EMULOGIC®
MICROPROCESSOR
DEVELOPMENT SYSTEMS

Order Number: RLN-Ø1ØØ

SOFTWARE RELEASE NOTICE

The release bulletins in this package contain important information about this shipment of EMULOGIC software. Be sure you read all notices before attempting to use this product.

Emulogic, Inc. provides release bulletins to advise you of special features, handling, or exceptions in the shipped version of our products. By doing so, we hope to help you use our products more effectively. If you experience difficulties with release notice information or with this software product, contact EMULOGIC Application Engineering department.

The following bulletins are included in this package:

Number ØØ3 -- 68ØØØ Emulation Software Names

RELEASE BULLETIN NUMBER ØØ3

SOFTWARE: EMULOGIC 68ØØØ Emulation Support Package (ESP) Software
SUBJECT: Emulation Software File Names

1) EMULOGIC ESP emulation software is activated when the RUN command specifies an appropriate emulation software file name. Support for the Motorola MC68ØØØ and compatible microprocessors up to 8-MHz has been available from Emulogic, Inc. under the name LØ12ØØ. The current version is 4.ØØ.

2) Support for the MC68ØØØ 1Ø-MHz and compatible microprocessors is now available under the name LØØ3ØØ also at version level 4.ØØ.

3) Be sure to invoke the correct ESP emulation software, especially if you work with both 8-MHz and 1Ø-MHz microprocessors on the same ECL-3211 MDS.

Addendum To:
68000 CROSS ASSEMBLER MANUAL SUPPLEMENT
18 November 1983

The information contained in this addendum extends and expands the previously published 68000 Chip Supplement to the Cross Assembler Manual (order number: CAS-2004-00) of June 1983. Future editions of the 68000 supplement may replace this addendum and earlier supplements.

LONG ADDRESS DEFINITION

The pseudop ADDRES has been added to allow the user to define a long address either absolutely or symbolically (see listing).

Addresses referenced can be absolute, relocatable, or global. In all cases two words are allocated for the address and the value established either at assembly time (absolute) or at link time (relocatable or global).

IMMEDIATE EFFECTIVE ADDRESS MODE

The immediate effective address mode has been extended to allow reference to symbols (absolute, relocatable, or global). The amount of space allocated for the immediate value is determined by the operation size, as demonstrated in the program listing example. (Long - two words; word - one word; byte - one word, must be absolute.)

The symbolic reference is resolved at assembly time (absolute) or at link time (relocatable or global).

If the operation is word and the resolvled symbol value is greater than FFFF the linker will indicate an error.

68000 Sample Program Listing

```
.MAIN.    X68000   V1.08 6-OCT-83 03:29:24 PAGE 1


      1 000000                                     .PSECT
      2          0010                              .RADIX   16
      3                                            .GLOBL   FAR
      4
      5 0000       00      00      12              ADDRES   1234
                   34
      6 0004       00      56      78              ADDRES   56(789A)
                   9A
      7 0008     0000'   0038'                     ADDRES   TAG1
      8 000C     0000G   0000G                     ADDRES   FAR
      9
     10 0010       06      93      12              ADDI.L   #1234#5678,(A3)
                   34      56      78
     11 0016       06      53      12              ADDI     #1234,(A3)
                   34
     12 001A       06      13      00              ADDI.B   #56,(A3)
                   56
     13 001E       06      93    0000'             ADDI.L   #TAG1,(A3)
                 0038'
     14 0024       23      FC    0000'             MOVE.L   #TAG1,FAR
                 0038'   0000G   0000G
     15 002E       28      3C    0000G             MOVE.L   #FAR,D4
                 0000G
     16 0034       36      3C    0038'             MOVE     TAG1,D3
     17
     18 0038       4E      71          TAG1:       NOP
     19          0001                              .END


.MAIN.   X68000    V1.08 6-OCT-83 03:29:24 PAGE 1-1
SYMBOL TABLE

FAR    = ****** G   TAG1     0038R

. ABS.   0000       00
         003A       01
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  308 WORDS  ( 2 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  51 PAGES
,T68=T68
```

CONVENTIONS

The linker for the 68000 is named ELINK3.SAV. References to "the linker" are, therefore, references to ELINK3. The slash (/) mark is used, in the syntax of ECL-3211 MDS commands, to indicate that the expression immediately following is a command switch.

SETTING THE STARTING ADDRESS FOR A PSECT

The maximum size for any program section (either ASECT or PSECT) is 64K bytes. Larger programs can be made modularly and can use up to 127 (decimal) PSECTs.

The linker's Q switch option is used to locate the starting address of each PSECT. The syntax for the Q option is:

```
/Q:n
```

The value, "n", represents the number of PSECTs whose base addresses you define at link time. You can link from 1 to 177 octal (127 decimal) at one time. The default value is 8 decimal. By controlling this value, the user can allocate an appropriate amount of space to the Q table, thus providing the maximum space for the symbol table.

NOTE

> We recommend that you include the Q option when running the linker because 68000 code is usually assembled in segments (that is, PSECTs). If the Q option is excluded, all PSECTs are concatenated in the order of entry.

The linker responds to the Q option by displaying the following prompt:

```
LOAD SECTION:ADDRESS?
```

Your response to this should be the PSECT's name and its absolute base address (hexadecimal). For example,

```
LOAD SECTION:ADDRESS? CODE:12340
```

In this case, program section CODE will be aligned to a starting address of 12340. The address entered must be on a word boundary.

SYMBOL TABLE FILES

The linker J switch should be used when creating symbol table files (.STB) for the 68000. The J switch provides for 24-bit symbols in the symbol table file and may be used in conjunction with the Q switch or, optionally, by itself. The syntax for a command string using the J switch is:

```
TEST,TEST,TEST=TEST/J
```

68010 PERMANENT SYMBOL TABLE

The following is a summary of the mnemonic codes which support the additional 68010 instructions now included in the EMULOGIC 68000 Macro Cross Assembler software.

TABLE A-1. ADDITIONAL 68010 INSTRUCTION SET SUMMARY

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
MOVE from CCR (.W)	CCR,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	MOVE FROM CONDITION CODES
MOVEC (.L)	Rc,Rn Rc=SOURCE=control register Rn=DESTINATION=data or address register Rn,Rc Rn=SOURCE=data or address register Rc=DESTINATION=control register Currently defined control registers are: Source Function Code (SFC) register Destination Function Code (DFC) register User Stack Pointer (USP) Vector Base Register (VBR)	MOVE TO/FROM CONTROL REGISTER
MOVES (.B,.W,.L)	Rn,<ea> Rn=SOURCE=data register or address register <ea>=DESTINATION=ALTERABLE MEMORY ADDRESSING modes <ea>,Rn <ea>=SOURCE=ALTERABLE MEMORY ADDRESSING modes Rn=DESTINATION=data register or address register	MOVE TO/FROM ADDRESS SPACE
RTD (unsized)	#displacement	RETURN AND DEALLOCATE PARAMETERS

68000 CHIP SUPPLEMENT

TO THE

CROSS ASSEMBLER MANUAL

CAS-2004-00

TABLE OF CONTENTS

LIST OF TABLES

INTRODUCTION

This guide is a supplement to the Emulogic Relocatable Macro Cross Assembler Manual, providing specific assembler information for writing software programs to run on the 68000 microprocessor. Explained herein are:

- the features of the Emulogic 68000 cross assembler,
- the 68000 cross assembler instruction set,
- the procedures for assembling 68000 source programs and then linking the object files, and
- MACRO 11 conditions affecting assembly.

Sample assembly output listings are also included.

FEATURES OF THE 68000 CROSS ASSEMBLER

The Emulogic MC68000 macro cross assembler has been implemented according to technical material contained in the following documents:

- MC68000 16-Bit Microprocessor User's Manual (Dated 9/1/79)
- MC68000 Resident Structured Assembler Reference Manual (Dated 11/80)

The Emulogic cross assembler provides the basic capability indicated in these documents except as indicated below:

1. PDP-11 MACRO-11 capability is supported, as long as the following are provided:

   - General Assembler Usage
   - General Assembler Directives
   - Macro Directives
   - Symbol and Expression Rules

2. The following MC68000 assembler output options have been implemented:

OPT	BRL	{FORWARD BRANCH LONG}
OPT	BRS	{FORWARD BRANCH SHORT}
OPT	FRL	{FORWARD REFERENCE LONG}
OPT	FRS	{FORWARD REFERENCE SHORT}

   No other OPT options have been implemented.

3. All size and jump (branch) modifying opcode suffix types (.B, .L, .W, and .S) have been implemented.

4. Variants on instruction types (A, Q, I, and M {see 2.6}) are all implemented but must be selected by the programmer. The assembler does not choose the variant automatically.

5. Maximum size for any one program section (ASECT or PSECT) is 64K bytes. 256 program sections can be handled.

6. All addressing modes are implemented.

   PC relative addressing will be chosen by the assembler if appropriate and legal. The programmer can force PC relative addressing if legal.

   Tables 2-6A through 2-6E are followed except for the changes required since neither { OPT PCS } nor { OPT PCO } are implemented.

7. Structured control statements (Chapter 6) are not implemented.

Table S-1 lists the effective addressing mode categories supported by the assembler, and Table S-2 lists the effective address mode syntax under RT-11. Refer to both of these as you read the description of the 68000 instruction set in the next section.

TABLE S-1. EFFECTIVE ADDRESSING MODE CATEGORIES

EFFECTIVE ADDRESS			ADDRESSING CATEGORIES			
MODES	MODE	REGISTER	DATA	MEMORY	CONTROL	ALTERABLE
Dn	000	register num.	X			X
An	001	register num.				X
An@	010	register num.	X	X	X	X
An@+	011	register num.	X	X		X
An@-	100	register num.	X	X		X
An@(d)	101	register num.	X	X	X	X
An@(d,ix)	110	register num.	X	X	X	X
xxx.W	111	000	X	X	X	X
xxx.L	111	001	X	X	X	X
PC@(d)	111	010	X	X	X	
PC@(d,ix)	111	011	X	X	X	
#xxx	111	100	X	X		

TABLE S-2. 68000 ASSEMBLER SYNTAX FOR EFFECTIVE ADDRESS MODES

EFFECTIVE ADDRESS MODES	ASSEMBLER SYNTAX (Under RT-11)
Dn	D0 - D7
An	A0 - A7 including SP
An@	(A0) - (A7) including SP
An@+	(A0)+ - (A7)+ including SP
An@-	-(A0) - -(A7) including SP
An@(d)	d(A0) - d(A7) including SP
An@(d,ix)	d(A0,Ri) - d(A7,Ri) including SP where Ri = A0 - SP .W or .L = D0 - D7 .W or .L
xxx.W	* (1234) will produce a short address 1234(S) will produce a short address *a tag or label is legal
xxx.L	* 12(3456) will produce a long address 3456(L) will produce a long address *a tag or label is legal
PC@(d)	d(PC)
PC@(d,ix)	d(PC,ix) where ix = A0 - SP .W or .L = D0 - D7 .W or .L
#xxx	#1234 for immediate word data #1234#5678 for immediate long data #1234#0ABCD note: 0 immediately prior to HEX notation

68000 PERMANENT SYMBOL TABLE

The following is a summary of the mnemonics for the 68000 instruction set. The instructions appear in alphabetical order.

These mnemonics are stored in a permanent symbol table and therefore are automatically recognized by the cross assembler. References to and operations with the registers within the 68000 microprocessor are legal. For a detailed description of the instruction set refer to the "MC68000 16-Bit Microprocessor User's Manual" (9/1/79).

> NOTE: Modifying opcode suffixes (.B,.W,.L) relative to an instruction appear in parenthesis to the immediate right of the instruction mnemonic.

TABLE S-3. 68000 INSTRUCTION SET SUMMARY

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
ABCD (.B)	Dn,Dn -(An),-(An)	ADD DECIMAL WITH EXTEND
ADD (.B,.W,.L)	<ea>,Dn <ea>=SOURCE=ALL ADDRESSING modes Dn,<ea> <ea>=DESTINATION=ALTERABLE ADDRESSING modes	ADD BINARY
ADDA (.W,.L)	<ea>,An <ea>=SOURCE=ALL ADDRESSING modes	ADD ADDRESS
ADDI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	ADD IMMEDIATE
ADDQ (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=ALTERABLE ADDRESSING modes	ADD QUICK

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
ADDX (.B,.W,.L)	Dn,Dn -(An),-(An)	ADD EXTENDED
AND (.B,.W,.L)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes Dn,<ea> <ea>=DESTINATION=ALTERABLE MEMORY ADDRESSING modes	AND LOGICAL
ANDI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING or STATUS REGISTER	AND IMMEDIATE
ASL,ASR (.B,.W,.L)	Dn,Dn #data,Dn <ea>=MEMORY ALTERABLE ADDRESSING modes	ARITHMETIC SHIFT
Bcc (.S,.L)	<label>	BRANCH CONDITIONALLY
cc = CC carry clear CS carry set EQ equal GE greater or equal GT greater HI high LE less or equal LS low or same LT less MI minus NE not equal PL plus VC no overflow VS overflow		
BCHG	Dn,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes #data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	TEST A BIT AND CHANGE

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
BCLR	Dn,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes #data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	TEST A BIT AND CLEAR
BRA (.S,.L)	<label>	BRANCH ALWAYS
BSET	Dn,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes #data,<ea> <ea>=DESTINATION=DATA ALERABLE ADDRESSING modes	TEST A BIT AND SET
BSR (.S,.L)	<label>	BRANCH TO A SUBROUTINE
BTST	Dn,<ea> <ea>=DESTINATION=DATA ADDRESSING modes #data,<ea> <ea>=DESTINATION=DATA ADDRESSING modes EXCEPT IMMEDIATE ADDRESSING mode	TEST A BIT
CHK (.W)	<ea>,Dn <ea>=upper bound operand word=DATA ADDRESSING modes	CHECK A REGISTER AGAINST BOUNDS
CLR (.B,.W,.L)	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	CLEAR AN OPERAND
CMP (.B,.W,.L)	<ea>,Dn <ea>=SOURCE=All ADDRESSING modes	COMPARE
CMPA (.W,.L)	<ea>,An <ea>=SOURCE=ALL ADDRESSING modes	COMPARE ADDRESS

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
CMPI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	COMPARE IMMEDIATE
CMPM (.B,.W,.L)	(An)+,(An)+	COMPARE MEMORY
DBcc (.W)	Dn,<label> cc = CC carry clear CS carry set EQ equal F never true GE greater or equal GT greater HI high LE less or equal LS low or same LT less MI minus NE not equal PL plus T true VC no overflow VS overflow	TEST CONDITION, DECREMENT AND BRANCH
DIVS (.W)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes	SIGNED DIVIDE
DIVU (.W)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes	UNSIGNED DIVIDE
EOR (.B,.W,.L)	Dn,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	EXCLUSIVE OR LOGICAL
EORI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE or STATUS REGISTER	EXCLUSIVE OR IMMEDIATE

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
EXG (.L)	Rx,Ry Rx=data register or address register Rx=data register if exchange is between data and address registers Ry=data register or address register Ry=address register if exchange is between data and address registers	EXCHANGE REGISTERS
EXT (.W,.L)	Dn	SIGN EXTEND
JMP (unsized)	<ea> <ea>=CONTROL ADDRESSING modes	JUMP
JSR (unsized)	<ea> <ea>=CONTROL ADDRESSING modes	JUMP TO SUBROUTINE
LEA (.L)	<ea>,An <ea>=SOURCE=CONTROL ADDRESSING modes	LOAD EFFECTIVE ADDRESS
LINK (unsized)	An,#displacement	LINK AND ALLOCATE
LSL,LSR (.B,.W,.L)	Dn,Dn #data,Dn <ea> <ea>=MEMORY ALTERABLE ADDRESSING modes	LOGICAL SHIFT
MOVE (.B,.W,.L)	<ea>,<ea> <ea>=SOURCE=ALL ADDRESSING modes <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	MOVE DATA FROM SOURCE TO DESTINATION
MOVE to CCR (.W)	<ea>,CCR <ea>=SOURCE=DATA ADDRESSING modes	MOVE TO CONDITION CODES

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
MOVE to SR (.W)	<ea>,SR <ea>=SOURCE=DATA ADDRESSING modes	MOVE TO THE STATUS REGISTER
MOVE from SR (.W)	SR,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	MOVE FROM THE STATUS REGISTER
MOVE USP (.L)	USP,An AN,USP	MOVE USER STACK POINTER
MOVEA (.W,.L)	<ea>,An <ea>=SOURCE=ALL ADDRESSING modes	MOVE ADDRESS
MOVEM (.W,.L)	<register list>,<ea> <ea>=DESTINATION=CONTROL ALTERABLE modes OR PREDECREMENT ADDRSSING mode <ea>,<register list> <ea>=SOURCE=CONTROL ADDRESSING modes OR POSTINCREMENT ADDRESSING mode	MOVE MULTIPLE REGISTERS

NOTE: EXAMPLES OF CORRECT ASSEMBLER SYNTAX FOR THE LOAD/STORE MULTIPLE INSTRUCTION:

```
MOVEM       (A6)+,A0-A4/D6/D2-D5

MOVEM       A1-A3/D3/A4/D5-D7/,-(A4)

MOVEM       (A4)+,ALLREG    {A0-A7 AND D0-D7}
```

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
MOVEP (.W,.L)	Dn,d(An) d(An),Dn	MOVE PERIPHERAL DATA
MOVEQ (.L)	#data,Dn	MOVEQ
MULS (.W)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes	SIGNED MULTIPLY

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
MULU (.W)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes	UNSIGNED MULTIPLY
NBCD	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	NEGATE DECIMAL WITH EXTEND
NEG (.B,.W,.L)	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	NEGATE
NEGX (.B,.W,.L)	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	NEGATE WITH EXTEND
NOP (unsized)		NO OPERATION
NOT (.B,.W,.L)	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	LOGICAL COMPLEMENT
OR (.B,.W,.L)	<ea>,Dn <ea>=SOURCE=DATA ADDRESSING modes Dn,<ea> <ea>=DESTINATION=ALTERABLE MEMORY ADDRESSING modes	INCLUSIVE OR LOGICAL
ORI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE OR STATUS REGISTER	INCLUSIVE OR IMMEDIATE
PEA (.L)	<ea> <ea>=CONTROL ADDRESSING modes	PUSH EFFECTIVE ADDRESS
RESET (unsized)		RESET EXTERNAL DEVICES

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
ROL,ROR (.B,.W,.L)	Dn,Dn #data,Dn <ea> <ea>=MEMORY ALTERABLE ADDRESSING modes	ROTATE (WITHOUT EXTEND)
ROXL,ROXR (.B,.W,.L)	Dn,Dn #data,Dn <ea> <ea>=MEMORY ALTERABLE ADDRESSING modes	ROTATE WITH EXTEND
RTE (unsized)		RETURN FROM EXCEPTION
RTR (unsized)		RETURN AND RESTORE CONDITION CODES
RTS (unsized)		RETURN FROM SUBROUTINE
SBCD	Dn,Dn -(An),-(An)	SUBTRACT DECIMAL WITH EXTEND
Scc	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	SET ACCORDING TO CONDITION
cc = CC carry set CS carry clear EQ equal F false GE greater or equal GT greater HI high LE less or equal LS low or same LT less MI minus NE not equal PL plus T always true VC no overflow VS overflow		

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
STOP (unsized)	#data	LOAD STATUS REGISTER AND STOP
SUB (.B,.W,.L)	<ea>,Dn <ea>=SOURCE=ALL ADDRESSING modes Dn,<ea> <ea>=DESTINALTION=ALTERABLE MEMORY ADDRESSING modes	SUBTRACT BINARY
SUBA (.W,.L)	<ea>,An <ea>=SOURCE=ALL ADDRESSING modes	SUBTRACT ADDRESS
SUBI (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	SUBTRACT IMMEDIATE
SUBQ (.B,.W,.L)	#data,<ea> <ea>=DESTINATION=ALTERABLE ADDRESSING modes	SUBTRACT QUICK
SUBX (.B,.W,.L)	Dn,Dn -(An),-(An)	SUBTRACT WITH EXTEND
SWAP (.W)	Dn	SWAP REGISTER HALVES
TAS	<ea> <ea>=DATA ALTERABLE ADDRESSING modes	TEST AND SET AN OPERAND
TRAP (unsized)	#vector	TRAP
TRAPV (unsized)		TRAP ON OVERFLOW

TABLE S-3. 68000 INSTRUCTION SET SUMMARY (CONTD.)

MNEMONIC	ADDRESSING MODES/OPERAND(S)	DESCRIPTION
TST (.B,.W,.L)	<ea> <ea>=DESTINATION=DATA ALTERABLE ADDRESSING modes	TEST AN OPERAND
UNLK (unsized)	An	UNLINK

IMPLEMENTED MC68000 OUTPUT OPTIONS:

OPT	BRL	{FORWARD BRANCH LONG}
OPT	BRS	{FORWARD BRANCH SHORT}
OPT	FRL	{FORWARD REFERENCE LONG}
OPT	FRS	{FORWARD REFERENCE SHORT}

To resolve forward references, both relative and absolute, the assembler will use the longer form of the effective address in the operand reference. This default may be overridden for a single operation by appending a .S to the instruction mnemonic providing that it is legal syntax for that instruction: BRA.S TAG

The default may also be changed to a shorter format by specifying:

```
OPT     BRS
BRA     TAG
BSR     TAG
```

which designates that forward relative branches should use the shorter displacement format.

Likewise, a forward absolute reference such as: NBCD 1234 will default to the longer form of the effective address. Since it is illegal to append a .S to the instruction mnemonic, the default may be changed to a shorter format by specifying:

```
OPT     FRS
NBCD    1234
```

which designates that forward absolute references should be short.

NOTE: No override option is possible on instructions with a given size code specification.

CALLING THE 68000 CROSS ASSEMBLER

To invoke the 68000 cross assembler from the system device, respond to the system prompt "." by typing the following:

```
        .R X68000 <CR>
```

The assembler should respond with an asterisk "*" prompt. At this point, it is ready to accept command string input from the keyboard and to perform an assembly. Everything typed in to the left of the = sign is output. Everything typed in to the right of the = sign is input.

```
        EX)     *TEST68=TEST68
```

This will produce an output object file: TEST68.OBJ

```
        EX)     *TEST68,TEST68=TEST68
```

This will produce an output object file: TEST68.OBJ
and an output listing file: TEST68.LST

The 68000 cross assembler can be terminated at any time by typing ^C (Control C) from the keyboard. This is accomplished by depressing the key marked "CTRL" and the "C" key simultaneously. A ^C should be echoed on the console screen.

1. To terminate the 68000 cross assembler after you have invoked it and prior to entering the command string, TYPE ^C.

```
        EX)  .R X68000
             *^C
             .
             (returns to the system monitor prompt)
```

2. To terminate the 68000 cross assembler after you have invoked it and entered a command string, (i.e., assembly of source files is in process) type two consecutive ^C's.

```
        EX)  .R X68000
             *TEST68,TEST68=TEST68
             ^C^C
             .
             (returns to the system monitor prompt)
```

CALLING THE LINKER

ELINK3.SAV is the linker for the Emulogic MC68000 cross assembler. It processes the cross assembler output object module(s) and produces an executable absolute load module.

With the following exceptions, ELINK3.SAV operates similarly to the RT-11 linker, LINK:

1. A .SAV file is not created.

2. An .LDA file is created.

3. ELINK3.SAV defaults to .LDA output formatted binary file; therefore, /L option is not required.

4. The map is produced in hex notation.

5. 24 bit addressing is accomplished with a /D:xxx/B:xxxxxx option included in the command string to the linker. This /D:xxx/B:xxxxxx option allows you to specify a 24 bit bottom address for your relocatable code.

   NOTE: xxx's represent octal values.

To call the linker from the system device, respond to the system prompt "." by typing:

```
        .R ELINK3
```

The linker should respond with an asterisk "*" prompt. At this point, it is ready to accept command string input from the keyboard and to perform the linkage. Everything typed in to the left of the = sign is output. Everything typed in to the right of the = sign is input.

```
        EX)     *TEST68=TEST68
```

```
This will produce output file:          TEST68.LDA  from TEST68.OBJ
```

```
        EX)     *TEST68,TEST68=TEST68
```

```
This will produce output files:         TEST68.LDA
                                        TEST68.MAP  from TEST68.OBJ
```

```
         EX)    *TEST68,TEST68,TEST68=TEST68
```

```
This will produce output files:         TEST68.LDA
                                        TEST68.MAP
                                        TEST68.STB  from TEST68.OBJ
```

NOTE: Termination of the linker is accomplished in the same manner as for the assembler.

The following example assembles and links three programs using the library:

```
        .R X68000
        *TT68T,TT68T=TT68T      (OBJ AND LST)
        *TL68T,TL68T=TL68T      (OBJ AND LST)
        *TI68T,TI68T=TI68T      (OBJ AND LST)
        *TJ68T,TJ68T=TJ68T      (OBJ AND LST)
        *^C

        .R LIBR
        *LIBT68=TT68T,TL68T,TI68T     (OBJ)
        *^C

        .R ELINK3
        *T68T,T68T,T68T=TJ68T,LIBT68/D:372/B:20000  (LDA,MAP AND STB)
        *^C
```

NOTE: Remember /D:xxx/B:xxxxxx are octal values. They are reflected in the load map in hex.

MACRO-11 PROVISIONS

The Emulogic 68000 cross assembler is written under Digital Equipment Corporation's PDP-11 MACRO-11 assembler. As a result, the following are applicable:

1. The maximum size for any one program section (both ASECT and/or PSECT) is 64K bytes. Addressing over 64K bytes is accomplished at link time via the /D:xxx/B:xxxxxx Emulogic ELINK3 option.

   Truncation errors will result in your source if you attempt to set the program counter over 64K bytes.

   ```
   EX)  .=0FEFE8
   ```

   will produce a "T" error in your assembly listing to flag the truncation.

   However, the following can occur:

   ```
   EX)  .=0FFFE
        NOP
        NOP
        NOP
        NOP
   ```

   Resulting in: the second NOP is at PC=0000. when the assembler reaches the 64K byte limit, it forces the program counter to wrap around to 0000 and continues to assemble code from 0000. Therefore, we recommend inspection of your .LST files created at assembly time.

2. Calculation of relative offsets is performed based on even word boundaries. Therefore, should you stipulate in your source an odd forward absolute reference:

   ```
   EX)          BCC   1123
   ```

   The offset will be calculated to relfect a displacement value representative of the last-most even word.

   Likewise, the relative offset of an odd backward absolute reference will be calculated to the last-most even word.

   Therefore, should you attempt to branch and/or jump to an odd absolute address, the assembler will perform relative offset calculations based on even word boundaries. It is recommended you inspect your .LST files at assembly time.

3. During the first assembly pass, basically, the assembler will determine the length of the instructions and assemble them according to length as one word, two words, etc. During the second assembly pass, source statements containing MACRO-11 detected errors are flagged. For details on PDP-11 MACRO-11 error codes, directives, and more, refer to:

   Digital Equipment Corporation PDP-11
   MACRO-11 Language Reference Manual

4. Global or relocatable address data is passed from the assembler to the linker as sector number and value (address relative to sector beginning). The linker must determine the absolute address. Protocol between the assembler and linker indicates that a short (one word) or long (two word) address is desired. The assembler will allocate the appropriate number of words. If the linker finds that a short address has been allocated but a long address is required, it shall indicate an error condition.

   ```
   ?LINK-W-ADDRESS TOO LARGE FOR FORCED ABSOLUTE SHORT
   ```

For complete description of the linker, refer to the RT-11 System User's Guide (Section 11, LINKER)

SAMPLE 68000 ASSEMBLY LISTINGS

The remainder of this supplement consists of sample output listings from the 68000 cross assembler.

EMULOGIC

68000

User's Guide Supplement

Order Number: CSU-03004-01

68000 USER'S GUIDE SUPPLEMENT

This document supplements the ECL-3211 System User's Guide by providing operational information specific to the emulation of Motorola's 68000 and compatible microprocessors. This document describes special set-up procedures, conditions, and limitations to be noted when emulating the 68000. It is assumed here that the reader has read the User's Manual and is already familiar with the details of the 68000. Ready access to the technical literature is a plus.

This supplement covers five general areas.

1) Installation

2) Initialization

3) Abbreviations (p.3)

4) Unique Features (p.7)

5) Electrical (DC) Characteristics (p.15)

*** INSTALLATION ***

System installation instructions will be found in the User's Manual.

Note that the 10 MHz 68000 requires a Dual Connector Card on the ECL-3211's High Speed Memory. Instructions for installation of the 10 MHz Dual Connector Card will be found in the System installation section in the User's Guide.

*** INITIALIZATION ***

Type on the keyboard "RUN L01200" to load the Emulation Software into the ECL-3211. Note that a user can use the Operating System's RENAME function to give the file a name the user would prefer. Additionally, a Command File can be created causing L01200 to run.

There are no special initialization instructions for the 68000.

*** ABBREVIATIONS ***

SYSTEM DISPLAY

These are seen on the top half of the display when using the Emulation Software. All of these registers and flags can be loaded with user prefered values with the SET Command or ALTER mode as described in the User Manual or HELP file.

***	*** DESCRIPTION ***	
D0 - D7	Data Registers 0-7	32 bits/8 hex digits
PC	Program Counter	24 bits/6 hex digits
A0 - A6	Address Registers 0-6	32 bits/8 hex digits
US	User Stack Pointer	32 bits/8 hex digits
SS	Supervisor Stack Pntr	32 bits/8 hex digits
T	Trace Mode Flag	bit 15
S	Supervisor/User Mode Flag	bit 13
I	Interrupt Mask	octal decode of bits 8,9,10
X	Extend Flag	bit 4
N	Negative	bit 3
Z	Zero	bit 2
V	Overflow	bit 1
C	Carry	bit 0
SR	Status Register	16 bits/4 hex digits

TRACE DISPLAY

Note: Low=0 High=1 Don't Care=X

"1" and "0" refer to ELECTRICAL, NOT logical levels; though for ECL-3211 functions logical and electrical coincide.

These are seen when examining the Trace.

***	*** DESCRIPTION ***
F C	Function Code; FC0,FC1,FC2 decoded
B A	BGACK-L; Bus Grant Acknowledge
B G	BG-L; Bus Grant
B R	BR-L; Bus Request
B E	BERR-L; Bus Error
I P	Interrupt Priority; IPL0-L,IPL1-L,IPL2-L decoded
V M	VMA-L; Valid Memory Address
V P	VPA-L; Valid Peripheral Address
U D	UDS-L; Upper Data Strobe
E	Enable
L D	LDS-L; Lower Data Strobe
R S	RESET-L
H L	HALT-L
R W	R/W-L; Read/Write
I	Instruction Fetch

BREAKPOINT DISPLAY Note: Low=0 High=1 Don't care=X

These are seen when examining or setting Breakpoints.

E0-E7	Pod External Input 0-7
SW1	Logical Switch 1 External Trigger 1
SW2	Logical Switch 2 External Trigger 2
SW3	Logical Switch 3
SW4	Logical Switch 4
INS	Instruction Fetch; this is a signal generated by the Emulator. A "1" indicates the fetch of the first byte/word of an Op Code as a Breakpoint Condition.
ROM	ROM access; "1" means trigger on a read from an address designated as ROM.
CO1	Counter 1 expired (CO1=0)
CO2	Counter 2 expired (CO2=0)
ADDR	Address Bus; 24 bits (UDS is A0)
DATA	Data Bus; 16 bits
E	Enable (output)
FC0	Function Code 0
FC1	Function Code 1
FC2	Function Code 2
HALT	HALT-L
RESET	RESET-L
LDS	Lower Data Strobe-L
VMA	Valid Memory Address-L
VPA	Valid Peripheral Address-L
RW	Read/Write-L
IPL0	Interrupt Priority Level 0-L
IPL1	Interrupt Priority Level 1-L
IPL2	Interrupt Priority Level 2-L
BGACK	Bus Grant Acknowledge-L
BG	Bus Grant-L
BR	Bus Request-L
BERR	Bus Error-L
PH	Phantom Status; JUMP or CALL Note: this is a Breakpoint Action rather than a Breakpoint Condition.

L01200

The file name for the Emulation Software is L01200. It is accessed through the Operating System hosted by the ECL-3211's CPU.

MAX FREQ

The maximum specified frequency of operation without WAIT states in Maps 1-4 or Target memory is 8 MHz for the 8 MHz 68000 and 10 MHz for the 10 MHz 68000. (Map 0 must be off.)

For operation in Map 0 without WAIT states, the maximum frequency is 6 MHz for both 8 MHz and 10 MHz 68000 Pods.

Both 8 MHz and 10 MHz 68000 Pods must be clocked at 4 MHz to use DEC Internal Memory. There will be WAIT states using this bank of memory.

Minimum operating frequency for both products is 4 MHz.

RESET

The ECL-3211's RESET command resets the 68000 Pod only, and does not reset the Target. A Reset generated by the Target or the User's program has effect during emulation only.

HALT

Whether internally or externally generated, a HALT condition must be serviced by the user. The 68000 Pod has no facility for terminating a HALT condition, and if Emulation mode is left while the 68000 Pod is in a HALT condition, all register settings will be lost.

NO TARGET

Not having the 68000 Pod deployed in a target will not affect the operation of the Emulator in any way, assuming the user does not try to access resources in the Target.

TRACE DATA CAPTURE

If the Trace has been turned on, it takes a "snapshot" of conditions during each Machine Cycle when the conditions are valid. For example, the Data bus is sampled when it contains valid Data. Address information is sampled when there is a valid Address on the bus. Control signals are sampled at the same time as the Data unless they must be sampled at a different point in the Machine Cycle. (The Trace is turned on by defining a Breakpoint with conditions that will be met and an Action statement including Set Trace, as described in the User's Guide and HELP file.)

Instructions are disassembled in the Trace as they appeared on the Data bus when they were fetched.

In the event that a Branch or Jump instruction is executed, the word that was Prefetched will appear in the Trace. If the word is a complete instruction, it will be disassembled. If the word is the first word of a multi-word instruction, the Trace Incomplete notation, "--T.I.--", will appear.

Note that the External Inputs are not sampled simultaneously in a Machine Cycle. External Inputs 0-3 are sampled during the valid address time of a Machine Cycle and External Inputs 4-7 are sampled during valid data time.

BREAKPOINT ACTION

Defined Breakpoint conditions are tested and resolved at the end of the Machine Cycle. Any Breakpoint Actions for a Breakpoint with conditions that have been met in a Machine Cycle commence at the completion of that Machine Cycle. For the action of SET TRACE (ST), the Trace will not start to record activity until the next instruction fetch.

--E.C.--

In the Trace Buffer under the column titled "Instruction", there will be entries titled "--E.C.--". "EC" stands for Emulation Cycle and represents cycles used to purge the 68000's instruction pipeline when control is transfered back to the Emulator. This appears in the Trace Buffer when emulation is stopped due to user action from the keyboard, or when a valid and active Breakpoint has as a defined action a Pause or a Halt. It does not affect the timing of a program in any way.

CLOCK

The Emulator provides two sources (Emulator and Target) and three different modes of providing a clock signal to the 68000 Pod; External, Internal, Internal source with external drive (Internal Drive). Both modes of Internal clock have a guarantee for 100 Kilohertz resolution.

--- External ---

External Clock is the mode in which the Target Circuit (the circuit the Pod is deployed in) provides the clock. Since it is buffered in the Pod with TTL logic, the clock signal must be TTL driven or equivalent. Do not clock the Pod with a Crystal/RC Network circuit.

Type "FREQ EXT" to select this mode.

--- Internal ---

Internal Clock is the mode in which the 68000 Pod is clocked by the Emulator. No clocking signal is taken from or given to the Target. Pin 15 (CLK) on the Pod's Target Cable Plug is electrically disconnected.

Type "FREQ xxxx" to select the Internal Clock mode. "xxxx" is the value of the frequency in units of Kilohertz. There is no need to specify "Internal" at any point.

--- Internal Drive ---

Internal Drive is the mode in which the 68000 is receiving its clock from the Emulator (as in Internal) but the signal the Pod's 68000 is receiving is also connected to Pin 15 (CLK) on the Target Cable Plug for use by the Target circuit. In this way, other components can be driven from the same clock that drives the Pod's 68000 and thus the entire Target Circuit can be provided with a software controllable clock.

Type "FREQ xxxx IND" to select Internal Drive mode. "xxxx" is the value of the frequency in units of Kilohertz.

SET COMMAND

Register values can be altered to user preferred values prior to emulation when in Command mode. The format used is described in the User's Manual or the HELP file. The abbreviations used to refer to the 68000's registers are listed above in the section on Abbreviations under System Display. The Abbreviations on the screen representing the registers are basically those used to change the register's contents. Note that the ECL-3211 provides two ways of changing the contents of Registers or other parameters as discussed in the User's Guide: ALTER mode and the SET command. The later is discussed here.

To illustrate, consider the following examples:

a) Typing "SE D0=12345678" will change the contents of Data Register 0 to "12345678".

b) Typing "SE A0=12345678,A6=87654321" will change the contents of Address Register 0 to "12345678" and the contents of Address Register 6 to "87654321".

c) Typing "SE MEM 123=5555" will change the contents of Memory address 000123 to "5555".

d) Typing "SE MEM 1500=1111,2222,3333" will change the contents of Memory addresses 001500 to "1111", 001502 to "2222", 001504 to "3333".

e) Typing "SE MEM 0-7FE=AA" will change the contents of Memory addresses 00000000 through 0007FE inclusive with "00AA".

f) Typing "SE MEM 0-0=AA" will set only Memory address 000000.

BYTE/WORD

The ECL-3211 has two modes of transfering data to and from a Target, BYTE or WORD. This is relevant when using the SET or LOAD commands to write values to addresses that are physically present in the Target. Data can be read or written to the Target in bytes or words to provide compatibility with the I/O components in the Target. To illustrate, a Target using 6800 family I/O components should use BYTE mode. A Target with I/O components that are designed for a 16 bit Data Bus should use WORD mode.

The existing mode is indicated in the Switch display on the line labeled "EXTERNAL FETCH/PUT". (To access the Switch display, type "SW" while in Command mode.) To select the mode of data transfers, type "WORDS" or "BYTES" when the Emulator is in Command mode. The HELP file sections can be accessed by typing "HE WO" for Word mode, or "HE BY" for Byte mode. The default is BYTE.

PHANTOMS

To branch to a Phantom code segment when a defined set of Breakpoint conditions are met, a user must 1) define Phantom as being a Breakpoint Action for the defined conditions and 2) select the mode of Phantom. The 68000 Emulator offers the user a choice as to the mode of Phantom to be employed; JUMP or CALL.

The choice is implemented when the Emulator is in Command Mode. Type "BR n CALL" or "BR n JMP" as appropriate, where "n" is the number identifying which Breakpoint is being defined. To illustrate, type "BR 3 JMP" to select JUMP mode for a Phantom defined in Breakpoint 3. Remember that to take a Phantom, a user must specify a Phantom as a Breakpoint Action.

The choice of CALL for one Breakpoint does not prevent a user from selecting JUMP for a different Breakpoint.

For a Phantom CALL, 1) the value of 6 should be subtracted from the the value pointed to by the Stack Pointer to adjust the stored PC to return to the address where the Phantom was inserted 2) the user code should be terminated with an RTS instruction. To illustrate, below is an example of the last instructions of a Phantom:

```
        .
        .
        .
        SUBQ.L #6,(SP)     /5D97
        RTS                /4E75
```

DTACK (Rev 4+)

This section is for User's with Rev. 4.00 or later Emulation Software.

To ease product and software development, the 68000 Emulator offers three options for providing the 68000 with DTACK. There is also the option to choose the number of WAIT states (up to 14) the Emulator will delay its generation of DTACK to simulate memory banks of varying access times. Note that this feature is not available for Map 0.

The DTACK signal the 68000 Pod receives can be sourced:

1) from the ECL-3211 alone
2) from the Target alone
3) from whichever DTACK is received later --the ECL-3211's or the Target's.

If the third option is selected, the 68000 Pod waits for both DTACKs to be present to create the effect that the one that is issued last is the one that is responded to by the 68000 Pod. This provides a margin of safety for "slow" RAM in the Target to prevent the premature termination of Read and Write operations.

Note that any reference to External (Target) Memory will automatically use the Target's DTACK for the duration of those references. The user's DTACK selection will be preserved.

When sourcing DTACK from the Target, a 5 microsecond timer will always be present on every machine cycle. If the Target does not generate the DTACK, the Watchdog Timer will.

The DTACK configuration is found in the Switch display (accessed while the Emulator is in Command mode). Changes are made while in Command mode.

Type "DTA ECL" to select the ECL-3211 as the source of DTACK.

Type "DTA TAR" to select the Target as the source of DTACK.

Type "DTA TAR,ECL" or "DTA ECL,TAR" to select the later of the Target's or the Emulator's DTACK.

Type "WAIT ON" to enable the Emulator's delay of DTACK when referencing the Emulator's High Speed Memory. Type "WAIT OFF" to disable this feature. This feature is normally disabled.

To select the number WAIT states (up to 14) the ECL-3211 will delay its DTACK, use the syntax "WAIT XX=YY" where XX is the Map Number and YY is the number of desired WAIT states. Type "WAIT 1=14" to select a delay of 14 WAIT states in DTACK generation for any access to user defined Map 1. (Note that this is the maximum number of WAIT states possible.) Type "WAIT 3=2" to select a delay of 2 WAIT states for any access to user defined Map 3. Note that Map 0 will not support the selectable WAIT state feature.

DTACK (Rev. 1, 2, and 3)

This section is for User's with Revisions 1.xx, 2.xx, or 3.xx of the Emulation Software.

To ease product and software development, the 68000 Emulator offers three options for providing the 68000 Pod with DTACK. There is also the option to choose the number of WAIT states (up to 14) the Emulator will delay its generation of DTACK to simulate memory banks of varying access times. Note that this feature is not available for Map 0.

The DTACK signal the 68000 receives can be sourced:

1) from the ECL-3211 alone
2) from the Target alone
3) from whichever DTACK is received later --the ECL-3211's or the Target's.

If the third option is selected, the 68000 Pod waits for both DTACKs to be present to create the effect that the one that is issued last is the one that is responded to by the 68000 Pod. This provides a margin of safety for "slow" RAM in the Target to prevent the premature termination of Read and Write operations.

Note that any reference to External (Target) Memory will automatically use the Target's DTACK for the duration of those references. The user's DTACK selection will be preserved.

When sourcing DTACK from the Target, a 5 microsecond timer will always be present on every machine cycle. If the Target does not generate the DTACK, the Watchdog Timer will.

The DTACK configuration is found in the Switch display (accessed while the Emulator is in Command mode). Changes are made while in Command mode.

SW A	Enables Target DTACK
SW B	Disables ECL-3211 DTACK, enables Target DTACK
SW C	Enables ECL-3211 programmed WAIT states to permit simulation of banks of RAM with varying Access Times

The allowed combinations of DTACK switch settings are:

*** SWITCH ***			*** DTACK ***
A	B	C	
0	0	0	-- ECL only
1	0	0	-- the later of ECL-3211 or Target
1	1	0	-- Target only
1	0	1	-- the later of (ECL-3211 + WAIT) or Target

To select the number of WAIT states (up to 14) the ECL-3211 will delay its DTACK, use the syntax "WAIT XX=YY" where XX is the Map Number and YY is the number of desired WAIT states. Type "WAIT 1=14" to select a delay of 14 WAIT states in DTACK generation for any access to user defined Map 1. (Note that this is the maximum number of WAIT states possible.) Type "WAIT 3=2" to select a delay of 2 WAIT states for any access to user defined Map 3. Note that Map 0 will not support the selectable WAIT state feature.

NOT EMULATING

When the Emulator is not in Emulation mode, a number of 68000 signals remain active. They are listed here:

A1-A23	(pins 29-48, 50-52)
R/W-L	(pin 9)
LDS-L	(pin 8)
UDS-L	(pin 7)
AS-L	(pin 6)
FC0,FC1,FC2	(pins 26-28)
E	(pin 20)
BG-L	(pin 11)
VMA-L	(pin 19)
VPA-L	(pin 21)
BR-L	(pin 13)
DTACK-L	(pin 10)
BGACK-L	(pin 12)

CLK (pin 15) will be generated when not in Emulation mode if Internal Drive is selected as the Clocking mode.

D0-D16 (pins 1-5, 54-64) will be Tri-Stated.

HALT-L (pin 17) and RESET-L (pin 18) will be open.

IPL0, IPL1, and IPL2 (pins 23-25) will be pulled High.

BERR-L (pin 22) will be pulled High.

*** ELECTRICAL (DC) CHARACTERISTICS ***

Signal	Buffer Type	Output Drive		Input Load		Delay, additional	Termination, pull-up R
	74xxx	High mA	Low mA	High mA	Low mA	nSec typical	ohms
A1-A23	LS245	-15.0	64.0	0.07	-1.0	7	--
D0-D16	LS245	-15.0	64.0	0.07	-1.0	7	--
R/W-L	LS245	-15.0	64.0	--	--	7	10K
LDS-L	LS245	-15.0	64.0	--	--	7	10K
UDS-L	LS245	-15.0	64.0	--	--	7	10K
AS-L	LS245	-15.0	64.0	--	--	7	10K
FC0	LS245	-15.0	64.0	--	--	7	10K
FC1	LS245	-15.0	64.0	--	--	7	10K
FC2	LS245	-15.0	64.0	--	--	7	10K
E	F241	-15.0	64.0	--	--	7	--
VMA-L	LS245	-15.0	64.0	--	--	7	--
BR-L	F241	--	--	0.02	-1.6	7	1K
DTACK-L	F08	--	--	0.02	-0.6	21	1K
IPL0-2	F32	--	--	0.02	-0.6	7	10K
VPA-L	F32	--	--	0.02	-0.6	7	10K
BERR-L	F32	--	--	0.02	-0.6	7	10K
BGACK-L	F241	--	--	0.02	-1.6	7	1K
Clock, Ext	F253	--	--	0.02	-0.6		1K
Clock, Int Dr	F253	-1.0	20.0	--	--		1K
BG-L	none; it is driven directly by the Pod's 68000						1K
HALT-L	CD4066 analog switch; negligible loading						1K
RESET-L	CD4066 analog switch; negligible loading						1K