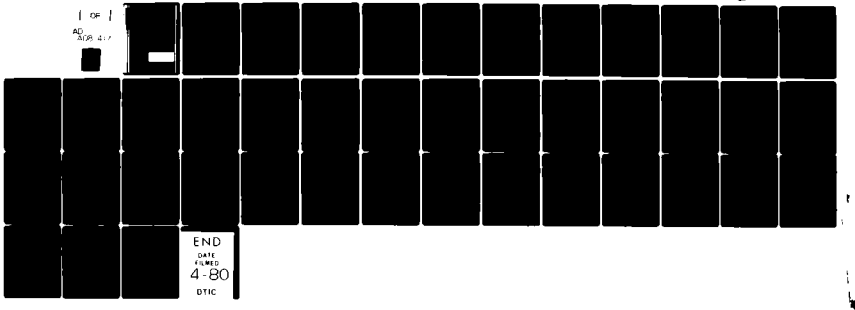
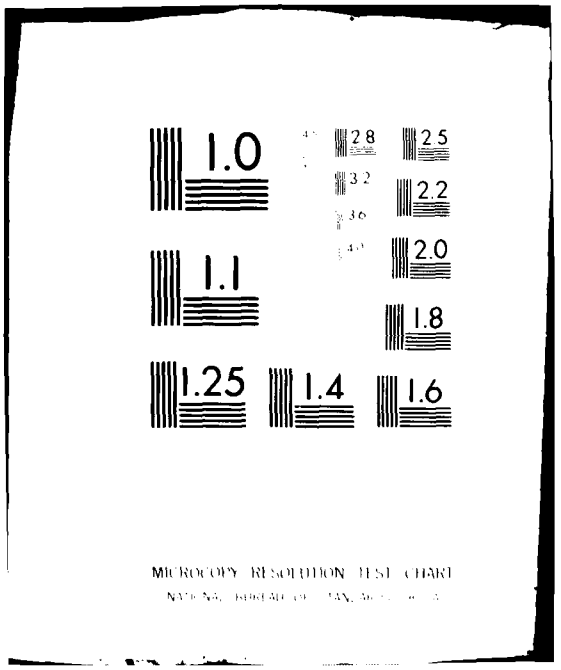
AD-A081 417 DEFENCE RESEARCH ESTABLISHMENT VALCARTIER (QUEBEC) F/G 9/2
AN ASSEMBLER AND SIMULATOR FOR THE 8048/8748/8035 INTEL MICROCO--ETC(U)
JAN 80 R CARBONNEAU, B MONTMINY, P COTE
UNCLASSIFIED DREV-R-4162/80 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
MICROCOPY RESOLUTION TEST CHART

UNCLASSIFIED
UNLIMITED DISTRIBUTION

CRDV RAPPORT 4162/80
DOSSIER: 3633A-010
JANVIER 1980

DREV REPORT 4162/80
FILE: 3633A-010
JANUARY 1980

AN ASSEMBLER AND SIMULATOR FOR THE 8048/8748/8035 INTEL MICROCOMPUTERS

R. Carbonneau

B. Montminy

P. Côté

Centre de Recherches pour la Défense
Defence Research Establishment
Valcartier, Québec

BUREAU - RECHERCHE ET DEVELOPPEMENT
MINISTERE DE LA DÉFENSE NATIONALE
CANADA

RESEARCH AND DEVELOPMENT BRANCH
DEPARTMENT OF NATIONAL DEFENCE
CANADA

CRDV R-4162/80
DOSSIER: 3633A-010

UNCLASSIFIED

DREV-R-4162/80
FILE: 3633A-010

AN ASSEMBLER AND SIMULATOR FOR THE 8048/8748/8035 INTEL MICROCOMPUTERS

by

R. Carbonneau, B. Montminy and P. Côté

CENTRE DE RECHERCHES POUR LA DEFENSE

DEFENCE RESEARCH ESTABLISHMENT

VALCARTIER

Tel: (418) 844-4271

Québec, Canada

January/janvier 1980

NON CLASSIFIE

RESUME

Nous présentons un programme FORTRAN servant à la traduction des mnémoniques en code machine et à la simulation d'un programme écrit pour les micro-ordinateurs de type 8048/8748/8035. Ce programme, qui permet l'utilisation du langage symbolique d'Intel et l'emploi d'étiquettes pour les instructions "sauts", simule exactement le comportement du micro-ordinateur dans les applications réelles. Il permet en outre la simulation d'interruptions et l'impression des résultats intermédiaires. (NC)

ABSTRACT

A FORTRAN program used to translate man-readable statements into machine-understandable code and to simulate a program written for the Intel 8048/8748/8035 microcomputers is described. This program allows programming of the microprocessor in symbolic language and the use of labels for jump instructions. The simulator duplicates exactly the behavior of the microcomputer in a real-world application. It is also possible to simulate interrupts and print out intermediate results. (U)

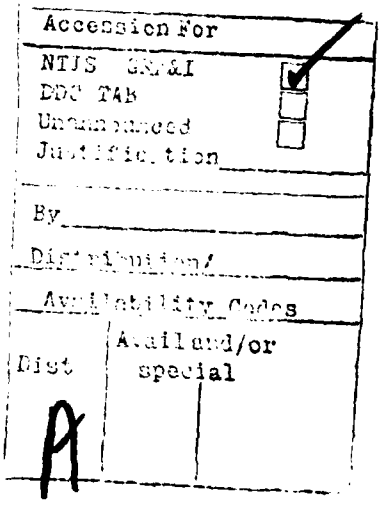


TABLE OF CONTENTS

TABLE OF CONTENTS

1.0 INTRODUCTION

An assembler and simulator for the 8080 microprocesor has already been described. The first version (Ref. 1) written in APL was superseded by a new version written in FORTRAN-IV language (Ref. 2).

On the basis of the previous experience, an assembler and simulator was written for the Intel 8048/8748/8035 microcomputers in FORTRAN-IV language. Files are used to store any assembler program and the simulated microprocessor memory to permit unlimited-length program edition. It is also possible to transfer directly the data generated by the assembler to a 8748/PROM programmer similar to the one described in Ref. 3.

The FORTRAN language permits the fast execution of the program instructions and the use of the editor supplied with a computer system.

The FORTRAN program first translates the Intel 8048/8748/8035 program into machine language; gives an address to all labels; finds errors in the program, if any; and executes the program exactly as the microcomputer. It also simulates interrupts and prints out intermediate data during execution. The technique used in this program is the same as that used in Ref. 2 and may be extended to any other microprocessor or microcomputer.

Section 2.0 briefly describes the principle of operation of the assembler and the simulator, Section 3.0 gives details of their use and Section 4.0 the procedure to follow to program the 8748 or the PROMs for the 8035. The work was performed at DREV between June and July, 1977 under PCN 33A10, Improvement to equipment.

2.0 PRINCIPLE OF OPERATION

2.1 The Assembler

Program assembling is done by a two-pass assembler. In the first pass, all the binary coding of instructions is accomplished except for the coding of jump addresses in all jump and call instructions. A table containing all labels with their respective addresses is also constructed. Any syntax error, illegal instruction or illegal argument will also be detected in the first pass. The second pass is exclusively used to give values to byte 2 and byte 3 of all jump and call instructions. Illegal jumps will be detected in the first or in the second pass.

During the assembling, the ROM content is kept on the computer file system (file #2) up to a maximum of 4096 memory positions, and two other files are also constructed; these are only scratch pad files used in the simulation.

The present program limitations, 5000 lines and 500 labels, are artificial and may be changed if the need arises.

2.2 The Simulator

After the program has been assembled, a simulation may begin. The simulation faithfully represents the operation of the microcomputer in a real-world application. In particular, the jumps, the calls to subroutines, the returns from subroutines, the stack, the flags and the internal and external RAM are manipulated in the same way as in the 8048/8748/8035 microcomputers. It is also possible to print out the microcomputer status, at any time, with the PRIN command and to output a memory map of 64 bytes located between 0 and 1088 from the data memory. These print-outs will not interfere with the result of the assembling. These commands are used to analyze and to monitor the program. Similarly, it is also possible to simulate both external and timer/event-counter interrupts at any point in the program.

A few other special instructions may be used throughout the program; their complete description is given in the next section.

3.0 UTILIZATION OF THE PROGRAM

One must first write his program in a file with a currently available editor on a computer system. The program must then be copied in order to remove any deleted line and to reorder any fractional numbered line. On the DREV editor, the command is "COPY name OVER name, 1" where name is the file name.

3.1 Basic Instruction Set

The following remarks should be taken into account while writing the program.

1) There must be at least one instruction per line. Each instruction may optionally be followed by a semicolon and a comment. The maximum length of a line is 128 characters. There is no restriction on the comment content.

2) An instruction may be optionally preceded by a label and a colon. The latter is used only if a label is present. The label may be of any length, but only the first four (4) characters will be meaningful. During the assembling, a label table is constructed and is printed at the end. The only restriction on the label characters is that they must not be a space, a semicolon or a colon. Nonexecutable instructions should not have a label.

3) All instructions must be followed by their appropriate arguments. These are described in Ref. 4 and are reproduced in Appendix A. There must be a space between the mnemonic and the first argument and a comma between the arguments of a two-argument-instruction. Numerical arguments A must respect the following convention: A shall be a nonfractional decimal number with $0 \leq A < 256$.

4) The mnemonic MOVP3 used by Intel is replaced by MOV3 in the assembler/simulator.

3.2 Special Instructions

A few special instructions have been added to the basic instruction set to facilitate the assembling and the simulation. These are:

1) PRIN

This instruction is used without argument and will print out the processor status, i.e. the following: program counter (PC), instruction line number, accumulator, registers R0 to R7, stack pointer (SP), 4 flags (Carry, intermediate carry, F0 and F1), enable external interrupt (IEX), enable timer interrupt (ITIM), memory bank select (MB) and register bank (BS) flip-flops, and a special counter, ST, which calculates the number of cycles elapsed since the beginning of the simulation (if the processor is working at 6 MHz, one cycle represents 2.5 µs. PRIN will not affect the assembling and should be used to help in debugging a program.

2) INTE AND INTT

These instructions simulate an interrupt that will happen exactly at the position where the instruction is put in the program. INTE simulate an external interrupt, whereas INTT simulates an interrupt from the timer event-counter. No argument is needed. The assembled program will not be changed by these instructions. They may be used anywhere in the program to know the effect of an interrupt happening at this particular point.

3) END

This instruction must be the last one of a given program. No argument is needed and assembling is unaffected.

4) ASSI

This special instruction is used to assign a number ($0 \leq A < 256$) to a particular ROM address. One instruction must be used for each byte to be assigned. Simulation is not affected.

5) MAP

One uses this instruction, during the simulation, to print any 64-byte block of the 1088-byte data memory and this, without affecting the assembling. A numerical argument is required: MAP 0 is used to the print internal data memory block and Map 1 to Map 16, to print external data memory blocks (64 bytes).

6) VAR

The VAR instruction is used to assign an address to a label which is not part of the program being assembled. It affects only the assembling. A label is required for this instruction with, as a right argument, the address to be associated to it.

7) BASE

This instruction is used to change the program counter value. The instruction following BASE in the program will have the value of the argument of BASE as program counter.

8) ASSD

This instruction may be used to facilitate the addressing of a portion of a program in a relocatable context. The first argument must be a label and the second one must represent the address where the label address will be stored (two bytes are written). This instruction affects only the assembling.

NOTE: All nonexecutable instructions immediately following a CALL, JMP, JMPP, DJNZ and conditional Jumps will not be executed during the simulation. There is no restriction for the assembling.

Once the FORTRAN program, given in Appendix B, has been compiled (the object program is called OBJ2) and your program is ready, the procedure is started by typing

```
SET F:1,DC/NAME;IN
SET F:2,DC/F2;INOUT;SAVE
SET F:3,DC/F3;INOUT;SAVE
SET F:4,DC/F4;INOUT;SAVE
```

Where NAME identifies the file in which the 8048 program is stored. F2 is the memory file, whereas F3 and F4 are two additional scratch pad files. These SETs should be entered only once per LOGON. The execution of OBJ2 is initiated by RUN OBJ2.

3.3 Error Messages

1) Label error: A label is missing or there are two or more similar ones.

2) Argument error: The argument is unacceptable or is missing.

3) Syntax error: Probably a nonexisting instruction.

4) BAD KEY or MISSING RECORD: An attempt has been made to write or read in an undefined record. If this happens during the simulation, one must check if:

a) The stack pointer or the stack content is wrong;
b) the program is longer than 5000 lines;or if
c) there are more than 500 labels.

4.0 INTERFACE WITH THE 8748/PROM PROGRAMMER

The 8048, 8748 and 8035 microcomputers differ from each other as follows:

a) The 8048 data memory is a ROM. It must then be mask-programmed at the factory;
b) the 8748 program memory is an EPROM; and
c) the 8035 has no internal program memory.

Consequently, only the 8748 can be programmed by a special programmer, whereas for the 8035 the program has to be put into a conventional PROM.

To permit the transfer of the assembled program from the computer to the programmer, the following steps should be taken.

The following program MOR is compiled and linked to APLFNS.LPR

```
 1.000          INTEGER TYPE(2),SIZE(2)
 2.000          INTEGER R(256,4)
 3.000          TYPE(1)=2
 4.000          TYPE(2)=2
 5.000          SIZE(1)=1
 6.000          SIZE(2)=1024
 7.000          PRINT 2
 8.000 2        FORMAT('WHAT IS THE STARTING MEMORY PAGE?')
 9.000          DEFINE FILE 2(256,256,U,ICA)
10.000          READ 1,N
11.000        1 FORMAT(I)
12.000          N=N+1
13.000          READ(2'N)(R(I,1),I=1,256)
14.000          READ(2'N+1)(R(I,2),I=1,256)
15.000          READ(2'N+2)(R(I,3),I=1,256)
16.000          READ(2'N+3)(R(I,4),I=1,256)
17.000          CALL FTIE(5,'FA')
18.000          CALL FREPLACE(5,1,R,SIZE,TYPE)
19.000          CALL FUNTIE(5)
20.000          CALL EXIT
21.000           END
```

The object program is called MORP and its execution is initiated by

```
START MORP
```

To the question "What is the starting memory page?", one must answer by a number A ($0 \leq A \leq 15$). The program will then transfer 4 pages (1024 bytes) in an APL file named FA. When the following operations are completed:

```
APL
)LOAD W
ASSPROM
)SAVE
```

a vector A of 1024 elements is saved in the workspace W. This vector is directly accessible by a PROM programmer similar to the one described in Ref. 3 but adapted to the 8748/8035 microcomputers. The listing of ASSPROM is:

```
      ∇ASSPROM
[1]   'FA'FTIE 6
[2]   A←FREAD 6,1
[3]   FUNTIE 6
      ∇
```

5.0 CONCLUSION

An assembler and simulator for the 8048, 8748 and 8035 microcomputers has been written. Its performance and speed are similar to those of the assembler and simulator used for the 8080.

After the assembling performed in two passes, it is possible to execute the program while simulating the exact microcomputer behavior. This simulator will considerably speed up design and debugging of programs written for the 8048 series microcomputers.

The technique used for this assembler and simulator can be accommodated to any type of microprocessor or microcomputer.

6.0 ACKNOWLEDGEMENTS

The authors express special thanks to Mr. J.-N. Bérubé who has developped the technical programming used throughout this work.

7.0 REFERENCES

1. Bérubé, J.N., "An assembler and a simulator for the 8080 microprocessor" DREV M-2402/76, June 1976, UNCLASSIFIED

2. Bérubé, J.N., Carbonneau, R., Montminy, B., Côté, P., "An improved version of an assembler and simulator for the 8080 microprocessor" DREV R-4124/78, November 1978, UNCLASSIFIED

3. Montminy, B., Bérubé, J.N., Carbonneau, R., Côté, P., "An Automatic Programmer for the 2708/2704 Erasable Programmable Read Only Memory" DREV R-4131/79, February 1979, UNCLASSIFIED

4. Intel MCS-48 Microcomputer User's Manual, November 1976.

APPENDIX A

INSTRUCTION SET

No.	INSTRUCTIONS	ACCEPTED ARGUMENTS	REMARKS
1	ADD	A,R_r	R_r = R0 to R7
		A,@R	R = R0 or R1
		A,#data	$0 \leq$ data ≤ 255
2	ADDC	A,R_r	
		A,@R	
		A,#data	
3	ANL	A,R_r	
		A,@R	
		A,#data	
		P1,#data	
		P2,#data	
		BUS,#data	
4	ORL	A,R_r	
		A,@R	
		A,#data	
		P1,#data	
		P2,#data	
		BUS,#data	
5	XRL	A,R_r	
		A,@R	
		A,#data	

No.	INSTRUCTIONS	ACCEPTED ARGUMENTS	REMARKS
6	INC	A	
		R_r	
		@R	
7	DEC	A	
		R_r	
8	CLR	A	
		C	
		F0	
		F1	
9	CPL	A	
		C	
		F0	
		F1	
10	DA	A	
11	SWAP	A	
12	RL	A	
13	RLC	A	
14	RR	A	
15	RRC	A	
16	IN	A,P1	
		A,P2	
17	OUTL	P1,A	
		P2,A	
		BUS,A	
20	INS	A,BUS	
21	MOVD	A,P_p	P_p = P4 to P7
		P_p,A	
22	ANLD	P_p,A	

No.	INSTRUCTIONS	ACCEPTED ARGUMENTS	REMARKS
23	ORLD	P_p,A	
24	JMPP	@A	
25	JMP	LABEL	
26	DJNZ	R_r,LABEL	
27	JC	LABEL	
28	JNC	LABEL	
29	JZ	LABEL	
30	JNZ	LABEL	
31	JTO	LABEL	
32	JNTO	LABEL	
33	JT1	LABEL	
34	JNT1	LABEL	
35	JF0	LABEL	
36	JF1	LABEL	
37	JTF	LABEL	
38	JNI	LABEL	
39 to 46	JB0 to JB7	LABEL	
47	CALL	LABEL	
48	RET		
49	RETR		
50	MOV	A,R_r	
		A,@R	
		A,#data	
		R_r,A	
		@R,A	
		R_r,#data	
		@R,#data	
		A,PSW	
		PSW,A	
		A,T	
		T,A	

No.	INSTRUCTIONS	ACCEPTED ARGUMENTS	REMARKS
51	XCH	A,R_r	
		A,@R	
52	XCHD	A,@R	
53	MOVX	A,@R	
		@R,A	
54	MOVP	A,@A	
55	MOV3	A,@A	
56	STRT	T	
		CNT	
57	STOP	TCNT	
58	EN	TCNTI	
		I	
59	DIS	TCNTI	
		I	
60	SEL	RB0	
		RB1	
		MB0	
		MB1	
61	ENTO	CLOCK	
62	NOP		
63	INTE		
64	PRIN		
65	END		
66	ASSI	AA,data	$0 \leq AA \leq 4096$
67	MAP	0 to 16	
68	LABEL: VAR	AA	
69	BASE	AA	
70	ASSD	LABEL, AA	
71	INTT		

APPENDIX B

LISTING OF THE FORTRAN PROGRAM

(ASSEMBLER AND SIMULATOR FOR THE INTEL 8048/8748/8035).

```
1.000        DIMENSION I(32),IR12(4),LABR(500),IPCR(500),KS(64)
2.000        DIMENSION CTT(10),JUMP(21),PEXP(4),PEXPV(4)
3.000        DIMENSION IDM(1088)
4.000        COMMON IBUF(3,256),NC(3),IFL(3)
5.000        DATA IEI,LABC,IPC,JJ,INS,IST,IL,ITCNTI/0,0,0,0,0,0,1,0/
6.000        DATA JUMP/246,230,198,150,54,38,86,70,182,118,22,134,
7.000       118,50,82,114,146,178,210,242,4/
8.000        DEFINE FILE 2(16,256,U,ICA),3(20,256,U,ICB),4(16,256,U,ICB)
9.000        INTEGER CP,PC,CODE,REG,DAT1,IR2P,TB,TF,S,ACC,PSW,MB,BUS
10.000      1,P1,P2,F0,F1,ICY1,BUSV,SP,BS
11.000       BUSV=0
12.000       DO 476 J=1,3
13.000       DO 486 K=1,256
14.000 486   IBUF(J,K)=0
15.000       PEXPV(J)=0
16.000       NC(J)=1
17.000 476   IFL(J)=0
18.000       PEXPV(4)=0
19.000 409   JJ=JJ+1
20.000       CALL SI(JJ,0,LABP,LAB,INST,IR1P,IR2P,IR12,I)
21.000       CALL REWR(JJ-1,2,IPC,1)
22.000       IF(LABP.EQ.0)GOTO 400
23.000       LABC=LABC+1
24.000       LABR(LABC)=LAB
25.000       IPCR(LABC)=IPC
26.500  400  IF(INST.GE.63) GOTO 475
27.000       CALL REWR (IL-1,3,JJ,1)
28.000       IL=IL+1
29.000       IF(INST.LT.25.OR.INST.GT.47)GOTO 602
30.000  603  CALL REWR (IL-1,3,JJ,1)
31.000       IL=IL+1
32.000       GOTO 475
33.000  602  IF (IR2P.NE.1)GOTO 475
34.000       CALL SEARCH (I,123,IR12(3),IR12(4),CP,PC)
35.000       IF(CP.EQ.1)GOTO  603
36.000   475 GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,409,409,20,21,22,
37.000      123,24,25,26,27,27,27,27,27,27,27,27,27,27,27,27,27,27,
38.000      227,27,27,27,27,27,27,48,49,50,51,52,53,54,55,
39.000      356,57,58,59,60,61,62,409,409,81,82,409,84,85,409,409),INST
40.000       PRINT 481,JJ
41.000   481 FORMAT('AT LINE NO ',I5,' THE INST. DOES NOT EXIST')
42.000       GOTO 409
43.000 703   CALL SYNTAXE (I,IR12(1),IR12(2),IR12(3),IR12(4),CODE,REG,DAT1,
44.000      1JJ,IR2P,INST)
45.000       GOTO (409,801,802,803,804,805,806,807,808),CODE+1
46.000  801  K=CTT(1)+REG
```

```
47.000          GOTO 703
48.000    802   K=CTT(2)+REG
49.000          GOTO 708
50.000    803   K=CTT(3)+(REG*16)
51.000          GOTO 708
52.000    804   K=CTT(4)
53.000          GOTO 708
54.000    805   K=CTT(5)
55.000          GOTO 708
56.000    806   K=CTT(6)
57.000          GOTO 708
58.000    807   K=CTT(7)
59.000          GOTO708
60.000    808   K=CTT(8)+32*REG
61.000    708   IF(K.GT.999)GOTO 509
62.000          IF(INST.EQ.26)GOTO 704
63.000          GOTO 707
64.000    509   PRINT 510,JJ
65.000     510  FORMAT ('WRONG ARGUMENTS AT LINE ',I5)
66.000          GOTO 409
67.000     547  CALL S1(I,IR12(1),L,1)
68.000          IF(L.EQ.215)GOTO 548
69.000          IF(L.NE.194)GOTO 509
70.000          K=K3+69
71.000          GOTO 550
72.000     548  CALL S1(I,IR12(1)+1,L,2)
73.000          DO 549 L1=241,242
74.000          IF(L.EQ.L1)GOTO 551
75.000     549  CONTINUE
76.000          GOTO 509
77.000     551  K=K3+L1-171
78.000     550  CALL CTE(I,IR12(3)+1,IR12(4),DAT1)
79.000          GOTO 704
80.000     701  CALL   SYNTAXE(I,IR12(1),IR12(2),IR12(3),IR12(4),CODE,REG,DAT1,
81.000        1JJ,IR2P,INST)
82.000          IF (CODE.EQ.1.OR.CODE.EQ.8) K=K1+REG
83.000          IF (CODE.EQ.2) K=K2+REG
84.000          IF (CODE.EQ.3)GOTO 702
85.000          IF(CODE.EQ.4)K=K3
86.000          IF (INST.EQ.50.AND.CODE.EQ.0)GOTO 527
87.000          IF(INST.EQ.3.AND.CODE.EQ.0)GOTO 547
88.000          IF(INST.EQ.4.AND.CODE.EQ.0)GOTO 547
89.000          IF(INST.EQ.53.AND.CODE.EQ.0)GOTO 577
90.000          IF(CODE.EQ.6.OR.CODE.EQ.7)GOTO 527
91.000          IF(CODE.EQ.0)GOTO 409
92.000    707   CALL WR1(K,IPC)
93.000          GOTO 409
94.000     702  K=K3
95.000     704  CALL WRI(K,DAT1,IPC)
96.000          GOTO 409
97.000      1   K1=104
```

```
 98.000        K2=96
 99.000        K3=3
100.000        GOTO 701
101.000     2  K1=120
102.000        K2=112
103.000        K3=19
104.000        GOTO 701
105.000     3  K1=88
106.000        K2=80
107.000        K3=83
108.000        GOTO 701
109.000     4  K1=72
110.000        K2=64
111.000        K3=67
112.000        GOTO 701
113.000     5  K1=216
114.000        K2=208
115.000        K3=211
116.000        GOTO   701
117.000     6  CALL SET(CTT)
118.000        CTT(4)=23
119.000        CTT(1)=24
120.000        CTT(2)=16
121.000        GOTO 703
122.000     7  CALL SET(CTT)
123.000        CTT(4)=7
124.000        CTT(1)=200
125.000        GOTO 703
126.000     8  CALL SET(CTT)
127.000        CTT(4)=39
128.000        CTT(5)=151
129.000        CTT(8)=133
130.000        GOTO 703
131.000     9  CALL SET (CTT)
132.000        CTT(4)=55
133.000        CTT(5)=167
134.000        CTT(8)=149
135.000        GOTO 703
136.000    10  CALL SET(CTT)
137.000        CTT(4)=87
138.000        GOTO 703
139.000    11  CALL SET(CTT)
140.000        CTT(4)=71
141.000        GOTO 703
142.000    12  CALL SET(CTT)
143.000        CTT(4)=231
144.000        GOTO 703
145.000    13  CALL SET(CTT)
146.000        CTT(4)=247
147.000        GOTO 703
148.000    14  CALL SET(CTT)
```

```
149.000          CTT(4)=119
150.000          GOTO 703
151.000       15 CALL SET(CTT)
152.000          CTT(4)=103
153.000          GOTO 703
154.000       16 K1=9
155.000          GOTO 701
156.000       17 K1=57
157.000          K3=2
158.000          GOTO 701
159.000       20 K=8
160.000          GOTO 707
161.000       21 CALL SET(CTT)
162.000          CTT(1)=60
163.000          GOTO 703
164.000       22 CALL SET(CTT)
165.000          CTT(1)=156
166.000          GOTO 703
167.000       23 CALL SET(CTT)
168.000          CTT(1)=140
169.000          GOTO 703
170.000       24 K=179
171.000          GOTO 707
172.000       25 K=4
173.000          GOTO 704
174.000       26 IR2P=0
175.000          CALL SET(CTT)
176.000          CTT(1)=232
177.000          GOTO 703
178.000    27    K=JUMP(INST-26)
179.000          GOTO 704
180.000    48    K=131
181.000          GOTO 707
182.000    49    K=147
183.000          GOTO 707
184.000       50 S=0
185.000          K1=248
186.000          K2=240
187.000          K3=35
188.000          GOTO 701
189.000       51 CALL SET(CTT)
190.000          CTT(1)=40
191.000          CTT(2)=32
192.000          GOTO 703
193.000       52 CALL SET (CTT)
194.000          CTT(2)=48
195.000          GOTO 703
196.000       53 S=0
197.000          K2=128
198.000          GOTO 701
199.000       54 K=163
```

```
200.000          GOTO 707
201.000     55   K=227
202.000          GOTO 707
203.000    577   IF(S.EQ.1)GOTO 509
204.000          S=1
205.000          K2=144
206.000          GOTO 578
207.000     56   CALL SET(CTT)
208.000          CTT(7)=85
209.000          CTT(5)=69
210.000          GOTO 703
211.000     57   K=101
212.000          GOTO 707
213.000     58   CALL SET(CTT)
214.000          CTT(7)=37
215.000          CTT(6)=5
216.000          GOTO 703
217.000     59   CALL SET (CTT)
218.000          CTT(7)=53
219.000          CTT(6)=21
220.000          GOTO 703
221.000     60   CALL SET(CTT)
222.000          CTT(3)=197
223.000          GOTO703
224.000     61   K=117
225.000          GOTO 707
226.000     62   K=0
227.000          GOTO 707
228.000    527   IF(CODE.EQ.6)K=199
229.000          IF(CODE.EQ.7)K=66
230.000          IF(CODE.EQ.7.AND.S.EQ.1)K=98
231.000          IF (CODE.EQ.6.AND.S.EQ.1)K=215
232.000          IF(CODE.EQ.7.OR.CODE.EQ.6)GOTO 707
233.000          IF (S.EQ.1)GOTO 528
234.000          S=1
235.000          K1=168
236.000          K2=160
237.000    578   TB=IR12(1)
238.000          TF=IR12(2)
239.000          IR12(2)=IR12(4)
240.000          IR12(1)=IR12(3)
241.000          IR12(4)=TF
242.000          IR12(3)=TB
243.000          GOTO 701
244.000    528   CALL S1(I,IR12(3),L,3)
245.000          IF (L.EQ.124)GOTO 529
246.000          IF(L.NE.217)GOTO 509
247.000          CALL S1 (I,IR12(3)+1,L,4)
248.000          DO 533 L1=240,247
249.000          IF (L1.EQ.L) GOTO 537
250.000    533   CONTINUE
```

```
251.000        GOTO 509
252.000   537  CALL WR1 (184+L1-240,IPC)
253.000        GOTO 539
254.000   529  CALL S1 (I,IR12(3)+1,L,5)
255.000        IF(L.NE.217) GOTO 509
256.000        CALL S1(I,IR12(3)+2,L,6)
257.000        DO 531 L1=240,241
258.000        IF(L1.EQ.L)GOTO 532
259.000   531  CONTINUE
260.000        GOTO 509
261.000   532  CALL WR1 (176+L1-240,IPC)
262.000   539  CALL CTE(I,IR12(1)+1,IR12(2),K)
263.000        CALL WR1 (K,IPC)
264.000        GOTO 409
265.000 81     JJ=0
266.000        IPC=0
267.000 415    JJ=JJ+1
268.000        CALL SI(JJ,1,LABP,LAB,INST,IR1P,IR2P,IR12,I)
269.000        IF(INST.EQ.26)IR12(1)=IR12(3)
270.000        IF(INST.EQ.26)IR12(2)=IR12(4)
271.000        IF(INST.GE.25.AND.INST.LT.48) CALL LABEL(I,IR12(1),
272.000       1IR12(2),LABR,IPCR,IPC,LABC,JJ,INST)
273.000        IF(INST.GE.25.AND.INST.LT.48)GOTO 415
274.000        IF(INST.EQ.65)GO TO 412
275.000        IF(INST.EQ.69)GO TO 504
276.000        IF(INST.EQ.70)GOTO 505
277.000        IF (INST.GE.63)GO TO 415
278.000        IPC=IPC+1
279.000        IF(IR2P.EQ.0)GOTO 415
280.000        CALL S1(I,IR12(3),L,7)
281.000        IF(L.NE.123)GOTO415
282.000        IPC=IPC+1
283.000        GOTO 415
284.000 505    CALL CTE(I,IR12(1),IR12(2),K1)
285.000        K1=K1-1
286.000        CALL LABEL(I,IR12(3),IR12(4),LABR,IPCR,K1,LABC,JJ,INST)
287.000        GOTO 415
288.000  466   FORMAT (I)
289.000 412.   PRINT974
290.000 974    FORMAT('END OF ASSEMBLING')
291.000        IF(LABC.EQ.0)GOTO 485
292.000        PRINT 478
293.000   478  FORMAT(//,'THE LABELS ARE, WITH THEIR ADDRESSES')
294.000        DO 477 J=1,LABC
295.000   477  PRINT 479,IPCR(J),LABR(J)
296.000   479  FORMAT(5X,I8,5X,A4)
297.000        WRITE(2'NC(1))(IBUF(1,J),J=1,256)
298.000 485    JJ=0
299.000        PRINT 503
300.000 503    FORMAT('DO YOU WANT A SIMULATION? YES=1,NO=0')
301.000        READ 466,K3
```

```
302.000       IST=0
303.000       IF(K3.NE.1) GO TO 281
304.000       IPC=0
305.000       SP=0
306.000       BUSV=0
307.000       MB=0
308.000       BS=0
309.000       IEI=0
310.000       ITCNTI=0
311.000       F0=0
312.000       F1=0
313.000 410   JJ=JJ+1
314.000       CALL SI(JJ,2,LABP,LAB,INST,IR1P,IR2P,IR12,I)
316.000       GOTO (201,202,203,204,205,206,207,208,209,210,211,212,
317.000      1213,214,215,216,217,410,410,220,
318.000      2221,222,223,224,225,226,227,228,229,230,231,232,233,234,
319.000      3235,236,237,238,239,239,239,239,
320.000      4239,239,239,239,247,248,249,250,251,252,253,254,255,256,
321.000      5257,258,259,260,261,262,263,264,
322.000      6265,266,267,268,269,270,271),INST
323.000       GO TO 410
324.000  201  K1=104
325.000       K2=96
326.000       K3=3
327.000       K5=0
328.000  405  CALL GETOP(K1,K2,K3,K,IPC,K8,IST,IDM,BS)
329.000       CALL ICY(ICY1,(K+K5),ACC)
330.000  333  ACC=ACC+K+K5
331.000       CALL CAR(ICAR,ACC,256)
332.000       GOTO 410
333.000  202  K1=120
334.000       K2=112
335.000       K3=19
336.000       K5=ICAR
337.000       GOTO 405
338.000   787 IST=IST+1
339.000       CALL REWR(IPC-1,1,K4,0)
340.000       CALL REWR(IPC,1,K,0)
341.000       K10=ACC
342.000       GOTO(154,153,152),K9-K4
343.000  152  ACC=BUS
344.000       IF (BUSV.EQ.0)PRINT 483
345.000       GOTO 155
346.000  153  ACC=P1
347.000       GOTO 155
348.000  154  ACC=P2
349.000  155  IPC=IPC+1
350.000       GOTO 788
351.000   203 CALL GETOP(88,80,83,K,IPC,K8,IST,IDM,BS)
352.000       K9=155
353.000       K6=2
```

```
354.000        I1=2
355.000  444   K7=0
356.000        K5=1
357.000        IF(K8.EQ.1)GOTO 787
358.000   788  DO 442 J=1,8
359.000        K3=0
360.000        K1=K-(K/2)*2
361.000        K2=ACC-(ACC/2)*2
362.000        IF ((K1+K2).EQ.K6.OR.(K1+K2).EQ.I1)K3=1
363.000        K7=K3*K5+K7
364.000        ACC=ACC/2
365.000        K=K/2
366.000  442   K5=K5*2
367.000        ACC=K7
368.000        IF(K8.EQ.0)GOTO 410
369.000        GOTO(354,353,352),K9-K4
370.000  352   BUS=ACC
371.000        PRINT 435,JJ,BUS
372.000        GOTO 411
373.000  353   P1=ACC
374.000        PRINT 436,JJ,1,P1
375.000        GOTO 411
376.000  354   P2=ACC
377.000        PRINT 436,JJ,2,P2
378.000  411   ACC=K10
379.000        GOTO 410
380.000  483   FORMAT ('NO DATA ON BUS')
381.000  204   CALL GETOP(72,64,67,K,IPC,K8,IST,IDM,BS)
382.000        K9=139
383.000        K6=1
384.000        I1=2
385.000        GOTO 444
386.000  205   CALL GETOP(216,208,211,K,IPC,K8,IST,IDM,BS)
387.000        K9=0
388.000        K6=1
389.000        I1=1
390.000        GOTO 444
391.000  206   K=1
392.000        CALL INCOP(23,24,16,K,IPC,IST,K8,ACC,IDM,BS)
393.000  899   CALL CAR(K,ACC,256)
394.000        GOTO 410
395.000  207   K=-1
396.000        CALL INCOP(7,200,0,K,IPC,IST,K8,ACC,IDM,BS)
397.000        GOTO 899
398.000  208   CALL REWR (IPC,1,K4,0)
399.000        IF(K4.EQ.39) ACC=0
400.000        IF(K4.EQ.151) ICAR=0
401.000        IF(K4.EQ.133) F0=0
402.000        IF(K4.EQ.165) F1=0
403.000  446   IPC=IPC+1
404.000        IST=IST+1
```

```
405.000       GOTO 410
406.000  209  CALL REWR (IPC,1,K4,0)
407.000       IF (K4.EQ.55) ACC=255-ACC
408.000       IF(K4.EQ.167) ICAR=ABS(ICAR-1)
409.000       IF(K4.EQ.149) F0=ABS(F0-1)
410.000       IF(K4.EQ.181) F1=ABS(F1-1)
411.000       GOTO 446
412.000  212  ACC=ACC*2
413.000       CALL CAR(K,ACC,256)
414.000       ACC=ACC+K
415.000       GOTO 446
416.000  213  ACC=ACC*2+ICAR
417.000       CALL CAR(ICAR,ACC,256)
418.000       GOTO 446
419.000  214  K1=ACC/2
420.000       ACC=K1+128*(ACC-K1*2)
421.000       GOTO 446
422.000  215  K1=ACC/2
423.000       K2=ACC-K1*2
424.000       ACC=K1+ICAR*128
425.000       ICAR=K2
426.000       GOTO 446
427.000  210  K1=ACC/16
428.000       K2=ACC-K1*16
429.000       IF(K2.GE.10.OR.ICY1.EQ.1)ACC=ACC+6
430.000       K1=ACC/16
431.000       IF(K1.GE.10.OR.ICAR.EQ.1)ACC=ACC+6*16
432.000       ICAR=0
433.000       CALL CAR(ICAR,ACC,256)
434.000       GOTO 446
435.000  211  K1=ACC/16
436.000       K2=ACC-(K1*16)
437.000       ACC=K1+K2*16
438.000       GOTO 446
439.000  216  CALL REWR (IPC,1,K4,0)
440.000       PRINT 465,JJ,IPC,K4-8
441.000  465  FORMAT('AT LINE NO ',I5,' (PC=',I4,') INPUT AT PORT NO '
442.000      1,I2,' IS:')
443.000       READ 466,K1
444.000       ACC=K1
445.000  437  IST=IST+1
446.000       GOTO 446
447.000  217  CALL REWR (IPC,1,K4,0)
448.000       IF (K4.EQ.2)PRINT 435,JJ,ACC
449.000       IF (K4.EQ.57.OR.K4.EQ.58)PRINT 436,JJ,K4-56,ACC
450.000  435  FORMAT('AT LINE NO ',I5,' BUS LATCHED AT ',I3)
451.000  436  FORMAT('AT LINE NO ',I5,' OUTPUT PORT NO ',I1,' LATCHED AT ',I3)
452.000       IF(K4.NE.2)GOTO 391
453.000       BUS=ACC
454.000       BUSV=1
455.000       GOTO 437
```

```
456.000  391  IF(K4.EQ.57)P1=ACC
457.000       IF(K4.EQ.58)P2=ACC
458.000       GOTO 437
459.000  474  PRINT 975
460.000  975  FORMAT('NO DATA ON BUS')
461.000       GOTO 437
462.000   220 PRINT 998,JJ,IPC
463.000  998  FORMAT('AT LINE NO ',I5,' (PC=',I4,') INPUT FROM BUS IS:')
464.000       READ 466,K1
465.000       ACC=K1
466.000       GOTO 437
467.000  221  CALL REWR (IPC,1,K4,0)
468.000       IF(K4.LE.15) GOTO 439
469.000       PRINT 432,JJ,K4-56,ACC-(ACC/16)*16
470.000  432  FORMAT('AT LINE NO ',I5,' EXPANSION PORT NO',I2,' IS',I3)
471.000       PEXP(K4-59)=ACC-ACC/16
472.000       PEXPV(K4-59)=1
473.000       GOTO 437
474.000  439  PRINT 931,JJ,IPC,K4-8
475.000   931 FORMAT ('AT LINE NO ',I5,'(PC= ',I4,' )INPUT'
476.000      1' AT EXPANSION PORT',I2,' IS:')
477.000       READ 466 ,K1
478.000       ACC=K1-(K1/16)*16
479.000       PEXP(K4-11)=ACC
480.000       PEXPV(K4-11)=1
481.000       GOTO 437
482.000  222  CALL REWR (IPC,1,K4,0)
483.000       K4=K4-155
484.000       IF (PEXPV(K4).EQ.0) GOTO 576
485.000       K6=2
486.000       GOTO 978
487.000  576  PRINT 977,JJ
488.000  977  FORMAT ('NODATA ON PORT AT LINE ',I5)
489.000       GOTO 437
490.000  223  CALL REWR (IPC,1,K4,0)
491.000       K4=K4-139
492.000       IF(PEXPV(K4).EQ.0)GOTO 576
493.000       K6=1
494.000  978  K7=0
495.000       K8=ACC
496.000       K9=PEXP(K4)
497.000       K5=1
498.000       DO 563 J=1,4
499.000       K3=0
500.000       K1=K-(K/2)*2
501.000       K2=K8-(K8/2)*2
502.000       IF((K1+K2).EQ.K6.OR.(K1+K2).EQ.2)K3=1
503.000       K7=K3*K5+K7
504.000       K8=K8/2
505.000       K9=K9/2
505.000  563  K5=K5*2
```

```
507.000       PEXP(K4)=K7
508.000       PRINT 564,JJ,K4+3,K7
509.000  564  FORMAT('AT LINE NO ',I5,' EXPANSION PORT ',I2,
510.000      1' LOGICALLY CHANGED TO ',I3)
511.000       GOTO 437
512.000  224  IPC=ACC+(IPC/256)*256
513.000   632 IST=IST+2
514.000  633  CALL REWR (IPC,3,K1,0)
515.000       JJ=K1-1
516.000       GOTO 410
517.000  225  CALL REWR (IPC,1,K1,0)
518.000       IPC=IPC+1
519.000       CALL REWR (IPC,1,K2,0)
520.000  656  IPC=K2+((K1/32)*256)+MB*2048
521.000       IST=IST+2
522.000       GOTO 633
523.000  226  K=-1
524.000       CALL INCOP(0,232,0,K,IPC,IST,K8,ACC,IDM,BS)
525.000       IST=IST+1
526.000       IPC=IPC+1
527.000       IF(K8.EQ.0) GOTO 410
528.000       IPC=IPC-1
529.000       CALL REWR (IPC,1,K1,0)
530.000       IPC=K1+(IPC/256)*256
531.000       GOTO 633
532.000  657  IST=IST+2
533.000       GOTO 410
534.000  227  IPC=IPC+2
535.000       IF(ICAR.EQ.0) GOTO 657
536.000  699  IPC=IPC-2
537.000       CALL REWR (IPC+1,1,K1,0)
538.000       IPC=K1+(IPC/256)*256
539.000       GOTO 632
540.000  228  IPC=IPC+2
541.000       IF(ICAR.EQ.1)GOTO 657
542.000       GOTO 699
543.000  229  IPC=IPC+2
544.000       IF(ACC.NE.0)GOTO 657
545.000       GOTO 699
546.000  230  IPC=IPC+2
547.000       IF(ACC.EQ.0)GOTO 657
548.000       GOTO 699
549.000  231  K2=0
550.000       K3=1
551.000  908  IPC=IPC+2
552.000       PRINT 928,JJ,K2
553.000  928  FORMAT('AT LINE NO ',I5,' T',I1,' SIGNAL IS:')
554.000       READ 466,K1
555.000       IF(K1.NE.K3)GOTO 657
556.000       GOTO 699
557.000  232  K2=0
```

```
558.000       K3=0
559.000       GOTO 908
560.000  233  K2=1
561.000       K3=1
562.000       GOTO 908
563.000  234  K2=1
564.000       K3=0
565.000       GOTO 908
566.000  235  IPC=IPC+2
567.000       IF(F0.NE.1)GOTO 657
568.000       GOTO 699
569.000  236  IPC=IPC+2
570.000       IF(F1.NE.1) GOTO 657
571.000       GOTO 699
572.000  237  IPC=IPC+2
573.000       PRINT 929,JJ
574.000  929  FORMAT('AT LINE NO ',I5,' TIMER FLAG IS:')
575.000       READ 466,K1
576.000       IF(K1.NE.1)GOTO 657
577.000       GOTO 699
578.000  238  IPC=IPC+2
579.000       IF(IEI.NE.0)GOTO 657
580.000       GOTO 699
581.000  239  CALL REWR (IPC,1,K1,0)
582.000       K1=K1/32
583.000       K1=2**K1
584.000       K1=ACC/K1
585.000       K1=K1-(K1/2)*2
586.000       IPC=IPC+2
587.000       IF (K1.NE.1)GOTO 657
588.000       GOTO 699
589.000   247 IF (INST.EQ.63.OR.INST.EQ.71)GOTO 839
590.000       IPC=IPC+2
591.000  839  K1=IPC-(IPC/256)*256
592.000       ICY1=ICY1-(ICY1/2)*2
593.000       ICAR=ICAR-(ICAR/2)*2
594.000       K2=(IPC/256)+16*BS+32*F0+64*ICY1+128*ICAR
595.000       IF(K2.GT.2047)K2=K2-2048
596.000       IDM(9+SP*2)=K1
597.000       IDM(10+SP*2)=K2
598.000       SP=SP+1
599.000       IF(SP.NE.8) GOTO 901
600.000  469  PRINT 471,JJ
601.000  471  FORMAT ('STACK POINTER OVERFLOWS AT LINE ',I5)
602.000       CALL EXIT
603.000  901  IF(INST.EQ.63.OR.INST.EQ.71)GOTO   781
604.000       CALL REWR (IPC-2,1,K1,0)
605.000       CALL REWR (IPC-1,1,K2,0)
606.000       GOTO 656
607.000  248  K3=0
608.000  721  SP=SP-1
```

```
609.000       IF(SP.EQ.-1)GOTO 469
610.000       K1=IDM(9+SP*2)
611.000       k2=IDM(10+SP*2)
612.000       IF (K3.EQ.0)GOTO 722
613.000       K3=K2/16
614.000       BS=K3-(K3/2)*2
615.000       K3=K3/2
616.000       F0=K3-(K3/2)*2
617.000       K3=K3/2
618.000       ICY1=K3-(K3/2)*2
619.000       K3=K3/2
620.000       ICAR=K3-(K3/2)*2
621.000  722  IPC=K1+256*(K2-(K2/16)*16)+MB*2048
622.000       IST=IST+2
623.000       GOTO 633
624.000  249  K3=1
625.000       GOTO 721
626.000  250  CALL GETOP(248,240,35,K,IPC,K8,IST,IDM,BS)
627.000       IF(K8.EQ.1) GOTO 907
628.000       ACC=K
629.000       GOTO 410
630.000  907  CALL REWR (IPC-1,1,K4,0)
631.000       IF(K4.GE.168.AND.K4.LE.175)IDM(1+K4-168+(24*BS))=ACC
632.000       IF(K4.EQ.160.OR.K4.EQ.161)GOTO 859
633.000       IF(K4.GE.176.OR.K4.LE.98) GOTO 858
634.000       GOTO 410
635.000  859  K=IDM(1+K4-160+(24*BS))
636.000  860  K=K-(K/64)*64
637.000       IDM(1+K)=ACC
638.000       GOTO 410
639.000  858  IF (K4.NE.66) GOTO 813
640.000       PRINT 811,JJ
641.000  811  FORMAT('AT LINE NO ',I5,' TIMER-EVENT/COUNTER VALUE IS:')
642.000       READ 466,K1
643.000       IF (K1.GT.255.OR.K1.LT.0) GOTO 811
644.000       ACC=K1
645.000       GOTO 410
646.000  813  IF (K4.NE.98) GOTO 816
647.000       PRINT 815,JJ,ACC
648.000  815  FORMAT ('AT LINE NO ',I5,' TIMER-EVENT/COUNTER PRESET AT ',I3)
649.000       GOTO 410
650.000  816  CALL REWR (IPC,1,K3,0)
651.000       IF (K4.LT.184) GOTO 817
652.000       IF(K4.GT.191)GOTO 819
653.000       IDM(1+K4-184+(24*BS))=K3
654.000  818  IST=IST+1
655.000       IPC=IPC+1
656.000       GOTO 410
657.000  817  K=IDM(1+K4-176+(24*BS))
658.000       K=K-(K/64)*64
659.000       IDM(1+K)=K3
```

```
660.000       GOTO 818
651.000  819  IF(K4.EQ.215)GOTO 820
662.000       ACC=(128*ICAR)+(64*ICY1)+(32*F0)+(16*BS)+8+SP
663.000       GOTO 410
664.000  820  SP=ACC-(ACC/8)*8
665.000       K1=ACC/16
666.000       BS=K1-(K1/2)*2
667.000       K1=K1/2
668.000       F0=K1-(K1/2)*2
669.000       K1=K1/2
670.000       ICY1=K1-(K1/2)*2
671.000       K1=K1/2
672.000       ICAR=K1-(K1/2)*2
673.000       GOTO 410
674.000  251  CALL REWR (IPC,1,K4,0)
675.000       IF (K4.GE.40) GOTO 733
676.000       K=IDM(1+K4-32+(24*BS))
677.000       K=K-(K/64)*64
678.000       K1=IDM(1+K)
679.000       IDM(1+K)=ACC
680.000  734  ACC=K1
681.000       GOTO 446
682.000  733  K1=IDM(1+K4-40+(24*BS))
683.000       IDM(1+K4-40+(24*BS))=ACC
684.000       GOTO 734
685.000  252  CALL REWR (IPC,1,K4,0)
686.000       K=IDM(1+K4-48+(24*BS))
687.000       K=K-(K/64)*64
688.000       K2=IDM(1+K)
689.000       K1=K2-(K2/16)*16
690.000       IDM(1+K)=((K2/16)*16)+(ACC-(ACC/16)*16)
691.000       ACC=K1+(ACC/16)*16
692.000       GOTO 446
693.000  253  CALL REWR (IPC,1,K4,0)
694.000       PRINT 767,JJ
695.000   767 FORMAT('AT LINE NO ',I5,' HARDWARED BANK SWITCHING FOR EXTERNAL '
696.000      1'DATA MEMORY IS (0-3):')
697.000       READ 466,K3
698.000       BUSV=0
699.000       IF(K4.GT.129) GOTO 766
700.000       ACC=IDM(65+(256*K3)+IDM(1+K4-128+(24*BS)))
701.000  765  IST=IST+1
702.000       GOTO 446
703.000  766  IDM(65+(256*K3)+IDM(1+K4-144+(24*BS)))=ACC
704.000       GOTO 765
705.000  254  IPC=IPC+1
706.000       K3=ACC+(IPC/256)*256
707.000  917  CALL REWR (K3,1,K4,0)
708.000       ACC=K4
709.000       IST=IST+2
710.000       GOTO 410
```

```
711.000  255  IPC=IPC+1
712.000       K3=ACC+3*256
713.000       GOTO 917
714.000  256  CALL REWR (IPC,1,K4,0)
715.000       IF (K4.EQ.85) PRINT 430,JJ
716.000       IF (K4.EQ.69) PRINT 431,JJ
717.000  430  FORMAT('AT LINE NO ',I5,' TIMER ENABLED')
718.000  431  FORMAT ('AT LINE NO ',I5,' EVENT COUNTER ENABLED')
719.000       GOTO 446
720.000  257  PRINT 492,JJ
721.000  492  FORMAT ('AT LINE NO ',I5,' TIMER /EVENT COUNTER DISABLED')
722.000       GOTO 446
723.000  258  CALL REWR (IPC,1,K4,0)
724.000       IF(K4.EQ.37) GOTO 833
725.000       PRINT 434,JJ
726.000  434  FORMAT ('AT LINE NO',I5,' EXTERNAL INTERRUPT ON')
727.000       IEI=1
728.000       GOTO 446
729.000  833  PRINT 495,JJ
730.000  495  FORMAT ('AT LINE NO',I5,' TIMER-EVENT/COUNTER INTERRUPT ON')
731.000       ITCNTI=1
732.000       GOTO 446
733.000  259  CALL REWR (IPC,1,K4,0)
734.000       IF (K4.EQ.53) GOTO 834
735.000       PRINT 496,JJ
736.000  496  FORMAT ('AT LINE NO',I5,' EXTERNAL INTERRUPT OFF')
737.000       IEI=0
738.000       GOTO 446
739.000  834  PRINT 497,JJ
740.000  497  FORMAT ('AT LINE NO',I5,' TIMER-EVENT/COUNTER INTERRUPT OFF')
741.000       ITCNTI=0
742.000       GOTO 446
743.000  260  CALL REWR (IPC,1,K4,0)
744.000       IF (K4.EQ.197) BS=0
745.000       IF (K4.EQ.213) BS=1
746.000       IF (K4.EQ.229) MB=0
747.000       IF (K4.EQ.245) MB=1
748.000       GOTO 446
749.000  261  PRINT 438,JJ
750.000   438 FORMAT ('AT LINE NO',I5,' CLOCK PRESENT ON TEST PIN 0')
751.000  262  GOTO 446
752.000  263  IF (IEI.EQ.0) GOTO 410
753.000       PRINT 971,JJ
754.000  971  FORMAT ('AT LINE NO ',I5,' EXTERNAL INTERRUPT BEING SERVED')
755.000       IEI=0
756.000       GOTO 247
757.000  271  IF(ITCNTI.EQ.0) GOTO 410
758.000       PRINT 472,JJ
759.000  472  FORMAT('AT LINE NO',I5,' TIMER-EVENT/COUNTER INTERRUPT'
760.000      1' BEING SERVED')
761.000       ITCNTI=0
```

```
762.000       GOTO 247
763.000  781  IF (INST.EQ.63)  IPC=3
764.000       IF (INST.EQ.71)  IPC=7
765.000       GOTO 633
766.000  264  PRINT 473,JJ,IPC,ACC,(IDM(BS*24+K),K=1,8),SP,IST,ICAR,
767.000      1ICY1,IEI,ITCNTI,MB,BS,F0,F1
768.000  473  FORMAT(' LINE   PC  ACC   R0   R1   R2   R3   R4   R5'
769.000      1'   R6   R7   SP STEP  CAR   AC  IEX ITIM   MB'
770.000      2'   BS   F0   F1',/,21I5)
771.000       GOTO 410
772.000  265  GOTO 485
773.000  266  GOTO 410
774.000  267  CALL CTE(I,IR12(1),IR12(2),K)
775.000  499  DO 597 J=1,64
776.000   597 KS(J)=IDM(J+K*64)
777.000       PRINT 500,JJ,K,KS
778.000       GOTO 410
779.000  500  FORMAT('AT LINE NO',I5,' MEMORY MAP NO ',I4,'IS:',/,4(16I6,/))
780.000  268  GOTO 410
781.000  269  GOTO 410
782.000   270 GOTO 410
783.000  82   CALL CTE(I,IR12(1),IR12(2),K)
784.000       CALL CTE(I,IR12(3),IR12(4),K1)
785.000       CALL REWR(K,1,K1,1)
786.000       GOTO 409
787.000 85    CALL CTE(I,IR12(1),IR12(2),IPC)
788.000       IL=IPC+1
789.000       GO TO 409
790.000 504   CALL CTE(I,IR12(1),IR12(2),IPC)
791.000       GOTO 415
792.000 84    CALL CTE(I,IR12(1),IR12(2),K)
793.000       IPCR(LABC)=K
794.000       GOTO 409
795.000 281   WRITE(2'NC(1))(IBUF(1,J),J=1,256)
795.500       WRITE(4'NC(3))(IBUF(3,J),J=1,256)
796.000 467   CALL EXIT
797.000       END
798.000       SUBROUTINE CTE(I,LB,LF,IR)
799.000       DIMENSION I(32)
800.000        DECODE(LF,2,I)LB-1,LF-LB+1,IR
801.000     2 FORMAT(NX,IN)
802.000       RETURN
803.000       END
804.000       SUBROUTINE REWR (IAD,NF,K,M)
805.000       COMMON IBUF(3,256),NC(3),IFL(3)
806.000       K1=IAD/256
807.000       K2=IAD-K1*256+1
808.000       K1=K1+1
809.000       IF(K1.NE.NC(NF)) GO TO 1
810.000     6 IF (M) 2,3,2
811.000 3     K=IBUF(NF,K2)
```

```
812.000         RETURN
813.000 2       IBUF(NF,K2)=K
814.000         IFL(NF)=1
815.000         RETURN
816.000 1       IF(IFL(NF)) 4,5,4
817.000 5       READ(NF+1'K1)(IBUF(NF,J),J=1,256)
818.000         NC(NF)=K1
819.000         GO TO 6
820.000 4       WRITE(NF+1'NC(NF))(IBUF(NF,J),J=1,256)
821.000         IFL(NF)=0
822.000         GO TO 5
823.000         END
824.000         SUBROUTINE WRI(I,J,IPC)
825.000         CALL REWR(IPC,1,I,1)
826.000         CALL REWR(IPC+1,1,J,1)
827.000          IPC=IPC+2
828.000         RETURN
829.000         END
830.000         SUBROUTINE WR1(I,IPC)
831.000         CALL REWR(IPC,1,I,1)
832.000         IPC=IPC+1
833.000          RETURN
834.000         END
835.000         SUBROUTINE CAR(IC,I,IM)
836.000         IC=0
837.000         IF(I.LT.IM) GO TO 1
838.000          IC=1
839.000         I=I-IM
840.000         RETURN
841.000 1       IF(I.GE.0) RETURN
842.000         IC=1
843.000         I=I+IM
844.000         RETURN
845.000         END
846.000         SUBROUTINE REGRR(IPC,K)
847.000         CALL REWR(IPC,1,K1,0)
848.000         K2=K1/16
849.000         K1=K1/64
850.000         K=K2-K1*4
851.000         K=K*2+1
852.000         RETURN
853.000         END
854.000         SUBROUTINE REGR(IPC,IRR,K)
855.000         DIMENSION IRR(10)
856.000         CALL REWR(IPC,1,K,0)
857.000         K=K-(K/8)*8
858.000         IF(K.EQ.6) GO TO 1
859.000         K=IRR(K+1)
860.000         RETURN
861.000 1       CALL REWR(IRR(5)*256+IRR(6),1,K,0)
862.000         RETURN
```

```
863.000        END
864.000        SUBROUTINE FFZSP(IZER,ISIG,IPAR,L)
865.000        IZER=0
866.000        IF(L.EQ.0) IZER=1
867.000        ISIG=0
868.000        IF(L.GE.128) ISIG=1
869.000        IPAR=0
870.000        I=L
871.000        DO 1 J=1,8
872.000        IPAR=I-(I/2)*2+IPAR
873.000 1      I=I/2
874.000        IPAR=IABS(IPAR-(IPAR/2)*2-1)
875.000        RETURN
876.000        END
877.000        SUBROUTINE S1(I,N,L,J)
878.000        DIMENSION I(32)
879.000        DATA MASK/8Z000000FF/
880.000      1 M=1+(N-1)/4
881.000      2 K=N-(M-1)*4
882.000      3 II=ISL(I(M),8*(K-4))
883.000      5 L=IAND(MASK,II)
884.000      4 RETURN
885.000        END
886.000        SUBROUTINE SEARCH(I,NC,LB,LF,CP,PC)
887.000        INTEGER CP,PC
888.000        DIMENSION I(32)
889.000        DO 1 N=LB,LF
890.000        CALL S1(I,N,L,8)
891.000        IF(L.EQ.NC) GO TO 2
892.000      1 CONTINUE
893.000        CP=0
894.000        RETURN
895.000      2 CP=1
896.000        PC=N
897.000        RETURN
898.000        END
899.000        SUBROUTINE SI(J,K,LABP,LAB,INST,IR1P,IR2P,IR12,I)
900.000        DIMENSION INS(71),IR12(4),I(32)
901.000        DATA INS/4HADD ,4HADDC,4HANL ,4HORL ,4HXRL ,4HINC ,
902.000       14HDEC ,4HCLR ,4HCPL ,4HDA  ,4HSWAP,4HRL  ,4HRLC ,
903.000       24HRR  ,4HRRC ,4HIN  ,4HOUTL,4H****,4H0000,4HINS ,
904.000       34HMOVD,4HANLD,4HORLD,4HJMPP,4HJMP ,4HDJNZ,4HJC  ,
905.000       44HJNC ,4HJZ  ,4HJNZ ,4HJT0 ,4HJNT0,4HJT1 ,4HJNT1,
906.000       54HJF0 ,4HJF1 ,4HJTF ,4HJNI ,4HJB0 ,4HJB1 ,4HJB2 ,
907.000       64HJB3 ,4HJB4 ,4HJB5 ,4HJB6 ,4HJB7 ,4HCALL,4HRET ,
908.000       74HRETR,4HMOV ,4HXCH ,4HXCHD,4HMOVX,4HMOVP,4HMOV3,
909.000       84HSTRT,4HSTOP,4HEN  ,4HDIS ,4HSEL ,4HENTO,4HNOP ,
910.000       94HINTE,4HPRIN,4HEND ,4HASSI,4HMAP ,4HVAR ,4HBASE,
911.000       A4HASSD,4HINTT/
912.000        INTEGER CP,PC
913.000        ENCODE (128,12,I)
```

```
914.000     12 FORMAT(4H      )
915.000        CALL DIRECT READ(1,I,32,J)
916.000        IR1P=0
917.000        IR2P=0
918.000        LB=1
919.000        LF=128
920.000        CALL SEARCH(I,94,LB,LF,CP,PC)
921.000        IF(CP.EQ.1) LF=PC-1
922.000        CALL CLSP(I,LF,-1)
923.000        LABP=0
924.000        CALL CLSP(I,LB,1)
925.000        CALL SEARCH(I,122,LB,LF,CP,PC)
926.000        IF(CP.EQ.1) GO TO 2
927.000 13     CALL CLSP(I,LB,1)
928.000        CALL SEARCH(I,64,LB,LF,CP,PC)
929.000        LFF=LF
930.000        IF(CP.EQ.1) LFF=PC-1
931.000        GO TO 4
932.000      2 IF(K.NE.0) GO TO 6
933.000        LABP=1
934.000        LAB=4H
935.000        LK=1
936.000        DO 7 L1=LB,PC-1
937.000        CALL S1(I,L1,L,9)
938.000        IF(L.EQ.1H )GO TO 7
939.000        L=ISL(L,8*(4-LK))
940.000        MASK=ISC(8ZFFFFFF00,8*(4-LK))
941.000        LAB=IOR(IAND(MASK,LAB),L)
942.000        IF(LK.EQ.4) GO TO 6
943.000 7      LK=LK+1
944.000      6 LB=PC+1
945.000        GO TO 13
946.000      4 INST=4H
947.000        LK=1
948.000        DO 9 L1=LB,LB+3
949.000        CALL S1(I,L1,L,10)
950.000        IF(L1.GT.LFF)L=64
951.000        L=ISL(L,8*(4-LK))
952.000        MASK=ISC(8ZFFFFFF00,8*(4-LK))
953.000        INST=IOR(IAND(MASK,INST),L)
954.000 9      LK=LK+1
955.000 14     LB=LFF+1
956.000        DO 10 L1=1,71
957.000        IF(INS(L1).EQ.INST) GO TO 11
958.000     10 CONTINUE
959.000     11 INST=L1
960.000        CALL CLSP(I,LF,-1)
961.000        IF(LB.GT.LF) RETURN
962.000        IR1P=1
963.000        CALL CLSP(I,LB,1)
964.000        CALL SEARCH(I,107,LB,LF,CP,PC)
```

```
 965.000        IR12(1)=LB
 966.000        IF(CP.EQ.1) GO TO 1
 967.000        IR12(2)=LF
 968.000        RETURN
 969.000      1 LK=PC-1
 970.000        CALL CLSP(I,LK,-1)
 971.000        IR12(2)=LK
 972.000        IR2P=1
 973.000        LB=PC+1
 974.000        CALL CLSP(I,LB,1)
 975.000        IR12(3)=LB
 976.000        IR12(4)=LF
 977.000        RETURN
 978.000        END
 979.000        SUBROUTINE CLSP(I,LB,J)
 980.000        DIMENSION I(32)
 981.000      2 CALL S1(I,LB,L,11)
 982.000        IF(L.NE.64)RETURN
 983.000        LB=LB+J
 984.000        GOTO 2
 985.000        END
 986.000        SUBROUTINE LABEL(I,K1,K2,LABR,IPCR,K3,LABC,JJ,INST)
 987.000        DIMENSIONI(32),LABR(500),IPCR(500)
 988.000        DECODE (K2,1,I)K1-1,K
 989.000 1      FORMAT(NX,A4)
 990.000        LAI=0
 991.000        DO 2 J=1,LABC
 992.000        IF(K.EQ.LABR(J)) LAI=LAI+1
 993.000 2      IF(K.EQ.LABR(J)) LAB=IPCR(J)
 994.000        IF(LAI.EQ.1)GOTO 3
 995.000        PRINT 4,LAI,K
 996.000 4      FORMAT('THERE ARE ',I3,' LABELS CALLED ',A4)
 997.000 3      K3=K3+1
 998.000        IF(INST.EQ.25.OR.INST.EQ.47.OR.INST.EQ.70)GOTO 17
 999.000        K1=(K3-1)/256
1000.000        IF(INST.EQ.26)K1=K3/256
1001.000        K2=LAB/256
1002.000        IF(K1.NE.K2.AND.INST.NE.70)GOTO 23
1003.000        LAB=LAB-K2*256
1004.000        CALL WR1(LAB,K3)
1005.000        RETURN
1006.000  23    PRINT 7,K,JJ
1007.000        GOTO 50
1008.000      6 IF(K1.GT.15) PRINT 7,K,JJ
1009.000        IF(K1.LE.15)PRINT 8,K,JJ
1010.000    7   FORMAT('LABEL  ',A4,' NOT ACCESSIBLE FROM LINE ',I5)
1011.000  8     FORMAT('MB MUST BE 1 FOR THE LABEL ',A4,' TO BE '
1012.000       1'ACCESSED FROM LINE ',I5)
1013.000        IF(K1.GT.15)GOTO 50
1014.000        LAB=LAB-2048
1015.000   17   K1=LAB/256
```

```
1016.000        IF(K1.GT.7)GOTO 6
1017.000        K2=4+K1*32
1018.000        IF (INST.EQ.47)K2=20+K1*32
1019.000        LAB=LAB-K1*256
1020.000        IF (INST.EQ.70)GOTO 18
1021.000        CALL WRI(K2,LAB,K3-1)
1022.000    50  K3=K3+1
1023.000        RETURN
1024.000     18 K2=K1
1025.000        CALL WRI(K2,LAB,K3)
1026.000        RETURN
1027.000        END
1028.000        SUBROUTINE SYNTAXE (I,LB1,LF1,LB2,LF2,CODE,REG,DAT,
1029.000       1JJ,IR2P,INST)
1030.000        DIMENSION I(32)
1031.000        INTEGER CP,PC,CODE,REG,DAT,TB,TF
1032.000        TB=LB1
1033.000        TF=LF1
1034.000        CODE=0
1035.000        REG=0
1036.000        DAT=0
1037.000        IF(INST.GE.21.AND.INST.LE.23)GOTO 650
1038.000        IF(INST.NE.17)GOTO 608
1039.000        LB1=LB2
1040.000        LF1=LF2
1041.000        LB2=TB
1042.000        LF2=TF
1043.000    608 CALL S1(I,LB1,L,12)
1044.000        IF(IR2P.EQ.0)GOTO 611
1045.000        IF (L.NE.193.OR.LB1.NE.LF1)GOTO 604
1046.000        CALL S1(I,LB2,L,13)
1047.000        IF(L.EQ.124) GOTO 601
1048.000        IF(L.EQ.123)GOTO 602
1049.000        IF(L.EQ.215)GOTO 612
1050.000        IF(L.EQ.194)GOTO 615
1051.000        IF(L.EQ.227)GOTO 644
1052.000        IF(L.NE.217) GOTO 604
1053.000     9  CALL S1(I,LB2+1,L,14)
1054.000        DO 10 L1=240,247
1055.000        IF(L1.EQ.L)       GOTO 11
1056.000   10   CONTINUE
1057.000        GOTO 604
1058.000   644  CODE=7
1059.000        RETURN
1060.000     11 CODE=1
1061.000   14   REG=L1-240
1062.000        RETURN
1063.000   601  CALL S1(I,LB2+1,L,15)
1064.000        IF(L.NE.217) GOTO 604
1065.000    34  CALL S1(I,LB2+2,L,16)
```

```
1067.000        IF (L1.EQ.L)GOTO        13
1068.000     12 CONTINUE
1069.000        GOTO 604
1070.000     13 CODE=2
1071.000        REG=L1-240
1072.000        IF(INST.EQ.60)CODE=3
1073.000        IF(DAT.EQ.1)REG=REG+2
1074.000        RETURN
1075.000   602  CALL CTE (I,LB2+1,LF2,K)
1076.000        IF(K.LT.0.OR.K.GT.255) GOTO604
1077.000        CODE=3
1078.000        DAT=K
1079.000        RETURN
1080.000   657  CODE=6
1081.000        RETURN
1082.000   604  IF(INST.EQ.50.OR.INST.EQ.53.OR.INST.EQ.3.OR.INST.EQ.4)RETURN
1083.000        PRINT 605,JJ
1084.000   605  FORMAT ('WRONG  ARGUMENTS AT LINE  ',I5)
1085.000        RETURN
1086.000   611  IF(L.NE.193)GOTO 613
1087.000   615  CODE=4
1088.000        RETURN
1089.000   612  CALL S1(I,LB2+1,L,17)
1090.000        IF(L.EQ.226)GOTO 657
1091.000        DO 16 L1=241,242
1092.000        IF(L1.EQ.L)GOTO 17
1093.000   16   CONTINUE
1094.000        GOTO 604
1095.000   614  CALL S1(I,LB2+1,L,18)
1096.000        DO 24 L1=240,241
1097.000        IF (L1.EQ.L)GOTO 25
1098.000    24  CONTINUE
1099.000        GOTO 604
1100.000    25  REG=L1-240
1101.000        GOTO 18
1102.000   17   REG=L1-241
1103.000   18   CODE=8
1104.000        RETURN
1105.000   613  LB2=LB1
1106.000        IF(L.EQ.217.AND.INST.EQ.60)GOTO 34
1107.000        IF(L.EQ.217)GOTO 9
1108.000        IF (L.EQ.124)GOTO 601
1109.000        IF (L.EQ.195)CODE=5
1110.000        IF(L.EQ.212)DAT=1
1111.000        IF(L.EQ.212)GOTO 34
1112.000        IF (L.EQ.201)CODE=6
1113.000        IF (L.EQ.227)CODE=7
1114.000        IF(L.EQ.198)GOTO 614
1115.000        IF (CODE.EQ.0)GOTO 604
1116.000        RETURN
```

```
1118.000       IF(INST.NE.21.OR.L.EQ.193)GOTO 651
1119.000       IF(DAT.EQ.48)GOTO 604
1120.000       LB1=LB2
1121.000       LF1=LF2
1122.000       LB2=TB
1123.000       LF2=TF
1124.000       DAT=48
1125.000       GOTO 650
1126.000   651 IF(L.NE.193.OR.LB2.NE.LF2)GOTO 604
1127.000       CALL S1(I,LB1,L,20)
1128.000       IF(L.NE.215)GOTO 604
1129.000       CALL S1(I,LB1+1,L,21)
1130.000       DO 19 L1=244,247
1131.000       IF(L1.EQ.L)GOTO 21
1132.000    19 CONTINUE
1133.000       GOTO 604
1134.000    21 REG=(L1-DAT)-244
1135.000       CODE=1
1136.000       RETURN
1137.000       END
1138.000       SUBROUTINE SET(CTE)
1139.000       DIMENSION CTE(10)
1140.000       DO 2 J=1,10
1141.000       CTE(J)=1000
1142.000     2 CONTINUE
1143.000       RETURN
1144.000       END
1145.000       SUBROUTINE GETOP(K1,K2,K3,K,IPC,K8,IST,IDM,BS)
1146.000       DIMENSION IDM(1088)
1147.000       CALL REWR(IPC,1,K4,0)
1148.000       K8=0
1149.000       K=1029
1150.000       IF (K4.GE.K1.AND.K4.LE.(K1+7))K=IDM(1+K4-K1+(24*BS))
1151.000       IF (K4.NE.K2.AND.K4.NE.(K2+1))GOTO 800
1152.000       K=IDM(1+K4-K2+(24*BS))
1153.000       K=K-(K/64)*64
1154.000       K=IDM(1+K)
1155.000   800 IST=IST+1
1156.000       IPC=IPC+1
1157.000       IF (K4.EQ.K3)GOTO 3
1158.000       IF(K.EQ.1029)K8=1
1159.000       RETURN
1160.000     3 IST=IST+1
1161.000       CALL REWR(IPC,1,K,0)
1162.000       IPC=IPC+1
1163.000       RETURN
1164.000       END
1165.000       SUBROUTINE INCOP(K1,K2,K3,K,IPC,IST,K8,ACC,IDM,BS)
1166.000       DIMENSION IDM(1088)
1167.000       INTEGER ACC
1168.000       K8=1
```

```
1169.000        IPC=IPC+1
1170.000        IST=IST+1
1171.000        CALL REWR (IPC-1,1,K4,0)
1172.000        IF (K4.EQ.K1)ACC=ACC+K
1173.000        IF (K4.GE.K2.AND.K4.LE.(K2+7))IDM(1+K4-K2+(24*BS))=K+IDM
1174.000       1(1+K4-K2+(24*BS))
1175.000        IF(IDM(1+K4-K2+(24*BS)).EQ.0)     K8=0
1176.000        CALL CAR(K9,IDM(1+K4-K2+(24*BS)),256)
1177.000        IF(K4.NE.K3.AND.K4.NE.(K3+1))  RETURN
1178.000        K9=IDM(1+K4-K3+(24*BS))
1179.000        K9=K9-(K9/64)*64
1180.000        IDM(1+K9)=K+IDM(1+K9)
1181.000        CALL CAR(K9,IDM(1+K9),256)
1182.000        RETURN
1183.000        END
1184.000        SUBROUTINE ICY(ICY1,K,I)
1185.000        K1=K-(K/16)*16
1186.000        K2=I-(I/16)*16
1187.000        ICY1=0
1188.000        IF((K1+K2).GE.16)ICY1=1
1189.000        RETURN
1190.000        END
--EOF HIT AFTER 1190.
*
```

CRDV R-4162/80 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Assembleur et simulateur pour les micro-ordinateurs 8048/8748/8035"
par R. Carbonneau, B. Montminy et P. Côté

Nous présentons un programme FORTRAN servant à la traduction des mnémoniques en code machine et à la simulation d'un programme écrit pour les micro-ordinateurs de type 8048/8748/8035. Ce programme, qui permet l'utilisation du langage symbolique d'Intel et l'emploi d'étiquettes pour les instructions "sauts", simule exactement le comportement du micro-ordinateur dans les applications réelles. Il permet en outre la simulation d'interruptions et l'impression des résultats intermédiaires. (NC)

CRDV R-4162/80 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Assembleur et simulateur pour les micro-ordinateurs 8048/8748/8035"
par R. Carbonneau, B. Montminy et P. Côté

Nous présentons un programme FORTRAN servant à la traduction des mnémoniques en code machine et à la simulation d'un programme écrit pour les micro-ordinateurs de type 8048/8748/8035. Ce programme, qui permet l'utilisation du langage symbolique d'Intel et l'emploi d'étiquettes pour les instructions "sauts", simule exactement le comportement du micro-ordinateur dans les applications réelles. Il permet en outre la simulation d'interruptions et l'impression des résultats intermédiaires. (NC)

CRDV R-4162/80 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Assembleur et simulateur pour les micro-ordinateurs 8048/8748/8035"
par R. Carbonneau, B. Montminy et P. Côté

Nous présentons un programme FORTRAN servant à la traduction des mnémoniques en code machine et à la simulation d'un programme écrit pour les micro-ordinateurs de type 8048/8748/8035. Ce programme, qui permet l'utilisation du langage symbolique d'Intel et l'emploi d'étiquettes pour les instructions "sauts", simule exactement le comportement du micro-ordinateur dans les applications réelles. Il permet en outre la simulation d'interruptions et l'impression des résultats intermédiaires. (NC)

CRDV R-4162/80 (NON CLASSIFIE)

Bureau - Recherche et Développement, MDN, Canada.
CRDV, C.P. 880, Courcelette, Qué. G0A 1R0

"Assembleur et simulateur pour les micro-ordinateurs 8048/8748/8035"
par R. Carbonneau, B. Montminy et P. Côté

Nous présentons un programme FORTRAN servant à la traduction des mnémoniques en code machine et à la simulation d'un programme écrit pour les micro-ordinateurs de type 8048/8748/8035. Ce programme, qui permet l'utilisation du langage symbolique d'Intel et l'emploi d'étiquettes pour les instructions "sauts", simule exactement le comportement du micro-ordinateur dans les applications réelles. Il permet en outre la simulation d'interruptions et l'impression des résultats intermédiaires. (NC)

DREV R-4162/80 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An assembler and simulator for the 8048/8748/8035 Intel microcomputers"
by R. Carbonneau, B. Montminy and P. Côté

A FORTRAN program used to translate man-readable statements into machine-understandable code and to simulate a program written for the Intel 8048/8748/8035 microcomputers is described. This program allows programming of the microprocessor in symbolic language and the use of labels for jump instructions. The simulator duplicates exactly the behavior of the microcomputer in a real-world application. It is also possible to simulate interrupts and print out intermediate results. (U)

DREV R-4162/80 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An assembler and simulator for the 8048/8748/8035 Intel microcomputers"
by R. Carbonneau, B. Montminy and P. Côté

A FORTRAN program used to translate man-readable statements into machine-understandable code and to simulate a program written for the Intel 8048/8748/8035 microcomputers is described. This program allows programming of the microprocessor in symbolic language and the use of labels for jump instructions. The simulator duplicates exactly the behavior of the microcomputer in a real-world application. It is also possible to simulate interrupts and print out intermediate results. (U)

DREV R-4162/80 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An assembler and simulator for the 8048/8748/8035 Intel microcomputers"
by R. Carbonneau, B. Montminy and P. Côté

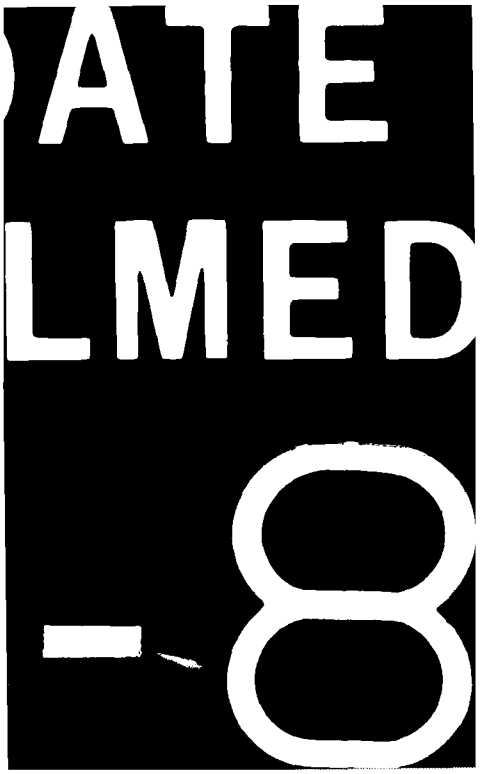
A FORTRAN program used to translate man-readable statements into machine-understandable code and to simulate a program written for the Intel 8048/8748/8035 microcomputers is described. This program allows programming of the microprocessor in symbolic language and the use of labels for jump instructions. The simulator duplicates exactly the behavior of the microcomputer in a real-world application. It is also possible to simulate interrupts and print out intermediate results. (U)

DREV R-4162/80 (UNCLASSIFIED)

Research and Development Branch, DND, Canada.
DREV, P.O. Box 880, Courcelette, Que. G0A 1R0

"An assembler and simulator for the 8048/8748/8035 Intel microcomputers"
by R. Carbonneau, B. Montminy and P. Côté

A FORTRAN program used to translate man-readable statements into machine-understandable code and to simulate a program written for the Intel 8048/8748/8035 microcomputers is described. This program allows programming of the microprocessor in symbolic language and the use of labels for jump instructions. The simulator duplicates exactly the behavior of the microcomputer in a real-world application. It is also possible to simulate interrupts and print out intermediate results. (U)


ATE
LMED
8