

NAKED MINI® 4

OPTIONAL INSTRUCTIONS EMULATOR

(EMLTR4)

93063-90

MAY 1981



Engineering Reference Number
90-93063-00E0

NAKED MINI® Division

**HERTFORD HOUSE
DENHAM WAY, MAPLE CROSS
RICKMANSWORTH, HERTFORDSHIRE WD3 2XD**

Telephone Rickmansworth 71211

Telex 922654

Printed in U.S.A.

REVISION HISTORY

<u>Revision</u>	<u>Date</u>	<u>Comments</u>
A0	Dec. 1978	Original issue.
B0-C0		Skipped for configuration control.
D0	Aug. 1979	Adds Divide instruction.
E0	May 1981	Corrects floating point instruction mnemonic and unimplemented Instruction Trap mnemonics. The information in this manual corresponds to the OS4 software package released at revision level F000.

Computer Automation, Inc., reserves the right to make changes in specifications at any time and without notice. The information furnished by Computer Automation, Inc., in this publication is believed to be accurate and reliable. However, no responsibility is assumed by Computer Automation, Inc., for its use; nor for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of Computer Automation, Inc.

Computer Automation, Inc., welcomes comments and suggestions for improvements to our manuals. Should difficulty be encountered with the information in the manual or if you have a suggestion for manual improvement, please send the comments or suggestions to Computer Automation, Inc., Technical Publications, Department 11-0770, 18651 Von Karman, Irvine, California, 92713. Please provide as much information as possible to help evaluate the comment or suggestion.

CONTENTS

SECTION 1	INTRODUCTION	1-1
	1.1 OPERATING ENVIRONMENT	1-1
	1.2 REFERENCE DOCUMENTATION	1-2
SECTION 2	OPERATIONAL OVERVIEW	2-1
	2.2 PROGRAM DESIGN AND MEMORY UTILIZATION	2-1
SECTION 3	PROGRAM OPERATION	3-1
	3.1 CREATING THE USER PROGRAM	3-1
	3.1.1 Referencing EMLTR4 Modules with LOAD Directives	3-1
	3.1.2 Using Noninterruptable Code	3-4
	3.1.3 Preparing a User Program for Execution Under RTX4	3-5
	3.1.4 Preparing a User Program for Standalone Execution	3-5
	3.2 ASSEMBLING, LINKING, AND EXECUTING THE USER PROGRAM	3-5

FIGURES

<u>Figure</u>		<u>Page</u>
3-1	LOAD Directive Operands Used to Reference EMLTR4 Instruction Modules	3-4
3-2	Unimplemented Instruction/Emulator Interface	3-6

TABLE

<u>Table</u>		
3-1	Optional Firmware Instructions	3-2

SECTION 1

INTRODUCTION

The NAKED MINI® 4 Optional Instruction Emulator (EMLTR4) is a program that is used on a NAKED MINI® 4 processor to emulate the entire set of LSI 4/10 and LSI 4/30 optional firmware instructions. The EMLTR4 program allows the programmer to upgrade his processor's instruction set to that of any model in the NAKED MINI® 4 Family up to a standard LSI 4/90. It also allows the programmer to add the LSI 4/90 optional floating point instructions to his set.

EMLTR4 does not include the following instructions:

- LSI 4/90 List optional instructions
- LSI 4/10S instructions
- NAKED MINI® 4/Series 5 instructions
- SCOUT SAL instruction.

When linked to a user program under OS4 or OMEGA4 control the EMLTR4 program performs the same functions the optional firmware does. To the programmer, the only apparent difference between using EMLTR4 and the optional firmware is the lower performance speed of EMLTR4.

1.1 OPERATING ENVIRONMENT

The emulation of optional instructions requires that the user program to be linked with EMLTR4 be in NAKED MINI® 4 Object Program Format (i.e., by being assembled by the NAKED MINI® 4 Assembler under OS4 or OMEGA4). The EMLTR4 program must be linked to the user program by the OS4 LINK utility or the OMEGA4 linking loader, LAMBDA4. The resultant linked program must be executed under an OS4, OMEGA4 or RTX4 system. Thus, all hardware needed to run OS4 or OMEGA4 is required to prepare a program for EMLTR4 execution. The EMLTR4 program itself requires only 1000 words of read/write memory.

The OS4 system or OMEGA4 system paper tapes or diskette are required to develop and link the user program. OS4, OMEGA4, or RTX4 paper tapes or diskette are required to execute it. EMLTR4 is supplied in paper tape form under engineering part number B06302. It is also available as a library file, EMLTR.LIB, on the RTX4 Library Diskette number, 00-93410-01.

1.2 REFERENCE DOCUMENTATION

The documents listed below supply additional information needed by the user of EMLTR4:

OS4 System Reference Manual (93460-90)

OMEGA4 System Reference Manual (93500-90)

Real-Time Executive (RTX4) Reference Manual (94500-90)

NAKED MINI® 4 Family Assembler Reference Manual (93500-80)

LSI 4/10 Computer Reference Manual (20990-91)

LSI 4/30 Computer Reference Manual (20991-91)

LSI 4/90 Computer Reference Manual (20945-91)

SECTION 2

OPERATIONAL OVERVIEW

This section provides an operational overview.

2.1 OPERATIONAL OVERVIEW

EMLTR4 is composed of modules of code which, when linked to a user program, emulate any optional firmware instructions included in the user program.

The actual emulation of instructions occurs during user program execution. When the processor encounters an optional instruction not implemented in its firmware, it invokes an Unimplemented Instruction Trap (UIT). This trap condition causes the processor to save in location :84 the address of the next user program instruction. The processor saves in location :85 the instruction that invoked the trap, and then it executes a routine that begins in location :86. The routine beginning in location :86 causes a branch to the EMLTR4 program, and a fast indexing scheme based on the instruction skeleton selects the EMLTR4 module containing the needed code. During EMLTR4 execution, interrupts are serviced normally.

After the instructions in this selected EMLTR4 module are executed, program control transfers to the address originally stored in location :84--that is, the address of the next user program instruction. All registers are returned to their pretrap status, with the exception of registers whose contents were supposed to be altered by the instruction. User program execution continues as before, with subsequent optional instructions invoking the UIT routine.

If EMLTR4 decides that the instruction that caused the UIT is in fact an unimplemented instruction, it branches to location R:UIRTN (an external location) with all registers and locations :84 and :85 retaining the same values as upon entry. Interrupts are inhibited on entry to location R:UIRTN.

2.2 PROGRAM DESIGN AND MEMORY UTILIZATION

EMLTR4 is structured into modules of code that emulate individual optional instructions and into modules containing only LOAD directives that reference sets of individual instruction modules. When only a subset of all the optional firmware instructions is required, it is necessary to link only the needed modules of code with the user program. The programmer can control which modules are to be linked by appropriately coding his user program. This is discussed in the next section of this manual.

The EMLTR4 program itself is reentrant. It can be activated by several activities (under RTX4) at one time. EMLTR4 requires a minimum stack area of 30 words to be used as a scratch pad area for each emulation. Thus, in a multitask program where each task assumes its own local stack area, only one copy of EMLTR4 is required.

SECTION 3

PROGRAM OPERATION

When emulation of optional instructions is desired, the user program must be created, assembled, and linked under an OS4 or OMEGA4 system. Use of EMLTR4 requires, therefore, that the programmer follow standard OS4 or OMEGA4 program development procedures. In addition, in developing his program he must take special steps that allow needed EMLTR4 modules to be linked with the user program; these are defined below. To prepare his user program for execution under RTX4, the programmer must take further steps also defined below.

3.1 CREATING THE USER PROGRAM

The programmer who wishes to use EMLTR4 to emulate optional instructions first creates his user program according to instructions in the OS4 System Reference Manual or the OMEGA4 System Reference Manual. He may include any of the EMLTR4-supported instructions in his program, but he must include NAKED MINI@ 4 Assembler LOAD directives for any EMLTR4-supported instructions he uses, as described below. The programmer must also include code in the user program to set the L and K stack registers. These registers should be set to provide a minimum stack area of 30 words for use as scratch pad area for each emulation.

3.1.1 Referencing EMLTR4 Modules with LOAD Directives

LOAD directives are used in the user program to direct the linker to link a user program and an EMLTR4 module; this module either can contain code that emulates a single optional instruction or it can contain LOAD directives that reference a set of individual instruction modules. When a LOAD directive is used to link one of the individual instruction modules listed in Table 3-1, it generally takes the form:

LOAD EM:mnemonic

where: mnemonic is the mnemonic of the instruction to be linked.

Longer LOAD directive operands require abbreviation, as is the case when the bit manipulation instructions are referenced. The COPYE instruction uses four different skeletons, each referenced by an individual operand. All LOAD directive operands used to reference individual instructions are listed beside the instructions in Table 3-1.

Table 3-1. Optional Firmware Instructions (Page 1 of 2)

MULTIPLE MEMORY REFERENCE INSTRUCTIONS						
MNEMONIC	SKELETON	STANDARDS(S)/OPTIONAL(O)			REFERENCE TO EMLTR4 ROUTINE	
		4/10	4/30	4/90		
ADDE	070A	0	0	S	EM: ADDE	
ANDE	070E	0	0	S	EM: ANDE	
CEA	0704	0	S	S	EM: CEA	
COPYE(AQ,AX,AY)	0706	0	S	S	EM: LDR2	
COPYE(AQ,AX,AY)	0707	0	S	S	EM: STR	
COPYE(A,Q,X,Y)	0708	0	0	S	EM: LDR	
COPYE(A,Q,X,Y)	0709	0	0	S	EM: STR	
CSKE	270F	0	0	S	EM: CSKE	
CSM	0705	0	S	S	EM: CSM	
EXCHE	070C	0	0	S	EM: EXCHE	
DIV	0701	S*	S	S	EM: DIV	
DMSE	670D	0	0	S	EM: DMSE	
IMSE	470D	0	0	S	EM: IMSE	
JMPE	070D	0	0	S	EM: JMPE	
JSKE	0703	0	S	S	EM: JSKE	
JSTE	270D	0	0	S	EM: JSTE	
ORE	270E	0	0	S	EM: ORE	
SUBC	2702	0	S	S	EM: SUBC	
SUBE	070B	0	0	S	EM: SUBE	
XORE	070F	0	0	S	EM: XORE	
*DIV operands on the LSI 4/10 must be in the first quadrant. EM:DIV emulates the DIV instuction for operands in the other three quadrants.						
SHIFT AND BIT MANIPULATION INSTRUCTIONS						
MNEMONIC	SKELETON	STANDARD(S)/OPTIONAL(O)			REFERENCE TO EMLTR4 ROUTINE	
		4/10	4/30	4/90		
ROTATEL(DOUBLE)	0F02	0	S	S	EM: RDL	
ROTATELO(DOUBLE)	0F03	0	S	S	EM: RDLO	
ROTATER	0E0A	0	S	S	EM: RR	
ROTATER(DOUBLE)	DF0A	0	S	S	EM: RDR	
ROTATERO(DOUBLE)	0F0B	0	S	S	EM: RDRO	
SHIFTL	0E00	0	S	S	EM: SL	
SHIFTL(DOUBLE)	0F00	0	S	S	EM: SDL	
SHIFTLO(DOUBLE)	0F01	0	S	S	EM: SDLO	
SHIFTR	0E08	0	S	S	EM: SR	
SHIFTR(DOUBLE)	0F08	0	S	S	EM: SDR	
SHIFTRA(DOUBLE)	0F0C	0	S	S	EM: RDRA	
SHIFTRO(DOUBLE)	0F09	0	S	S	EM: SDRO	

Tabke 3-1. Optional Firmware Instructions (Page 2 of 2)

FLOATING POINT INSTRUCTIONS						
MNEMONIC	SKELETON	STANDARD(S)/OPTIONAL(O)			REFERENCE TO	
		4/10	4/30	4/90	EMLTR4	ROUTINE
ADDF	5C00	0	0	0	EM: ADDF	
CSKF	5C04	0	0	0	EM: CSKF	
DIVF	5C03	0	0	0	EM: DIVF	
FIX	030C	0	0	0	EM: FIX	
FLOAT	030D	0	0	0	EM: FLOAT	
MULF	5C02	0	0	0	EM: MULF	
SUBF	5C01	0	0	0	EM: SUBF	

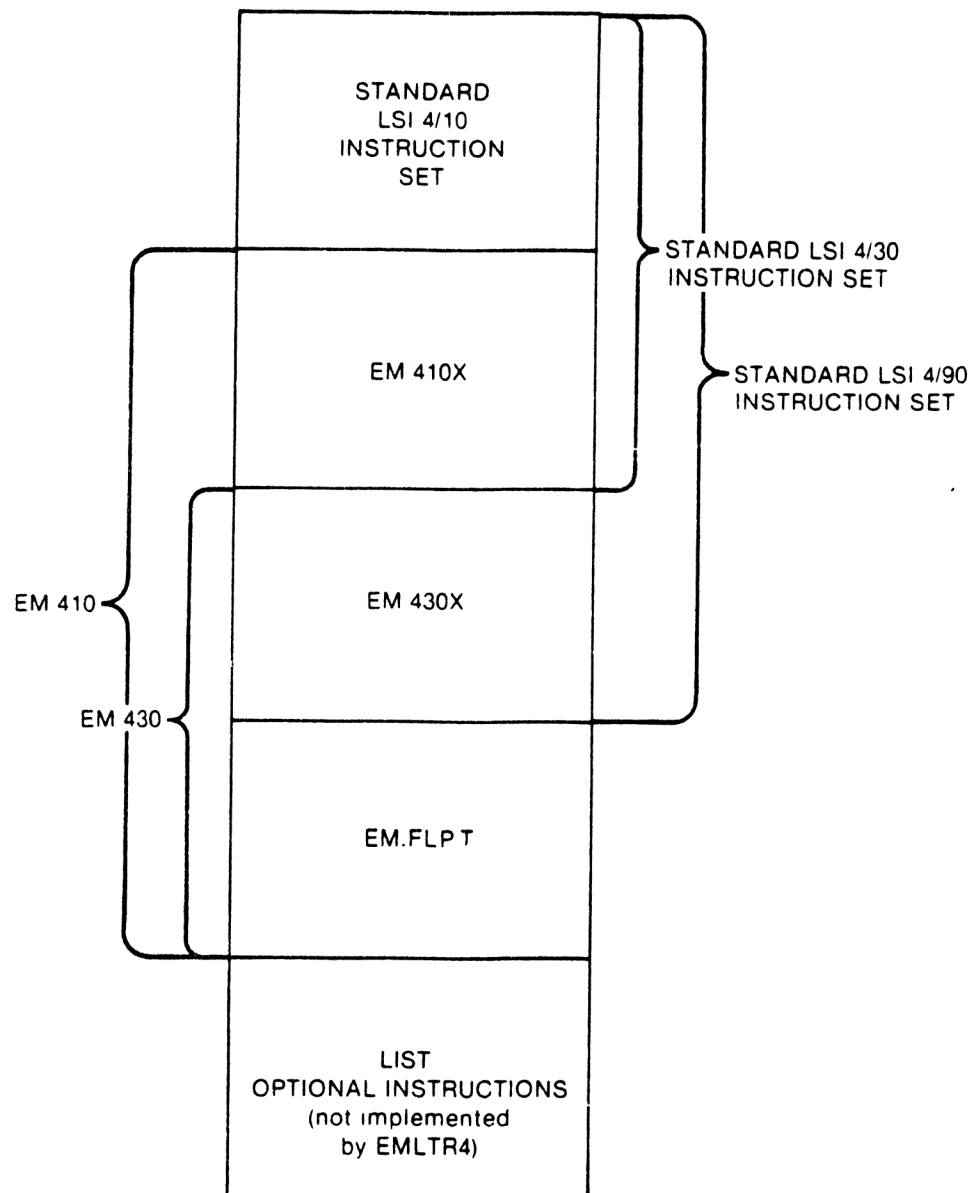
The programmer can direct EMLTR4 to emulate an entire set of instructions by using one of the following module names as a LOAD directive operand. These modules contain only LOAD directives referencing individual instruction modules.

- EM:410X references all instructions optional to the LSI 4/10 processor, but standard on the LSI 4/30 processor
- EM:430X references all instructions optional to the LSI 4/30 processor (less those in EM:410X) but standard on the LSI 4/90 processor
- EM:FLPT references all floating point instructions

The programmer can use one of the following module names as a LOAD directive operand to emulate an even larger set of instructions. These modules contain LOAD directives referencing combinations of the EM:410X, EM:430X, and EM:FLPT modules, which in turn reference the individual instruction modules.

- EM:410 references all instructions optional to the LSI 4/10.
- EM:430 references all instructions optional to the LSI 4/30.

Figure 3-1 diagrams the modular structure of EMLTR4. Table 3-1 lists the optional instructions which make up the five modules and specifies whether each instruction is standard or optional on each of the three NAKED MINI® 4 Family models.



6-1

Figure 3-1. LOAD Directive Operands Used to Reference EMLTR4 Instruction Modules

3.1.2 Using Noninterruptable Code

The programmer should exercise care when using the SIN instruction to suppress interrupts for a specified number of instruction executions. Since each instruction executed in the emulation of an optional instruction is considered a separate instruction, the programmer may have difficulty determining a suitable operand for the SIN instruction.

3.1.3 Preparing a User Program for Execution Under RTX4

A user program that includes instructions to be emulated be executed under RTX4. In creating the user program, the programmer must follow all special instructions detailed in this section. In addition, he must include the following LOAD directive in his program:

LOAD R:EMLTR

When coded as part of a user program, this LOAD directive instructs RTX4 to load an EMLTR4 interface module during the linking process. When this directive is not included, the user program is generated without EMLTR4 support in RTX4.

3.1.4 Preparing a User Program for Standalone Execution

A user program executed as a standalone program (i.e., not linked with RTX4 or OS4), but which includes instructions to be emulated can also use EMLTR4. In this case the user must provide a UIT processor which interfaces to EMLTR4.

The UIT processor should include a JSK instruction to location R:EMLTR. The programmer should also provide a return location R:UIRTN to handle actual unimplemented instructions. An example of such a routine is provided as Figure 3-2.

3.2 ASSEMBLING, LINKING, AND EXECUTING THE USER PROGRAM

After the user program is correctly coded, the programmer assembles it using the NAKED MINI® 4 Assembler, thereby deriving an object program in the NAKED MINI® 4 Object Program Format required for linking under OS4. The programmer then must ascertain that the EMLTR4 diskette file is accessible for linking (e.g., by copying it to his user program diskette). Linking of EMLTR4 to the user program occurs when the programmer enters the /LINK command under OS4 control, including the file EMLTR.LIB as a parameter.

The linked program can be executed normally under the OS4, OMEGA4, or RTX4 system. The programmer should note that execution speed may be appreciably less than if the optional instructions were implemented in firmware.

0051	0061	NAM	R:UITH
	0066	EXTN	R:EMUL,R:CNTH,R:USTH,R:AETH
* ENTER HERE FOR EMULATION CAUSED CONTEXT SAVE. * MAY REQUIRE DISABLING INTERRUPTS BY RESETTNG HIT & OF STATUS. * ENTRY IS BY: R:UITH * JSK * EXITS WITH CONTEXT ON USER STACK, MUST DISARLE). * INTERRUPTS UNCHANGED (USER, MUST DISARLE).			
0051 4309	0179	R:UITH	XA:V
0052 0040		PUSH	SAVE USER CONTEXT
0053 9F83 0087	0180	JMP	*VEXTON THEN INVOKE FMULATOR IF ONE IS PRESENT.
* UNIMPLEMENTED INSTRUCTION TRAP			
0084 0000	0251	AHS	UITA:
0086 FE4A 0051	0252	RES	2,0
0087 0000	0253	JSK	R:UITH
	0254	WORD	R:EMUL
	0255	VEXTON	ADDRESS OF THE EMULATOR ROUTINE
	0256	RTX:	
0000 0002	0328	KEL	
SYSTEM			
0000 8000	0352	KEL	DUMMY:
0000 0002	0353	SHARABLE	DUMMY:
	0354	ROMMARLE	DUMMY:
0000	0355	NAM	R:FMUL
	0356	EXTN	R:TRAPX
	0357	EQU	\$
0000 0000	0358	R:EMUL	
0000 8382 0003	0359	COPY	UITA:+1,A
0001 4900	0360	COPY	=XV:UINTP,N
0002 9F81 0004	0361	JMP	R:TRAPX
0003 0085			
0004 0000 R:TRAPX	0362	FND	
			A = INSTRUCTION CAUSING TRAP W = OFFSET INTO EXCEPTION PROCESSING VECTOR PROCFSS EXCEPTION

Figure 3-2. Unimplemented Instruction/Emulator Interface