HEWLETT PACKARD

HEWLETT-PACKARD COMPANY
LOGIC SYSTEMS DIVISION

HP 64000 Logic Development System

SYSTEM RELEASE BULLETIN

Part Number: 5958-6019
E0187

Printed: JANUARY 1987

```
 SSSSS     RRRRRR     BBBBBB
S     S    R     R    B     B
S          R     R    B     B
 SSSSS     RRRRRR     BBBBBB
      S    R   R      B     B
S     S    R    R     B     B
 SSSSS     R     R    BBBBBB
```

SYSTEM RELEASE BULLETIN

64000 Logic Development System

JANUARY 1987

This System Release Bulletin (SRB) documents all fixes and enhancements that are incorporated in the latest release of software for the 64000 Logic Development System.

The SRB is provided as a benefit of Hewlett-Packard's Software Support Services.

The five sections of the SRB are:

SOFTWARE RELEASE CONTENTS - lists the new revision codes for the 64000 products.

PRODUCT INDEX - lists product names and numbers which are included in this issue.

KPR NUMBER INDEX - sequential list of SR numbers.

KEYWORD INDEX - brief description of each SR.

KNOWN PROBLEM REPORTS - the actual reports.

Software release contents

Product name		Product number	uu.ff
*68000 ASSEMB		64845	01.11
*68000 ASSEMB	300	64845S004	01.10
*68000 ASSEMB	500	64845S001	01.50
*68000 ASSEMB	VAX	64845S003	01.70
*68HCII ASSEMB		64865	01.00
*68HCII ASSEMB	300	64865S004	01.00
*68HCII ASSEMB	500	64865S001	01.00
*68HCII ASSEMB	VAX	64865S003	01.00
*80286 EMULATION		64228	01.01
*8051 ASSEMB		64855	01.08
*8051 ASSEMB	300	64855S004	01.20
*8051 ASSEMB	500	64855S001	01.50
*8051 ASSEMB	VAX	64855S003	01.60
*8096 ASSEMB		64860	01.03
*8096 ASSEMB	300	64860S004	01.20
*8096 ASSEMB	500	64860S001	01.30
*8096 ASSEMB	VAX	64860S003	01.40
*F9450 EMULATION		64286	01.04
*HP TEAMWORK SA	300	64710S004	01.00
*SOFTKEY EDITOR	300	64790S004	01.00
*SW PERF ANALYZER	300	64310S004	01.10
*TMS 32010 MODULES		64285	01.01
*USER INTERFACE	300	64808S004	01.10
*USER INTERFACE	500	64808S001	01.10

Product index

Product name		Product number	Page
68000 ASSEMB	300	64845S004	1
68000 ASSEMB	500	64845S001	3
68000 ASSEMB	VAX	64845S003	5
8051 ASSEMB		64855	7
8051 ASSEMB	300	64855S004	9
8051 ASSEMB	500	64855S001	10
8051 ASSEMB	VAX	64855S003	11
8096 ASSEMB		64860	12
8096 ASSEMB	300	64860S004	15
8096 ASSEMB	500	64860S001	16
8096 ASSEMB	VAX	64860S003	17
F9450 EMULATION		64286	18

Report number index

Report #	page	Report #	page	Report #	page	Report #	page
5000129189	7	5000142463	13	D200053421	1	D200055269	18
5000129304	7	D200048314	3	D200053876	8	D200055301	16
5000133041	12	D200048322	5	D200054189	16	D200055319	17
5000133058	12	D200048330	1	D200054197	17	D200055327	15
5000133066	12	D200049296	4	D200054205	15	D200059451	2
5000133074	13	D200049304	6	D200054288	10	D200059501	2
5000133710	13	D200049312	2	D200054296	11	D200063172	19
5000136796	3	D200053405	3	D200054304	9	D200064261	19
5000141820	8	D200053413	5	D200054643	18	D200064279	19

Keyword index

- 68000 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64845S004	00.00	Linker output file should use alternate file extension.	D200049312	2
	64845S004	01.00	Macro def. including .IF, within a IF causes assembler to stop code gen.	D200053421	1
	64845S004	01.00	Link_sym file contains bad data in relocatable name record.	D200059451	2
	64845S004	01.00	"-v" option does not work with asm inside pmon	D200059501	2
MACRO	64845S004	01.00	Conditional instr. .IF with rational oper. in Macro creates bad code	D200048330	1

- 68000 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64845S001	00.00	Linker output file should use alternate file extension.	D200049296	4
	64845S001	01.40	Macro def. including .IF, within a IF causes assembler to stop code gen.	D200053405	3
ASSEMBLER	64845S001	01.40	LR error flagged for correct offset using PC+INDEX+OFFSET mode of addr.	5000136796	3
MACRO	64845S001	01.40	Conditional instr. .IF with rational oper. in Macro creates bad code	D200048314	3

- 68000 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64845S003	00.00	Linker output file should use alternate file extension.	D200049304	6
	64845S003	01.50	Macro def. including .IF, within a IF causes assembler to stop code gen.	D200053413	5
MACRO	64845S003	01.50	Conditional instr. .IF with rational oper. in Macro creates bad code	D200048322	5

- 8051 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64855	01.06	HIGH does not funct. correctly on label defined using DS	5000141820	8
CODE GENERATOR	64855	01.05	JMP command generates a SJUMP instead of a LJMP when jmping to ext label	5000129189	7
	64855	01.05	Incorrect opcode "MOV A,ACC" allowed by our assembler	5000129304	7
	64855	01.06	Relative offset from PC incorrectly calculated with the CJNE opcode	D200053876	8

- 8051 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
CODE GENERATOR	64855S004	01.00	Relative offset from PC incorrectly calculated with the CJNE opcode	D200054304	9

- 8051 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
CODE GENERATOR	64855S001	01.30	Relative offset from PC incorrectly calculated with the CJNE opcode	D200054288	10

- 8051 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
CODE GENERATOR	64855S003	01.40	Relative offset from PC incorrectly calculated with the CJNE opcode	D200054296	11

Keyword index

- 8096 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64860	01.00	DCW and DCB will not function with negative operands.	5000133041	12
	64860	01.00	SKIP listing control pseudo does not work.	5000133058	12
	64860	01.00	Linker generates incorrect absolute code.	5000133066	12
	64860	01.00	Problem with "CSEG" generating nonsence code	5000133074	13
	64860	01.00	Using indexed addressing mode with a label generates LR error.	5000133710	13
CODE GENERATOR	64860	01.02	8096 Jump instruc. w/ $ as operand access PC after instr. instead before	5000142463	13

- 8096 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64860S004	01.00	SKIP listing control pseudo does not work.	D200054205	15
	64860S004	01.00	DCW and DCB will not function with negative operands.	D200055327	15

- 8096 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64860S001	01.10	SKIP listing control pseudo does not work.	D200054189	16
	64860S001	01.10	DCW and DCB will not function with negative operands.	D200055301	16

- 8096 ASSEMB -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64860S003	01.20	SKIP listing control pseudo does not work.	D200054197	17
	64860S003	01.20	DCW and DCB will not function with negative operands.	D200055319	17

- F9450 EMULATION -

Keyword	Product number	uu.ff	Description	Report #	page
********none********	64286	01.02	Keybd Simio drops last character if odd # typed in & CA in User Memory.	D200054643	18
	64286	01.02	Symbol INT_SER_PTR must be mapped to emul. memory for "break" to work.	D200055269	18
	64286	01.03	display memory mnemonic may fail after display trace mnemonic	D200063172	19
	64286	01.03	The inverse assembler fails to show the mnemonic for a very short branch	D200064261	19
	64286	01.03	The inverse assembler may get confused in some cases.	D200064279	19

Number: D200048330 Product: 68000 ASSEMB   300 64845S004   01.00

Keywords: MACRO

One-line description:
Conditional instr. .IF with rational oper. in Macro creates bad code

Problem:
The use of the conditional instruction, .IF, with rational operator (.EQ.,.NE.,.LT.,.GT.,.LE.,.GE.) in a macro functions incorrectly. The following program demonstrates this problem:

```
       BUG          MACRO           &VAR
                    .IF &VAR .LE. 0 SUB&&&&
                    NOP
                    NOP
       SUB&&&&      NOP
                    NOP
                    MEND

                    BUG  3
                    BUG -1
                    BUG  0
                    END
```

Passing a 3 appears to create correct code, but 0 causes a ML error. Passing -1 to the MACRO creates code which doesn't call the subroutine. This is incorrect since -1 is less than 0. This same problem occured with all the rational operators on all processors. The problem was consistant on the 64000, VAX, and 9000.

Signed off 09/09/86 in release 401.10

Number: D200053421 Product: 68000 ASSEMB   300 64845S004   01.00

One-line description:
Macro def. including .IF, within a IF causes assembler to stop code gen.

Problem:
If you have a ".IF" in a macro definition and that macro definition is within a conditional assembly "IF" then no code is generated. The program provided demonstrates the problem.

```
"68000"

ESSAI          EQU            0
               IF             ESSAI

MAC            MACRO
               .IF            ESSAI.EQ.0      FIN
LABEL          MOVE           D3,D4
FIN            MEND

               MAC
               ENDIF

START          MOVE           D4,D5
```

Temporary solution:
Pull the macro definition outside of the conditional if. No code will be generated for the definition.

```
"68000"

ESSAI       EQU     0

MAC         MACRO
            .IF     ESSAI.EQ.0   FIN
LABEL       MOVE    D3,D4
FIN         MEND


            IF      ESSAI
            MAC
            ENDIF

START       MOVE    D4,D5
```

Signed off 09/09/86 in release 401.10

Number: D200059451 Product: 68000 ASSEMB   300 64845S004   01.00

One-line description:
Link_sym file contains bad data in relocatable name record.

Number: D200059501 Product: 68000 ASSEMB   300 64845S004   01.00

One-line description:
"-v" option does not work with asm inside pmon

Problem:
Note that the status messages do not increment.

Number: D200049312 Product: 68000 ASSEMB   300 64845S004   00.00

One-line description:
Linker output file should use alternate file extension.

Signed off 09/09/86 in release 401.10

Number: 5000136796 Product: 68000 ASSEMB   500 64845S001   01.40

Keywords: ASSEMBLER

One-line description:
LR error flagged for correct offset using PC+INDEX+OFFSET mode of addr.

Temporary solution:
Temporary solution:

"68000"

```
        ORG     OFFH
        MOVE    TABLE-($+2)[PC,D0],D1
TABLE   DS      1
```

Number: D200048314 Product: 68000 ASSEMB   500 64845S001   01.40

Keywords: MACRO

One-line description:
Conditional instr. .IF with rational oper. in Macro creates bad code

Problem:
The use of the conditional instruction, .IF, with rational operator (.EQ.,.NE.,.LT.,.GT.,.LE.,.GE.) in a macro functions incorrectly.
The following program demonstrates this problem:

```
       BUG            MACRO            &VAR
                      .IF &VAR .LE. 0 SUB&&&&
                      NOP
                      NOP
       SUB&&&&        NOP
                      NOP
                      MEND

                      BUG  3
                      BUG -1
                      BUG  0
                      END
```

Passing a 3 appears to create correct code, but 0 causes a ML error.
Passing -1 to the MACRO creates code which doesn't call the subroutine.
This is incorrect since -1 is less than 0. This same problem occured with all the rational operators on all processors. The problem was consistant on the 64000, VAX, and 9000.

Signed off 09/09/86 in release 101.50

Number: D200053405 Product: 68000 ASSEMB   500 64845S001   01.40

One-line description:
Macro def. including .IF, within a IF causes assembler to stop code gen.

Problem:
If you have a ".IF" in a macro definition and that macro definition is within a conditional assembly "IF" then no code is generated.

The program provided demonstrates the problem.

"68000"

```
ESSAI           EQU             0
                IF              ESSAI

MAC             MACRO
                .IF             ESSAI.EQ.0      FIN
LABEL           MOVE            D3,D4
FIN             MEND

                MAC
                ENDIF

START           MOVE            D4,D5
```

Temporary solution:
Pull the macro definition outside of the conditional if. No code will be generated for the definition.

"68000"

```
ESSAI        EQU     0

MAC          MACRO
             .IF     ESSAI.EQ.0   FIN
LABEL        MOVE    D3,D4
FIN          MEND


             IF      ESSAI
             MAC
             ENDIF

START        MOVE    D4,D5
```

Signed off 09/09/86 in release 101.50

Number: D200049296 Product: 68000 ASSEMB   500 64845S001   00.00

One-line description:
Linker output file should use alternate file extension.

Signed off 09/09/86 in release 101.50

Number: D200048322 Product: 68000 ASSEMB   VAX 64845S003   01.50

Keywords: MACRO

One-line description:
Conditional instr. .IF with rational oper. in Macro creates bad code

Problem:
The use of the conditional instruction, .IF, with rational operator (.EQ.,.NE.,.LT.,.GT.,.LE.,.GE.) in a macro functions incorrectly. The following program demonstrates this problem:

```
       BUG           MACRO            &VAR
                     .IF &VAR .LE. 0 SUB&&&&
                     NOP
                     NOP
       SUB&&&&       NOP
                     NOP
                     MEND

                     BUG -3
                     BUG  1
                     BUG  0
                     END
```

Passing a 1 appears to create correct code, but 0 causes a ML error. Passing -3 to the MACRO creates code which doesn't call the subroutine. This is incorrect since -3 is less than 0. This same problem occured with all the rational operators on all processors. The problem was consistant on the 64000, VAX, and 9000.

Signed off 09/09/86 in release 301.70

Number: D200053413 Product: 68000 ASSEMB   VAX 64845S003   01.50

One-line description:
Macro def. including .IF, within a IF causes assembler to stop code gen.

Problem:
If you have a ".IF" in a macro definition and that macro definition is within a conditional assembly "IF" then no code is generated. The program provided demonstrates the problem.

```
"68000"

ESSAI           EQU             0
                IF              ESSAI

MAC             MACRO
                .IF             ESSAI.EQ.0      FIN
LABEL           MOVE            D3,D4
FIN             MEND

                MAC
                ENDIF

START           MOVE            D4,D5
```

Temporary solution:
Pull the macro definition outside of the conditional if. No code will be generated for the definition.

```
"68000"

ESSAI        EQU      0

MAC          MACRO
             .IF      ESSAI.EQ.0   FIN
LABEL        MOVE     D3,D4
FIN          MEND


             IF       ESSAI
             MAC
             ENDIF

START        MOVE     D4,D5
```

Signed off 09/09/86 in release 301.70

Number: D200049304 Product: 68000 ASSEMB   VAX 64845S003   00.00

One-line description:
Linker output file should use alternate file extension.

Signed off 09/09/86 in release 301.70

Number: 5000129189 Product: 8051 ASSEMB 64855 01.05

Keywords: CODE GENERATOR

One-line description:
JMP command generates a SJUMP instead of a LJMP when jmping to ext label

Problem:
The following example generates a SJMP when it should generate a LJMP:

```
    "8051"
      EXT SETUP
      CSEG
      ORG 0
      JMP SETUP
      END
```

This generates an out of range error during linking when SETUP has a value such that 8 signed bits cannot refer to it relative to opcode location. Since the EXT is assigned a value of 0000 by the assembler, it assumes that any org less than 80H needs only a SJMP.

Temporary solution:
Use LJMP SETUP instead of JMP

or

Put the EXT's at the end of the file. For Example:

```
              PROG
              ORG 0
              JMP SETUP     {opcode 20000 - LJMP}
              EXT SETUP
              END
```

Signed off 10/24/86 in release 501.08

Number: 5000129304 Product: 8051 ASSEMB 64855 01.05

Keywords: CODE GENERATOR

One-line description:
Incorrect opcode "MOV A,ACC" allowed by our assembler

Problem:
The instruction "MOV A,ACC" was assemble and emulated by our products; however, the Intel 8051 goes into the weeds at this instrcution.
At first glance the machine code in the asembler listing appears valid (MOV A,ACC ->0000 E5E0), but the bottom of page 8-35 in Intel's microcontroller handbook states: *MOV A,ACC is not a valid instruction.

Neither our manuals nor AMD's user manual mention this instruction.

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 501.08

Number: 5000141820 Product: 8051 ASSEMB 64855 01.06

One-line description:
HIGH does not funct. correctly on label defined using DS

Problem:
The pseudo instruction HIGH does not function properly when operating on a label that was defined with the DS pseudo
For example:

```
     "8051"
            ORG 1234H
     LABEL1  EQU  $
     LABEL2  EQU  3344H
     LABEL3  DS   1
             MOV  A,#HIGH(LABEL1)   correct - moves 12H into A
             MOV  A,#HIGH(LABEL2)   correct - moves 33H into A
             MOV  A,#HIGH(LABEL3)   wrong - moves 34H into A
             MOV  DPTR,#LABEL3      correct - moves 1234H into DPTR
```

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 501.08

Number: D200053876 Product: 8051 ASSEMB 64855 01.06

Keywords: CODE GENERATOR

One-line description:
Relative offset from PC incorrectly calculated with the CJNE opcode

Problem:
The assembler generates B413FE for the following statement:

```
              CJNE A,#13H,$+1
```

The first byte (B4) is the correct opcode for CJNE, and the second byte (13) is the correct representation for the immediate data.
The third byte (FE) is not the correct relative offset.

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 501.08

```
Number: D200054304  Product: 8051 ASSEMB      300 64855S004        01.00

Keywords: CODE GENERATOR

One-line description:
Relative offset from PC incorrectly calculated with the CJNE opcode

Problem:
The assembler generates B413FE  for the following statement:
              CJNE A,#13H,$+1
The first byte (B4) is the correct opcode for CJNE, and the second
byte (13) is the correct representation for the immediate data.
The third byte (FE) is not the correct relative offset.

Temporary solution:
No  known temporary solution.
```

```
Number: D200054288  Product: 8051 ASSEMB      500 64855S001        01.30

Keywords: CODE GENERATOR

One-line description:
Relative offset from PC incorrectly calculated with the CJNE opcode

Problem:
The assembler generates B413FE  for the following statement:
              CJNE A,#13H,$+1
The first byte (B4) is the correct opcode for CJNE, and the second
byte (13) is the correct representation for the immediate data.
The third byte (FE) is not the correct relative offset.

Temporary solution:
No known temporary solution.
```

Number: D200054296 Product: 8051 ASSEMB VAX 64855S003 01.40

Keywords: CODE GENERATOR

One-line description:
Relative offset from PC incorrectly calculated with the CJNE opcode

Problem:
The assembler generates B413FE for the following statement:

```
              CJNE A,#13H,$+1
```

The first byte (B4) is the correct opcode for CJNE, and the second byte (13) is the correct representation for the immediate data. The third byte (FE) is not the correct relative offset.

Temporary solution:
No known temporary solution.

Number: 5000133041 Product: 8096 ASSEMB 64860 01.00

One-line description:
DCW and DCB will not function with negative operands.

Problem:
An assembler error is generated if the operand of a DCB or DCW instruction is a negative number. According to the manual, negative values in the legal range are valid operands.

```
"8096"

LABEL1     DCB    -10
                  ^IC - Illegal constant, illegal character found
                       in constant
```

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 001.03

Number: 5000133058 Product: 8096 ASSEMB 64860 01.00

One-line description:
SKIP listing control pseudo does not work.

Problem:
The SKIP listing control instruction does not work. It generates assembler errors when used in the source program.

Temporary solution:
Insert formfeeds (control L) in the listing file before printing it.

Signed off 10/24/86 in release 001.03

Number: 5000133066 Product: 8096 ASSEMB 64860 01.00

One-line description:
Linker generates incorrect absolute code.

Problem:
If the following code is assembled and linked with PROG at 2080H, the absolute code is incorrect at address 200CH. Instead of generating 2084H, it generates 2480H.

```
"8096"
    PROG
    LD     SP,#100H
L1  PUSHF
    ORG    200CH
    DCW    L1
```

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 001.03

Number: 5000133074 Product: 8096 ASSEMB 64860 01.00

One-line description:
Problem with "CSEG" generating nonsense code

Signed off 10/24/86 in release 001.03

Number: 5000133710 Product: 8096 ASSEMB 64860 01.00

One-line description:
Using indexed addressing mode with a label generates LR error.

Problem:
The following code generates a legal range (LR) error when the indexed addressing mode is used. Also, the code generated for the SJMP instruction is incorrect.

```
"8096"
AX        EQU     30H
BX        EQU     31H

          ORG     2800H
          LDB     AX,TABLE[BX]
                          ^LR ERROR
          SJMP    L1   (*The code generated is 2001, but should be 2002*)
          CLR     AX
L1:       NOP

          ORG     2986H
TABLE     DCB     0
```

Temporary solution:
No known temporary solution.

Signed off 10/24/86 in release 001.03

Number: 5000142463 Product: 8096 ASSEMB 64860 01.02

Keywords: CODE GENERATOR

One-line description:
8096 Jump instruc. w/ $ as operand access PC after instr. instead before

Problem:
All 8096 jump instructions using $ as an operand use the PC value after the jump instruction. It should use the PC value prior to this instruction executing.
For example,

```
                                                opcodes
     "8096"                                 actual      expected
     IOSO         EQU           15H
                  JGE           $           D6FF          D6FE
                  JBS           IOSO,7,$    3F15FF        3F15FD
```

Temporary solution:

No known temporary solution.

Signed off 10/24/86 in release 001.03

```
Number: D200054205  Product: 8096 ASSEMB      300 64860S004        01.00
```

One-line description:
SKIP listing control pseudo does not work.

Problem:
The SKIP listing control instruction does not work. It generates
assembler errors when used in the source program.

Temporary solution:
Insert formfeeds (control L) in the listing file before printing
it.

```
Number: D200055327  Product: 8096 ASSEMB      300 64860S004        01.00
```

One-line description:
DCW and DCB will not function with negative operands.

Problem:
An assembler error is generated if the operand of a DCB or DCW
instruction is a negative number. According to the manual,
negative values in the legal range are valid operands.

```
"8096"

LABEL1     DCB    -10
                  ^IC - Illegal constant, illegal character found
                        in constant
```

Temporary solution:
No known temporary solution.

```
Number: D200054189  Product: 8096 ASSEMB      500 64860S001        01.10
```

One-line description:
SKIP listing control pseudo does not work.

Problem:
The SKIP listing control instruction does not work. It generates
assembler errors when used in the source program.

Temporary solution:
Insert formfeeds (control L) in the listing file before printing
it.

```
Number: D200055301  Product: 8096 ASSEMB      500 64860S001        01.10
```

One-line description:
DCW and DCB will not function with negative operands.

Problem:
An assembler error is generated if the operand of a DCB or DCW
instruction is a negative number. According to the manual,
negative values in the legal range are valid operands.

```
"8096"

LABEL1     DCB    -10
                  ^IC - Illegal constant, illegal character found
                        in constant
```

Temporary solution:
No known temporary solution.

Number: D200054197 Product: 8096 ASSEMB VAX 64860S003 01.20

One-line description:
SKIP listing control pseudo does not work.

Problem:
The SKIP listing control instruction does not work. It generates assembler errors when used in the source program.

Temporary solution:
Insert formfeeds (control L) in the listing file before printing it.

Number: D200055319 Product: 8096 ASSEMB VAX 64860S003 01.20

One-line description:
DCW and DCB will not function with negative operands.

Problem:
An assembler error is generated if the operand of a DCB or DCW instruction is a negative number. According to the manual, negative values in the legal range are valid operands.

```
"8096"

LABEL1     DCB    -10
                  ^IC - Illegal constant, illegal character found
                        in constant
```

Temporary solution:
No known temporary solution.

Number: D200054643 Product: F9450 EMULATION 64286 01.02

One-line description:
Keybd Simio drops last character if odd # typed in & CA in User Memory.

Temporary solution:
Two temporary fixes exist for this problem.

1) Put the Control Address in emulation memory.

2) Modify the emulation monitor to the following:

```
Old Code:(Existing Monitor)      New Code:(Modified Monitor)

508  L R11,PRAM4  Byte Count.    508    L R11,PRAM4  Byte Count.
509  SRL R11,1    Word Count.    508.1  TBR 0,R11    Test for even count.
                                 508.2  BNZ EVEN     Even, so skip
                                 508.3  AIM R11,1    Make Count Even
                                 508.4EVEN
                                 509    SRL R11,1    Word Count.
```

Signed off 10/24/86 in release 601.04

Number: D200055269 Product: F9450 EMULATION 64286 01.02

One-line description:
Symbol INT_SER_PTR must be mapped to emul. memory for "break" to work.

Problem:
If the symbol INT_SER_PTR from the emulation boot routine is not mapped to emulation memory, then the internal jam-registers of the emulator are not programmed. The internal jam-registers are programmed with the two addresses INT_SER_PTR and LINKAGE_PTR so that the emulator will be able to break into the monitor properly. It may be desirable to map the address location INT_SER_PTR to user memory since INT_SER_PTR must be located in address state 0 and the user can very well program the three words associated with INT_SER_PTR in user ROM. As it is now, INT_SER_PTR must be in emulation memory in address state 0, necessitating that the user map a whole 2K block of address state 0 as emulation memory.

Temporary solution:
Two work-arounds exist for this problem.
1) To avoid having to map a 2K block of memory in address state 0 as emulation memory, first map the INT_SER_PTR location as emulation memory, load the absolute file thus programming the appropriate address for INT_SER_PTR, then modify the configuration remapping address state 0 as user memory as desired. For this method to work, it is the user's responsibility to be sure that the INT_SER_PTR location in user memory is appropriatley programmed.

2) To avoid having to map a 2K block of memory in address state 0 as emulation memory, first re-link your absolute file, moving the module that contains INT_SER_PTR to an available address state mapped as emulation memory. Load the absolute file. Note that the jam-register is programmed only with the lower 16 bits

of the INT_SER_PTR address. Now, you must be sure that you have the appropriate locations in address state 0 programmed with the correct values. For example, if you re-linked such that INT_SER_PTR was located at 31234H, the jam register would be programmed with 1234H, and therefore you should be sure that user memory at 1234H is programmed with the correct values.

Signed off 10/24/86 in release 601.04

Number: D200063172 Product: F9450 EMULATION 64286 01.03

One-line description:
display memory mnemonic may fail after display trace mnemonic

Problem:
Display memory mnemonic will show all "unused prefetch"'s if the last trace display had an "unused prefetch" as the last disassembled state.

Signed off 10/24/86 in release 601.04

Number: D200064279 Product: F9450 EMULATION 64286 01.03

One-line description:
The inverse assembler may get confused in some cases.

Problem:
In some cases, the inverse assembler will fail to properly perform inverse assembly. For example:

```
     LIM R1,1234H
     OR R1,5678H
     AND R1,6789H
     JS R13,4321H  <---- THIS INSTRUCTION IS MARKED AS AN UNUSED
     JC 07H,0,R12        PREFETCH
```

Signed off 10/24/86 in release 601.04

Number: D200064261 Product: F9450 EMULATION 64286 01.03

One-line description:
The inverse assembler fails to show the mnemonic for a very short branch

Signed off 10/24/86 in release 601.04

HEWLETT PACKARD

5958-6019, January 1987
E0187

Printed in U.S.A.