1-1-1995

Implementation Of Data Recording Devices

Darren D. Humphrey

INSTITUTE FOR SIMULATION AND TRAINING

Contract Number N61339-94-C-0024
CDRL A00Q
STRICOM
March 8, 1995

Implementation of Data Recording Devices

Institute for Simulation and Training
3280 Progress Drive
Orlando FL 32826

University of Central Florida
Division of Sponsored Research

IST-TR-95-02

INSTITUTE FOR SIMULATION AND TRAINING

Implementation of Data Recording Devices

IST-TR-95-02
March 8, 1995

Prepared For:
STRICOM
Contract Number N61339-94-C-0024
CDRL A00Q

Prepared by:
Darren D. Humphrey

Institute for Simulation and Training • 3280 Progress Drive • Orlando, Florida 32826
University of Central Florida • Division of Sponsored Research

TABLE OF CONTENTS

1 Introduction

1.1 Purpose

This report is a contract deliverable item, CDRL A00Q, under subtask 3.2.5.2, entitled "Data recording Tools", of the STRICOM contract N61339-94-C-0024, entitled "TRIDIS: A Testbed For Research In Distributed Interactive Simulation".

1.2 Abbreviations and Acronyms

DAT	*D*igital *A*udio *T*ape
DLIF	*D*ata *L*ogger *I*nterchange *F*ormat
DIS	*D*istributed *I*nteractive *S*imulation
DMA	*D*irect *M*emory *A*ccess
FFS	*F*ast *F*ile *S*ystem
IBM PC	*I*nternational *B*usiness *M*achines *P*ersonal *C*omputer
IDLF	*IST D*ata *L*ogger *F*ormat
I/ITSEC	*I*nterservice/*I*ndustry *T*raining *S*ystems and *E*ducation *C*onference
IST	The *I*nstitute for *S*imulation and *T*raining
MS-DOS	*M*icro*s*oft *D*isk *O*perating *S*ystem
MVME	*M*otorola *VME*
RAID	*R*edundant *A*rray of *I*nexpensive *D*isks
RAM	*R*andom *A*ccess *M*emory
SCSI	*S*mall *C*omputer *S*ystems *I*nterface
STRICOM	*U.S. A*rmy *S*imulation *T*raining and *I*nstrumentation *C*ommand
TAR	*T*ape *A*rchive
TRIDIS	A *T*estbed for *R*esearch *I*n *DIS*
VME	*V*ersabus *E*

1.3 Background

As required by subtask 3.1.2.4 entitled "IDEMO Traffic Analysis", Institute for Simulation and Training (IST) personnel working on the Testbed for Research in DIS (TRIDIS) project logged data from the Distributed Interactive Simulation (DIS) demonstration at the 1994 Interservice/Industry Training Systems and Education Conference (I/ITSEC). Data logging tools available to IST at the time were not powerful enough to reliably log data at the transmission rates expected on the I/ITSEC network. Further, the IBM PC hardware platform used by the IST test tools, which can handle at most 200-300 packets per second, was not deemed powerful enough even if the software was improved.

To solve the problems of capacity and reliability, a fast high capacity logger was developed specifically for the I/ITSEC demonstration. It was used at I/ITSEC to record approximately five Gigabytes of data, with peak traffic rates of approximately 500 packets per second; this was nowhere near the logger's limit of about 1,000 packets per second.

This report will describe in detail the data logger used at I/ITSEC; both the details of the software and the actual hardware setup used at the conference.

1.4 Review of Existing Recording Systems

Except for tools existing at IST, existing recording systems were not examined. The primary reason other tools were not examined was lack of time. There was not enough time available prior to the I/ITSEC demonstration to gather information on available tools that could possibly be powerful enough to handle the loads required for I/ITSEC.

The Mak VRLink logger was reviewed but did not log at fast enough speeds to be used for our purposes.

2 Hardware

2.1 Platform Selection

The IBM PC hardware that early IST test tools run on has been shown to be inadequate for heavy network loads. In particular, the MS-DOS file system is not structured for high speed, high volume disk accesses. To meet the expected load requirements for the I/ITSEC demonstration (approximately 300-500 packets per second), a new platform was needed. Several in house platforms were examined for suitability.

The Silicon Graphics Onyx and Indy platforms were examined. Based on past experience, Silicon Graphics computers have been able to log approximately 400-600 DIS packets per second, sustained. This recording rate left little margin for error and little room for future growth in network load.

The Motorola MVME197 single board computer system was found to be the best choice. Using Motorola's Fast File System (FFS), a single Motorola target board was able to log over 1000 packets per second of DIS traffic, sustained. FFS is designed for maximum throughput between the single board computer and a disk device. The advantage of FFS is increased throughput. Disadvantages include fixed file size, and incompatibility with the UNIX file system.

The MVME197 computer runs Motorola's VMEexec real time operating system; the advantage of a real time operating system is its completely deterministic performance. A UNIX based system could suffer from periods of poor performance, based on the performance of other tasks.

2.2 Hardware Setup

Each "logger station" consists of a 6U VME chassis with two Motorola MVME197 single board computers. One board runs Motorola System V R4 UNIX, and is designated as the "host" board. The other board runs Motorola VMEexec 3.0, and is designated as the "target" board. Each board has its own local SCSI bus, serial ports, and Ethernet port.

The host board has local file storage which stores the binary image of the logger program and the VMEexec operating kernel. The target board has disks formatted with Motorola's FFS.

2.3 I/ITSEC Hardware Setup

Three logger stations were used at the I/ITSEC demo; all were connected to the main I/ITSEC demo network. Each logger had approximately 1.6 Gigabytes of storage dedicated to logging. The target board of each logger station logged data directly from the I/ITSEC network. The host board of each logger station was connected to an internal "administration" network. A Motorola X-Terminal connected to the administration network was used as the display device for the user interface processes of the loggers.

One logger station was a spare in case one of the two main logger stations failed. The remaining two logger stations were run in "double buffer" fashion; the first logger would run until nearly full, and then the operator would start the second logger. Output from the target board's console would inform the operator when a logger was nearly full.

While the first logger was being backed up onto tape, the second logger would capture data. This process was repeated until the end of each day.

At first, the utilities provided by Motorola were used to move files from the FFS to a UNIX disk. The Motorola utilities proved to be too slow, so a new utility was written on the spot. This new utility moved files at a much higher rate than the Motorola utilities. After files were moved to a UNIX disk, they were archived to tape using UNIX Tape ARchive (TAR) format. Two different copies of the data were made, each on different media. 4mm and 8mm Digital Audio Tape (DAT) were the media types used.

3 Software

3.1 Software Design Goals

The logger was designed with several goals in mind.

- First, the logger had to be powerful enough to handle the expected network load. This goal was accomplished by selecting the appropriate hardware, OS, and filesystem.

- Second, the logger had to be reliable. Reliability was achieved through use of redundant hardware (discussed in more detail later).

- Third, the impact of faults had to be minimized. If a logger were to sustain a run time fault (e.g., a power surge), damage to previously logged data should be minimized. Fault tolerance was achieved, to a degree, by segmenting log files into standalone segments. Had a logger suffered a fault, the worst case would have been a loss of the currently opened log file segment. This did not happen.

- Fourth was ease of translation. At the time the new logger was being written, new log file format standards were being designed by STRICOM and Loral. The IST logger file format was designed for easy translation into other log file formats.

- Fifth was completeness of data. To support IST data analysis efforts, the data logger was designed to log complete network frames at the Ethernet frame level.

3.2 Software Design

The logger consists of two processes: a X11/Motif user interface and a real time logger task.

The user interface task runs on the host board, and the logger task runs on the target board. The user interface task communicates with the logger task via the VME bus. The user interface task starts and stops logging, and controls when the program exits. The logger task gets Ethernet frames from the network and writes them to its local disks. The logger task also responds to commands from the user interface task. The next two sections describe each task in detail.

3.3 User Interface Task

The user interface task is an X11/Motif program which runs on the host board. It uses VMEexec queues to communicate with the logger task which runs on the target board. Because it uses VMEexec queues, it must be compiled using the VMEexec *mkmk* command.

The user interface task simply waits for a button press from the user and then sends a message to the logger task via a VMEexec queue. The messages contain the name of the button which was pressed by the user (i.e., STOP, START, or QUIT).

3.4 The Logger Task

The logger task performs the actual data logging function. It logs entire Ethernet frames from the target board's local Ethernet port and stores them onto disks attached to the target board's SCSI bus. The logger will span multiple disks; that is, it will continue logging on the next available disk when the current disk is full.

To help localize damage caused by faults (e.g., a power outage) logged data is stored in small chunks. Each chunk contains at most 16 Megabytes of data, or one hour of data, whichever is less. The theory is that if a logger were to crash, only the file which is being written to at the time would be lost; the other files would likely be intact.

Configuration information written into the logger contains the device number and free space for each disk connected to the target board's SCSI bus. This information is currently hard coded for each configuration; if further development is scheduled for this task, the logger will be modified to automatically detect available disks and determine the amount of free space on each

disk. The logger uses this information to decide when to switch to the next disk available for logging.

The logger task uses a VMEexec supplied device driver to access the Ethernet port on the single board computer. The device driver offers easy, high level access to the port with reasonable performance. Using the device driver has some limitations, though. For instance, even though the hardware reference manual shows that the MVME197 board supports promiscuous mode access to the Ethernet port, the device driver itself does not implement promiscuous mode. Because of this limitation, only traffic broadcast of the Ethernet level was captured. Point to point and other non-broadcast traffic was not captured.

3.5 Logger File Format

Data logging tools developed by IST over the last two years have used various versions of the IST Data Logger Format (IDLF). The data logger described in this report uses IDLF Version 3. The formats of Versions 1 and 2 are discussed also.

IDLF 1.0 is the format used by the IBM PC based data loggers. An IDLF 1.0 file consists of a series of variable length packet records. Each packet record contains a fixed length header and a variable length data field. The packet record header is described by the following code:

```
struct packet_header
{
        UINT_16 length;
        UINT_32 seconds;
        UINT_32 micro_seconds:
}
```

In IDLF 1.0, the packet record header is stored in IBM PC (Intel) byte order. IDLF 2.0 is identical to IDLF 1.0, but the packet record header is stored in Network byte order.

The logger described in this report uses IDLF 3.0. IDLF 3.0 is essentially the same as IDLF 2.0 with the addition of a file header. The packet record header in IDLF 3.0 is slightly different, but is still stored in Network byte order. The file header is supposed to contain information about the time and date when the log file was created. Due to a bug in the logger code, the header of the files logged for IDEMO 94 actually contains garbage. This bug will be very easy to fix if further development is scheduled, and does not affect the integrity of the log file data. The header is 2 bytes long and should be ignored.

The second difference is a change to the header size which is written preceding every packet. The C data structure for the each packet header is:

```
struct packet_header
{
   UINT_32  length;
   UINT_32  seconds;
   UINT_32  micro_seconds;
}
```

The *length* field denotes the length of the packet to follow (not including the length of the *packet_header* structure), in octets (8 bit bytes). The *length* field is now a 32 bit unsigned integer.

The *seconds* field denotes how many seconds had elapsed *relative to the start of this log file* when the packet was received. The *seconds* field is a 32 bit unsigned integer (unchanged).

The *micro_seconds* field denotes how many micro seconds had elapsed *relative to the seconds field* when this packet was received. The *micro_seconds* field is a 32 bit unsigned integer (unchanged).

The overall format of the log file looks like this:

file_header (2 bytes)
pdu_header (6 bytes)
pdu_data (depends on *pdu_header.length*)
.....
pdu_header (6 bytes)
pdu_data (depends on *pdu_header.length*)

The *file_header* and *pdu_header* structures are written to disk in *network byte order*. This is different from the PC version, which writes its headers in *PC byte order*. This is version 3 of the IST Logger file format.

3.5.1 Data Logger Interchange Format (DLIF)

At the request of STRICOM, two members of IST's technical staff participated in discussions to draft a standard format for the interchange of logged data fields. The two TRIDIS team members spent several days in meetings to devise a logger file interchange format. When the final file format was selected, TRIDIS personnel were directed not to implement DLIF, but to use conversion programs which would be supplied by other contractors. To date, IST has not received any conversion tools.

4 Performance Measurements

Before using the logger at I/ITSEC, load tests were performed in IST's laboratory to assess the reliability and performance of the logger. Using a load testing program developed by Larry

Breneman at IST, the logger was subjected to heavy network traffic loads. Network load was progressively stepped up from approximately 100 packets per second of DIS data to nearly 2,500 packets per second of DIS data.

At frequent intervals during the test, the logger would print out total packets logged and total elapsed logging time. These numbers would be used to compute the average traffic rate which the logger could handle. The results showed that for short bursts, the logger could handle approximately 1,200 packets per second. For long bursts, the maximum sustained rate was found to be 1,000 packets per second. The increase in performance for short bursts is due to RAM buffering of incoming packets.

The performance numbers are average performance over relatively short intervals. Peak performance numbers were not measured.

5 Unfinished Business

Several parts of the logger remain unfinished. To start with, there is no companion playback tool for the logger. Data must be converted into a different format to be played back or examined. The log file format is very similar to the format used by the IST Scanner, and in fact will work with some versions of the Scanner. Coding a playback utility should not be too difficult, because the logger file format is very close to the format used by current playback utilities.

The logger currently does not support promiscuous mode capture. To add this capability will either require accessing the MVME197 Ethernet port at a lower level, or obtaining the source code to the Motorola device driver and modifying it.

Higher logging speeds may be possible through a number of methods. First, accessing the Ethernet port at a lower level will undoubtedly allow for more efficient transfer from the Ethernet to the SCSI bus. Highly efficient direct DMA transfers may allow much higher throughput to the SCSI bus than with the VMEexec device driver. Higher disk transfer rates may be achieved through lower level disk accesses or faster disks. A RAID (Redundant Array of Inexpensive Disks) could boost disk throughput by a factor of 10. Completely bypassing a file system of any sort would improve throughput also. By combining these techniques, a MVME197 based logger should be able to consistently log 5000 DIS packets per second.

The logger software needs some cleanup and fine tuning. The code was produced in a relatively short period of time, with a single use in mind. Further improvements in robustness, design, and commenting are needed to "flesh out" the logger. There are also a few minor bugs which need to be addressed (including the file header bug described earlier).

The source code and system build files for the Data Logger will be delivered to STRICOM on an 8mm DAT tape, accompanying this report. The source code for the utilities used to move data from FFS to UNIX will be included.

5.1 Enhancing and Modifying TESTBED Data Logger and Mak Logger

The TESTBED logger was ported to run on a generic UNIX System V platform, using Berkeley sockets and the snoop protocol. The snoop protocol allows a process which is using sockets to "snoop" the various levels of the protocol stack. In effect, this allows a process to examine entire Ethernet frames; precisely what is needed for data logger. The UNIX version of the logger writes IDLF 2.0, and has been tested on Silicon Graphics systems.

Unfortunately, the Motorola version of UNIX System V does not support the snoop protocol, so the TESTBED logger was not ported to Motorola. The Silicon Graphics version of the TESTBED logger was delivered to the Joint Interoperability Testing Center to support their use of the Silicon Graphics version of the IST Scanner.

6 References

[VMERef] "VMEexec Reference Manual Volume 1", Motorola, Inc., 1993

[VMEDev] "VMEexec Device Driver Reference Manual Volume 1", Motorola Inc., 1993.