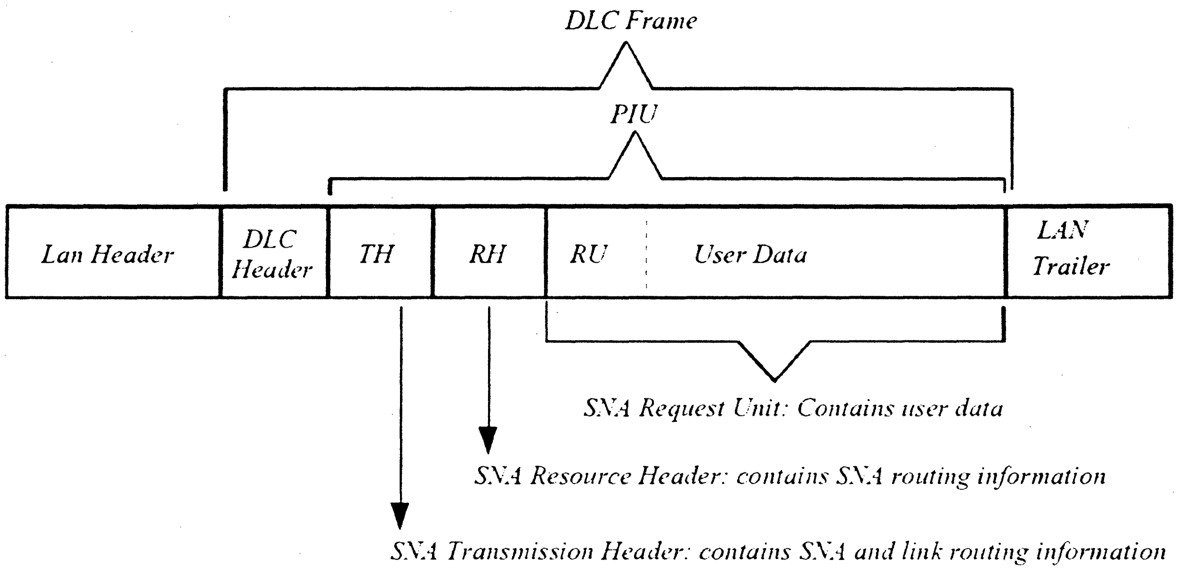
DLC Frame
PIU
Lan Header
DLC Header
TH
RH
RU
User Data
LAN Trailer
SNA Request Unit: Contains user data
SNA Resource Header: contains SNA routing information
SNA Transmission Header: contains SNA and link routing information

Figure 3. Data with DLC and SNA headers

General Tuning Guidelines

SNA Guidelines

Regardless of the platform, any SNA implementation will respond favorably to the tuning guidelines suggested in this section. Some variations in how SNA products are implemented will lead to some variation, though. You may have to make exceptions to these guidelines.

Use APPC(LU6.2): the most advanced SNA LU

Recommendation

Look for bulk data transfer applications in your network not written to the APPC API and either migrate them to APPC or replace them with APPC based data transfer applications. Plan to write all new applications using APPC.

Recommending you use LU6.2 in this tutorial may seem like stating the obvious, but differences exist between LU6.2 and older LU types that bear mentioning.

Early LUs, (e.g.LU0, LU2, LU3) are clearly dated technology. Of these, LU2 is the one most often misused for data transfer. This LU protocol is suited to old terminals which had limited ability to run their own communications protocols. LU2 devices can only send 32K bytes of data at a time, and must simulate an **ENTER** key to acknowledge EVERY record the client sends. These characteristics, as well as the high amount of control overhead needed to implement 3270 data stream, limit LU2's ability to move large amounts of data efficiently.

APPC(LU6.2) has features like a large send/receive record size (the architecture allows up to 2 gigabytes to be sent or received at a time, and some products allow up to 64 kbytes) and enhanced data buffering which enable it to move data very efficiently; and, once a LU6.2 conversation has been established, the size of the control overhead relative to the data being transported becomes very small.

There are several documents available to help with migration from LU2 to LU6.2. Contact IBM's APPC Market Enablement group to obtain any of the following:

1. APPC Design for HLLAPI and 3270 Programmers
2. HLLAPI or APPC: Design Criteria
3. CICS 3270 to APPC Client/Server: A Migration Methodology

Use modes wisely

Recommendation

For bulk data transfer applications, use a mode which has large RU and pacing window values, and uses a class of service which has a MEDIUM or LOW transmission priority. The architected mode #BATCH is a good mode to use for bulk data transfer.

SNA gives users many options to customize the characteristics of individual data sessions. These characteristics (like RU sizes, pacing window values, transmission priority, and security requirements) are held in profiles called MODES. It is important to use a mode that best balances the needs of an application to do its job with the needs of all users in the network. In this way, no "greedy" applications can dominate the use of network resources while other applications are locked out. Bulk data transfer requires modes with large RUs and pacing window sizes, and a secondary (not highest) transmission priority.

Request Units(RUs)

Recommendation

Specify an RU as large as possible for the mode your file transfer application uses. Then, using available system management tools, ensure that the RU size negotiated matches what you configured (see the individual product sections in "Platform Specific Recommendations" on page 27 for those tools).

IN ADDITION, BE SURE THE RU SIZE YOU CONFIGURE IS AN INTEGER MULTIPLE OF THE DLC FRAME SIZE. THIS WILL HELP YOU AVOID SENDING SMALL DLC FRAMES. SENDING SMALL DLC FRAMES IS DETRIMENTAL TO LINK THROUGHPUT.

- *If you plan to configure an RU which is close in size to your DLC frame size, be careful that an entire RU can fit into a DLC frame. This requires that the length of the DLC header be subtracted from the overall DLC frame size to determine an appropriate RU size. Some platforms have configuration options that automatically do this calculation for you. Examples include:*
  - *DEFAULT RU SIZE in Communications Manager/2*
  - ** for the RU size in NS/DOS's MODE.NSD file*
  - **CALC for the RU in the Mode Definition (MODD) statement in AS/400s.*
- *If you make your RU size larger than the DLC frame size, select an RU size that is as close as possible to an even multiple of the DLC frame size. There can be up to 9 bytes of SNA header information attached to an RU in APPN networks, and up to 29 bytes in Subarea networks:*
  - *For Subarea networks, make your RU size an even multiple of (the DLC frame size - 29)*
  - *For APPN networks, make your RU size an even multiple of (the DLC frame size - 9).*

Exception

The AS/400 calculates the most efficient RU size for the user. See "AS/400 tuning recommendations" on page 34 for AS/400 tuning recommendations.

A RU is the construct in SNA that carries user application data through the network. Its size can be uniquely configured for each MODE you have configured.

SNA products generally ship with default mode tables specifying RUs less that 2k bytes. This suits interactive traffic and users with small data transfer requirements. But, for bulk data transfers, paying attention to RU size is very important. A size of 4k bytes is good, and in terms of single session throughput, going all of the way to a 64k byte RU will theoretically provide a gain in performance.

Trade offs

Increasing the size of RUs that a machine sends and receives increases the amount of memory needed to store those RUs. This affects a data receiver more than a data sender. A data receiver must have enough memory available to receive a whole window of RUs: **(RU size) x (window size) bytes** while data senders only need one RU buffer.

Segmenting RUs: Segmenting occurs when an entire RU cannot fit into a DLC frame for transmission. The RU is broken into segments small enough to fit the DLC frame size. It is then up to the data receiver to reassemble. Example:

An RU segmented into two DLC frames

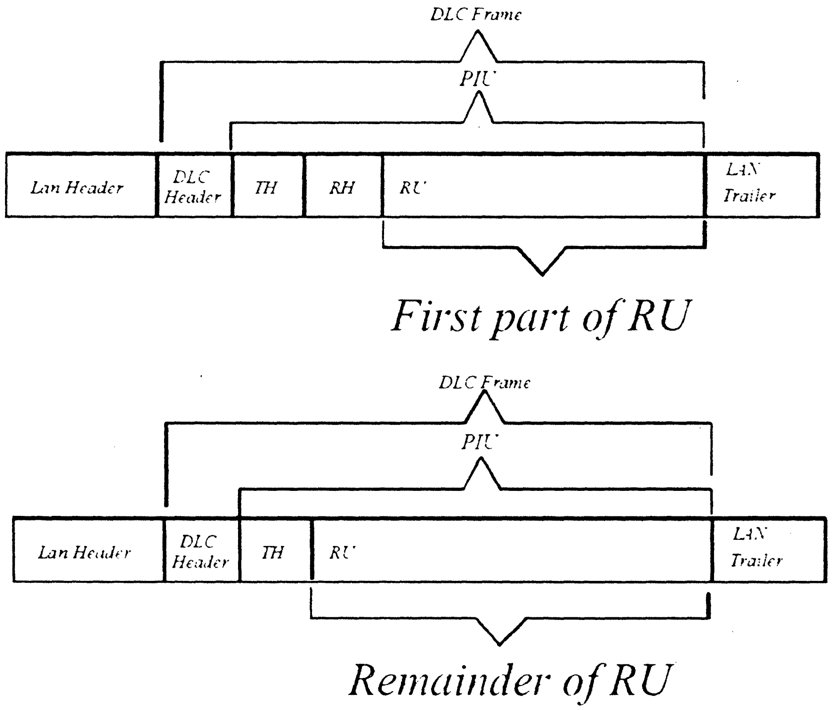

Figure 4. An RU segmented into two DLC frames

Segmenting should be utilized to its fullest capability with one caveat: ensure that all other tuning has been done before employing segmentation. It is best to increase the DLC frame size and pacing windows(DLC and RU) first.

Pacing windows

Recommendation

If your SNA product supports adaptive pacing, use the maximum allowable RU pacing window. Also insure that during bulk data transfers, the window is increasing to at least 8. See Appendix C, "Determining pacing window sizes" on page 63 for product specific help with this task. If the window is not opening up to 8, there is a resource constraint or configuration problem along the data path that must be addressed.

If your SNA product only supports fixed pacing use a window size of 8. Then, increase the window size and note any performance change. If there is no increase in performance, return the value to 8.

Letting your computer sit idle is a waste of resources. Without an adequate pacing window configured for your APPC sessions, both the client and server will wait for data acknowledgements to travel through the network. Large pacing windows help eliminate send wait behavior in data transfers. Data traveling from one machine to another can be sent continually while that machine receives acknowledgements, thus using network capacity very effectively.

There are two pacing techniques in SNA: fixed and adaptive pacing. Fixed pacing was the original pacing scheme used in SNA. Adaptive pacing is a more advanced scheme. Products that support adaptive pacing can perform fixed pacing when communicating with a machine that only supports fixed pacing. The decision on the type of pacing used on sessions is automatically negotiated between two machines; the user does not have to configure the type of pacing two machines will use.

Fixed pacing: If the two communicating machines use fixed pacing, ensure a large pacing value is configured. This may require a little planning for memory consumption based on the buffers needed to support the window.

A word of caution about fixed pacing and memory consumption: Pacing helps performance to a point, but has negligible effect when increased beyond that point. Since every network behaves differently, some experimentation will be required to determine at what value to stop increasing the window size. It is important not to specify a value that is any larger than necessary, as some buffers set aside for pacing will never be used.

Adaptive pacing: Adaptive pacing algorithms use memory availability to determine the best window size. Memory availability in the receiving machine will limit the size of the pacing window. Therefore, it is very efficient, since adaptive pacing algorithms are designed to use only what memory is available at the time of data transfer. If little memory is available due to some other system activity at that time, few buffers will be available. If many buffers are free, they will be used to increase throughput.

In short, adaptive pacing will never squander storage because the window size changes exactly as the amount of available memory changes. For machines using adaptive pacing, ensure the configuration allows for enough buffer space for a window of 8 RUs bytes or more. This may require looking at buffer availability statistics if memory shortage is suspected in slowing down adaptive pacing.

It is also important with adaptive pacing to realize that physical memory and available buffer constraints along the path between communicating LUs affect the size that the adaptive pacing window can be increased to.

Send and receive pacing: Both of the above pacing techniques allow for different size windows based on the direction of data flow. So, while a machine may be able to send 8 frames before requiring an acknowledgement, its partner may require an acknowledgement for each frame it sends. So when increasing pacing window sizes, be sure that both send and receive windows are increased. This will ensure consistent response times for sending and receiving data.

Example

MachineA uses fixed pacing, and is using a send window of 8 and a receive window of 63:

- When machineA is sending data, the send pacing window(8) governs number of RUs it can SEND before receiving a message saying it's OK to send more. MachineA can only send 8 RUs before it must receive an acknowledgement. If this acknowledgement is not received by machineA before the 8th RU is sent, it must stop sending data and wait for an acknowledgement.
- When machineA is receiving data, its RECEIVE pacing window governs how many RUs it can receive. The sending machine will only send 63 RUs unless MachineA sends a message stating it can receive more. It will send an acknowledgement for every 63rd RU received.

Mode Consistency and Propagation

Recommendation

Determine what modes your data transfer applications use. Then, make sure those modes are consistently defined and propagated across your entire network.

SNA modes are profiles which determine the characteristics of sessions. For example: the architected mode #BATCH is generally defined to have a large RU size, large pacing windows, and a medium to low data transmission priority. When a session is established as a #BATCH session, the session takes on those characteristics of the #BATCH profile.

To get consistent response times from your SNA network, session characteristics must be consistent throughout your network. They way you can achieve consistency is by paying close attention to the modes your applications use: make them consistent and ensure they are defined everywhere(propagated) in your network.

Consistency

The easiest way to ensure consistency in modes is to make your applications use one of five architected user modes on their ALLOCATE command:

- The "BLANK" mode
- #BATCH
- #BATCHSC

- #INTER
- #INTERSC

These modes ship with many SNA products so using them is an easy way to gain consistency.

If your applications use a specialized mode, be sure the mode is defined consistently in your network.

Propagation

After ensuring a mode's consistency, you must thoroughly propagate the definition in your network. This means you must define the mode in all of the products that will be communicating using the mode.

The cost of inconsistency

With a mix of LUs sharing the same modes, it is not always obvious where the mode definition governing session setup will be taken from. In some cases, it is taken from the definition for the bind initiating LU (the Initiator), and in other cases taken from the definition for the receiver of a bind (the Receiver). So a bind sent from B to A might have differing characteristics from one sent from A to B.

The following example is a result of mode inconsistency and propagation. These circumstances would lead to a widely varying response time from the same application doing the same operation. This tends to make users upset.

Example

1. A PC running Communications Manager/2 (CM/2) is configured so the mode, #BATCH, is to have a session limit of 4, and the minimum number contention winner sessions is set to 2.
2. CICS's session definition for the CM/2 station specifies AUTOCONNECT(ALL).
3. The parameters in CICS, VTAM, NCP and CM/2 are such that if CICS is the Initiator in a session with mode #BATCH, the RU size is negotiated to 256 bytes and if CICS is the Receiver, the RU is 8192 bytes.
4. When the first(SNASVCMG) session is established between CM/2 and CICS, all sessions are bound. This leaves CM/2 with two Conwinner sessions(sessions 1 and 2) and two Conloser sessions(sessions 3 and 4).
5. A little later on, both sessions which have 8192 byte RUs are in use when a file transfer application starts a conversation from CM/2 to CICS.
6. Since two sessions are available, the session chosen will be one with a 256 byte RU, and the file transfer happens at a snail's pace.
7. Later on, all sessions are free, and the file transfer application allocates a conversation to CICS.
8. Since a Conwinner session is free, it is selected. Since the RU size for this session is 8196 bytes, the file transfer completes much quicker than the previous one.

In the above example, CM/2's Conwinner and Conloser sessions had such widely different RU sizes because of mode consistency and propagation problems.

VTAM's definition for CICS specified a mode table where there was no #BATCH entry(a propagation problem). Because there was no #BATCH entry available used default values to set up sessions 3 and 4. These default values(like RU size = 256 bytes) were a lot smaller than what CM 2 asked for(a consistency problem).

Data Link Control(DLC) Recommendations

The DLC used to connect SNA protocols to physical networks will affect user response time. In this section, I describe the issues you must be aware of when tuning SNA to run on your Token Ring.

DLC adapter card selection

Recommendation

When possible (i.e. affordable), upgrade to the highest capacity DLC adapter card.

An adapter is an installable circuit board which allows your computer to attach to, and communicate over, data networks. It provides the physical and the logical connection between your PC and a network transport facility (i.e. X.25 networks). These boards generally slip into one of your PC's expansion slots.

Evolution in DLC adapter cards has made their selection a performance issue. New hardware architectures have implemented DLC protocols that can achieve throughputs near the speed of the media they transmit data over. All data either send from, or received by, your PC must go through the adapter. Therefore, any tuning performed on an adapter will lead to more efficient use of the transport network by the upper layers of SNA.

Later, in "Platform Specific Recommendations" on page 27, I include a ranking of different Token Ring cards based on available test data.

DLC frame size

Recommendation

Send the largest possible DLC frame size allowed for a given link. This will require that the partner machine can receive such a large frame. If two machines have two different frame sizes configured, the lesser of the two sizes will be used.

Exception

Noisy (error prone) links.

When configurable, increasing the packet size that a DLC uses to transmit data strongly increases data throughput for large data transfers. This is true regardless of the transport(e.g SDLC, Token Ring):

- The more user data sent per packet, the more efficient the transport becomes due to the increase in the ratio of user data to control data.

- The greater the number of bytes that can be transmitted in one packet, the less times that the sending machine has to contend for bandwidth on the transmission medium.
- A small DLC frame size can force unnecessary RU segmentation(see "Segmenting RUs" on page 19). Also, depending on the SNA implementation, a small DLC frame size can limit the size of RUs that can be sent over a link (see example below).

Example

A link, LINK1, is defined with a maximum frame size of 1024 bytes. For the link protocol, the link header and trailer total 44 bytes. All sessions going over LINK1 might have a limit on their RU size of 1024-44=980 bytes.

DLC pacing

Recommendation

Use the maximum allowable DLC pacing window your SNA product will allow. links.

Exceptions

- *For links connecting CM/2 to NS/DOS, a window size of 2 is optimal.*
- *It is only appropriate to set a large DLC pacing window for reliable transports. Unreliable transports (those with high or widely fluctuating bit error rates) can penalize or negate the benefits of a large window. It is intuitive: an unreliable link increased the possibility of data retransmission. The more data that a sending machine has outstanding on a poor link, the more likely that large retransmissions will have to occur.*

Similar to RU pacing, DLC pacing controls the number of DLC frames a machine can send before stopping data transmission and waiting to receive a link level acknowledgement. If a small window is specified, data transfer may intermittently stop in order to wait for acknowledgement for data packets previously sent. Excessive waiting for acknowledgements can be avoided in most cases by specifying a large DLC pacing window. A more effective use of bandwidth is thus achieved.

DLC pacing windows on a particular machine are different depending on the direction of data flow. So, like with RU pacing windows, remember to increase both the send and receive windows when tuning your SNA product.

In most products, increasing the window size leads to better throughput.

Trade offs

In SNA, the DLC is the facility that guarantees the safe delivery of data. This means data which a DLC has sent, but not received an acknowledgement for, must be kept in memory. This stored data will only be discarded when a message is received that a copy of it has been received safely. The data receiver must also guarantee that it has set aside enough memory to receive all of this data. For these reasons, increasing the window sizes increases the memory requirement to run SNA.

Efficient memory usage is most important in small machines. The memory consumed by DLC pacing windows will limit the number of sessions and links the SNA product can support. In order to balance better performance with efficient memory usage, a maximized pacing window may not always be necessary:

- If the remote machine is on the same physical LAN segment, acknowledgements will flow back to the sender quickly. In this environment, a small DLC pacing value may suffice since very few frames would ever be sent before being acknowledged.
- If the communicating machines are further apart, perhaps connecting over a few miles of bridged LANs or hundreds of miles of SDLC connections, then a larger window will be needed.

In general, the larger the distance and/or the greater the number of devices that separate two machines, the larger the DLC pacing window should be.

Platform Specific Recommendations

Communications Manager/2 (CM/2) tuning recommendations

The way users make changes to CM/2 is through the **CMSETUP** application. This can be found in the *Communications Manager Folder* on the OS/2 desktop.

Users of Extended Services(ES) must use the **Advanced** pull down menu to make configuration changes.

Increasing RU Size

CM/2 has two settings for RU size, and each has its merits.

Default RU size will set the RU size exactly large enough to fit in one DLC frame. When a large DLC frame is used, this may suffice for small data transfers.

Maximum RU size will allow the RU size to become as large as 16,384 bytes. This size RU will fit in one frame for a 16 megabit per second Token Ring, and cause segmentation to occur for DLC's which support less than a 16,393 byte frame. Regardless of the maximum frame size that the underlying transport supports, increasing the RU size will increase the efficiency of bulk data transfer.

Recommendation

Use 16,384 for your RU size.

Session pacing window value

CM/2 allows for a receive pacing window of up to 63 RUs.

CM/2's send window will be determined, based on availability of buffers at the machines the data travels through, while the data transfer is in progress. See Appendix C, "Determining pacing window sizes" on page 63 for instructions on how to tell what size pacing window Communications Manager is using.

Recommendation

Always choose 63.

Personal Computer(PC) LAN adapter cards

Following is a listing, in order of most to least capability, of the PC LAN adapter architectures:

- Microchannel cards
  1. *LAN Streamer(32 bit) adapter and *LAN Streamer(16 bit) adapter are the best for OS/2 desktops, servers and local bridges.
  2. IBM Token Ring Network 16/4 Adapter/A "Shorty" adapters are best for remote bridges for DOS desktops because of their low memory requirement.

3. Turbo Adapter
4. Standard adapter

Note: *

Machines with Lan Streamer cards can get additional performance increases from upgrades to their CPUs. With LAN streamer cards, the central CPU perfroms most of the DLC function.

- Eisa bus cards
  1. For OS/2 or other multitasking operating systems, the IBM 16/4 Busmaster EISA card is the best performer.
  2. For the DOS operating system and bridging devices, the IBM Token Ring Network **ISA-16** is best.
- Isa BUS(AT) cards
  1. For OS/2 or other multitasking operating systems, the IBM Token Ring Adapter II is the best performer.
  2. For the DOS operating system and bridging devices, the IBM Token Ring Network ISA-16 is best.

Recommendation

When possible (i.e. affordable), upgrade to the highest capacity DLC adapter card.

DLC frame size

CM/2 allows the user to configure a DLC frame size for most of the data transports it supports.

Recommendation

Maximize the DLC frame size. This will require that the partner machine is also configured to receive such a large frame.

DLC pacing

Most, but not all, DLC types that CM/2 supports have a configurable pacing window.

Recommendation

For those DLCs which allow a configurable pacing window, maximize the window. If memory shortages appear, consider lowering the size of the window to free up some memory (see "Trade offs" on page 24 for a discussion of memory and window size trade offs).

Networking Services/DOS (NS/DOS) tuning recommendations

RU size

RU sizes in NS/DOS are limited by the DLC frame size. Only one RU per frame is allowed.

Recommendation

Code '' for RU size in your NS/DOS MODE.NSD file. This will size the RU optimally inside of one DLC frame.*

RU pacing window size

Recommendation

Maximize the RU send pacing window and the RU receive pacing window. If memory shortages eventually appear, consider lowering the RECEIVE pacing window to free some storage. Lowering the SEND RU pacing window size won't save any storage.

NS/DOS doesn't allow users to set a *send* RU pacing window size on a mode definition, only a *receive* pacing window. The *send* window size will be set during session negotiation with the partner. This value can be found in the mode definition, in the partner computer, for the mode you are using for your bulk data transfer sessions.

With fixed pacing, memory may be wasted. Unfortunately, the level of fixed pacing that is appropriate will be dependent on where the destination machine is, and therefore how quickly acknowledgements can make it back to the DOS machine. See "Trade offs" on page 19 for guidance on this issue.

PC LAN adapter cards

Recommendation

When possible (i.e. affordable), upgrade to the highest capacity DLC adapter card.

Following is a listing, in order of most to least capability, of the PC LAN adapter architectures:

- Microchannel cards
  1. *LAN Streamer(32 bit) adapter and *LAN Streamer(16 bit) adapter are the best for OS/2 desktops, servers and local bridges.
  2. IBM Token Ring Network 16/4 Adapter/A "Shorty" adapters are best for remote bridges for DOS desktops because of their low memory requirement.
  3. Turbo Adapter
  4. Standard adapter

Note: *

Machines with Lan Streamer cards can get additional performance increases from upgrades to their CPUs. With LAN streamer cards, the central CPU perfroms most of the DLC function.

- Eisa bus cards
  1. For OS/2 or other multitasking operating systems, the IBM 16/4 **Busmaster** EISA card is the best performer.
  2. For the DOS operating system and bridging devices, the IBM Token **Ring** Network **ISA-16** is best.
- Isa BUS(AT) cards
  1. For OS/2 or other multitasking operating systems, the IBM Token Ring Adapter II is the best performer.
  2. For the DOS operating system and bridging devices, the IBM Token Ring Network ISA-16 is best.

DLC frame size

Recommendation

Maximize your DLC frame size. This is done via the TRMF statement in the CONFIG.NSD file. If you are using Token Ring, apply PTF# IC06030 to enable NS/DOS to support a 4464 byte frame. The machine that NS/DOS is sending data to will then have to be configured to receive frames this large. If memory shortages appear, consider lowering the size of the window to free up some memory (see "Trade offs" on page 24 for a discussion of memory and window size trade offs).

The frame size for the DLC that data traffic is transported on should be as large as possible, whether it is Ethernet, Token Ring, SDLC, etc. In NS/DOS, the maximum RU size is directly affected by the DLC frame size which sessions are flowing over.

For Token Ring support in NS/DOS 1.0, there was a bug that was recently fixed which limited maximum I-frame size to 2048 bytes even if a value of 4464 bytes was specified. The PTF number for the fix is IC06030.

DLC pacing

Recommendation

If you are configuring a NS/DOS station to talk with CM/2, use 2 for the DLC send and receive pacing windows. Otherwise, maximize the DLC pacing window sizes to 8. If memory shortages appear, consider lowering the size of the window to free up some memory (see "Trade offs" on page 24 for a discussion of memory and window size trade offs).

The MAXIN and MAXOUT parameters tell NS/DOS what values to use for receive and send DLC pacing, respectively.

The MAXIN and MAXOUT parameters are specified on the TRLD statement in the CONFIG.NSD file.

SNA Services/6000 v1.2 tuning recommendations

The parameters affecting SNA Services/6000 are set via the System Management Interface Tool(SMIT). You can skip directly to the SNA panels by entering "smit sna".

DLC frame size

Recommendation

Use the SYSTEM_DEFINED setting.

The SYSTEM_DEFINED option will optimize the DLC frame size based on the characteristics of the DLC transport.

DLC pacing

Recommendation

Set the send and receive values to 127.

During my tests, I discovered a problem with the Token Ring driver in SNA Services 6000. The driver shipped with AIX version 3.2.4 had a problem when connecting to NCP V6R2. The symptoms of the problem is:

- When transferring data via an APPC session, and using a DLC send pacing window greater than 10, the link connecting SNA Services 6000 to NCP fails.

I applied APAR IX41022 to fix the problem.

127 frames is the maximum window size configurable in SNA Services.

LAN adapter queue depth

Recommendation

Set the send and receive queue depths to 150.

RU size

Recommendation

Set the RU size to 3840, the maximum for SNA Services/6000 v1.2.

RU pacing window size

Recommendation

Set the send pacing window to 127.

Optimal memory buffer settings

AIX allows the user to do memory management for some AIX subsystems. SNA is one of those subsystems.

Recommendation

Use the following commands to increase the amount of memory SNA Services/6000 can use:

- ***no -o thewall = 8192***
- ***no -o lowclust = 256***
- ***no -o lowmbuf = 256***
- ***no -o mb_cl_hiwat = 512***

There are four buffer threshold values the user can set which control how much memory SNA Services/6000 can use to send and receive data:

1. "mb_cl_hiwat"

   The maximum number of free clusters SNA Services/6000 will allow in its pool of buffers before it starts returning clusters to the AIX system. Clusters are 4 Kbytes long.

2. "lowclust"

   The smallest number of free clusters SNA Services/6000 will allow in its pool of buffers. If the number of free clusters in SNA Services/6000' space dips below this value, SNA Services/6000 requests more clusters from the AIX system.

3. "lowmbuf"

   Similar to "lowclust": the smallest number of free mbufs SNA Services/6000 will allow. mbufs are 256 bytes long.

4. "thewall"

   Controls the maximum amount of clusters and mbufs SNA Services/6000 can use. This number is in units of Kbytes. All of this memory can be paged.

Increasing mb_cl_hiwat and thewall will improve performance by decreasing the number of times SNA Services/6000 must request and free memory. Increasing lowclust, lwombuf, and thewall will give SNA Services/6000 more memory to work with. SNA Services/6000 can then better handle bursty and busy conditions in the network.

These increases in memory consumption must be weighed against the memory needs other applications and subsystems on your AIX system.

AS/400 tuning recommendations

You can change the parameters of AS/400 communications system through the AS/400 control language or through system panels. The AS/400 Communications: Advanced Peer-to-Peer Networking Guide will help with making necessary changes.

AS/400 performance tools

You should use AS/400 performance tools if available on your system:

- Performance Tools/400(5738-PT1)
  - Automatic System Tuning - available with the base operating system through the QPFRADJ system parameter(set to 1 for IBM standard settings, 2 for dynamic tuning).
  - Advisor
  - Specialized tools: Capacity planner, System Planner, Transaction report.
- Performance Investigator/400(PRPQ 5799-PRG)

RU size

For calculating RU size, OS/400 gives the user a lot of help. The *CALC value in the Mode Definition(MODD) provides optimal RU sizes.

Recommendation

Always use *CALC for the RU size.

RU pacing

Recommendation

Use 7 for both IN and OUT pacing in the Mode Definition(MODD).

LAN adapters(IOPs)

Recommendation

When possible (i.e. affordable), upgrade to the highest capacity DLC adapter card.

Different IOPs(adapters) have different performance characteristics. Some important issues are:

- High capability adapters have the potential to overrun lesser capable adapters. This has the effect of causing retransmissions and time outs on the links between disparate adapters. The AS/400 performance tools can help determine the frequency of time outs and retransmissions. If they are occurring frequently, configure the adapters to use LANMAXOUT=2 and LANACKFRQ=1.
- For interactive environments, the adapter utilization should be kept under 60% to guarantee consistent response times. But, for bulk data transfer environments, it is acceptable to exceed this threshold. The utilization of the adapter can be determined using the AS/400 performance tools.

- Many adapters can be attached to a single AS/400. It is important to distribute work evenly across these adapters for maximal performance. For help with task, see AS/400 Communications: Local Area Network Guide(SC41-0004).
- The more advanced adapters(2617 and 2619) have a microcode feature which assists the main processor in segmenting RU data (see "Segmenting RUs" on page 19 for a discussion of segmentation). This reduces the load on the main CPU and will be of great benefit in bulk data transfer when segmentation is likely to occur.

The adapters are listed here in order of most to least capability:

Token Ring adapters

1. Model 2619 - 16/4 Mbit/sec Network adapter/HP
2. Model 2626 - 16/4 Mbit/sec Network adapter/A
3. Model 2636 or Model 6130 IOP with 6134 adapter - 16/4 Mbit/sec Token Ring
4. Model 6160 or Model 6130 IOP with 6034 adapter - 4 Mbit/sec Token Ring

Ethernet Lan adapters

1. Model 2617 IOP Ethernet(IEEE 802.3) Adapter HP
2. Model 2625 IOP or Model 6130 OPP with 2635 adapter - Ethernet subsystem

DLC Frame size: MAXFRAME parameter

OS/400 allows administrators to increase the size of the DLC frame for all of its supported DLCs. The default frame sizes for these are not optimal.

Recommendation

Maximize the DLC frame size

- *For Ethernet(ELAN) Token Ring(TRLAN) SDLC and ISDN, the frame size is changed only when MAXFRAME is changed in both the Line Description(LIND) and Controller Description(CTLD)*
- *For X.25 transport, change the DFTPKTSIZE in the LIND.*

DLC pacing: LANMAXOUT and LANACKFRQ parameters

Internal IBM tests show that for most environments, the *CALC values for these values is optimal. *CALC sets LANACKFRQ = 1 and LANMAXOUT = 2. For **pure** bulk data transfer environments, however, increasing these is appropriate.

Recommendation

*Unless you are sure that a link is dedicated to bulk data transfer, choose *CALC for LANMAXOUT and LANACKFRQ.*

*For links dedicated to bulk data transfer, increase them to LANACKFRQ = 2 and LANMAXOUT = 4 and note any change. If no improvement results from this change, return the values to *CALC.*

NCP tuning recommendations

Make changes to NCP parameters by changing NCP definition statements and Generating a new executable NCP module. For assistance with NCP Generation use the NCP, SSP, and EP Generation and Loading Guide.

RU Size: MAXBFRU and UNITSZ parameters

Recommendation

For each NCP, do the following:

1. *Set UNITSZ to 4000 bytes.*
2. *Determine the size of the largest PIU which NCP will have to transfer. This will be slightly smaller than the size of the DLC frames that NCP receives from other machines. See "Basic concepts" on page 13 for details on PIUs.*
3. *Then divide that PIU size by the UNITSZ value, round up to the nearest integer. That will be the best MAXBFRU value.*

   Note:

   You may have to increase this MAXBFRU value by 1 if your sessions are failing. In my tests, I sent 4000 byte frames from CM/2 to VTAM and a MAXBFRU = 2 was required to avoid session failures.
4. *Then, make sure that the following parameters follow this rule:*

   MAXDATA(VTAM, NCP) > UNITSZ*MAXBFRU > PIU size

The number of buffers set aside in NCP to receive an RU is specified by the parameter MAXBFRU. Each of these buffers is UNITSZ in length. As NCP receives data, it fills these buffers up, waiting for all data to arrive. If too many buffers are specified per RU, there are two adverse side effects:

- Unused buffers are wasted
- NCP will wait unnecessarily. NCP waits until all of the buffers specified by the MAXBFRU parameter are full before forwarding the data. Eventually, if those buffers don't fill, a timer expires and the buffers are forwarded. This is inefficient behavior.

The size of these buffers, UNITSZ, also affects performance. Large internal buffers allow NCP to transfer bulk data more efficiently than small buffers.

BFRS

NCP main memory buffer sizes can be altered. BFRS defines the length of these buffers.

Recommendation

Use 240, the maximum value allowed.

SLODOWN

NCP will enter a protective mode called slow down when it is in danger of running out of storage. You can tell NCP different thresholds at which to enter slow down via the SLODOWN parameter. The number specified is a percentage; so a value 6 says to NCP enter slow down when you only have 6% of your NCP's memory free for new data.

Recommendation

Use a value of 6. This will allow NCP to conserve 6% of its buffers when network traffic is heaviest.

DELAY

The DELAY option allows you to choose a timer value which tells VTAM how long to hold FM data(user data) before sending it to NCP. This timer strongly influences the rate at which pacing acknowledgements and confirmation messages get back to NCP attached machines. If the DELAY value is too high, pacing mechanisms will be disrupted.

Recommendation

Set DELAY to 0.

NCP Token Ring lan adapters

Different NCP Token Ring adapters have different capabilities. They are listed here in best to worst order:

1. TIC2 - 64K bytes of storage 16 or 4 Mbits/sec
2. TIC1 - 2576 bytes of storage, 4 Mbit/sec speed only

DLC frame size

RCVBUFC

This parameter controls the size of buffers used to receive DLC frames inbound to NCP.

RCVBUFC must be large enough to receive two whole DLC frames from a peripheral station. If this value is too small, NCP may have to ask an end station to retransmit data. Also, connection failures may occur if an end station sends a frame which is too large for NCP to receive in one buffer.

Recommendation

Make RCVBUFC at least twice as large as MAXTSL(see below). The maximum values are:

- ***4,095 bytes for TIC1 adapters.***
- ***32,000 bytes for TIC2 adapters.***

MAXTSL

This parameter affects the size of Token Ring DLC frames that NCP can send to, and receive from, peripheral Token Ring stations.

Recommendation

Follow these guidelines based on the adapter type in your NCP:

- ***For TIC1 adapters, never exceed 1108 bytes.***
- ***For TIC2 adapters running at 4M bits/sec, use 4060 bytes.***
- ***For TIC2 adapters running at 16M bits/sec, use 16,000 bytes.***

DLC Pacing

T2TIMER

The T2TIMER definition statement specifies three different values for use by the NCP. It appears as follows:

T2TIMER = (T1,T2,W)

- T1 is the acknowledgement time out value for local LAN connections
- T2 is the acknowledgement time out for remote connections(connections to a partner on a different physical LAN segment).
- W is the receive acknowledgement count.

When NCP receives DLC frames, it keeps a timer and a count active for each link. Either the timer expires, or the number of frames received equals W. In either case, an link level acknowledgement is sent for all frames received since the NCP last sent an acknowledgement. W is also called the receive pacing window size.

There is a double standard in the way that NCP treats the acknowledgement window (W) specified in the T2TIMER:

- For Subarea connections, the value W will be used as the acknowledgement window size.
- For peripheral connections, NCP uses only two window sizes: 1 and 2. If the T2TIMER statement is specified for a peripheral connection, NCP uses a value of 2. If the T2TIMER operand is not specified, a value of one is used.

Recommendation

- *Use the T2TIMER operand on LINE and BUILD statements defining peripheral connections to machines which support a SEND window of greater than 1. Code T2TIMER = (1,1,2).*
- *If NCP is connected to a lot of peripheral stations that don't support a SEND window of greater than 1, then either use a local timeout (T1) of less that 200 milliseconds(< 0.002) or don't specify T2TIMER at all. This will keep the end stations from waiting too long for NCP acknowledgements.*
- *For subarea connections, use large values for the parameters specified by T2TIMER. Experimentation to find the proper window size is necessary because, as with any fixed pacing scheme, raising the window above a certain level offers no pay back and will waste storage. Try starting with large values like T2TIMER = (1,3,16). Then, note any changes in performance(not an easy task, I admit!) when you change to (1,3,32), (1,3,64) and (1,3,127).*

MAXOUT

This parameter specifies the number of DLC frames NCP can send before requiring a DLC acknowledgement. The proper number for this parameter must be determined by the capacity of the receiving machine.

Recommendation

Set MAXOUT as large as possible based on the type of link you are setting it for (see below).

VTAM tuning recommendations

To make changes to VTAM, you must make changes to VTAM's definition members(files). Help with this task can be found in the VTAM Resource Definition Reference.

RU size

RU sizes are coded in a unique way in VTAM. The actual value is specified as a base and mantissa in the **RUSIZES** statement in the **MODEENT** macroinstruction. This macroinstruction is part of a mode table which VTAM uses to read mode information.

The value **RUSIZES = X'8787'** would mean that both SLU to PLU and PLU to SLU RU sizes for a particular mode would be **8*2**7** or 1024 bytes.

Recommendation.

VTAM supports up to a 64k byte RU size. Use RUSIZES=X'8D8D' *for the modes which your bulk data transfer applications use to transfer data. The name of this mode table can be found in the APPL statement for the VTAM application (see example below to help determine what your VTAM application is).*

Example of a VTAM application

A VTAM application is any system resource that opens an Access Control Block with VTAM. If your host application uses CICS for its CPIC Application Programming Interface (API), then CICS is your VTAM application. Likewise, APPC/MVS provides a CPIC API for host applications and could be your VTAM application.

APPC/MVS was one of my VTAM applications for my tests tests. The following is the definition I used.

TOMVSAP APPL ACBNAME=TOMVSAP,APPC=YES...

MODETAB=tunemode,...,VPACING=32,...
DLOGMOD=#INTER...

Any changes I made to #BATCH (the mode I used for bulk data transfer tests) were made to the mode table *tunemode*. More information on changing a mode table is available in Appendix D, "Replacing a mode table in VTAM" on page 65.

IOBUF

The IOBUF operand tells VTAM what size to make the buffers it uses to send and receive data from channel attached controllers (e.g. 3172, 3174, 3745). The larger the buffer size, the less processing that is required to receive and send large amounts of data.

Recommendation

Set IOBUF to 4000 bytes.

Matching parameters across platforms for better performance

VTAM with CICS 3.3

SENDSIZE and RECEIVESIZE

These parameters control the data buffer size that CICS uses when exchanging application data with VTAM. They can range from 1 to 30,720 bytes. CICS can limit RU size negotiation with the SENDSIZE and RECEIVESIZE parameters. These are coded in the connection definitions for end stations.

Recommendation

Code both SENDSIZE and RECEIVESIZE large enough to accommodate the RU size that end stations want to negotiate.

Automatic session setup

The AUTOCONNECT option on a session definition in CICS, and the AUTOSES option on the APPL statement in VTAM, can have a side effect that can lead to performance problems.

AUTOCONNECT(ALL) and AUTOSES are functions that can be used to bind as many sessions as a mode's session limit when a LU first contacts another. When VTAM binds all of the sessions for a particular mode, it might not have all of the information necessary to make the Conwinner and Conloser sessions equivalent. VTAM, NCP and CICS offer so many different options that might lead to the RU sizes in Conwinner and Conloser sessions, between a host and another machine, to be very different. Depending on the session selected for a conversation, bulk data throughput will differ.

Recommendation

For CICS, make the SENDSIZE and RECEIVESIZE equal for any end station which will perform bulk data transfer with CICS. See "SENDSIZE and RECEIVESIZE" for further explanation on SENDSIZE and RECEIVESIZE.

For end stations connecting to VTAM or NCP follow the recommendations in "Mode Consistency and Propagation" on page 21.

For VTAM/NCP, make sure the following parameters match up:

- *MAXDATA > = MAXTSL*
- *RCVBUFC = 2 x MAXTSL*
- *IOBUF size(VTAM start option) = UNITSZ(NCP) = 4000*
- *For any bulk data transfer Mode, make sure the RUSIZES on the MODE definition has the same value for all RUs(e.g. RUSIZES = x'ABAB' and not RUSIZES = x'ABCD').*

When coding VTAM definitions:

- *Define the VTAM application(like CICS or APPC/MVS) and the PU connecting to VTAM(like NS/DOS or SNA Services/6000) with the SAME mode table.*

Appendix A. Test lab results - microcomputer and mainframe network

The above platform specific recommendations came from lab work performed at IBM.

Methodology

At the time of print of this publication, only single APPC session throughput was investigated. Further investigations to determine the best parameters for multiple session throughput, and data transfer from the host to desktop machines, may be done in the future.

The benchmarks

The details of the design of the File Transfer benchmark used for the tests can be found in "Classic Client-Server transactions using APPC" by Dr. John Walker and Lance Bader(1).

Memory to memory data transfer

The goal of the tests conducted at IBM was to determine how to tune the SNA communications stack most effectively. For that reason, a memory to memory CPIC benchmark was selected which simulates an efficient file transfer operation. The measurement recorded was total throughput achievable by the SNA transport. I O overhead and operating system calls to retrieve data were, therefore, not part of the data transfer measurement. The overhead required for allocating APPC conversations and, or performing non communications processing is beyond the scope of this tutorial. See Appendix E, "Additional performance related publications" on page 67 for additional performance source material. For that reason, the maximum throughputs reported here will not be representative of what you can achieve in a real data transfer environment where disk access is incurred. I report the maximum numbers and gains here to give you a frame of reference as to what is attainable in environments similar to my test networks.

What was measured

Conversation allocate(CMALLC) and deallocate(CMDEAL) verbs are not included in the measurement times in this benchmark. Only the time required to issue all of the CMSENDs to send all of the data and a CMRCV to receive a confirm at the end of the transfer was recorded.

Many of these transactions were performed and an average time was derived for each data transfer transaction.

The network

Microcomputer and Mainframe test network

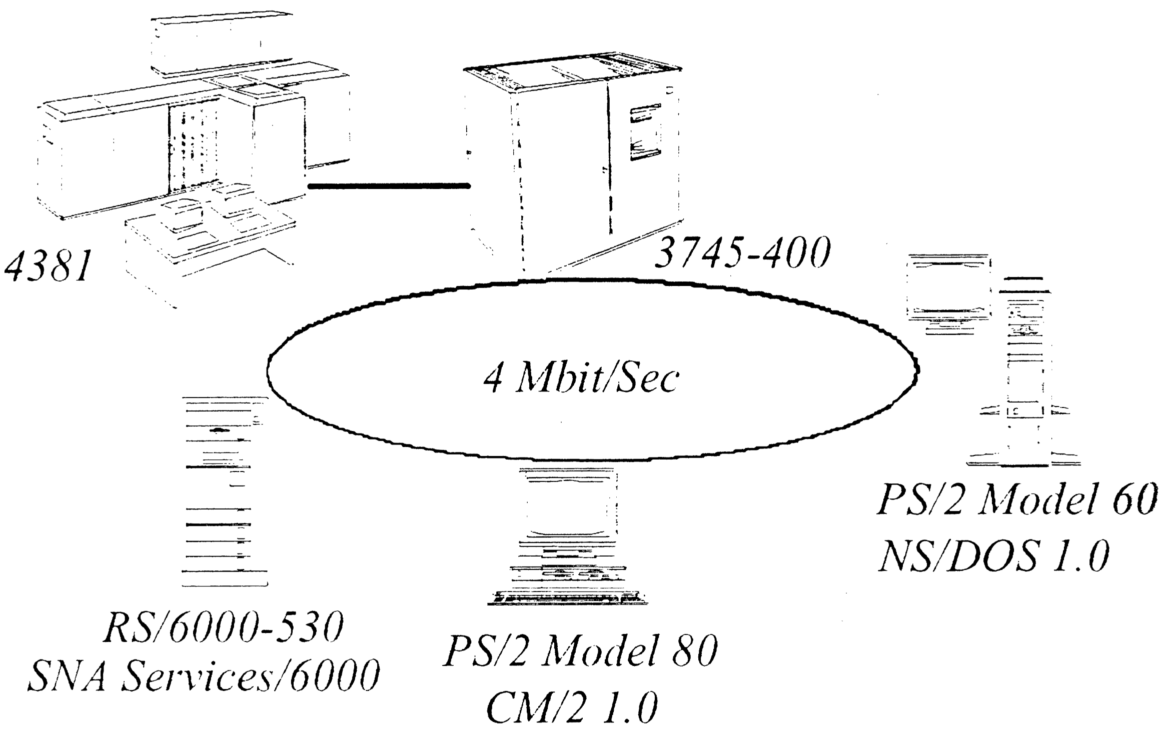

Figure 5. Microcomputer Mainframe network

Highlights

Before tuning was performed on the network, a baseline set of runs were made. The state of the VTAM NCP Composite network node used as a server was that of a typical production machine with many differing user demands. Many upgrades and changes had been made to VTAM and NCP over time. The end station products used were loaded fresh out of the shrink wrap and the shipped defaults were used for the baseline run. For the baseline run, the following was achieved:

- Maximum throughput achieved for NS DOS to VTAM was

  a 100,000 byte file transferred in 31 seconds = **0.0258 Mbit/sec.**

- Maximum throughput achieved for OS 2 to VTAM was

  a 1,000,000 byte file transferred in 6.4 seconds = **1.25 Mbit/sec.**

- Maximum throughput achieved for RS to VTAM was

  1,000,000 byte file transferred in 8.66 seconds = **0.924 Mbit/sec.**

Tuning was then performed on the machines in the network using all known tuning techniques, with the following results:

- Maximum throughput achieved for NS/DOS to VTAM was

  a 100,000 byte file transferred in .53 seconds = **1.51 Mbit/sec, a 98% reduction in transfer time relative to the base run.**

  Note: Paying close attention to the MAXBFRU and DELAY parameters in NCP opened up NS/DOS's throughput immensely.

- Maximum throughput achieved for OS/2 to VTAM was

  a 1,000,000 byte file transferred in 5.1 seconds = **1.57 Mbit/sec, a 20% reduction in transfer time relative to the base run.**

  Note: RU size and DLC pacing had the most effect on OS/2 performance

- Maximum throughput achieved for RS to VTAM was

  a 1,000,000 byte file transferred in 4.74 seconds = **1.69 Mbit/sec, a 45% reduction in transfer time relative to the base run.**

- Using either CICS or APPC/MVS provided nearly equivalent results for all client machines.

Details of the data transfer and hardware used

In the tests conducted using the microcomputer and mainframe network, data transfer rates were measured for data sent from the small machines to the MVS host.

To scale the data transfer sizes to the machines' capabilities, two "file sizes" were used for the benchmark:

1. 100,000 bytes for sending from NS/DOS to VTAM
2. 1,000,000 bytes for CM/2 to VTAM and SNA Services/6000 to VTAM VTAM.

The machines used in the network were:

IBM 4381

- Memory requirement was less that 1 Megabyte so storage size was not a factor
- MVS 4.2.2
- APPC/MVS provided with MVS operating system
- CICS 3.30
- VTAM 4.1

IBM 3745-410

- 8 Megabytes RAM
- Type6 3 Megabyte block channel adapter
- TIC2 Token Ring adapter set for 4 Mbit/sec

RS/6000 Power station 530

- SNA Services/6000 v1.2
- AIX 3.2.4

- 32 Megabytes main storage

PS/2 Model 80

- Communications manager/2 1.0
- Intel 80386 20 Megahertz processor
- 12 Megabytes of memory
- IBM 16/4 Mb Token Ring Adapter/A set for 4 Mbit/sec
- OS/2 2.1
- 32 bit Microchannel bus

PS/2 Model 60

- Networking Services/DOS 1.0
- DOS 6.0
- Intel 80286 10 Megahertz processor
- IBM 16/4 Mb Token Ring Adapter set for 4 Mbit/sec
- 7 Megabytes of memory
- 16 bit Microchannel bus

Base and tuned parameters

The following charts detail the changes I made to achieve the performance improvements listed above. In each chart, the **?CODED** column describes where each parameter is coded(configured) in the software products used.

Base parameters: When I made the baseline performance runs, no tuning was done. Here's how the machines looked to start off.

Figure 6 (Page 1 of 2). Baseline parameter settings.

Parameter	Value	?CODED
CICS		
SENDSIZE	4096 bytes	Session definition
RECEIVESIZE	4096 bytes	Session definition
VTAM		
RUSIZES	x'F7F7'	MODEENT statement
PACING	*default*	LU statement for end station(e.g. DOS)
VPACING	0	PU statement for end station(e.g. DOS)
Mode PSNDPAC	3	MODEENT statement
Mode SSNDPAC	3	MODEENT statement
Mode SRCVPAC	3	MODEENT statement
IOBUF	(200.384,,,100,50)	VTAM start list
NCP		
MAXDATA	65535	PCCU statement
Channel VR pacing	(255,255)	PATH statement for VTAM/NCP Channel
MAXOUT	*default*	PU statement
MAXTSL	2044	LINE statement
RCVBUFC	4095	LINE statement
T2TIMER	*default*	BUILD statement
DELAY	*default*	PCCU statement
BFRS	240	PCCU statement
SLODOWN	50	BUILD statement
INBFRS	40	HOST statement
MAXBUFRU	40	HOST statement
UNITSZ	384	HOST statement
OS2		

Figure 6 (Page 2 of 2). Baseline parameter settings.

Parameter	**Value**	**?CODED**
RU size	*default*	CMSETUP - Modes
PACING	Receive pacing = 8	CMSETUP Modes panel
DLC Frame size	4000 bytes	CMSETUP DLC panel
DLC SEND PACING	3	CMSETUP DLC panel
DLC RECEIVE PACING	3	CMSETUP DLC panel
RS/6000		
Lan adapter Transmit Q size	30	SMIT tok() profile
Lan adapter Receive Q size	30	SMIT tok() profile
DLC Transmit Count	10	SMIT DLC logical link profile
DLC Receive Count	127	SMIT DLC logical link profile
DLC Window SIze	SYSTEM_ DEFINED	SMIT DLC logical link profile
lowclust	29	″no -o lowclust″
lowmbuf	88	″no -o lowmbuf″
thewall	2048	″no -o thewall″
mb_cl_hiwat	58	″no -o mb_cl_hiwat″
Mode RU size	2816	SMIT Mode profile
Mode Send Pacing	3	SMIT Mode profile
Mode Receive Pacing	3	SMIT Mode profile
DOS		
RU SIZE	* (960 negotiated)	MODE.NSD file
RU Receive Pacing	3	MODE.NSD file
DLC FRAME SIZE	987	CONFIG.NSD file
DLC MAXIN	2	TRLD statement - CONFIG.NSD file
DLC MAXOUT	2	TRLD statement - CONFIG.NSD file

Tuned parameters

Figure 7 (Page 1 of 4). Tuned network parameter settings.

Parameter	Value	?CODED
	CICS	
SENDSIZE	30,720 bytes	Session definition
RECEIVESIZE	30,720 bytes	Session definition
	VTAM	
RUSIZES	x'8D8D'	MODEENT statement
PACING	*default*	LU statement for end station(e.g. DOS)
VPACING	32	PU statement for end station(e.g. DOS)
Mode PSNDPAC	63	MODEENT statement
Mode SSNDPAC	63	MODEENT statement
Mode SRCVPAC	63	MODEENT statement
IOBUF	(200,4000,,,100,50)	VTAM start list
	NCP	
MAXDATA	65535	PCCU statement
Channel VR pacing	(255,255)	Channel PATH statement
MAXOUT	*default*	PU statement
MAXTSL	4060	LINE statement
RCVBUFC	16240	LINE statement
T2TIMER	(1,1,8)	BUILD statement
DELAY	0	PCCU statement
BFRS	240	PCCU statement
SLODOWN	6	BUILD statement
INBFRS	40	HOST statement
UNITSZ	4000	HOST statement
MAXBUFRU	2	HOST statement
	OS2	
RU size		CMSETUP - Modes
PACING	Receive pacing = 63	CMSETUP Modes panel
DLC Frame size	4000 bytes	CMSETUP DLC panel

Figure 7 (Page 2 of 4). Tuned network parameter settings.

Parameter	**Value**	**?CODED**
DLC SEND PACING	8	CMSETUP DLC panel
DLC RECEIVE PACING	8	CMSETUP DLC panel
RS/6000		
Lan adapter Transmit Q size	150	SMIT tok() profile
Lan adapter Receive Q size	150	SMIT tok() profile
DLC Transmit Count	127	SMIT DLC logical link profile
DLC Receive Count	127	SMIT DLC logical link profile
DLC Window SIze	SYSTEM_ DEFINED	SMIT DLC logical link profile
lowclust	256	"no -o lowclust"
lowmbuf	256	"no -o lowmbuf"
thewall	8192	"no -o thewall"
mb_cl_hiwat	512	"no -o mb_cl_hiwat"
Mode RU size	3840	SMIT Mode profile
Mode Send Pacing	63	SMIT Mode profile
Mode Receive Pacing	63	SMIT Mode profile
DOS		
RU SIZE	*	MODE.NSD file
RU Receive Pacing	8	MODE.NSD file
DLC FRAME SIZE	4100	CONFIG.NSD file
DLC MAXIN	8	TRLD statement - CONFIG.NSD file

Figure 7 (Page 3 of 4). Tuned network parameter settings.		
Parameter	**Value**	**?CODED**
DLC MAXOUT	8	TRLD statement - CONFIG.NSD file

Appendix B. "APPC Optimal Performance Configuration on OS/2 and DOS"

This document describes the APPC parameters that can be changed from the default values to optimize APPC performance on OS/2 and DOS. These findings are the results of performance testing conducted by APPC Market Enablement in Research Triangle Park, North Carolina.

The goal of the tests was to use standard transactions to determine optimal performance of APPC. The configuration was tuned independently for each transaction to provide the maximum number of transactions per second.

The results indicate that protocol performance is dependent on the size of the transaction records as well as the type of adapter being used.

This document presents the test methodology and the results of the performance tests.

Test Environment

Microcomputer test network

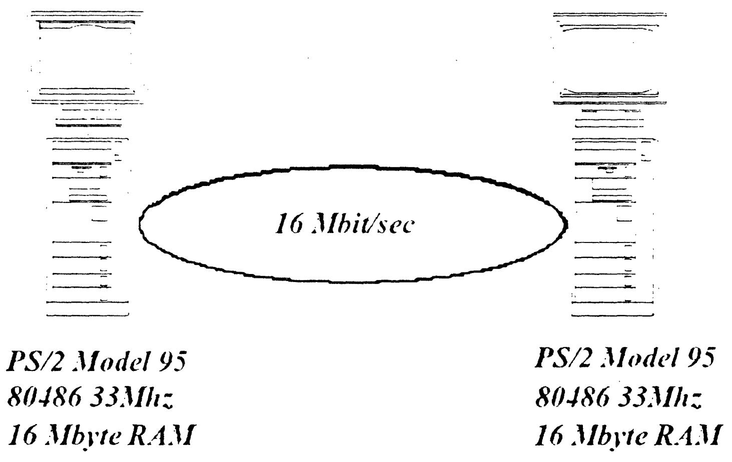

Figure 8. Microcomputer network

Highlights

Maximum throughput

CM/2 server CM/2 client Turbo adapter card: **8.89 Mbit/sec**

CM/2 server CM/2 client Busmaster adapter card: **10.91 Mbit/Sec**

NS/DOS server CM/2 client Turbo adapter card: **5.63 Mbit/Sec**

CM/2 server NS/DOS client Turbo adapter card: **3.86 Mbit/Sec**

Both the OS/2 and DOS tests were run on two 80486 33 megahertz (MHz) IBM Model 95s. Each machine had 16 megabytes (MB) of memory and was configured for 64 kilobytes (KB) of shared RAM on the token-ring card. The two machines were connected with a dedicated 16 megabit (Mb) token-ring.

OS/2

For the OS/2 tests, two token-ring cards were configured on each machine. IBM 16/4 Adapter A Turbo token-ring adapters and IBM Busmaster adapters were used. All performance measurements were obtained with the machines communicating with the same type of adapter. Separate measurements were obtained for communications using the Turbo cards and using the Busmaster cards.

DOS

For the DOS tests, only the Turbo adapters were used.

Software

OS/2

The OS/2 platforms were running OS/2 2.0 with Extended Services 1.0 without any available corrective services deliveries (CSDs). Extended Services provides the APPC protocol stack for OS/2. The OS/2 machines were configured as LEN Nodes for the APPC tests.

DOS

The DOS platforms were running DOS 5.0 with Networking Services/DOS, which provides APPC support.

Transactions

The transactions simulated for these tests were a 100,000-byte file transfer transaction and a 100 byte inquiry transaction. These are the two transactions that were selected by the SAA performance council for comparisons.

File Transfer Transaction

For the file transfer transaction the client sends 100,000 bytes to the server using 32,000-byte application-level buffers. The client receives one acknowledgement from the server when the transaction completes, indicating the data has arrived successfully. The measurements obtained were based on the average of 1,000 repetitions of this transaction.

Inquiry Transaction

The client sends 100 bytes to the server and the server returns 100 bytes to the client. 100-byte application level buffers were used. The measurements obtained were based on the average of 20,000 repetitions of this transaction.

Benchmarks

The following APPC benchmarks replicate the file transfer and inquiry transactions. To provide a true measure of the API transaction time, the benchmarks do not perform any disk access or memory copies.

The Bader Benchmarks, which are generally accepted for APPC client-server benchmarking, were used to obtain the APPC performance measurements.

OS/2

The OS/2 benchmarks used the APPC API with basic long-running conversations. The conversation is maintained after the transaction ends and can be re-used for the next repetition of the transaction. Long-running conversations eliminate the overhead of conversation set-up in the performance measurements.

To minimize the factors that affected performance on the OS/2 machines, the benchmarks were run in OS/2 full-screen mode without any other processes running in the background.

Both the client and the server were OS/2 machines.

DOS

Networking Services DOS provides the Common Programming Interface for Communications (CPI-C) in place of the APPC programming interface. Consequently, the CPI-C versions of the benchmarks were used for the DOS tests.

Networking Services/DOS cannot link directly with another DOS platform running Networking Services/DOS. The DOS performance benchmarks were run with the DOS machine communicating with an OS/2 machine for these tests. Measurements were obtained both with the OS/2 machine performing the server duties while communicating with a DOS client and with the OS/2 machine acting as a client with a DOS server.

Configuration for Optimal Performance

In order to tune the protocols for optimal performance, some default values of the configuration parameters were altered.

When large amounts of data are being transferred, APPC provides the best performance when most parameters in the configuration are set to their maximum value. These settings increase the size of the packets that are sent across the link and utilize the adaptive pacing provided by APPC.

The DLC RU size was the factor that most significantly impacted performance with large data transfers for both OS/2 and DOS. However, for OS/2, this is one parameter that should not be maximized. OS/2 APPC performance for the file transfer transaction improved as the RU was increased up to a point. The value chosen for the final performance measurements on OS/2 was a value in the middle of the range providing maximum performance. For DOS, this value should be maximized for optimal file transfer performance.

Although there were a few parameters that could be tuned to improve performance for the inquiry transaction, performance was generally not affected by variations of the parameter settings. Increasing the mode receive pacing window resulted in slight performance improvements on both the OS/2 and DOS platforms but all other parameters provided equal performance as had been provided using the default values. The inquiry transaction performed well when the parameters were increased to match the optimal performance configuration for the file transfer transaction. Optimizing APPC configuration for large data transfers also optimizes performance for smaller data transfers.

For APPC on DOS, the DLC send and receive window count was an important factor in performance tuning. Networking Services/DOS performs better when this value is decreased to 2 from the default value of 4.

Definition of Parameters

Following are descriptions of the parameters that were changed from the default values to provide optimal performance for each transaction.

Adapter Card Buffer Size
: Size of the buffers on the network adapter card. It is applicable to busmaster adapters only and is defined in the OS/2 Busmaster .NIF file.

Adapter Driver Receive Buffer Size
: Size of each receive buffer. It is applicable to busmaster adapters only and is defined in the OS/2 Busmaster .NIF file.

DLC Frame Size
: Physical frame size used by the LAN adapter for communications from your workstation. This parameter is defined in the Networking Services/DOS CONFIG.NSD file.

DLC Send/Receive Window Count
: Number of frames that the station can send before receiving an acknowledgement or receive before sending an acknowledgment. It is defined in the OS/2 APPC .CFG file

Max DLC RU
: Maximum request/response units supported by this adapter and defined in the OS/2 APPC .CFG file.

Mode Receive Pacing Window
: Protects the nodes on a session from flooding that can occur when one node sends data faster than the other node can receive and process it. This parameter is defined in the OS/2 APPC .NDF file.

Session Receive Pacing Window
: Used to determine the amount of storage required in the RU buffer pool. Pacing windows are negotiated between your workstation and the partner LU when the session is activated. This parameter is defined in the Networking Services/DOS MODE.NSD file.

Parameter Settings

Following are the values used for specific configuration parameters.

Notes:

- The first platform listed indicates the client. For example, OS/2-DOS indicates an OS/2 client communicating with a DOS server.
- Parameters that did not affect configuration are not listed in the following tables.
- Table entries containing default values indicate the default value provided performance that is better than or equal to all other settings.
- Table entry '--' indicates the parameter is not applicable in the test configuration.
- Total Improvement is a percentage improvement from the time obtained when the benchmarks were run with the default settings.

Table 1 - Tuned Values for APPC Parameters on OS/2

This chart indicates the values of configuration parameters that provide the best performance on an OS/2 machine. Each parameter is named in the first column with the default value for this parameter which is set at installation specified in the second column.

The following four columns (3-6) indicate the values to which these parameters should be set for optimal performance on an OS/2 machine when communicating with another OS/2 machine. In some cases, the parameter values vary depending on the adapter being used and/or the type of transaction being run. This chart breaks the number down accordingly to indicate these variances.

The last four columns (7-10) indicate the values to which these parameters should be set for optimal performance on an OS/2 machine when communicating with a DOS machine running Networking Services/DOS. In some cases, the values for these parameters vary depending on the type of transaction being run and whether the OS/2 machine is acting as a client or as a server. Columns seven (7) and eight (8) indicate the values for the parameters if the OS/2 machine is the client. Columns nine (9) and ten (10) indicate the values for the parameters if the OS/2 machine is the server. This chart breaks the number down accordingly to indicate these variances.

Figure 9. Tuned Values for APPC Parameters on OS 2

Param.	Default	OS/2-OS/2				OS/2-DOS		DOS-OS/2	
		Turbo		Busmaster					
		File Xfer	Inquiry	File Xfer	Inquiry	File Xfer	Inquiry	File Xfer	Inquiry
Mode Receive Pacing Window	4	63	24	63	4	63	24	63	24
Max DLC RU	1024	7168	1024	7168	1024	7168	1024	7168	1024
DLC Send Receive Window Count	4	8	4	8	4	2	2	2	2
Adapter Card Buffer Size	1032 (bus.)	--	--	2048	1032	--	--	--	--
Adapter Driver Receive Buffer Size	256 (Turbo), 2048 (bus.)	2040	256	2048	2048	2040	256	2040	256
Total Improvement	--	35%	5.8%	35%	0.0%	41%	57%	41%	57%

Table 2 - Tuned Values for APPC Parameters on DOS

This chart indicates the values of configuration parameters that provide the best performance on a DOS machine running Networking Services/DOS when communicating with an OS 2 machine running Extended Services.

Each parameter is named in the first column with the default value for this parameter which is set at installation specified in the second column. The following four columns (3-6) indicate the values to which these parameters should be set for optimal performance on a DOS machine running Networking Services/DOS when communicating with an OS 2 machine running Extended Services. In some cases, the values for these parameters vary depending on the type of transaction being run and whether the OS 2 machine is acting as a client or as a server. Columns three (3) and four (4) indicate the values for the parameters if the DOS machine is the server. Columns five (5) and six (6) indicate the values for the parameters if the DOS machine is the client. This chart breaks the number down accordingly to indicate these variances.

Figure 10. Tuned Values for APPC Parameters on DOS

Parameter	Default	OS/2-DOS		DOS-OS/2	
		File Xfer	Inquiry	File Xfer	Inquiry
Session Receive Pacing Window	3	63	16	63	16
DLC Frame Size	256	16393	1929	16393	1929
Total Improvement	--	14.3%	74.5%	14.3%	93.5%

Performance Results

OS/2

Following are the results of the OS/2 performance tests after tuning the configuration for optimal performance.

Figure 11. OS/2 APPC Performance Results (Inquiry)

Adapter Card	Seconds/Transaction	Transactions/Second	Effective Throughput
Turbo	.0049 secs	204.1	.327 Mb
Busmaster	.0050 secs	200.0	.320 Mb

Figure 12. OS/2 APPC Performance Results (File Transfer)

Adapter Card	Seconds/Transaction	Transactions/Second	Effective Throughput
Turbo	.0900 secs	11.111	8.89 Mb
Busmaster	.0733 secs	13.643	10.91 Mb

DOS

Following are the results of the DOS performance tests after tuning the configuration for optimal performance.

Figure 13. DOS APPC Performance Results (Inquiry)

	Seconds/Transaction	Transactions/Second	Effective Throughput
DOS Server	.0070 secs	142.9	.229 Mb
DOS Client	.0070 secs	142.9	.229 Mb

Figure 14. DOS APPC Performance Results (File Transfer)

	Seconds/Transaction	Transactions/Second	Effective Throughput
DOS Server	.1420 secs	7.04	5.63 Mb
DOS Client	.2070 secs	4.83	3.86 Mb

Conclusion

APPC's performance improves significantly when the default parameters are tuned for the file transfer transaction. Since, this configuration also provides optimal performance for the inquiry transaction, those seeking optimal performance should tune APPC configuration as specified in the previous charts.

Appendix C. Determining pacing window sizes

- SNA Services/6000

  The pacing calues used for a session can be found in the MODE profile that the session is using. You can access this mode profile by using SMIT.

- CM/2

  To know what size pacing send window CM/2 is using, a data transfer in progress must be traced. I recommend two ways of accomplishing this:

  1. Use the APPNPACE utility available from APPC Market Enablement APPNPACE should be available in mid March, 1994.
  2. You can check what size the window is opening up to by using APPNT(start APPN Trace), APPNF(Format APPN trace).
     a. Issue "APPNT" in an OS/2 command prompt.
     b. Run your data transfer.
     c. Issue "APPNF *filename*(no extension!)" in an OS/2 command prompt.

        Note:

        CM/2 creates three files at this time: *filename*.SUM, *filename*.DET, and *filename*.TRC
     d. Edit *filename*.SUM, and search for "IPM"

        The number appearing in the brackets of **IPM(#)** is the window the receiving machine is allowing CM/2 to send.

  To check CM/2's receive pacing window, you must look at a mode definition configuration panel. If you were using the mode #BATCH, for example, to transfer data over and you wanted to see the receive pacing window for #BATCH:

  1. For CM/2 use the CMSETUP application to look at mode definitions.
  2. For Extended Services, follow this series of menus: **ADVANCED- > CONFIGURATION- > SNA FEATURE PROFILES- > SNA FEATURE CONFIGURATION- > OTHER SNA FEATURES.**

- NS/DOS

  Use the "NSD LIST /SE" command. Among the data displayed will be the send and receive pacing window for each section.

Appendix D. Replacing a mode table in VTAM

A mode table must be compiled and reloaded into VTAM before any changes made to it take effect.

Compiling

See the VTAM Network Implementation Guide, IBM publication SC32-6419, for details.

Reloading

Use the MODIFY TABLES command to replace the current mode table with the new(most recently compiled) version. I used

```
MODIFY NET,
   TABLES,OLD = TUNEMODE, NEW = TUNEMODE,OPTION = LOAD
```

Where NET was the MVS procedure name(proc) that VTAM was started with.

Appendix E. Additional performance related publications

1. "Performance Considerations in an APPC/MVS Environment" - IBM publication GG66-3206.
2. "VTAM™ Performance Benchmark Study Results Version 4 Release 1 for MVS/ESA" - IBM Publication GC31-6415. IBM publication GG66-3206.
3. The AS/400 Communications: Advanced Peer-to-Peer Networking Guide IBM Publication SC31-8188.
4. NCP, SSP, and EP Generation and Loading Guide - IBM Publication SC30-3348
5. VTAM Resource Definition Reference - IBM Publication SC31-6427.

Appendix F. Bibliography

1. "Classic Client-Server transactions using APPC" - John Walker, Lance Bader. **IBM Personal Systems Developer** Spring 1991, pp.34-42.
2. "IBM AS/400 Programming: Version 2 Release 2 Performance Capabilities Reference" - Internal IBM publication.
3. "NCP version 5 Network Performance and Tuning" - IBM publication number GG24-3469-00.
4. "Start your Engines for Improved SNA performance" - Joe Mohen. **Data Communications** November 1991, pp.29-30.
5. "16 Mbps EISA Token-Ring Performance Test Report" - internal IBM publication.
6. "16 Mbps Token-Ring AT Bus Adapter Performance Test Report" - internal IBM publication.
7. "16 Mbps Token-Ring Performance Test Report" - internal IBM publication.
8. "AIX SNA Services/6000 LU 6.2 performance: Version 1" - internal IBM publication.