ATSUKO COMPUTING
INTERNATIONAL

ACI-042985-FR

April 29, 1985

National A[illegible]autics and Space Administration
George C. [illegible]all Space Flight Center
Marshall [illegible] Flight Center, Alabama 35812

Attention: AP29-H

Subject: Final Report - (Purchase Order H-78185B)

Dear Sir,

Atsuko Computing International (ACI) is pleased to submit this Final Report for Purchase Order H-78185B, entitled "System Enhancements of Mesoscale Analysis and Space Sensor (MASS) Computer System", as an Enclosure to this letter.

This work was initiated March 1, 1985 and completed April 26, 1985. If you have any questions concerning this report, please contact me at (205) 533-7590 (ACI Office) or 453-0400 (NASA Work area).

Sincerely,

ATSUKO COMPUTING INTERNATIONAL

John S. Hickey
Principal Investigator

JSH/jh

Enclosure: Final Report

Copies of Enclosure to:

AP29-H	1 (Letter only)
AS24D	5
AT01	1
CC01/Wofford	1
EM13A-15	1
ED44/MacLean	3
NASA Scientific & Technical Information Facility	1 + repro

(NASA-CR-171463) SYSTEM ENHANCEMENTS OF MESOSCALE ANALYSIS AND SPACE SENSOR (MASS) COMPUTER SYSTEM Final Report (Atsuko Computing International) 31 p HC A03/MF A01 CSCL 09B G3/62

N85-30708

Unclas
21681

ATSUKO COMPUTING
INTERNATIONAL

ACI-042985-FR

SYSTEM ENHANCEMENTS OF MESOSCALE ANALYSIS AND SPACE SENSOR (MASS) COMPUTER SYSTEM

FINAL REPORT

Prepared for:

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
GEORGE C. MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, ALABAMA 35812

Attention:

AP29-H

Under Purchase Order:

H-78185B

Prepared by:

John S. Hickey
Shogo Karitani

April 29, 1985

PREFACE

This is the Final Report prepared by Atsuko Computing International (ACI), under Purchase Order H-78185B, entitled "System Enhancements of Mesoscale Analysis and Space Sensor (MASS) Computer System", for the Atmospheric Sciences Division of the Marshall Space Flight Center. The NASA technical monitor for this contract is Ms. Laura MacLean/ED44.

Prepared by:

John S. Hickey — Date: 4/29/85

Shogo Karitani — Date: 4/29/85

TABLE OF CONTENTS

1. INTRODUCTION

Atsuko Computing International (ACI) is pleased to submit this Final Report under Purchase Order H-78185B, entitled "System Enhancements of Mesoscale Analysis and Space Senor (MASS) Computer System", to the Atmospheric Sciences Division of the Systems Dynamics Laboratory at the Marshall Space Flight Center.

The Atmospheric Sciences Division (ASD) is currently involved in the interactive information processing for the Mesoscale Analysis and Space Sensor (MASS) program. Specifically, the ASD is concerned with the developemnt and implementation of new space-borne remote sensing technology to observe and measure atmospheric processes. These space measurements and conventional observational data are being processed together to gain an improved understanding of the mesoscale structure and dynamical evolution of the atmosphere relative to cloud developemnt and precipitation processes.

To satisfy the ASD's vast data processing requirements, a Research Computer System consisting of three primary computers was developed (HP-1000F, Perkin-Elmer 3250, and Harris/6) which provides over twenty scientists with a wide range of capabilities for processing and displaying interactively large volumes of remote sensing data.

ACI personnel have been directly involved in the design, developement, and integration of both the software and hardware of the MASS Research Computer System. ACI's major effort has been to develop a MASS Data Base Management and Analysis System on the HP-1000F computer and then to extend these capabilities by integration with the Perkin-Elmer and Harris/6 computers using the MSFC's Apple III microcomputer workstations.

The primary objective of this research study performed by ACI were to design hardware enhancements for computer integration and to provide data conversion and transfer between machines. A sequence of tasks performed by ACI under this contract to accomplish these objectives follows:

- o -- Design and determine the requirements necessary to implement a "Patch Panel" communication network for the Atmospheric Sciences Division's MASS Computer System.
- o -- Initialize the HP-1000F computer system disc structure to utilize the RTE-VI operating system "Command Interpreter" file strucutre.
- o -- Develop utility routines to convert various data word formats (16-bit, 24-bit, and 32-bit) from one ASD computer to another (HP-1000F, Harris/6, and Perkin-Elmer 3250) to allow for the transfer of large data between machines.

1.1 OVERVIEW

The remainder of this report documents and summarizes the results of the entire contract work, including recommendations and conclusions based on experienced and results obtained. Appendix A provides detailed "patch panel" design and Appendix B provides "source code" listing of the utility routines developed.

2.0 OVERALL RESULTS

During this research study entitled, "System Enhancements of Mesoscale Analysis and Space Sensor (MASS) Computer System", ACI performed all tasks as defined within the purchase order and details the results of each task in the following subsections.

2.1 DESIGN "PATCH PANEL" COMMUNICATION NETWORK

ACI has designed a "patch panel" communications network and identified the requirements necessary to implement such a network for the Atmospheric Sciences Division's MASS computer system. The following items have been identified and designed (see Appendix A):

- o -- Patch Panel Cabling Diagram
- o -- RJ11 (wall mount) to MOD-TAP (502-100)
- o -- MOD-TAP (502-100)
- o -- Office to Computer Room Cable
- o -- Apple III to RJ11 Cable
- o -- HP MUX tp RJ11 Cable
- o -- PE MUX to RJ11 Cable

2.2 INITIALIZE HP-1000F RTE-VI "COMMAND INTERPRETER" DISC STRUCTURE

ACI has initialize the HP-1000F RTE VI operating system "Command Interpreter" disc file structure. Currently of the HP 7933 (400MB) disc has been formatted and initialized to allow for this feature. Only after extensive testing of all the features provided by the Command Interpreter should all the HP disc be initialized and converted over.

Both FMGR and CI files are accessible under the CI mode, while only FMGR files are accessible under FMGR mode, therefore currently all files can be accessed even though not converted to CI structure. However, once converted to CI, they will not be accessible from FMGR. Table 2-1 shows the current FMGR and CI disc structures.

2.3 DEVELOP UTILITY ROUTINES TO CONVERT DATA

ACI has developed utility routines to convert various data word formats (16-bit, 24-bit, and 32-bit) from the ASD computers (HP-1000F, Harris/6, and Perkin-Elmer 3250) to allow for the transfer of large data sets between machines. Table 2-2 details the specific routines developed and functions provided. A complete "source code" listing is provided in Appendix B.

3.0 CONCLUSIONS & RECOMMENDATIONS

In summary the "patch panel" design by ACI should be easily implemented (both timewise and cost) to satisfy the current and future needs of the user's. The CI disc structure has been tested and working for a subset of the HP 400mb disc. It is recommended only after complete testing and agreement by all users, that all the HP disc be converted to the CI structure. The utility routines developed are currently available and additonal routines should be developed as a need requires.

TABLE 2-1 · HP Disc Structure

HP 7900 (5mb) DISC UTILIZATION

LU#	LABLE	#TRKS	TYPE	FRMT	USER/DESCRIPTION
36	ED44	203	FIXED	FMGR	ED44 USERS
37	CRN#37	203	REMOVE	FMGR	USER'S PACK

HP 7906 (20mb) DISC UTILIZATION

LU#	LABLE	#TRKS	TYPE	FRMT	USER/DESCRIPTION
02	SYSTEM	203	FIXED	FMGR	SYSTEM (RESERVE)
03	AUXSYS	193	FIXED	FMGR	SYSTEM (RESERVE)
31	NORM	203	FIXED	FMGR	NORM (RESERVE)
32	CSC	203	FIXED	FMGR	CSC USER'S
33	ED42	203	REMOVE	FMGR	ED42 USER'S
34	BIGGIE	398	REMOVE	FMGR	GENERAL USER'S
35	HPSOFT	203	REMOVE	FMGR	HP-SYS SOFTWARE

HP 7925 (120mb) DISC UTILIZATION

LU#	LABLE	#TRKS	TYPE	FRMT	USER/DESCRIPTION
44	IMGDTA	1200	REMOVE	FMGR	IMAGE DATA
45	SNDDTA	1600	REMOVE	FMGR	SOUNDING DATA
46	SGLDTA	1200	REMOVE	FMGR	SINGLE LEVEL DATA
47	GRDDTA	1200	REMOVE	FMGR	GRID DATA
48	TMPDTA	400	REMOVE	FMGR	TEMPORARY DATA
49	UTILLB	400	REMOVE	FMGR	UTILITY LIBRARY
50	TY6PRG	300	REMOVE	FMGR	TYPE 6 PROGRAMS
51	USER01	105	REMOVE	FMGR	HICKEY (RES)
52	USER02	105	REMOVE	FMGR	ROTHERMEL (RES)
53	USER03	105	REMOVE	FMGR	MEYER (RES)
54	USER04	105	REMOVE	FMGR	ARNOLD (RES)
55	USER05	105	REMOVE	FMGR	WILSON (RES)
56	USER06	105	REMOVE	FMGR	ROBERTSON (RES)
57	USER07	105	REMOVE	FMGR	JEDLOVEC (RES)
58	USER08	105	REMOVE	FMGR	GOODMAN (RES)
59	USER09	105	REMOVE	FMGR	KARITANI (RES)

HP 7933 (400mb) DISC UTILIZATION

LU#	LABLE	#TRKS	TYPE	FRMT	USER/DESCRIPTION
40	DATA01	12000	REMOVE	FMGR	DATA TYPE 1 (RES)
41	DATA02	6000	REMOVE	FMGR	DATA TYPE 2 (RES)
42	DATA03	3300	REMOVE	FMGR	DATA TYPE 3 (RES)
43	DATA04	3386	REMOVE	CI	DATA TYPE 4 (RES)

TASK 1. McIDAS HARRIS/6 to HP-1000F

INT24 - Converts a 24-bit right justified McIDAS grid data from Harris/6 to HP-1000F two's compliment 24-bit integer number.

INT16 - Converts a 24-bit integer*4 right justified number to two 16-bits integer*2 numbers.

INT12 - Converts a 24-bit integer*4 right justified number to two 12-bit integer*2 numbers (8 and 16 bits).

INT08 - Converts a 24-bit integer*4 right justified number to two 16-bit integer*2 numbers (16 and 8 bits).

PAK - Packs logical*1 8-bit data into HP-1000F 16-bit word.

PAK16 - Packs two 16-bit words (using only right 8-bits) into one 16-bit word.

TASK 2. HP-1000F to McIDAS HARRIS/6

TAPEW - Writes a McIDAS grid file tape.

IHP2P - Converts two 16-bit integer*2 HP numbers using lower 12-bits into 24-bit integer*4 number.

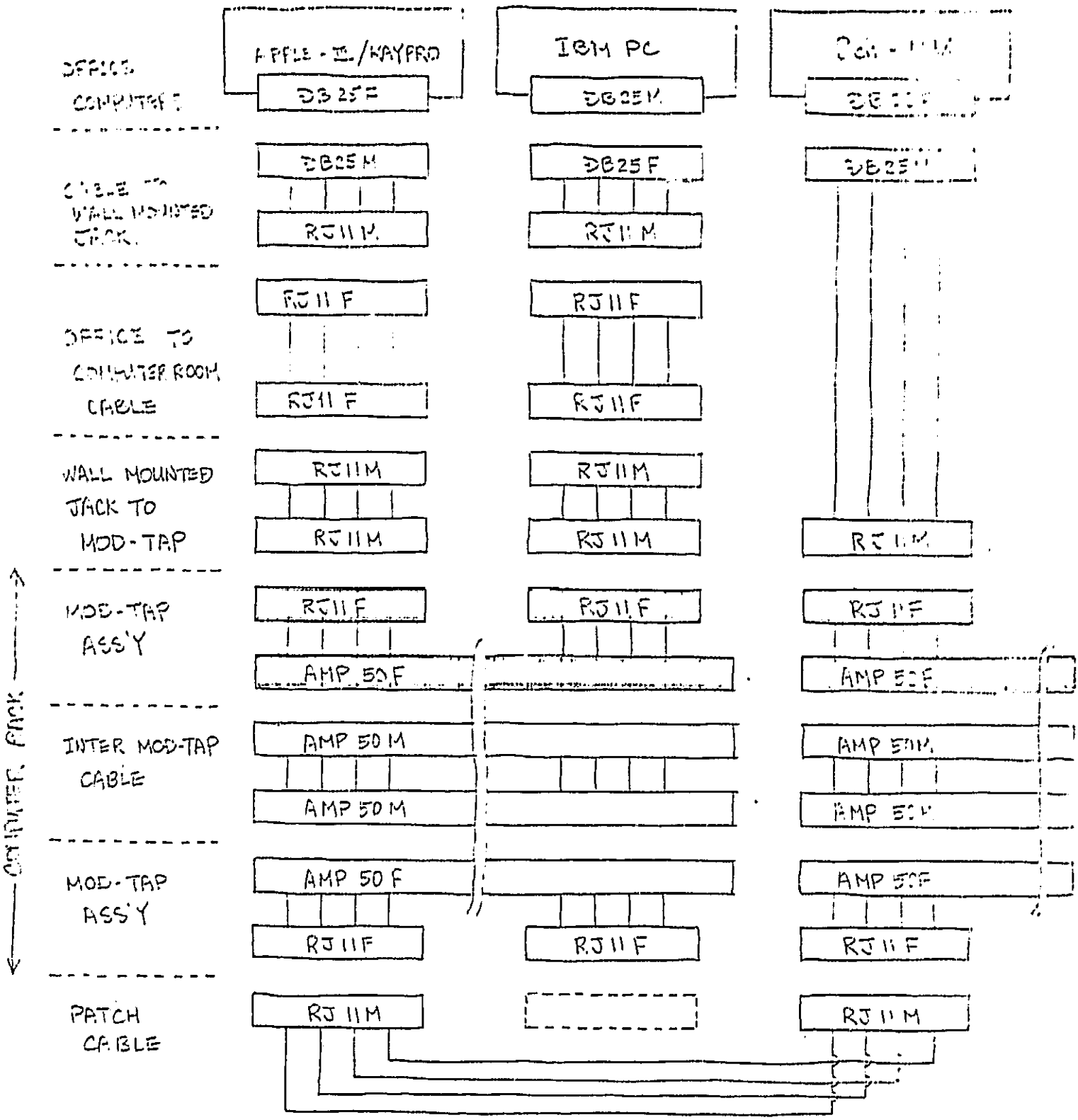
IHP3P - Converts three 16-bit integer*2 HP numbers using lower 8-bits into 24-bit integer*4 number.

IHP12 - Converts two 12-bit integer*2 HP numbers to 24-bit integer*4.

IHP24 - Converts HP two's complement integer*4 word into a 24-bit right justified negative number.

TABLE 2-2 HP-1000F Utility Routines

APPENDIX A -- PATCH PANEL DESIGN

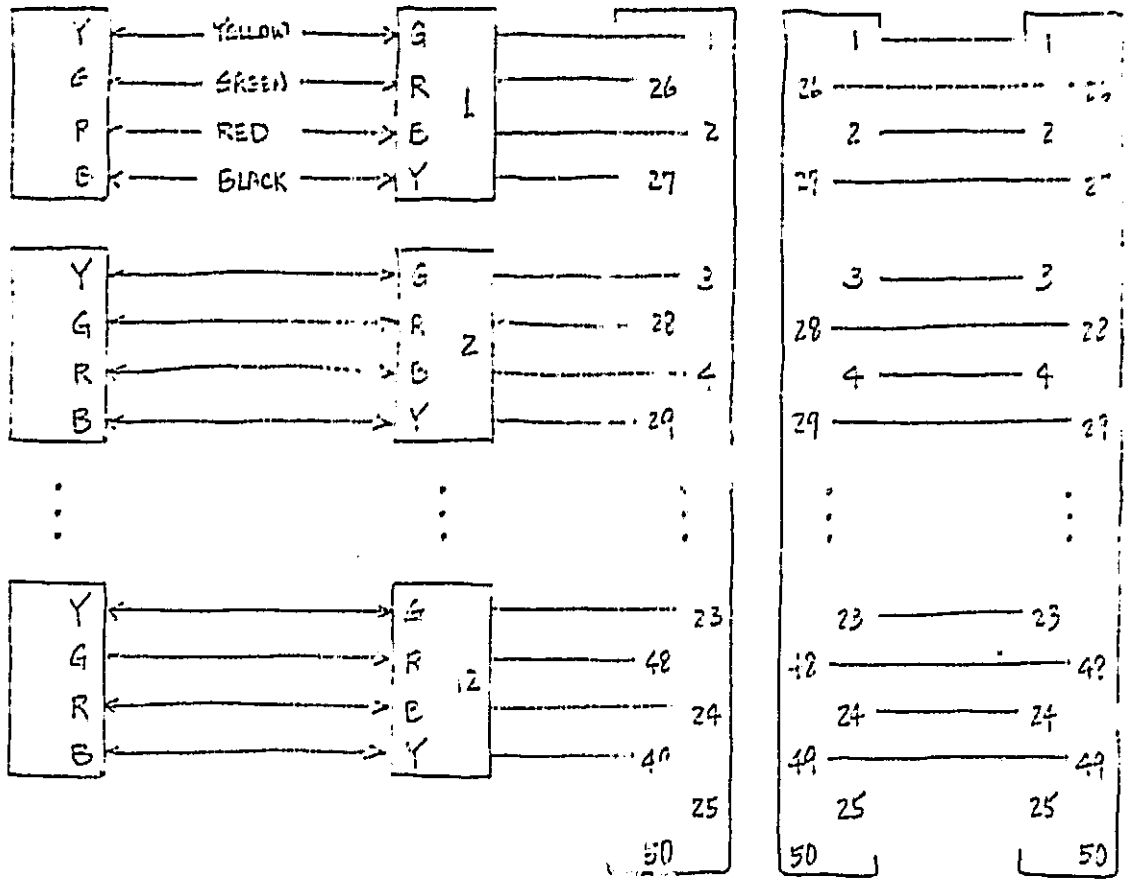


PATCH PANEL CABLING DIAGRAM

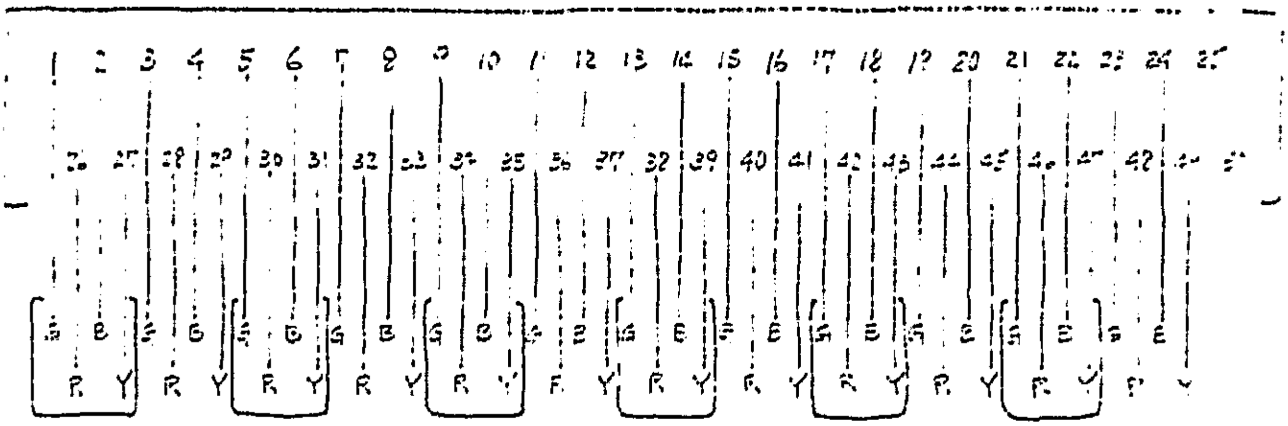
Date

PC II
(Wire Wrap)

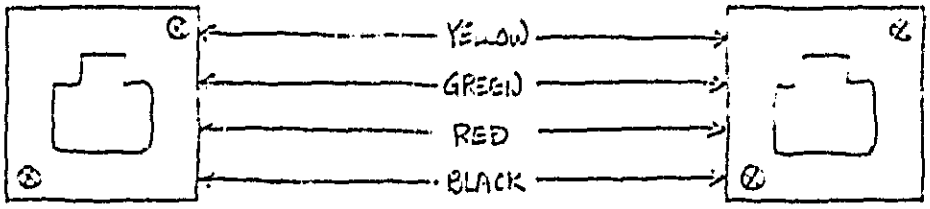
MDD-TI-P 502-100



MODE-TAP 502-100

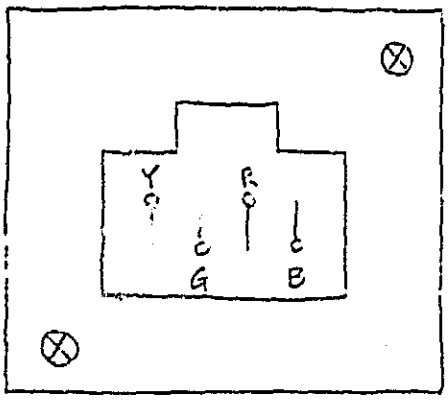


Office to Computer Room Cable



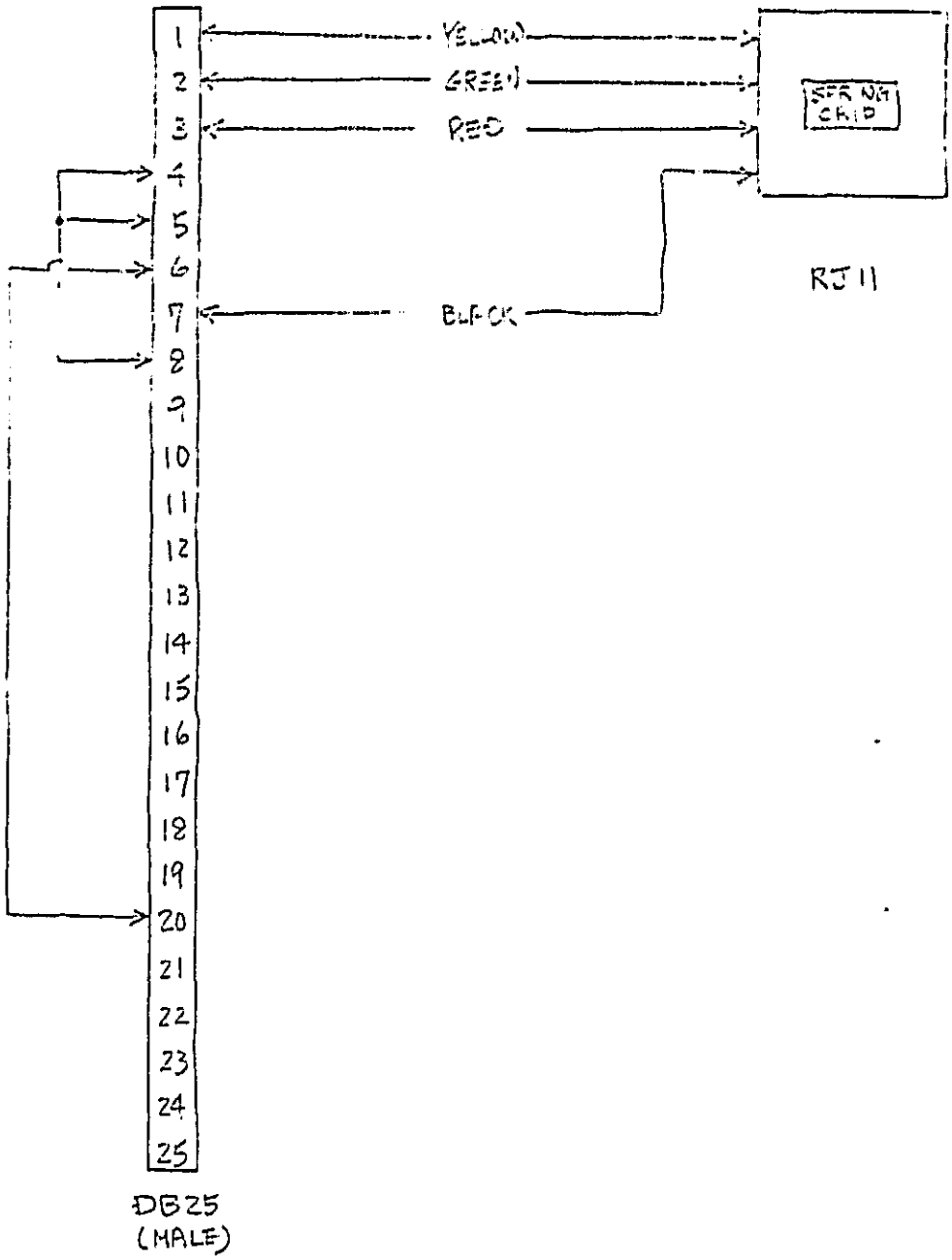
RJ 11
(WALL MOUNT)

RJ 11
(WALL MOUNT)



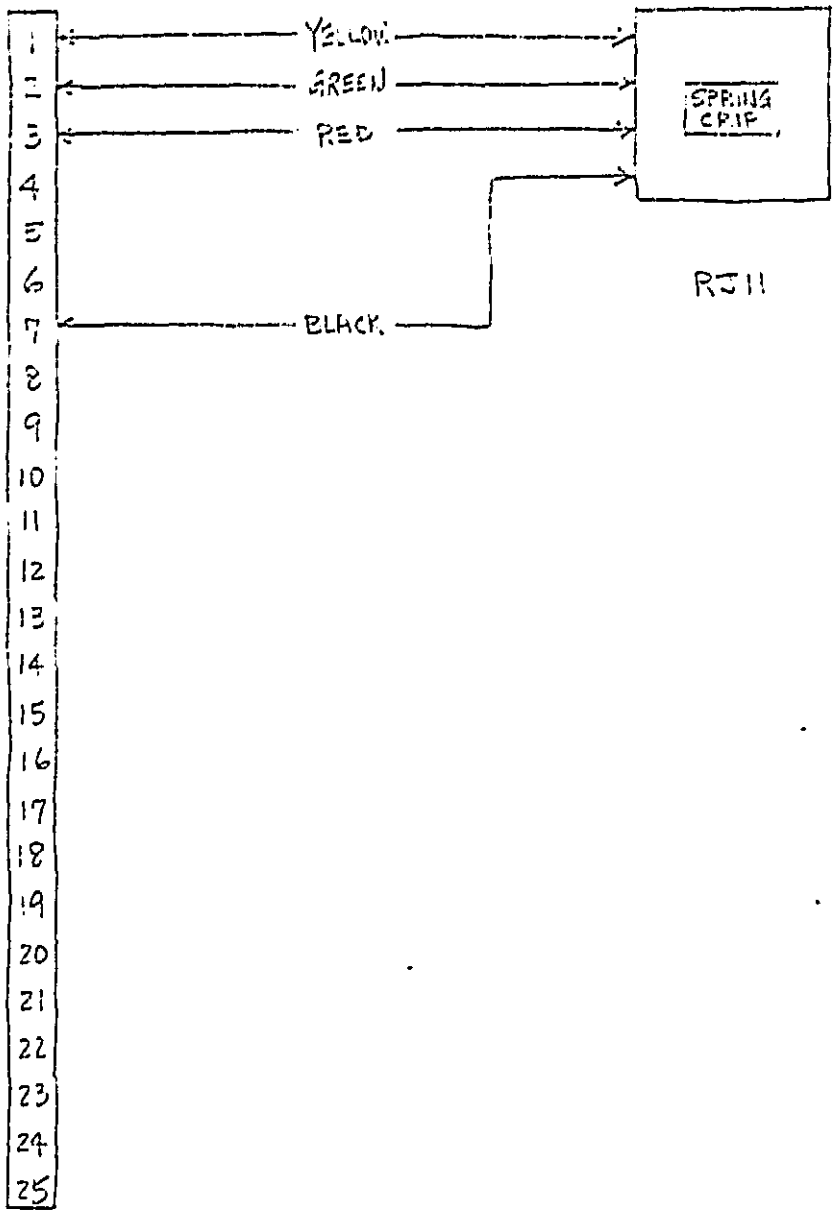
Date

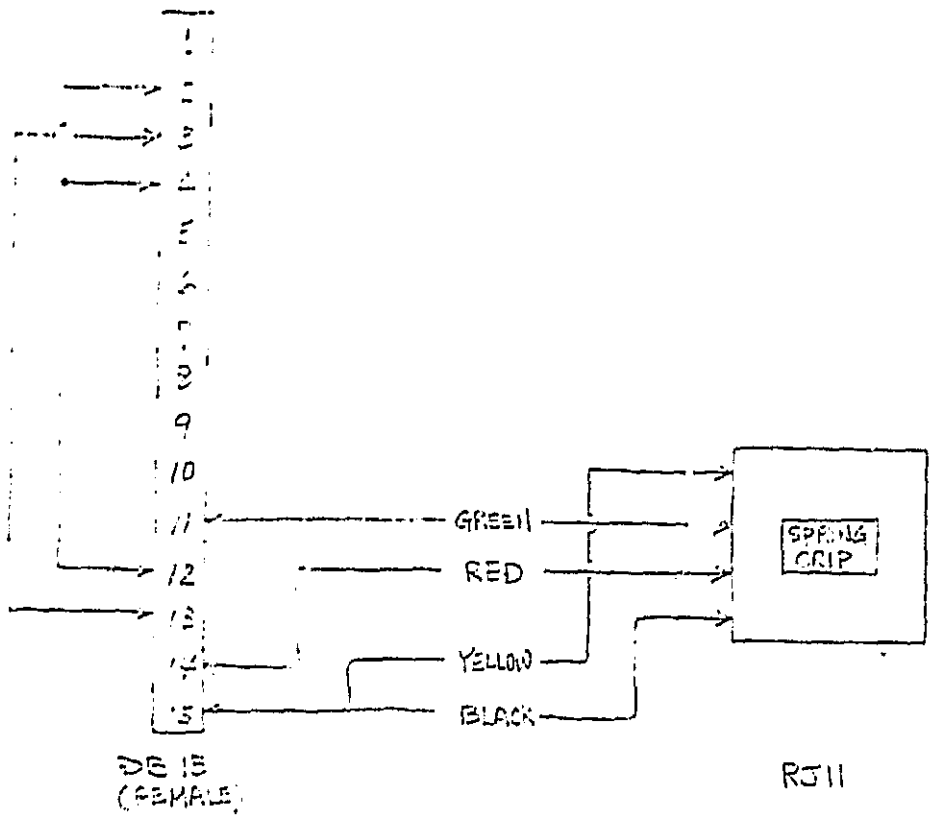
APPLE II to RJ11 Cable



Witnessed & Understood by me,	Date	Invented by:

HP MUX to RJ11 Cable




GREEN
RED
YELLOW
BLACK
SPRING GRIP
DB 15 (FEMALE)
RJ11

APPENDIX B -- HP-1000F UTILITY ROUTINES

```
      SUBROUTINE TAPEW(IARRY,ILEN,IFRMT)
C*************************************************************
C** SUBROUTINE 'TAPEW' WRITES A McIDAS GRID FILE TAPE       **
C**                                                         **
C**    IARRY --  IS INPUT ARRAY INTEGER*4                   **
C**     ILEN --  IS LENGTH OF INPUT ARRAY                   **
C**    IFRMT --  FORMAT OF INPUT ARRAY                      **
C** where:                                                  **
C**     = 0  --  END OF DATA, WRITE EOF TO TAPE             **
C**     = 1  --  112 WORD HEADER RECORD                     **
C** with      FORMAT(A3,6R3,2A3,13R3,2A3,86R3)              **
C**     = 2  --  112 WORD GRID FILE SIZE DIRECTORY          **
C**               16 WORD GRID DIRECTORY ENTRIES            **
C**              (UP TO 7 PER 112 WORD SECTOR)              **
C**               112 WORD GRID DATA PER SECTOR             **
C**              (UP TO 560 WORDS --> 5 SECTORS)            **
C** with     FORMAT(n112R3,nn(2R3,A3,13R3), nnR3)          **
C*************************************************************
C**  WRITTEN BY:   JOHN S. HICKEY (ACI)                     **
C**  DATE:         FEBRUARY 13, 1985                        **
C**  REVISED:      MARCH 8, 1985                            **
C*************************************************************
C
C** TYPE STATEMENTS
C
      DIMENSION IBUF(640),IFMT(30)
      COMMON/GRD/IGFLG,IGMAX,IGCNT
      COMMON/GMAX/MNGRID
      INTEGER*4 IARRY(  1),IPLUS
C
C** DATA STATEMENTS
C
      DATA IPLUS/8388607/
C
C++++++++++++++++++++++++++++++++++++
C++ CHECK FOR TYPE '0' FORMAT?    -+-
C++++++++++++++++++++++++++++++++++++
C
      IF(IFRMT.EQ.0) THEN
C
C** WRITE END-OF-FILE TO TAPE
C
        ICHWD = 110B
        CALL EXEC(3,ICHWD)
      ENDIF
C
C** IF TYPE '0' FORMAT RETURN TO CALLING PROGRAM
C
      IF(IFRMT.EQ.0) RETURN
C
C++++++++++++++++++++++++++++++++++++
C++ CHECK FOR TYPE '1' FORMAT?    -+-
C++++++++++++++++++++++++++++++++++++
C
      IF(IFRMT.EQ.1) THEN
```

```
 57          IGFLG = 1
 58          IGMAX = (MNGRID/7)
 59          IF(MOD(MNGRID,7).NE.0) IGMAX = IGMAX + 1
 60          IGCNT = IGMAX
 61 C
 62 C** CHECK FOR NEGATIVE NUMBERS (R3 FORMAT)
 63 C
 64          DO 10 I=1,112
 65 C
 66 C** SKIP (A3 FORMAT) DATA
 67 C
 68          IF(I.EQ.1.OR.I.EQ.10.OR.I.EQ.11.OR.I.EQ.25.OR.I.EQ.26) GO TO 10
 69          IF(IARRY(I).LE.IPLUS) GO TO 10
 70 C
 71 C** CALL 'IHP24' CONVERSION ROUTINE
 72 C
 73          CALL IHP24(IARRY(I),IARRY(I))
 74 10       CONTINUE
 75 C
 76 C** WRITE 112 WORD HEADER RECORD
 77 C
 78          CALL CODE
 79          WRITE(IBUF,1001) (IARRY(K),K=1,112)
 80 1001     FORMAT(A3,8R3,2A3,13R3,2A3,86R3)
 81          CALL EXEC(2,8,IBUF,168)
 82        ENDIF
 83 C
 84 C** IF FORMAT TYPE = '1' GO TO 999
 85 C
 86        IF(IFRMT.EQ.1) GO TO 999
 87 C
 88 C++++++++++++++++++++++++++++++++++
 89 C++ CHECK FOR TYPE '2' FORMAT?    +
 90 C++++++++++++++++++++++++++++++++++
 91 C
 92        IF(IFRMT.EQ.2) THEN
 93 C
 94 C** IF IGFLG = 1
 95 C
 96          IF(IGFLG.EQ.1) THEN
 97            IHD = 1
 98            IGFLG = IGFLG + 1
 99            ICNT = 4
100              IF((IGCNT-ICNT).GE.0) THEN
101                IENT = ICNT
102                IDSEC= 0
103                IGCNT = IGCNT - ICNT
104              ELSE
105                IENT = IGCNT
106                IDSEC= ICNT - IGCNT
107                IGCNT = 0
108              ENDIF
109 C
110 C** IF IGFLG > 1
111 C
```

```
112          ELSE
113            IHD = 0
114            IGFLG = IGFLG + 1
115            ICNT = 5
116              IF((IGCNT-ICNT).GE.0) THEN
117                IENT = 5
118                IDSEC= 0
119                IGCNT = IGCNT - ICNT
120              ELSE
121                IENT = IGCNT
122                IDSEC= 5 - IGCNT
123                IGCNT = 0
124              ENDIF
125          ENDIF
126 C
127 C** COMPUTE NUMBER OF GRID DATA POINTS
128 C
129          IDAT = IDSEC * 112
130 C
131 C** COMPUTE INDEXING OFFSET
132 C
133         IF(IGFLG.EQ.2) IOFF = 112 + IENT*112
134         IF(IGFLG.GT.2) IOFF = IENT*112
135 C
136 C** COMPUTE NUMBER OF GRID ENTRIES
137 C
138          IGSEC = IENT*7
139 C
140 C** CHECK FOR NEGATIVE NUMBERS (R3 FORMAT)
141 C** (USING OFFSET TO CHECK ONLY GRID DATA)
142 C
143          IF(IDAT.NE.0) THEN
144            DO 20 I=1,IDAT
145            INDX = I + IOFF
146            IF(IARRY(INDX).LE.IPLUS) GO TO 20
147            CALL IHP24(IARRY(INDX),IARRY(INDX))
148 20         CONTINUE
149          ENDIF
150 C
151 C** FORMULATE VARIABLE FORMAT
152 C
153 C** CHECK FOR HEADER, GRID ENTRY, AND GRID DATA
154 C
155          IF(IHD.NE.0.AND.IGSEC.NE.0.AND.IDAT.NE.0) THEN
156            ENCODE(60,21,IFMT) IHD,IGSEC,IDAT
157 21         FORMAT('(I1,'(112R3)',I3,'(2R3,A3,13R3)',I3,'(R3))')
158          ENDIF
159 C
160 C** CHECK FOR HEADER AND GRID ENTRY
161 C
162          IF(IHD.NE.0.AND.IGSEC.NE.0.AND.IDAT.EQ.0) THEN
163            ENCODE(60,22,IFMT) IHD,IGSEC
164 22         FORMAT('(I1,'(112R3)',I3,'(2R3,A3,13R3))')
165          ENDIF
166 C
```

```
167 C** CHECK FOR GRID ENTRY AND GRID DATA
168 C
169         IF(IHD.EQ.0.AND.IGSEC.NE.0.AND.IDAT.NE.0) THEN
170           ENCODE(60,23,IFMT) IGSEC,IDAT
171 23        FORMAT('(I3,'(2R3,A3,13R3)',I3,'(R3))')
172         ENDIF
173 C
174 C** CHECK FOR GRID ENTRY ONLY
175 C
176         IF(IHD.EQ.0.AND.IGSEC.NE.0.AND.IDAT.EQ.0) THEN
177           ENCODE(60,24,IFMT) IGSEC
178 24        FORMAT('(I3,'(2R3,A3,13R3))')
179         ENDIF
180 C
181 C** CHECK FOR GRID DATA ONLY
182 C
183         IF(IHD.EQ.0.AND.IGSEC.EQ.0.AND.IDAT.NE.0) THEN
184           ENCODE(60,25,IFMT) IDAT
185 25        FORMAT('(I3,'(R3))')
186         ENDIF
187 C
188 C** WRITE 560 WORD IARRY TO TAPE
189 C
190         CALL CODE
191         WRITE(IBUF,IFMT) (IARRY(K),K=1,560)
192         CALL EXEC(2,8,IBUF,840)
193       ENDIF
194 C
195 C** CHECK FORMAT TYPE > 2?
196 C
197       IF(IFRMT.GT.2) THEN
198         WRITE(1,1004) IFRMT
199 1004    FORMAT(" *** WARNING -- TYPE",I2," FORMAT NOT DEFINED ***")
200       ENDIF
201 C
202 C** ZERO OUT ARRAY
203 C
204 999   DO 1100 I=1,ILEN
205 1100  IARRY(I) = 0
206 C
207 C** RETURN TO CALLING PROGRAM
208 C
209       RETURN
210       END


FTN4X COMPILER: HP92834 REV.2303 (830113)

**  NO WARNINGS **  NO ERRORS **   PROGRAM:  1518     COMMON: (NONE)
```

```
      SUBROUTINE IHP2P(IL12,IR12,I24)
C**********************************
C** SUBROUTINE 'IHP2P' CONVERTS TWO**
C** (16-BIT) INTEGER*2 HP NUMBERS  **
C** USING ONLY THE LOWER 12-BITS   **
C** INTO A 24-BIT INTEGER*4 NUMBER **
C**********************************
C
C** TYPE STATEMENTS
C
      INTEGER*4 IL24,IR24,I24
C
C** CONVERT UPPER 12-BITS
C
      IL24 = IL12
      IL24 = IAND(ISHFT(IL24,12),77770000B)
C
C** CONVERT LOWER 12-BITS
C
      IR24 = IAND(IR12,7777B)
C
C** PACK INTO 24-BITS
C
      I24 = IOR(IL24,IR24)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 42 .COMMON: (NONE)

```
      SUBROUTINE IHP3P(IL1,IL2,IL3,I24)
C************************************
C** SUBROUTINE 'IHP3P' CONVERTS    **
C** THREE ( 3-BIT) INTEGER*2 HP    **
C** NUMBERS USING ONLY LOWER 8-BITS**
C** INTO A 24-BIT INTEGER*4 NUMBER **
C************************************
C
C** TYPE STATEMENTS
C
      INTEGER*4 IU8,IM8,IL8,I24
C
C** CONVERT UPPER  8-BITS
C
      IU8 = IL1
      IU8 = IAND(ISHFT(IU8,16),77600000B)
C
C** CONVERT MIDDLE 8-BITS
C
      IM8 = IL2
      IM8 = IAND(ISHFT(IM8,8),177400B)
C
C** CONVERT LOWER 8-BITS
C
      IL8 = IAND(IL8,377B)
C
C** PACK INTO 24-BITS
C
      I24 = IOR(IU8,IM8)
      I24 = IOR(I24,IL8)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 65 COMMON: (NONE)

```
      SUBROUTINE IHP12(IL12,IR12,I24)
C**********************************
C** SUBROUTINE 'IHP12' CONVERTS TWO**
C** 12-BIT INTEGER*2 HP NUMBERS    **
C** INTO A 24-BIT INTEGER*4 NUMBER **
C**********************************
C
C** TYPE STATEMENTS
C
      INTEGER*4 IL24,IR24,I24
C
C** CONVERT UPPER 12-BITS
C
      IL24 = IAND(ISHFT(IL12,12),77770000B)
C
C** CONVERT LOWER 12-BITS
C
      IR24 = IAND(IR12,7777B)
C
C** PACK INTO 24-BITS
C
      I24 = IOR(IL24,IR24)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 37 COMMON: (NONE)

```
      SUBROUTINE IHP24(I24,IC24)
C**********************************
C** SUBROUTINE 'IHP24' CONVERTS AN **
C** HP TWO'S COMPLEMENT INTEGER    **
C** (24-BIT) WORD INTO A (24-BIT)  **
C** RIGHT JUSTIFIED NEGATIVE NUMBER**
C**********************************
C
C** TYPE STATEMENTS
C
      INTEGER*4 I24,IC24,IMASK,ITEMP
C
C** INITIALIZE SIGN BIT MASK (24TH BIT)
C
      IMASK = 8388608
C
C** MASK OUT SIGN BIT
C
      ITEMP = IAND(I24,37777777B)
C
C** COMPLEMENT DATA
C
      ITEMP = IMASK - ITEMP
C
C** SET SIGN BIT
C
      IC24 = IOR(ITEMP,IMASK)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 37 :COMMON: (NONE)

```
      SUBROUTINE PAK16(IL8,IR8,IPK16)
C*************************************************
C** SUBROUTINE 'PAK16' TAKES TWO 16-BIT WORDS    **
C** (ONLY RIGHT 8-BITS USED) AND MAKES ONE 16-   **
C** BIT INTEGER.                                 **
C**                                             **
C** NOTE:  IL8 = 16-BIT WORD (00000000XXXXXXXX)  **
C**        IR8 = 16-BIT WORD (00000000YYYYYYYY)  **
C**      IPK16 = 16-BIT WORD (XXXXXXXXYYYYYYYY)  **
C**                                             **
C*************************************************
C
      ITMP1 = IAND(ISHFT(IL8,8),177400B)
      ITMP2 = IAND(IR8,377B)
      IPK16 = IOR(ITMP1,ITMP2)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 22 COMMON: (NONE)

```
      SUBROUTINE PAK(IA1,IA2,ILEN)
C*************************************************
C** DESCRIPTION:  SUBROUTINE 'PAK' PACKS IBM     **
C**               LOGICAL*1 8-BIT DATA INTO HP   **
C**               16-BIT WORDS.                  **
C**                                              **
C** CALLING SEQ:  CALL PAK(IA1,IA2,ILEN)         **
C**               IA1-- INPUT ARRAY CONTAINING   **
C**                     ONE 8-BIT IBM FORMATTED  **
C**                     DATA ELEMENT PER INDEXING**
C**                     ADDRESS.                 **
C**               IA2-- OUTPUT ARRAY CONTAINING  **
C**                     TWO 8-BIT HP FORMATTED   **
C**                     DATA ELEMENTS PER INDEX- **
C**                     ING ADDRESS.             **
C**              ILEN-- NUMBER OF 8-BIT DATA     **
C**                     ELEMENTS TO BE PACKED.   **
C**                                              **
C*************************************************
C
C** DIMENSION STATEMENTS
C
      DIMENSION IA1(1),IA2(1)
C
C** LOOP TO PACK 'ILEN' NUMBER OF DATA ELEMENTS
C
      DO 10 I=1,ILEN,2
      J = (I/2) + 1
      ITEMP1 = IAND(IA1(I),177400B)
      ITEMP2 = IAND(ISHIF(IA1(I+1),-8),377B)
      IA2(J) = IOR(ITEMP1,ITEMP2)
10    CONTINUE
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 61 .COMMON: (NONE)

```
391       SUBROUTINE INT08(I24,IL16,IR08)
392 C*****************************************
393 C** SUBROUTINE INT08 CONVERTS A 24-BIT   **
394 C** INTEGER*4 RIGHT JUSTIFIED NUMBER TO  **
395 C** TWO 16-BIT INTEGER*2 NUMBERS.        **
396 C**                                     **
397 C** NOTE:  IL16 = BITS 09-23             **
398 C**        IR08 = BITS  0-7              **
399 C**                                     **
400 C*****************************************
401 C
402 C** TYPE STATEMENTS
403 C
404       INTEGER*4 I24
405 C
406 C** CONVERT UPPER 16-BITS
407 C
408       IL16 = IAND(ISHFT(I24,-8 ),177777B)
409 C
410 C** CONVERT LOWER  8-BITS
411 C
412       IR08 = IAND(I24,377B)
413 C
414 C** RETURN TO CALLING PROGRAM
415 C
416       RETURN
417       END


FTN4X COMPILER: HP92834 REV.2303 (830113)

 **  NO WARNINGS **  NO ERRORS **   PROGRAM:    24       COMMON: (NONE)
```

```
      SUBROUTINE INT12(I24,IL12,IR12)
C*****************************************
C** SUBROUTINE INT12 CONVERTS A 24-BIT   **
C** INTEGER*4 RIGHT JUSTIFIED NUMBER TO  **
C** TWO 12-BIT INTEGER*2 NUMBERS.        **
C*****************************************
C
C** TYPE STATEEMENTS
C
      INTEGER*4 I24
C
C** CONVERT UPPER 12-BITS
C
      IL12 = IAND(ISHFT(I24,-12),7777B)
C
C** CONVERT LOWER 12-BITS
C
      IR12 = IAND(I24,7777B)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 23 COMMON: (NONE)

```
441       SUBROUTINE INT16(I24,IL08,IR16)
442 C*****************************************
443 C** SUBROUTINE INT16 CONVERTS A 24-BIT   **
444 C** INTEGER*4 RIGHT JUSTIFIED NUMBER TO  **
445 C** TWO 16-BIT INTEGER*2 NUMBERS.        **
446 C**                                      **
447 C** NOTE:  IL08 = BITS 16-23             **
448 C**        IR16 = BITS  0-15             **
449 C**                                      **
450 C*****************************************
451 C
452 C** TYPE STATEEMENTS
453 C
454       INTEGER*4 I24
455 C
456 C** CONVERT UPPER  8-BITS
457 C
458       IL08 = IAND(ISHFT(I24,-16),377B)
459 C
460 C** CONVERT LOWER 16-BITS
461 C
462       IR16 = IAND(I24,177777B)
463 C
464 C** RETURN TO CALLING PROGRAM
465 C
466       RETURN
467       END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 24 :COMMON: (NONE)

```
      SUBROUTINE INT24(I24,IC24)
C**********************************
C** SUBROUTINE INT24 CONVERTS A  **
C** 24-BIT RIGHT JUSTIFIED NEGA- **
C** TIVE NUMBER TO HP-TWO COMPLI-**
C** MENT 24-BIT INTEGER NUMBER.  **
C**********************************
C
C** TYPE STATEMENTS
C
      INTEGER*4 I24,IC24,IMASK,ITEMP
C
C** INITIALIZE DATA MASK
C
      IMASK = 8388608
C
C** CONVERT DATA
C
      ITEMP = IAND(I24,37777777B)
      IC24 = (IMASK-- ITEMP)*(-1)
C
C** RETURN TO CALLING PROGRAM
C
      RETURN
      END
```

FTN4X COMPILER: HP92834 REV.2303 (830113)

** NO WARNINGS ** NO ERRORS ** PROGRAM: 32 COMMON: (NONE)