BUS NETWORK SIMULATION MODEL

P. R. Popick - Project Programmer
P. H. Watson - Advisory Engineer

International Business Machines Corporation
Federal Systems Division
9500 Godwin Drive
Manassas, Virginia 22110

Abstract

This paper describes a simulation model of a packet switched, deadline scheduled bus network. This simulation is used to evaluate the communications requirements for a proposed sonar system. This is a discrete event simulation using the GPSS language.

INTRODUCTION

This paper describes a simulation model of a packet switched, deadline scheduled bus network. This simulation is used to evaluate the communications requirements for a proposed sonar system. In this system each message is divided into fixed size packets. Each packet contends on a priority basis for resources as the packet moves through the network. The packetizing of messages allows the bus network to be responsive to high priority messages characteristics of realtime systems.

The bus network consists of five local cabinet buses interconnected via two array buses and two system buses. Each of the local cabinet buses has local bus ports to external devices and bridges to interconnect the buses. The local bus ports use a bus interface unit (BIU) to connect the external devices to the bus. These BIUs perform message initiation, packet processing, packet deadline scheduling, packet queuing and message termination. The bridges interconnect the buses by providing packet processing, packet deadline scheduling and packet queuing.

The protocol used is the same for all the buses in the network. Briefly this protocol requires three independent subbuses; a command subbus, a token subbus and a data subbus. The sender initiates a message by sending a command request to the receiver along the command subbus. The receiver allocates a data buffer and returns a token to the sender along the token subbus. The sender sends a data packet to the receiver along the token subbus. The sender sends a data packet to the receiver along the data subbus. The receiver returns a token if there are additional data packets in the message. Otherwise the receiver returns a command acknowledgment to the sender along the command subbus.

Each message is defined to the model by specifying a sending BIU, a receiving BIU, a message size and a message priority plus several other fields which are explained in detail in "Message Definition" section. Whenever a message enters the model the message contends with other messages for the use of a processor or a bus facility. When a processor or facility is being used the message waits in a queue. A message is selected from a queue on either a priority basis or a first-in-first out (FIFO) basis. When the message is taken from the queue the model delays to simulate the processing overhead. Each processing step through the model consists of various statistics detailing the usage of the facilities and queues. An end of run report provides bus loading, BIU loading, bus and BIU queue statistics, messages transfer times and delineation of messages which missed deadlines.

MODEL DESIGN

MODEL RESOURCES AND CONFIGURATION

The bus network simulation model consists of five local cabinet buses with sixteen local bus ports, two system buses with eight consoles and two array buses with five local bus ports. Figure 1 depicts the model resources with the buses and bus ports. The resources for the local cabinet buses are numbered from one to one hundred. The five cabinets are lettered A through E. In each cabinet the first, tenth, eleventh and the twentieth resources are bridges which interconnect the cabinet buses to the system buses and the array buses. The remainder of the resources on the cabinet buses are BIUs which connect external devices to the local cabinet bus. The bridges are designated with two numbers which reflect the allocation of double resources for bridges in the model. This allows simulation of the bridges' parallel processing capabilities for input and output. In the BIUs the input and output are performed serially and the input processing must contend with the output processing.

The bridges and the consoles on the system bus are numbered from 101 to 127, while the bridges and devices on the array buses are numbered from 130 to 149. Normally only one of the system buses is used during a run. The number of cabinets and the number of bus ports per cabinet is determined by defining messages for each of the bus ports required. Any BIU or cabinet which has no message defined for it is not in the configuration for the model run. The resources in Figure 1 represent the maximal

Proceedings of the 1982
Winter Simulation Conference
Highland * Chao * Madrigal, Editors

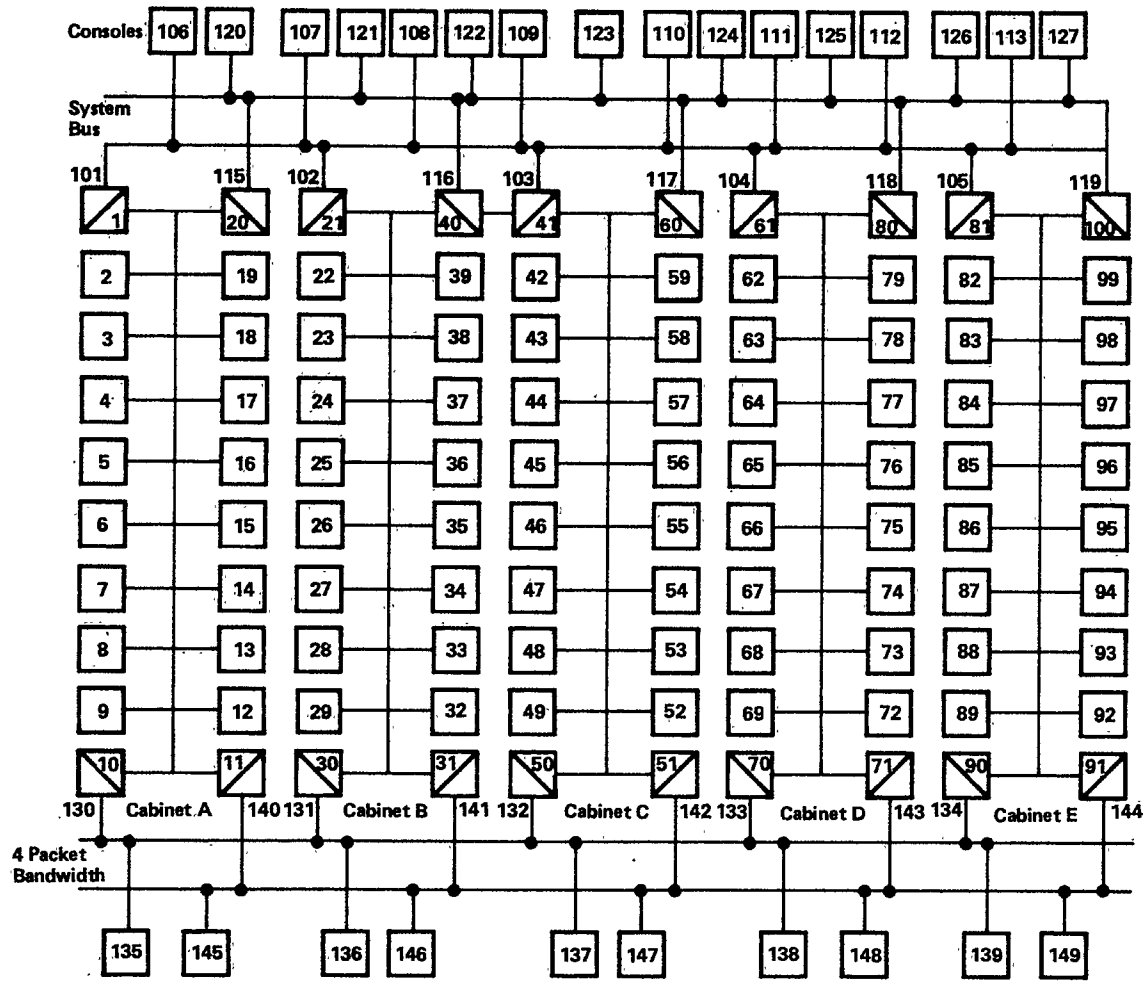

Figure 1. Bus Network Model Resources

The resources in Figure 1 represent the maximal configuration, and any subset of this configuration may be defined for evaluated purposes.

Each BIU, bridge, console or other device consists of GPSS resources which simulate the queues, buses, buffer loaders and unloaders, buffers and processing capabilities of the hardware. Whenever a message enters the model the message contends with the other messages for the use of a processor or a bus facility. When the processor or the facility is being used the message enters a queue. A message is selected from the queue on either a priority or a FIFO basis. When a message enters a processor the model delays to simulate the processing overhead. Each processing step through the model has a unique delay associated with it. The following sections describe these resources and delays in detail.

MESSAGE FLOW

The message flow through the model for an intercabinet message is shown in Figure 2. A message begins with a command request which is sent along the command subbuses and the bridges to the receiver on a priority basis. Delays associated with the transfer of the command along the route are assessed at each stage of the transfer. At the receiver the message request is handled on a FIFO basis for the IOCB setup and input buffer acquisition. A token is sent along the token subbuses and the two bridges with the processing delays assessed as indicated in Figure 2. At each bridge an I/O buffer must be acquired for the data packet.

Once the token has reached the sender an output buffer is acquired and loaded. The packet is then sent along the data bus to the first bridge. At the bridge the process is split into two processes. The first process acquires another I/O buffer and returns a token to the sender. The other process waits for reception of the previous token (if not the first packet) and sends the packet to the next bridge or BIU via the data subbus. These two processes proceed independently moving packets according to the traffic and the availability of the necessary buffers. The bridges limit the number of buffers allocated to any one message to allow throughput of all messages.

The processing of packets continues until the last packet of a message is sent. No token is returned when the last packet of a message is accepted at a bridge or BIU. Instead a command acknowledge indicating successful reception of the message is sent along the command subbus by the

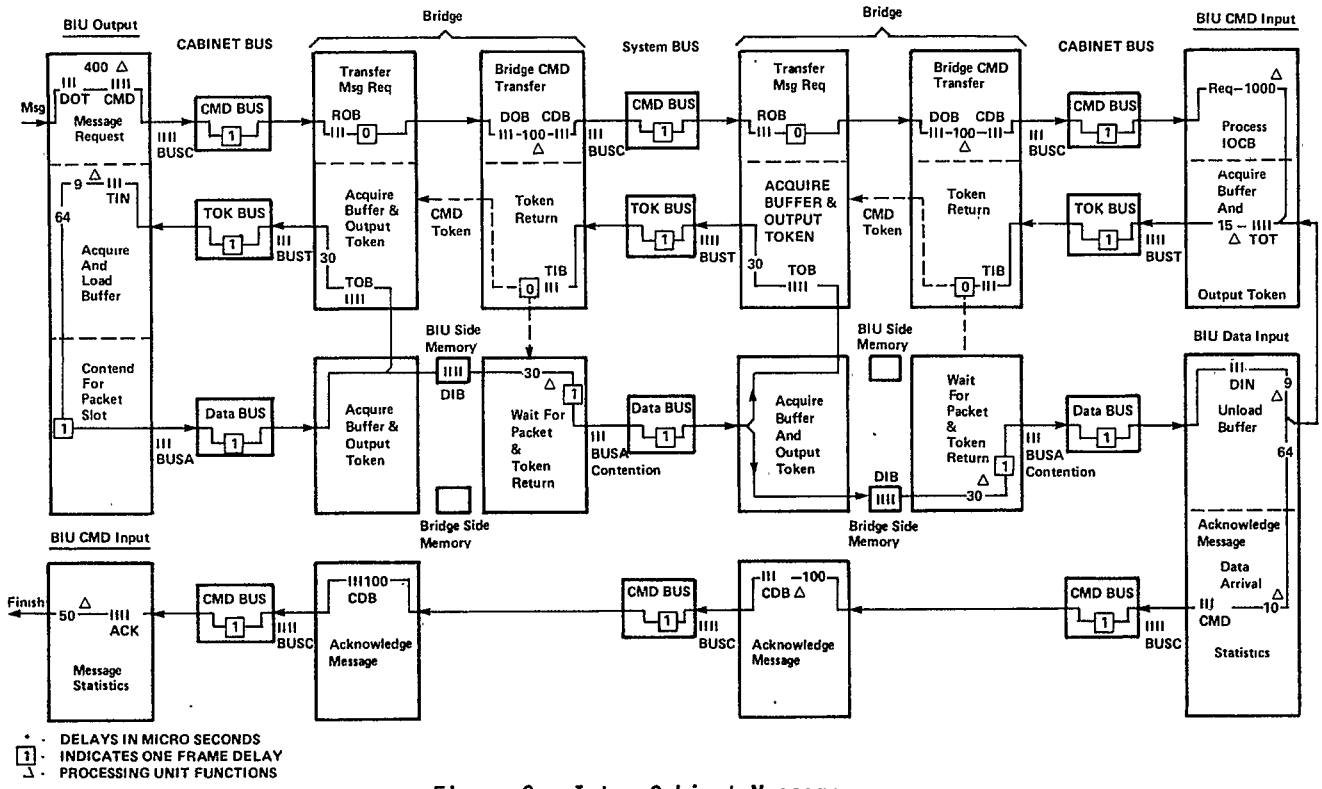

Figure 2. Inter-Cabinet Message

receiving BIU. Whenever a transfer across a bus occurs, there is a delay of one bus frame which is indicated in Figure 2 by a number "1" inside a square box. The bus frame is parameterized and may be set to different values for each bus. This feature allows construction and evaluation of the network with differing bus rates to aid in determination of the configuration which best supports the application.

BUSES

Each bus depicted in Figure 1 is composed of three subbuses; a command subbus, a token subbus and a data subbus. These subbuses are contended for independently but operate on the same frame synchronization.

- o COMMAND SUBBUS - The model consists of one command subbus for each cabinet bus plus two system command subbuses and two array command subbuses. Each message has a command request sent to the receiver to initiate the message, and an acknowledgment is sent to the sender to indicate successful completion of the message.
- o TOKEN SUBBUS - The model consists of one token subbus for each cabinet bus plus two system token subbuses and two array token subbuses. A token is sent by the receiver to the sender to indicate that a buffer is available for the next data packet of the message. The number of tokens sent is equal to the number of packets sent.
- o DATA SUBBUS - The model consists of one data subbus for each cabinet plus two system data subbuses and two array data subbuses. A data packet is sent by the sender to the receiver after each token is received indicating a buffer is available in the receiver.

QUEUES

Figure 2 also indicates all of the queues a message enters with three or four vertical lines. There are ten basic queues in the model which are either FIFO or priority queues, using time to complete to determine the priority. The following list specifies the ten queues, the queue departure criteria and a brief description.

- o Data output queue (priority) - The initial sender queue a message enters.
- o Command Output Queue (FIFO) - The sender queue a message enters after command request processing and before entering the command bus queue.
- o Command Bus Queue (priority) - Contains at most one entry from each bus port for a given command bus. The entries in the queue contend on a priority basis for a command bus.
- o Request Queue (FIFO) - The first receiver queue entered where messages contend for the IOCB setup on a FIFO basis.

- o Token Output Queue (priority) - The receiver queue entered after buffer reservation due to the arrival of a command request or a data packet.

- o Token Input Queue (priority) - Sender queue entered when a token is received from the token bus.

- o Token Bus Queue (priority) - Contains at most one entry from each bus port for a given token bus. The entries in the queue contend on a priority basis for the token bus.

- o Data Bus Queue (priority) - Contains entries from each bus port which contend on a priority basis for the data bus.

- o Data Input Queue (FIFO) - Receiver input queue for processing packets taken from the data bus.

- o. Acknowledge Input Queue (FIFO) - Sender queue for final message acknowledgment from the receiver.

BUFFER LOADERS AND BUFFER UNLOADERS

Each BIU in the model contains a packet loader and a packet unloader. These loaders and unloaders move data between the BIU I/O buffers and the attached devices memory. Tokens received into the token input queue of a particular resource contend on a priority basis to depart from the queue and gain the services of the loader. The loader may process only one token at a time. Similarly, data packets arriving at a particular resource contend on a FIFO basis to gain the services of the unloader. The unloader may service only one packet at a time. Tokens may be returned by the receiver prior to the completion of the unloading of a data packet providing there is an available input buffer.

BUFFERS

Each system bus bridge has one hundred 512 byte I/O buffers available on the cabinet bus side and fifty 512 byte buffers on the system bus side. Each array bus bridge has twenty-five 2048 byte I/O buffers available on the cabinet bus side and fifteen 2048 byte buffers on the array bus side. All other resources have eight 512 byte input buffers and eight 512 byte output buffers.

RESOURCE PROCESSING

Each device in the model contains units which are responsible for the processing function associated with the various phases of a message transfer. This processing unit is capable of doing only one task at a time. Each processing function contends for the processing unit on a priority basis according to the following list:

1 - IOCB processing
2 - ACK input processing
3 - Message Request processing
4 - ACK output processing
5 - Token Output processing
6 - DMA loader processing
7 - DMA unloader processing

The items with higher numbers in the above list take precedence over all items with lower numbers. The IOCB and the Message Request processing functions represent software interruptable processing. Whenever there is a request for the processing unit from any of the other functions within the same BIU the software interruptable tasks are suspended and resumed after the higher priority processing is completed. The processing function and associated delays are represented in Figure 2 by a triangle. All of the delays shown in Figure 2 are the default settings and may be altered for design tradeoff studies.

MESSAGE DEFINITION

A message is defined by specifying the frequency, deviation from frequency, start time, message route, message identifier, message size and message deadline. The following list describes each of these parameters in detail.

- o Frequency - The frequency for synchronous messages is the period between message generations in microseconds.

- o Deviation from Frequency - The deviation from the frequency is expressed in microseconds and represents a normal distribution interval about the time specified.

- o Start Time - The time, in microseconds, at which the first message for a message definition is generated. The following message is generated at start time plus the frequency plus or minus a random percentage of the deviation.

- o Message Route - A message route requires the source, the destination and up to four intermediate points. For intracabinet messages the source and destination are specified and the remaining four fields are filled with zeros.

For messages to or from a console on an array bus device, the message is divided into two subroutes. The first subroute is from the source to the first bridge using the number in the bridge block on the side of the approaching bus as indicated in Figure 1. The second subroute is from the bridge, using the number in the bridge block on the side of the departing bus, to the destination. For example, a message from console 106 to BIU 3 requires a subroute from 106 to 101 and a second subroute from 1 to 3.

Intercabinet messages require definition of three subroutes in a manner similar to that used for the console message definition. For example, a message from BIU 9 to BIU 22 requires subroutes from 9 to 1, from 101 to 102 and from 21 to 22.

- Message Identifier - A unique number between 1 and 999 should be given to each message defined in a given run.
- Message Size - The message size is specified in bytes to a maximum of two raised to thirtieth power minus one (1,073,741,823).
- Message Deadline - The time by which the message must reach its destination. This number is always less than the message frequency.

GPSS IMPLEMENTATION CONSIDERATIONS

There are three aspects of the GPSS implementation that are worth noting. The first is concerned with the architecture of the model, the second with minimizing CPU utilization and the third with operating GPSS under the IBM VM/CMS operating system environment.

Model Architecture

The model is comprised of nine processing modules: BIU Output, BIU Command Input, BIU Data Input, Bridge Data Input, Bridge Data Output, Bridge Command Input, Data Bus, Command Bus and Token Bus. Figure 2, while primarily designed to illustrate command, token and data flow, illustrates the manner in which these modules are chained together to simulate a message transfer. As previously indicated, transactions enter the model as messages awaiting transmission (Figure 2, upper left). The original transaction becomes a message request, then a token, then a data packet and finally a message acknowledgment. Transactions carry sufficient information in their parameters to identify to a module the particular set of resources (buffers, bus, etc.) it is to utilize when processing that transaction. For example, the Token Bus module handles the token transfers for each of the nine buses currently identified to the model. Likewise, the BIU Output module simulates the BIU output processing for each of the 106 BIUs in the network. In contrast to an approach where each BIU, Bridge or other resource has its own set of simulation logic, this approach has the benefit of reducing the number of executable statements in the model and facilitating upgrades to the model to reflect changes in the design being modeled. Of the executable statements in the model, only 10-15 percent are involved in making the model reentrant.

CPU Utilization

One method employed to minimize run time cost is to limit the number of transactions appearing at any one time on the GPSS current events chain. Since GPSS is constantly scanning this chain, a large number of entries increases CPU utilization dramatically. This minimization is accomplished by extensive use of user chains and GATE blocks. TEST blocks utilizing the conditional entry mode (which places delayed transactions on the current events chain) were not allowed. When it is necessary to place a transaction on a queue because a required resource is unavailable, it is placed on a user chain (the QUEUE block was never utilized) and a transaction SPLIT from the original is sent to a GATE block to await the availability of the resource. Transactions held at a GATE block and on user chains are not placed on the current events chain. When the resource becomes available a transaction leaves the GATE block and sets a "resource busy" flag to hold any other transactions waiting at the GATE block. The transaction then enters an UNLINK block causing the highest priority transaction to be taken off the appropriate user chain and sent on its way. The SPLIT transaction is then terminated having served its purpose.

A second technique used to reduce CPU cost is to avoid having transactions needlessly cycling in the model when there is no simulated demand on the system. For example, one approach to simulating bus transfers is to have a transaction cycling in a loop with a one-bus delay to check if any data packets are ready to be sent. While simple to implement, this approach needlessly consumes considerable CPU resources. The approach implemented instead used data packet transactions to drive the simulated data bus transfers. Thus if there is no data to transfer, the system is quiescent until there is data to transfer.

A final technique used to reduce CPU usage is the design of the simulation scenarios. When using the model our primary interest is in the network behavior during worst-case situations. The scenarios are designed to begin with a worst-case situation rather than running the model for extended periods in order for the worst-case situation to occur.

GPSS in a VM/CMS Environment

GPSS V - OS is used to implement the model. To execute GPSS V with the VM/CMS environment, it was necessary to create a VM/CMS executive to resolve the I/O references, i.e., to replace the OS JCL. What proved interesting is the way memory is allocated to this VM/CMS version of GPSS. During the design of the model, there was continuing demand for additional memory. The REALLOCATE statement was used to increase the storage allocation without encountering any limit. To date the GPSS common has been increased from the default value of 25,600 bytes to 195,000 bytes. The model currently uses 500,000 bytes of memory.

MODEL OUTPUT

MODEL OUTPUT DATA AVAILABLE

The Output of the model consists of statistics about the resources of the model, such as the queues, buses, loaders, unloaders, buffers and message arrival times. Queue statistics are printed for each queue entered and include the total number of entries, average time per entry and the maximum contents of the queue. The bus statistics are printed for each bus utilized and include the average utilization and the total

number of entries.

The loader statistics are printed for each BIU, console or other device which sent a message during a run. Similarly the unloader statistics are printed for each BIU, console or array device which received a message. These statistics include the utilization and the total number of entries. There are no loader statistics for bridges because it is assumed the bridges can move and receive data directly from the bus into memory. The buffer statistics are printed for each of the resources utilized during a run and include the allocated capacity, average contents, average utilization, total number of entries, average time in microseconds (ms) a buffer was held and the maximum number of buffers used simultaneously. The bus network simulation model is parameterized to allow analysis of alternative bus speeds, processing delays and packets sizes. The speed of any bus, and thereby the capacity, may be changed through the entry of one command. Similarly, the amount of memory allocated to a bus interface unit may be changed to determine the minimum memory required for a particular message scenario.

The bus network simulation model has been used to evaluate our network design and various alternative configurations. The parameterization has allowed the simulation model to be updated and refined as more details about the hardware and software are known. The evaluations have led to several enhancements to the protocol and will continue to be used to direct development requirements.

SAMPLE MODEL RESULTS

The bus network model tabulates the message response for each bus interface unit. A sample plot from this data is shown for three local cabinet bus scenarios in Figure 3. All messages in each scenario have a deadline of 39 ms which is shown by the heavy black horizontal line. Scenarios two and three represent bus loads of two and four times the bus load in scenario one. All messages in scenario one completed in less than 12 ms, while for scenario two (twice the bus load) all the messages completed in less than 20 ms. For scenario three (four times the bus load), all the messages completed in less than 39 milliseconds.

Another scenario for the acoustic subsystem of a sonar system was used to evaluate the capabilities of the network to support this application. This scenario consisted of 487 messages of varying periods executed for 190 ms. All messages were initiated during the first millisecond to ensure analysis of the peak load situation. For this scenario the devices on the array bus used a load/unload delay of 32 ms, while

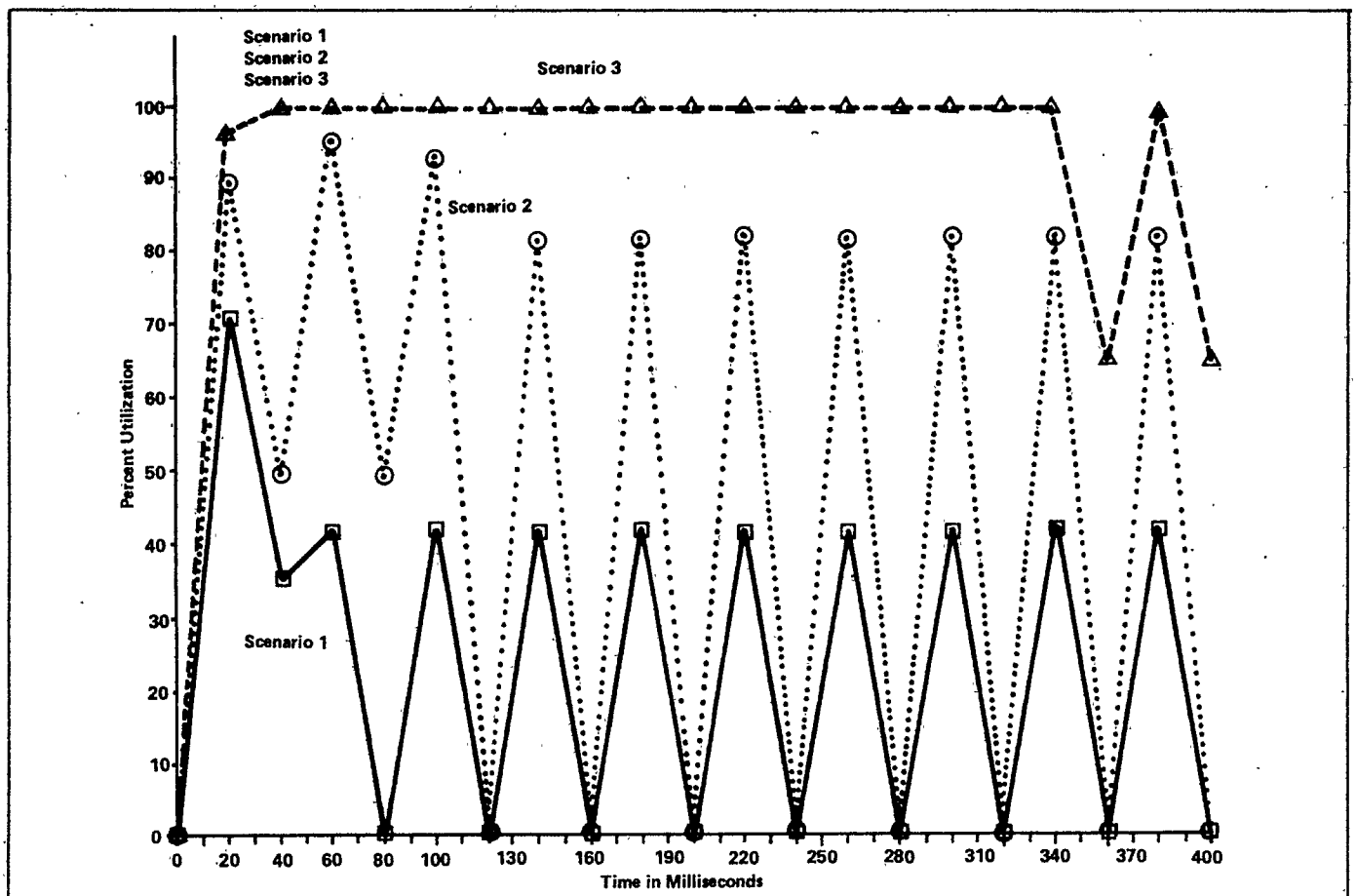

Figure 3. Data Bus Utilization Plot

all other devices used a load/unload delay of 64 ms. During the 190 ms execution time, statistics were tabulated every 20 ms.

Of the 487 messages approximately 80 percent of the messages completed in 75 percent of their message deadline. Table 1 contains average bus loads and bandwidth for this 190 ms execution. During the first 20 ms period, the extraordinary load generated by starting all messages simultaneously caused the system command bus to be 96% utilized. The system command bus queue reaches a maximum size of nineteen in this first 20 ms period. This high utilization was reduced as this initial peak load was processed. The average utilization of the command bus for the entire period was approximately 32 percent.

The system data bus reached nearly full utilization (99%) during the second 20 ms period as all the command requests were processed and caused data transfers to be initiated. The average utilization for the system data bus for the entire period was approximately 70 percent. Figure 4 contains a graph of the system command and data subbus utilization. The graph illustrates the periodic nature of the bus traffic after the initial peak load is worked off. The peak utilizations of the command and data subbuses occur at 100 and 180 ms.

Table 1. Average Bus Loads For Acoustic Scenarios

	PERCENT UTILIZATION			PACKETS	EQUIVALENT
Bus	Command	Token	Data	Transferred	Bandwidth
CAB A	9	66	66	3701	9 megabytes
CAB B	8	60	60	3236	8 megabytes
CAB C	9	38	38	2059	5.5 megabytes
CAB D	8	34	34	1845	4.9
System	32	69	69	1210	3.2
ARRAY 1	8	58	58	971	10.4
ARRAY 2	8	33	33	575	6.2

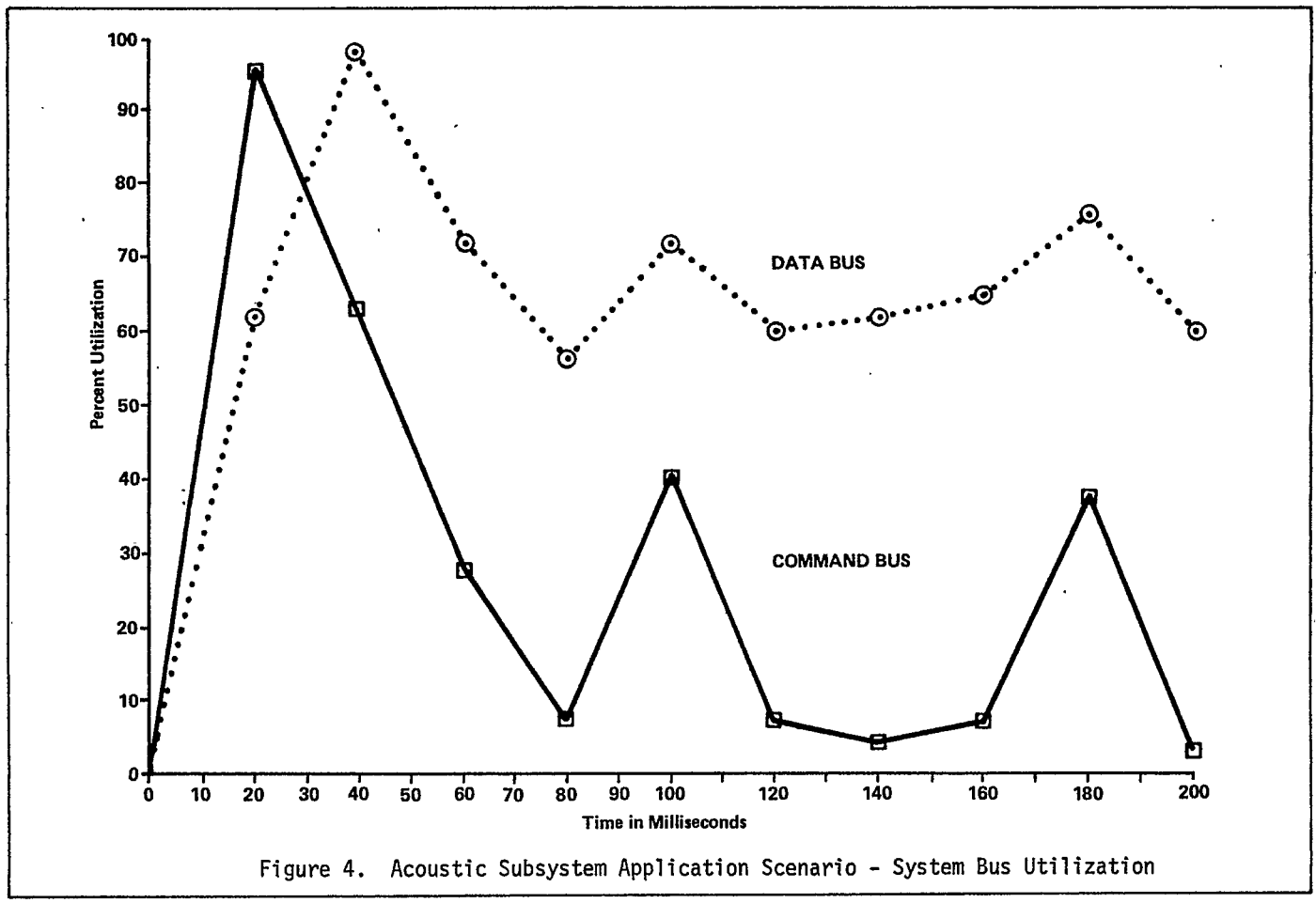

Figure 4. Acoustic Subsystem Application Scenario - System Bus Utilization