TEST PLAN FOR 32-BIT MICROCOMPUTERS

FOR THE WATER RESOURCES DIVISION

CHAPTER A: TEST PLAN FOR ACQUISITION OF PROTOTYPE 32-BIT MICROCOMPUTERS

By Norman E. Hutchison, Arlen W. Harbaugh, Richard A. Hollway, and Charles F. Merk

---

U.S. GEOLOGICAL SURVEY

Open-File Report 87-455

Reston, Virginia

1987

DEPARTMENT OF THE INTERIOR

DONALD PAUL HODEL, Secretary

U.S. GEOLOGICAL SURVEY

Dallas L. Peck, Director

For additional information
write to:

Assistant Chief Hydrologist for
Scientific Information Management
U.S. Geological Survey
440 National Center
Reston, Virginia 22092

Copies of this report can
be purchased from:

U.S. Geological Survey
Books and Open-File Reports Section
Federal Center
Box 25425
Denver, Colorado 80225

TABLE OF CONTENTS

		Page
Abstract		1
1.	General information	1
	1.1 Summary	1
	1.2 Pretest background	2
	1.3 Objectives of the plan	2
2.	Plan for testing microcomputers	3
	2.1 Identifying qualified vendors	3
	2.2 Tests performed	3
	2.3 Test requirements	4
	2.4 Work group organization	4
3.	Evaluation methodology	5
	3.1 Test scripts	5
	3.2 Evaluation criteria	5
4.	Test descriptions (scripts)	6
	4.1 Shared data access	6
	4.2 Basic performance	6
	4.3 Test Fortran 77 compilation and execution	7
	4.4 Data base management system test	7
	4.5 Tymnet connection	8
5.	Selected references	8

APPENDIXES

A.	Microcomputer evaluation matrix	A-1
B.	Script for testing shared data access	B-1
C.	Scripts and programs for testing basic performance	C-1
D.	Scripts and programs for testing Fortran 77	D-1
E.	Script for data base management system test	E-1

TEST PLAN FOR 32-BIT MICROCOMPUTERS FOR THE WATER RESOURCES DIVISION

CHAPTER A: TEST PLAN FOR ACQUISITION OF PROTOTYPE 32-BIT MICROCOMPUTERS

by Norman E. Hutchison, Arlen W. Harbaugh, Richard A. Hollway, and Charles F. Merk

ABSTRACT

The Water Resources Division (WRD) is evaluating 32-bit microcomputers to determine how they can complement, and perhaps later replace, the existing network of minicomputers. The WRD is also designing a National Water Information System (NWIS) that will combine and integrate the existing National Water Data Storage and Retrieval System (WATSTORE), National Water Data Exchange (NAWDEX), and components of several other existing systems.

The purpose of this report is to document the procedures and testing done in a market evaluation of 32-bit microcomputers. The results of the testing are documented in the NWIS Project Office. The market evaluation was done to identify commercially available hardware and software that could be used for implementing early NWIS prototypes to determine the applicability of 32-bit microcomputers for data base and general computing applications. Three microcomputers will be used for these prototype studies. The results of the prototype studies will be used to compile requirements for a Request for Procurement (RFP) for hardware and software to meet the WRD's needs in the early 1990's.

The identification of qualified vendors to provide the prototype hardware and software included reviewing industry literature, and making telephone calls and personal visits to prospective vendors. Those vendors that appeared to meet general requirements were required to run benchmark tests.

1. GENERAL INFORMATION

1.1. Summary

The Water Resources Division (WRD) is evaluating 32-bit microcomputers to determine how they can complement the existing network of Prime* minicomputers. By 1990, technology may have advanced to the point that microcomputers could replace the minicomputers as the primary computing resource. The WRD is also designing a National Water Information System (NWIS) that will combine and integrate the existing National Water Data Storage and Retrieval System (WATSTORE), National Water Data Exchange (NAWDEX), and components of several other existing systems. This is an extensive design and development effort that will involve many WRD field offices and headquarters personnel over the next 3-5 years and cost several million dollars to complete.

* Use of brand and trade names in this report is for identification purposes only and does not constitute endorsement by the U.S. Geological Survey.

The purpose of this report is to document the procedures and testing done in a market evaluation of 32-bit microcomputers. The results of the testing are documented in the NWIS Project Office. The market evaluation was done to identify commercially available hardware and software that could be used for implementing early NWIS prototypes to determine the applicability of 32-bit microcomputers for data base and general computing applications. The results of the prototype studies will be used to compile requirements for a Request for Procurement (RFP) for hardware and software to meet the WRD's needs in the early 1990's.

Three microcomputers will be used to evaluate their general computational potential as well as their capability for data base management to meet the above requirements. The requirement is for hardware and software with general computing and networking capability for running large models and rapid prototyping of data base applications. The software must provide easy access to data, data manipulation capability, and report generation. The hardware will include disk drives, magnetic tape drives (9-track and cartridge tape), 32-bit microcomputer work stations with at least 4 megabytes of memory, and communications interface to the local area network and GEONET, the Geological Survey's X.25 telecommunications network. The software will include a relational DBMS with SQL query language, operating system to support distributed processing and transportability, and FORTRAN and C programming language compilers.

The identification of qualified vendors to provide the prototype hardware and software included reviewing industry literature, and making telephone calls and personal visits to prospective vendors. Those vendors that appeared to meet general requirements were required to run benchmark tests. The tests conducted to evaluate the microcomputers included tests for shared data access, basic performance, FORTRAN 77 compilation and execution, relational data base management system performance, and access to TYMNET. General functional requirements were specified in the tests. Performance requirements were not specified because one of the purposes of the market evaluation was to determine what those requirements should be.

1.2. Pretest Background

Previous evaluations of INFO and Prime DBMS running on the Prime minicomputers demonstrated that they could not provide the level of performance required by the NWIS. Earlier planning for the NWIS also specified a national archive to be maintained on the Reston node. A Request for Information (RFI) for the national archive was advertised and 6 responses were received. The Information Management and Computer Committee (IMACUAC) later recommended that microcomputers be evaluated for the development and implementation of the NWIS and that a distributed data base be considered. The information in the RFI and responses to it will help define the data base requirement for the microcomputers. A work group composed of WRD personnel was formed to accomplish the market evaluation.

1.3. Objectives of the Plan

The WRD has an 8-year contract (ending in March 1991) with Prime Computer, Inc. for 75 minicomputers with various levels of upgrade. By 1990, WRD will need to award another contract for computing resources. The options

are to award a contract for minicomputers to replace the Prime minicomputers as they become nonserviceable, to contract for microcomputers, or contract for a combination of minicomputers and microcomputers. Large mainframes with terminals will not be an option unless WRD changes it's policy decision to implement a distributed information system. The first objective of this procurement is to acquire prototype hardware and software to evaluate the general computing capability of microcomputers as an alternative to minicomputers.

The second objective is to evaluate the capability of microcomputers to manage the NWIS integrated data base in a distributed environment. By integrating the hydrologic data bases, WRD expects to be able to provide better access to water data, eliminate redundant data, and improve productivity. WRD is attempting to use off-the-shelf software, to the greatest extent possible, to reduce development costs. A DBMS facilitates the logical integration of data and elimination of duplicate data.

Rapid prototyping is one of the development strategies being used by the NWIS, and the requested hardware and software will be used for this purpose. Rapid prototyping involves partitioning and scaling of design and development projects to fit available resources, allows concepts and technologies to be tested and evaluated as independent models, and provides for users to test and evaluate individual components of the system as they are developed. These prototypes will attempt to build a data-base application with off-the-shelf hardware and software, including disks, tapes, and workstations; and DBMS, edit, update, and query software.

The purpose of this report is to document the procedures and testing done in a market evaluation of 32-bit microcomputers. The results of the testing are documented in the NWIS Project Office. The market evaluation was done to identify commercially available hardware and software that could be used for implementing early NWIS prototypes to determine the applicability of 32-bit microcomputers for data base and general computing applications. The results of the prototype studies will be used to compile requirements for a Request for Procurement (RFP) for hardware and software to meet the WRD's needs in the early 1990's. A WRD Work Group was formed to conduct the tests and market evaluation.

2. PLAN FOR TESTING MICROCOMPUTERS

2.1. Identifying Qualified Vendors

The identification of qualified vendors included reviewing vendor and industry literature, and making telephone calls and personal visits to prospective vendors. Those vendors that appeared to meet the general requirements in Section 3.2 below were required to run a benchmark. The list of factors considered in identifying qualified vendors is shown in Appendix A.

2.2. Tests Performed

The following tests were conducted:

a. Shared data access (also referred to as the LAN test)

The purpose of this test was to demonstrate that disk files located on a remote system on the LAN can be accessed in the same way as local files. Also, the system performance when accessing remote files was measured and compared to performance when accessing local files.

b. Basic performance (also referred to as the C compiler test)

The purpose of this test was to demonstrate basic computer capabilities considered to be essential in any system purchased. Two scripts involving programs written in the C programming language were used. The fundamental capabilities being tested were arithmetic operations, input, output, and multitasking.

c. FORTRAN 77 Test

The purpose of this test was to demonstrate that the FORTRAN compiler works correctly, and to measure the speed of both the compiler and the compiled program. Two programs were tested -- one to show that most functions of FORTRAN worked correctly, and the other to test how well one of the large USGS scientific modeling programs performed.

d. Data Base Test

Several subsets of NWIS data files were loaded into a test data base using a relational design similar to the present NWIS design. Loading, indexing, and sample retrievals were timed.

e. TYMNET Test

The purpose of this test was to demonstrate the capability to connect the microcomputers to TYMNET using X.25 packet switching protocol.

2.3. Test Requirements

Vendors electing to conduct the benchmark tests, after meeting all or most of the general requirements in Section 3.2, were required to provide the necessary hardware, software, and personnel to perform the tests. Hardware required included two microcomputers with appropriate resources to meet the evaluation criteria listed in section 3.2 below, and access to a local area network. Software required included a multitasking operating system, FORTRAN 77 and C compilers, and a relational DBMS. A record was kept of all hardware and software to be proposed by the vendor, whether or not it was tested.

2.4. Work Group Organization

A Work Group composed of members from the NWIS Project, the WATSTORE Program, the Distributed Information System (DIS) Program, and one field office was organized to conduct the evaluation. Test Teams were selected from the Work Group for specific test assignments. The Work Group members selected were:

Arlen Harbaugh, Chairman	DIS
Jo Porter	DIS
Daphne Chinn	DIS
Charlie Merk	WATSTORE
Richard Hollway	Oregon State Office
Norman Hutchison	NWIS

3. EVALUATION METHODOLOGY

3.1. Test Scripts

Scripts describing all five tests and how they were to be conducted were prepared and are presented in Section 4 and Appendixes B through E. The scripts were sent to each vendor conducting the tests prior to the start of testing, so they had enough information to prepare and set up the tests. The designated Test Team reviewed the scripts with the vendor upon arrival at the test site to make sure they understood the tests and the test procedure. The test scripts could also be modified to provide for recording time information, notes on the configuration, and any irregularities encountered.

3.2. Evaluation Criteria

This was an informal benchmark test to determine the general functionality of the microcomputers. There were no performance requirements because one of the purposes of the testing was to help determine these requirements. The tests were for functional capability, but some timing was done for comparison purposes and to help determine appropriate performance requirements.

The requirement was for hardware and software with general computing and networking capability for running large models and rapid prototyping of data base applications.

The hardware provided had to include:

a. 32-bit microcomputer workstations with full 32-bit memory transfer and 4-MB of memory expandable to 16-MB.

b. fixed disk drives capable of storing at least 100-MB per drive.

c. magnetic tape drives (9-track and cartridge tape).

d. floppy disk drives.

e. floating point hardware.

The software provided had to include:

a. operating system to support distributed processing and transportability.

b. relational data base management system with SQL query language and FORTRAN interface.

c. standard FORTRAN 77 compiler.

d. Index Sequential Access Method (ISAM) subroutine library.

e. full screen editor.

f. spreadsheet software.

g. word processing.

h. electronic mail.

The combined hardware and software was to provide the following capabilities:

a. connect to Local Area Network (LAN).

b. connect to TYMNET using X.25 protocol.

c. support shared data base on LAN (files located on a remote system can be accessed the same way as local files).

d. multitasking.

4. TEST DESCRIPTIONS (SCRIPTS)

The tests conducted to evaluate the microcomputers included tests for shared data access, basic performance, FORTRAN 77 compilation and execution, relational data base management system performance, and access to TYMNET. The full scripts and programs of the first four tests are listed in Appendixes B through E. The TYMNET script is short and is presented in the text in Section 4.5. Some of the language in the scripts is in UNIX operating system notation and will be meaningless to the lay reader.

4.1. Shared data access

The purpose of this test was to demonstrate that disk files located on a remote system on the LAN can be accessed in the same way as local files. Also, the system performance when accessing remote files was measured and compared to performance when accessing local files. Two utilities, copy and sort, were used in this demonstration. The source file was the same for all of the copy and sort commands, and contained 10,000 80-character records. The script for testing shared data access is contained in Appendix B.

4.2. Basic Performance

The purpose of this test was to demonstrate basic computer capabilities considered to be essential in any system purchased. Two scripts involving programs written in the C programming language were used. The fundamental capabilities being tested were arithmetic operations, input, output, and multitasking. Performance was timed in parts of this test in order to allow comparison among different vendors. There were no absolute performance requirements. There were two scripts involved in this test. All programs were supplied by the Water Resources Division.

Script one for testing basic performance (also referred to as the C I/O script) required that several C programs be compiled and run. The programs were run by a single command file that used intertask communication, called piping, and file redirection. Piping makes use of multitasking. To further test multitasking, the command file was invoked multiple times simultaneously.

Script two (also referred to as the C data base script) tested input and output both to disk and to the user terminal. The test consisted of timimg the compilation of the program and executing it interactively. Nothing was timed during the execution. After execution using a file on the local system, it was executed using a remote file on another system across the LAN to further test transparent use of remote files across a LAN. Script one and two as well as two programs are listed in Appendix C.

4.3. Test FORTRAN 77 Compilation and Execution

The purpose of this test was to demonstrate that the FORTRAN 77 compiler works correctly, and to measure the speed of both the compiler and the compiled program. Two programs were tested -- one to show that most functions of FORTRAN work correctly, and one to test how well one of the large WRD scientific models performed. Both programs were supplied by the Water Resources Division.

Script one (also called the Training data base) tested many standard FORTRAN 77 capabilities, including direct access input and output. Compilation was timed, and the program was executed interactively without timing. The execution was repeated using a remote data file. Script one and a listing of the two programs required for this script are contained in Appendix D.

Script two (also referred to as the model script) tested the compilation and execution of a 5000 line model program that is typical of those used by WRD scientists. Time of execution was measured when running by itself and when three copies were running simultaneously. The time of compilation was also measured. A FORTRAN 66 version of the program was also compiled to see if the compiler could handle this code. The program is documented in McDonald and Harbaugh (1984). Script two is also listed in Appendix D.

4.4. Data Base Management System Test

Six subsets of NWIS data files were loaded into a test data base using a relational design similar to the present NWIS design. The data were in a Rapport DBMS format, so it had to be converted for input to the vendor's DBMS. Indexing and sample retrievals were run, and the load, indexing, and retrieval runs were timed. There were seven steps to run the test, including:

a. Assemble the six test data sets. There were test data sets for site header, agency, unit-values header, unit-values data, water-quality header, and water-quality data files.

b. Convert files to data base load format, if necessary.

c. Data base definition.

d. Preallocate space for the data base.

e. Data base load.

f. Table indexing.

g. Data base retrieval tests using SQL.

The complete script for the DBMS test is listed in Appendix E.

4.5. TYMNET Connection

The purpose of this test was to demonstrate the capability to connect the microcomputers to TYMNET using X.25 packet switching protocol. This required a phone line connected from the microcomputer on which the test was being run into a TYMNET switching engine. Also, there was a second microcomputer located elsewhere and connected to TYMNET so that a connection between two microcomputers could be made.

SCRIPT FOR TESTING THE CONNECTION TO TYMNET

This test is designed mainly to test functionality of the TYMNET connection. Performance is not being tested. The connections must be made using a synchronous X.25 connection -- not through a TYMNET pad. There are three steps to follow.

a. Connect via TYMNET to a remote system similar to the one being called from and logon to the system. Once connected, try out a few commands to demonstrate that a satisfactory connection was made.

b. Use remote file transfer commands to transfer a small file back and forth between the two systems.

c. Attempt to connect to one of the U.S. Geological Survey Prime computers and logon to it.

5. SELECTED REFERENCES

Edwards, M.D. and others, 1986, Conceptual Design for the National Water Information System: U.S. Geological Survey Open-File Report 86-604, 37 p.

Edwards, M.D., 1987, Plan for the design, development, implementation, and operation of the National Water Information System: U.S. Geological Survey Open-File Report 87-29, 53 p.

McDonald, M.G. and Harbaugh, A.W., 1984, A Modular three-dimensional finite-difference ground-water flow model: U.S. Geological Survey Open-File Report 83-875, 528 p.

APPENDIX A

MICROCOMPUTER EVALUATION MATRIX

APPENDIX A: MICROCOMPUTER EVALUATION MATRIX

FEATURE	VENDOR /MODEL			
1. Full 32-bit memory transfer				
2. Max expandible memory(MB)				
3. Connect to LAN				
4. Connect to X.25 network				
5. Fixed disk storage(MB)				
6. Floppy disk storage(KB)				
7. Floating point hardware				
8. Distributed DB capability				
9. Centronic parallel prt port				
10. Operating system				
11. Multi-tasking				
12. Full F77 compiler				
13. F77 transparent access to 32MB virtual memory				
14. LAN supports shared DB				
15. Full screen editor				
16. Spread sheet software				
17. Relational DBMS				
18. R DBMS move data mini to PC				
19. R DBMS w 100MB databases				
20. R DBMS interface HL languag				
21. ISAM subroutine library				
22. ISAM interface to F77				

23. Shared database as own				
24. Shared DB on file server				
25. Shared DB on micro				
26. Shared DB duplicated				
27. LAN access security				
28. Rec # concurrent users				
29. Visit (V) telephone con (T) or literature review (L)				

APPENDIX B

SCRIPT FOR TESTING SHARED DATA ACCESS

APPENDIX B: SCRIPT FOR TESTING SHARED DATA ACCESS (Input/Output(I/O) Across the LAN Compared to Local I/O)

File SORT.DATA contains approximatley 10,000 80-character records. This test involves copying that file and sorting it using local disk files and disk files on another system across the LAN. The "Sort Test" consists of sorting the file based on a single key in columns 13-18 (where the first column is column 1). The copy tests will be shown in standard UNIX notation, as follows.

```
cp source_file destination_file .
```

The sort tests will be indicated by

```
Sort_Test source_file destination_file .
```

Substitute the proper syntax for the system under test.

For this test all commands were issued from one system, referred to as the local system. There was also a remote system connected via the LAN. In the pathnames below, "remote" indicates a pathname on the remote system. "Local" as part of a pathname indicates a file on the local system in the situation when the home directory is on the remote system. A simple file name indicates a file in the current home directory, which may be on the remote or local system. The specific pathnames can be changed as appropriate. All copies and sorts were timed. The tests were repeated several times to determine how repeatable the timings were. Initially, there must exist a copy of sort.data on both the local and remote systems.

For tests 1-8, the home directory was and should be on the local system.

1. `cp sort.data copy.tmp` Time: _____
2. `cp sort.data remote/copy.tmp` Time: _____
3. `cp remote/sort.data copy2.tmp` Time: _____
4. `cp remote/sort.data remote/copy2.tmp` Time: _____
5. `Sort_Test sort.data sort.tmp` Time: _____
6. `Sort_Test sort.data remote/sort.tmp` Time: _____
7. `Sort_Test remote/sort.data sort2.tmp` Time: _____
8. `Sort_Test remote/sort.data remote/sort2.tmp` Time: _____

For tests 9-10, the home directory was on the remote system. These tests were the same as tests 7-8, except that if any temporary files were created, they might be created on the remote system rather than on the local one. Any such temporary files could have a major impact on performance.

9. `Sort_Test sort.data local/sort3.tmp` Time: _____
10. `Sort_Test sort.data sort3.tmp` Time: _____
11. When done, make a comparison between sort.tmp and sorted.data to insure that the sort produced correct results.

APPENDIX C

SCRIPTS AND PROGRAMS FOR TESTING BASIC PERFORMANCE

APPENDIX C: SCRIPTS AND PROGRAMS FOR TESTING BASIC PERFORMANCE

SCRIPT ONE FOR TESTING BASIC PERFORMANCE
(C I/O Script)

Test program, writeout.c, performs some arithmetic array addressing and pipes output to program, inout.c, which reads input, performs some arithmetic array addressing, and writes out to standard output or screen. Both programs are listed following this script.

Steps to run this test are listed below.

1. Copy files off of floppy onto system disk.

```
doscp X:/compile compile
doscp X:/test.cmd test.cmd
doscp X:/gid.c gid.c
doscp X:/writeout.c writeout.c
doscp X:/inout.c inout.c
```

2. Compile and link 'C' programs using file 'compile'.

```
. compile
```

3. Send jobs to the batch queue. Send one job at a specific time, then send two jobs at a specific time and so forth up to the maximum allowable simultaneous processes. Another run should be made simultaneously with the F77 model execution.

```
at HH:MM test.cmd
at HH:MM test.cmd
     .
     .
     .
```

4. Check contents of files for list of numbers. Filenames should be in the form 'procid.HH:MM:SS' or 2304.13:23:47. Each file should be approximately 117,758 bytes in length with the beginning time in line one and ending time in last line (suggest using the UNIX command 'tail filename'). On a PC AT running Xenix, a single run took 86 seconds.

PROGRAMS FOR TESTING BASIC PERFORMANCE

1. Test program, writeout.c, does some arithmetic array addressing and pipes output to program inout.c listed in section 2 below.

```
/*      writeout
      Program to benchmark the time for 3 adds, one multiply
      and one divide and write output to stdout. */
#include <stdio.h>
main()
{
      long int i,j;
      float a,b,c,d[100];
      a = 1;
      b = 2;
      c = 3;
      for (j = 0; j < 100 ; j++)
      {
            d[0] = 1;
            for (i = 1; i < 100; i++){
                  d[i] = a + (b * c) + (c / b) + d[i-1];
                  printf ("%g\n",d[i]);
            }
      }
}
```

2. Test program, inout.c, reads input, does some arithmetic array addressing, and writes out to standard output or screen.

```
/*      inout.c
      Program to benchmark the time for 3 adds, one multiply
      and one divide, read from stdin and write to stdout. */
#include <stdio.h>
main()
{
      long int i,j;
      float a,b,c,d[100];
      a = 1;
      b = 2;
      c = 3;
      for (j = 0; j < 100 ; j++)
      {
            for (i = 1; i < 100; i++){
                  scanf ("%g", &d[i-1]);
                  d[i] = a + (b * c) + (c / b) + d[i-1];
                  printf ("inout: %g\n",d[i]);
            }
      }
}
```

NOTE: { is actually the character .
      } is actually the character .

SCRIPT TWO FOR TESTING BASIC PERFORMANCE

The second script (also referred to as the C data base script) tests input and output both to disk and to the user terminal. The test consists of timimg the compilation of the program and executing it interactively. Nothing will be timed during the execution. After execution using a file on the local system, it will be executed using a remote file on another system across the LAN to further test transparent use of remote files across a LAN. The script follows.

```
ls -l final.c
-rw-r--r--   1 201      50         15417 Apr  2 15:19 final.c
$ ls -l final.c
-rw-r--r--   1 201      50         15417 Apr  2 15:19 final.c
$ cc final.c -o final  /* The compilation/load time should be noted. */
final.c
$ ls -l final*
-rwxr-xr-x   1 201      50         15185 May 30 12:52 final
-rw-r--r--   1 201      50         15417 Apr  2 15:19 final.c
-rw-r--r--   1 201      50          5016 May 30 12:52 final.o
$ final
Usage: final filename
$ final grades   /*  Rerun entire test with file on remote system */
File: grades does not exist; creating new file

Enter ssno or control C to exit: 540568200

        Input/Edit student: 540568200

Press CTRL_A to end input
Press ENTER to accept current value between < >

Last name     <>          :Farmer
First name    <>          :Fred
Middle initial <>         :E
Phone number <>           :5551212

Enter test scores; control c to end

         1. <>            :342
         2. <>            :65
         3. <>            :43
         4. <>            :67
         5. <>            :34
         6. <>            :54
         7. <>            :7
         8. <>            :4
         9. <>            :32
        10. <>            :4

Enter homework scores; control c to end

         1. <>            :32
```

```
         2. <>            :45
         3. <>            :765
         4. <>            :34
         5. <>            :564
         6. <>            :67
         7. <>            :435
         8. <>            :43
         9. <>            :546
        10. <>            :6565
        11. <>            :34
        12. <>            :54
        13. <>            :65
        14. <>            :56
        15. <>            :342
        16. <>            :546
        17. <>            :43
        18. <>            :54
        19. <>            :56
        20. <>            :45
Final grade  <>           :A
Comment on next line :
<>
Finished all assignments

Student : 540568200     Farmer, Fred E
Phone number:   5551212
        Test scores

 1.      342
 2.      65
 3.      43
 4.      67
 5.      34
 6.      54
 7.      7
 8.      4
 9.      32
10.      4

Press ENTER to continue...
        Homework scores

 1.      32
 2.      45
 3.      765
 4.      34
 5.      564
 6.      67
 7.      435
 8.      43
 9.      546
10.      6565
11.      34
12.      54
```

```
13.      65
14.      56
15.      342
16.      546
17.      43
18.      54
19.      56
20.      45
 Final grade     : A
 Comment:
 Finished all assignments

 Store the above SRE [y/n]?y

 Enter ssno or control C to exit:
 $ ls -l grades
 -rw-r--r--   1 201      50          1030 May 30 12:54 grades
 $ final grades

                    CLASS RECORD DATA BASE SYSTEM

 Enter ssno or control C to exit: 540568200

 Input/Edit student: 540568200

 Press CTRL_A to end input
 Press ENTER to accept current value between < >

 Last name    <Farmer>   :
 First name   <Fred>     :
 Middle initial <E>      :
 Phone number <5551212>  :

 Enter test scores; control c to end

         1. <342>        :
         2. <65> :
         3. <43> :

Enter homework scores; control c to end

         1. <32> :
         2. <45> :
         3. <765>        :
         4. <34> :
         5. <564>        :
         6. <67> :
 Final grade  <A>        :

 Comment on next line :
 <Finished all assignments>

 Student : 540568200     Farmer, Fred E
 Phone number:   5551212
```

```
        Test scores

 1.      342
 2.      65
 3.      43
 4.      67
 5.      34
 6.      54
 7.      7
 8.      4
 9.      32
10.      4

 Press ENTER to continue...
        Homework scores

 1.      32
 2.      45
 3.      765
 4.      34
 5.      564
 6.      67
 7.      45
 8.      43
 9.      546
10.      6565
11.      34
12.      54
13.      65
14.      56
15.      342
16.      546
17.      43
18.      54
19.      56
20.      45
 Final grade     : A
 Comment:
 Finished all assignments

 Store the above SRE [y/n]?

 Enter ssno or control C to exit: 111111111

 Input/Edit student: 111111111

 Press CTRL_A to end input
 Press ENTER to accept current value between < >

 Last name    <>          :User
 First name   <>          :Joe
 Middle initial <>        :
 Phone number <>          :2221212
```

```
Enter test scores; control c to end

        1. <>           :432
        2. <>           :5
        3. <>           :34
        4. <>           :5
        5. <>           :

Enter homework scores; control c to end

        1. <>           :354
        2. <>           :6
        3. <>           :45
        4. <>           :5643
        5. <>           :45
        6. <>           :4
        7. <>           :3
        8. <>           :

Final grade  <>         :C
Comment on next line :
<>
Too many missing assignments

Student : 111111111     User, Joe
Phone number:   2221212
        Test scores

1.      432
2.      5
3.      34
4.      5

Press ENTER to continue...
        Homework scores

1.      354
2.      6
3.      45
4.      5643
5.      45
6.      4
7.      3
Final grade     : C
Comment:
Too many missing assignments

Store the above SRE [y/n]?y

Enter ssno or control C to exit: 222222222

        Input/edit student: 222222222

Press CTRL_A to end input
```

```
Press ENTER to accept current value between < >

Last name     <>          :Foobar
First name    <>          :Sam
Middle initial <>         :Yao
Phone number <>           :3334545

Enter test scores; control c to end

        1. <>             :34
        2. <>             :56
        3. <>             :23
        4. <>             :3
        5. <>             :0
        6. <>             :45
        7. <>             :23
        8. <>             :54
        9. <>             :

Enter homework scores; control c to end

        1. <>             :342
        2. <>             :56
        3. <>             :342
        4. <>             :65
        5. <>             :78
        6. <>             :435
        7. <>             :324
        8. <>             :564
        9. <>             :32
       10. <>             :32
       11. <>             :5
       12. <>             :65
       13. <>             :78
       14. <>             :54453
       15. <>             :
       16. <>             :67
       17. <>             :534
       18. <>             :4
       19. <>             :6
       20. <>             :43

Final grade  <>           :B
Comment on next line :
<>
Missed two tests

Student : 222222222      Foobar, Sam Yao
Phone number:    3334545
           Test scores

1.       34
2.       56
3.       23
```

```
4.      3
5.      0
6.      45
7.      23
8.      54

Press ENTER to continue...
        Homework scores

1.      342
2.      56
3.      342
4.      65
5.      78
6.      435
7.      324
8.      564
9.      32
10.     32
11.     5
12.     65
13.     78
16.     67
17.     534
18.     4
19.     6
20.     43
Final grade     : B
Comment:
Missed two tests

Store the above SRE [y/n]?y

Enter ssno or control C to exit:
$ ls -l grades
-rw-r--r--   1 201      50          1482 May 30 12:57 grades
$ final grades

                    CLASS RECORD DATA BASE SYSTEM


Enter ssno or control C to exit: 222222222

        Input/Edit student: 222222222

Press CTRL_A to end input    Press ENTER to accept current value between <
>
Last name    <Foobar>   :
First name   <Sam>      :
Middle initial <Yao>    :
Phone number <3334545>  :

Enter test scores; control c to end
```

```
        1. <34> :
        2. <56> :
        3. <23> :
        4. <3>  :
        5. <0>  :
        6. <45> :
        7. <23> :
        8. <54> :
        9. <>           :
       10. <>           :

Enter homework scores; control c to end

        1. <342>        :
        2. <56> :
        3. <342>        :
        4. <65> :
        5. <78> :
        6. <435>        :
        7. <324>        :
        8. <564>        :
        9. <32> :
       10. <32>         :
       11. <5> :
       12. <65>         :
       13. <78>         :
       14. <>           :
       15. <>           :
       16. <67>         :
       17. <534>        :
       18. <4> :
       19. <6> :
       20. <43>         :

Final grade  <B>        :
Comment on next line :
<Missed two tests>


Student : 222222222     Foobar, Sam Yao
Phone number:   3334545
        Test scores

1.      34
2.      56
3.      23
4.      3
5.      0
6.      45
7.      23
8.      54
```

```
Press ENTER to continue...
        Homework scores

1.      342
2.      56
3.      342
4.      65
5.      78
6.      435
7.      324
8.      564
9.      32
10.     32
11.     5
12.     65
13.     78
16.     67
17.     534
18.     4
19.     6
20.     43
Final grade     : B
Comment:
Missed two tests

Store the above SRE [y/n]?

Enter ssno or control C to exit: 111111111


Input/Edit student: 111111111

Press CTRL_A to end input
Press ENTER to accept current value between < >

Last name    <User>     :
First name   <Joe>      :
Middle initial <>       :
Phone number <2221212>  :

Enter test scores; control c to end

        1. <432>        :
        2. <5>  :
        3. <34> :
        4. <5>  :
        5. <>           :
        6. <>           :
        7. <>           :
        8. <>           :
        9. <>           :
       10. <>           :

Enter homework scores; control c to end
```

```
        1. <354>        :
        2. <6>  :
        3. <45> :
        4. <5643>       :
        5. <45> :
        6. <4>  :
        7. <3>  :
        8. <>           :
        9. <>           :
       10. <>           :
       11. <>           :
       12. <>           :
       13. <>           :
       14. <>           :
       15. <>           :
       16. <>           :
       17. <>           :
       18. <>           :
       19. <>           :
       20. <>           :

Final grade  <C>        :
Comment on next line :
<Too many missing assignments>

Student : 111111111     User, Joe
Phone number:   2221212
        Test scores

1.      432
2.      5
3.      34
4.      5

Press ENTER to continue...
        Homework scores

1.      354
2.      6
3.      45
4.      5643
5.      45
6.      4
7.      3
Final grade     : C
Comment:
Too many missing assignments

Store the above SRE [y/n]?

Enter ssno or control C to exit: 540568200

Input/Edit student: 540568200
```

```
Press CTRL_A to end input
Press ENTER to accept current value between < >

Last name     <Farmer>    :
First name    <Fred>      :
Middle initial <E>        :
Phone number <5551212>    :

Enter test scores; control c to end
        1. <342>          :
        2. <65> :
        3. <43> :
        4. <67> :
        5. <34> :
        6. <54> :
        7. <7>  :
        8. <4>  :
        9. <32> :
       10. <4>  :

Enter homework scores; control c to end

        1. <32> :
        2. <45> :
        3. <765>          :
        4. <34> :
        5. <564>          :
        6. <67> :
        7. <435>          :
        8. <43> :
        9. <546>          :
       10. <6565>         :
       11. <34>           :
       12. <54>           :
       13. <65>           :
       14. <56>           :
       15. <342>          :
       16. <546>          :
       17. <43>           :
       18. <54>           :
       19. <56>           :
       20. <45>           :

Final grade  <A>          :
Comment on next line :
<Finished all assignments>

Student : 540568200      Farmer, Fred E
Phone number:   5551212
        Test scores

1.      342
2.      65
3.      43
```

```
 4.      67
 5.      34
 6.      54
 7.      7
 8.      4
 9.      32
10.      4

 Press ENTER to continue...
         Homework scores

 1.      32
 2.      45
 3.      765
 4.      34
 5.      564
 6.      67
 7.      435
 8.      43
 9.      546
10.      6565
11.      34
12.      54
13.      65
14.      56
15.      342
16.      546
17.      43
18.      54
19.      56
20.      45
 Final grade     : A
 Comment:
 Finished all assignments

 Store the above SRE [y/n]?

 Enter ssno or control C to exit:
```

When rerunning script using remote filespec, note any response differences.

APPENDIX D

SCRIPTS AND PROGRAMS FOR TESTING FORTRAN 77

APPENDIX D: SCRIPTS AND PROGRAMS FOR TESTING FORTRAN 77

SCRIPT ONE FOR TESTING FORTRAN 77

Script one (also called the Training database) will test many standard Fortran 77 capabilities, including direct access input and output. Compilation will be timed, and the program executed interactively without timing. The execution will be repeated using a remote data file.

1. Compile and load program TRAINGEN.F77 (listed in Appendix D).

2. Execute TRAINGEN to create the direct access template TRAIN.DAT.

3. Compile and load program TRAIN.F77 (listed in Appendix D). These processes should be timed.

4. Execute TRAIN. '<--' marks user input.

```
              TRAINING DATA BASE UPDATE & RETRIEVAL
                       M E N U

                   1 = Enter new data
                   2 = List Training of an individual
                   3 = List Training for ALL District Personnel during year
                   4 = List all those who have taken a particular course
                   5 = DELETE ENTRY by name and date
                   6 = EDIT an entry
                   7 = EXIT Program

         Enter choice ==>
    1  <--

Enter a <CR> to return to print menu

Enter Individual name ==>
HOLLWAY, R  <--

Enter name of course ==>
GW CONCEPTS  <--

Enter course number (NNNN) ==>
1111  <--

Enter location of training ==>
PDX  <--

Enter date of training (MM/YY) ==>
05/85  <--

Enter sponsor (AAA) ==>
DIS  <--

Enter number of class hours ==>
38  <--
```

```
HOLLWAY, R     GW CONCEPTS                          1111 PDX      05/85DIS38

Is the above entry OK [Y/N]?
Y  <--
Enter a <CR> to return to print menu

Enter Individual name ==>    <--

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
2  <--

Enter individual's name ==>    HOLLWAY, R  <--

  NAME           COURSE                              CSN  LOCATION  DATE   S   HR
HOLLWAY, R     GW CONCEPTS                          1111  PDX       05/85  DIS  38
HOLLWAY, R     PRIME INFO                           1234  DEN       12/84       40

Enter a <CR> to continue...
<--

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
1  <--

Enter a <CR> to return to print menu

Enter Individual name ==>
SMITH, S  <--

Enter name of course ==>
SW MODELING  <--
```

```
Enter course number (NNNN) ==>
4444  <--

Enter location of training ==>
HQ  <--

Enter date of training (MM/YY) ==>
06/82  <--

Enter sponsor (AAA) ==>
SWB  <--

Enter number of class hours ==>
40  <--
```

NAME	COURSE	CSN	LOCATION	DATE	S	HR
SMITH, S	SW MODELING	4444	HQ	06/82	SWB	40

```
Is the above entry OK [Y/N]?
Y  <--

Enter a <CR> to return to print menu

Enter Individual name ==>
 <--

                TRAINING DATA BASE UPDATE & RETRIEVAL
                        M E N U

                     1 = Enter new data
                     2 = List Training of an individual
                     3 = List Training for ALL District Personnel during year
                     4 = List all those who have taken a particular course
                     5 = DELETE ENTRY by name and date
                     6 = EDIT an entry
                     7 = EXIT Program

Enter choice ==>
2  <--

    Enter individual's name ==>    SMITH, S  <--
```

NAME	COURSE	CSN	LOCATION	DATE	S	HR
SMITH, S	SW MODELING	4444	HQ	06/82	SWB	40

```
Enter a <CR> to continue...
<--
```

```
                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
5  <--

    Enter individual's name to be deleted ==>
    HOLLWAY, R  <--

    Enter date of course (MM/YY) ==>
    12/84  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

   HOLLWAY, R  PRIME INFO                        1234  DEN       12/84      40

Is this the entry to be deleted? [Y/N] *
Y  <--

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
2  <--

    Enter individual's name ==>    HOLLWAY, R  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR
  HOLLWAY, R    GW CONCEPTS                     1111  PDX       05/85  DIS  38

Enter a <CR> to continue...
<--
```

```
                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
6  <--

Enter name to be edited ==>
HOLLWAY, R  <--

Enter date of course (MM/YY) ==>
05/83  <--

** RECORD HOLLWAY, R       05/83 NOT FOUND IN DATA BASE

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
2  <--

Enter individual's name ==>
HOLLWAY, R  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

  HOLLWAY, R    GW CONCEPTS                       1111  PDX       05/85  DIS  38

Enter a <CR> to continue...   <--
```

```
                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
6  <--

Enter name to be edited ==>
HOLLWAY, R  <--

Enter date of course (MM/YY) ==>
05/85  <--

                            E D I T    M E N U
                        Select the corresponding number

                1 = Change Course Name
                2 = Change Course Number
                3 = Change Location
                4 = Change Date
                5 = Change Sponsor
                6 = Change Number of Hours
                7 = Write out New Record
                8 = RETURN TO MAIN MENU

Enter choice ==>
6  <--

Enter number of class hours ==>
24  <--

                            E D I T    M E N U
                        Select the corresponding number

                1 = Change Course Name
                2 = Change Course Number
                3 = Change Location
                4 = Change Date
                5 = Change Sponsor
                6 = Change Number of Hours
                7 = Write out New Record
                8 = RETURN TO MAIN MENU

Enter choice ==>
7  <--
```

```
    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

  HOLLWAY, R    GW CONCEPTS                      111    PDX      05/85 DIS  24

Is the above entry OK [Y/N]?
Y  <--

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                  M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
3  <--

Enter year for listing ==>
85  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

  HOLLWAY, R    GW CONCEPTS                     1111    PDX      05/85 DIS  24
  SMITH, S      DUMMY COURSE                    2222    PDX      05/85      10

Enter a <CR> to continue...
<--

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                  M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
3  <--

Enter year for listing ==>
84  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

Enter a <CR> to continue...
<--
```

```
                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
5  <--

Enter individual's name to be deleted ==>
HOLLWAY, R  <--

Enter date of course (MM/YY) ==>
12/85  <--

** RECORD HOLLWAY, R        12/85 NOT FOUND IN DATA BASE

                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
3  <--

Enter year for listing ==>
85  <--

    NAME          COURSE                          CSN  LOCATION  DATE   S   HR

  HOLLWAY, R    GW CONCEPTS                      1111  PDX       05/85  DIS 24
  SMITH, S      DUMMY COURSE                     2222  PDX       05/85      10

Enter a <CR> to continue...   <--
```

```
                    TRAINING DATA BASE UPDATE & RETRIEVAL
                                M E N U

                    1 = Enter new data
                    2 = List Training of an individual
                    3 = List Training for ALL District Personnel during year
                    4 = List all those who have taken a particular course
                    5 = DELETE ENTRY by name and date
                    6 = EDIT an entry
                    7 = EXIT Program

Enter choice ==>
7  <--

    **** STOP  NORMAL TERMINATION
```

5. Change the filespec to one of a remote system and rebuild and rerun TRAINGEN and TRAIN. Note any response differences.

1. Program traingen creates the direct access template TRAIN.DAT.

```
      PROGRAM TRAINGEN
C
C     THIS PROGRAM WILL INITIALIZE THE TRAIN.DAT FILE TO 500
C     BLANK RECORDS FOR USE WITH TRAIN.F77.  THIS IS A DIRECT
C     ACCESS FILE, DO NOT ATTEMPT TO LOOK AT THIS USING THE EDITOR.
C
C      WRITTEN BY: RICHARD HOLLWAY
C                  U. S. GEOLOGICAL SURVEY
C                  PORTLAND, OR
C                  JANUARY 1982
C
C      VARIABLES:
C            NAME       INDIVIDUAL NAME
C            COURSE     COURSE NAME
C            CSN        COURSE NUMBER
C            SITE       LOCATION OF TRAINING
C            DATE       DATE OF TRAINING
C            SPONSR     SPONSOR
C            HOURS      NUMBER OF COURSE HOURS
C
C      **************************************************
C
      CHARACTER COURSE*40,SITE*10,SPONSR*3,DATE*5
      INTEGER HOURS
      CHARACTER NAME*15,CSN*4
C
      HOURS = 0
      COURSE = '                                        '
      SITE = '          '
      SPONSR = '   '
      DATE = '     '
      NAME = '               '
      CSN = '    '
C
      OPEN(UNIT=5,FILE='TRAIN.DAT',ACCESS='DIRECT',STATUS='NEW',
     $      RECL=79,FORM='UNFORMATTED')
C
      DO 10 I=1,500
      K = I
      WRITE(5'K)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
10      CONTINUE
C
      STOP 'NORMAL TERMINATION '
      END
```

2. Program train updates and lists a small data base for training data in the Oregon State Office.

```
PROGRAM TRAIN
C
```

```
C     PROGRAM TO UPDATE AND LIST A SMALL DATA BASE FOR TRAINING
C     DATA IN THE OREGON DISTRICT OF WRD.
C
C     RUN TRAINGEN INITIALLY TO GENERATE TRAIN.DAT FILE.
C
C     WRITTEN BY: RICHARD HOLLWAY
C                 U. S. GEOLOGICAL SURVEY
C                 PORTLAND, OR
C                 JANUARY 1982
C
C     MODIFIED 22-NOV-82 TO MAKE READONLY VERSION USING COMPILER

           /DE OPTION
C
C     MODIFIED 6-JUN-83 FOR CONVERSION TO PRIME
C          LINKED WITH JMLAENEN>TRAINING>TRAIN.CPL FOR SORTING
           AND LISTING
C
C     VARIABLES:
C           NAME       INDIVIDUAL NAME
C           COURSE     COURSE NAME
C           CSN        COURSE NUMBER
C           SITE       LOCATION OF TRAINING
C           DATE       DATE OF TRAINING
C           SPONSR     SPONSOR
C           HOURS      NUMBER OF COURSE HOURS
C
C     ************************************************************
C
      CHARACTER COURSE*40,SITE*10,SPONSR*3,TWO,J,BLANK,F
      CHARACTER SCREEN,S,CR,ANS,Y,DATE*5,YR1*2,SEMI,H,DATE1*5
      CHARACTER BUFFER*128,PHNAME*32,LPFILE*32
      LOGICAL*2 EDIT,CPLFLG,EXISTS
      INTEGER*2 CHOICE,HOURS,PRT,ZERO
      INTEGER*2 FUNIT,CODE,ARGSL,USER
      CHARACTER NAME*15,CSN*4,NAME1*15,CSN1*4,BLANK4*4,BLANK5*5
C     COMMON/P/PRT,CHOICE,HOURS,COURSE,SITE,DATE,SPONSR,YR1
      COMMON/CH/NAME,CSN,NAME1,CSN1
C     DATA TWO,J,BLANK,S/'2','J',' ','S'/
      DATA Y/'Y'/,SEMI,H/';','H'/,BLANK4,BLANK5,ZERO/'    ','     ',0/
      DATA F/'F'/,CPLFLG/.TRUE./,FUNIT/7/
C
C     OPEN(UNIT=5,FILE='TRAIN.DAT',ACCESS='DIRECT',
     $RECL=79,STATUS='OLD',FORM='UNFORMATTED')
      LOC=0
      PRT = 1
C
5     EDIT=.FALSE.
10    FORMAT (A1)
C
15    WRITE(1,20)
20    FORMAT(////,10X,'TRAINING DATA BASE UPDATE & RETRIEVAL ',//
      $,20X,'M E N U ',//,15X,'1 = Enter new data',//,15X,'2 = List ',
      $'Training of an individual',//,15X,'3 = List Training for ',
```

```
     $'ALL District Personnel during year',//,15X,
     $'4 = List all those who have taken a particular course',//,15X,
     $'5 = DELETE ENTRY by name and date',//,15X,
     $'6 = EDIT an entry',//,15X,
     $'7 = EXIT Program',////)
     WRITE (1,30)
     30 FORMAT ('      Enter choice ==> ')
          READ(1,*,ERR=15,END=999)CHOICE
          IF(CHOICE.EQ.7)GOTO 999
          IF(CHOICE.LT.1.OR.CHOICE.GT.7)GOTO 15
          IF(CHOICE.NE.1)GOTO 300
C
C    UPDATE SECTION
C

100  LOC=LOC+1
     IF(LOC.LE.500)GOTO 102
     WRITE(1,101)
101    FORMAT(//,10X,'**  DATA BASE FULL  **',//)
     GOTO 5
102   READ(5'LOC,END=100,ERR=100)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
     IF(COURSE(1:1).NE.BLANK)GOTO 100
C
105  WRITE(1,110)
110  FORMAT(/,' Enter a <CR> to return to print menu ',//)
     WRITE (1,115)
 115 FORMAT ('Enter Individual name ==> ')
     READ(1,120,ERR=105,END=999)NAME
     IF (NAME .EQ. '') GO TO 5
120  FORMAT(A15)
C
125  WRITE(1,130)
130  FORMAT(//)
     WRITE (1,135)
 135 FORMAT ('Enter name of course ==> ')
     READ(1,140,ERR=125,END=999)COURSE
     IF (COURSE .EQ. '') GO TO 5
     IF(EDIT)GOTO 860
140  FORMAT(A40)
C
145  WRITE(1,130)
     WRITE (1,150)
 150 FORMAT ('Enter course number (NNNN) ==> ')
     READ(1,152,ERR=145,END=999)CSN
152  FORMAT(A4)
     IF(EDIT)GOTO 860
C
155  WRITE(1,130)
     WRITE (1,157)
 157 FORMAT ('Enter location of training ==> ')
     READ(1,160,ERR=155,END=999)SITE
160  FORMAT(A10)
     IF (SITE .EQ. '') GO TO 5
     IF(EDIT)GOTO 860
```

```
C
165  WRITE(1,130)
     WRITE (1,167)
 167 FORMAT ('Enter date of training (MM/YY) ==> ')
     READ(1,170,ERR=165,END=999)DATE
170  FORMAT(A5)
     IF (DATE .EQ. '') GO TO 5
C
     IF (DATE(2:2) .EQ. '' .OR. DATE(3:3) .EQ. '' .OR.
     &  DATE(4:4) .EQ. '' .OR. DATE(5:5) .EQ. '') GO TO 165
     IF(EDIT)GOTO 860
C
175  WRITE(1,130)
     WRITE (1,177)
 177 FORMAT ('Enter sponsor (AAA) ==> ')
     READ(1,180,ERR=175,END=999)SPONSR
180  FORMAT(A3)
     IF(EDIT)GOTO 860
C
185  WRITE(1,130)
     WRITE (1,190)
 190 FORMAT ('Enter number of class hours ==> ')
     READ(1,*,ERR=185,END=999)HOURS
     IF(HOURS.GT.99)GOTO 185
     IF(EDIT)GOTO 860
C
192  CONTINUE
     WRITE(1,430)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
430  FORMAT(A15,A40,A4,1X,A10,A5,A3,I2.2)
     WRITE(1,130)
     WRITE (1,440)
 440 FORMAT ('Is the above entry OK [Y/N]? ')
     READ(1,10,ERR=192,END=999)ANS
     IF(ANS.NE.Y)GOTO 100
C
     WRITE(5'LOC,END=100,ERR=100)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
     IF(EDIT)GOTO 5
     GOTO 100
C
300  IF(CHOICE.EQ.5)GOTO 700
     IF(CHOICE.EQ.6)GOTO 800
     LINES=1
C
     GOTO (400,500,600),CHOICE-1
C
400  WRITE(1,130)
     NAME1=''
     WRITE (1,410)
 410 FORMAT ('Enter individual''s name ==> ')
     READ(1,120,ERR=400,END=999)NAME1
     CALL PRINT
     GOTO 899
C
500  WRITE(1,130)
```

```
      WRITE (1,510)
  510 FORMAT ('Enter year for listing ==> ')
      READ(1,520,ERR=500,END=999)YR1
520   FORMAT(A2)
C
      CALL PRINT
      GOTO 899
C
600   WRITE(1,130)
      WRITE (1,610)
  610 FORMAT ('Enter course number for listing (NNNN) ==> ')
      READ(1,152,ERR=600,END=999)CSN1
C
      CALL PRINT
      GOTO 899
C
700   WRITE(1,130)
      KNT=0
      WRITE (1,710)
  710 FORMAT ('Enter individual''s name to be deleted ==> ')
      READ(1,120,ERR=700,END=999)NAME1
C
715   WRITE(1,130)
      WRITE (1,720)
  720 FORMAT ('Enter date of course (MM/YY) ==> ')
      READ(1,170,ERR=715,END=5)DATE1
      IF (DATE1 .EQ. '') GO TO 5
C
      IF (DATE1(2:2) .EQ. '' .OR. DATE1(3:3) .EQ. '' .OR.
     &  DATE1(4:4) .EQ. '' .OR. DATE1(5:5) .EQ. '') GO TO 715
C
      DO 750 I=1,500
      L=I
      READ(5'L,END=750,ERR=750)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
      IF(NAME.NE.NAME1)GOTO 750
      IF(DATE.NE.DATE1)GOTO 750
      LOC=I
      GOTO 760
750   CONTINUE
C
      WRITE(1,855)NAME1,DATE1
      GOTO 5
C
760   WRITE(1,430)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
      WRITE(1,130)
      WRITE(1,770)
  770 FORMAT ('Is this the entry to be deleted? [Y/N] * ')
      READ(1,10,ERR=760,END=5)ANS
      IF(ANS.NE.Y)GOTO 5
      WRITE(5'LOC)(BLANK5,K=1,3),(BLANK,K=1,40),BLANK4,(BLANK,K=1,18)
     $,ZERO
      GOTO 5
C
800   WRITE(1,130)
```

```
      WRITE (1,810)
  810 FORMAT ('Enter name to be editted ==> ')
      READ(1,120,ERR=800,END=5)NAME1
      IF (NAME1 .EQ. '') GO TO 5
C
  815 WRITE(1,130)
      WRITE (1,820)
  820 FORMAT ('Enter date of course (MM/YY) ==> ')
      READ(1,170,ERR=815,END=5)DATE1
      IF (DATE1 .EQ. '') GO TO 5
C
      IF (DATE1(2:2) .EQ. '' .OR. DATE1(3:3) .EQ. '' .OR.
     &  DATE1(4:4) .EQ. '' .OR. DATE1(5:5) .EQ. '') GO TO 815
C
      EDIT=.TRUE.
C
      DO 850 I=1,500
      L=I
      READ(5'L,ERR=850,END=850)NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
      IF(NAME.NE.NAME1)GOTO 850
      IF(DATE.NE.DATE1)GOTO 850
      LOC=I
      GOTO 860
  850 CONTINUE
      WRITE(1,855)NAME1,DATE1
  855 FORMAT(//,' ** RECORD ',A15,2X,A5,' NOT FOUND IN DATA BASE')
      GOTO 5
C
  860  WRITE(1,870)
  870  FORMAT(////,10X,' E D I T    M E N U ',//,5X,
         $      'Select the corresponding number',//,15X,
         1      '1 = Change Course Name',//,15X,
         2      '2 = Change Course Number',//,15X,
         3      '3 = Change Location',//,15X,
         4      '4 = Change Date',//,15X,
         5      '5 = Change Sponsor',//,15X,
         6      '6 = Change Number of Hours',//,15X,
         7      '7 = Write out New Record',//,15X,
         8      '8 = RETURN TO MAIN MENU',//)
      WRITE (1,880)
  880 FORMAT ('                          Enter choice ==> ')
      READ(1,*,ERR=860,END=5)CHOICE
      IF(CHOICE.LT.1.OR.CHOICE.GT.8)GOTO 860
C
      GOTO (125,145,155,165,175,185,192,5),CHOICE
C
  899 WRITE(1,130)
      WRITE (1,910)
  910 FORMAT ('Enter a <CR> to continue... ')
      READ(1,10,ERR=5,END=5)CR
      GOTO 5
C
  999   STOP ' NORMAL TERMINATION '
      END
```

```
      SUBROUTINE PRINT
C
      CHARACTER COURSE*40,SITE*10,SPONSR*3,BLANK,ESC,LBRAC,TWO,J
      CHARACTER DATE*5,YRI*2,SEMI,H
      INTEGER*2 CHOICE,HOURS,PRT
      CHARACTER NAME*15,CSN*4,NAME1*15,CSN1*4
      COMMON/P/PRT,CHOICE,HOURS,COURSE,SITE,DATE,SPONSR,YR1
      COMMON/CH/NAME,CSN,NAME1,CSN1
C
      DATA LBRAC,TWO,J,BLANK,S/'[','2','J',' ','S'/
      DATA SEMI,H/';','H'/
C
C
C     CLEAR SCREEN
C
5     WRITE (PRT,30)
30
      FORMAT(/,T4,'NAME',T19,'COURSE',T56,'CSN',T61,'LOCATION',T71,$'DATE',T76,
'SP;,T79,'HR',/,TR,'_____',T19,'_____',T56,'_____',$T61,'_____',T71,'_____',
T76,'_____',T79,'_____',/,' ')
C
C     ** DOUBLE SPACE ANNUAL REPORT **
C
      DO 50, I=1,500
      LOC=I
      READ (5'LOC,ERR=50,END=999) NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
C
      GOTO (100,200,300),CHOICE-1
C
100   IF (NAME.NE.NAME1) GOTO 50
      GOTO 500
C
300   IF (CSN.NE.CSN1) GOTO 50
C
C
500   WRITE (PRT,530) NAME,COURSE,CSN,SITE,DATE,SPONSR,HOURS
530   FORMAT(A15,A40,A4,1X,A10,A5,A3,12.2)
50    CONTINUE
C
999   RETURN
      END
```

SCRIPT TWO FOR TESTING FORTRAN 77
(Compiling and Executing a Large Model)

1. Time the compilation of file F77MODEL.F77. This is a 5000 line Fortran 77 program. It assumes the use of preallocated files, that is, there are no OPEN statements. File unit numbers for one input file and one output file are stored in variables at the beginning of the program. The input unit is stored in variable INBAS, which is 5 initially. The output unit is stored in variable IOUT, which is 6 initially. Change these as needed before compiling.

2. Load the program.

3. Time the execution of the test problem. The following files will be used on the indicated units. These have to be preallocated.

bench>runbig.bas	on unit INBAS as described above
runbigf77.list	on unit IOUT as described above
bench>runbig.bcf	on unit 7
bench>runbig.wel	on unit 8
bench>runbig.rch	on unit 9
bench>runbig.sip	on unit 10
bench>runbig.oc	on unit 11

The unit numbers used for the BCF, WEL, RCH, SIP, and OC files can be changed by modifying file RUNBIG.BAS. Print the output and save it so that accuracy can be checked.

4. Repeat steps 1-3 for file F66MODEL.FTN if possible. This is a Fortran 66 version of the same program.

5. Attempt to execute 3 copies of the model simultaneously. This will use slightly more than 4 MB of memory, so paging will be occurring. Time the execution of one of the copies to show the impact of paging.

APPENDIX E

SCRIPT FOR DATA BASE MANAGEMENT SYSTEM TEST

APPENDIX E: SCRIPT FOR DATA BASE MANAGEMENT SYSTEM TEST

Six subsets of NWIS data files will be loaded into a test data base using a relational design similar to the present NWIS design.

The data will be in a Rapport DBMS format, so it may have to be converted for input to another DBMS. Indexing and sample retrievals will be run, and the load and retrieval runs will be timed.

The overall approach of this test is to duplicate the Rapport and Oracle tests run previously by Mitre personnel under contract to the USGS NWIS.

Results will be collected in two areas,

a. Data base load times
b. Query response times

Given the limited time for testing, these results will be used to verify functionality and ease of use. Timings will be compared, but one must keep in mind that the time frame does not permit extensive data base design or tuning of the data base load and retrieval procedures.

Steps to run test

1. Six data files will be available for the test.

   For each file - the name, number of records, total number of bytes, and number of indexes are listed here.

   a. Site header file (site_hdr)

      322 records
      42,504 bytes
      9 indexes:

      site_id
      agency_id
      state_code
      county
      district
      hydro_unit
      latitude
      long_degrees || long_mm_ss || lat_long_seq
      prim_geo

   b. Agency file (agency)

      1,012 records
      56,672 bytes
      0 indexes

   c. Unit values header file (uv_header)

      2,000 records
      152,000 bytes

5 indexes:

uv_site_id
uv_parm || uv_stat
uv_parm
uv_stat
uv_year || uv_month || uv_day

d. Unit values data file (uv_data)

5,000 records
120,000 bytes
1 index:

uv_xref

e. Water-quality header file (qw_header)

2,000 records
108,000 bytes
3 indexes:

qw_site_id
qw_xref
qw_begin_yr || qw_begin_mon || qw_begin_day

f. Water-quality data file (qw_data)

8,450 records
253,500 bytes
2 indexes:

qw_xref
qw_parm

2. Convert files to data base load format, if necessary.

The data for the files listed in step 1 above exists in the format acceptable for the Rapport DBMS.

If this format is not acceptable by the DBMS being tested, this step will require a computer program to be written to read the six data files in the Rapport input format and convert each record to a format acceptable by the DBMS load program.

3. Data base definition.

Create the tables for the relational DBMS test. Listed below are the table definitions for the DBMS test. The definitions below are the Oracle DBMS create statements used in a prior test. It may be necessary to modify these definitions if a DBMS other than Oracle is used.

```
create table agency (
   agency_code     char (6),     agency_name     char (50) )
space agspace
/
create table qw_data (
   qw_xref     number (6),      qw_parm       number (5),
   qw_value    number (12,5),   qw_remarks    char (3),
   qw_prec     char (1) )
space qwdspace
/
create table qw_header (
   qw_site_id          char (16),  qw_begin_yr         number (4),
   qw_begin_mon        number (2), qw_begin_day        number (2),
   qw_begin_tim        number (4), qw_end_date         number (8),
   qw_end_time         number (4), qw_geo_unit         char (8),
   qw_medium           char (2),   qw_xref             number (6) )
space qwhspace
/
create table site_hdr (
   site_id       char (16),   agency_id       char (6),
   latitude      char (6),    long_degrees    char (3),
   long_mm_ss    char (4),    lat_long_seq    char (3),
   state_code    char (2),    district        char (2),
   county        char (4),    site_type1      char (2),
   site_type2    char (2),    site_type3      char (2),
   site_name     char (48),   prim_geo        char (8),
   hydro_unit    char (8),    create_date     number (8),
   update_date   number (8),  update_time     number (4)
space shspace
/
create table uv_data (
   uv_xref        number (8),     uv_sample_tm     number(4),
   uv_value       number (12,6) )
space uvdspace
/
create table uv_header (
   uv_site_id    char (16),     uv_cross_sec    number (12,5),
   uv_depth      number (12,5), uv_year         number (4),
   uv_month      number (2),    uv_day          number (2),
   uv_begin_tim  number (4),    uv_nr_sample    number (4),
   uv_parm       number (5),    uv_stat         number (5),
   uv_xref       number (8) )
space uvhspace
/
```

4. Preallocate the space for the data base. Record the time of the data load for the preallocation steps.

5. Data base load. Record the time of the data load for each file.

The following is an example of the Oracle data base load(odl).

```
* load the agency records (ld_ag.ctl)
```

```
define record in_ag as
 in_code     (char (6), loc (1)),
 in_name     (char(50), loc (10));

define source src_ag
 from a:\ag.fix
 length 60
 containing in_ag;

for each record
 insert into agency
 (agency_code, agency_name)
 values
  (in_code, in_name)
next record

* load the qw data records   (ld_qwd.ctl)

define record in_qwd as
 in_xref     (char (8), loc (0)),
 in_parm     (char (5), loc (10)),
 in_value    (char(12), loc (16)),
 in_remarks  (char (3), loc (30)),
 in_prec     (char (1), loc (36));

define source src_qwd1
 from a:\qwdload.dbd
 length 38
 containing in_qwd;

for each record
 insert into qw_data
 (qw_xref, qw_parm, qw_value, qw_remarks, qw_prec)
 values
 (in_xref, in_parm, in_value, in_remarks, in_prec)
next record

* load the qw header records  (ld_qwh.ctl)

define record in_qwh as
 in_site     (char (9), loc (1)),
 in_begyr    (char (4), loc (19)),
 in_begmo    (char (2), loc (24)),
 in_begday   (char (2), loc (27)),
 in_begtm    (char (4), loc (30)),
 in_enddt    (char (4), loc (35)),
 in_endtm    (char (4), loc (40)),
 in_geo      (char (8), loc (46)),
 in_medium   (char (2), loc (57)),
 in_xref     (char (8), loc (61));

define source src_qwh
 from a:\qwh.fix
 length 70
```

```
 containing in_qwh;

for each record
 insert into qw_header
 (qw_site_id, qw_begin_yr, qw_begin_mon, qw_begin_day,
  qw_begin_tim, qw_end_date, qw_end_time, qw_geo_unit,
  qw_medium, qw_xref)
 values
  (in_site, in_begyr, in_begmo, in_begday,
   in_begtm, in_enddt, in_endtm, in_geo,
   in_medium, in_xref)
next record

* load the site header records  (ld_sh.ctl)

define record in_sh as
 in_site     (char (9), loc (1)),
 in_agency   (char (4), loc (20)),
 in_lat      (char (6), loc (29)),
 in_ldeg     (char (3), loc (38)),
 in_lmmss    (char (4), loc (44)),
 in_llseq    (char (2), loc (51)),
 in_state    (char (2), loc (57)),
 in_dist     (char (2), loc (62)),
 in_county   (char (4), loc (67)),
 in_t1       (char (2), loc (74)),
 in_t2       (char (2), loc (79)),
 in_t3       (char (2), loc (84)),
 in_name     (char(48), loc (89)),
 in_geo      (char (8), loc (140)),
 in_hydro    (char (8), loc (151)),
 in_cdt      (char (6), loc (161)),
 in_udt      (char (6), loc (168)),
 in_utm      (char (4), loc (175));

define source src_sh
 from c:\oracle\eval\sh.fix
 length 180
 containing in_sh;

for each record
 insert into site_hdr
 (site_id, agency_id, latitude, long_degrees, long_mm_ss,
  lat_long_seq, state_code, district, county, site_type1,
  site_type2, site_type3, site_name, prim_geo, hydro_unit,
  create_date, update_date, update_time)
 values
  (in_site, in_agency, in_lat, in_ldeg, in_lmmss,
   in_llseq, in_state, in_dist, in_county, in_t1,
   in_t2, in_t3, in_name, in_geo, in_hydro,
   in_cdt, in_udt, in_utm)
next record

* load the uv data records  (ld_uvd.ctl)
```

```
define record in_uvd as
 in_xref      (char (8), loc (0)),
 in_samp      (char (4), loc (9)),
 in_val       (char(12), loc (14));

define source src_uvd
 from a:\uvd1.fix
 length 30
 containing in_uvd;

for each record
 insert into uv_data
 (uv_xref, uv_sample_tm, uv_value)
 values
  (in_xref, in_samp, in_val)
next record

* load the unit values header records   (ld_uvh.ctl)

define record in_uvh as
 in_site      (char(16), loc (1)),
 in_xsec      (char(12), loc (19)),
 in_depth     (char(12), loc (32)),
 in_year      (char (4), loc (45)),
 in_mon       (char (2), loc (50)),
 in_day       (char (2), loc (53)),
 in_time      (char (4), loc (56)),
 in_samp      (char (4), loc (61)),
 in_parm      (char (5), loc (66)),
 in_stat      (char (5), loc (72)),
 in_xref      (char (8), loc (78));

define source src_uvh
 from c:\rapport\lnk\uvh.fix
 length 90
 containing in_uvh;

for each record
 insert into uv_header
 (uv_site_id, uv_cross_sec, uv_depth, uv_year, uv_month, uv_day,
  uv_begin_tim, uv_nr_sample, uv_parm, uv_stat, uv_xref)
 values
  (in_site, in_xsec, in_depth, in_year, in_mon, in_day,
   in_time, in_samp, in_parm, in_stat, in_xref)
next record
```

6. Table indexing.

   Record the times of the creation of the indexes.

7. Data base retrieval tests using the interactive query language (SQL).

Record the time for execution of each retrieval. Sample SQL commands to execute the first two retrieval queries are given. However, the specific values for query parameters, e.g. station identification numbers, must be changed to fit the particular data supplied. SQL commands for the remaining retrievals are not available at this time. These commands will be supplied in advance of the test. Otherwise, they will be generated at the time of the test.

R6. uvh,uvd*

query: print the maximum, mininum, and mean temperature and ph for a specified group of stations in colorado (provide a range of station numbers.)

```
* sql query R6

select uv_header.uv_site_id,min(uv_data.uv_value),
max(uv_data.uv_value),avg(uv_data.uv_value)
from uv_header, uv_data
where uv_header.uv_site_id between ' 066000000' and
      ' 99999999' and
      uv_header.uv_parm = 60 and
      uv_header.uv_stat = 11 and
      uv_header.uv_xref = uv_data.uv_xref
group by uv_header.uv_site_id
```

R7. sh,uvh,uvd*

query: print all unit values collected in county 059 in colorado with parameter code 60 and statistic code 11.

```
* sql query r7

select uv_header.uv_site_id,uv_data.uv_value
from uv_header, uv_data, site_hdr
where site_hdr.state_code='08' and
      site_hdr.county='059' and
      uv_header.uv_site_id = site_hdr.site_id and
      uv_header.uv_parm=60 and uv_header.uv_stat=11 and
      uv_header.uv_xref = uv_data.uv_xref
group by uv_header.uv_site_id, uv_data.uv_value
```

R8. sh,uvh,uvd *

query: count and print the number of days for which discharges between 5 cfs and 500 cfs were recorded for stations in county 059 in colorado for the period 1 october 1959 to 30 september 1982.

R9. qwh,qwd *

query: print the sample dates, all parameters, values, and remarks for a list of specified stations if specific conductance is collected for the station.

R10. sh,qwh,qwd *

query: print the station numbers, parameters, values, and remarks for qw stations within a specified range of station numbers during a specified time period.

R11. sh,qwh,qwd *

query: retrieve all qw parameters and values for a specified range of station numbers, during a specified time period, for a list of specified parameters. print the minimum and maximum values for each unique parameter retrieved.

R12. sh,qwh,qwd *

query: retrieve all qw parameters, values and remarks for stations in colorado during a specified time period. print all retrieved data and a count of the number of unique station numbers retrieved for the period.

R13. sh,qwh,qwd *

query: print a list of all stations in colorado where qw parameters 681 and 691 are collected.

R14. sh,qwh *

query: print a list of all unique geologic unit codes in colorado and a total count of unique codes.

NOTE: * The abbreviations used in the retrieval tests have the following meanings:

sh - site header file
uvh - unit values header file
uvd - unit values data file
qwh - water-quality header file
qwd - water-quality data file