interface
TECHNOLOGY

Document No. 32

MODEL 488

PROGRAMMABLE IEEE BUS MONITOR/ANALYZER

INSTRUCTION MANUAL

June, 1979 (First Issue)
June, 1980 (Rev I)

150 E. Arrow Highway - San Dimas, California 91773 . Telephone 714/599-0848

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

APPENDICIES

LIST OF ILLUSTRATIONS

LIST OF TABLES

PROPRIETARY NOTICE

This document and the technical data·contained herein are proprietary to Interface Technology and shall not, without the written permission of Interface Technology, be used in any form or part to solicit competitive quotations from within customer complex or other competitive sources. The information provided herein may be used for operation and maintenance purposes or for purposes of incorporation into technical specifications and other documents which specify procurement from Interface Technology.

WARRANTY

Interface Technology warrants each instrument manufactured to be free of defects in material and workmanship for a period of one year from the date of shipment to the original purchaser. Interface Technology will service, replace, or adjust any defective part or parts, free of charge, when the instrument is returned to Interface Technology freight prepaid, and when examination reveals that the fault has not occurred because of misuse or abnormal conditions of operation. Instruments repaired beyond the effective date of warranty or when abnormal useage has occurred will be charged at applicable rates. Interface Technology will submit an estimate for such charges commencing repair if so requested. For any questions concerning this warranty or shipping call Interface Technology or our sales representative in your area.

PERSONNEL SAFETY

The equipment described in this manual contains voltages hazardous to human life and safety and which is capable of inflicting personal injury.

For all procedures involving component insertion or withdrawal, the equipment must be powered off to prevent component damage. It is also recommended for such procedures that the primary power cord be disconnected from the equipment's rear panel connector to prevent accidental contact with primary power circuits.

While physical measures have been built into the equipment to prevent accidental contact with high voltages during maintenance and troubleshooting procedures, the user should still exercise caution. Careless probing inside the equipment may result in the exposure of high voltage terminals.

Before operating the unit ensure that the primary power outlet includes a functional protective ground (earth) circuit. Do not defeat the unit's internal protective ground circuit to chassis by using a two conductor adapter plug or other such device.

I INTRODUCTION

1.1 General Description

The Model 488 Programmable Bus Monitor/Analyzer enables the evaluation, control, and testing of a user's IEEE Std 488-1978 bus system. The Model 488 provides the following key features:

. IEEE Std 488-1978 bus compatible.

. Integral bus monitor mode (sec. 3.2).

- selectable trigger; records up to 511 bus transactions

. Programmable bus controller/analyzer modes (sec. 3.3, 3.4).

- simplified bus mnemonic or machine language instructions

. Plug-in EPROM card for user-defined programs (sec. 3.7).

. 511 data bytes for transmission, recording or comparison (sec. 3.5).

. Simplified front panel operation (sec. 3.1).

. Remote control interfaces (sec. IV).

Figure 1-1 provides a front view of the Model 488 and illustrates the front panel operating controls. In order to simplify front panel operation, the user is automatically prompted through the necessary operating sequences by the instrument's input/output microprocessor.

Figure 1-2 illustrates the unit's rear panel features including the TEST INTERFACE connector. The optional connectors for the remote control interfaces are grouped and designated as CONTROL INTERFACES.

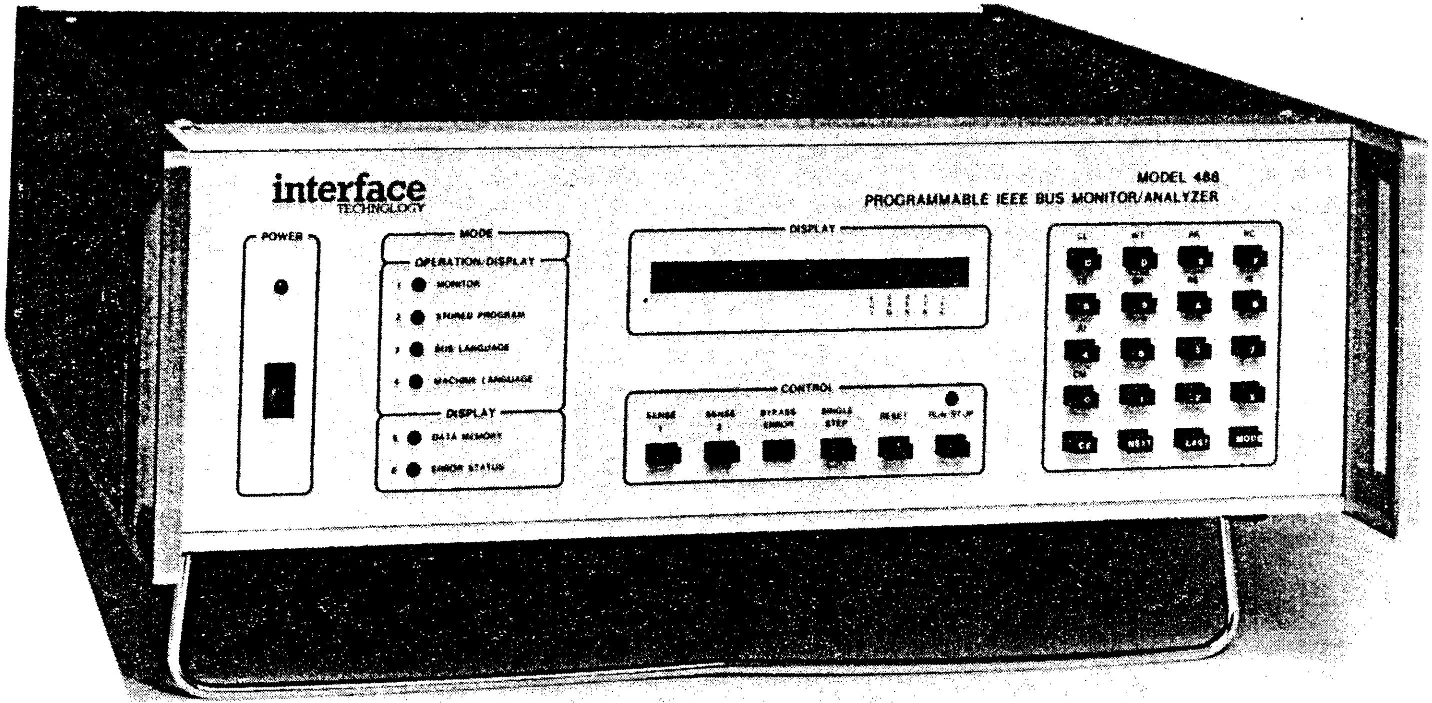
Operation of the Model 488 assumes the user is basically familiar with the principles and concepts of the IEEE Std 488-1978. Copies are available from IEEE, 345 East 47th Street, New York, New York 10017.

1.2 Options

1.2.1 Stored Program Card (option 488-305)

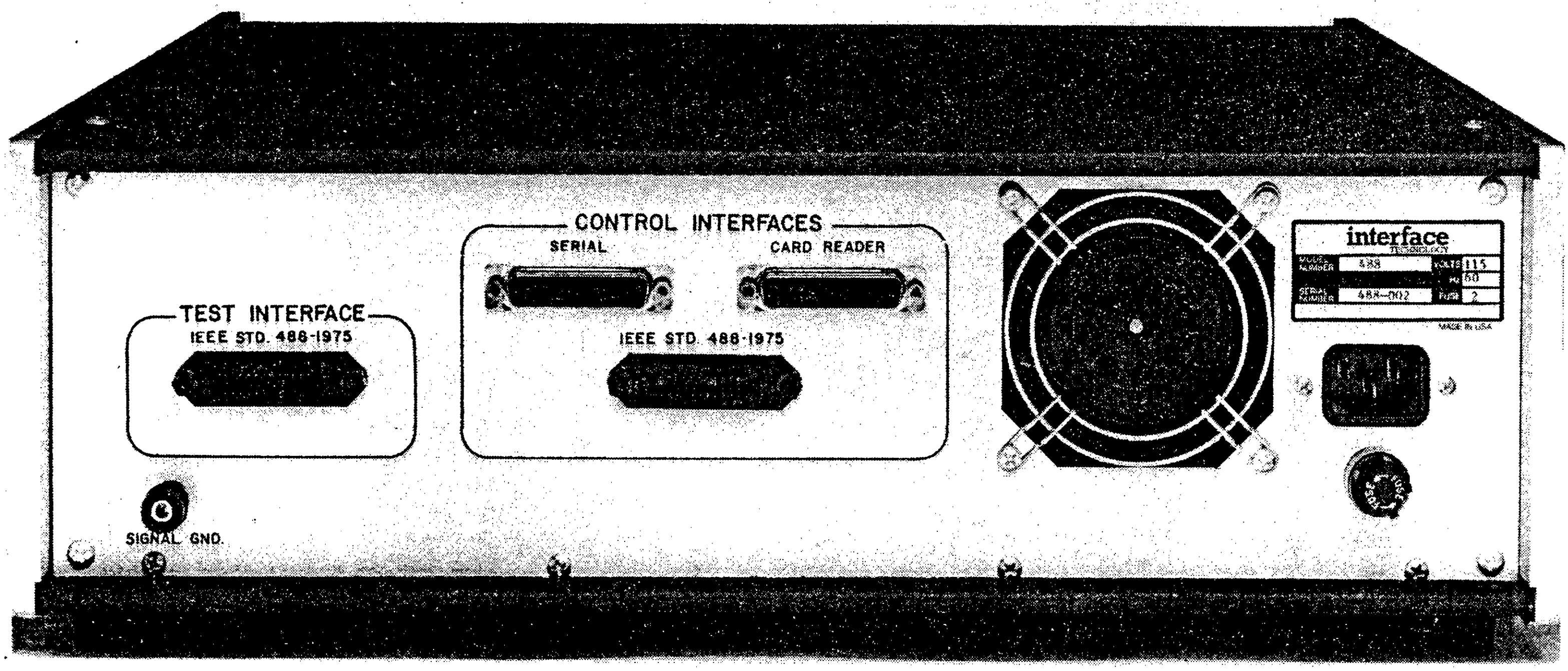
The optional plug-in stored program card provides for storage of user-written programs such as those necessary for calibration or production testing. The card is capable of holding eight erasable programmable read only memory (EPROM) ICs which must be programmed external to the Model 488. The card is supplied less EPROMs. The following 5 volt EPROM types may be used:

i) Intel 2758	1K x 8
ii) Intel 2716	2K x 8
iii) Intel 2732	4K x 8
iv) Texas Instr. 2516	2K x 8



MODEL 488 PROGRAMMABLE IEEE BUS MONITOR/ANALYZER

FIGURE 1-1



MODEL 488 REAR PANEL

FIGURE 1-2

With Intel 2732s installed, the card provides 32K x 8 of storage. Reference section 3.7 for a description of the stored program mode.

1.2.2 IEEE Std 488-1978 Control Interface (option 488-304)

The optional IEEE Std 488-1978 control interface is a plug-in card which provides for remote control of the Model 488. The interface is thus capable of loading and reading the memories, starting and stopping the programs, and reading the status of the unit. Reference section 4.2.

1.2.3 RS-232C/TTY Control Interface (option 488-303)

The RS-232C/TTY control interface is also an optional plug-in card which provides for Model 488 remote control. It is capable of loading and reading the memories, starting and stopping program execution, and reading the resulting status. Reference section 4.3.

1.2.4 Card Reader (options 488-301, 302)

Option 488-301 includes both a mark-sense card reader and the required interface to the Model 488. The card reader enables the user to record frequently used programs on cards thus providing a hard copy medium which may be used to quickly load programs and data into the Model 488 memories.

Option 488-302 includes the Model 488 card reader interface only and may be used as a parallel eight bit interface for loading the instrument. Reference section 4.1.

1.2.5 Miscellaneous Options

The following is a list of additional options to the Model 488:

i)	IEEE bus cable, 1 meter	(option 400-306)
ii)	IEEE bus cable, 2 meters	(option 400-307)
iii)	IEEE bus cable, 4 meters	(option 400-308)
iv)	rack mount adapter, 19 in.	(option 400-013)
v)	additional documentation	(option 488-300)

1.3 Specifications

Test Interface:

Compliant with IEEE Std 488-1978, electrically and mechanically

Open-collector drivers

Control and sense of all bus lines

Bus Processor:

200 nsec instruction cycle

68 instruction types including standard subroutines, bus control/sense, and general purpose programming

3 levels of subroutines

255 instruction RAM locations

256 instruction ROM locations (i.e., standard subroutines)

511 data RAM locations

Standard ROM Subroutines:

Accommodate all IEEE 488-1978 interface messages except TCT.

3.2 usecs minimum for interface message handshake sequence.

4.0 usecs minimum for data transmission handshake sequence.

3.4 usecs minimum for data recording handshake sequence.

2.8 usecs minimum for data comparison handshake sequence.

User-developed subroutines may be used instead of the standard subroutines to obtain variations in speed and performance.

Power:

115 Vac, 50/60 Hz, 110 W

100/200/220/240 Vac, 50/60 Hz available

Physical:

134 mm (5.25 in.) high

432 mm (17 in.) wide

430 mm (16.8 in.) deep including 46 mm (1.8 in.) for handles

Weight:

11 kg (24.2 lbs)

II INSTALLATION/MAINTENANCE

2.1 Installation

2.1.1 Unpacking and Inspection

Prior to unpacking, examine the exterior of the shipping carton for any signs of damage. The Model 488 is packed in a molded plastic-foam form within a cardboard carton. The molded form holds the unit securely in the carton and absorbs any reasonable external shock normally encountered in transit. Carefully remove the unit from the carton and inspect the exterior of the instrument for any signs of damage. If damage is discovered, file a claim with the carrier who transported the unit.

The shipping container and the packing material should be saved in case reshipment is required.

Included in the shipping container with the instrument are the instruction manual, power cord, and external option accessories.

2.1.2 Mechanical Installation

The Model 488 is equipped with a collapsible tilt bail which may be used to elevate the front of the instrument for convenient bench use. Pull the bail down and forward to use.

The instrument may be mounted in a standard 19-inch rack with the optional rack-mount kit (option 400-013). To install, remove the small decorative side panels located near the front of the instrument. Attach each of the rack mount flanges using the two screws supplied.

2.1.3 Electrical Installation

Prior to connecting power to the Model 488, confirm that the primary power source is compatible with the source requirements listed on the rear panel serial number plate. Unless otherwise specified at the time of purchase, the Model 488 is wired to use 115 Vac, 50/60 Hz primary power and includes the required 2 amp fuse. The unit's power transformer may be factory wired to use the following primary power sources:

a) 100 Vac, 50/60 Hz, 2 amp fuse

b) 200, 220, or 240 Vac, 50/60 Hz, 1 amp fuse

Confirm that the unit's front panel POWER switch is off; i.e., bottom-half of switch depressed.

An ac line cord having a three-pin plug is supplied with the instrument. When the cord is plugged into the Model 488's rear panel receptacle, the round pin of the plug is connected to the instrument's case. This pin must be connected to a good quality earth ground when the plug is installed in the user's primary power receptacle.

Connect the ac line cord to the Model 488's rear panel receptacle and to the primary power source.

2.2 Functional Verification

Table 2-1 is a detailed step-by-step procedure to verify the Model 488 operation. This procedure is primarily intended to be used by calibration or metrology lab personnel at the time of initial receipt of the unit and at the periodic maintenance interval. While a first time user could also use this procedure to gain familiarity with the unit, it is not necessary to do so. The user may proceed directly to the operation description of the mode he is interested in (reference Section III).

It is not to be implied that the procedure of Table 2-1 is a 100% test of the Model 488 functionality. Rather, it verifies the basic performance of the unit and will detect 80-90% of potential problems.

The test essentially performs an "end-around" operation using the unit's IEEE bus transceivers. Instructions and data are manually entered into the unit's memory. The unit is then directed to execute the entered program. The stored data are transmitted and, using the transceivers, received and recorded. The recorded data are manually verified upon completion.

Prior to running the test, ensure that no cable is connected to the Model 488's rear panel connector designated TEST INTERFACE.

If an error occurs during entry of the data of Table 2-1, a blinking question mark will be displayed in the right-most position of the alphanumeric display. Depress the CE (clear entry) key to clear the erroneous data and restore the original contents.

If the expected display response is not obtained at any step of the procedure, proceed as follows. Depress the POWER switch to turn off the unit's power. Remove the ac line cord from the primary power source. Remove the top cover of the Model 488. Examine the interior of the unit for loose cables, cards, etc. and make the necessary corrections. Repeat the test of Table 2-1. If problems persist, reference the warranty notice herein following the index.

If it is desired to repeat the test, note that it is necessary to reload the data memory each time (steps 15-53). Execution of the program causes the contents of data memory to be overwritten. Following data memory reloading, reselect the machine language mode and repeat steps 160-182.

FUNCTIONAL VERIFICATION PROCEDURE
TABLE 2-1

Step	Entry	Display Response	Comments
1	POWER	MODE? MONITOR	LEDS = POWER,MONITOR.
2	NEXT	TRIG? 0-4,D	
3	MODE	MODE? MONITOR	
4	2	MODE? STD PGMS	
5	NEXT	PGM?	LEDS = STD PGM. POWER omitted in following.
6	MODE	MODE? STD PGMS	
7	3	MODE? BUS LANG	
8	NEXT	LINE NUM? 00	LEDS = BUS LANG.
9	MODE	MODE? BUS LANG	
10	4	MODE? MACH LAN	
11	NEXT	ADDR?'000	LEDS = MACH LANG.
12	MODE	MODE? MACH LAN	
13	6	MODE? ERR STAT	
14	NEXT	NO ERROR	LEDS = MACH LANG, ERR STAT.
15	MODE	MODE? ERR STAT	
16	5	MODE? DATA MEM	
17	NEXT	ADDR?'000 ASCII	LEDS = MACH LANG, DATA MEM.
18	NEXT	000 DAB NL	
19	NEXT	001 DAB NL	
20	3	001 DAB'3	
21	0	001 DAB 0	
22	NEXT	002 DAB NL	
23	3	002 DAB'3	
24	9	002 DAB 9	
25	NEXT	003 DAB NL	
26	32	003 DAB 2	Intermediate step display response omitted.
27	NEXT	004 DAB NL	
28	33	004 DAB 3	
29	NEXT	005 DAB NL	
30	34	005 DAB 4	
31	NEXT	006 DAB NL	
32	0D	006 DAB CR	
33	NEXT	007 DAB NL	
34	0A	007 DAB LF	
35	NEXT	008 DAB NL	
36	FF	008 DAB'FF	
37	NEXT	009 DAB NL	
38	14	009 DAB D4	
39	NEXT	00A DAB NL	
40	5F	00A DAB _	
41	NEXT	00B DAB NL	
42	3F	00B DAB ?	
43	NEXT	00C DAB NL	
44	55	00C DAB U	
46	NEXT	00D DAB NL	

FUNCTIONAL VERIFICATION PROCEDURE
TABLE 2-1

Step	Entry	Display Response	Comments
47	3E	00D DAB ⦣	
48	NEXT	00E DAB NL	
49	52	00E DAB R	
50	NEXT	00F DAB NL	
51	5F	00F DAB _	
52	NEXT	010 DAB NL	
53	3F	010 DAB ?	
54	LAST	00F DAB _	
55	LAST	00E DAB R	
56	MODE	MODE? DATA MEM	
57	NEXT	ADDR?'00E ASCII	
58	F	ADDR?'00E HEX	
59	NEXT	00E DAB'52	
60	LAST	00D DAB'3E	
61	RESET	ADDR?'000 ASCII	
62	MODE	MODE? DATA MEM	
63	4	MODE? MACH LAN	
64	NEXT	ADDR?'000	LEDS = MACH LANG.
65	NEXT	000	
66	E	000 E	
67	B	000 FCL	
68	NEXT	001	
69	E	001 E	
70	8	001 FOT	
71	1	001 FOT 1	
72	NEXT	002	
73	E11	002 FRE 1	Display response to intermediate steps omitte
74	NEXT	003	
75	E51	003 FDV 1	
76	NEXT	004	
77	E01	004 FAT 1	
78	NEXT	005	
79	385E	005 LDI 5E	
80	NEXT	006	
81	4007	006 JUN 007	
82	NEXT	007	
83	2C00	007 STD 000	
84	NEXT	008	
85	3407	008 LC2 007	
86	NEXT	009	
87	E00	009 FAT 0	
88	NEXT	00A	
89	E31	00A FEI 1	
90	NEXT	00B	
91	2001	00B LDD 001	
92	NEXT	00C	
93	400D	00C JUN 00D	

FUNCTIONAL VERIFICATION PROCEDURE
TABLE 2-1

Step	Entry	Display Response	Comments
94	NEXT	00D	
95	28	00D STR	
96	NEXT	00E	
97	6211	00E JL2 T 011	
98	NEXT	00F	
99	24	00F LDN	
100	NEXT	010	
101	400D	010 JUN 00D	
102	NEXT	011	
103	24	011 LDN	
104	NEXT	012	
105	E30	012 FEI 0	
106	NEXT	013	
107	E50	013 FDV 0	
108	NEXT	014	
109	E01	014 FAT 1	
110	NEXT	015	
111	E21	015 FIF 1	
112	NEXT	016	
113	4017	016 JUN 017	
114	NEXT	017	
115	28	017 STR	
116	NEXT	018	
117	E20	018 FIF 0	
118	NEXT	019	
119	E51	019 FDV 1	
120	NEXT	01A	
121	3405	01A LC2 005	
122	NEXT	01B	
123	24	01B LDN	
124	NEXT	01C	
125	401D	01C JUN 01D	
126	NEXT	01D	
127	28	01D STR	
128	NEXT	01E	
129	601B	01E JL2 F 01B	
130	NEXT	01F	
131	E00	01F FAT 0	
132	NEXT	020	
133	24	020 LDN	
134	NEXT	021	
135	4022	021 JUN 022	
136	NEXT	022	
137	28	022 STR	
138	NEXT	023	
139	E01	023 FAT 1	
140	NEXT	024	

FUNCTIONAL VERIFICATION PROCEDURE
TABLE 2-1

Step	Entry	Display Response	Comments
141	3402	024 LC2 002	
142	NEXT	025	
143	24	025 LDN	
144	NEXT	026	
145	4027	026 JUN 027	
146	NEXT	027	
147	28	027 STR	
148	NEXT	028	
149	6025	028 JL2 F 025	
150	NEXT	029	
151	E94	029 FPI 4	
152	NEXT	02A	
153	402A	02A JUN 02A	
154	LAST	029 FPI 4	
155	LAST	028 JL2 F 025	
156	RESET	ADDR?'000	
157	F	ADDR?'000 HEX	
158	NEXT	000 EB00	
159	NEXT	001 E801	
160	RESET	ADDR?'000	
161	RUN	DONE	
162	MODE	MODE? MACH LAN	
163	5	MODE? DATA MEM	
164	NEXT	ADDR?'000 ASCII	
165	NEXT	000 MTA ^ 00110	
166	NEXT	001 DAB 0 00100	
167	NEXT	002 DAB 9 00100	
168	NEXT	003 DAB 2 00100	
169	NEXT	004 DAB 3 00100	
170	NEXT	005 DAB 4 00100	
171	NEXT	006 DAB CR 00100	
172	NEXT	007 DAB LF 10100	
173	NEXT	008 IFC 00111	
174	NEXT	009 DCL 00110	
175	NEXT	00A UNT 00110	
176	NEXT	00B UNL 00110	
177	NEXT	00C MTA U 00110	
178	NEXT	00D MLA ⋝ 00110	
179	NEXT	00E DAB R 00100	
180	NEXT	00F UNT 00110	
181	NEXT	010 UNL 00110	
182	NEXT	011 DAB NL	Test Complete

2.3 Calibration

The Model 488 is a digital instrument and requires limited calibration in the conventional sense. It is recommended that the power supply voltage be checked/adjusted each six months as follows:

a) With power applied to the unit, remove the top cover.

> CAUTION
> Hazardous voltages are contained within the instrument. While the high voltage terminals have been protected to prevent accidental contact, careless probing may result in their exposure.

b) Connect a DVM across the electrolytic capacitor located near the top edge of the rear side of the front panel. Observe polarity markings on the capacitor.

c) The voltage should be +5.00 +/- 0.05 Vdc.

d) To adjust the voltage, locate the adjustment potentiometer on the power supply. The pot is towards the rear of the power supply between the positive and negative power distribution terminals. Carefully insert an insulated shaft screwdriver and rotate slowly until the DVM indicates the preferred voltage.

It is also recommended that the Functional Verification Procedure of Table 2-1 be performed at the six month calibration interval.

2.4 Maintenance

In order to perform fault correction maintenance on the Model 488, personnel must familiarize themselves with the following:

a) machine language mode operation, Section 3.4
b) data memory operation, Section 3.5
c) theory of operation, Section V
d) logic diagrams, Appendix M

The following guidelines are provided to aid in maintenance troubleshooting:

a) Isolate the problem to the input/output processor (IOP) or to the bus processor (BP). Mode initialization and setup are concerned only with the IOP (reference section 3.1.3).

b) Determine if the IOP is able to communicate with the BP memory successfully. Each time a machine language or data memory location is accessed, the IOP fetches the current contents and enters the new contents when sequenced.

c) Determine if the machine language instructions with op codes '20 to 'EB function as required using the single instruction modes of section 3.4.10. The test of Table 2-1 provides for testing of many of the instructions.

If problems persist or questions arise, reference the warranty notice following the index and the list of representatives at the rear of this manual.

III LOCAL CONTROL OPERATION

This section describes operation of the Model 488 while under local control using the front panel controls and indicators. The following topics are discussed:

Subsection	Topic
3.1	Controls and Indicators Description
3.2	Monitor Mode Operation
3.3	Bus Language Mode Operation
3.4	Machine Language Mode Operation
3.5	Data Memory Mode Operation
3.6	Error Status Mode Operation
3.7	Stored Program Mode Operation

Remote control of the Model 488 takes precedence over local control. Initiation of communication between the Model 488 and a remote controlling device such as a calculator causes the front panel to display REMOTE CNTRL. Except for the local lockout condition, the user may regain local control by depressing the RESET key. It should be noted that the Model 488 may immediately revert back to remote control if remote communication continues.

3.1 Controls and Indicators Description

Figure 3-1 illustrates the front panel controls and indicators of the Model 488. Use of the front panel controls and indicators is, of course, dependent on the selected mode of operation which, in turn, requires the user to be familiar with each mode's operating characteristics. However, there are certain general control functions common to the operating modes which are discussed below.

3.1.1 POWER Switch and LED

Depression of the POWER switch in order to apply power to the Model 488 causes the unit to execute its initialization sequence resulting in the following:

(a) POWER LED illuminates

(b) display indicates MODE? MONITOR

(c) MONITOR mode LED illuminates

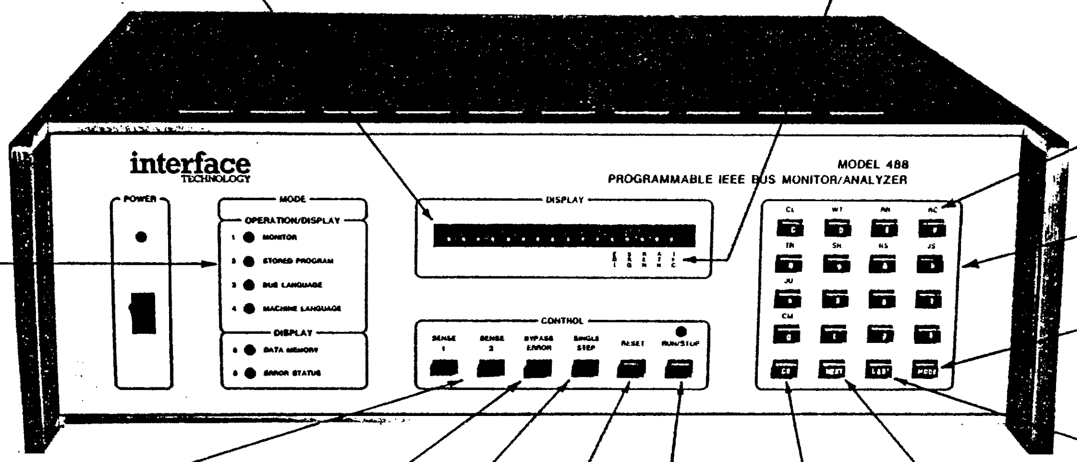
(d) bus language program memory is cleared

(e) machine language program memory is cleared

(f) data memory is cleared

(g) all bus lines are set to the passive false state.

FIGURE 3-1
MODEL 488
SWITCHES AND INDICATORS



16 Digit Alphanumeric Display

Indicate status of
bus management lines
when data recorded
in Data Memory.

Bus Language
Op Code Mnemonics

Hex Keyboard

MODE LEDs
Indicate currently
active mode(s).

MODE
Causes MODE? (current mode)
to be displayed. Enables
subsequent selection of a
new mode.

LAST
Decrements Bus Language
line number, Machine Language
program memory address, or
Data Memory address in order
to observe preceding address's
memory contents.

SENSE SWITCHES
Enable manual alteration
of program sequencing. SS1
also used to select BUS
LANGUAGE single step or
burst data transfer modes.

CE
CLEAR ENTRY
Clears keyboard entry
and restores to original
value. Erroneous entry
indicated by blinking ?.

RESET
Re-initializes
current mode.
Stops unit.
Clears erroneous
keyboard entry.

NEXT
Sequences to next prompt
display during mode
selection and setup.
Increments Bus Language
line number, Machine
Language program memory
address, or Data Memory
address in order to observe
succeeding address's
memory contents.

BYPASS ERROR
Directs the unit
to disregard all
detected errors.

SINGLE STEP
Steps past current
error when running.
Executes single
instruction of Bus
Language or Machine
Language when not
running.

RUN/STOP
LED indicates RUN
state when on and
program state when
off. Switch depression
changes state of LED.

It is not recommended that the Model 488 (or any instrument) be powered up or down if it is connected to an active bus. Indeterminate and invalid bus messages may result.

3.1.2 Entry Errors, CE Key

Invalid front panel entries made by the user are indicated on the display by a flashing question mark symbol in the right-most display position. Depression of the CE (clear entry) key restores the display to the original contents which existed prior to the user's entry.

Not all invalid user entries result in an error indication. During various phases of a mode's operation, depression of certain keys are simply ignored. For example, hexadecimal data entries are ignored while the unit is in the running state.

3.1.3 Mode Selection and Setup

Mode selection is accomplished in accordance with the following procedure:

Key Entry	Display Response	Comment
MODE	MODE? (current mode)	
numeric 1-6	MODE? (selected mode)	1=MONITOR 2=STD PGMS 3=BUS LANG 4=MACH LAN 5=DATA MEM 6=ERR STAT
NEXT	mode's initial setup state	Mode LEDs reflect change to new mode. MONITOR= TRIG? Ø-4/D STD PGMS = PGM? BUS LANG = LINE NO? ØØ MACH LAN = ADDR?'ØØØ DATA MEM = ADDR?'ØØØ ASCII ERR STAT = NO ERROR

The Model 488 includes the six modes noted in the previous paragraph. Monitor, stored program, bus language and machine language (modes 1-4) are referred to as operating/display modes. Data memory and error status (modes 5,6) are referred to as display only modes. The Model 488 will always be in one (only) of the operating/display modes as indicated by the mode LEDs. However, the information provided by the alphanumeric displays is dependent on the display mode. If neither of the display only modes (5,6) has been selected, the alphanumeric displays correspond to the current operating/display mode. If either of the display modes has been selected, the alphanumeric displays provide information relative to the selected display only mode.

During mode selection and setup, the NEXT key is used to advance to the next prompting display. If the user had entered data which were valid and as desired, the NEXT key is depressed to both store the entry and to continue to the next prompt state. Likewise, if the user did not wish to change the existing display data, the NEXT key is depressed to simply continue to the next prompt state. The LAST key is an invalid input during mode selection and setup and is, instead, used only in memory accessing.

Monitor, stored program and bus language result in the generation of an equivalent set of machine language instructions in order to execute their respective mode programs. This assembly of machine language instructions occurs when either the unit is directed to run or the mode is changed to machine language.

3.1.4 RESET Key

Depression of the RESET key causes the following:

(a) returns the Model 488 to local control if previously in remote control.

(b) stops the Model 488 if previously running.

(c) clears all bus interface lines.

(d) clears the bus language line number to 00.

(e) clears the machine language program memory address to '000.

(f) clears the data memory address to '000.

(g) sets the display to the current mode's initial prompt state.

(i) MONITOR = TRIG? Ø-4/D

(ii) STD PGMS = PGM?

(iii) BUS LANG = LINE NO? ØØ

(iv) MACH LAN = ADDR?'ØØØ

(v) DATA MEM = ADDR?'ØØØ ASCII

(vi) ERR STAT = NO ERROR

3.1.5 RUN/STOP Key and LED

The RUN LED indicates the current operating state of the Model 488. When illuminated, the Model 488 is in the running state. When extinguished, the unit is in the programming state. Depression of the RUN/STOP key will cause the state of the RUN LED to change.

3.1.6 BYPASS ERROR

The BYPASS ERROR switch is intended to permit the test program to cycle while disregarding errors in order that further diagnostic analysis may be made with an oscilloscope. If the switch is depressed, the Model 488 will ignore all detected errors. If the switch is released, the unit halts on detected errors.

It should be noted that when the Model 488 disregards errors detected by the standard subroutines, the program will continue with the next sequential operation if either a DATA ERROR or STAT ERROR had occurred. However, if a HDWR ERROR had been detected, the program is reinitiated at address or line number zero. Hardware errors such as DAC @ ATN or NO LISTENER are essentially fatal and simple continuance would result in the generation of additional errors tending to mask the original problem. Refer to section 3.6 for more information on the errors detected by the Model 488's standard subroutines.

3.2 Monitor Mode Operation

The monitor mode enables the user to record and store up to 511 bus transactions in the Model 488's data memory following the detection of a user specified trigger condition. The first type of transaction recorded is the assertion of the interface clear (IFC) bus management line. The other transaction type consists of all multiline messages associated with handshake sequences. Handshake transmissions include both interface messages (commands) and device dependent data bytes (DAB). The multiline interface messages (commands) are encoded for display in terms of IEEE Std 488-1978 message mnemonics. Refer to Appendix F for a glossary of mnemonics. The data memory mode enables the user to select displaying multiline message arguments in terms of ASCII or hex characters. The states of the five bus management lines (EOI, SRQ, REN, ATN, IFC) are also recorded and stored along with each transaction.

The first transaction recorded (i.e., the trigger condition) may be selected by the user from the following:

Trigger Entry No.	Display	Description
-	TRIG? Ø-4,D	Indicates no trigger condition has been defined.
Ø	TRIG? IFC	Triggers data memory recording when the interface clear (IFC) line is first asserted.
1	TRIG? DAV	DAV, data available. Triggers recording on the first complete handshake sequence. Effectively records any and all transfers.

2	TRIG? DAB'dd c	DAB, data byte. Triggers recording on device dependent (i.e., ATN false) data byte. 'dd = two hex characters defining data, entered via keyboard ('00 ≤'dd ≤'FF). c = ASCII equivalent character of 'dd. Refer to Appendix H. If no equivalent ASCII character exists (i.e., MSB=1), hex characters are repeated on display.
3	TRIG? MLA'dd c	MLA, my listen address. Triggers when device with hex address 'dd is setup to be a listener (i.e., to receive data). 'dd = two hex characters ('00 ≤'dd ≤'1E) entered via keyboard. Refer to Appendix G. c = ASCII equivalent character of 'dd plus two msbs=01 associated with MLA.
4	TRIG? MTA'dd c	MTA, my talk address. Triggers when device with hex address 'dd is setup to be a talker (i.e., to transmit data). 'dd = two hex characters ('00 ≤'dd ≤'1E) entered via keyboard. Refer to Appendix G. c = ASCII equivalent character of 'dd plus two msbs=10 associated with MTA.
D	TRIG? MTA'1E ∧ D	Monitor demonstration mode. Character D is appended to the display to alert the user to the demonstration mode. User may alter the argument 'dd of MTA as in 4 above but it will have no effect upon the demonstration mode results.

The operation of the Model 488 monitor mode is best described with the step-by-step procedure and example of Table 3-1. This table provides a quick reference to the monitor mode operating sequence.

3.3 Bus Language Operation

The bus language mode enables the user to create programs using a simple bus-oriented instruction set which exercises and tests both the devices and their interfaces. The instructions perform the following functions:

a) clear all device interfaces and the devices themselves

b) transmit data to a device

c) receive and record data transmitted from a device

d) receive and compare data transmitted from a device

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
POWER	MODE? MONITOR	POWER LED and MONITOR mode LED illuminate. It is not recommended that power be applied to the Model 488 (or any bus device) which has previosuly been connected to an operating bus. Unpredictable operation may result.
NEXT	TRIG? 0-4,D	Requests one of following as a keyboard input to define the trigger byte:
		0 = IFC interface clear. Triggers DATA MEMORY recording when the first interface clear message is asserted.
		1 = DAV data available. Triggers on the first complete multiline message handshake sequence. Effectively records any and all handshake transfers.
		2 = DAB'dd c data byte. Triggers on device dependent data byte; i.e., attention, ATN, false. 'dd = two hex characters (00 ≤ 'dd ≤ 'FF) defining data, entered via keyboard. c = ASCII equivalent character of 'dd. Refer to Appendix H. If no ASCII equivalent exists (i.e., msb=1), hex characters are repeated on display.
		3 = MLA'dd c my listen address. Triggers when device with address 'dd is setup as a listener (i.e., to receive data). 'dd = two hex characters (00 ≤ 'dd ≤ '1E) entered via keyboard. Refer to Appendix G. c = ASCII equivalent character of 'dd plus two msbs=Ø1 associated with MLA.
		4 = MTA'dd c my talk address. Triggers when device with address 'dd is setup as a talker (i.e., to transmit data). 'dd = two hex characters (00 ≤ 'dd ≤ '1E) entered via keyboard. Refer to Appendix G. c = ASCII equivalent character of 'dd plus two msbs=1Ø associated with MTA.

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
		D = MTA'1E∧D ; MONITOR demonstration mode. Character D is appended to the display to alert the user to the demonstration mode. User may alter the argument 'dd of MTA as in 4 above but it will have no effect upon the demonstration mode results. The Model 488 must not be connected to the IEEE bus when running the demonstration program.
2	TRIG? DAB'00 NL	Selects triggering on a device dependent data byte equal to '00 (ASCII NUL).
0	TRIG? DAB'0	First trigger byte of 0 entered.
D	TRIG DAB'OD CR	Selects triggering on a device dependent data byte equal to 'OD (ASCII carriage return).
NEXT	XFRS? 511	Continues to the next prompting state. Requests the number(nnn,decimal) of transfers to be recorded in DATA MEMORY. 001 ≤ nnn ≤ 511.
5	XFRS? 5	
0	XFRS? 50	
0	XFRS 500	Specifies recording of 500 transfers in DATA MEMORY.
NEXT	RUN?	Continues to the next prompting state. Requests RUN direction.

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
RUN	RUNNING Oaa	Model 488 first clears its DATA MEMORY, assembles the appropriate MONITOR mode program in the MACHINE LANGUAGE program memory, and starts the processor. As long as either the trigger condition has not been detected or an insufficient number of transfers have not been recorded, the display will continue to indicate RUNNING Oaa. Oaa is an indication of the program steps being executed. If during program execution, the Model 488 detects that the NRFD (not ready for data) line has not been set true within one second of the assertion of the DAV (data available) line, the unit will halt and display either HARDWARE ERROR 028 or HARDWARE ERROR 040. 028 indicates the handshake occurred while IFC was true and 040 indicates the handshake occurred while IFC was false. User interrogation of the ERROR STATUS mode will result in a display of HNDSHK TIME-OUT. Refer to section 3.6 for further information on the ERROR STATUS mode. The error halt may be bypassed by depressing BYPASS ERROR. The remainder of the MONITOR mode operation description assumes the user wishes to abort the previous MONITOR program.
RESET	TRIG? DAB'OD CR	Stops the program and returns to the trigger prompt state.
CE	TRIG? 0-4,D	Requests new triggering condition.
D	TRIG? MTA'1E D	Selects MONITOR demonstration program.
NEXT	XFRS? 017 D	Advances to the next prompt state and requests the number of transfers. The demonstrator program simulates the recording of 17 transfers.

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
NEXT	RUN? D	Advances to the next prompt state and requests RUN direction.
RUN	RUNNING Oaa	RUNNING display persists momentarily and may not be visible.
-	DONE	Display indicates the program has completed the monitoring process.
MODE	MODE? MONITOR	Enables change of mode.
5	MODE? DATA MEM	Display indicates operator's selection. Note the mode is not changed until NEXT is entered.
NEXT	ADDR?'000 ASCII	Mode LEDs indicate both MONITOR and DATA MEMORY modes. MONITOR data is recorded in DATA MEMORY starting at hex address location '000.
NEXT	000 MTA ∧ 00110	First data recorded is specified trigger byte. Last five binary characters indicate status of the five bus management lines at the time of the recording.
NEXT	001 DAB Ø 00100	Data memory address advances to '001 and indicates the recorded transfer was a device dependent (ATN low) data byte equal to ASCII Ø (hex '3Ø) with only the REN bus management line true.
NEXT	002 DAB 9 00100	Data memory address advances to '002.
NEXT	003 DAB 2 00100	
NEXT	004 DAB 3 00100	
NEXT	005 DAB 4 00100	
LAST	004 DAB 3 00100	Data memory address decrements to '004.

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
NEXT	005 DAB 4 00100	
NEXT	006 DAB CR 00100	Data memory advances to '006 and indicates recorded transfer was an ASCII carriage return.
NEXT	007 DAB LF 10100	Data memory advances to '007 and indicates recorded transfer was an ASCII line feed. Also note that the EOI bus management line was asserted indicating the END message was true (END = EOI $\wedge \overline{ATN}$).
NEXT	008 IFC 00111	Indicates that the interface clear message was asserted. The binary 1 indicator of the IFC bus management line is redundant to the recorded message in this case, although it is possible for handshake transfers to occur while IFC is asserted. Note that ATN is asserted.
NEXT	009 DCL 00110	Indicates recording of the device clear interface message (ATN asserted).
NEXT	00A UNT 00110	Indicates recording of untalk interface message.
NEXT	00B UNL 00110	Indicates recording of unlisten interface message.
NEXT	00C MTA U 00110	Records message which setup device with address '15 as a talker in order to send data.
NEXT	00D MLA > 00110	Records message which setup device with address '1E as a listener in order to receive device dependent data.
NEXT	00E DAB R 00100	Records device dependent data byte equal to ASCII R.
NEXT	00F UNT 00110	Records untalk interface message.
NEXT	010 UNL 00110	Records unlisten interface message.

MONITOR MODE OPERATION
TABLE 3-1

Key Entry	Display Response	Comments
NEXT	011 DAB NL	Lack of binary data above five bus management line mnemonics indicates preceding address contained the last recorded transfer.
MODE	MODE? DATA MEM	Enables selection of a new mode.
NEXT	ADDR?'011 ASCII	Remains in DATA MEMORY mode and advances to next prompt state. Note that the address remains at its last value.
F	ADDR?'011 HEX	Selects display of DATA MEMORY contents in hex format.
0	ADDR?'0 HEX	
0	ADDR?'00 HEX	
E	ADDR?'00E HEX	Specifies DATA MEMORY address of '00E with hex display format.
NEXT	00E DAB'52 00100	Displays contents of DATA MEMORY address '00E in hex format equivalent to ASCII R.
NEXT	00F UNT 00110	
RESET	ADDR?'000 ASCII	Returns to address prompt state. Resets to address '000. Reverts to ASCII data format.

e) trigger a device

f) wait for service request

g) receive and compare status transmitted from a device

h) jump instructions to control program sequencing.

While in the bus language mode, the Model 488 serves as the bus controller. A system's normal controller must either be disconnected or disabled. The number of devices which the Model 488 may control while operating in bus language is in accordance with IEEE Std 488-1978; i.e., 14.

Each bus language instruction results in the generation of a fixed sequence of machine language instructions which, in turn, provide the necessary bus protocol messages to accomplish the instruction's function. All references to data by the bus language instructions are with respect to data either stored or recorded in the Model 488's data memory.

During execution of a user's bus language program, the Model 488 performs the programmed data and status tests and also automatically performs bus interface tests. Detection of an error causes the unit to halt for subsequent interrogation by the user to determine details of the error.

The following paragraphs provide details of bus language operation.

3.3.1 Mode Selection

To select the bus language mode, depress the MODE key and proceed as follows:

Entry	Display	Comment
MODE	MODE? current mode	
3	MODE? BUS LANG	
NEXT	LINE NO? nn	Mode LEDs reflect change to BUS LANGUAGE

To both select bus language and to clear its associated program memory, perform the following:

Entry	Display	Comment
MODE	MODE? current mode	
3	MODE? BUS LANG	
CM	CLEAR BUS LANG	
NEXT	LINE NO? 00	Mode LEDs reflect change to BUS LANGUAGE. Subsequent examination of the BUS LANGUAGE program memory will demonstrate the requested clear.

3.3.2 Reset Function

Depression of the RESET KEY at any time will cause the following:

a) stop the program if running.

b) preset the line number to 00.

c) preset the data memory address to '000.

d) cause a display of LINE NO? 00.

e) clear all bus control lines.

3.3.3 Line Number Definition

Bus language instructions may be considered to be loaded into a memory in which locations are identified by line numbers rather than addresses. The line numbers are two digit decimal numbers ranging from 00 to 60. The first instruction of a user's bus language program must be located at line number 00. All succeeding line numbers must be assigned continuously up to the user's last instruction; i.e., no gaps are permitted in line number sequences.

Following initial power-up of the Model 488 or execution of the clear memory (CM) function, the bus language memory is cleared; i.e., all line numbers are unassigned. Therefore, the user is intially able to access only line number 00. Following entry of a bus language program, the user may then randomly access the contents of previously defined line numbers.

Following the mode selection sequence or depression of the RESET key, the user is requested to enter the desired line number: LINE NO? nn. After entry of a valid line number, depression of the NEXT key causes the selected line number and its current contents to be displayed. If the line number were invalid, a flashing question mark is displayed prompting the user to depress the CE key which, in turn, restores the original line number.

3.3.4 Instruction Entry and Editing

To add a bus language instruction or to change an existing instruction, the user need only depress the instruction's associated op code mnenonic key and then enter the required instruction arguments. Entry of the instruction op code temporarily replaces the current line number's contents. The arguments are validated as they are entered. Should the user make an entry error, a flashing question mark will be displayed prompting the user to depress the CE key (clear entry). The CE key restores the original line number's contents.

Had the user previously entered a valid bus language instruction, depression of the NEXT key stores the instruction as the contents of the current line number. The line number is then incremented and displayed along with its contents. The LAST key performs the same function as the NEXT key except that the line number is decremented. Use of the NEXT and LAST keys does not necessarily require the user to enter a new instruction; i.e., they may be used to simply increment/decrement the line numbers in order to inspect the current contents of the bus language program memory. Note that the use of the NEXT and LAST keys are subject to the restriction of line number assignment previously discussed.

3.3.5 Instruction Description

Table 3-2 provides a summary description of the bus language instruction set. Depression of an instruction's associated op code mnemonic key results in the display of the mnemonic and a series of underlines in order to prompt the user to enter the necessary instruction arguments. An apostrophe preceding an arguent field denotes that the argument is in a hexadecimal format; lack of a preceding apostrophe indicates the field is in decimal.

Instructions which transfer data (WT, RR, RC) reference the contents of the Model 488 data memory which the user is responsible for establishing prior to executing the bus language program.

Each bus language instruction results in the generation of a fixed sequence of machine language instructions which, in turn, include the necessary bus language mode overhead and the bus interface protocol. The machine language program memory is capable of storing 255 machine language instructions. The user may observe the equivalent machine language instructions of a bus language program by simply changing the mode from bus language to machine language. It should also be noted the user could then alter the generated machine language instructions to fit his specific requirements.

The following paragraphs provide additional details of each bus language instruction including the associated bus interface protocol.

3.3.5.1 CL, Clear

The CL instruction accomplishes both an interface clear (IFC) and a device clear (DCL) routine. Generally, it is preferrable that a CL instruction be assigned to line number 00 since the CL instruction is nearly the most basic communication which may occur on the bus. The IFC routine essentially sets the IFC control line true, clears the attention (ATN) line, and, after 100 usec, tests that the three handshake lines are false. It then clears IFC, asserts ATN, and, after 2 usec, tests that the not-data-accepted (NDAC) is true. The DCL routine performs the necessary handshake communication to transmit the associated multiline message. The handshake transfer is tested for sequencing errors. Even if the device(s)

TABLE 3-2
MODEL 488
BUS LANGUAGE INSTRUCTION SET

Keyboard Entry	Description
CL	Clear interface and all devices.
WT'xx'aaa nnnE	Write data block to device 'xx.* 'xx: device's listen address, '00 ≤ 'xx ≤ '1E. 'aaa: data memory address of first byte, '000 ≤ 'aaa ≤ '1FE. nnn: number of data bytes, 001 ≤ nnn ≤ 511. E: transmit END message (optional).
RR'xx'aaa nnn	Read and record data block received from device 'xx.* 'xx: device's talk address, '00 ≤ 'xx ≤ '1E. 'aaa: data memory address of first storage location, '000 ≤ 'aaa ≤ '1FE. nnn: number of bytes to be recorded, 001 ≤ nnn ≤ 511. If E is entered, recording will continue until either an END message is received or data memory is full (511 bytes maximum).
RC'xx'aaa nnnE	Read and compare data block received from device'xx.* If the received data does not compare to the corresponding data stored in data memory, the program will halt and display an error message. 'xx: device's talk address, '00 ≤ 'xx ≤ '1E. 'aaa: data memory address of first comparison location, '000 ≤ 'aaa ≤ '1FE. nnn: number of bytes to be compared, 001 ≤ nnn ≤ 511. E: test for END message (optional).
TR'xx	Trigger device 'xx. 'xx: device's listen address, '00 ≤ 'xx ≤ '1E.
SR	Wait for service request (SRQ).
RS'xx'mm'ss	Read and compare serial poll status byte from device 'xx. If the received status byte does not compare to the expected byte, the program will halt and display an error message. 'xx: device's talk address, '00 ≤ 'xx ≤ '1E. 'mm: mask to enable/disable (1/0) comparison of selected bits, '00 ≤ 'xx ≤ 'FF, 'ss: expected status byte.
JS t nn	Jump to instruction line number nn if SENSE SWITCH 2 is in selected position; otherwise, continue to the next line number. t: SENSE SWITCH 2; 0=not depressed (switch out), 1=depressed (switch in). nn: instruction line number (decimal).
JU nn	Jump unconditionally to line number nn. nn: instruction line number (decimal).

NOTES

1. All hexadecimal fields are preceded by an apostrophe. All other fields are decimal.
2. Refer to Appendix G for bus address code conversion.

*3. If SENSE SWITCH 1 is depressed, one byte will be transmitted, recorded, or compared for each depression of the SINGLE STEP switch.

*4. aaa + nnn must be less than 511, the maximum data memory address.

under test does not implement the DCL function, it must still perform the handshake sequence.

The CL instruction results in the generation of five machine language instructions.

3.3.5.2 WT, Write

The Wt instruction first clears the current talker (UNT) and listeners (UNL) and then sets up the specified device as a listener (MLA'xx). The remainder of the WT instruction accomplishes the data transfer. If SENSE 1 is not depressed, the referenced data memory contents are sequentially transmitted.at a rate up to 250 KHz, dependent on the slowest device. If SENSE 1 is depressed, a data byte is transmitted following each depression of the SINGLE STEP key. If an E had been entered (optional) as the last argument, the END message (END=ATN EOI) is transmitted concurrently with the last data byte.

The WT instruction results in the generation of 13 machine language instructions.

3.3.5.3 RR, Read and Record

The RR instruction first clears the current talker (UNT) and listeners (UNL) and then sets up the specified device as a talker (MTA'xx). As in the WT instruction, SENSE 1 determines whether the data transfer occurs in burst mode or single step mode. The user has the option of specifying whether a specific number of data bytes are recorded in the data memory or until an END message is received (subject to 511 bytes maximum). The RR instruction will wait ad infinitum until the specified number of data bytes are received (or, optionally, until an END occurs).

The RR instruction results in the generation of 13 machine language instructions.

3.3.5.4 RC, Read and Compare

The RC instruction first clears the current talker (UNT) and listeners (UNL) and then sets up the specified device as a talker (MTA'xx). As in the WT instruction, SENSE 1 determines whether the transfers occur in burst or single step mode. The first data byte received is compared to the specified data memory contents. Succeeding data bytes are compared to succeeding data memory contents. If a comparison error is detected, the program will halt and indicate DATA ERROR. The user may determine further details of the error by proceeding to the error status mode. The user may optionally test for the occurrence of the END message in which case the program will halt if either an END message did not occur concurrent with the last data byte or an END message occurred before the last expected byte. As in the RR instruction, the RC instruction will continue waiting until the specified number of transfers has occurred.

The RC instruction results in the generation of 13 machine language instructions.

3.3.5.5 TR, Trigger

The TR instruction first clears the current talker (UNT) and listeners (UNL) and then sets up the specified device as a listener (MLA'xx). It then transmits the group execute trigger (GET) message to trigger the listening device.

The TR instruction results in the generation of 7 machine language instructions.

3.3.5.6 SR, Service Request

The SR instruction simply waits for the occurrence of a service request message (SRQ).

The SR instruction results in the generation of 4 machine language instructions.

3.3.5.7 RS, Read Status

The RS instruction first clears the current talker (UNT) and listeners (UNL). It then transmits the serial poll enable message (SPE) and sets up the specified device as a talker (MTA'xx). The instruction then compares the received data byte to the expected status byte in accordance with the mask enable. If an error is detected, the program halts and displays STAT ERROR. The user may then proceed to the error status mode to obtain details of the error. Following the comparison, the RS instruction transmits the serial poll disable message (SPD).

The RS instruction results in the generation of 12 machine language instructions.

3.3.5.8 JS, Jump on SENSE 2

The JS instruction tests for the specified state of the SENSE 2 switch and branches accordingly. The JS instruction provides the capability to manually alter the program sequencing.

The JS instruction results in the generation of 4 machine language instructions.

3.3.5.9 JU, Jump

The JU instruction performs an unconditional jump to the specified line number. A frequent application of the JU instruction is as a halt state to indicate completion of the program. The JU instruction is simply specified to jump to its own line number. The display will then indicate RUNNING nn where nn is the JU instruction's line number when the halt state is encountered.

The JU instruction results in the generation of 4 machine language instructions.

3.3.6 Programming Example

Table 3-3 provides an example of setting up and programming in bus language. The resultant program which is entered performs the following:

a) clears the interface

b) transmits a data table to a device

c) triggers the device to execute its task

d) waits for service request from the device

e) reads the device's status byte

f) if SENSE 2 is depressed, retriggers the device; otherwise, the program halts.

3.3.7 Program Running

Once the bus language instructions and the contents of data memory have been established, the user is prepared to run the program. It is generally advisable to begin execution of a program with the instruction of line number 00. Depression of RESET sets the current line number to 00. It is also necessary to insure that the three alternate action switches (SENSE 1, SENSE 2, BYPASS ERROR) are set to the desired position. The user may then proceed to the running state by depressing the RUN/STOP key. Illumination of the RUN LED indicates the running state.

Assuming no errors occur, the display will indicate RUNNING nn where nn provides a sampling of the instruction line numbers being executed. A constant line number display indicates that the program is predominately executing one instruction throughout the total program execution period. Such a condition obviously exists if the program included a halt. But likewise, the line number display will remain relatively constant while waiting for an external event. For example, a wait for service request (SR) instruction may be predominate in a program's execution period. Also, recall that the RR and RC instructions will wait until the specified number of bytes have been received.

If SENSE 1 is depressed, each transfer of the WT, RR and RC instructions occurs upon depression of the SINGLE STEP key.

The user may stop the program by again depressing the RUN/STOP key. The RUN LED will extinguish indicating the return to the programming state. The display will indicate the instruction which was being executed at the time of the stop. Restarting of a program by again depressing RUN/STOP

BUS LANGUAGE PROGRAMMING EXAMPLE
TABLE 3-3

Key Entry	Display Response	Comments
POWER	MODE? MONITOR	Power LED and MONITOR mode LED illuminate.
3	MODE? BUS LANG	Note that the mode is not changed until NEXT is depressed.
NEXT	LINE NO? ØØ	Advances to next prompt state. Mode LEDs indicate change from MONITOR to BUS LANGUAGE.
NEXT	ØØ	Displays line number ØØ and its contents. Because of powerup, the bus language program memory has been cleared.
CL	ØØCL	Inserts a CL instruction at line number ØØ.
NEXT	Ø1	Advances to line number Ø1.
WT	Ø1WT'__'___ ___	Inserts a WT instruction.
1E ØØØ Ø2Ø	Ø1WT'1E'ØØØ Ø2Ø	Specifies transmitting 2Ø data bytes which are stored in data memory at 'ØØØ-'Ø13 to device with listen address '1E. Since a suffix E had not been entered, no END message will be transmitted.
NEXT	Ø2	
TR	Ø2TR'__	Inserts a TR instruction.
ØE	Ø2TR'ØE	Specifies triggering device 'ØE.
NEXT	Ø3	
SR	Ø3SR	Inserts a wait for service request (SR) instruction.
NEXT	Ø4	
RS	Ø4RS'__'__'__	Inserts a read status (RS) instruction.

BUS LANGUAGE PROGRAMMING EXAMPLE
TABLE 3-3

Key Entry	Display Response	Comments
1E 7F 21	Ø4RS'1E'7F'21	Specifies reading the status byte from device '1E. The mask is set to enable comparison of the 7 lsbs. The expected status byte is '21.
NEXT	Ø5	
JS	Ø5JS _ __	Inserts a jump on SENSE 2 instruction (JS).
1 Ø2	Ø5JS 1 Ø2	Specifies jumping to line number Ø2 if SENSE 2 is depressed.
NEXT	Ø6	
JU	Ø6JU __	Inserts an unconditional jump instruction (JU).
Ø7	Ø6JU Ø7	Specifies jumping to line number Ø7 (i.e., a NOP).
CE	Ø6	Clears out last entry and restores to original contents (i.e., void).
JU	Ø6JU __	
Ø6	Ø6JU Ø6	Specifies a halt.
RESET	LINE NO? ØØ	The entered program may now be verified using the following sequence.
NEXT	ØØCL	
NEXT	Ø1WT'1E'ØØØ Ø2Ø	
NEXT	Ø2TR'ØE	Specified address should have been '1E.
TR	Ø2TR'__	
1E	Ø2TR'1E	
NEXT	Ø3SR	

BUS LANGUAGE PROGRAMMING EXAMPLE
TABLE 3-3

Key Entry	Display Response	Comments
NEXT	Ø4RS'1E'7F'21	
NEXT	Ø5JS 1 Ø2	
NEXT	Ø6JU Ø6	
LAST	Ø5JS 1 Ø2	
RESET	LINE NO? ØØ	
Ø8	LINE NO? Ø8 ?	Entry error. Line number Ø8 had not been previously defined.
CE	LINE NO? ØØ	
Ø7	LINE NO? Ø7	Valid entry. Line number Ø7, while undefined, is the next sequential line number to be assigned.
MODE	MODE BUS LANG	The user would now change to data memory mode to load the required transmission data at '000-'013. Refer to section 3.5 for a description of data memory operation.

is not recommended since the previous stop may have occurred somewhere in the middle of the execution of a bus language instruction. Rather, RESET should be used to fetch line number 00 again before starting.

The user may also stop program execution by depressing the RESET key which results in a display of LINE NO? 00.

3.3.8 Assembly Errors

When the RUN/STOP key is depressed to put the unit into the running state, the Model 488 automatically assembles the bus language instructions into a series of machine language instructions. If an invalid line number reference is detected during the assembly process, an error is indicated on the display. Subsequent depression of the CE key automatically displays the instruction possessing the invalid line number reference.

With reference to the example of Table 3-3, assume that the instruction of line number 06 had been a JU 08 instead of a JU 06. The following events would then occur:

Key Entry	Display
RESET	LINE NO? ØØ
RUN/STOP	ASSY ERR Ø2 Ø6
CE	Ø6JU Ø8

3.3.9 Test Errors

Should the bus language program detect an error, the display will indicate the appropriate message. For example, if the program detected an interface response error while executing the instruction of line number 01, the display would indicate HDWR ERROR 01. Additional details of the error condition may then be obtained by entering the error status mode. Refer to section 3.6 for a description of the error status mode.

If the user depresses SINGLE STEP following detection of an error, the Model 488 will continue program execution and will again halt if an error is detected. If the user depresses BYPASS ERROR, the unit will also continue running but will ignore all detected errors. It should be noted that DATA ERRORs and STAT ERRORs continue with the next sequential operation but HDWR ERRORS are referred to as nonrecoverable and automatically restart the program at line number 00. Nonrecoverable errors (e.g., DAC @ ATN) are essentially "fatal" to further bus operation and simple continuance with the next operation would be futile.

3.3.10 Single Instruction Execution

As opposed to running a bus language program, each instruction may be executed manually by depressing the SINGLE STEP key while the unit is in the programming state (i.e., RUN LED off). Each instruction is executed as in the running state with normal error detection. Note that if SENSE 1 is depressed, the WT, RR, and RC instructions will still require depression of the SINGLE STEP key to transfer each byte.

3.4 Machine Language Mode Operation

The Model 488's machine language mode permits the user to develop test programs at the machine code level. Advantages to machine language programming consist of increased memory usage efficiency and of customized tests peculiar to a user's instrument or system. In order to make machine language programming a manageable task, two features have been provided. First, instructions are entered from the front panel in hex format but are displayed in mnemonic format following validation. This feature provides the user with an immediate check of his desired vs. actual entry. Secondly, 25 of the 68 machine language instructions may be considered as macro-instructions in that they call "standard" subroutines stored in ROM. These standard subroutines omit the need for the user to discretely program bus signal protocol sequences such as handshakes and provide integral tests of interface operation and data transfers.

It should be noted that the machine language standard subroutines assume that the Model 488 is the controller-in-charge. The system's normal controller should either be de-activated or disconnected.

In order to fully understand the machine language mode, the user must be aware of the basic Model 488 architecture. (Details of the architecture are provided in Section V). The Model 488 consists of two primary functional elements: the input/output processor (IOP) and the bus processor (BP). The IOP controls all communications between the front panel switches and indicators, controller interfaces (e.g., RS-232C, card reader, controller IEEE), and the bus processor. The BP is a high speed processor which performs the actual IEEE bus test programs. The I/O processor loads the bus processor program memory (i.e., machine language memory) and the data memory in accordance with user entries. The IOP then directs the BP to begin execution. The bus processor communicates with the IOP via an interrupt to designate program completion or detection of an error.

3.4.1 Mode Selection

To select the machine language mode, proceed as follows:

Entry	Display	Comments
MODE	MODE? current mode	
4	MODE? MACH LAN	
NEXT	ADDR?'aaa	Mode LEDs reflect change to MACH LAN.

To both select machine language and clear its associated program memory, proceed as follows:

Entry	Display	Comments
MODE	MODE? current mode	
4	MODE? MACH LAN	
CM	CLEAR MACH LAN	
NEXT	ADDR?'aaa	Mode LEDs reflect change to MACH LAN. Subsequent examination of the MACH LAN memory will demonstrate the requested clear.

Prior to depressing NEXT, had the user decided he did not wish to clear the memory he could have depressed CE and the display would have reverted to "MODE? current mode".

3.4.2 Reset Function

Depression of the RESET key at any time will cause the following:

a) stop the program if running.

b) preset the machine language memory address to '000.

c) preset the data memory address counter to '000.

d) cause a display of ADDR?'ØØØ.

e) clear all bus control lines.

3.4.3 Machine Language Memory Addressing

The machine language program memory occupies addresses '000-'1FF of the bus processor memory. Addresses '000-'0FE are RAM memory providing storage for 255 user instructions. Addresses '100-'1FF are ROM memory and contain the standard subroutines. Note that address '0FF is unavailable to the user since it is used for communication between the IOP and BP.

Following the mode selection sequence or depression of the RESET key, the user is requested to enter the desired machine language program memory address: ADDR?'aaa. After entry of a valid address, depression of the NEXT key causes the selected address and its contents to be displayed. If the address had been invalid, a flashing question mark would be displayed, prompting the user to depress the CE key, which in turn, would restore the original address.

3.4.4 Instruction Entry and Editing

To add a machine language instruction or to change an existing instruction, the user must enter the necessary hex data to define the instructions op code and argument (if any). The user entered data temporarily replace the previous contents of the selected address. Upon entry of the second hex character, the associated op code mnemonic is displayed. The argument data are validated during the entry process. Should the user make an entry error, a flashing question mark will be displayed prompting the user to depress the CE key. The CE key restores the original contents of the selected address.

Had the user entered a valid machine language instruction, depression of the NEXT key stores the instruction as the contents of the selected address. The address is then incremented and displayed along with its contents. The LAST key performs the same function as the NEXT key except that the address is decremented. Use of the NEXT and LAST keys does not necessarily require the user to enter a new instruction; they may be used to simply increment/decrement the address in order to inspect the memory contents.

3.4.5 Instruction Description

Table 3-4 provides a detailed description of the machine language mode instructions. Appendix C is a summary listing of the machine language instructions.

The first 25 instructions (op codes '00-'18) are referred to as the standard subroutine instructions since they call the ROM subroutines stored at addresses '100-'1FF. The remaining 43 are conventional machine code instructions. Except for those noted, the majority are executed in one bus processor cycle of 200 nsec. The jump instructions permit the use of user defined subroutines with a maximum of 3 levels of nesting including calls to the standard subroutines.

Table 3-4 lists the hexadecimal keyboard data which must be entered to insert each of the instructions. The entered hex data for certain instructions are further modified or translated prior to storing in the machine language program memory. The translation typically consists of the addition of an implied argument. All instructions which undergo a translation are indicated in Table 3-4. If no translated machine code is listed, the data are entered directly into the program memory without modification.

Subsection 3.4.6 provides information on application of the machine language instructions.

TABLE 3-4
MODEL 488
MACHINE LANGUAGE INSTRUCTION SET

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
IFC	0 0 ☐ ☐ 0 1 0 0	1	Interface Clear Transmits Interface Clear using IFC subroutine. Translated machine code data is 0100.
DCL	0 1 ☐ ☐ 0 2 1 4	1	Device Clear Transmits Device Clear message using multiline interface message (MIM) subroutine. Translated machine code is 0214.
SDC	0 2 ☐ ☐ 0 2 0 4	1	Selective Device Clear Transmits Selective Device Clear message using MIM subroutine. Bus protocol requires selected devices be set up previously as listeners. Translated machine code data is 0204.
MLA aa	0 3 a a 0 2 7/3 0-F/0-E	1	My Listen Address Transmits My Listen Address message using MIM suboutine. aa: listen address, 00 ≤ aa ≤ 1E Translated machine code is: 0 2 001a a
UNL	0 4 ☐ ☐ 0 2 3 F	1	Unlisten Transmits Unlisten message using MIM subroutine. Translated machine code data is 023F.
MTA aa	0 5 a a 0 2 4/5 0-F/0-E	1	My Talk Address Transmits My Talk Address message using MIM subroutine. aa: talk address, 00 ≤ aa ≤ 1E. Translated machine code format is: 0 2 010a a

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
UNT	0 | 6 | | 0 2 5 F	1	Untalk Transmits Untalk message using MIM subroutine. Translated machine code data is 025F.
SCG ss	0 | 7 | s | s 6 0-F 0 2 ――― 7 0-F	1	Secondary Command Transmits Secondary Command Message (MSA,PPE,PPD) using MIM subroutine. ss: secondary command 00 ⩽ss ⩽1F. Translated machine code format is: 0 | 2 | 011s | s
LLO	0 | 8 | | 0 2 1 1	1	Local Lockout Transmits Local Lockout message using MIM subroutine. Translated machine code data is 0211.
GTL	0 | 9 | | 0 2 0 1	1	Go To Local Transmits Go-To-Local message using MIM subroutine. Translated machine code data is 0201.
GET	0 | A | | 0 2 0 8	1	Group Execute Trigger Transmits Group-Execute-Trigger message using MIM subroutine. Translated machine code data is 0208.
SPE	0 | B | | 0 2 1 8	1	Serial Poll Enable Transmits Serial Poll Enable message using MIM subroutine. Translated machine code data is 0218.
SPD	0 | C | | 0 2 1 9	1	Serial Poll Disable Transmits Serial Poll Disable message using MIM subroutine. Translated machine code data is 0219.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
PPC	0 D 0 2 0 5	1	Parallel Poll Configure Transmits Parallel Poll Configure message using MIM subroutine. Translated machine code data is 0205.
PPU	0 E 0 2 1 5	1	Parallel Poll Unconfigure Transmits Parallel Poll Unconfigure message using MIM subroutine. Translated machine code data is 0215.
IDY pp	0 F p p 0 3 0-F 0-F	1	Identify Compares parallel poll response using IDY subroutine. pp: expected poll response. 00 ≤ pp ≤ FF. Prior to IDY, the user must provide a SAV pp instruction to pass the expected poll response to the IOP. Specific bits may be enabled for comparison using the LDM instruction. Also prior to the IDY, the user must initialize the system for a parallel poll; i.e., MLA, (MSA), PPC and PPE. Translated machine code format is: 0 3 p p
TRB	1 0 1 0 6 D	1	Transfer Byte Transfers a single device dependent data byte to all listener devices using the data transmit subroutine. Transmitted data byte must previously be defined by an LDI or an LDD. Transmission of the END message simultaneous with the byte may be specified by setting b0 of FTF. Translated machine code data is 106D.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
TRR	1 1 □ □ 1 1 6 E	1	Transfer Block Transmits a block of device dependent data bytes to all listener devices using the data transmit subroutine. Previous instructions must specify the first data memory address (LDD) and the number of bytes to be transmitted (LC2). Reference Note 2. The END message may be transmitted with the last byte by setting b0 of FTF. Translated machine code data is 116E.
TRS	1 2 □ □ 1 2 6 8	1	Transfer Single-Step Identical to TRR instruction except that a byte is transmitted following each depression of the SINGLE STEP switch. Translated machine code data is 1268.
RDB	1 3 □ □ 1 3 9 8	1	Record Byte Reads a single device dependent data byte from the bus and records it in the data memory. Uses the read and record data subroutine. Transmitting device must have been previously setup as a talker (MTA). The data memory address must be previously defined utilizing an LDD instruction. Translated machine code data is 1398.
RDR	1 4 □ □ 1 4 9 E	1	Record Block Reads a block of device dependent data bytes from the bus and records it in the data memory. Uses the read and record data subroutine. Transmitting device must have been previously setup as a talker (MTA). An LDD must be previously executed to define the first data memory address. The user may either record the number of bytes specified by an LC2 instruction or until an END message is received by setting b0 of FTF. Note that the subroutine clears b0 of FTF upon exiting. Reference note 2. Translated machine code data is 149E.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
RDS	1 5 1 5 9 A	1	Record Single-Step Identical to the RDR instruction except that a byte is recorded following each depression of the SINGLE STEP switch. Translated machine code data is 159A.
CPB	1 6 1 6 C D	1	Compare Byte Reads a single device dependent data byte from the bus and compares it to data in the data memory. Uses the read and compare data subroutine. If an error is detected and the BYPASS ERROR switch is set, the program will not halt. If the BYPASS ERROR switch is not set, the program will halt enabling the front panel to display the invalid data. Depression of the SINGLE STEP switch will permit the program to continue. Transmitting device must have been previously setup as a talker (MTA). Previous instructions must also specify the comparison data (LDD), the mask data (LDM), and whether the END message is to be tested when the byte is received (set b0 of FTF). Translated machine code data is 16CD. Invalid data is temporarily stored in data byte memory location 1FF.
CPR	1 7 1 7 C E	1	Compare Block Reads a block of device dependent data bytes from the bus and compares it to a corresponding data block in the data memory. Uses the read and compare data subroutine. If an error is detected and the BYPASS ERROR switch is set, the program will not halt. If the BYPASS ERROR switch is not set, the program will halt enabling display of the invalid data. Depression of the SINGLE STEP switch will permit the program to continue.

-continued-

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
			Transmitting device must have been previously setup as a talker (MTA). Previous instructions must also specify the first comparison data byte in the data memory (LDD), the number of bytes to be read (LC2), the mask data (LDM), and whether the END message is to be tested when the last byte is received (set b0 of FTF). Reference note 2. Translated machine code data is 17CE. Invalid data is temporarily stored in data byte memory location 1FF.
CPS	| 1 | 8 | | | 1 8 C 8	1	Compare Single-Step Identical to CPR except that a byte is compared following each depression of the SINGLE STEP switch. Translated machine code data is 18C8. Invalid data is temporarily stored in data byte memory location 1FF.
—	| 1 | 9 | | | ↓		Undefined
—	| 1 | F | | |		
LDD aaa	| 2 | a | a | a | 2 0-1 0-F 0-E	2	Load Data Direct Loads data byte into output register from specified data memory location. Does not cause transfer. Used to preset data byte memory address counter and to fetch associated data byte for output transmissions. Also utilized to specify address for input operations (data is disregarded). aaa: data memory address, 000 ≤ aaa ≤ 1FE.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
LDN	2 4 2 4 0 0	3	Load Next Data Increments data memory address counter and loads the associated data byte into the output register. Does not cause a transfer.
STR	2 8 2 8 0 0	3	Store Data Stores the bus data byte in the last specified data memory location. The data memory address counter is not affected. Note that LDD and LDN may be used to specify the data memory address.
STD aaa	2 110a a a 2 (C/D) 0-F 0-F	3	Store Data Direct Stores the bus data byte into the specified address of the data memory. The data memory address counter is not affected by this instruction. This instruction has a special application to storing invalid bus data. In order to permit the front panel processor to gain access to the invalid data, the STD instruction must be executed such that the invalid data is stored in data memory location 1FF. Following an internal interrupt, the IOP will then automatically read location 1FF. aaa: data memory address, 000 ≤ aaa ≤ 1FF.
LCO cc	3 0 c c 3 0 (2's compliment)	1	Load Loop Counter 0 Loads loop counter 0 with specified count. Typical applications consist of counting events or timing delay loops. Loop counter 0 is decremented by JL0 instruction. cc: loop count, hex, 00 ≤ cc ≤ FF. Translated machine code format is: 3 0 'c 'c loop count, hex, 2's complement

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
LC1 cc	$\boxed{3}\boxed{2}\boxed{c}\boxed{c}$ 3 2 (2's complement)	1	Load Loop Counter 1 Identical to LC0. Decremented by JL1. Translated machine code format is: $\boxed{3}\boxed{2}\underbrace{\boxed{'c}\boxed{'c}}_{\text{loop count hex, 2's complement}}$
LC2 ccc	$\boxed{3}\boxed{010c}\boxed{c}\boxed{c}$ (4/5) 3 $\frac{4}{5}$ (2's complement)	1	Load Loop Counter 2. Loads loop counter 2 with specified count. Decremented by JL2. ccc: loop count, hex, 000 ≤ccc ≤ 1FF. Loop counter 2 is used by various subroutines as the byte counter; i.e., LC2 must be executed prior to calling the executed subroutine. The programmer must use caution if loop counter 2 is used for any purpose other than as a byte counter. Translated machine code format is: $\boxed{3}\boxed{010c}\underbrace{\boxed{'c}\boxed{'c}}_{\text{loop count, hex, 2's complement}}$
LC3 cc	$\boxed{3}\boxed{6}\boxed{c}\boxed{c}$ 3 6 (2's complement)	1	Load Loop Counter 3 Loads loop counter 3 with specified count. Decremented by JL3. cc: loop count, hex, 00 ≤cc ≤ FF. Loop counter 3 is dedicated to subroutine usage for programmed time delays. The programmer must not use loop counter 3 in the main program. Translated machine code format is: $\boxed{3}\boxed{6}\underbrace{\boxed{'c}\boxed{'c}}_{\text{loop count, hex, 2's complement}}$

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
LDI dd	| 3 | 8 | d | d | 3 8 0-F 0-F	1	Load Data Immediate Loads specified data into output data register. Does not cause transfer. Note that LDI does not affect the data memory address counter as opposed to LDD and LDN. dd: data byte, $00 \leq dd \leq FF$.
LDM mm	| 3 | C | m | m | 3 C 0-F 0-F	1	Load Mask Load specified mask data into mask register. Used to enable specific bits for comparison operations. mm: mask data, $00 \leq mm \leq FF$, 0 = disable, 1 = enable.
SAV	| 3 | E | z | z | 3 E 0-F 0-F	1	Save Stores specified data in interface register accessible to front panel processor. zz: data, $00 \leq zz \leq FF$ Used to pass bus language line number which is currently being processed. Also used to pass expected bus status data.
JUN (S) aaa	| 4 | 0s0a | a | a | 4	1	Unconditional Jump Causes unconditional jump to specified address. aaa: jump address, hex, $000 \leq aaa \leq 1FF$. 000 - 0FF = RAM 100 - 1FF = ROM, containing subroutines. s: 0 = jump to main program 1 = jump to subroutine; current address +1 saved in stack
JRN	| 4 | 8 | | | 4 8 0 0	1	Jump and Return Causes program to jump to last address stored in stack; i.e., returns from subroutine to main program.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
JL0 (S) T/F aaa	5 0sta a a 5 0-7	1	Jump on Loop Count 0 Program jumps to specified address if loop counter 0 is equal to 1 (t=1) or is not equal to 1 (t=0). The state of the counter is tested at the beginning of the instruction. At the end of the instruction execution, the counter is decremented. Reference JUN format for description of s and aaa.
JL1 (S) T/F aaa	5 1sta a a 5 8-F	1	Jump On Loop Count 1 Identical to JL0.
JL2 (S) T/F aaa	6 0sta a a 6 0-7	1	Jump on Loop Count 2 Identical to JL0. Loop counter 2 is used as the byte counter for various subroutines. The programmer is cautioned about using loop counter 2 for any other purpose.
JL3 (S) T/F aaa	6 1sta a a 6 8-F	1	Jump on Loop Counter 3 Identical to JL0. Loop counter 3 is used as a programmable delay counter by various subroutines. The programmer is cautioned about using loop counter 3 for any other purpose.
JS1 (S) T/F aaa	7 0sta a a 7 0-7	1	Jump on SENSE SWITCH 1 Program jumps to specified address if SENSE SWITCH 1 is set (t=1) or reset (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JS2 (S) T/F aaa	7 1sta a a 7 8-F	1	Jump on SENSE SWITCH 2 Identical to JS1.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
JBE (S) T/F aaa	8 | 0sta | a | a 8 0-7	1	Jump on BYPASS ERROR switch Program jumps to specified address if BYPASS ERROR switch is set (t=1) or reset (t=0). The BYPASS ERROR switch is used by various subroutines. The programmer is cautioned about using it for other purposes. Reference JUN format for description of s and aaa. Reference note 3.
JFE (S) T/F aaa	8 | 1sta | a | a 8 8-F	1	Jump on END Enable Flag Program jumps to specified address if END enable flag is set (t=1) or is reset (t=0). Ref b0 of FTF. Reference JUN format for description of s and aaa. Reference note 3.
JAT (S) T/F aaa	9 | 0sta | a | a 9 0-7	1	Jump on ATN Program jumps to specified address if ATN line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JRE (S) T/F aaa	9 | 1sta | a | a 9 8-F	1	Jump on REN Program jumps to specified address if REN line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JIF (S) T/F aaa	A | 0sta | a | a A 0-7	1	Jump on IFC Program jumps to specified address if IFC line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JEI (S) T/F aaa	A | 1sta | a | a A 8-F	1	Jump on EOI Program jumps to specified address if EOI line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
JSR (S) T/F aaa	B 0sta a a B 0-7	1	Jump on SRQ Program jumps to specified address if SRQ line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JDV (S) T/F aaa	B 1sta a a B 8-F	1	Jump on DAV Program jumps to specified address if DAV line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JNR (S) T/F aaa	C 0sta a a C 0-7	1	Jump on NRFD Program jumps to specified address if NRFD line is true (t=1) or false(t=0). Reference JUN format for description of s and aaa. Reference note 3.
JND (S) T/F aaa	C 1sta a a C	1	Jump on NDAC Program jumps to specified address if NDAC line is true (t=1) or false (t=0). Reference JUN format for description of s and aaa. Reference note 3.
JCE (S) T/F aaa	D 0sta a a D 0-7	1	Jump on Compare Equal Program jumps to specified address if a comparison was detected (t=1) or was not detected (t=0). Comparator compares data stored in data memory output register against received bus data. Reference JUN format for description of s and aaa. Reference note 3.
JTF (S) T/F aaa	D 1sta a a D 8-F	1	Jump on Test Flag Progam jumps to specified address if the test flag is set (t=1) or reset (t=0). Reference b1 of FTF. The test flag is utilized by various subroutines. The programmer should exercise caution in using the test flag. Reference JUN format for description of s and aaa. Reference note 3.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
FAT T/F	E 0 f E 0 0 0/1	1	ATN Flag Causes ATN line to be true if f=1 or false if f =0.
FRE T/F	E 1 f E 1 0 0/1	1	REN Flag Causes REN line to be true if f=1 or false if f=0.
FIF T/F	E 2 f E 2 0 0/1	1	IFC Flag Causes IFC line to be true if f=1 or false if f=0.
FEI T/F	E 3 f E 3 0 0/1	1	EOI Enable Flag Causes EOI enable flag to set if f=1 or reset if f=0. When set, the flag enables the generation of the END message concurrent with transmission of the last byte of device dependent data or the generation of the IDY message for parallel polling.
FSR T/F	E 4 f E 4 0 0/1	1	SRQ Flag Causes SRQ line to be true if f=1 or false if f=0.
FDV T/F	E 5 f E 5 0 0/1	1	DAV Flag Causes DAV line to be true if f=1 or false if f=0.
FNR T/F	E 6 f E 6 0 0/1	1	NRFD Flag Causes NRFD line to be true if f=1 or false if f=0.
FND T/F	E 7 f E 7 0 0/1	1	NDAC Flag Causes NDAC line to be true if f=1 or false if f=0.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
FOT T/F	E 8 f E 8 0 0/1	1	Output Enable Flag The output register of the data memory is enabled onto the bus if f=1 or is disabled onto the bus if f=0.
FPI i	E 9 oiii E 9 · 0 0-5	1	IOP Interrupt Interrupts the IOP and transmits following status: iii 000 : hardware error 001 : data error 010 : status error 011 : line number (only BUS LANG) 100 : program complete 101 : single step request The FPI instruction must be followed by a jump-to-self instruction; i.e., JUN $. If the FPI is located in RAM (000-0FF), the IOP will restart at the next instruction beyond the JUN $. If the FPI is located in ROM (100-1FF), the IOP will restart at a predetermined location.
FTF f	E A f E A 0 0/1	1	Test Flags Controls test flags. Bit b0 state is sensed by JFE. Bit b1 is sensed by JTF. The test flags are used by various subroutines to pass arguments. It is recommended that they not be used by the programmer.
FCL	E B E B 0 0	1	Clear Flags Clears all flags.

Table 3-4
Model 488
Machine Language Instruction Set (continued)

Display Mnemonic	Keyboard Entry	Cycles(1)	Description
—	E C		Undefined
	↓		
—	E F		
—	F		Undefined

NOTES:

(1) No. of cycles (200 nsec/ea) excluding called subroutine.

(2) The number of bytes (LC2) must not cause attempted data memory accessing beyond the maximum address of 1FF; (LC2) + (LDD) < 1FF. Should this occur by error, the address is held at 1FF until the number of bytes has been satisfied. The data is meaningless.

(3) The referenced jump instructions sample the state of their respective sense lines at the beginning of the instruction cycle. The user should remain aware of this condition when operating in the single execution mode. For example, if the current instruction were a JS1 F aaa and SENSE SWITCH 1 were then depressed, the first SINGLE STEP execution of the instruction will disregard the switch change of state.

3.4.6 Instruction Application

This section discusses application of the machine language instructions to the generation of the functions necessary to test an instrument's IEEE Std 488-1978 interface. Methods of accomplishing the following bus functions are presented:

a) interface clear

b) listener addressing

c) talker addressing

d) device clear

e) remote/local

f) device trigger

g) serial poll

h) parallel poll

i) data transmission

j) data recording

k) data comparison

It is not to be implied that the methods described represent the only or the most efficient means of performing the bus functions. The methods center primarily about the use of the stardard ROM subroutine instructions. The referenced subroutines provide the machine level instructions necessary for testing and processing the bus message protocol and sequencing. The user is thereby relieved of developing programs to perform repetitive tasks such as handshake sequencing and bus management line control. As an alternative, the user may develop his own analogous RAM resident subroutines in order to test characteristics peculiar to his instrument or system. The procedure for developing customer subroutines is discussed in subsection 5.3.

The last paragraph provides application information regarding use of program interrupts.

3.4.6.1 Interface Clear

To accomplish the interface clear function, the user need only execute the IFC instruction (op code '00). It is recommended that an IFC instruction be located at address '000 of the user's machine language program. The IFC instruction verifies that the handshake lines (NRFD, NDAC, DAV) are not "stuck" and that an instrument is indeed connected to the bus. It also sets the attention (ATN) and remote enable (REN) lines to initially take control of the bus.

3.4.6.2 Listener Addressing

A device must be set up as a listener to receive device dependent data and to act upon certain multiline interface messages (i.e., commands) such as selective device clear (SDC) and group execute trigger (GET).

If a device's interface has a listen only capability (ton), it may be manually set up as a listener by activating the associated switch.

If a device's interface incorporates the L function (i.e., no secondary addressing), the user must execute an MLA'aa instruction (op code '03) where 'aa represents the hex code of the device's listen address switches. Refer to Appendix G for bus address code conversions.

If a device's interface incorporates the LE function (i.e., secondary addressing), the program must first execute an MLA'aa followed immediately by an SCG 'ss instruction (op code '07). The argument 'ss represents the desired secondary address. In this sequence of instructions, the SCG is implied to be an MSA message (my secondary address).

It is generally good programming practice to clear a device as a listener by executing an unlisten instruction (UNL, op code '04) when communications are complete to a device.

3.4.6.3 Talker Addressing

A device must be set up as talker to transmit device dependent data or status. All talkers must have unique addresses and only one may be active at a time.

If a device's interface includes the talk only (ton) capability, it may be manually set up as talker by activating the associated switch.

If a device's interface includes the T function (i.e., no secondary addressing), the user must execute an MTA'aa instruction (op code '05) where 'aa represents the hex code of the device's talk address switches. Refer to Appendix G for bus address code conversions.

If a device's interface incorporates the TE function (i.e., secondary addressing), the program must first execute an MTA'aa instruction immediately followed by an SCG'ss instruction (op code '07). The argument 'ss represents the desired secondary address. In this sequence of instructions, the SCG instruction is implied to be an MSA message (my secondary address).

It is generally good programming practice to execute an untalk instruction (UNT, op code '06) to remove the device as a talker when communications are complete from the device.

3.4.6.4 Device Clear

If implemented within a device's interface, the device clear function enables the controller to initialize a device's functions. The distinction between interface clear and device clear is that an interface clear affects only a device's interface while the device clear affects only the device itself.

If the user executes a DCL instruction (op code '01), all instruments possessing the device clear function will respond regardless of whether they had been previously set up as a listener. It is recommended that a DCL instruction be included as the second instruction of a user's program following the IFC instruction. Even if a device does not implement the device clear function, it still must respond to the command's handshake sequence which is tested during the execution.

If the user wishes to selectively clear specific devices, he must first set up the devices as listeners (MLA'aa) and then execute an SDC instruction (op code '02).

3.4.6.5 Remote/Local

If implemented within a device's interface, the remote/local function permits selection of device control from either the IEEE bus or from the device's operator panel.

If the local lockout capability is excluded (subset RL2), the remote/local function is straight forward in use. Recall that the IFC instruction sets the remote enable line (REN) true.

(i) to select remote control, the user need only execute an MLA'aa instruction, assuming the REN line is asserted.

(ii) to return the device to local control, the program must setup the device as a listener (MLA'aa and, optionally, MSA'ss) if not previously accomplished and then execute a go to local instruction (GTL, op code '09).

(iii) the REN line could also have been set false (op code 'E1) to return to local but would have affected all devices on the bus.

If the local lockout capability is included (subset RL1), the remote/local function is not as straight forward. The following procedure is only one of several methods for selecting remote or local control. The user should reference the RL state diagram of IEEE Std 488-1978 to develop and test the remaining methods.

(i) to proceed from the local control state to the local control with lockout state, the program should execute a local lockout instruction (LLO, op code '08), assuming the REN line is asserted.

(ii) to advance to the remote control with lockout state, the user should then execute an MLA'aa instruction.

(iii) to return to the local control with lockout state, the program must first set up the device as a listener (MLA'aa and, optionally, MSA'ss) and then execute a GTL instruction.

(iv) as under the no lockout capability procedure, the REN line could be cleared to return to the local state.

3.4.6.6 Device Trigger

The optional device trigger function permits the controller to trigger or start the basic operation of all devices which are set up as listeners. Assuming a device's interface implements the device trigger function, the user must first set up the device as a listener (MLA'aa and, optionally, MSA'ss) if not previously accomplished and then execute a group execute trigger instruction (GET, op code 'OA).

3.4.6.7 Serial Poll

A device's interface which includes the serial poll capability permits the controller to read the device's status byte. Closely related to the serial poll capability is a device interface's optional service request function. The service request function enables the device to set the interface SRQ line in order to request service from the controller.

The two methods described below may be used to process the service request function and the serial poll capability. The first reads and records the device status byte while the second reads and compares the status byte. To read and store device status:

Instruction	Comments
JSR F $	Wait for service request; i.e., jump to current address if SRQ line is false. ($ = address of current instruction).
SPE	Enable interface for serial poll.
UNL	Disable current listeners.
MTA'aa	Set up first device as talker to transmit status.
(SCG'ss)	MSA; optional secondary address.
LDD'aaa	Fetch location in Model 488's data memory to store status byte.
RDB	Read and record status byte.
MTA'ab	Repeat for next device.
(SCG'sb)	
LDD'aab	
RDB	
UNT	Disable current talker.
SPD	Disable serial poll.

To read and compare device status:

Instruction	Comments
JSR F $	Wait for service request; i.e., jump to current address if SRQ is false.
SPE	Enable interface for serial poll.
UNL	Disable current listeners.
FTF 0	Disable testing for END message.
MTA'aa	Set up first device as talker to transmit status.
(SCG'ss)	MSA: optional secondary address.
LDD'aaa	Fetch data byte from Model 488 data memory which is to be compared to received status byte.
LDM'mm	Load comparator's mask enable register. "1" enables bit comparison.
CPB	Read and compare device status byte.
MTA'ab	Repeat for second device
(SCG'sb)	
LDD'aab	
LDM'mb	
CPB	
UNT	Disable current talker.
SPD	Disable serial poll mode.

3.4.6.8 Parallel Poll

The optional parallel poll function enables the controller to read a single status byte in which devices are assigned one bit each to communicate their status. The IDY instruction of the Model 488 is used to read the parallel poll status byte and compare it to an expected status byte. Prior to performing a parallel poll, the parallel poll functions of the device interfaces must be configured and enabled.

A device interface which has local configuration capability (subset PP2) of its parallel poll function is configured or disabled using local device controls such as switches.

A device interface which has remote configuration capability (subset PP1) may be initially configured and enabled using the following instruction sequence:

Instruction	Comments
UNL	Disable current listeners.
MLA'aa	Set up desired device as a listener.
(SCG'ss)	MSA; optional secondary addressing.
PPC	Set up device to configure for parallel poll.
SCG'ss	PPE, parallel poll enable. 'ss b4 = 0 b3 = sense of status bit b2-b0 = DIO line assignment
UNL	Disable device as listener

(repeat for additional device's as necessary)

Once the device interfaces have been configured and enabled for parallel poll, the following instruction sequence may be used to conduct the poll:

Instruction	Comments
LDM'mm	Load comparator mask enable register. "1" enables comparison.
SAV'pp	Store expected status byte in IOP accessible register.
IDY'pp	Perform poll and compare received status byte to expected status 'pp.

To disable or unconfigure the parallel poll functions of devices with remote configuration capability, execution of the parallel poll unconfigure instruction (PPU, op code 'OE) will affect all devices. To selectively disable/unconfigure, execute the following instruction sequence:

Instruction	Comments
UNL	Disable current listeners.
MLA'aa	Set up selected device as a listener.
(SCG'ss)	MSA, optional secondary address.
PPC	Setup for parallel poll configuration.

SCG'ss	PPD, parallel poll disable.
	'ss
	b4 = 1
	b3-b0 = don't care; preferred Ø

UNL	Disable device as listener.

3.4.6.9 Data Transmission

To transmit a single data byte to a selected device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MLA'aa	Set up selected device as a listener.
(SCG'ss)	MSA, optional secondary address.
LDD'aaa	Fetch data byte to be transmitted from Model 488 data memory. As an alternative, LDI'dd may be used where 'dd represents the data to be transmitted.
FTF 1/0	Select transmission of END message: "1" = END transmission.
TRB	Transmission data byte.
UNL	Disable device as a listener.

To transmit a block of data bytes to a device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MLA'aa	Set up selected device as a listener.
(SCG'ss)	MSA, optional secondary address.
LDD'aaa	Fetch first byte of data block resident in data memory.
LC2'nnn	Define number of bytes in data block.
FTF 1/0	Select transmission of END message: "1" = END transmission.
TRR	Transmit data block. As an alternative, TRS will cause each byte of the block to be transmitted upon depression of the SINGLE STEP key.
UNL	Disable device as a listener.

3.4.6.10 Data Recording

To read and record a single data byte transmitted from a selected device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MTA'aa	Set up selected device as a talker.
(SCG'ss)	MSA, optional secondary address.
LDD'aaa	Fetch address of data memory where received data byte is to be stored.
RDB	Read and record data byte.
UNT	Disable device as a talker.

To read and record a block of data bytes from a selected device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MTA'aa	Set up selected device as a talker.
(SCG'ss)	MSA, optional secondary address.
LDD'aaa	Fetch first address of data memory where received data block is to be stored.
LC2'nnn	Define number of bytes in data block. As an alternative, recording may continue until END message is received by executing instead FTF 1.
RDR	Read and record data block. As an alternative, an RDS instruction may be used in which case each data byte is recorded for each depression of the SINGLE STEP key.
UNT	Disable device as a talker.

The user should be aware that the read and record instructions will wait until the specified number of bytes have been received.

3.4.6.11 Data Comparisons

To read and compare a single data byte transmitted from a selected device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MTA'aa	Set up selected device as a talker.
(SCG'ss)	MSA, optional secondary address.
LDD'aaa	Fetch comparison byte from data memory.
LDM'mm	Load comparator mask enable register. "1" enables comparison.
FTF 1/0	Select for testing of END message. "1" = END test.
CPB	Read and compare data byte.
UNT	Clear device as a talker.

To read and compare a block of data bytes transmitted from a selected device:

Instruction	Comments
UNT UNL	Disable current talker and listeners if not previously accomplished.
MTA'aa	Set up device as a talker.
(SCG 'ss)	MSA, optional secondary address.
LDD'aaa	Fetch first comparison byte from data memory.
LC2'nnn	Define number of data bytes in block.
LDM'mm	Load comparator mask enable register. "1" enables comparison.
FTF 1/0	Select testing of END message. "1" = test END
CPR	Read and compare data block. As an alternative, a CPS instruction will read and compare each byte upon depression of the SINGLE STEP key.
UNT	Disable device as a talker.

The user should be aware that the read and compare instructions will wait until the specified number of bytes have been received.

3.4.6.12 Program Interrupts

The FPI instruction provides the means for the Model 488's bus processor (BP) to indicate an event to the input/output processor (IOP). The standard subroutine instructions use the FPI to mark detection of errors such that the IOP may further interrogate the BP to provide additional information related to the errors. The FPI also may be conveniently used by the user in his RAM program. An FPI instruction requires that it be immediately followed by a halt (i.e., JUN $ where $=address of JUN instruction itself).

A frequent application is to include an FPI 4/JUN $ at the end of the user's program. When executed, the display will indicate "DONE".

A second application is to include an FPI 5/JUN $ to prompt use of the SINGLE STEP key. When executed, the display will indicate "STEP? Oaa". Subsequent operator depression of the SINGLE STEP key causes the program to continue with the next instruction following the JUN $.

Likewise, the FPI 0,1,2, may be used by the user in programs which include special test routines located in RAM. When executed, the display will indicate the generic error class. As before, depression of SINGLE STEP will cause the program to continue with instructions following the JUN $.

3.4.7 Operating Example

Table 3-5 provides an example of setting up and programming in machine language. The resultant program which is entered would perform the following:

a) clear the interface
b) clear the device
c) transmits a data table to a device
d) triggers the device to execute its task
e) waits for service request from the device
f) reads the device's serial poll status byte
g) if SENSE 2 is depressed, retriggers the device
h) if SENSE 1 is depressed, re-initiates the program; otherwise halts.

3.4.8 Program Running

Once the machine language instructions and the contents of data memory have been established, the user is prepared to run the program. It is generally advisable to begin execution of a program with the instruction of address '000. Depression of RESET sets the current address to '000. It is also necessary to insure that the three alternate action switches (SENSE 1, SENSE 2, BYPASS ERROR) are set to the desired position. The user may then proceed to the running state by depressing the RUN/STOP key. Illumination of the RUN LED indicates the running state.

Assuming no errors occur, the display will indicate RUNNING aaa where aaa provides a sampling of the instruction addresses being executed. A constant address display indicates that the program is predominately executing one instruction throughout the total program execution period. Such a condition obviously exists if the program included a halt. But likewise, the address display will remain relatively constant while waiting for an external event. For example, a wait for service request (JSR F) instruction may be predominate in a program's execution period. Also, recall that the read and record and the read and compare instructions will wait until the specified number of bytes have been received.

Recall that the machine language program memory is cleared upon power-up and may be cleared using the CM function during mode setup. If the unit is directed to run and then encounters, due to a programming error, an address for which no instruction has been defined, the program automatically jumps to address '100 and halts. The display will indicate RUNNING aaa where aaa is the address for which no instruction has been defined.

The user may stop the program by again depressing the RUN/STOP key. The RUN LED will extinguish indicating the return to the programming state. The display will indicate the instruction which was being executed at the time of the stop. Restarting of a program by again depressing RUN/STOP is not recommended since the previous stop may have occurred somewhere in the middle of the execution of a standard subroutine instruction. Rather, RESET should be used to fetch address '000 again before restarting.

MACHINE LANGUAGE OPERATING EXAMPLE
TABLE 3-5

Key Entry	Display Response	Comments
POWER	MODE? MONITOR	Power LED and MONITOR mode LED illuminate.
4	MODE? MACH LANG	Note that the mode is not changed until NEXT is depressed.
NEXT	ADDR?'ØØØ	Advances to the next prompt state. Mode LEDs indicate change from MONITOR to MACH LANG.
NEXT	ØØØ	Fetches contents of address 'ØØØ. Since it is assumed the Model 488 had just been powered-up, the machine language program memory is cleared.
ØØ	ØØØ IFC	Enters IFC as first instruction to clear interface.
NEXT	ØØ1	Stores IFC as contents of address ØØØ and increments address to ØØ1.
Ø1	ØØ1 DCL	Enters DCL instruction to clear all devices.
NEXT	0Ø2	
Ø3	ØØ2 MLA	Enters MLA op code.
1E	ØØ2 MLA 1E	Enters address '1E as argument of MLA instruction. Will set up device '1E as a listener.
NEXT	ØØ3	
2ØØØ	ØØ3 LDD ØØØ	Will fetch the contents of data memory address 'ØØØ.
NEXT	ØØ4	
3414	ØØ4 LC2 Ø14	Will set loop counter 2 to count '14 bytes (decimal 20).
NEXT	ØØ5	

MACHINE LANGUAGE OPERATING EXAMPLE
TABLE 3-5

Key Entry	Display Response	Comments
EAØ	ØØ5 FTF Ø	Will prevent transmitting of END message.
NEXT	ØØ6	
11	ØØ6 TRR	Will cause transmission of 20 (decimal) bytes located in data memory addresses 'ØØØ-Ø13 to device '1E.
NEXT	ØØ7	
ØA	ØØ7 GET	Will trigger current listener (device '1E).
NEXT	ØØ8	
Ø4	ØØ8 UNL	Will disable current listeners.
NEXT	ØØ9	
BØØA	ØØ9 JSR Ø ØØA	Will jump to program address 'ØØA if SRQ line is false; otherwise, will continue with next instruction.
NEXT	ØØA	
ØB	ØØA SPE	Will set up devices for serial poll.
NEXT	ØØB	
Ø51E	ØØ8 MTA 1E	Will set up device '1E as a talker.
NEXT	ØØC	
2Ø2Ø	ØØC LDD Ø2Ø	Will fetch the contents of data memory 'Ø2Ø.
NEXT	ØØD	
3C7F	ØØD LDM 7F	Will load the comparator mask enable register with '7F; i.e., msb will be disabled.

MACHINE LANGUAGE OPERATING EXAMPLE
TABLE 3-5

Key Entry	Display Response	Comments
NEXT	ØØE	
16	ØØE CPB	Will cause comparison of received data byte to contents of data memory address 'Ø2Ø.
NEXT	ØØF	
Ø6	ØØF UNT	Will disable current talker.
NEXT	Ø1Ø	
ØC	Ø1Ø SPD	Will disable serial poll mode.
NEXT	Ø11	
Ø31E	Ø11 MLA 1E	Will set up device '1E as a listener in preparation for repeating program.
NEXT	Ø12	
7AØ7	Ø12 JS2 1 ØØ7	Will cause jump to program memory address 'ØØ7 if SENSE 2 is depressed; otherwise, will execute next instruction.
NEXT	Ø13	
72ØØ	Ø13 JS1 1 ØØØ	Will cause jump to program memory address 'ØØØ if SENSE 1 is depressed; otherwise, will execute next instruction.
NEXT	Ø14	
E94	Ø14 FPI 4	Will generate the done interrupt to the IOP.
NEXT	Ø15	
4Ø15	Ø15 JUN Ø15	Will halt program.

MACHINE LANGUAGE OPERATING EXAMPLE
TABLE 3-5

Key Entry	Display Response	Comments
LAST	Ø14 FPI 4	Decrements address counter and displays memory contents.
LAST	Ø13 JS1 1 ØØØ	
RESET	ADDR?'ØØØ	Returns to the mode's initial prompt state and sets the address counter to 'ØØØ.
ØØ8	ADDR?'ØØ8	Enters desired address.
NEXT	ØØ8 UNL	Fetches and displays contents of selected addresses.
NEXT	ØØ9 JSR Ø ØØA	Address argument should have been 'ØØ9 in order to halt or wait.
BØ99	ØØ9 JSR Ø Ø99	Still not correct.
CE	ØØ9 JSR Ø ØØA	Disregards previous entry and restores original contents.
BØØ9	ØØ9 JSR Ø ØØ9	
NEXT	ØØA SPE	
MODE	MODE? MACH LANG	The user would now change to the data memory mode to load the transmission data at 'ØØØ-Ø13 and the expected status byte at 'Ø2Ø. Refer to section 3.5 for a description of the data memory mode.

The user may also stop program execution by depressing the RESET key which results in a display of ADDR?'000.

3.4.9 Test Errors

Should the machine language program detect an error, the display will indicate the appropriate message. For example, if the program detected an interface response error while executing the instruction of address '001, the display would indicate HDWR ERROR 001. Additional details of the error condition may then be obtained by entering the error status mode. Refer to section 3.6 for a description of the error status mode.

If the user depresses SINGLE STEP following detection of an error, the Model 488 will continue program execution and will again halt if an error is detected. If the user depresses BYPASS ERROR, the unit will also continue running but will ignore all detected errors. It should be noted that DATA ERRORs and STAT ERRORs continue with the next sequential operation but HDWR ERRORs are referred to as nonrecoverable and automatically restart the program at address '000. Nonrecoverable errors (e.g., DAC @ ATN) are essentailly "fatal" to further bus operation and simple continuance with the next operation would be futile.

3.4.10 Single Instruction Execution

As opposed to running a machine language program, each instruction may be executed manually by depressing the SINGLE STEP key while the unit is in the programming state (i.e., RUN LED off). Each instruction is executed as in the running state. The standard subroutine instructions (op codes '00-'18) execute their complete function for a depression of the SINGLE STEP key and provide the normal error detection. Note that the TRS, RDS, and CPS instructions will still require depression of the SINGLE STEP key to transfer each byte. User written interrupts (FPI instructions) which are located in RAM memory are ignored.

In addition to the above mode which is referred to as Single Instruction Normal (SIN), another mode referred to as Single Instruction Maintenance (SIM) is provided. In the SIM mode, each machine language instruction of the standard subroutine instructions is executed individually as opposed to executing the standard subroutine instructions as macro-instructions. Also, the SINGLE STEP key must be depressed to execute each cycle of the multiple cycle instructions (LDD, LDN, STR, STD). All interrupts are ignored while operating in SIM.

To utilize the SIM mode, the following procedure should be used:

Entry	Display	Comments
MODE	MODE? MACH LAN	
SINGLE STEP	MODE? MACH LANM	Suffix M added to denote SIM.
NEXT	ADDR?'aaa	
NEXT	aaa ___·_____	Depression of SINGLE STEP will execute each machine language instruction/cycle. Interrupts are ignored.
MODE	MODE? MACH LANG	Reselection of the mode prompt state automatically reverts the mode to SIN.

3.4.11 Hex Display of Instructions

Machine language instructions are normally entered in hexadecimal format and displayed in mnemonic format. Entered hex data are further modified or translated prior to storing in the machine language program memory. The translation typically consists of the addition of an implied argument. Under certain operating circumstances, it is necessary to read back the translated hex machine code. Specifically, the stored program mode, card reader, RS-232C interface, and IEEE controller interface require entry in translated machine code.

Entry	Display	Comments
RESET	ADDR?'000	
F	ADDR?'000 HEX	F utilized to select hex format.
aaa	ADDR?'aaa HEX	aaa=desired address. This step is optional.
NEXT	aaa (hex contents)	

The contents of the machine language program memory will remain in the hex display format until the ADDR?'aaa prompt state is re-entered.

3.5 Data Memory Mode Operation

The Model 488's data memory provides 511 memory bytes for data which are to be transmitted, received and recorded, and compared to data received. While data are entered in hexadecimal format, memory contents may be displayed in either hexadecimal or ASCII formats. Data which are recorded from the bus also include the state of the five bus management lines; i.e., EOI,SRQ,REN,ATN,IFC.

3.5.1 Mode Selection

To select the data memory mode, proceed as follows:

Entry	Display	Comments
MODE	MODE? current mode	
5	MODE? DATA MEM	
NEXT	ADDR?'aaa ASCII	Mode LEDS reflect selection of DATA MEM. In addition, one of the operating mode LEDs will be illuminated.

To both select and clear the data memory, proceed as follows:

Entry	Display	Comments
MODE	MODE? current mode	
5	MODE? DATA MEM	
CM	CLEAR DATA MEM	
NEXT	ADDR?'aaa ASCII	Mode LEDs reflect selection of DATA MEM. Subsequent examination of the memory will demonstrate the clear function.

Prior to depressing NEXT, had the user decided he did not wish to clear the memory he could have depressed the CE key to return the display to MODE? current mode.

3.5.2 Reset Function

Depression of the RESET key at any time will cause the following:

a) stop the program if running.
b) clear all bus control lines.
c) preset the data memory address to '000.
d) cause a display of ADDR?'000 ASCII.
e) preset bus language and machine language line number/address to zero.

3.5.3 Data Memory Addressing

Data memory is addressed from '000 to '1FF. The user may use the 511 locations from '000 to '1FE. Address '1FF is unavailable to the user since it is used to pass error data from the bus processor to the input/output processor. Reference Appendix I for a decimal to hex conversion chart.

Following the mode selection sequence or depression of the RESET key, the user is requested to enter the desired memory address: ADDR?'aaa ASCII. After entry of a valid address, depression of the NEXT key causes the selected address and its contents to be displayed. If the address had been invalid, a flashing question mark would be displayed prompting the user to depress the CE key which, in turn, would restore the original address.

3.5.4 Data Display

The contents of data memory are displayed with the appropriate IEEE bus message mnemonic as a prefix. Refer to Appendix F for a glossary of IEEE bus message mnemonics. Data which are stored in data memory as the result of a user's front panel entry are displayed with the mnemonic DAB as a prefix to indicate device dependent data. Likewise, device dependent data which are recorded in data memory from the IEEE bus include DAB as a display prefix. Multiline interface messages (i.e.,commands such as DCL,MLA,etc.) recorded in the data memory as a result of a monitor program are displayed with the appropriate mnemonic. All data recorded in data memory from the IEEE bus also include the binary states of the five bus management lines; i.e., EOI, SRQ, REN, ATN, IFC.

The contents of data memory may be selectively displayed in either an ASCII or hexadecimal format. To select the hexadecimal display format, proceed in a method similar to the following:

Entry	Display	Comments
MODE	ØØA DAB NL	Contents of data memory address '00A is an ASCII NUL.
MODE	MODE? DATA MEM	
NEXT	ADDR?'ØØA ASCII	This prompting state may also be reached by depressing RESET in which case the address will be '000.
F	ADDR?'ØØA HEX	F is utilized to select the hex format. A different address may also be entered prior to depressing NEXT.
NEXT	ØØA DAB'ØØ	

The data memory will remain in the hex display mode until the ADDR?'aaa prompt state is re-entered.

When in the ASCII display mode, hexadecimal characters are displayed when no equivalent ASCII character exists. Recall that the IEEE bus provides an eight bit data bus. If the most significant bit (msb) of a data byte is false (0), the equivalent ASCII character corresponding to the remaining seven bits is displayed. Refer to Appendix H for the ASCII character codes. If the msb of the eight bit byte is true (1), the two equivalent hexadecimal characters are displayed. To avoid confusion between ASCII characters and similar hex characters, the display of all hex data in data memory is prefixed with an apostrophe.

3.5.5 Data Entry and Editing

To add data or to change existing data, the user must enter the necessary hex data. The user entered data temporarily replace the previous contents of the selected address. The appropriate ASCII character is displayed upon entry of the second hex character while in the ASCII display mode. Should the user make an entry error (e.g., too many characters entered), a flashing question mark will be displayed prompting the user to depress the CE key. The CE key restores the original contents of the selected address.

Had the user entered valid data, depression of the NEXT key stores the data as the contents of the selected address. The address is then incremented and displayed along with its contents. The LAST key performs the same function as the NEXT key except that the address is decremented. Use of the NEXT and LAST keys does not necessarily require the user to enter new data; they may be used to simply increment/decrement the address in order to inspect the memory contents.

3.5.6 Operating Example

Tables 3-6 provide an example of setting up the data memory mode and of entering, editing, and examining the contents of data memory.

3.6 Error Status Mode Operation

The Model 488 error status mode provides additional information on detected errors. This subsection discusses the following topics:

Subsection	Title
3.6.1	Operating Sequence
3.6.2	Standard Errors
3.6.3	Common Causes of Errors
3.6.4	User-Written Error Routines

Strictly speaking, the error status mode only provides additional information for errors detected in the monitor mode or errors detected by the machine language standard subroutine instructions. However, the error status mode also applies to bus language programs since the bus language instructions indirectly use the standard subroutine instructions. For the stored program and machine language modes, the error status mode only provides additional information for errors detected by the standard subroutine instructions.

DATA MEMORY OPERATING EXAMPLE
TABLE 3-6

Entry	Display Response	Comments
MODE	MODE? current mode	
5	MODE? DATA MEM	Reflects selection of DATA MEM. Note mode is not changed until NEXT is entered.
CM	CLEAR DATA MEM	Reflects selection of memory clear function.
NEXT	ADDR?'ØØØ ASCII	Mode LEDs reflect selection of DATA MEM.
NEXT	ØØØ DAB NL	Contents of address 'ØØØ is a null byte. NL represents ASCII NUL.
NEXT	ØØ1 DAB NL	Address increments. Contents also null byte.
NEXT	ØØ2 DAB NL	DATA MEM contents reflect clear memory function.
RESET	ADDR?'ØØØ ASCII	Causes display to return to initial prompt state. Address set to 'ØØØ.
NEXT	ØØØ DAB NL	
4	ØØØ DAB'4	Display temporarily reflects hex entry of 4.
1	ØØØ DAB A	Displays ASCII character A corresponding to hex entry of '41.
NEXT	ØØ1 DAB NL	
3	ØØ1 DAB'3	
5	ØØ1 DAB 5	Displays ASCII character 5 corresponding to hex entry of '35.
NEXT	ØØ2 DAB NL	

DATA MEMORY OPERATING EXAMPLE
TABLE 3-6

Entry	Display Response	
61	ØØ2 DAB *A	Displays *A representing ASCII character a corresponding to hex entry of '61.
NEXT	ØØ3 DAB NL	
ØC	ØØ3 DAB FF	Displays ASCII control character FF corresponding to hex entry of 'ØC.
NEXT	ØØ4 DAB NL	
8F	ØØ4 DAB'8F	Displays hex entry for which no equivalent ASCII character exists.
MODE	MODE? DATA MEM	
NEXT	ADDR?'ØØ4 ASCII	
F	ADDR?'ØØ4 HEX	Hex display mode selected.
NEXT	ØØ4 DAB'8F	Fetches contents of specified address and displays contents in hex format.
LAST	ØØ3 DAB'ØC	Decrements address and displays hex contents.
LAST	ØØ2 DAB'61	
RESET	ADDR?'ØØØ ASCII	Display mode reverts to ASCII format.
NEXT	ØØØ DAB A	
2Ø	ØØØ DAB	Displays ASCII character space corresponding to hex entry of '2Ø.
CE	ØØØ DAB A	Display contents returned to original value prior to user entry.

3.6.1 Operating Sequence

A typical error observation sequence is as follows:

Error	Display Response	Comments
____	HDWR ERROR 12	Indicates that a hardware error had been detected while executing the bus language instruction located at line number 12.
MODE	MODE? BUS LANG	
6	MODE? ERR STAT	
NEXT	HDWR ERROR	
NEXT	HANDSHAKE RFD	Indicates that the NRFD line did not sequence correctly during a handshake.
BYPASS ERROR	RUNNING nn	Program is directed to continue and will ignore all detected errors. As an alternative, the user may depress SINGLE STEP to continue the program in which case it will again halt on the next error.

3.6.2 Standard Errors

Appendix E provides a condensed list of the errors detected by the standard subroutine instructions and the monitor mode program. The right-hand columns indicated the applicable subroutines, the halt address, and the restart or continue address. Note that except for HNDSHK TIME-OUT, all HDWR ERRORs restart at program memory address zero and are thus referred to as nonrecoverable. Simple continuance with the next operation following a HDWR ERROR would generally result in the detection of numerous other errors which would tend to mask the original fundamental problem. For example, NDAC @ IFC NATN implies the NDAC line is "stuck" true; continuance would be futile.

The following paragraphs provide additional information on the standard errors.

3.6.2.1 NRFD @ IFC NATN

With ATN false, the NRFD line must be false within 100 usec of setting IFC true. Essentially, this test ensures that the NRFD handshake line is not being held true by a device. This error is nonrecoverable and must be resolved before proceeding with other tests.

3.6.2.2 NDAC @ IFC NATN

The NDAC line must also be false within 100 usec of setting IFC true if ATN is false; otherwise, a device is incorrectly forcing NDAC true. This error is nonrecoverable and must be resolved before proceeding with other tests.

3.6.2.3 DAV @ IFC NATN

The DAV line must be false within 100 usec of setting IFC true if ATN is false; otherwise, a device is incorrectly forcing DAV true. This error is nonrecoverable and must be resolved before proceeding with other tests.

3.6.2.4 DAC @ ATN

The NDAC line is tested for the true state 2 usec after the assertion of the ATN line. This error may be caused simply by no device being connected to the bus or by excessive cabling/propogation delays. This error is nonrecoverable and must be resolved before proceeding with other tests.

3.6.2.5 NO LISTENER

When the controller clears the ATN line in order to transmit device dependent data, a device which has been setup as a listener must initially set not-data-accepted (NDAC true) at the time it indicates ready-for-data (NRFD false). This error is commonly caused by incorrectly establishing a device's bus address. This error is nonrecoverable and must be corrected prior to continuing with other tests.

3.6.2.6 HANDSHAKE RFD

When a controller or talker has data available (DAV true) on the bus during handshake communications, the receiving device(s) must set NRFD true prior to indicating data-accepted (NDAC false). This error is nonrecoverable and must be corrected prior to continuing with other tests.

3.6.2.7 HANDSHAKE DAC

Upon completion of a handshake communicating sequence, the receiving devices must set not-data-accepted (NDAC true) prior to setting the ready-for-data condition (NRFD false). This error is nonrecoverable and must be corrected prior to proceeding with the remaining tests.

3.6.2.8 HANDSHAKE TIME-OUT

This error is detected by the monitor mode program. The NRFD line is expected to be true within one second after the controller or talker set

DAV true; i.e., an error is assumed if the receiving device has not responded within one second to data available. While it is both unusual and undesirable for a device to require such a period of time, it is not prohibited by the IEEE Std 488-1978. To override this error halt, the user need only depress the BYPASS ERROR switch. If the display indicated HDWR ERROR 040, the time-out occurred while IFC was false. If the display indicated HDWR ERROR 028, the time-out occurred while IFC was true. Limited interface message communications are permitted during IFC assertion.

3.6.2.9 aaaIS'dd, SB'dd

This error is detected by the read and compare subroutine instructions and simply states that for data memory address "aaa", the data received is "'dd" and should be "'dd". This error is recoverable and the program will continue with the next normal operation.

3.6.2.10 EARLY END

This error is also detected by the read and compare subroutine instructions. In this case, the user had specified the number of bytes expected to be received and had specified that the END message (END= $EOI \wedge \overline{ATN}$) was expected concurrent with the last byte. The detected error indicates that the END message had been received prior to the expected last byte. This error is recoverable and the program will continue with the next normal operation.

3.6.2.11 NO END

This error is similar to the EARLY END error except that in this case, the END message was not detected upon receipt of the last byte. As before, this is also a recoverable error.

3.6.2.12 IS'dd, SB'dd

This error is detected by the identify subroutine (IDY) and the bus language read status (RS) instruction. As for the data error, it simply states that the received data is "'dd" and should be "'dd". This is also a recoverable error.

3.6.3 Common Causes of Errors

Aside from interface malfunctions or interface design errors, a common cause of difficulties on the IEEE Std 488-1978 bus may be termed bus protocol misunderstandings. For example, several general purpose controller devices imbed various interface messages such as UNT or UNL either before or after a communication sequence. Likewise, these same devices may also imbed terminating characters such as carriage return and/or line feed following a data string. Other devices may or may not be able to respond to the END message. The point is that while none of these protocol variations is unmanageable, they are often not apparent to the user and cause confusion and misunderstanding before the problem is discovered. The Model 488's monitor mode provides the first step in identifying such problems. And while none of these protocol variations is incorrect, an interface designer may not be aware of the consequences thus necessitating the use of the bus language or machine language modes to adequately test his design.

If while running tests on a system interconnected via the IEEE Std 488-1978 bus a user encounters erratic or nonrepeatable operation, experience has indicated several possible solutions. The first is related to grounding problems. The user may wish to attempt both connecting and disconnecting the signal and chassis ground of the various devices. Secondly, while the standard does not restrict cable arrangements, several users have discovered significant differences in operation by altering the cabling arrangement.

3.6.4 User Written Error Routines

As was mentioned previously, the error status mode provides additional information on errors detected by the standard subroutine instructions. The user also has the capability to write his own error routines in the machine language RAM portion of program memory using FPI 0 (HDWR ERROR), FPI 1 (DATA ERROR) and FPI 2 (STAT ERROR). However, since these routines are user generated, the error status mode is unable to automatically display additional information. If the error status mode is selected following detection of an error by a user's error routine, the following typical response results:

Entry	Display Response	Comments
___	HDWR ERROR 04A	
MODE	MODE? MACH LANG	
6	MODE? ERROR STAT	
NEXT	HDWR ERROR	
NEXT	HDWR ERROR	i.e., no response.
SINGLE STEP	RUNNING aaa	The user could also have depressed BYPASS ERROR. Note that all user generated error routines continue with the next operation.

3.7 Stored Program Mode

The stored program mode enables the operator to run one, two, or a series of programs which have been previously written in machine language and stored on EPROMs located on an optional plug-in card of the Model 488. The stored program card may contain up to eight EPROMs. Using 4Kx8 EPROMs, the stored program card would thus provide 32Kx8 of erasable programmable read only memory for storage of user written programs.

3.7.1 Operation

Mode selection and initialization are illustrated below:

Entry	Display Response	Comments
MODE	MODE? current mode	
2	MODE? STD PGM	
NEXT	PGM? nnn mmm	Mode LEDs reflect change to STD PGM mode. nnn mmm are numbers of previously selected programs.

Depression of the RESET key at all times causes the following:

a) stops the program if running.
b) clears all bus lines.
c) displays currently selected program numbers: PGM? nnn mmm.
d) presets the machine language program memory address to '000.
e) presets the data memory address to '000.

The operator is permitted to enter one or two 3 decimal digit program numbers when prompted by PGM?. If two program numbers are entered, they may be selected to either run independently or run as a linked series. If the two numbered programs are to run independently, there is no restriction regarding the relationship of the first number to the second. If the two numbered programs are to be linked, the first program number must be less than the second.

Following selection of the program number(s), the unit may be directed to run the selected program(s). When the RUN key is depressed, the Model 488 first attempts to locate all the specified programs in the stored program EPROM directory. If unable to do so, a flashing question mark is displayed prompting the user to depress CE (clear entry). If the specified programs are located, the Model 488 then loads the machine language program memory and the data memory with the applicable contents of the first program from the stored program EPROMs.

Upon completing the first program (assuming no detection of errors), the Model 488 determines if another program had been requested. If so, the load and run process is performed once more. This process is repeated until all the requested programs have been run, at which time the display will indicate DONE. All functions of the normal machine mode operation are applicable including error detection and the error status mode.

When a selected program is loaded into program and data memories, neither the program nor data memory is first cleared. It is thus possible to load multiple programs concurrently although overwriting may result.

Table 3-7 illustrates operation of the stored program mode. For this example, it is assumed that the optional stored program EPROM card has been installed and contains stored program numbers 900, 901, 902, and 910. It is also assumed for illustration purposes that the Model 488 is connected to the instrument applicable to the referenced stored programs.

TABLE 3-7
STORED PROGRAM MODE OPERATION EXAMPLE

Entry	Display Response	Comments
MODE	MODE? current mode	
2	MODE? STD PGM	
NEXT	PGM? nnn mmm	Mode LEDs reflect change to STD PGM. nnn mmm = currently specified programs.
CE	PGM?	Clears current programs
900	PGM? 900	Enters program number 900.
NEXT	RUN?	Requests run direction.
RUN	RUNNING aaa	Indicates program is running. aaa = sampled program address. This display may exist only momentarily, dependent on the length of the program.
—	DONE	Unit has completed running the program.
RESET	PGM? 900	
CE	PGM?	
910	PGM? 910	
902	PGM? 910 902	Specifies running first program number 910 and then program number 902.
NEXT	RUN?	
RUN	RUNNING aaa	
	DONE	Unit has completed running both programs.
RESET	PGM? 910 902	

TABLE 3-7
STORED PROGRAM MODE OPERATION EXAMPLE

Entry	Display Response	Comments
CE	PGM?	
900	PGM? 900	Enters program number 900 as the first to be run.
C	PGM? 900-	"C" utilized to indicate a linked series of programs.
902	PGM? 900-902	Specifies running programs 900,901, and 902.
NEXT	RUN?	
RUN	RUNNING aaa	
—	DONE	Unit has completed running the three programs.
RESET	PGM? 900-902	
CE	PGM?	
800	PGM? 800	Enters program number 800.
NEXT	PGM? 800 ?	Flashing question mark indicates that program 800 could not be located.
CE	PGM? 900-902	Returns to the last valid program entries.
902	PGM? 902	Enters program number 902.
C	PGM? 902-	Specifies a linked series of programs.
900	PGM? 902-900 ?	Flashing question mark indicates that the second program number is not greater than the first.

3.7.2 Programming

The Model 488 plug-in stored program card may be ordered as option number 488-305 and is supplied without EPROMs. This card is to be installed in card slot A2J5 as indicated on the Model 488 top assembly drawing 1001 1126 contained in Appendix M.

CAUTION

The stored program card must only be installed or removed while power has been removed from the Model 488. Personnel safety considerations recommend that the primary power cord be disconnected from the unit's rear panel while installing or removing the stored program card.

The stored program card is capable of containing eight 24 pin EPROMs as illustrated on the card's assembly and logic diagram (drawing number 1001 1130) contained in Appendix M. The EPROMs are assigned component designations of U1-U8. The stored program may use the following 5 volt EPROM types (450 nsec max access):

a) Intel 2758,	1K x 8
b) Intel 2716,	2K x 8
c) Intel 2732,	4K x 8
d) Texas Inst. 2516,	2K x 8

Component platform U9 must be prepared in accordance with the notes on drawing 1001 1130 to accomodate the various EPROM types. The EPROMs are assigned the address space from '4000 to 'BFFF relative to the Model 488's input/output processor. The following represents the resultant address assignments for each of the EPROMs.

EPROM ADDRESS ASSIGNMENTS

	EPROM TYPE		
EPROM LOCATION	2758	2716 2516	2732
U1	'4000	'4000	'4000
U2	'4400	'4800	'5000
U3	'4800	'5000	'6000
U4	'4C00	'5800	'7000
U5	'5000	'6000	'8000
U6	'5400	'6800	'9000
U7	'5800	'7000	'A000
U8	'5C00	'7800	'B000

The EPROMs must be programmed externally to the Model 488. Table 3-8 summarizes the format of the EPROM contents. The EPROMs contain both a directory of programs and the actual stored program data.

TABLE 3-8
EPROM DATA FORMAT FOR STORED PROGRAMS

	EPROM ADDRESS	EPROM DATA	COMMENTS	
Directory Start	'4000	xn	most significant digit of program number. ØØØ ≤ nnn ≤ 999. x= don't care	
	'4001	nn	least significant digits of program number.	
	'4002	aa	most significant bytes of initial EPROM address corresponding to program number nnn	
	'4003	aa	least significant bytes of initial EPROM address corresponding to program number nnn	
	'4004	xm	stored program number mmm	
	'4005	mm		
	'4006	bb	initial EPROM address of program mmm information	
DIRECTORY	'4007	bb		
	:	:		
	:	:		
	'x-4	xp	stored program number ppp	
	'x-3	pp		
	'x-2	cc	initial EPROM address of program ppp information.	
	'x-1	cc		
Directory End	'x	FF	denotes end of directory	
Data Start	:	:		
	:	:		
	'aaaa		initial pgm memory address (msBs) for std pgm no. nnn	'000-'0FE
	+1		" " " " (lsBs) " " " " "	
	+2		no. of instructions (msBs) of std pgm no. nnn	'0000-'00FE
	+3		" " " (lsBs) " " " " "	if 0000, pgm memory not loaded.
	+4		first instruction msBs	
	+5		" " lsBs	
STORED PGMS	+6		second instruction msBs	
	+7		" " lsBs	
	:		:	
	:		:	
	:		:	

TABLE 3-8
EPROM DATA FORMAT FOR STORED PROGRAMS

	EPROM ADDRESS	EPROM DATA	COMMENTS	
	:			
	'aaaa+z		last instruction's lsBs	
	'wwww		initial data memory address (msBs) for std pgm no. nnn	'000-'1FE
	+1		" " " " (lsBs) " " " " "	
	+2		no. of data bytes (msBs) for std pgm no. nnn	'0000='01FE
	+3		" " " " (lsBs) " " " " "	if 0000,data memory not loaded.
	+4		first data byte	
	+5		second data byte	
	:		:	
	:		:	
	'wwww+y		last data byte	
	:			
	:			
	:			
Data End	'BFFF			

The directory is assumed to start at location '4000; i.e., the EPROM containing the directory must be in location U1. Each stored program requires 4 bytes in the directory. The first two bytes specify the program number while the second two bytes specify the location in the EPROMs where the actual stored program data starts. The programs need not be listed in any particular sequence in the directory. The end of the directory must be identified using the hex byte "FF" as a delimiter. The "FF" byte happens to be the unprogrammed state of the EPROMs. It is recommended the user leave spare locations at the end of his directory in order to add other programs at a later date. Since the directory information specifies the EPROM address to locate the stored program data, EPROMs may be randomly installed in U2-8.

Stored programs are to be written using machine language instructions. The EPROM contents of the stored programs must be programmed in "translated" machine code. Reference Table 3-4 and section 3.4.11. The most convenient method of developing stored programs is to enter the desired machine language program via the front panel controls and then readout the translated machine code in accordance with section 3.4.11.

Each stored program must use the DONE interrupt to indicate program completion. Reference section 3.4.61. The DONE interrupt is used to direct the unit to search for additional required stored programs.

With reference to Table 3-8, the EPROM contents of each stored program is divided into program memory and data memory information. The first two bytes of the program memory information specifies the starting address of program memory where the succeeding instructions are to be stored. The next two bytes specify the number of instructions (not bytes) to be loaded. The remainder of the program memory information is divided into byte pairs corresponding to 16 bit machine language instructions. Likewise, the first two bytes of the data memory information specify the initial data memory address for the succeeding data and the next two bytes specify the number of data bytes.

Recall that when the stored program's EPROM contents are loaded into the program and data memories, neither memory is initially cleared thereby loading concurrent multiple programs. Also, if the specified number of instructions in the EPROM's program memory information is zero, program memory will not be loaded. The same condition is true for the data memory byte count.

IV REMOTE CONTROL OPERATION

This section describes operation of the Model 488 while under remote control from one of the following optional interfaces:

a) card reader

b) IEEE 488-1978

c) RS-232C/TTY

These interfaces provide for various degrees of control of the Model 488. The card reader simply enables the operator to either load the machine language program memory or load the data memory. The IEEE 488-1978 and RS-232C/TTY interfaces permit loading and reading of the memories, starting and stopping program execution, and reading the status of the Model 488.

4.1 Card Reader

The card reader option (no. 488-301) consists of an optical card reader into which pencil marked cards may be inserted to load the Model 488. The card reader option provides a hard copy medium for loading frequently used machine language mode programs.

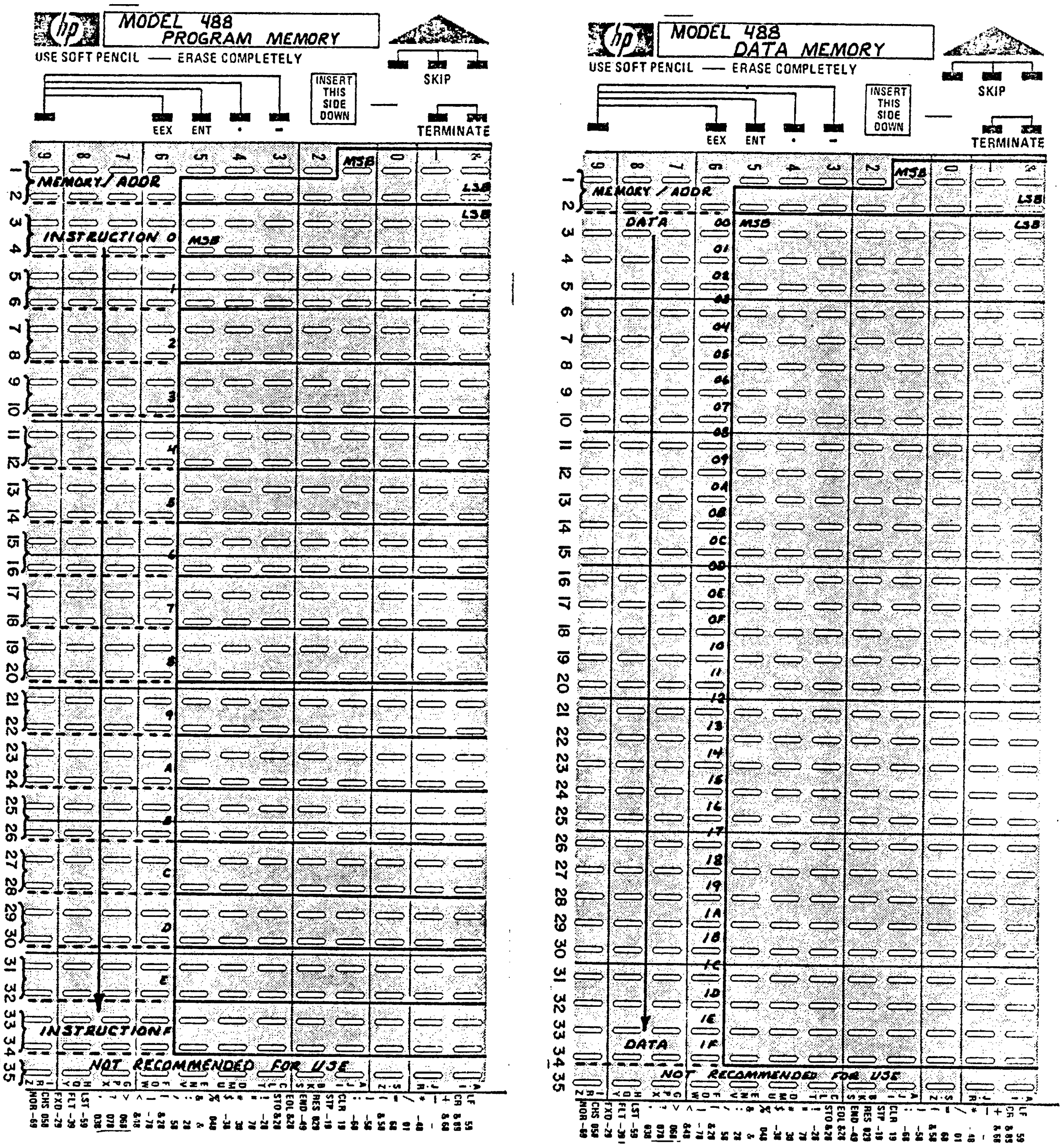
4.1.1 Card Programming

The card reader cards must be marked with binary data corresponding to the "translated" machine language instructions. Reference Table 3-4 and Section 3.4.11. The most convenient method to obtain the translated machine language instructions is to enter the program normally into the Model 488 using the front panel controls and indicators. The program should then be tested for correct operation and for programming errors. Using the procedure explained in Section 3.4.11, the translated machine language instructions may be readout in hexadecimal and used to mark the card reader cards. The contents ofddata memory may also be readout in hex and then used to mark the cards.

Figure 4-1 illustrates the coding formats for the card reader cards. The first field of each card selects which memory is to be loaded and the initial loading address. Each card then causes the loading of either 16 program memory instructions or 32 data memory bytes.

The cards should be marked using a soft lead (#2) pencil. A mark represents a binary one or true state whereas no mark represents a binary zero (false state). Erasures for correction must be done as completely as possible.

FIGURE 4-1
CARD READER CARD FORMATS



MEMORY/ADDRESS	DESTINATION
'000-'0FF	Program RAM, '000-'0FF
'100-'1FF	Program ROM, '100-'1FF
'200-'3FF	Data Memory, '000-'1FF
'400-'FFF	Invalid

NOTE: Marking on cards is for illustration purposes only. Make no marks other than binary data in shaded areas, on front or rear side.

No extraneous marking should be present below the top 7/8 inch of the card on either the front or the rear.

4.1.2 Operation

The card reader connects to the Model 488's designated rear panel connector. Upon connecting the card reader, the Model 488's display will frequently indicate REMOTE CNTRL ERR. The operator should depress the RESET key to clear the error.

Prior to loading cards, the operator should select the machine language mode; i.e.,

Entry	Display Response
MODE	MODE? current mode
4	MODE? MACH LAN
NEXT	ADDR? 'aaa

The cards must be inserted into the card reader with the marked side down. Release the card as soon as the card reader's motor engages the card. Do not push the card through. The Model 488's display will immediately indicate REMOTE CNTRL when a card is inserted.

The Model 488 checks that it receives exactly 35 data strobes corresponding to the data rows on each card. Through prolonged handling and use, it is not unusual for a card to accumulate dirt, smudges and marks which may either mask or add data strobes. If other than 35 strobes are received, the display will indicate REMOTE CNTRL ERR. The error may be cleared by depressing the RESET key.

Once the program and data cards have been loaded, the program may be started using the following procedure:

Entry	Display Response
____	REMOTE CNTRL
RESET	ADDR? '000
RUN	RUNNING aaa

4.2 IEEE Std 488 Controller Interface

The Model 488's IEEE controller interface is available as option number 488-304 and is a factory installed option.

4.2.1 Preparation for Use

Prior to attempting communication with the user's system controller, the desired interface address must be selected by setting five dipswitches on the option plug-in card.

CAUTION

> The Model 488 contains hazardous voltages. Careless probing inside the unit while power is applied may result in the exposure of high voltage terminals.
>
> Do not remove or insert the option card while power is applied to the Model 488. The card contains ICs which may be damaged by voltage transients.

The option card is located in the middle vertical card slot of the lower logic panel assembly (reference drawing number 10011126 of Appendix M). Dipswitches 1-5 are used to set the desired address with number 1 being the lsb (switches 6-8 are unused). Depressing a switch to the "on" position sets the corresponding address bit to a logical one.

The address switches may be changed while power is applied to the unit (reference the previous caution notes). Following changing of the address switches with power applied, the front panel RESET switch must be depressed to complete the address reselection.

Note that the option card must be installed at all times in a Model 488 supplied with the interface option. Such a unit will not operate if the option card is removed.

The Model 488's IEEE controller interface incorporates the following function subsets of IEEE Std 488-1978:

SH1,AH1,T6,L4,SR1,RLØ,PPØ,DCØ,DTØ,CØ
Remote/local, device clear, and device trigger functions are accomodated using data bytes rather than interface commands.

The interface option uses open-collector drivers on all signals.

4.2.2 Operation

All data communication with the Model 488 controller interfaces is accomplished using a limited set of ASCII characters. Hexadecimal data use the ASCII characters 0,1,...9,A,....F. Control of the Model 488 is accomplished using the characters J,K,L,M,R,S,T,U,V,W,Y, and Z. The ASCII comma is used to load data and increment the address. All other transmitted characters are ignored. This enables the user to imbed a carriage return/line feed in a transmitted data table for local page formatting. Likewise, data tables transmitted by the Model 488 include commas and carriage return/line feed.

The controller interfaces permit the user to operate the Model 488 only in the machine language mode. Since the user's controller typically includes a mass storage device (e.g., diskette, cassette, etc.), the monitor mode may be easily emulated. The monitor mode's machine language program (reference Appendix L) need only be duplicated in the controller's mass storage and then loaded into the Model 488 as any other program. Stored programs may be treated in a similar fashion.

All programming of the Model 488 via its controller interfaces is performed in machine language. The machine language instructions must be transmitted in the hexadecimal "translated" format described in section 3.4.11 and itemized in Table 3-4. The most convenient means to obtain the translated machine language instructions is to enter the program normally into the Model 488 using the front panel controls and indicators. The program should then be tested for correct operation and for programming errors. Using the procedure explained in Section 3.4.11, the translated machine language instructions may then be readout in hexadecimal. The user should be aware that the Model 488 does not perform validity testing of instructions received from its controller interfaces.

To load a byte of the Model 488 data memory, two ASCII-equivalent hexadecimal characters must be transmitted. Although the IEEE Std 488 bus is generally used as a 7-bit ASCII bus, it is in fact an eight bit bus. If the Model 488 is being used to test a bus which is limited to ASCII characters, the hexadecimal equivalent of each ASCII character must be loaded into its data memory; e.g., to transmit the character A on the test bus, the Model 488's data memory must be loaded with 41.

The ASCII comma is used during loading of program and data memories to load the previous data and increment the current address. The input data buffer is also cleared upon receipt of a comma. Characters are loaded into the input buffer's least significant digit and shifted left upon receipt of succeeding characters. Excessive characters are simply shifted left and dropped.

The receipt of any character (valid or not) excent Z causes the Model 488 front panel display to indicate REMOTE CNTRL. Detection of an invalid transmission seauence causes the disnlay to indicate REMOTE CNTRL ERR. The error display may be cleared by transmitting R or Z or depressing the front panel RESET key.

Table 4-1 provides a summary of the protocol for the Model 488 controller interfaces and is applicable to both the IEEE STD 488 and RS-232C/TTY interfaces. Table 4-2 provides a detailed description of the controller interfaces' protocol.

TABLE 4-1
PROTOCOL SUMMARY FOR CONTROLLER INTERFACES

Function	Input/Output	Format
RESET	I	R
Load Program Memory △7	I	Jaaa,dddd,dddd,.....,dddd,
Read Program Memory △7	I	Kaaa,nnn,
	0	dddd,dddd,.....,dddd,CRLF
Load Data Memory △7	I	Laaa,dd,dd,.....,dd
Read Data Memory △7	I	Maaa,nnn,
	0	dddd,dddd,.....,dddd,CRLF
BYPASS ERROR	I	Ud,
SENSE 1	I	Vd,
SENSE 2	I	Wd,
START/STOP	I	S
STEP	I	T
Read Status Byte △6	I	X
	0	dCRLF
Read Unit Status	I	Y
	0	ddddddddddddCRLF
Return to Local Control	I	Z

Notes:

1. All characters are ASCII. "aaa", "nnn", and "dddd" are ASCII equivalent of hexadecimal data.
2. "aaa" = address.
3. "d" = data.
4. "nnn" = number of instructions or bytes.
5. commas shown as inputs must be included.

△6 RS-232C/TTY controller interface only.

△7 must not be transmitted to the Model 488 while it is running a program.

TABLE 4-2
DETAILED PROTOCOL DESCRIPTION
FOR CONTROLLER INTERFACES

RESET

Control: R

Causes the following:
(a) stops the program if running.
(b) clears all bus control lines.
(c) presets the machine language memory address to '000.
(d) presets the data memory address to '000.

Load Program Memory

Control/Data: Jaaa,dddd,dddd,.....,dddd,

J: load program memory control character.
aaa: initial address (hex) to be loaded.
'000 ≤ 'aaa ≤ '0FE = program RAM
('100 ≤ 'aaa ≤ '1FF = program ROM)
dddd: translated instruction machine code (hex).
,: comma; used for control of load and address increment.

Read Program Memory

Control: Kaaa, nnn,

Response Data: dddd,dddd,.....,dddd,$C_R L_F$

K: read program memory control character.
aaa: initial address (hex) to be read from.
'000 ≤ 'aaa ≤ '0FE = program RAM
('100 ≤ 'aaa ≤ '1FF = program ROM)
nnn: number (hex) of instructions to be read.
dddd: translated instruction machine code (hex).
,: comma; used to load and as instruction delimiter.
$C_R L_F$: carriage return, line feed; signifies transmission complete.

Load Data Memory

Control/Data: Laaa,dd,dd,.....,dd,

L: load data memory control character.
aaa: initial address (hex) to be loaded, '000 ≤ 'aaa ≤ '1FE.
dd: data byte (hex) to be loaded.
,: comma; used to load and increment address.

Table 4-2
Continued

Read Data Memory

Control: Maaa, nnn,

Response Data: dddd,dddd,.....,ddddC_RL_F

M: read data memory control character.
aaa: initial address (hex) to be read from, '000 ≤ 'aaa ≤ '1FE.
nnn: number (hex) of data bytes to be read.
dddd: hex data; first two characters represent the auxiliary data byte, second two characters represent the data byte itself; auxiliary byte format:

b7	b6	b5	b4	b3	b2	b1	b0
↓	DAV	0	EOI	SRQ	REN	ATN	IFC

0 : contents from Model 488
1 : contents recorded from bus

, : comma, used to load and as data delimiter.
C_RL_F: carriage return, line feed; signifies transmission complete.

BYPASS ERROR

Control/Data: Ud,

U: BYPASS ERROR control character.
d: data; 0 = no error bypass, 1 = error bypass.
,: comma; used to load data.

SENSE 1

Control/Data: Vd,

V: SENSE 1 control character.
d: data; 0 = not depressed, 1 = depressed
,: comma; used to load data.

SENSE 2

Control/Data: Wd,

W: SENSE 2 control character.
d: data; 0 = not depressed, 1 = depressed.
,: comma; used to load data.

START/STOP

Control: S

Causes unit to change state between programming and running.

STEP

Control: T

Causes SINGLE STEP in response to step interrupt.

Table 4-2
Continued

Read Status Byte

(a) IEEE Controller Interface

Serial Poll Status Byte:

```
b7 b6 b5 b4  b3 b2 b1 b0
0  |  0  0   0  0  |  |
```

b6 → Interrupt Status
0 = no interrupt
1 = interrupt pending

b0 → Stopped/Running Status
0 = stopped
1 = running

b1 → Controller Interface Status
0 = no error
1 = error

When an interrupt is pending, Model 488 will wait until it is serviced; e.g., stepped, reset, stopped, etc.

Controller interface error automatically causes the interface to be re-initialized. The detected error concerns data transmission sequencing and not invalid characters (which are ignored).

(b) RS-232C/TTY Controller Interface

Control: X

Response Data: d$C_R L_F$

d : ASCII encoded status
0 = stopped, no interrupt pending
1 = running, no interrupt pending
A = running (waiting), interrupt pending

Read Unit Status

Control : Y

Response Data: dddddddddddd$C_R L_F$

Y : read unit status control character
d....d: status data
d1 d2 d3 = program memory address
d4 = interrupt class
0 = HDWR ERROR
1 = DATA ERROR
2 = STATUS ERROR
3 = invalid
4 = DONE
5 = STEP
6-E = undefined
F = no interrupt
d5 = interrupt type; DONE, STEP and no interrupt are type X; reference Appendix E for types corresponding to classes 0,1,and 2.

Table 4-2
Continued

d4/d5

	Class/Type		
	1/0	2/0	All Others
d6 =	data mem addr msb	space	space
d7 =	data mem addr	space	space
d8 =	data mem addr lsb	space	space
d9 =	"IS" msb	"IS" msb	space
d10=	"IS" lsb	"IS" lsb	space
d11=	"SB" msb	"SB" msb	space
d12=	"SB" lsb	"SB" lsb	space

Return to Local Control

Control: Z

Causes unit to return to local control/display. Resets unit to MACH LANG mode with a display of ADDR?'000.

4.2.3 Operating Example

Table 4-3 provides an example of remote programming the Model 488. The resultant program performs the following:

a) clears the interface
b) clears all devices
c) transmits a data table to device '1E
d) triggers device '1E to execute the task
e) waits for service request
f) reads the serial poll status byte of device '1E
g) if SENSE 2 is set, retriggers the device
h) if SENSE 1 is set, re-initiates the program; otherwise, halts indicating DONE.

Following the last step of the example, which starts the test program, the controller should periodically sample the Model 488's status byte, testing for a pending interrupt. Once it has been determined an interrupt is pending, the unit status (Y) should be requested. Dependent on the interrupt class (Yd4), subsequent action may be taken. For example, if the interrupt was due to a detected error (classes 0,1,2), the Model 488 may be directed to step (T) past it in order to search for other errors, or it may be directed to ignore all errors (U1,).

If the status byte fails to indicate a pending interrupt following a reasonable period of time, it is advisable to repeatedly request the unit status (Y). If the program address remains static, it indicates the Model 488 is waiting, for example, for a response from the unit under test. If the Model 488 is not waiting, the program memory address returned by the Y function will be random; in fact, the indicated address may be invalid since it may be sampled while it is changing.

TABLE 4-3
CONTROLLER INTERFACE
PROGRAMMING EXAMPLE

Data	Comments
R	resets unit.
J000,	sets up to load program memory starting at address '000.
0100,	IFC @ '000 : clear interface.
0214,	DCL @ '001 : clear all devices.
023E,	MLA '1E @ '002 : set up device '1E as a listener.
2000,	LDD 000 @ '003 : fetch contents of data memory address '000.
35F0,	LC2 010 @ '004 : define number of data bytes, 16.
EA00,	FTF 0 @ '005 : disable END transmission.
116E,	TRR @ '006 : transmit data table.
0208,	GET @ '007 : trigger device '1E.
023F,	UNL @ '008 : disable listeners.
B009,	JSR 0 009 @ '009 : wait for service request
0218,	SPE @ '00A : enable serial poll.
025E,	MTA '1E @ '00B : set up device '1E as talker.
2020,	LDD 020 @ '00C : fetch contents of data memory address '020.
3C7F,	LDM 7F @ '00D : set mask.
16CD,	CPB @ '00E : compare serial poll status byte.
025F,	UNT @ '00F : clear all talkers.
0219,	SPD @ '010 : disable serial poll.
023E,	MLA '1E @ '011 : set up device '1E as a listener.
7A07,	JS2 T 007 @ '012: if SS2 set, retrigger.
7200,	JS1 T 000 @ '013: if SS1 set, restart.
E904,	FPI 4 @ '014 : DONE interrupt.
4015,	JUN 015 : halt.
L000,	sets up to load data memory starting at address '000.
30,	ASCII 0 @ '000 : start of transmitted table
31,	1 @ '001
32,	2 @ '002
33,	3 @ '003
34,	4 @ '004
35,	5 @ '005
36,	6 @ '006
37,	7 @ '007
38,	8 @ '008
39,	9 @ '009
41,	A @ '00A
42,	B @ '00B
43,	C @ '00C
44,	D @ '00D
45,	E @ '00E
46,	F @ '00F : End of transmitted table.
L020,	sets up to load data memory at address '020
41,	A @ '020
K000,003,	sets up to read 3 instructions starting at pgm. addr. '000.

Table 4-3
Continued

Data	Comments
0100,0214,0235,	
M000,005,	sets up to read 5 bytes starting at data addr. '000.
30,31,32,33,34,	
R	resets unit; sets pgm and data addr '000.
U0,	de-activates BYPASS ERROR
V1,	sets SENSE 1
W0,	de-activates SENSE 2
S	starts Model 488 test program

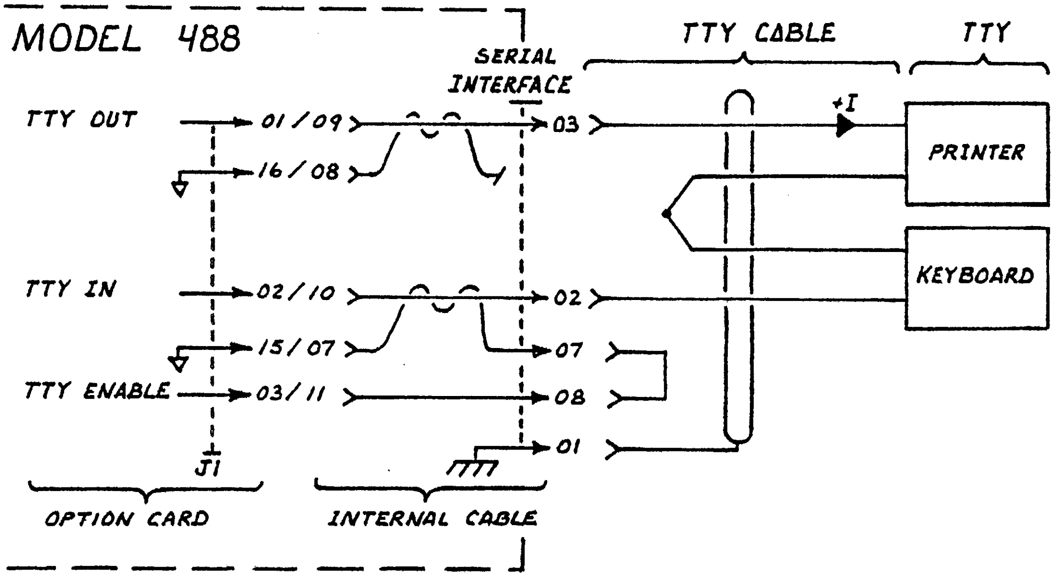
4.3 RS-232C/TTY Controller Interface

The Model 488's RS-232C/TTY controller interface is available as option number 488-303 and is a factory installed option.

4.3.1. Preparation for Use

The Model 488's rear panel connector, designated SERIAL INTERFACE, is configured to connect to the cable of an RS-232C terminal. Alternately, the Model 488 may be used as an RS-232C terminal by connecting the supplied terminal cable to the SERIAL INTERFACE connector and connecting the other end to the user's calculator or computer. The supplied terminal cable is end interchangeable.

A teletype (TTY) may also be connected to the SERIAL INTERFACE connector. The following diagram illustrates the TTY cable construction:



Note that the internal Model 488 cable which plugs into the option card must be rotated 180 degrees for TTY operation such that pin 09 of the cable plugs into pin 01 of card socket J1.

CAUTION
The Model 488 contains hazardous voltages. Careless probing inside the unit while power is applied may result in the exposure of high voltage terminals.

Do not remove or insert the option card while power is applied to the Model 488. The card contains ICs which may be damaged by voltage transients.

The RS-232C/TTY option card is located in the left-most vertical card slot of the lower logic panel assembly when viewed from the front (reference drawing number 10011126 of Appendix M). Prior to attempting communication, the transmission baud rate must be selected by setting the rotary switch on the option plug-in card. The correspondence between baud rate and the switch positions is as follows:

Switch Position	Baud Rate	No. of Stop Bits
0	110	2
1	150	1
2	300	1
3	600	1
4	1200	1
5	2400	1
6-9	invalid	-

The baud rate may be changed while power is applied (reference the previous caution notes). Following re-selection of the baud rate while power is applied, depress the front panel RESET switch to re-initialize the interface.

The Model 488's RS-232C/TTY interface does not echo transmit the received characters and therefore would normally be connected to a half-duplex terminal. The interface option does not delay when switching between the transmit and receive modes.

The RS-232C/TTY option card must be installed at all times in a Model 488 supplied with the interface option. Such a unit will not operate if the option card is removed.

The serial data format consists of the following:

(a) one start bit
(b) seven data bits, lsb first
(c) parity bit: transmitted false; received "don't care".
(d) one stop bit (other than 110 baud) or two stop bits at 110 baud.

The RS-232C interface option is a data channel implementation only. When connected to a terminal, CTS, DSR, and DCD are transmitted continuously true while the RTS and DTR inputs are not used. When connected as a terminal using the supplied cable, DTR is transmitted continuously true and RTS is open while the CTS, DSR, and DCD inputs are not used.

The TTY interface option is a 20 mA, neutral, interface, to be operated in half-duplex.

4.3.2 Operation

Refer to sections 4.2.2 and 4.2.3 for a description of the controller interface operation and an operating example

V THEORY OF OPERATION

This section provides information on the theory of operation of the Model 488. The monitor, stored program, and bus language modes are all executed in reality in the machine language mode. It is the intent of this section to illustrate the interrelationship of the modes to each other and to the hardware implementation.

The following topics are discussed:

a) Monitor Mode

b) Bus Language Mode

c) Standard Subroutines

d) Processor Description

5.1 Monitor Mode

This section provides a brief description of the standard machine language program for the monitor mode. This discussion is presented, first, so that the user is aware of the program design as it relates to its conventional use and, secondly, so that he may develop his own monitor-type program to meet his unique requirements.

When the monitor mode is directed to run, the equivalent machine language program is loaded into the bus processor program memory and the data memory is cleared. Appendix L contains the monitor program flowcharts. The user may examine this program by completing the monitor mode set up and initialization and, instead of depressing RUN, selecting the machine language mode.

The program is essentially grouped into three parts:

1) initialization, '000-'00C.
2) handshake communication while IFC is true, '00D-'02E
3) handshake communication while IFC is false, '02F-'046.

The instructions of addresses '003-'006 are configured by the input/output processor dependent on the selected trigger condition. Triggering may occur on two basic transaction types; i.e., assertion of IFC or handshake communications (DAV,DAB,MLA,MTA). During initialization, the program waits until DAV is false. NDAC is then set true to delay completion of handshakes, while NRFD remains false (i.e., RFD always true).

Aside from IFC and DAV triggering, each handshake byte is compared to the selected trigger byte and, if equal, is recorded. All succeeding transactions are also recorded until the specified number has been received.

During handshake communications, the loop counters are used to timeout a one second period while DAV is true. Approximately one second after DAV is set, an error will be indicated if no device has responded by setting NRFD true. As noted previously, this is not strictly an error on the IEEE 488-1978 bus, although it may generally be assumed that something is amiss. Depression of the BYPASS ERROR switch causes the error to be ignored.

Each transaction requires between 3 and 5 usec to be processed dependent upon whether the program is searching for the trigger condition (≠: 3 usec; =: 5 usec), is recording data while IFC is true (5usec), or is recording data while IFC is false (4 usec). The transaction processing period is, of course, also dependent on the system cabling, controller, and other devices.

5.2 Bus Language

The Model 488 bus language mode enables the user to prepare a program using simplified bus-oriented instructions. Each bus language instruction is translated into a fixed set of machine language instructions. Appendix K contains the machine language flowchart for each bus language instruction. The Model 488 input/output processor inserts the required arguments into each of the machine language instructions based upon the user-entered arguments of the bus language instructions.

As may be observed in Appendix K, the bus language instructions use both the standard subroutine and the conventional machine language instructions. The first three instructions (SAV, FPI, JUN) of each bus language instruction may be referred to as overhead. They provide no useful processing and are used only to indicate to the input/output processor which bus language instruction is currently being executed while in the run mode.

Following entry of a bus language program, the user may examine the equivalent machine language program by simply changing the mode to machine language. The equivalent program may then also be run (the bus language line number interrupts are ignored while in the machine language mode). The user may add additional machine language instructions to the equivalent program. When the mode is changed to machine language, a bus language instruction which specified an invalid reference to a non-existent line number is assembled as a jump-to-self. This may be used to advantage to change the jump-to-self instead to a jump to an address where additional machine language instructions may be entered. If the mode is returned to bus language, all modifications to the equivalent machine language program are lost

5.3 Standard Subroutines

This section describes details of the machine language standard ROM subroutines. This discussion is intended to acquaint the user with the test philosophy and techniques used in the standard subroutines. An understanding of the standard subroutines should permit the user to develop his own RAM-resident machine language subroutines in an analogous manner.

Appendix J contains the flowcharts for the standard subroutines. The standard subroutines are located in ROM at address '100-'1FF of the bus processor memory (also referred to as the program ROM memory). The standard subroutines are the following:

a) IFC : interface clear subroutine @ '120.
b) MIM : multiline interface message (i.e., command) subroutine @ '140.
c) IDY : identify subroutine @ '11E.
d) T : Data transmit subroutine @ '168.
e) R : data read/record subroutine @ '198.
f) C : Data read/compare subroutine @ '1C8.

5.3.1 Special Op Codes for the Standard Subroutines

The op codes of the first 25 machine language instructions ('00-'18) are dedicated to standard subroutine usage only. Reference Table 3-4. The following discussion relates to these special op codes and is provided only for completeness of understanding. The user is not able to modify these op codes in any way.

The instructions with entry op codes '00-'0F are related to the instructions stored at locations '100-'103, referred to as the BP ROM pointers. Upon entry of an instruction with op code '00-'0F, the Model 488 input/output processor translates the op code and appends the necessary argument. For example, a DCL is entered as a '01 and is translated to '0214. When executed by the bus processor, the translated instruction is processed as a jump-to-subroutine at address '102 and loads the output register with '14 (i.e., DCL multiline message). The BP ROM pointer at address '102 is an unconditional jump to the beginning of the MIM subroutine at address '140. Op codes '02-'0E are processed identically to op code '01. Op code '00 is translated to '0100 and jumps to the IFC subroutine at '120 via address '101. Op code '0Fpp is translated to '03pp and jumps to the IDY subroutine at '11E via address '103.

The BP ROM pointer at '100 is used as a halt to prevent program "runaway" when execution is attempted of a null instruction. Recall that the bus processor program RAM memory is cleared on power up and may be cleared using the CM key. If a user's program inadvertently jumps to an address which has been cleared (i.e., null instruction), and then executes the contents, it is processed as an unconditional jump to '100 in which a halt is stored. If the program is then stopped, the address containing the null instruction is displayed.

Instructions with entry op codes of '10-'18 are also translated by the input/output processor; however, an argument is not appended. Arguments for these instructions must be previously defined by other machine language instructions. For example, the RDR instruction is entered as '14 and is translated to '149E. The bus processor executes the translated instruction as jump-to-subroutine at address '19E, the entry point for the RDR instruction into the R subroutine.

5.3.2 Characteristics of the Standard Subroutines

The standard subroutine flowcharts of Appendix J identify the following:

aaa ERROR = error detection instructions

$\boxed{E}$ = error halts

$\boxed{W}$ = bus wait states

The error halts are the locations at which the bus processor halts following sending an error interrupt to the input/output processor. Reference Appendix E for summary listing of the standard subroutine errors. As opposed to error halts in RAM which restart at the next address, the input/output processor performs a table look-up to determine where to restart the bus processor standard subroutine instructions.

The Model 488 standard subroutines do not implement time-out errors for lack of response or activity on the bus. The bus wait states identified on the flowcharts are the address at which the bus processor delays awaiting a response from the bus. Table 5-1 provides a summary listing of the bus wait states.

The IFC subroutine is expected to be called by each user program in order to initialize the bus and to take control. Upon exitting the IFC subroutine, ATN is asserted to take control and REN is asserted to enable establishment of device remote control. The first time a device is set up as a listener (MLAaa) it will also be set into the remote control state if the device's interface implements the RL function. The REN line is not reset by any standard subroutine but may be cleared by the user's program (FRE 0). The GTL and LLO messages are available for use in accordance with the IEEE std 488-1978.

To relinquish control of the bus (i.e., ATN false) in order to allow two devices to independently communicate, the user's program need only execute FAT 0. To regain control, the user's program should call either the MIM or IDY subroutines wherein ATN is synchronously asserted.

The following paragraphs describe the characteristics of each of the standard subroutines.

5.3.2.1 IFC, Interface Clear

User-defined arguments:
None

Status of control lines and internal flags:
Uses LC3 internally as delay counter.
Exits with ATN and REN asserted; all others false.

Rates:
Not applicable since no data are transferred.
Assuming no errors, IFC instruction executes in 104 usec.

TABLE 5-1
STANDARD SUBROUTINE BUS WAIT STATES

Subroutine Address	Subroutine	Description
'108	IDY	Wait for current talker to set DAV false prior to taking control (i.e., setting ATN true).
'145	MIM	Wait for current talker to set DAV false prior to taking control (i.e., setting ATN true).
'14F	MIM	Wait for all devices to indicate ready-for-data (NRFD false) prior to sending command.
'151	MIM	Wait for all devices to indicate data-accepted (NDAC false) of received command.
'155	MIM	Wait for all devices to again indicate ready-for-data (NRFD false) following acceptance of command.
'173	T	Wait for all listeners to initially indicate ready-for-data (NRFD false) prior to sending first byte.
'179	T	Wait for all listeners to indicate ready-for-data (NRFD false) for succeeding data bytes.
'17C	T	Wait for all listeners to indicate data-accepted (NDAC false) of transmitted data byte.
'182	T	Wait for a listener to indicate not-data-accepted (NDAC true) at completion of data transmission.
'1A5	R	Wait for talker to indicate data-available (DAV true) to receive data byte.
'1AB	R	Wait for talker to indicate not-data-available (DAV false) following receipt of data byte.
'1B1	R	Wait for talker to indicate not-data-available (DAV false) following receipt of last data byte.
'1D3	C	Wait for talker to indicate data-available (DAV true) to receive data byte.
'1D9	C	Wait for talker to indicate not-data-available (DAV false) following receipt of data byte.
'1E0	C	Wait for talker to indicate not-data-available (DAV false) following receipt of last data byte.

Description:

Clears all flags and then sets IFC line.
Delays for 100 usec and tests status of NRFD,NDAC,DAV.
Clears IFC, sets ATN and REN.
Delays 2 usec and tests status of NDAC.

5.3.2.2 MIM, Multiline Interface Message

User-defined arguments:

MLA,MTA,SCG,IDY: inherent in instruction code.
All others: no argument

Status of control lines and internal flags:

Uses LC3 internally as a delay counter.
Synchronously sets ATN and REN if ATN false upon entry.
Exits with ATN and REN asserted, DAV false, output data disabled; all others unaltered.

Rates:

Assuming ATN asserted on entry, zero response time of bus devices, and no errors, instruction execution requires 4.6 usec of which 3.2 usecs is for the handshake sequence. Includes 2 usec delay for data stabilization.

Description:

Synchronously sets ATN and REN if ATN false upon entry.
Delays 2 usec for data stabilization.
Performs handshake transfer.

5.3.2.3 IDY, Identify

User-defined arguments:

Bus devices configured and enabled for parallel poll. Ref 3.4.6h.
Comparator mask enabled (LDM).
Save expected parallel poll response (SAV) for IOP readback.
Expected parallel poll response inherent to instruction code.

Status of control line and internal flags:

Uses LC3 internally as a delay counter.
Synchronously sets ATN and REN if ATN false upon entry.
Exits with ATN and REN asserted, end/identify enable flag false; all others unaltered.

Rates:

Assuming ATN asserted on entry and no errors, instruction requires 4.8 usecs of which 2.0 usec are for stabilization of IDY message.

Description:

Synchronously sets ATN and REN if ATN false upon entry.
Delays 2 usec for IDY message stabilization.
Compares parallel poll data (no handshake).

5.3.2.4 T, Data Transmit

User-defined arguments:

Fetch first data byte (LDD; TRB may also use LDI).
Number of bytes to be transferred (LC2; unnecessary for TRB).
Selection of END message transmission (FTF b0); optional.

Status of control line and internal flags:

LC2 used as byte counter
LC3 used internally as delay counter
FTF b0 set externally to select END
FTF b1 used internally for single step
Exits with ATN asserted; DAV cleared; FTF b0 and b1, end/identify enable, and output enable cleared; others unaffected.

Rates:

Assuming no errors and zero response time of bus devices, handshake sequence requires 4.0 usec. Includes 2.0 usec for data stabilization.

Description:

Clears ATN and checks that NDAC true when NRFD false.
Delays 2 usec for data stabilization.
Performs handshake transfer.
Repeats handshake transfer with 2 usec delay if necessary.
Reasserts ATN to regain control of bus after all transfers complete.

5.3.2.5 R, Data Read/Record

User-defined arguments:

First recording address (LDD).
Number of bytes to be recorded (LC2; unnecessary for RDB) or record until END received (FTF b0).

Status of control line and internal flags:

LC2 used as byte counter.
LC3 used internally as delay counter.
FTF b0 set externally to select recording to END.
FTF b1 used internally for single step.
Exits with ATN asserted; NRFD and NDAC cleared; FTF b0 and b1 cleared; others unaffected.

Rates:

Assuming no errors and zero response time of bus devices, nandshake sequence requires 3.4 usecs.

Description:

Sets NRFD and NDAC and clears ATN.
Clears NRFD and performs handshake sequence.
Stores data in data memory.
Repeats handshake sequence if necessary.
Reasserts ATN and clears NDAC when transfers complete.
Delays 2 usec.
Clears NRFD.

5.3.2.6 C, Data Read/Compare

User-defined arguments:

First comparison address (LDD).
Number of bytes to be compared (LC2; unnecessary for CPB).
Comparator enable mask (LDM).
Test for END (FTF b0); optional.

Status of control line and internal flags:

LC2 used as byte counter.
LC3 used internally as delay counter.
FTF b0 set externally to test for END.
FTF b1 used internally for single step.
Exits with ATN asserted; NRFD and NDAC cleared; FTF b0 and b1 cleared; all others unaffected.

Rates:

Assuming no errors and zero response time of bus devices, handshake sequence requires 2.8 usec.

Description:

Sets NRFD and NDAC and clears ATN.
Clears NRFD and performs handshake sequence.
Compares received data to data memory contents.
Repeats handshake sequence if necessary.
Reasserts ATN and clears NDAC when transfers complete.
Delays 2 usec.
Clears NRFD.

5.4 Processor Description

The Model 488's functional block diagram is illustrated in Figure 5-1. The detailed logic diagrams are contained in Appendix M.

The Model 488 is composed of two primary functional elements: the input/output processor (IOP) and the bus processor (BP). The BP is the high speed processor which accomplishes the bus testing and includes the functional elements illustrated in Figure 5-1. The BP contains a 1Kx16 RAM and a 256x16 ROM for storage of program instructions and data. Using an addressing mapping scheme, the ROM effectively overlaps the second quarter of the RAM memory. The overlaid portion of RAM (designated auxiliary BL RAM) is unavailable to the user and is utilized by the IOP to store the instructions of a bus language program. Figure 5-2 illustrates the user-accessible bus processor program/data memory.

The basic operation of the BP consists of fetching an instruction from the memory and loading it into the instruction register. The instruction register contents are then used to generate the control microcode which, in turn, directs the bit slice sequence to define the next memory address. In addition to memory address sequencing, the microcode generates the necessary controls to carry out the instruction's functions with respect to the bus control network. Except for those noted in Table 3-4, the bus processor executes instructions in one cycle of 200 nsecs. The bit slice sequencer provides four levels of subroutines of which one is reserved for storage of the data memory address leaving the remaining three available to the user.

The input/output processor controls all communications between the front panel switches and indicators, controller interfaces (e.g., RS-232C, card reader, controller IEEE), and the bus processor. The IOP loads the bus processor memory in accordance with the user's entries. The IOP then directs the BP to execute the program. While the BP is running, the IOP remains passive and does not interfere or impede the BP execution rate. When the BP either completes its program or detects an error, it generates an interrupt to request IOP intervention. The IOP may be directed to abort the BP program by depression of either RESET OR RUN/STOP.

As is illustrated in Figure 5-1, the IOP controls the BP through a PIA. When the IOP takes control of the BP, it may then directly access the BP memory and various BP registers. Once the input/output processor relinquishes control and starts the BP, the IOP may only monitor certain BP status registers until interrupted.

interface TECHNOLOGY

FIGURE 5-1
MODEL 488
FUNCTIONAL BLOCK DIAGRAM

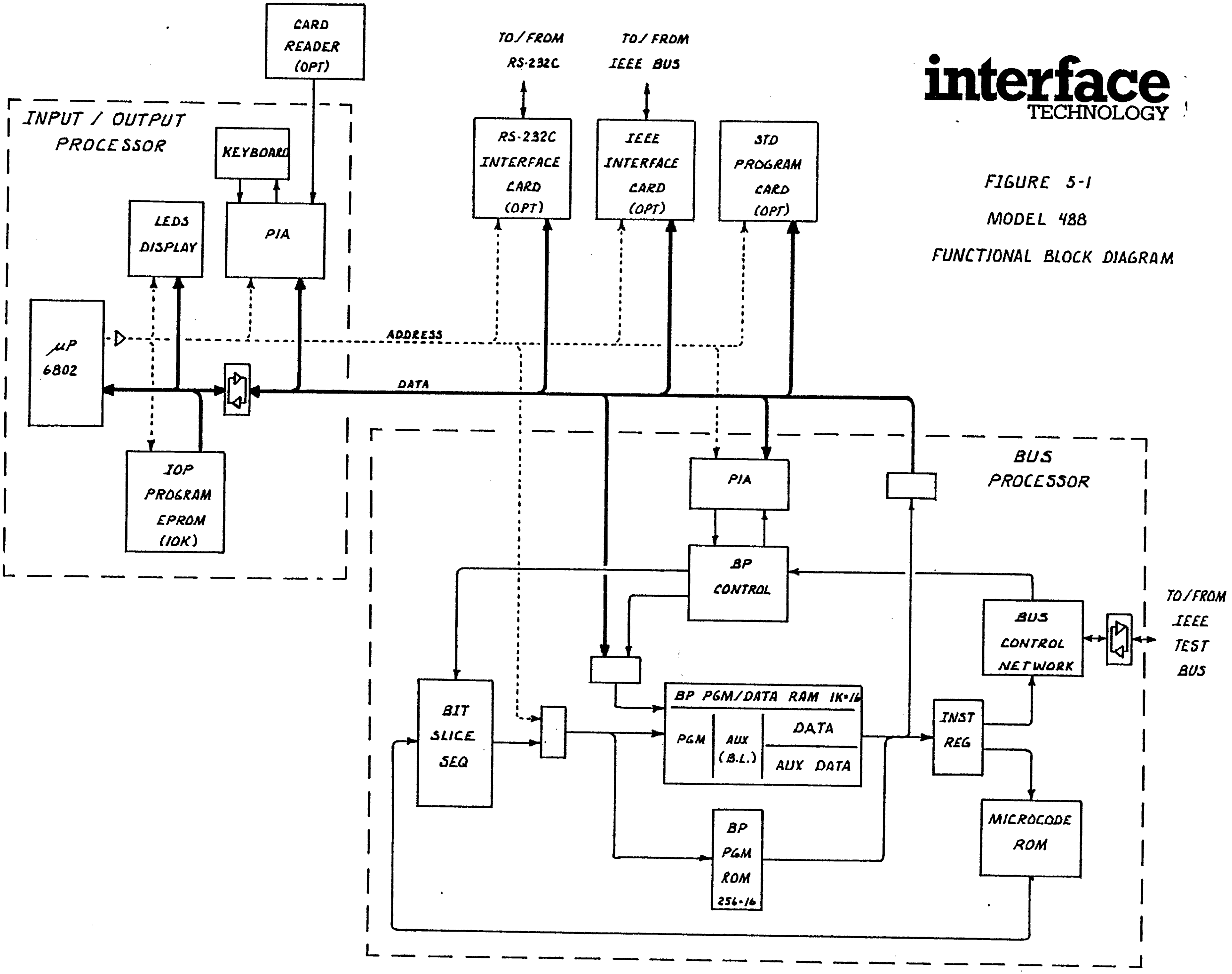
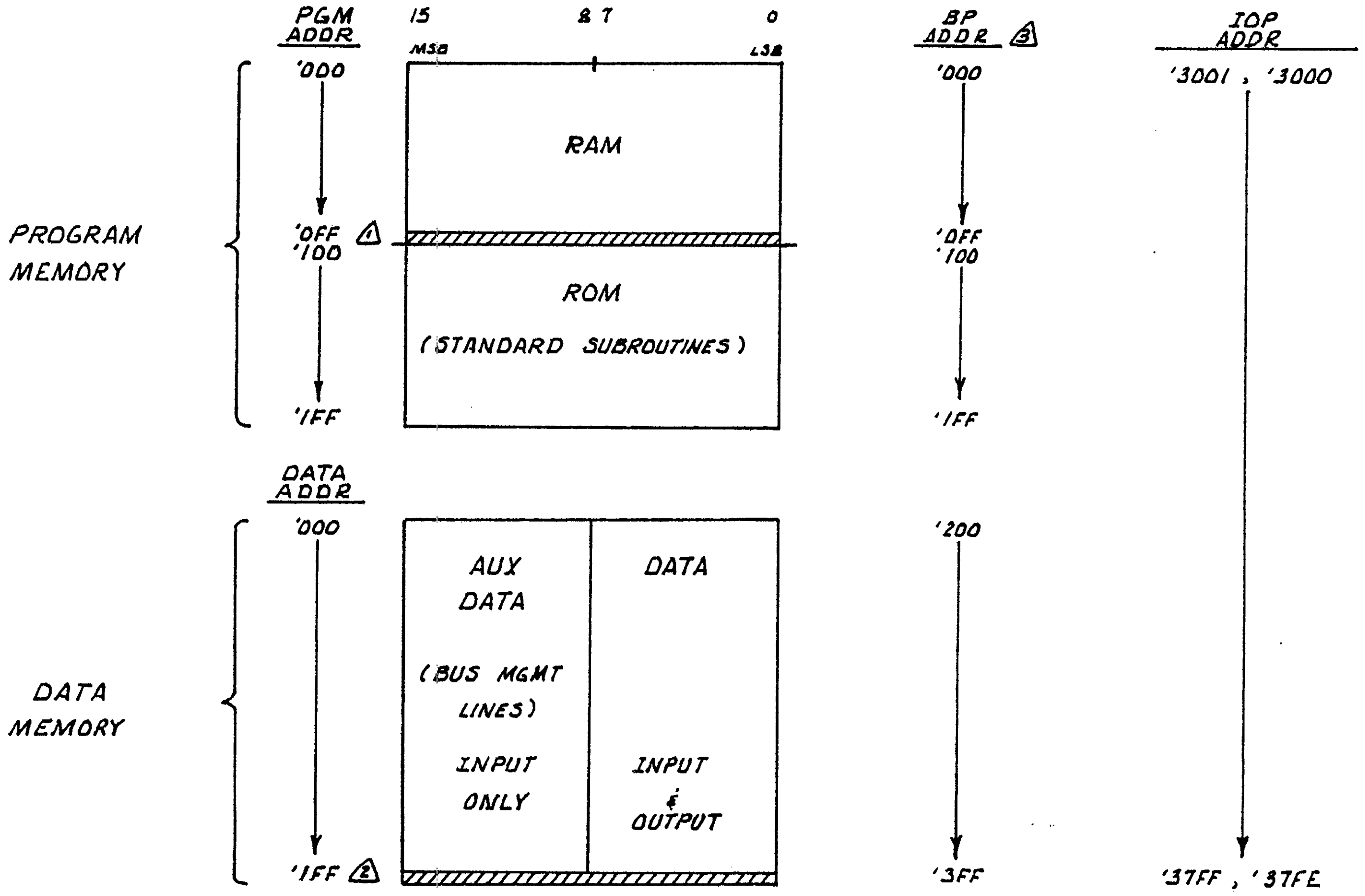


FIGURE 5-2

BUS PROCESSOR PROGRAM / DATA MEMORY



△1 UNAVAILABLE TO USER ; USED BY IOP TO JUMP TO FETCH ADDRESS.

△2 UNAVAILBLE TO USER ; STORES BUS DATA FOR IOP FETCH.

△3 INSTRUCTION OP-CODE APPENDS ADDRESS B9.

APPENDICIES

APPENDIX A
MONITOR MODE OPERATING SUMMARY

The following table illustrates operation of the Model 488 monitor mode. The procedure assumes it is desired to initiate or trigger the recording upon detection of the interface message which sets up the device with hexadecimal address '1E as a talker (ASCII talk address character ^). It also assumes a total of 17 transactions are to be recorded. Note that the recorded data is identical to that of the monitor mode demonstration program. Reference Table 3-1 for the demonstration program procedure.

Key Entry	Display Response	Comments
MODE	MODE? current mode	
1	MODE? MONITOR	
NEXT	TRIG? XXX'xx x	Previous trigger byte, if any.
CE	TRIG? 0-4,D	0 = IFC; interface clear 1 = DAV; data available. First handshake after RUN. 2 = DAB'dd c; data byte. '00 ≤ 'dd ≤ 'FF. c=ASCII equiv. of 'dd. Reference Appendix H. 3 = MLA'dd c; my listen address. '00 ≤ 'dd ≤ '1E. Reference Appendix G. 4 = MTA'dd c; my talk address. '00 ≤ 'dd ≤ '1E. Reference Appendix G. D = MTA'1E D; demo program. Fixed arguments.
4	TRIG? MTA'00 @	
1	TRIG? MTA'1	
E	TRIG? MTA'1E ^	
NEXT	XFRS? xxx	Number of transactions to be recorded. 3 decimal digits, 001 to 511.
0	XFRS? 0	
1	XFRS? 01	
7	XFRS? 017	
NEXT	RUN?	
RUN	RUNNING aaa	Running display continues until either: a) error detected: displays HDWR ERROR. Reference Appendix E. b) trigger byte and specified number of transactions recorded: displays DONE.
—	DONE	
MODE	MODE? MONITOR	
5	MODE? DATA MEM	
NEXT	ADDR?'000 ASCII	
NEXT	000 MTA ^ 00110	Trigger byte is recorded in data memory address '000. Data are displayed in ASCII format. State of 5 bus management lines also recorded.
NEXT	001 DAB 0 00100	ASCII zero.
NEXT	002 DAB 9 00100	ASCII 9.

APPENDIX A
MONITOR MODE OPERATING SUMMARY

Key Entry	Display Response	Comments
NEXT	003 DAB 2 00100	ASCII 2.
NEXT	004 DAB 3 00100	ASCII 3.
NEXT	005 DAB 4 00100	ASCII 4.
NEXT	006 DAB CR 00100	ASCII carriage return.
NEXT	007 DAB LF 10100	ASCII line feed. End $(=EOI \wedge \overline{ATN})$ message sent.
NEXT	008 IFC 00111	Interface clear.
NEXT	009 DCL 00110	Device clear.
NEXT	00A UNT 00110	Disables current talker.
NEXT	00B UNL 00110	Disables current listeners.
NEXT	00C MTA U 00110	Device with hex address '15 set up as talker.
NEXT	00D MLA > 00110	Device with hex address '1E set up as listener.
NEXT	00E DAB R 00100	ASCII R.
NEXT	00F UNT 00110	Disables current talker.
NEXT	010 UNL 00110	Disables current listener.
NEXT	011 DAB NL	Lack of bus management line status indicates preceding address contained last recorded data.
RESET	ADDR?'000 ASCII	
F	ADDR?'000 HEX	F utilized to select hex display format.
0	ADDR?'0 HEX	
0	ADDR?'00 HEX	
6	ADDR?'006 HEX	
NEXT	006 DAB'0D 00100	Hex equivalent of carriage return
NEXT	007 DAB'0A 10100	Hex equivalent of line feed.

APPENDIX B
MODEL 488
BUS LANGUAGE INSTRUCTION SUMMARY

Keyboard Entry — Description

CL

Clear interface and all devices.

WT'xx'aaa nnnE

Write data block to device 'xx.*
- 'xx: device's listen address, '00 ≤ 'xx ≤ '1E.
- 'aaa: data memory address of first byte, '000 ≤ 'aaa ≤ '1FE.
- nnn: number of data bytes, 001 ≤ nnn ≤ 511.
- E: transmit END message (optional).

RR'xx'aaa nnn

Read and record data block received from device 'xx.*
- 'xx: device's talk address, '00 ≤ 'xx ≤ '1E.
- 'aaa: data memory address of first storage location, '000 ≤ 'aaa ≤ '1FE.
- nnn: number of bytes to be recorded, 001 ≤ nnn ≤ 511. If E is entered, recording will continue until either an END message is received or data memory is full (511 bytes maximum).

RC'xx'aaa nnnE

Read and compare data block received from device'xx.* If the received data does not compare to the corresponding data stored in data memory, the program will halt and display an error message.
- 'xx: device's talk address, '00 ≤ 'xx ≤ '1E.
- 'aaa: data memory address of first comparison location, '000 ≤ 'aaa ≤ '1FE.
- nnn: number of bytes to be compared, 001 ≤ nnn ≤ 511.
- E: test for END message (optional).

TR'xx

Trigger device 'xx.
- 'xx: device's listen address, '00 ≤ 'xx ≤ '1E.

SR

Wait for service request (SRQ).

RS'xx'mm'ss

Read and compare serial poll status byte from device 'xx. If the received status byte does not compare to the expected byte, the program will halt and display an error message.
- 'xx: device's talk address, '00 ≤ 'xx ≤ '1E.
- 'mm: mask to enable/disable (1/0) comparison of selected bits, '00 ≤ 'xx ≤ 'FF,
- 'ss: expected status byte.

JS t nn

Jump to instruction line number nn if SENSE SWITCH 2 is in selected position; otherwise, continue to the next line number.
- t: SENSE SWITCH 2; 0=not depressed (switch out), 1=depressed (switch in).
- nn: instruction line number (decimal).

JU nn

Jump unconditionally to line number nn.
- nn: instruction line number (decimal).

NOTES

1. All hexadecimal fields are preceded by an apostrophe. All other fields are decimal.
2. Refer to Appendix G for bus address code conversion.

*3. If SENSE SWITCH 1 is depressed, one byte will be transmitted, recorded, or compared for each depression of the SINGLE STEP switch.

*4. aaa + nnn must be less than 511, the maximum data memory address.

APPENDIX C
MODEL 488
MACHINE LANGUAGE INSTRUCTION SUMMARY

Mnemonic	Entry	Comments
STANDARD		SUBROUTINES
IFC	00	△1
DCL	01	
SDC	02	
MLA aa	03aa	00 ≤ aa ≤ 1E
UNL	04	
MTA aa	05aa	00 ≤ aa ≤ 1E
UNT	06	
SCG ss	07ss	00 ≤ ss ≤ 1F
LLO	08	
GTL	09	
GET	0A	
SPE	0B	
SPD	0C	
PPC	0D	
PPU	0E	
IDY pp	0Fpp	00 ≤ pp ≤ FF
TRB	10	
TRR	11	
TRS	12	
RDB	13	
RDR	14	
RDS	15	
CPB	16	
CPR	17	
CPS	18	△1

CONDITIONAL JUMP INSTRUCTIONS BITS 8,9,10		
b 10 9 8	Hex	
F 0 aa	0/8	00≤aa≤FF
F 1 aa	1/9	
T 0 aa	2/A	
T 1 aa	3/B	
S F 0 aa	4/C	
S F 1 aa	5/D	
S T 0 aa	6/E	
S T 1 aa	7/F	00≤aa≤FF

Mnemonic	Entry	Comments	
LDD 0aa	20aa	00 ≤ aa ≤ FF	△2
LDD 1aa	21aa	00 ≤ aa ≤ FE	△2
LDN	24		△3
LDI dd	38dd	00 ≤ dd ≤ FF	
LDM mm	3Cmm	00 ≤ mm ≤ FF	
STR	28		△3
STD 0aa	2Caa	00 ≤ aa ≤ FF	△3
STD 1aa	2Daa	00 ≤ aa ≤ FF	△3
SAV zz	3Ezz	00 ≤ zz ≤ FF	
FPI 0	E90	HDWR ERROR int	
FPI 1	E91	DATA ERROR int	
FPI 2	E92	STAT ERROR int	
FPI 3	E93	LINE NO int	
FPI 4	E94	DONE int	
FPI 5	E95	STEP int	
JUN 0aa	40aa	00 ≤ aa ≤ FF	
JUN 1aa	41aa	00 ≤ aa ≤ FF	
JUN S 0aa	44aa	00 ≤ aa ≤ FF	
JUN S 1aa	45aa	00 ≤ aa ≤ FF	
JRN	48		
JS1 F 0aa	70aa	SENSE 1	
|	|		
JS1 S T 1aa	77aa		
JS2 F 0aa	78aa	SENSE 2	
|	|		
JS2 S T 1aa	7Faa		
JBE F 0aa	80aa	BYPASS ERROR	
|	|		
JBE S T 1aa	87aa		

△1 calls subroutine - multiple cycles
△2 2 cycles
△3 3 cycles

APPENDIX C
MODEL 488
MACHINE LANGUAGE INSTRUCTION SUMMARY

Mnemonic	Entry	Comments
LC0 cc	30cc	00 ≤ cc ≤ FF
JL0 F 0aa	50aa	
│	│	
JL0 S T 1aa	57aa	
LC1 cc	32cc	00 ≤ cc ≤ FF
JL1 F 0aa	58aa	
│	│	
JL1 S T 1aa	5Faa	
LC2 0cc	34cc	00 ≤ cc ≤ FF [4]
LC2 1cc	35cc	00 ≤ cc ≤ FF
JL2 F 0aa	60aa	
│	│	
JL2 S T 1aa	67aa	
LC3 cc	36cc	00 ≤ cc ≤ FF [5]
JL3 F 0aa	68aa	
│	│	
JL3 S T 1aa	6Faa	
FCL	EB	Clear bus lines
FTF f	EAf	f=0-3
JFE F 0aa	88aa	
│	│	FTF b0 [6]
JFE S T 1aa	8Faa	
JTF F 0aa	D8aa	
│	│	FTF b1 [7]
JTF S T 1aa	DFaa	
FAT f	E0f	ATN
JAT F 0aa	90aa	
│	│	
JAT S T 1aa	97aa	
FRE f	E1f	REN
JRE F 0aa	98aa	
│	│	
JRE S T 1aa	9Faa	

Mnemonic	Entry	Comments
FIF f	E2f	IFC
JIF F 0aa	A0aa	
│	│	
JIF S T 1aa	A7aa	
FEI f	E3f	EOI Enable
JEI F 0aa	A8aa	
│	│	
JEI S T 1aa	AFaa	
FSR f	E4f	SRQ
JSR F 0aa	B0aa	
│	│	
JSR S T 1aa	B7aa	
FDV f	E5f	DAV
JDV F 0aa	B8aa	
│	│	
JDV S T 1aa	BFaa	
FNR f	E6f	NRFD
JNR F 0aa	C0aa	
│	│	
JNR S T 1aa	C7aa	
FND f	E7f	NDAC
JND F 0aa	C8aa	
│	│	
JND S T 1aa	CFaa	
JCE F 0aa	D0aa	Compare
│	│	
JCE S T 1aa	D7aa	
FOT f	E8f	Output Enable

[4] used in standard subroutines as the byte counter
[5] used in standard subroutines as a delay counter
[6] used in standard subroutines to enable generation/detection of END
[7] used in standard subroutines to enable single byte transfers

APPENDIX D
STORED PROGRAM OPERATING SUMMARY

Step	Comments	Entry	Display Response
1	Install appropriate stored program printed circuit card in Model 488.		
1a	Depress POWER switch to off position.........	POWER	
1b	Remove AC power cord from rear of unit.		
1c	Remove top cover.		
1d	Install card in right-most card slot when viewing Model 488 from front. Card should be inserted such that components are on left.		
1e	Re-install top cover and AC power cord.		
1f	Depress POWER switch to on position	POWER	MODE? MONITOR
2	Connect unit under test to Model 488's rear panel connector designated TEST INTERFACE using an IEEE Std 488-1978 compatible cable.		
3	Select stored program mode	2	MODE? STD PGM
	Note: Operator entry errors are indicated by a flashing question mark. Depress CE to clear the entry.		
4	Depress NEXT to enter the stored program mode.. Mode LEDs reflect change to stored program mode.	NEXT	PGM?
5	Enter first program number as three decimal digits. For example	900	PGM? 900
	If no other program desired, go to step 9.		
	If one other independent program desired, go to step 6.		
	If a series of linked programs are desired, go to step 7.		
6	Enter second program number. For example	078	PGM? 900 078
	Go to step 9.		
7	Depress "C" key to indicate a linked series ...	C	PGM? 900-
8	Enter the number of the last program in the series. Entered number must be greater than the first program number. For example	902	PGM? 900-902
9	Depress NEXT key	NEXT	RUN?

APPENDIX D
STORED PROGRAM OPERATING SUMMARY

Step	Comments	Entry	Display Response
10	When ready, direct the test program(s) to start	RUN	RUNNING aaa
	If the Model 488 is unable to locate the specified program(s) in the EPROMs of the stored program card, depress the CE key and return to step 1 to verify the correct card has been plugged in.		
	The display will indicate DONE if all programs have completed execution.		
	Proceed with step 11 if DONE is indicated.		
	Proceed with step 13 if an error is indicated.		
11	Setup to execute the next program(s)	RESET	PGM? nnn mmm
	nnn mmm indicates previously selected program(s).		
12	Clear previously selected program(s)	CE	PGM?
	Go to step 5.		
13	If an error is detected, the display will indicate an error message. For example		HDWR ERROR 017
	The "017" indicates the instruction number at which the error was detected.		
14	Select ERR STAT mode........................	MODE	MODE? STD PGM
		6	MODE? ERR STAT
15	Depress NEXT to determine additional details of error. For example	NEXT	HDWR ERROR
		NEXT	DAC @ ATN
	Reference Appendix E for a complete listing of standard error messages.		
	Depress SINGLE STEP to step past the current error and continue program execution, or ...		
	depress BYPASS ERROR to ignore all errors and continue program.		
	Note that HDWR ERRORs are continued at the beginning of the program (i.e., restarted) and must be corrected prior to conducting other tests. Use BYPASS ERROR to aid in troubleshooting with an oscilloscope.		

APPENDIX D
STORED PROGRAM OPERATING SUMMARY

Step	Comments	Entry	Display Response
	DATA ERRORs and STAT ERRORs continue with the next operation. Use SINGLE STEP to identify all such errors and then BYPASS ERROR to use an oscilloscope.		

APPENDIX E
CONDENSED STANDARD ERROR LIST

04 , 08

MODE DISPLAY [1]	ERR STAT DISPLAY	COMMENTS	SUBR	PMA HALT	PMA RTN [2]	CLASS/TYPE NO. [3]
HDWR ERROR Oaa	NRFD @ IFC NATN	NRFD should be false if IFC true and ATN false.	IFC	132	000	0/1
	NDAC @ IFC NATN	NDAC should be false if IFC true and ATN false.	IFC	135	000	0/2
	DAV @ IFC NATN	DAV should be false if IFC true and ATN false.	IFC	138	000	0/3
	DAC @ ATN	NDAC should be true within 2 us of ATN setting.	IFC MIM IDY	13B 15A 119	000 000 000	0/4 0/5 0/0
	NO LISTENER	No device setup as listener. NDAC should be true.	T	18E	000	0/8
	HANDSHAKE RFD	NRFD should be true when NDAC false in handshake.	MIM T	15D 194	000 000	0/6 0/A
	HANDSHAKE DAC	NDAC should be true when NRFD false in handshake.	MIM T	160 191	000 000	0/7 0/9
	HNDSHK TIME-OUT	NRFD not set within 1 second of DAV.	M M	028 040	029 041	0/B 0/C
DATA ERROR Oaa	aaa IS'dd, SB'dd	aaa = data memory address.	C	1ED	1D6	1/0
	EARLY END	END (=EOI $\wedge$ $\overline{\text{ATN}}$) true before last byte.	C	1F0	1D8	1/1
	NO END	END not received with last byte.	C	1F3	1DF	1/2
STAT ERROR Oaa	IS'dd, SB'dd		IDY	11D	113	2/0

[1]: program memory address (PMA) for monitor, stored pgm, and mach language modes. Line number for bus language

[2]: if PMA RTN address = 000, program restarts at address 000; i.e., nonrecøverable error.

[3]: reference remote control interface unit status data

APPENDIX F
IEEE STD 488-1978
MESSAGE GLOSSARY

Mnemonic	Title	Comment
ACG	Addressed Command Group	Device must be addressed to use command.
ATN	Attention	UCG (uniline).
DAB	Data Byte	Device dependent data (ATN false).
DAC	Data Accepted	Handshake control. Complement of NDAC.
DAV	Data Available	Handshake control.
DCL	Device Clear	UCG ('14).
END	End of Transmission	Status. $\text{END} = \text{EOI} \wedge \overline{\text{ATN}}$.
GET	Group Execute Trigger	ACG ('08).
GTL	Go to Local	ACG ('01).
IDY	Identify	UCG (uniline). $\text{IDY} = \text{EOI} \wedge \text{ATN}$.
IFC	Interface Clear	UCG (uniline).
LLO	Local Lockout	UCG ('11).
MLA	My Listen Address	Listen address group (multiline).
MSA	My Secondary Address	SCG (multiline).
MTA	My Talk Address	Talk address group (multiline).
NDAC	Not Data Accepted	Handshake control.
NRFD	Not Ready for Data	Handshake control.
PPC	Parallel Poll Configure	ACG ('05).
PPD	Parallel Poll Disable	SCG (multiline).
PPE	Parallel Poll Enable	SCG (multiline).
PPRn	Parallel Poll Response	Status.
PPU	Parallel Poll Unconfigure	UCG ('15).
REN	Remote Enable	UCG (uniline).
RFD	Ready for Data	Handshake control. Complement of NRFD.
RQS	Request Status	Status. B7 of STE.
SCG	Secondary Command Group	PPE, PPD, MSA.
SDC	Selective Device Clear	ACG ('04).
SPD	Serial Poll Disable	UCG ('19).
SPE	Serial Poll Enable	UCG ('18).
SRQ	Service Request	Status (uniline).
STB	Status Byte	Status (multiline).
TCT	Take Control	ACG ('09).
UCG	Universal Command Group	Device need not be addressed to use command.
UNL	Unlisten	Listen address group ('3F).
UNT	Untalk	Talk address group ('5F).

APPENDIX G
BUS ADDRESS CONVERSION TABLE

Address Switch Settings					Decimal	ASCII		Model 488
					Address	Address Characters		Hex
5	4	3	2	1	Equiv	Listen	Talk	Entry
Ø	Ø	Ø	Ø	Ø	ØØ	SP	@	'ØØ
Ø	Ø	Ø	Ø	1	Ø1	!	A	'Ø1
Ø	Ø	Ø	1	Ø	Ø2	"	B	'Ø2
Ø	Ø	Ø	1	1	Ø3	#	C	'Ø3
Ø	Ø	1	Ø	Ø	Ø4	$	D	'Ø4
Ø	Ø	1	Ø	1	Ø5	%	E	'Ø5
Ø	Ø	1	1	Ø	Ø6	&	F	'Ø6
Ø	Ø	1	1	1	Ø7	'	G	'Ø7
Ø	1	Ø	Ø	Ø	Ø8	(	H	'Ø8
Ø	1	Ø	Ø	1	Ø9	)	I	'Ø9
Ø	1	Ø	1	Ø	1Ø	*	J	'ØA
Ø	1	Ø	1	1	11	+	K	'ØB
Ø	1	1	Ø	Ø	12	,	L	'ØC
Ø	1	1	Ø	1	13	-	M	'ØD
Ø	1	1	1	Ø	14	.	N	'ØE
Ø	1	1	1	1	15	/	O	'ØF
1	Ø	Ø	Ø	Ø	16	Ø	P	'1Ø
1	Ø	Ø	Ø	1	17	1	Q	'11
1	Ø	Ø	1	Ø	18	2	R	'12
1	Ø	Ø	1	1	19	3	S	'13
1	Ø	1	Ø	Ø	2Ø	4	T	'14
1	Ø	1	Ø	1	21	5	U	'15
1	Ø	1	1	Ø	22	6	V	'16
1	Ø	1	1	1	23	7	W	'17
1	1	Ø	Ø	Ø	24	8	X	'18
1	1	Ø	Ø	1	25	9	Y	'19
1	1	Ø	1	Ø	26	:	Z	'1A
1	1	Ø	1	1	27	;	[	'1B
1	1	1	Ø	Ø	28	<	\	'1C
1	1	1	Ø	1	29	=	]	'1D
1	1	1	1	Ø	3Ø	>	^	'1E
1	1	1	1	1	31 Invalid	? UNL	_ UNT	'1F Invalid

APPENDIX H
ASCII CHARACTER CODES

ASCII Char	Display Char	Hex Code
NUL	NL	'ØØ
SOH	SH	'Ø1
STX	SX	'Ø2
ETX	EX	'Ø3
EOT	ET	'Ø4
ENQ	EQ	'Ø5
ACK	AK	'Ø6
BEL	BL	'Ø7
BS	BS	'Ø8
HT	HT	'Ø9
LF	LF	'ØA
VT	VT	'ØB
FF	FF	'ØC
CR	CR	'ØD
SO	SO	'ØE
SI	SI	'ØF
DLE	DE	'1Ø
DC1	D1	'11
DC2	D2	'12
DC3	D3	'13
DC4	D4	'14
NAK	NK	'15
SYN	SN	'16
ETB	EB	'17
CAN	CN	'18
EM	EM	'19
SUB	SB	'1A
ESC	EC	'1B
FS	FS	'1C
GS	GS	'1D
RS	RS	'1E
US	US	'1F

ASCII Char	Display Char	Hex Code
SP		'2Ø
!	!	'21
"	"	'22
#	#	'23
$	$	'24
%	%	'25
&	&	'26
'	'	'27
(	(	'28
)	)	'29
*	*	'2A
+	+	'2B
,	,	'2C
-	-	'2D
.	.	'2E
/	/	'2F
Ø	0	'3Ø
1	1	'31
2	2	'32
3	3	'33
4	4	'34
5	5	'35
6	6	'36
7	7	'37
8	8	'38
9	9	'39
:	:	'3A
;	;	'3B
<	<	'3C
=	=	'3D
>	>	'3E
?	?	'3F

APPENDIX H
ASCII CHARACTER CODES

ASCII Char	Display Char	Hex Code
@	@	'4Ø
A	A	'41
B	B	'42
C	C	'43
D	D	'44
E	E	'45
F	F	'46
G	G	'47
H	H	'48
I	I	'49
J	J	'4A
K	K	'4B
L	L	'4C
M	M	'4D
N	N	'4E
O	O	'4F
P	P	'5Ø
Q	Q	'51
R	R	'52
S	S	'53
T	T	'54
U	U	'55
V	V	'56
W	W	'57
X	X	'58
Y	Y	'59
Z	Z	'5A
[	[	'5B
\	\	'5C
]	]	'5D
^	^	'5E
_	_	'5F

ASCII Char	Display Char	Hex Code
`	*@	'6Ø
a	*A	'61
b	*B	'62
c	*C	'63
d	*D	'64
e	*E	'65
f	*F	'66
g	*G	'67
h	*H	'68
i	*I	'69
j	*J	'6A
k	*K	'6B
l	*L	'6C
m	*M	'6D
n	*N	'6E
o	*O	'6F
p	*P	'7Ø
q	*Q	'71
r	*R	'72
s	*S	'73
t	*T	'74
u	*U	'75
v	*V	'76
w	*W	'77
x	*X	'78
y	*Y	'79
z	*Z	'7A
{	*[	'7B
¦	*\	'7C
}	*]	'7D
~	*^	'7E
DEL	*_	'7F

APPENDIX I

DECIMAL/HEX CONVERSION TABLE
000-255 DECIMAL
000-OFF HEXADECIMAL

DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX
000	000	032	020	064	040	096	060	128	080	160	0A0	192	0C0	224	0E0
001	001	033	021	065	041	097	061	129	081	161	0A1	193	0C1	225	0E1
002	002	034	022	066	042	098	062	130	082	162	0A2	194	0C2	226	0E2
003	003	035	023	067	043	099	063	131	083	163	0A3	195	0C3	227	0E3
004	004	036	024	068	044	100	064	132	084	164	0A4	196	0C4	228	0E4
005	005	037	025	069	045	101	065	133	085	165	0A5	197	0C5	229	0E5
006	006	038	026	070	046	102	066	134	086	166	0A6	198	0C6	230	0E6
007	007	039	027	071	047	103	067	135	087	167	0A7	199	0C7	231	0E7
008	008	040	028	072	048	104	068	136	088	168	0A8	200	0C8	232	0E8
009	009	041	029	073	049	105	069	137	089	169	0A9	201	0C9	233	0E9
010	00A	042	02A	074	04A	106	06A	138	08A	170	0AA	202	0CA	234	0EA
011	00B	043	02B	075	04B	107	06B	139	08B	171	0AB	203	0CB	235	0EB
012	00C	044	02C	076	04C	108	06C	140	08C	172	0AC	204	0CC	236	0EC
013	00D	045	02D	077	04D	109	06D	141	08D	173	0AD	205	0CD	237	0ED
014	00E	046	02E	078	04E	110	06E	142	08E	174	0AE	206	0CE	238	0EE
015	00F	047	02F	079	04F	111	06F	143	08F	175	0AF	207	0CF	239	0EF
016	010	048	030	080	050	112	070	144	090	176	0B0	208	0D0	240	0F0
017	011	049	031	081	051	113	071	145	091	177	0B1	209	0D1	241	0F1
018	012	050	032	082	052	114	072	146	092	178	0B2	210	0D2	242	0F2
019	013	051	033	083	053	115	073	147	093	179	0B3	211	0D3	243	0F3
020	014	052	034	084	054	116	074	148	094	180	0B4	212	0D4	244	0F4
021	015	053	035	085	055	117	075	149	095	181	0B5	213	0D5	245	0F5
022	016	054	036	086	056	118	076	150	096	182	0B6	214	0D6	246	0F6
023	017	055	037	087	057	119	077	151	097	183	0B7	215	0D7	247	0F7
024	018	056	038	088	058	120	078	152	098	184	0B8	216	0D8	248	0F8
025	019	057	039	089	059	121	079	153	099	185	0B9	217	0D9	249	0F9
026	01A	058	03A	090	05A	122	07A	154	09A	186	0BA	218	0DA	250	0FA
027	01B	059	03B	091	05B	123	07B	155	09B	187	0BB	219	0DB	251	0FB
028	01C	060	03C	092	05C	124	07C	156	09C	188	0BC	220	0DC	252	0FC
029	01D	061	03D	093	05D	125	07D	157	09D	189	0BD	221	0DD	253	0FD
030	01E	062	03E	094	05E	126	07E	158	09E	190	0BE	222	0DE	254	0FE
031	01F	063	03F	095	05F	127	07F	159	09F	191	0BF	223	0DF	255	0FF

APPENDIX I

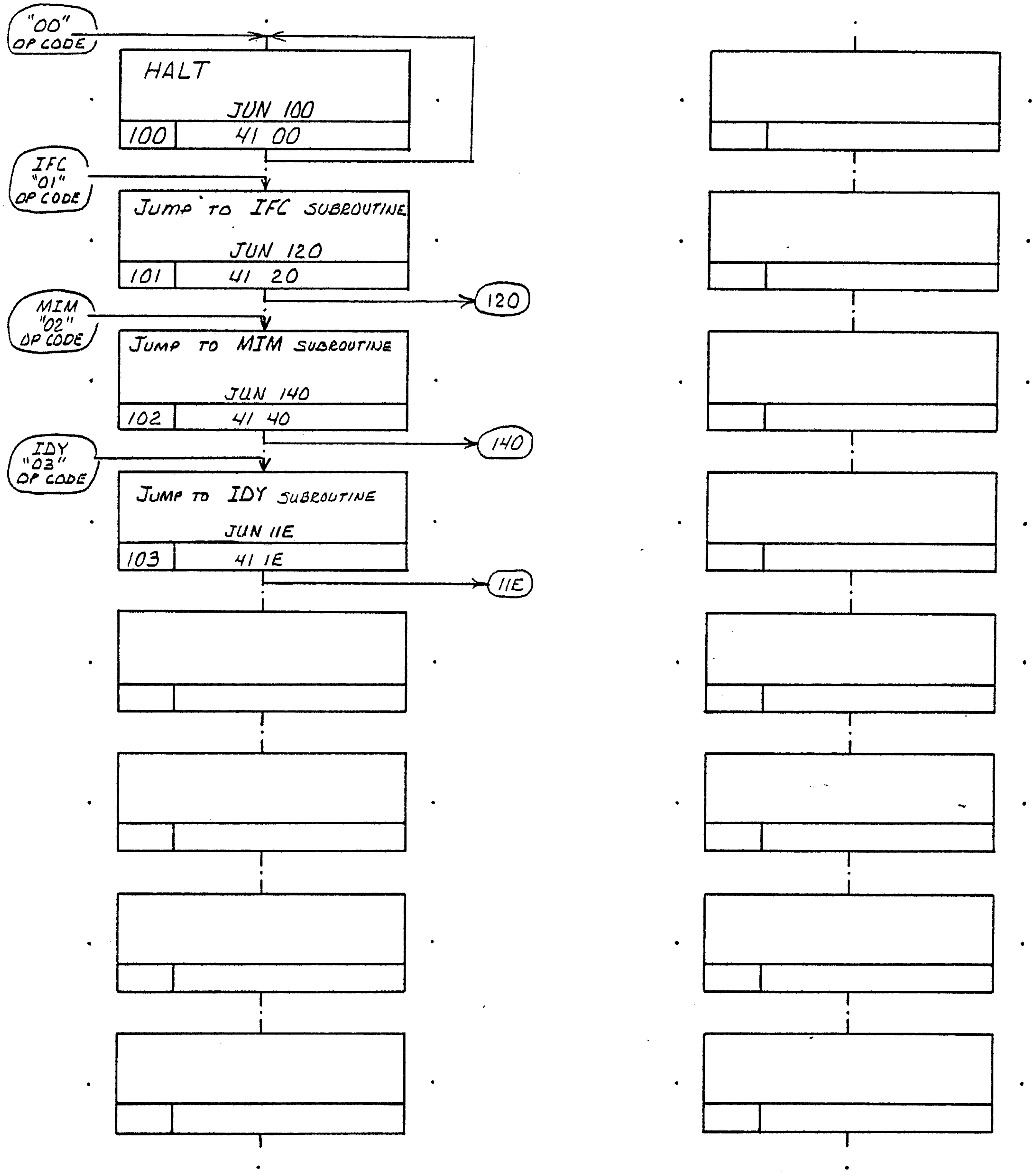
DECIMAL/HEX CONVERSION TABLE
256-511 DECIMAL
100-1FF HEXADECIMAL

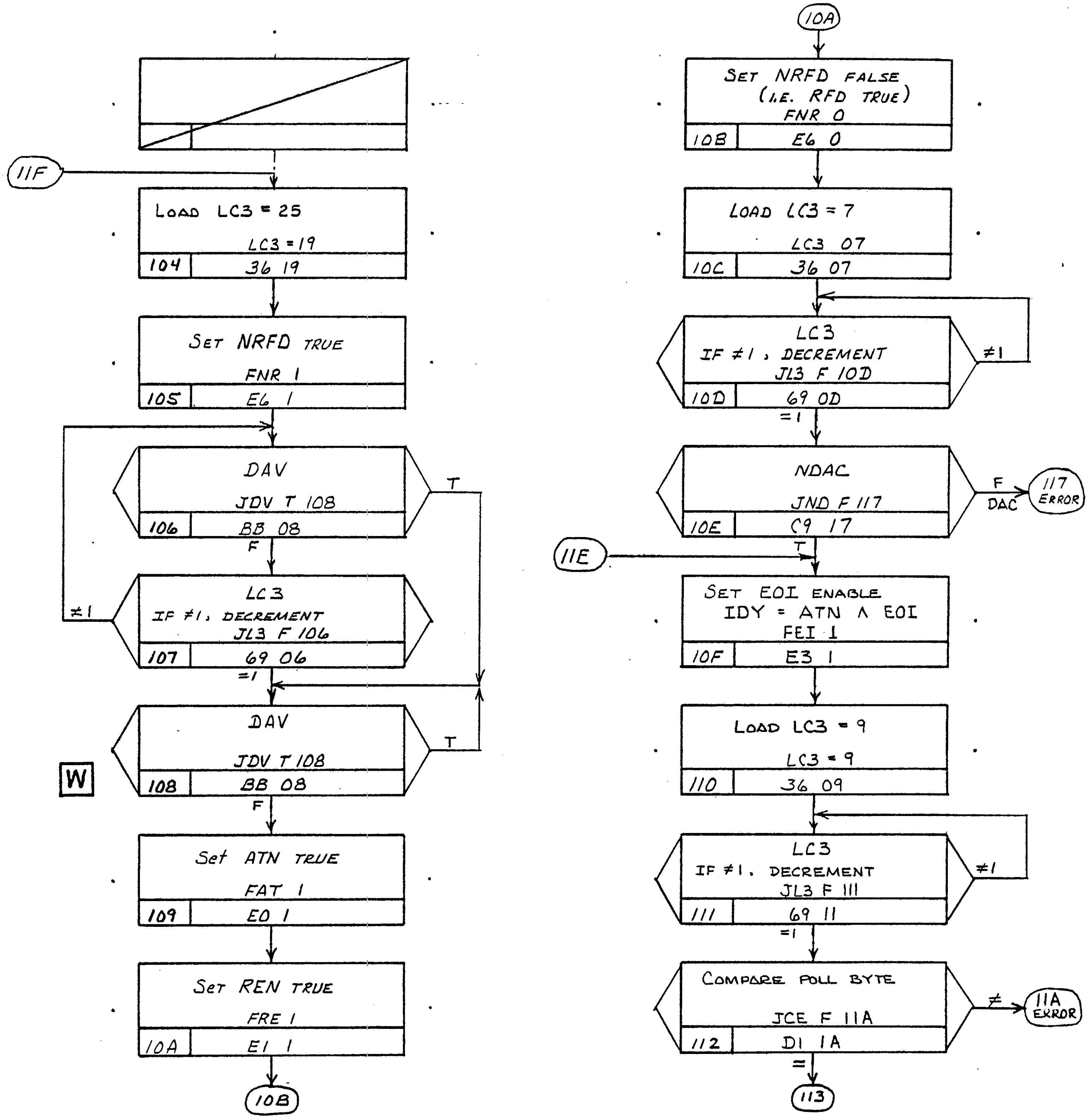
DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX	DEC	HEX
256	100	288	120	320	140	352	160	384	180	416	1A0	448	1C0	480	1E0
257	101	289	121	321	141	353	161	385	181	417	1A1	449	1C1	481	1E1
258	102	290	122	322	142	354	162	386	182	418	1A2	450	1C2	482	1E2
259	103	291	123	323	143	355	163	387	183	419	1A3	451	1C3	483	1E3
260	104	292	124	324	144	356	164	388	184	420	1A4	452	1C4	484	1E4
261	105	293	125	325	145	357	165	389	185	421	1A5	453	1C5	485	1E5
262	106	294	126	326	146	358	166	390	186	422	1A6	454	1C6	486	1E6
263	107	295	127	327	147	359	167	391	187	423	1A7	455	1C7	487	1E7
264	108	296	128	328	148	360	168	392	188	424	1A8	456	1C8	488	1E8
265	109	297	129	329	149	361	169	393	189	425	1A9	457	1C9	489	1E9
266	10A	298	12A	330	14A	362	16A	394	18A	426	1AA	458	1CA	490	1EA
267	10B	299	12B	331	14B	363	16B	395	18B	427	1AB	459	1CB	491	1EB
268	10C	300	12C	332	14C	364	16C	396	18C	428	1AC	460	1CC	492	1EC
269	10D	301	12D	333	14D	365	16D	397	18D	429	1AD	461	1CD	493	1ED
270	10E	302	12E	334	14E	366	16E	398	18E	430	1AE	462	1CE	494	1EE
271	10F	303	12F	335	14F	367	16F	399	18F	431	1AF	463	1CF	495	1EF
272	110	304	130	336	150	368	170	400	190	432	1B0	464	1D0	496	1F0
273	111	305	131	337	151	369	171	401	191	433	1B1	465	1D1	497	1F1
274	112	306	132	338	152	370	172	402	192	434	1B2	466	1D2	498	1F2
275	113	307	133	339	153	371	173	403	193	435	1B3	467	1D3	499	1F3
276	114	308	134	340	154	372	174	404	194	436	1B4	468	1D4	500	1F4
277	115	309	135	341	155	373	175	405	195	437	1B5	469	1D5	501	1F5
278	116	310	136	342	156	374	176	406	196	438	1B6	470	1D6	502	1F6
279	117	311	137	343	157	375	177	407	197	439	1B7	471	1D7	503	1F7
280	118	312	138	344	158	376	178	408	198	440	1B8	472	1D8	504	1F8
281	119	313	139	345	159	377	179	409	199	441	1B9	473	1D9	505	1F9
282	11A	314	13A	346	15A	378	17A	410	19A	442	1BA	474	1DA	506	1FA
283	11B	315	13B	347	15B	379	17B	411	19B	443	1BB	475	1DB	507	1FB
284	11C	316	13C	348	15C	380	17C	412	19C	444	1BC	476	1DC	508	1FC
285	11D	317	13D	349	15D	381	17D	413	19D	445	1BD	477	1DD	509	1FD
286	11E	318	13E	350	15E	382	17E	414	19E	446	1BE	478	1DE	510	1FE
287	11F	319	13F	351	15F	383	17F	415	19F	447	1BF	479	1DF	511	1FF

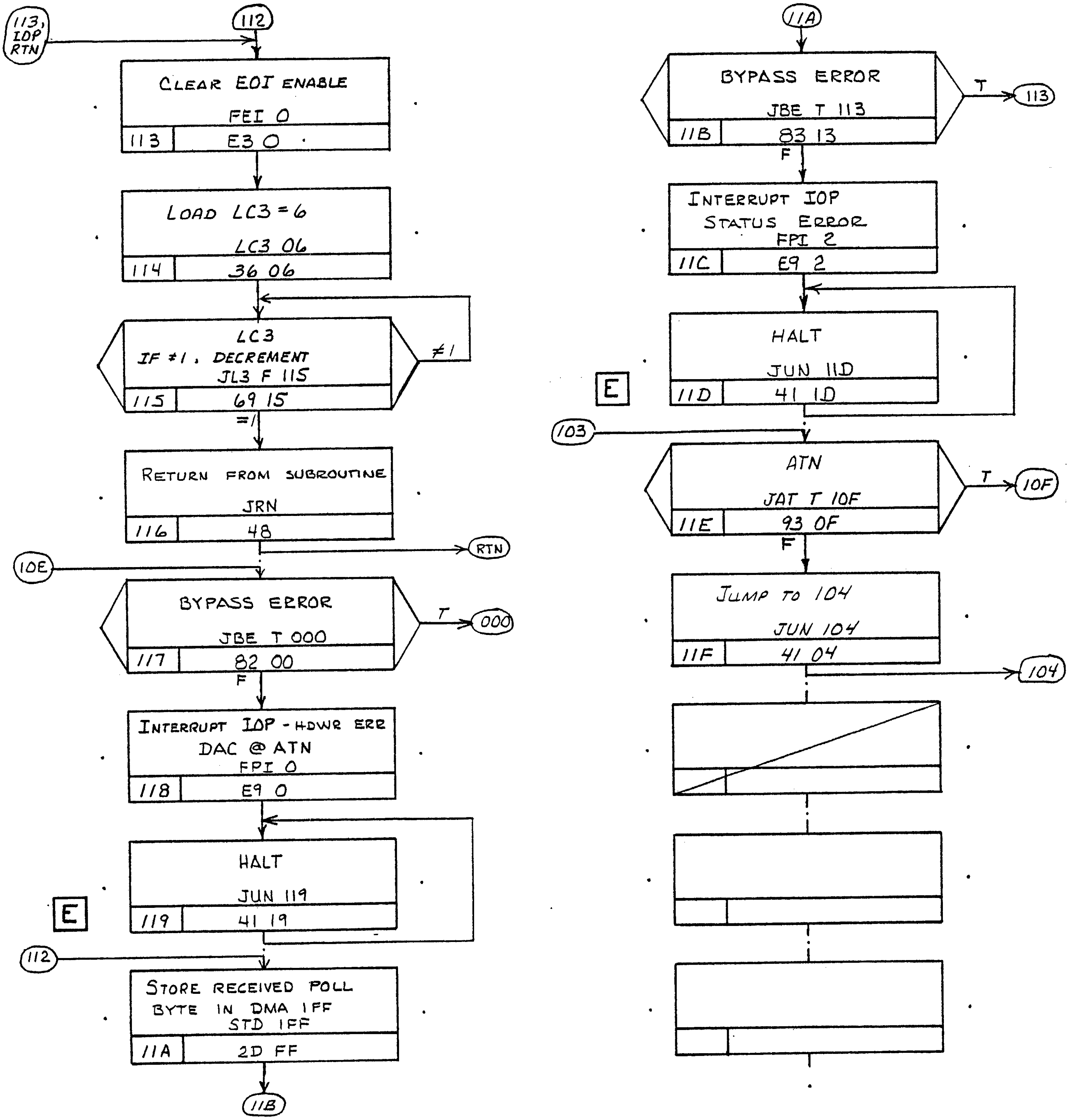
APPENDIX J
FLOWCHARTS FOR STANDARD SUBROUTINES

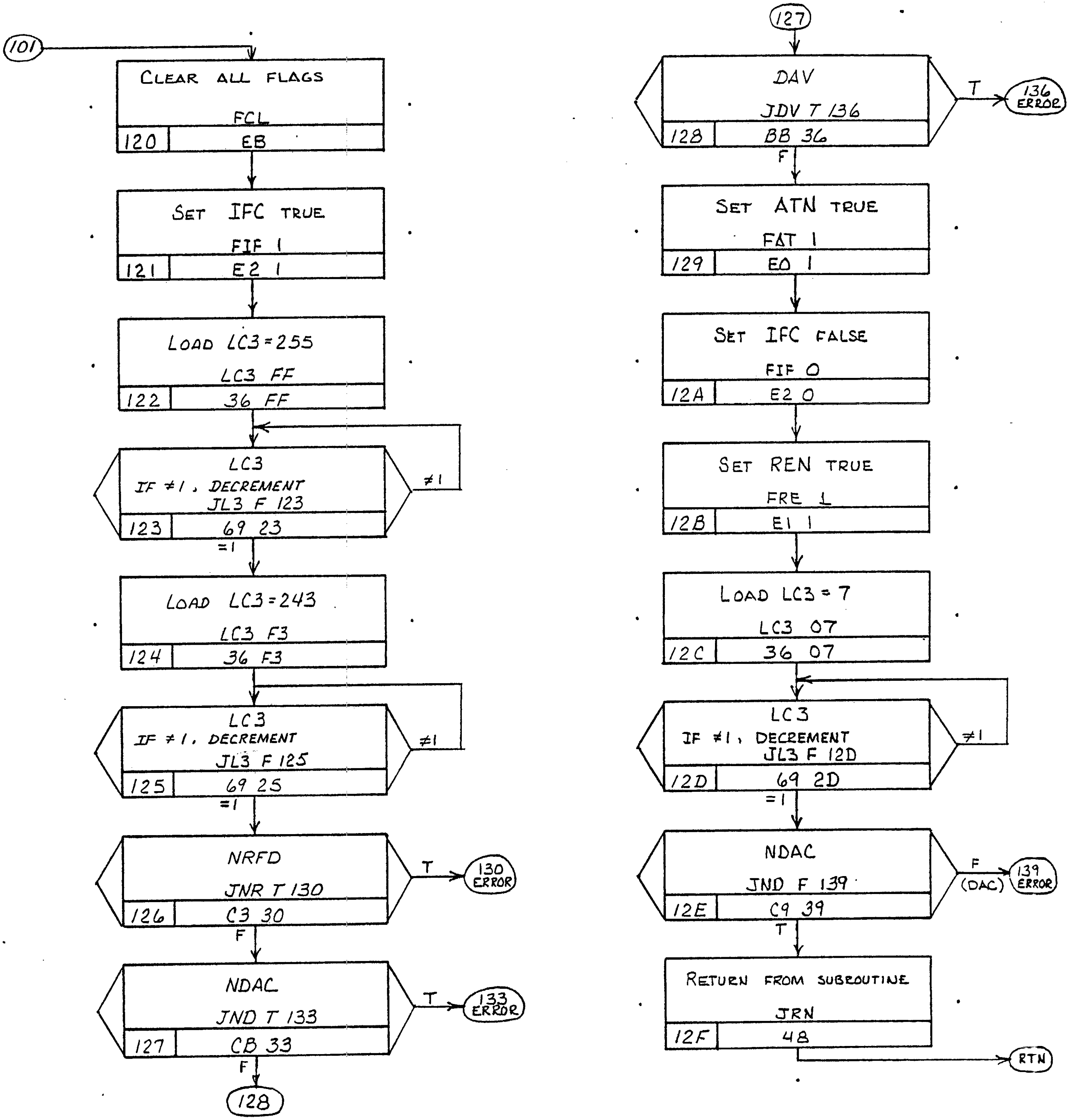
MODEL 488

9-7-78 BP ROM POINTERS PAGE 1 OF 1



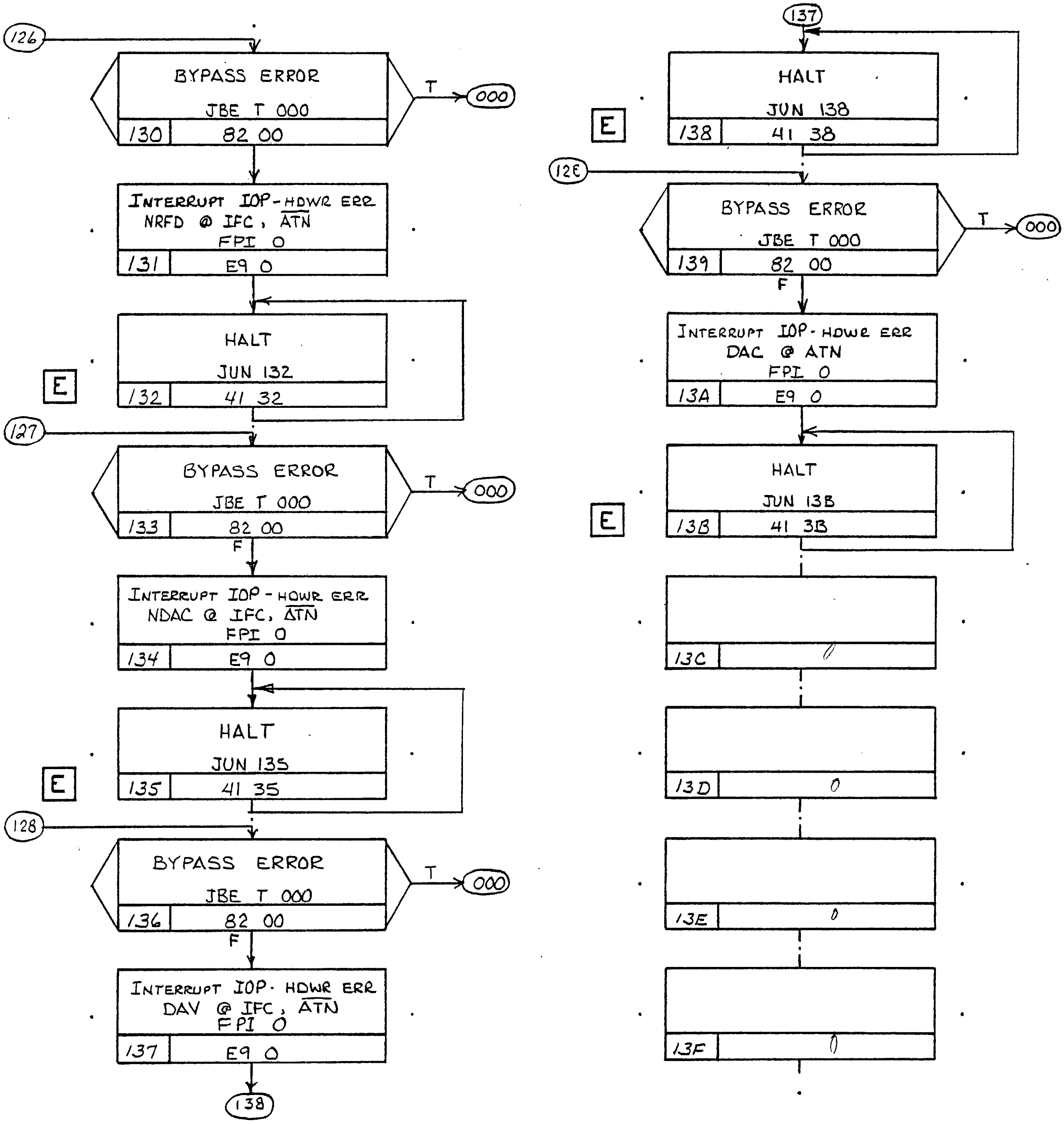

11F
LOAD LC3 = 25
LC3 = 19
104
36 19
SET NRFD TRUE
FNR 1
105
E6 1
DAV
JDV T 108
106
BB 08
T
F
LC3
IF ≠1, DECREMENT
JL3 F 106
107
69 06
≠1
=1
DAV
JDV T 108
108
BB 08
T
F
W
Set ATN TRUE
FAT 1
109
E0 1
Set REN TRUE
FRE 1
10A
E1 1
10B
10A
SET NRFD FALSE
(I.E. RFD TRUE)
FNR 0
10B
E6 0
LOAD LC3 = 7
LC3 07
10C
36 07
LC3
IF ≠1, DECREMENT
JL3 F 10D
10D
69 0D
≠1
=1
NDAC
JND F 117
10E
C9 17
F
DAC
117
ERROR
T
11E
SET EOI ENABLE
IDY = ATN ∧ EOI
FEI 1
10F
E3 1
LOAD LC3 = 9
LC3 = 9
110
36 09
LC3
IF ≠1, DECREMENT
JL3 F 111
111
69 11
≠1
=1
COMPARE POLL BYTE
JCE F 11A
112
D1 1A
≠
11A
ERROR
=
113

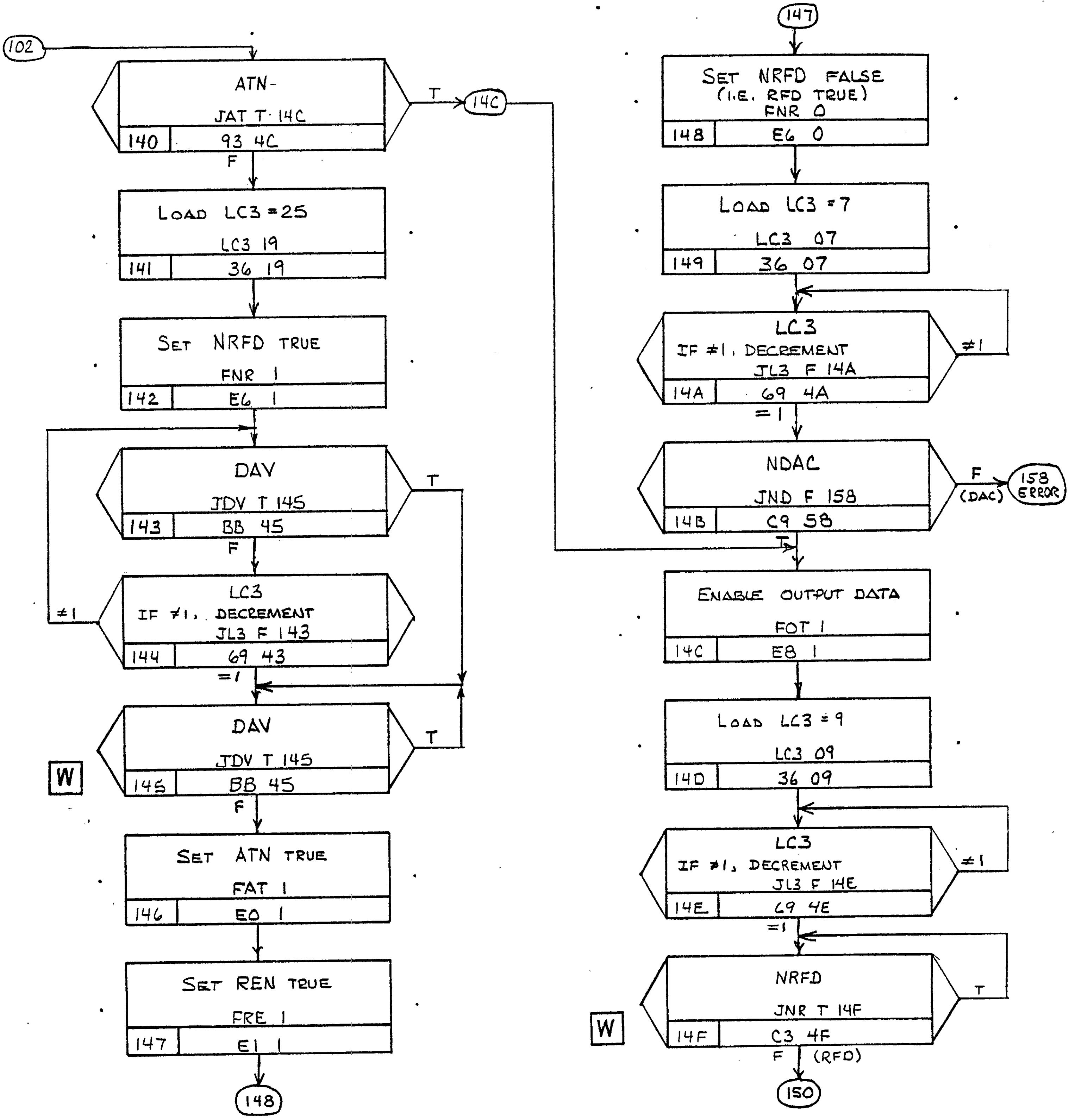

113, IDP RTN
112
CLEAR EOI ENABLE
FEI 0
113
E3 0
LOAD LC3 = 6
LC3 06
114
36 06
LC3
IF ≠1, DECREMENT
JL3 F 115
≠1
115
69 15
=1
RETURN FROM SUBROUTINE
JRN
116
48
RTN
10E
BYPASS ERROR
JBE T 000
T
000
117
82 00
F
INTERRUPT IOP - HDWR ERR
DAC @ ATN
FPI 0
118
E9 0
HALT
JUN 119
E
119
41 19
112
STORE RECEIVED POLL
BYTE IN DMA 1FF
STD 1FF
11A
2D FF
11B
11A
BYPASS ERROR
JBE T 113
T
113
11B
83 13
F
INTERRUPT IOP
STATUS ERROR
FPI 2
11C
E9 2
HALT
JUN 11D
E
11D
41 1D
103
ATN
JAT T 10F
T
10F
11E
93 0F
F
JUMP TO 104
JUN 104
11F
41 04
104

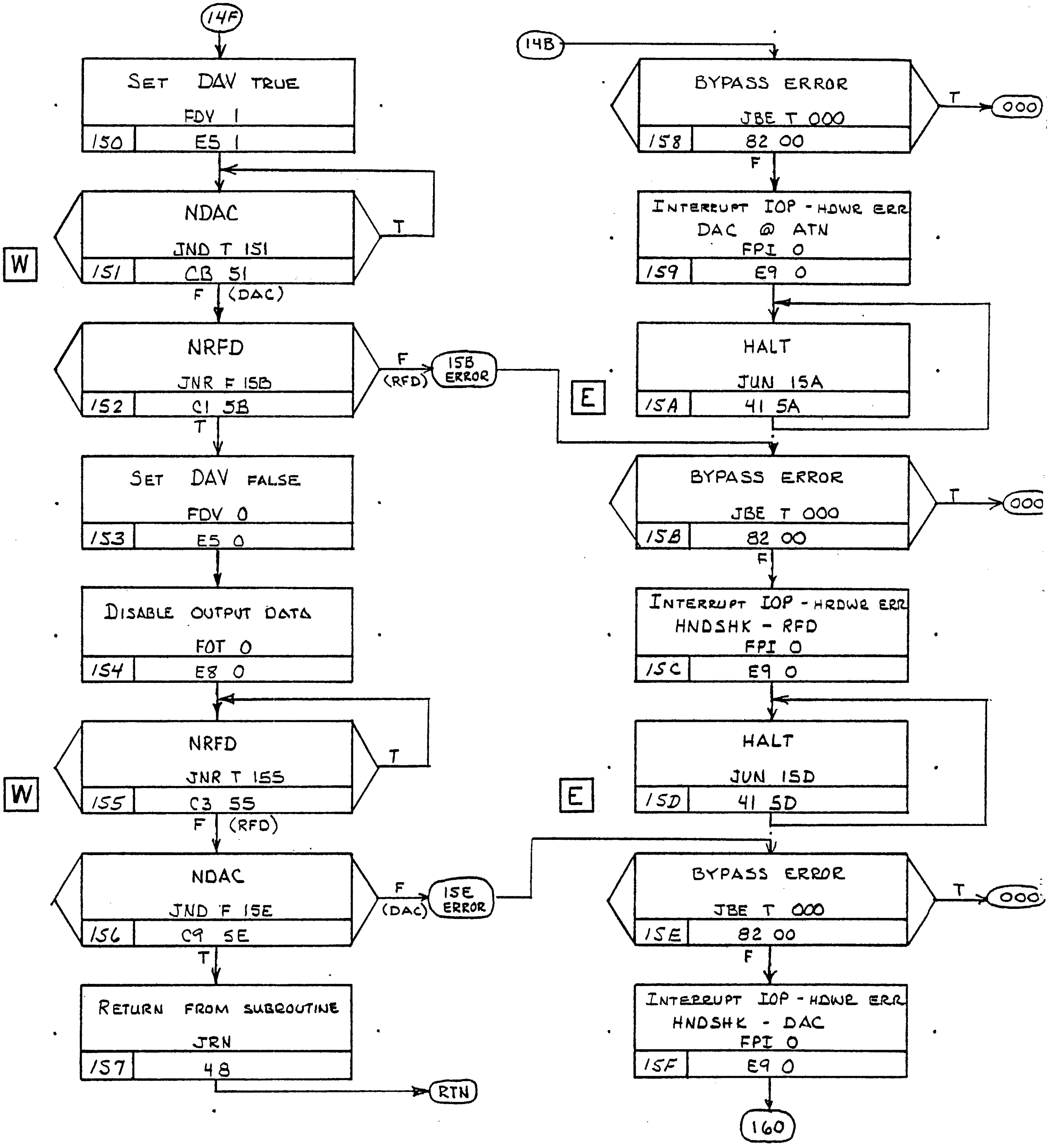

101
CLEAR ALL FLAGS
FCL
120
EB
SET IFC TRUE
FIF 1
121
E2 1
LOAD LC3=255
LC3 FF
122
36 FF
LC3
IF ≠1, DECREMENT
JL3 F 123
123
69 23
≠1
=1
LOAD LC3=243
LC3 F3
124
36 F3
LC3
IF ≠1, DECREMENT
JL3 F 125
125
69 25
≠1
=1
NRFD
JNR T 130
126
C3 30
T
130 ERROR
F
NDAC
JND T 133
127
CB 33
T
133 ERROR
F
128
127
DAV
JDV T 136
128
BB 36
T
136 ERROR
F
SET ATN TRUE
FAT 1
129
E0 1
SET IFC FALSE
FIF 0
12A
E2 0
SET REN TRUE
FRE 1
12B
E1 1
LOAD LC3 = 7
LC3 07
12C
36 07
LC3
IF ≠1, DECREMENT
JL3 F 12D
12D
69 2D
≠1
=1
NDAC
JND F 139
12E
C9 39
F
(DAC)
139 ERROR
T
RETURN FROM SUBROUTINE
JRN
12F
48
RTN

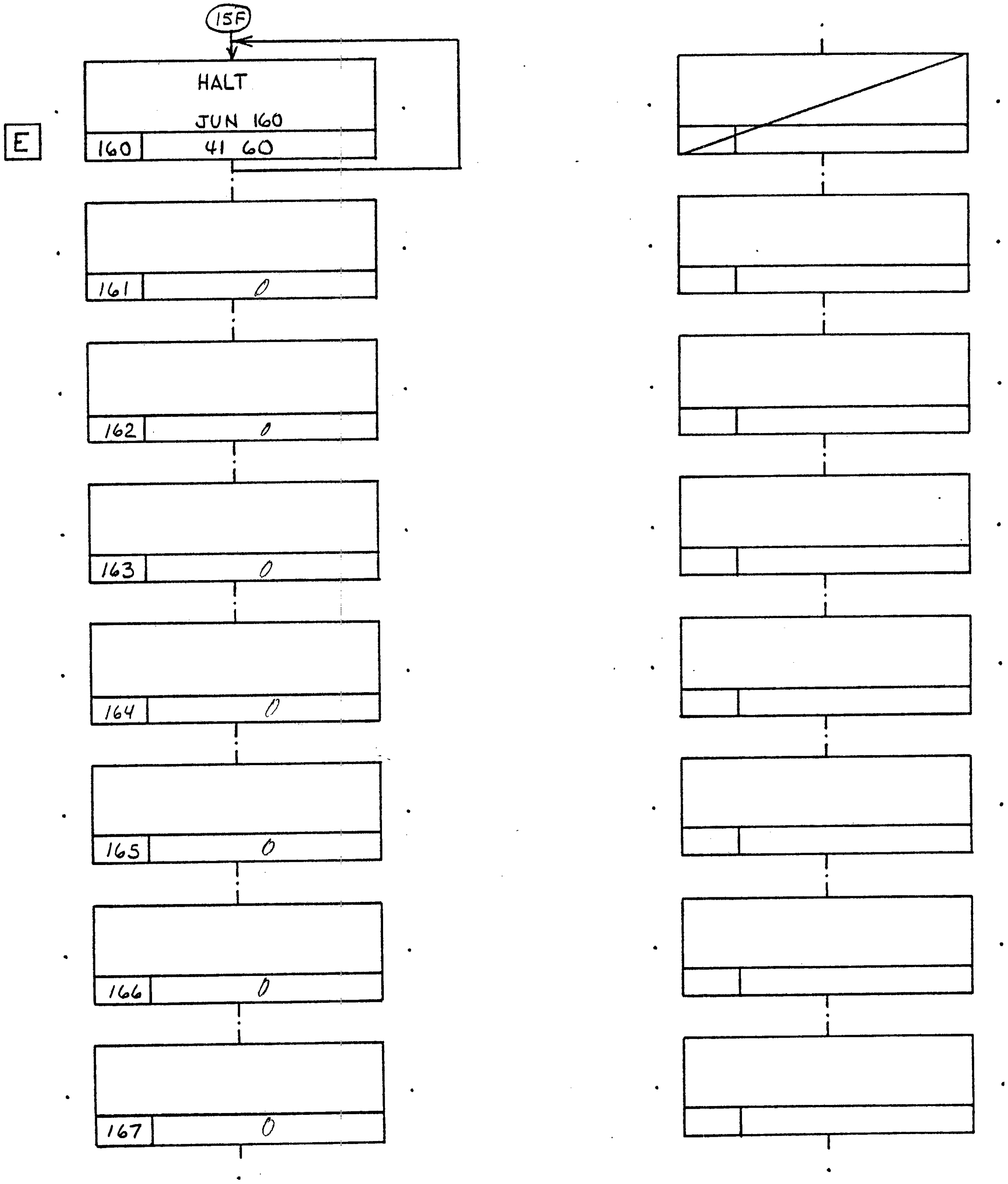
MODEL 488

IFC SUBROUTINE



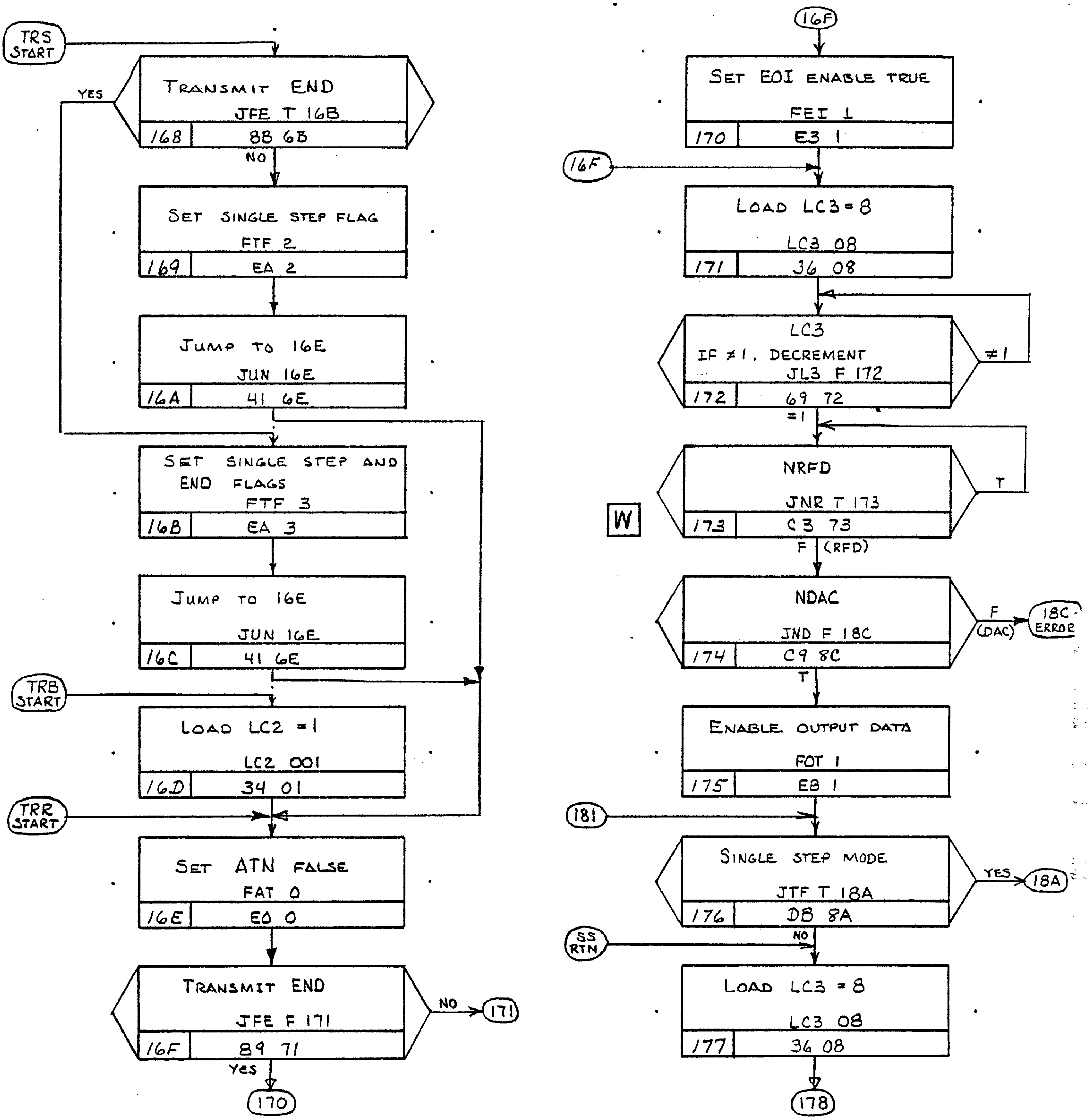

102
ATN-
JAT T 14C
140
93 4C
T
14C
F
LOAD LC3 = 25
LC3 19
141
36 19
SET NRFD TRUE
FNR 1
142
E6 1
DAV
JDV T 145
143
BB 45
T
F
≠1
LC3
IF ≠1, DECREMENT
JL3 F 143
144
69 43
=1
DAV
JDV T 145
145
BB 45
T
W
F
SET ATN TRUE
FAT 1
146
E0 1
SET REN TRUE
FRE 1
147
E1 1
148
147
SET NRFD FALSE
(I.E. RFD TRUE)
FNR 0
148
E6 0
LOAD LC3 = 7
LC3 07
149
36 07
LC3
IF ≠1, DECREMENT
JL3 F 14A
14A
69 4A
≠1
=1
NDAC
JND F 158
14B
C9 58
F
(DAC)
158
ERROR
T
ENABLE OUTPUT DATA
FOT 1
14C
E8 1
LOAD LC3 = 9
LC3 09
14D
36 09
LC3
IF ≠1, DECREMENT
JL3 F 14E
14E
69 4E
≠1
=1
NRFD
JNR T 14F
14F
C3 4F
T
W
F
(RFD)
150

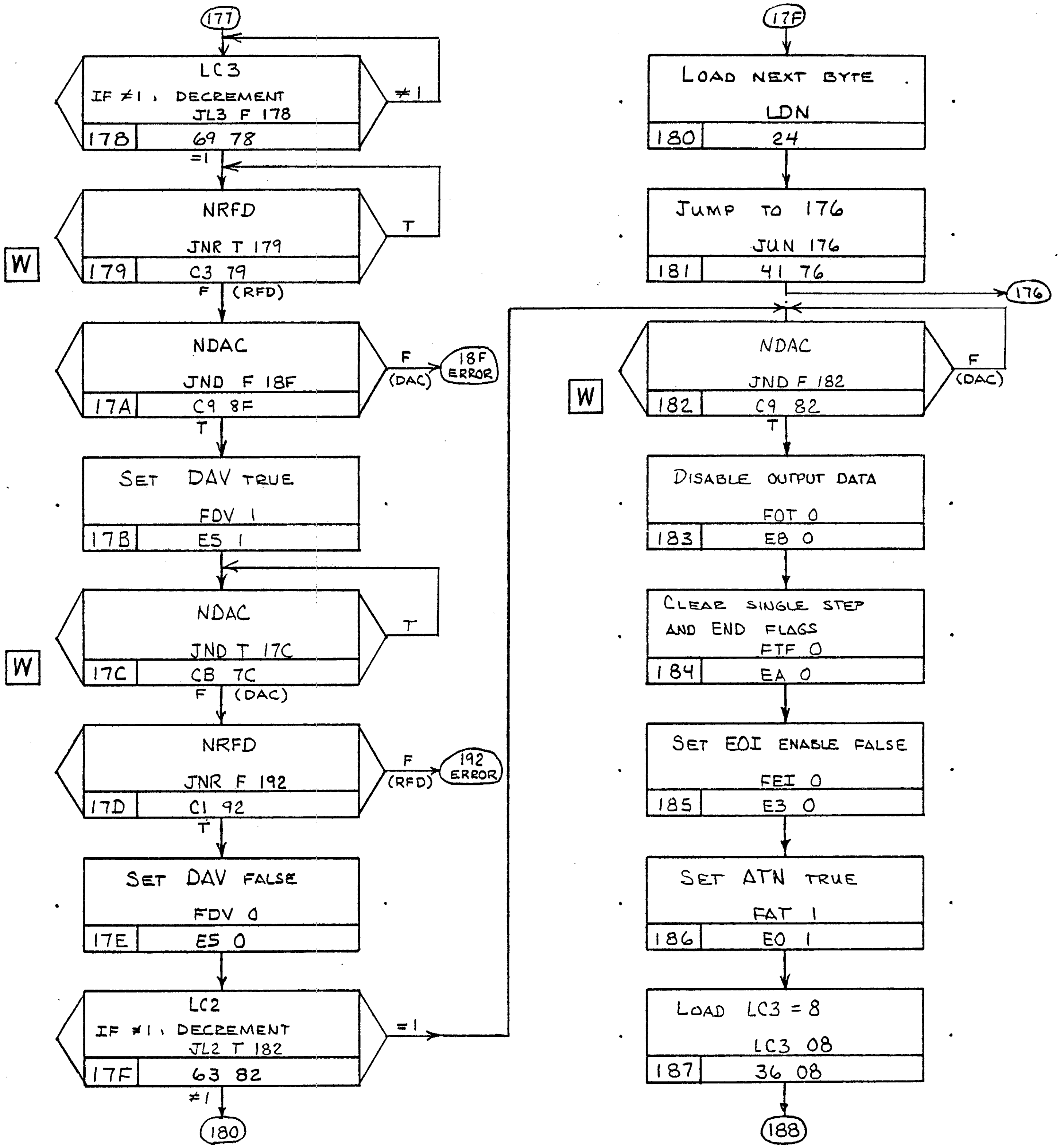

14F
SET DAV TRUE
FDV 1
150
E5 1
NDAC
JND T 151
151
CB 51
T
W
F (DAC)
NRFD
JNR F 15B
152
C1 5B
F
(RFD)
15B ERROR
T
SET DAV FALSE
FDV 0
153
E5 0
DISABLE OUTPUT DATA
FOT 0
154
E8 0
NRFD
JNR T 155
155
C3 55
T
W
F (RFD)
NDAC
JND F 15E
156
C9 5E
F
(DAC)
15E ERROR
T
RETURN FROM SUBROUTINE
JRN
157
48
RTN
14B
BYPASS ERROR
JBE T 000
158
82 00
T
000
F
INTERRUPT IOP - HDWR ERR
DAC @ ATN
FPI 0
159
E9 0
HALT
JUN 15A
15A
41 5A
E
BYPASS ERROR
JBE T 000
15B
82 00
T
000
F
INTERRUPT IOP - HRDWR ERR
HNDSHK - RFD
FPI 0
15C
E9 0
HALT
JUN 15D
15D
41 5D
E
BYPASS ERROR
JBE T 000
15E
82 00
T
000
F
INTERRUPT IOP - HDWR ERR
HNDSHK - DAC
FPI 0
15F
E9 0
160

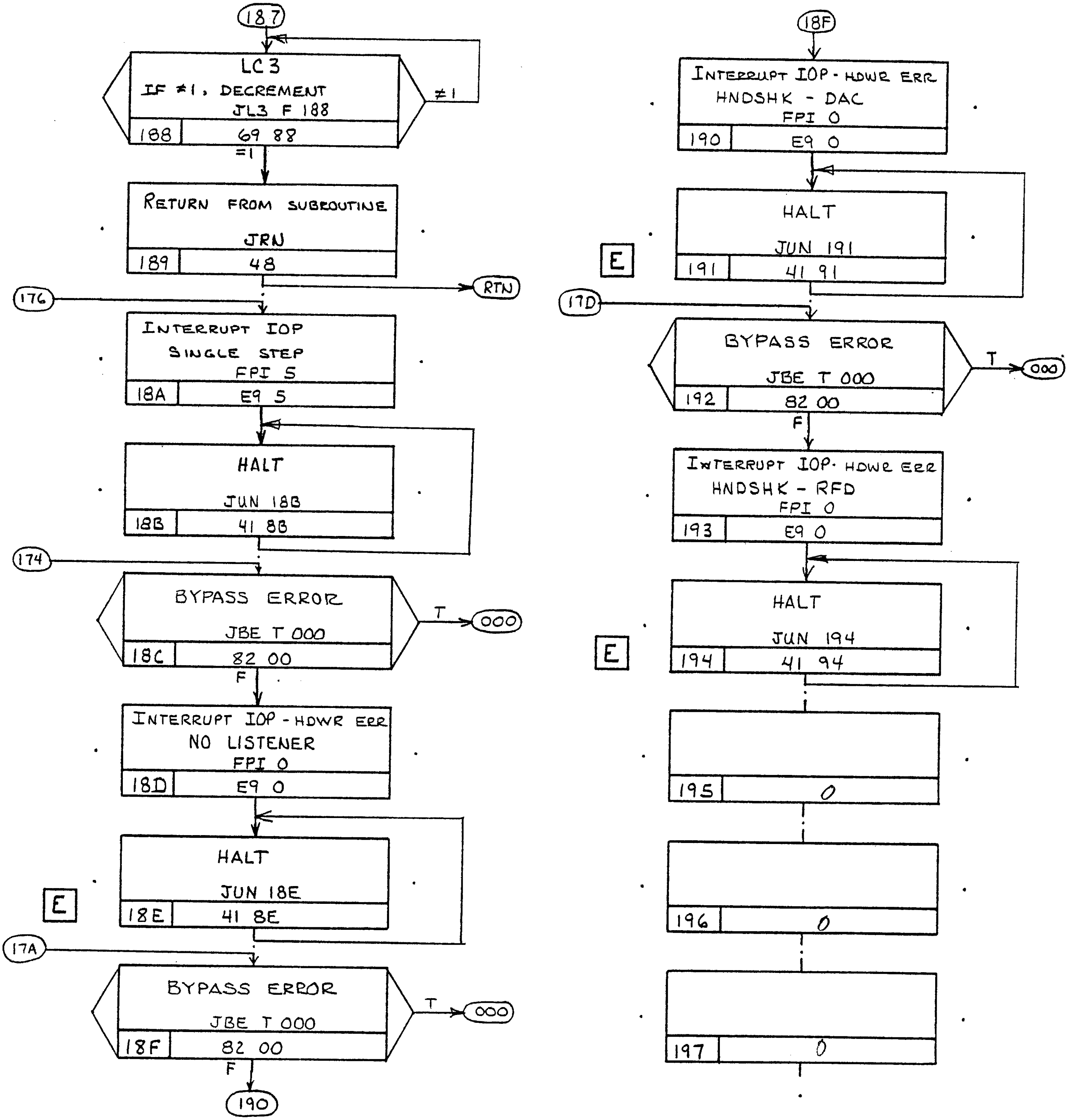

15F
HALT
JUN 160
E
160
41 60
161
0
162
0
163
0
164
0
165
0
166
0
167
0

MODEL 488

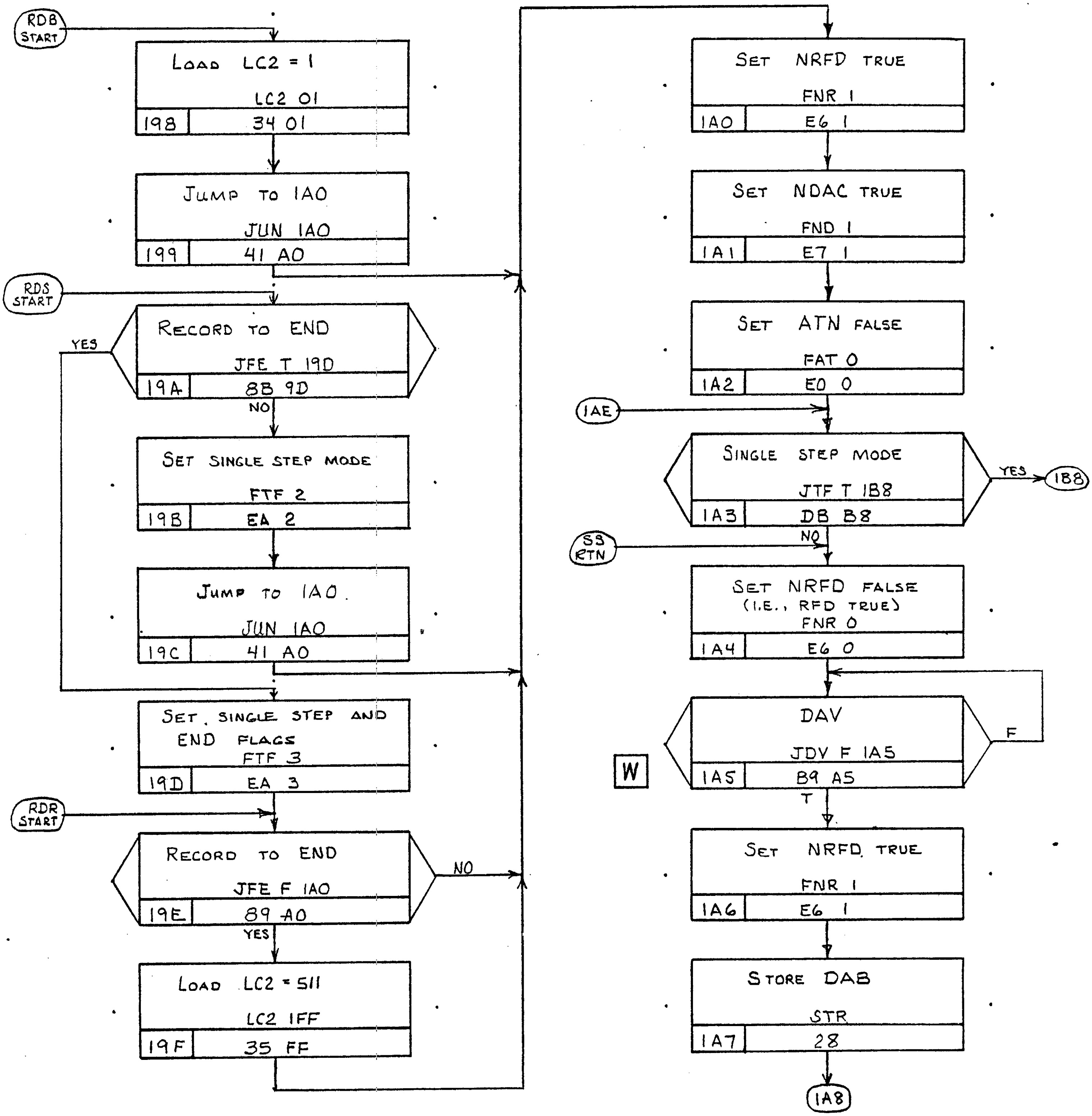
8-1-78 TRB.TRR.TRS SUBROUTINE PAGE 1 OF 3

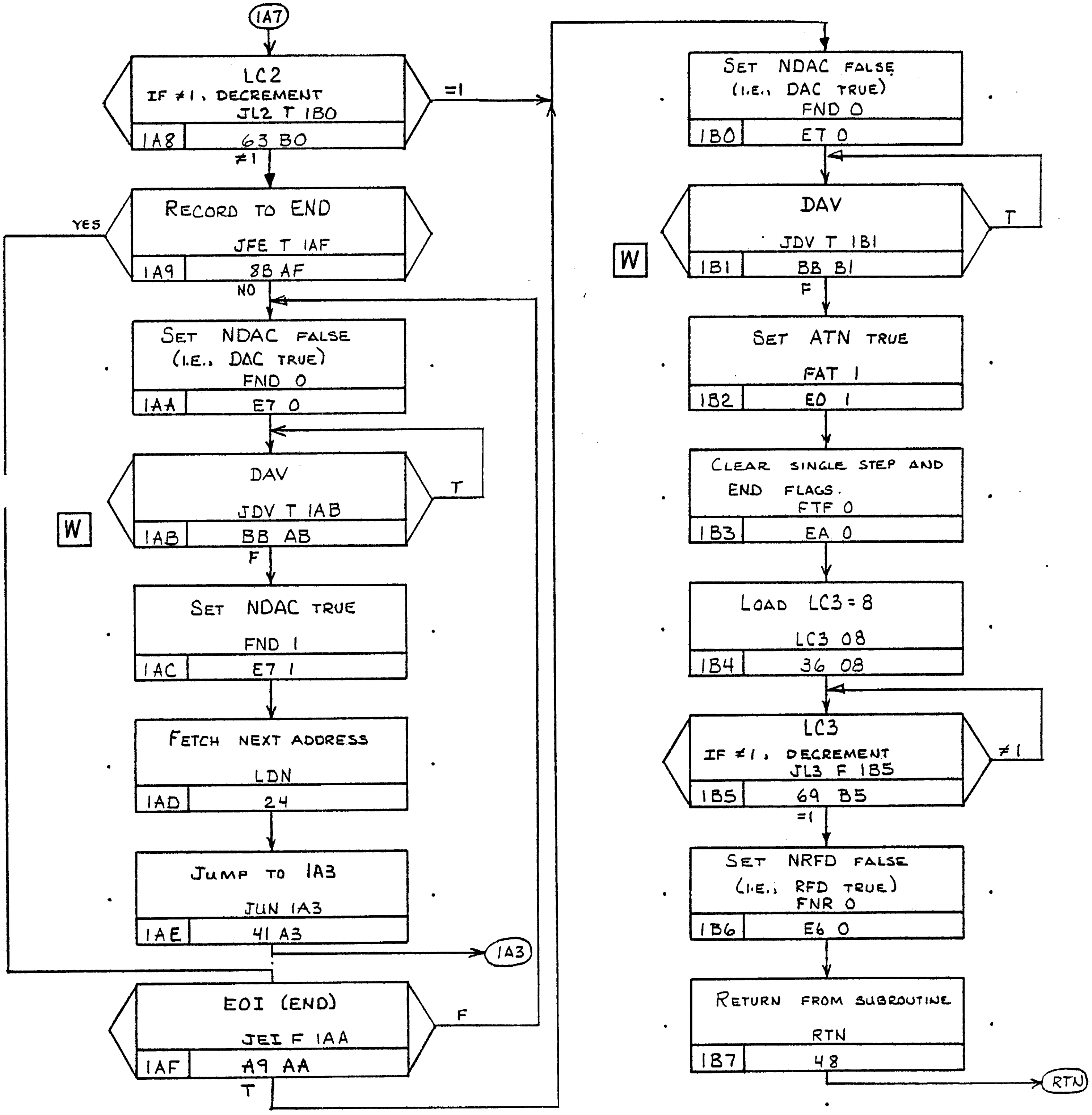


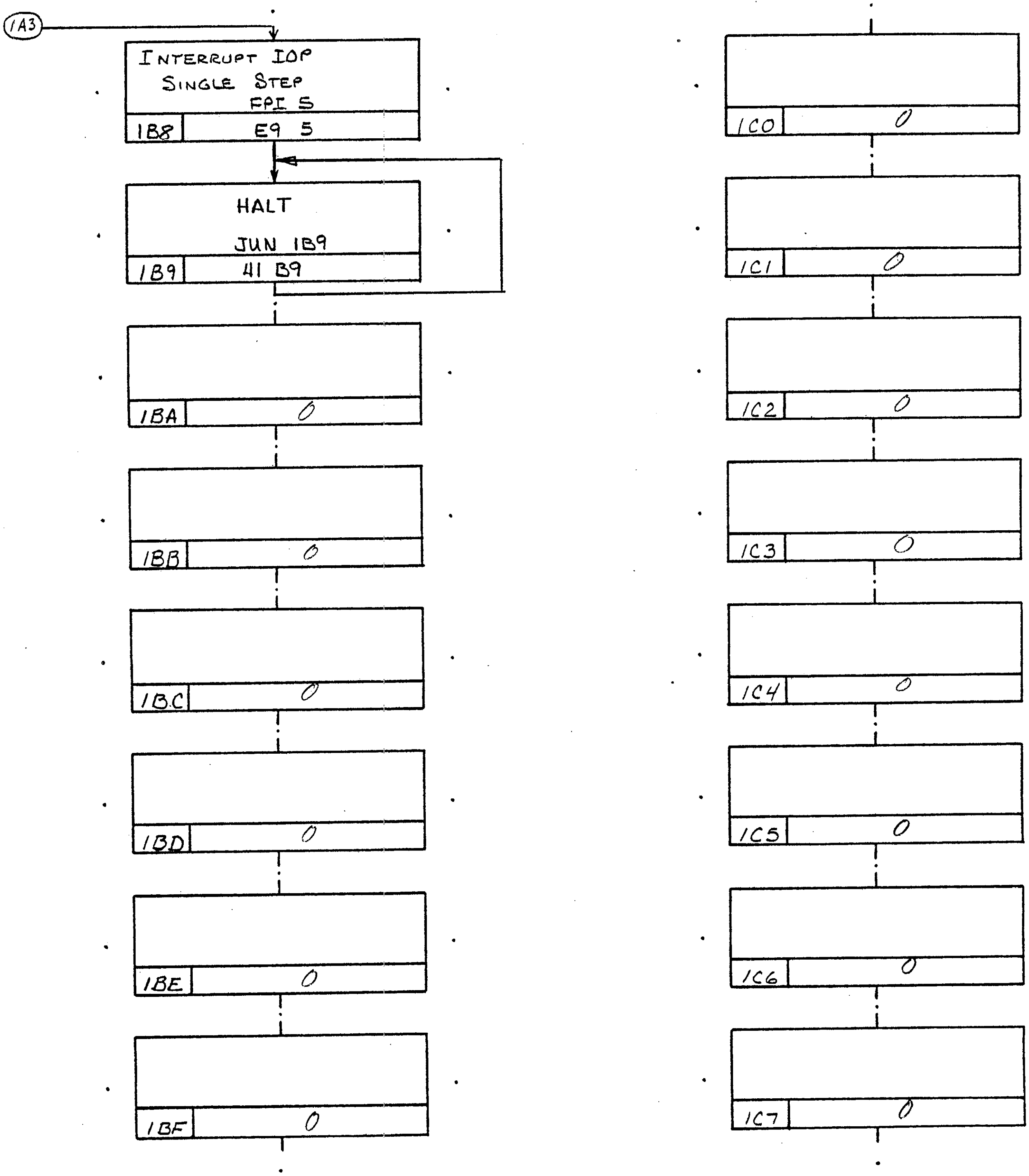

177
LC3
IF ≠1, DECREMENT
JL3 F 178
178
69 78
≠1
=1
NRFD
JNR T 179
179
C3 79
T
W
F
(RFD)
NDAC
JND F 18F
17A
C9 8F
F
(DAC)
18F
ERROR
T
SET DAV TRUE
FDV 1
17B
E5 1
NDAC
JND T 17C
17C
CB 7C
T
W
F
(DAC)
NRFD
JNR F 192
17D
C1 92
F
(RFD)
192
ERROR
T
SET DAV FALSE
FDV 0
17E
E5 0
LC2
IF ≠1, DECREMENT
JL2 T 182
17F
63 82
=1
≠1
180
17F
LOAD NEXT BYTE
LDN
180
24
JUMP TO 176
JUN 176
181
41 76
176
NDAC
JND F 182
182
C9 82
F
(DAC)
W
T
DISABLE OUTPUT DATA
FOT 0
183
E8 0
CLEAR SINGLE STEP
AND END FLAGS
FTF 0
184
EA 0
SET EOI ENABLE FALSE
FEI 0
185
E3 0
SET ATN TRUE
FAT 1
186
E0 1
LOAD LC3 = 8
LC3 08
187
36 08
188


187
LC3
IF ≠1, DECREMENT
JL3 F 188
188
69 88
≠1
=1
RETURN FROM SUBROUTINE
JRN
189
48
RTN
176
INTERRUPT IOP
SINGLE STEP
FPI 5
18A
E9 5
HALT
JUN 18B
18B
41 8B
174
BYPASS ERROR
JBE T 000
18C
82 00
T
000
F
INTERRUPT IOP - HDWR ERR
NO LISTENER
FPI 0
18D
E9 0
HALT
JUN 18E
18E
41 8E
E
17A
BYPASS ERROR
JBE T 000
18F
82 00
T
000
F
190
18F
INTERRUPT IOP - HDWR ERR
HNDSHK - DAC
FPI 0
190
E9 0
HALT
JUN 191
191
41 91
E
17D
BYPASS ERROR
JBE T 000
192
82 00
T
000
F
INTERRUPT IOP - HDWR ERR
HNDSHK - RFD
FPI 0
193
E9 0
HALT
JUN 194
194
41 94
E
195
0
196
0
197
0

RDB, RDR, RDS SUBROUTINE

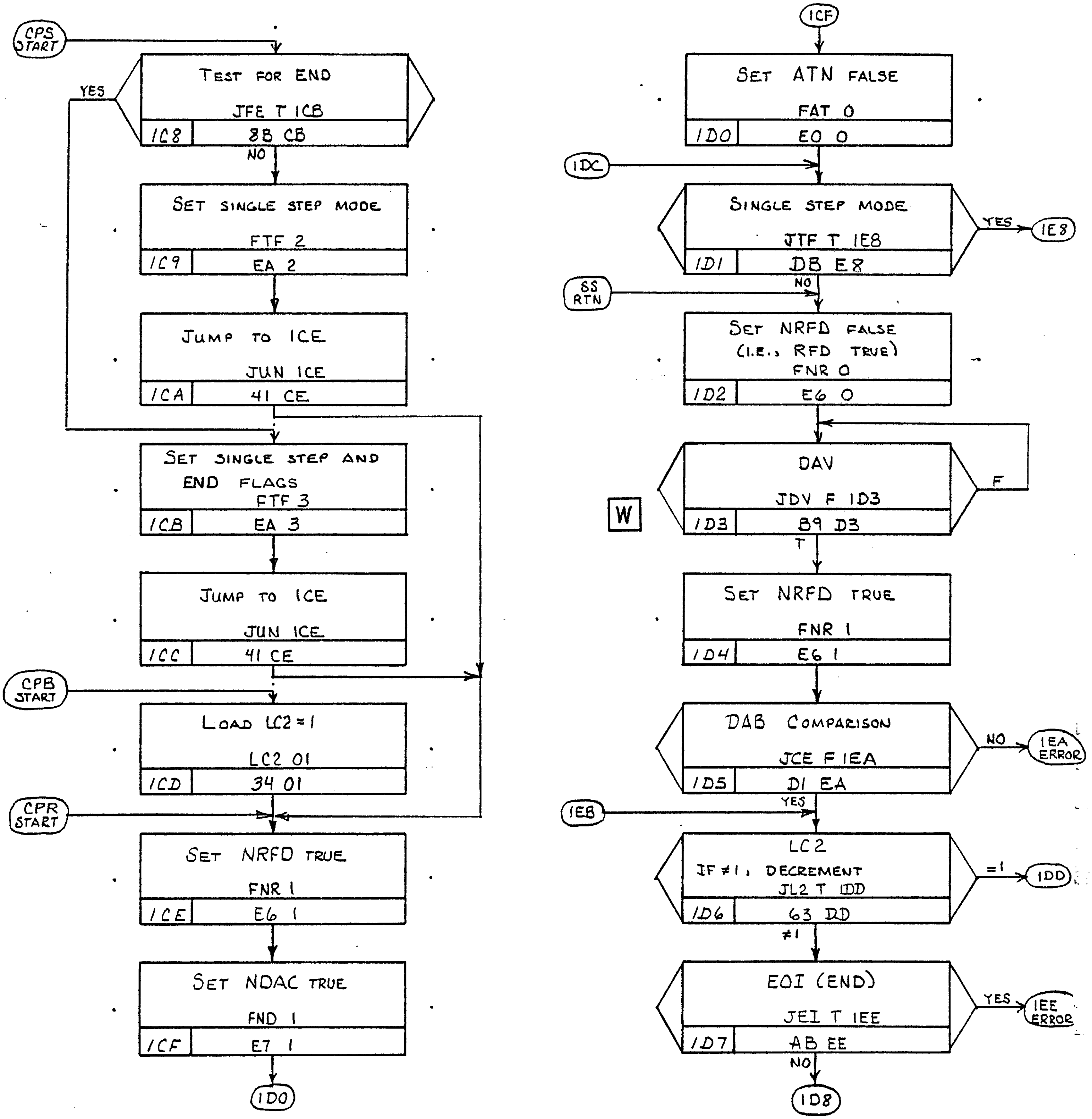



1A7
LC2
IF ≠1, DECREMENT
JL2 T 1B0
1A8
63 B0
=1
≠1
RECORD TO END
JFE T 1AF
1A9
8B AF
YES
NO
SET NDAC FALSE
(I.E., DAC TRUE)
FND 0
1AA
E7 0
DAV
JDV T 1AB
1AB
BB AB
T
F
W
SET NDAC TRUE
FND 1
1AC
E7 1
FETCH NEXT ADDRESS
LDN
1AD
24
JUMP TO 1A3
JUN 1A3
1AE
41 A3
1A3
EOI (END)
JEI F 1AA
1AF
A9 AA
F
T
SET NDAC FALSE
(I.E., DAC TRUE)
FND 0
1B0
E7 0
DAV
JDV T 1B1
1B1
BB B1
T
F
W
SET ATN TRUE
FAT 1
1B2
E0 1
CLEAR SINGLE STEP AND
END FLAGS.
FTF 0
1B3
EA 0
LOAD LC3=8
LC3 08
1B4
36 08
LC3
IF ≠1, DECREMENT
JL3 F 1B5
1B5
69 B5
≠1
=1
SET NRFD FALSE
(I.E., RFD TRUE)
FNR 0
1B6
E6 0
RETURN FROM SUBROUTINE
RTN
1B7
48
RTN


1A3
INTERRUPT IOP
SINGLE STEP
FPI 5
1B8
E9 5
HALT
JUN 1B9
1B9
41 B9
1BA
0
1BB
0
1BC
0
1BD
0
1BE
0
1BF
0
1C0
0
1C1
0
1C2
0
1C3
0
1C4
0
1C5
0
1C6
0
1C7
0

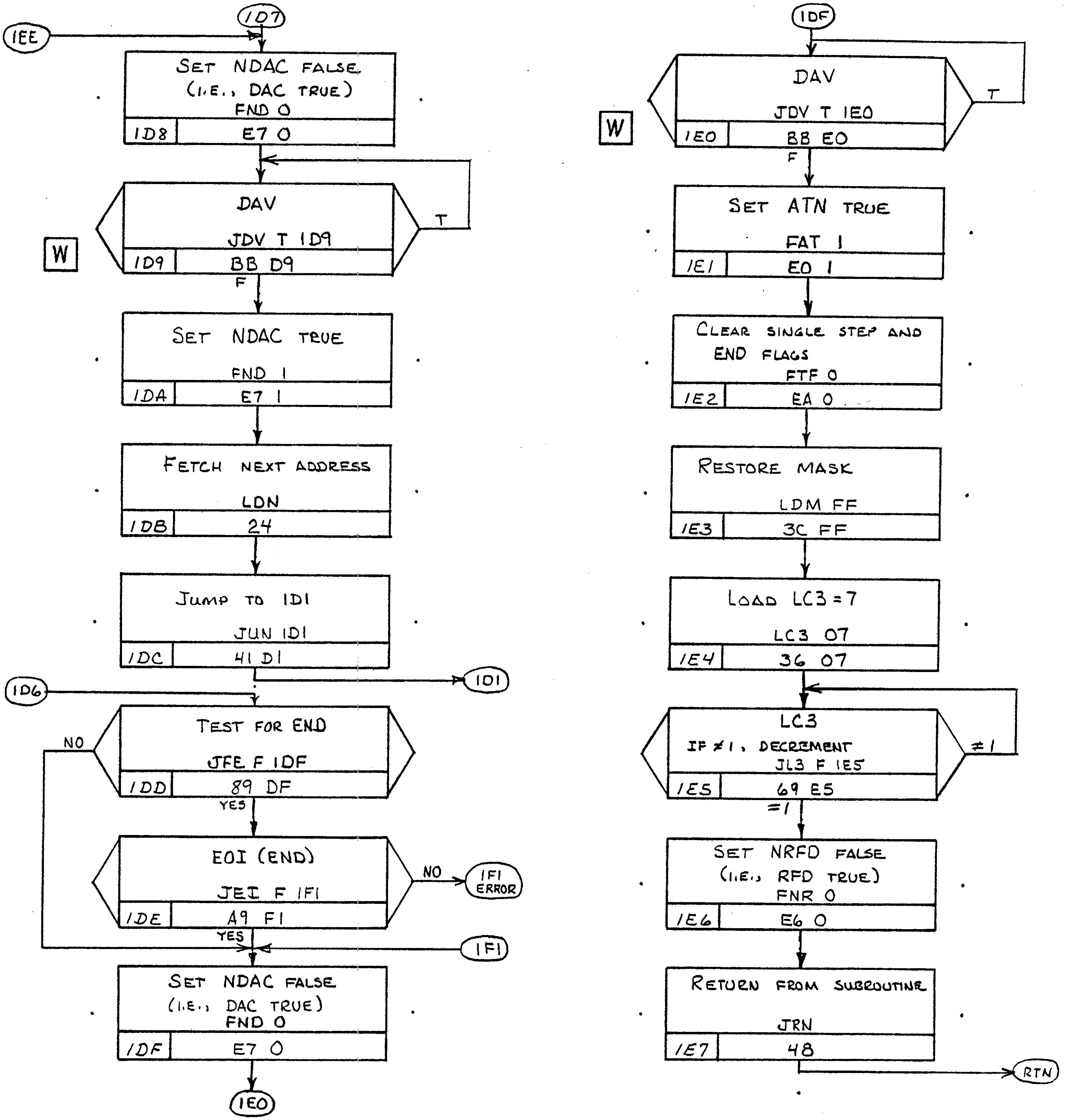
MODEL 488

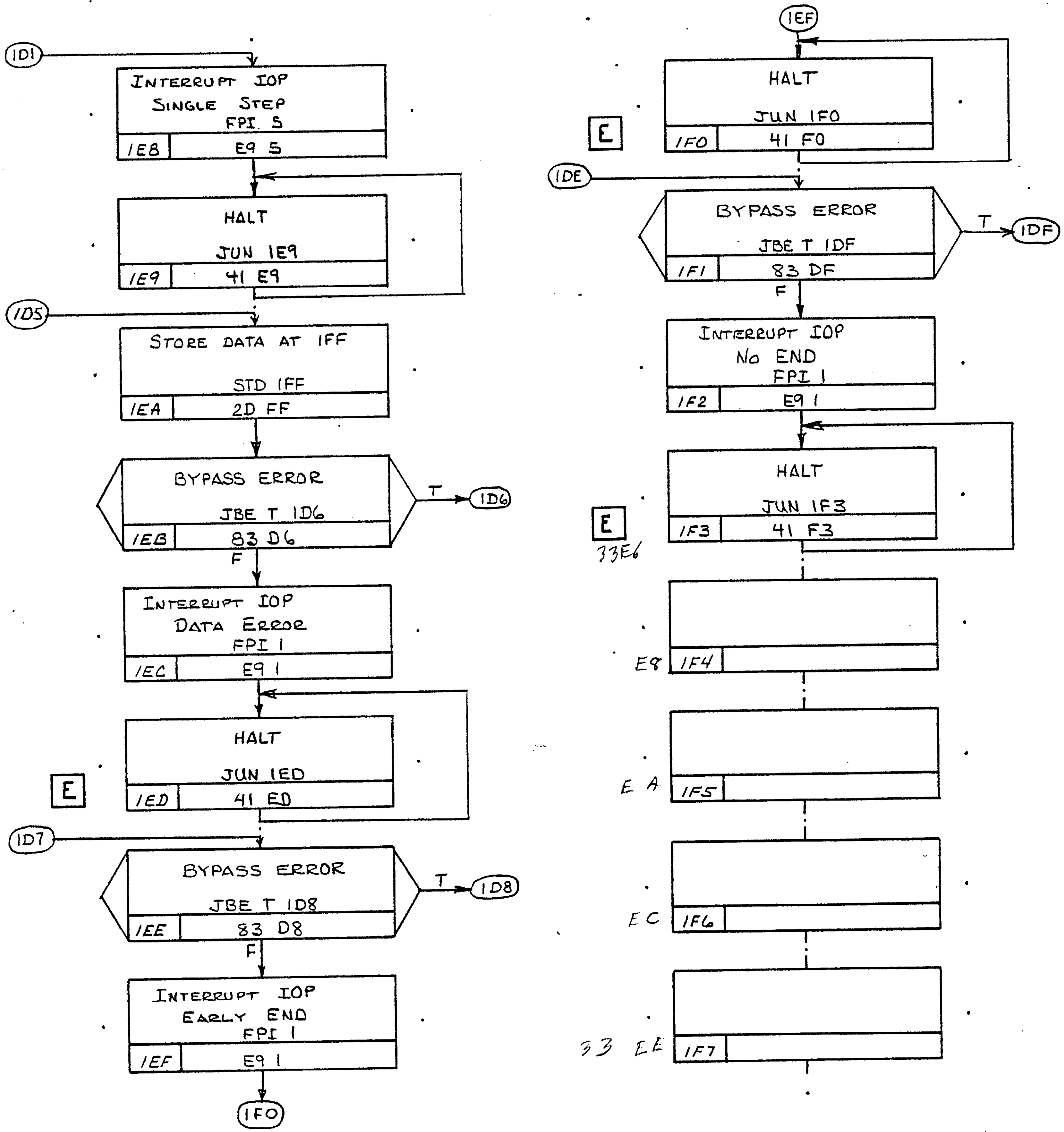
8-1-78 CPB, CPR, CPS SUBROUTINE PAGE 1 OF 3



MODEL 488

CPB, CPR, CPS SUBROUTINE

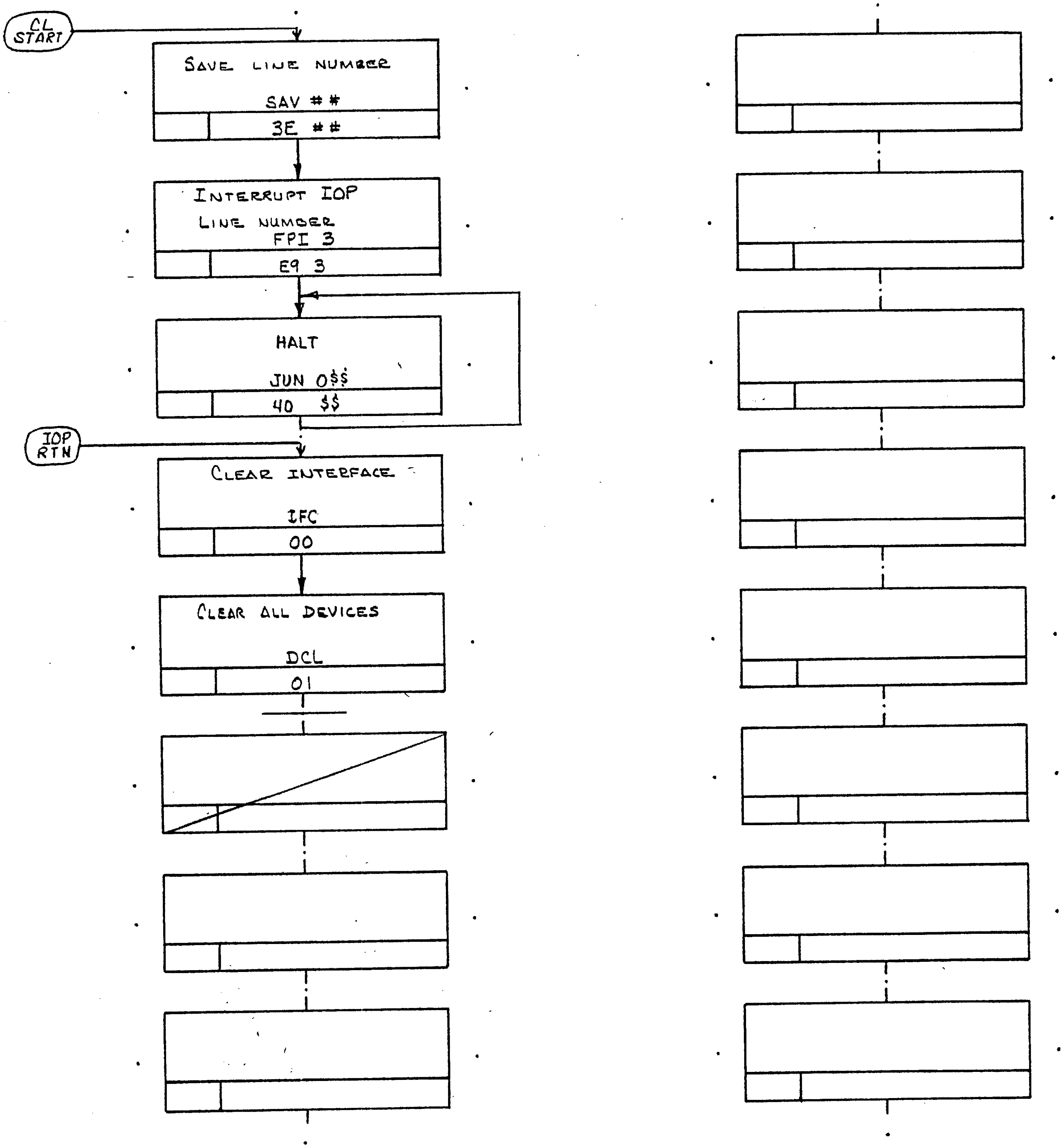



1D1
INTERRUPT IOP
SINGLE STEP
FPI 5
1E8
E9 5
HALT
JUN 1E9
1E9
41 E9
1D5
STORE DATA AT 1FF
STD 1FF
1EA
2D FF
BYPASS ERROR
JBE T 1D6
1EB
83 D6
T
1D6
F
INTERRUPT IOP
DATA ERROR
FPI 1
1EC
E9 1
HALT
JUN 1ED
E
1ED
41 ED
1D7
BYPASS ERROR
JBE T 1D8
1EE
83 D8
T
1D8
F
INTERRUPT IOP
EARLY END
FPI 1
1EF
E9 1
1F0
1EF
HALT
JUN 1F0
E
1F0
41 F0
1DE
BYPASS ERROR
JBE T 1DF
1F1
83 DF
T
1DF
F
INTERRUPT IOP
NO END
FPI 1
1F2
E9 1
HALT
JUN 1F3
E
33E6
1F3
41 F3
E8
1F4
E A
1F5
E C
1F6
33 EE
1F7

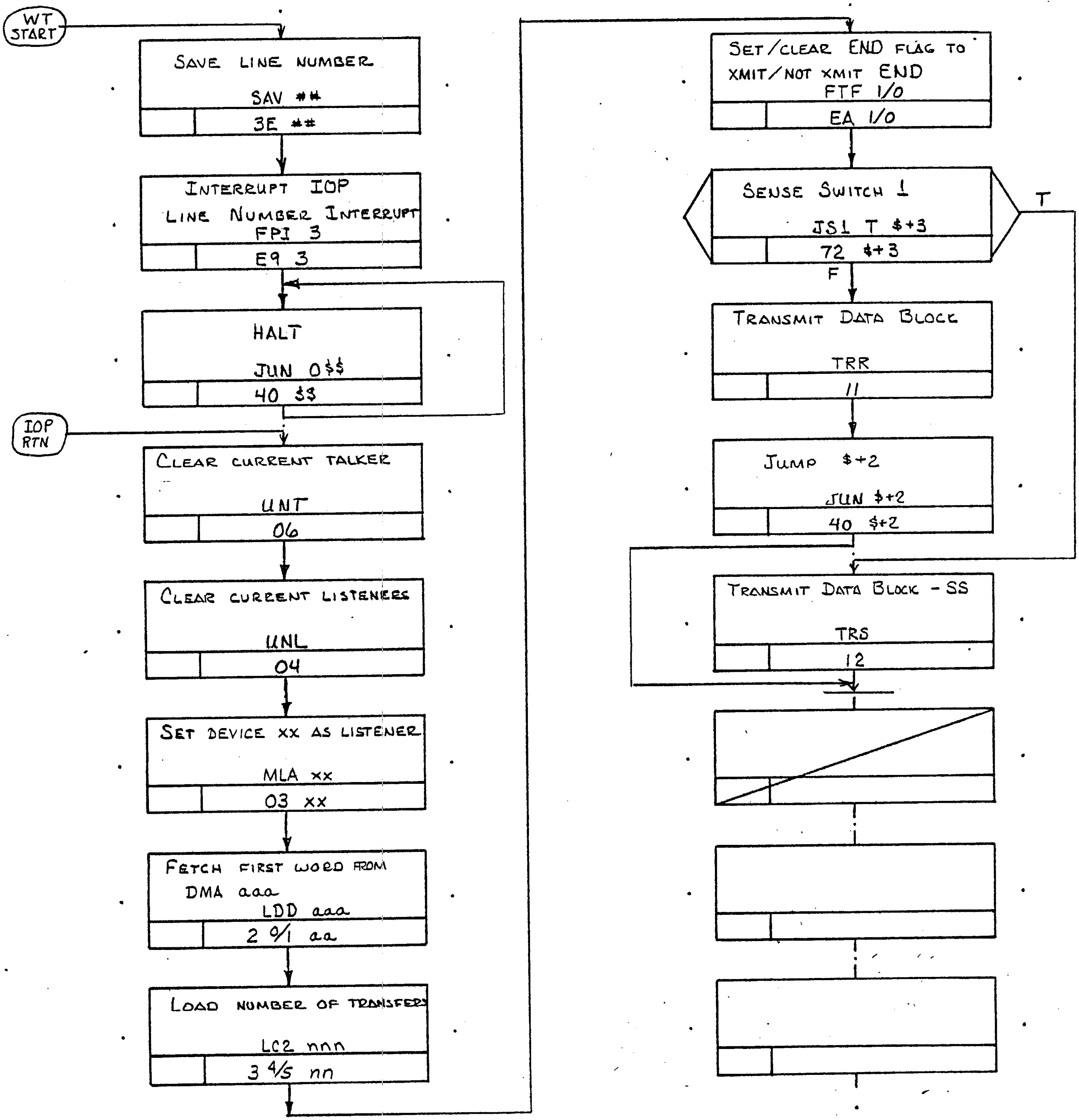
APPENDIX K

FLOWCHARTS FOR BUS LANGUAGE INSTRUCTIONS

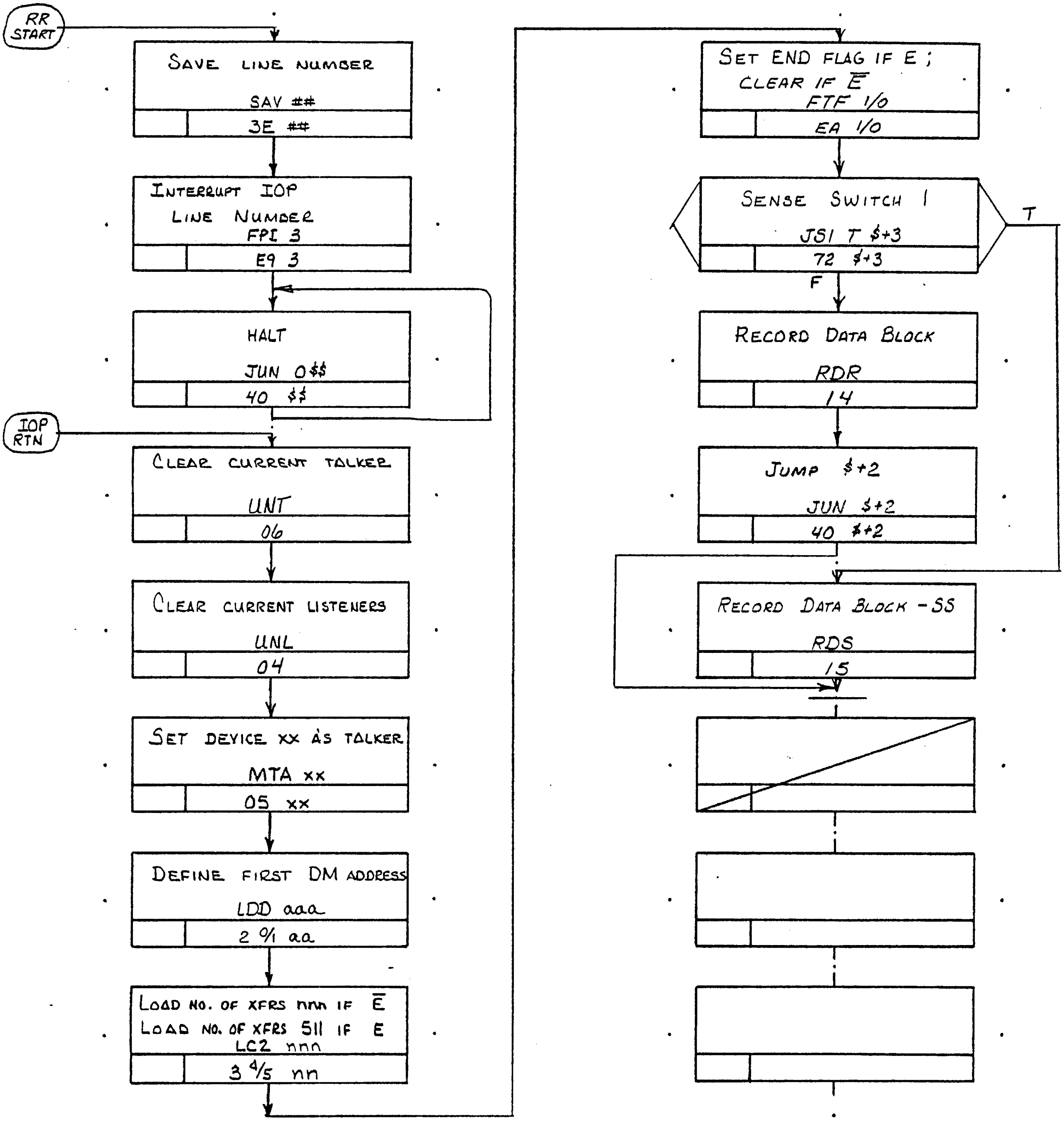
CL


CL START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP
LINE NUMBER
FPI 3
E9 3
HALT
JUN 0$$
40 $$
IOP RTN
CLEAR INTERFACE
IFC
00
CLEAR ALL DEVICES
DCL
01

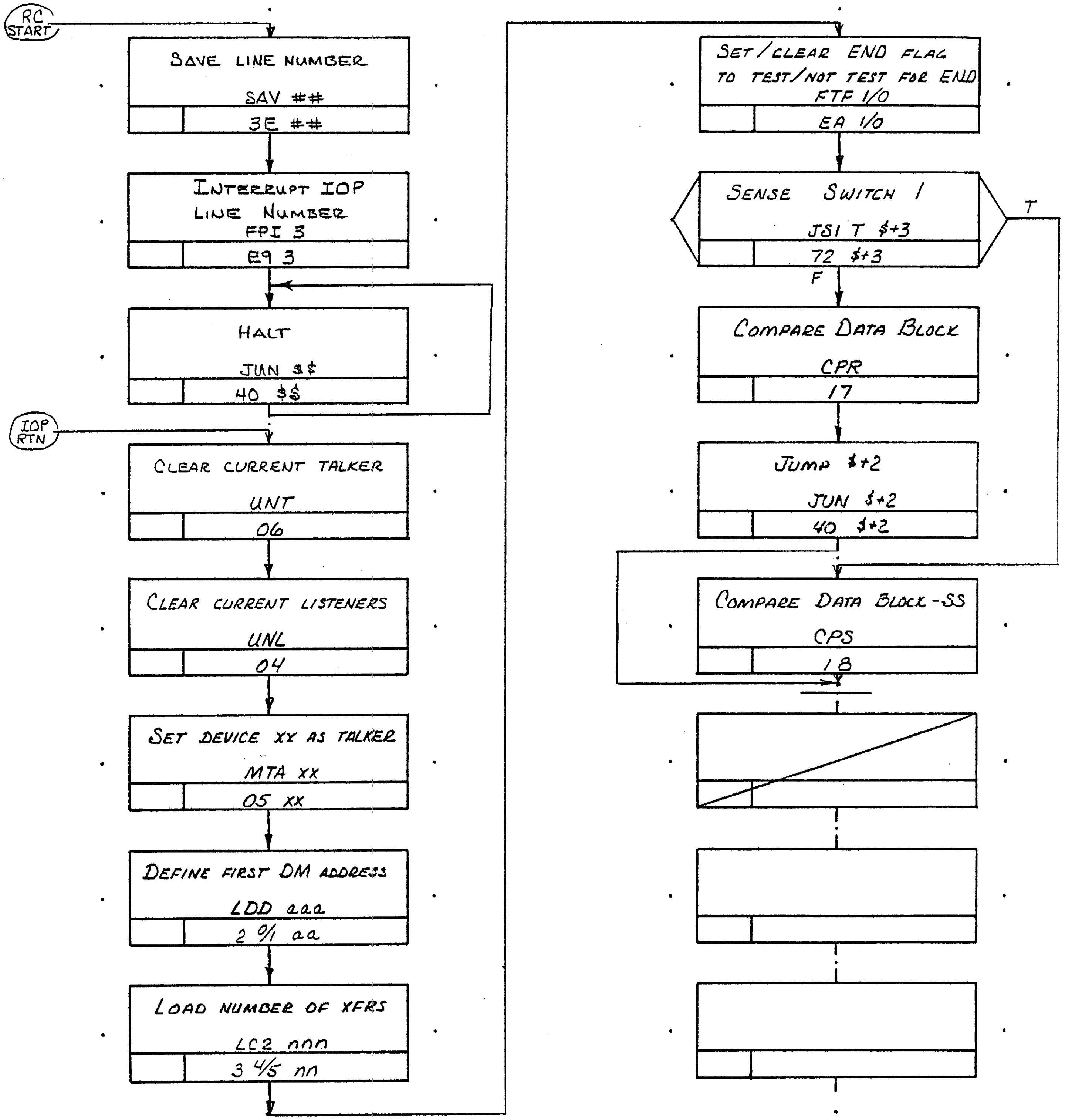
WT xx aaa nnnE



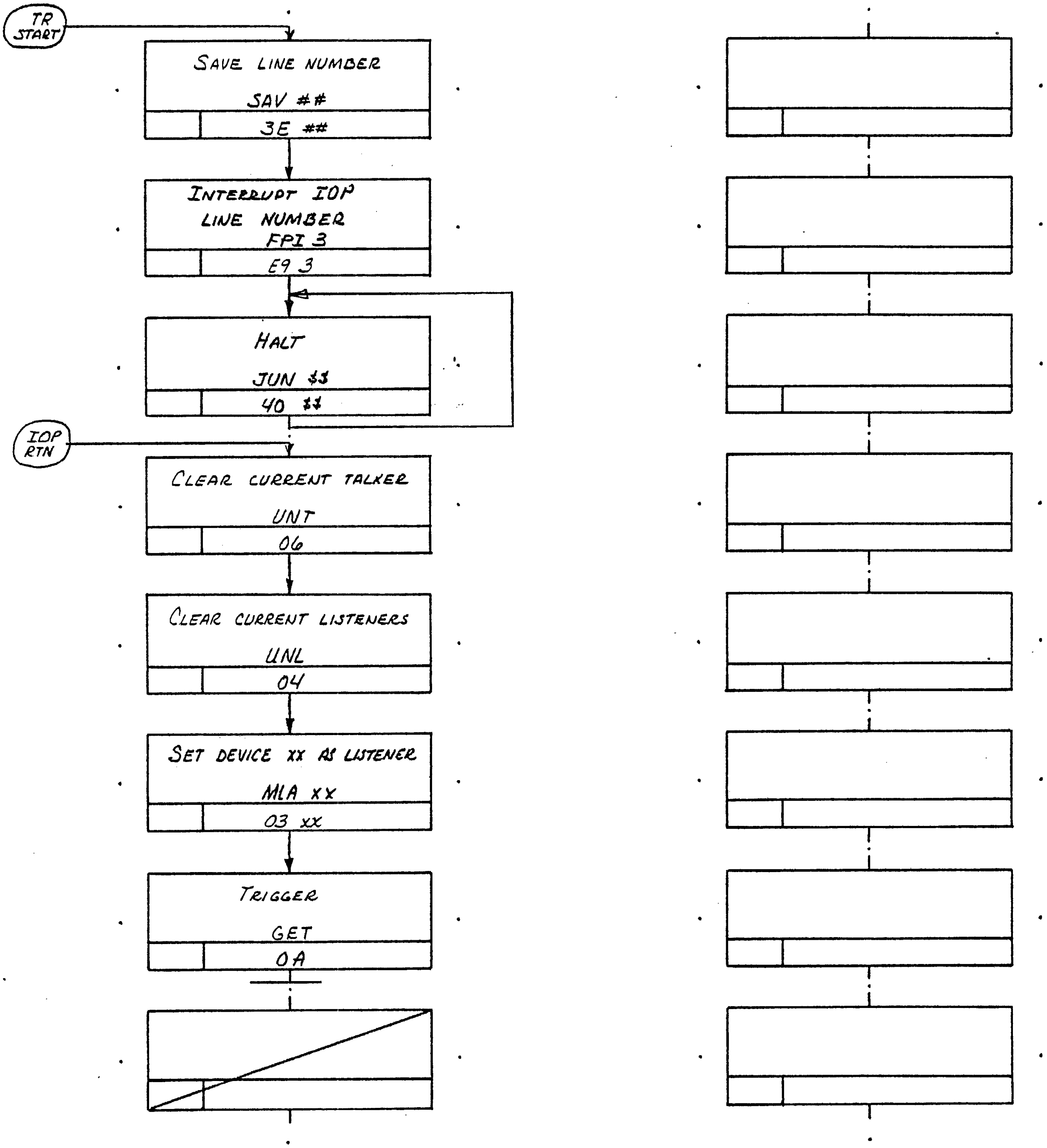
RR xx aaa $\genfrac{}{}{0pt}{}{nnn}{E}$



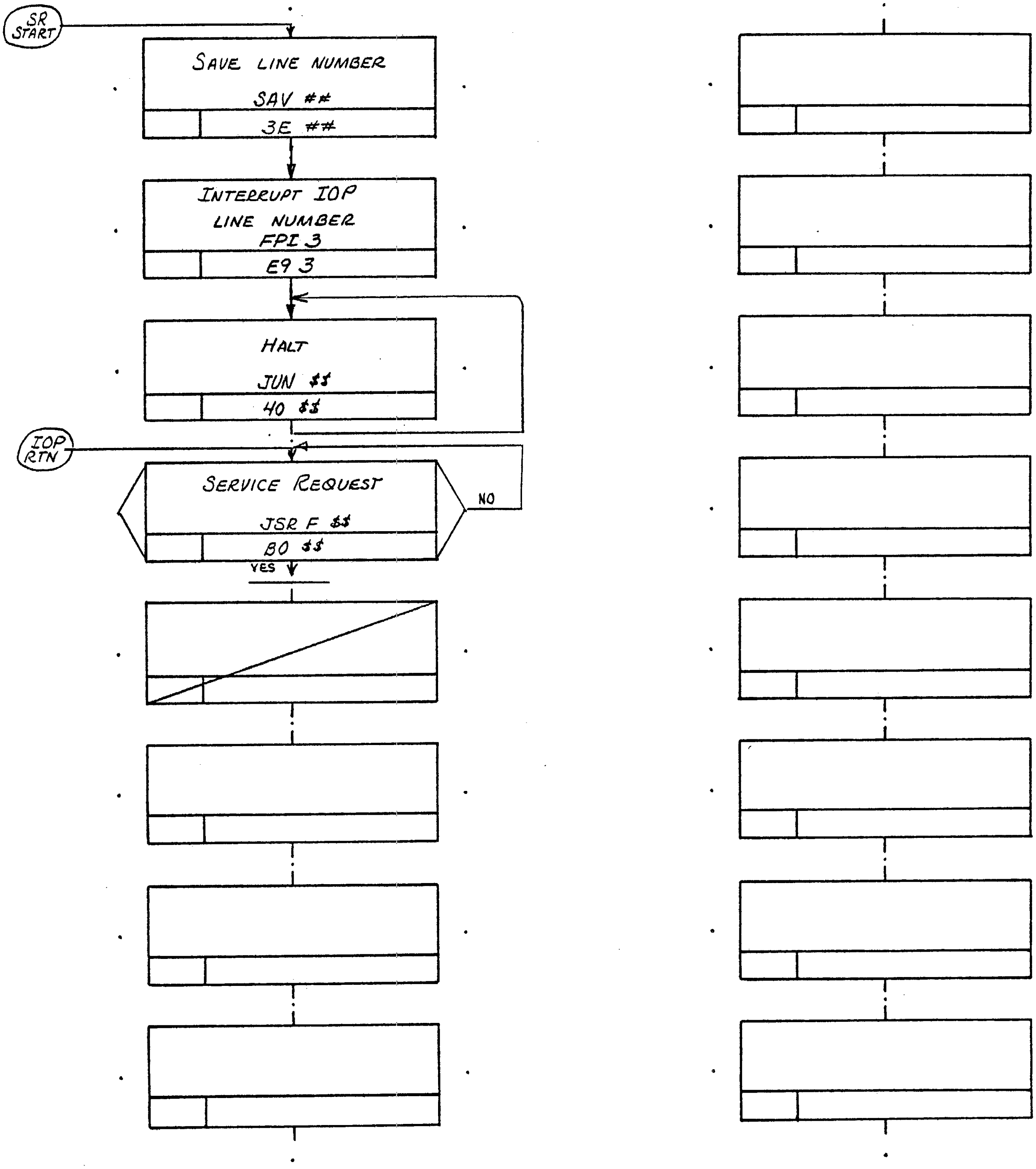
RC xx aaa nnnE



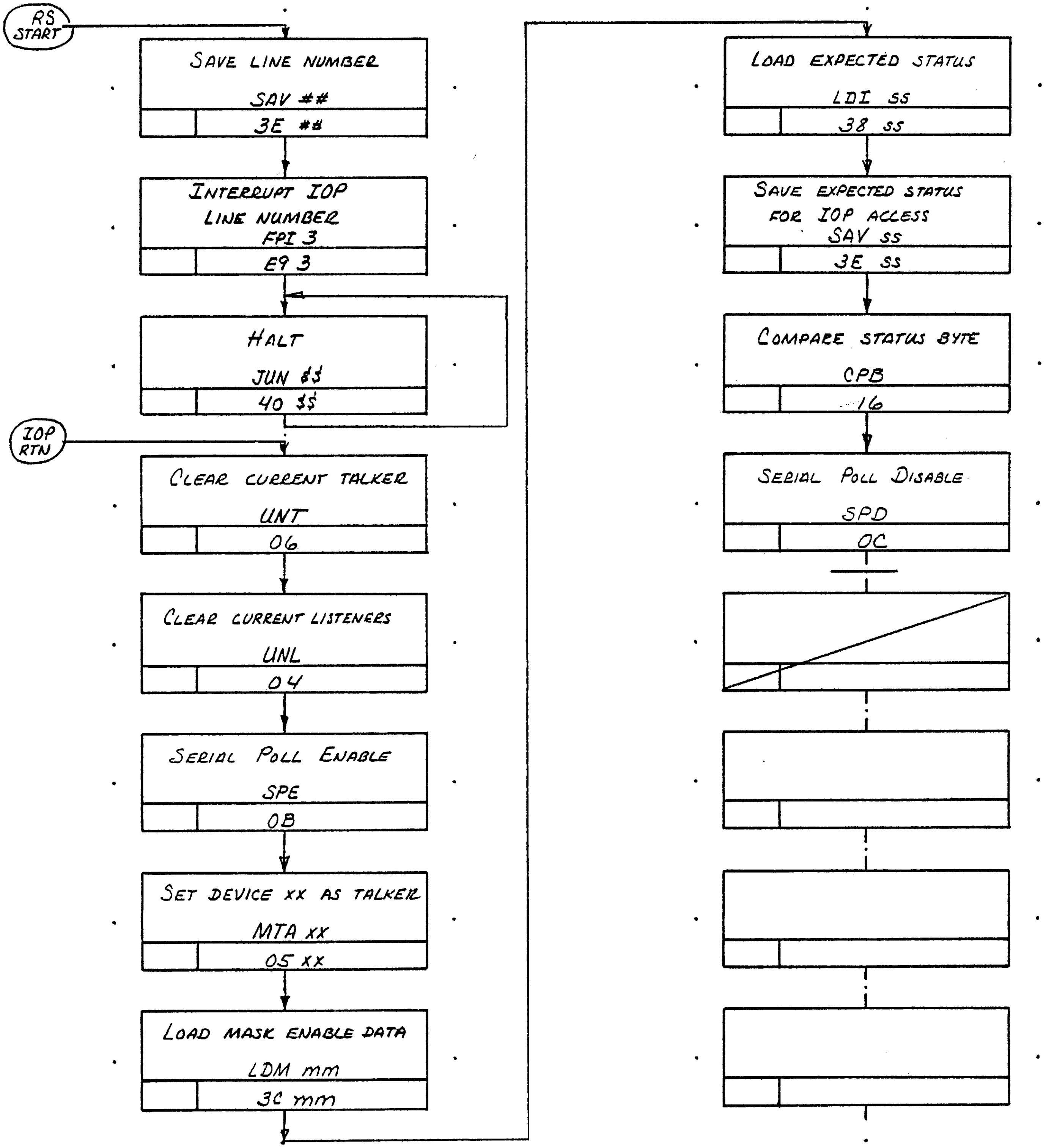
TR xx


TR START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP LINE NUMBER
FPI 3
E9 3
HALT
JUN $$
40 $$
IOP RTN
CLEAR CURRENT TALKER
UNT
06
CLEAR CURRENT LISTENERS
UNL
04
SET DEVICE XX AS LISTENER
MLA XX
03 XX
TRIGGER
GET
0A

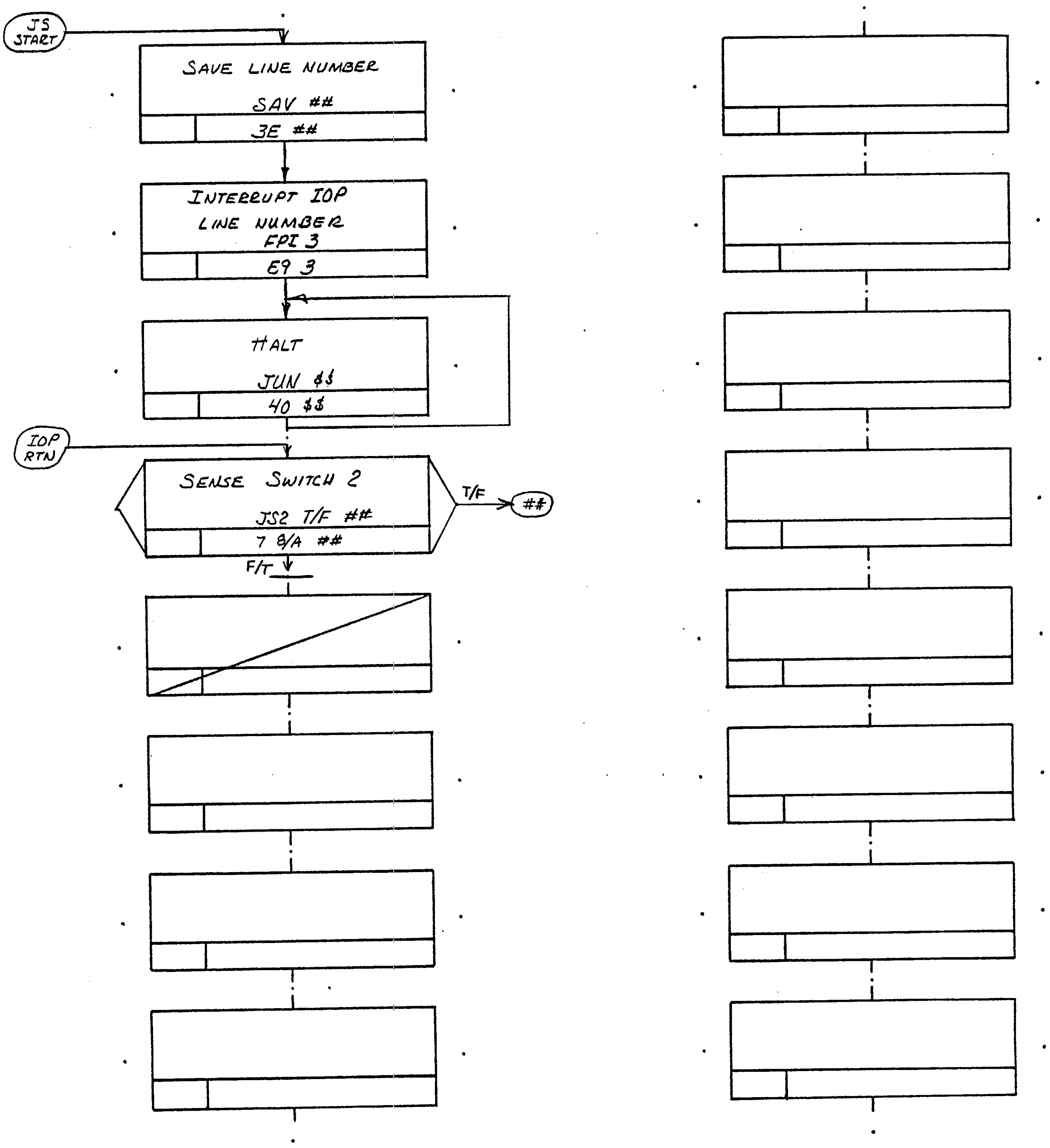
SR


SR START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP
LINE NUMBER
FPI 3
E9 3
HALT
JUN $$
40 $$
IOP RTN
SERVICE REQUEST
JSR F $$
80 $$
NO
YES

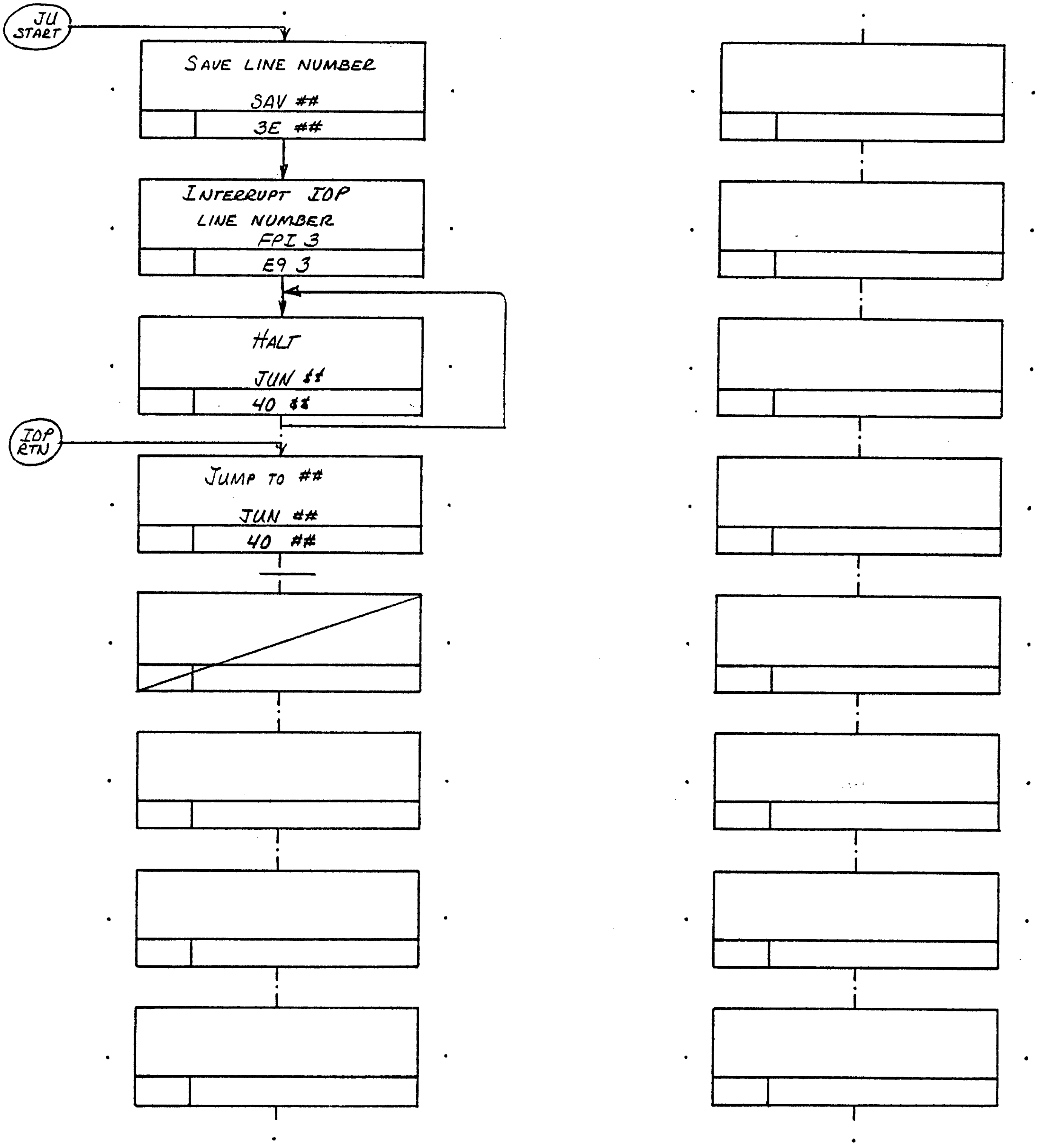
RS xx mm ss


RS START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP
LINE NUMBER
FPI 3
E9 3
HALT
JUN $$
40 $$
IOP RTN
CLEAR CURRENT TALKER
UNT
06
CLEAR CURRENT LISTENERS
UNL
04
SERIAL POLL ENABLE
SPE
0B
SET DEVICE XX AS TALKER
MTA XX
05 XX
LOAD MASK ENABLE DATA
LDM mm
3C mm
LOAD EXPECTED STATUS
LDI ss
38 ss
SAVE EXPECTED STATUS
FOR IOP ACCESS
SAV ss
3E ss
COMPARE STATUS BYTE
CPB
16
SERIAL POLL DISABLE
SPD
0C

JS t


JS START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP
LINE NUMBER
FPI 3
E9 3
HALT
JUN $$
40 $$
IOP RTN
SENSE SWITCH 2
JS2 T/F ##
7 8/A ##
T/F
##
F/T

JU

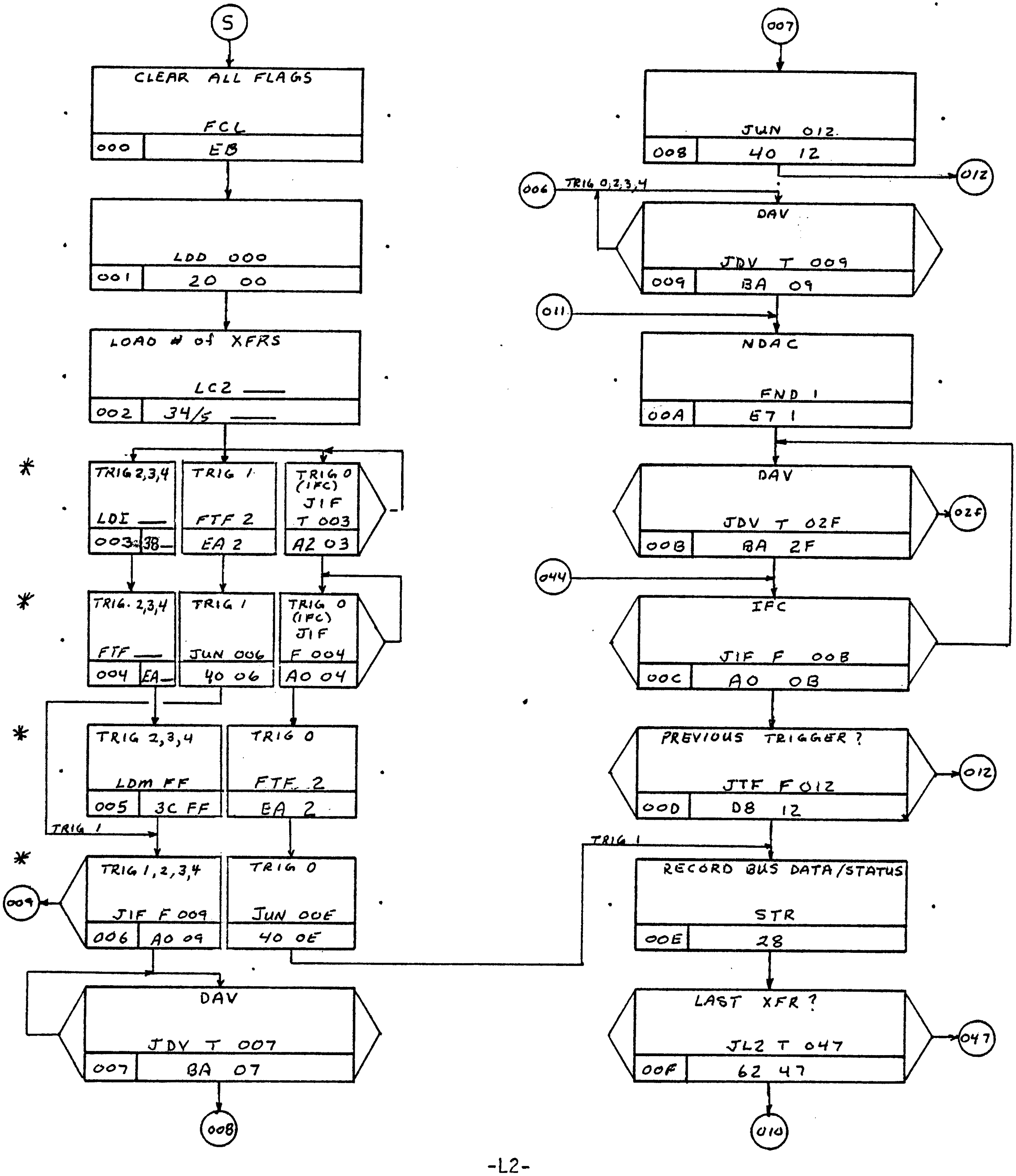

JU START
SAVE LINE NUMBER
SAV ##
3E ##
INTERRUPT IOP
LINE NUMBER
FPI 3
E9 3
HALT
JUN $$
40 $$
IOP RTN
JUMP TO ##
JUN ##
40 ##

APPENDIX L

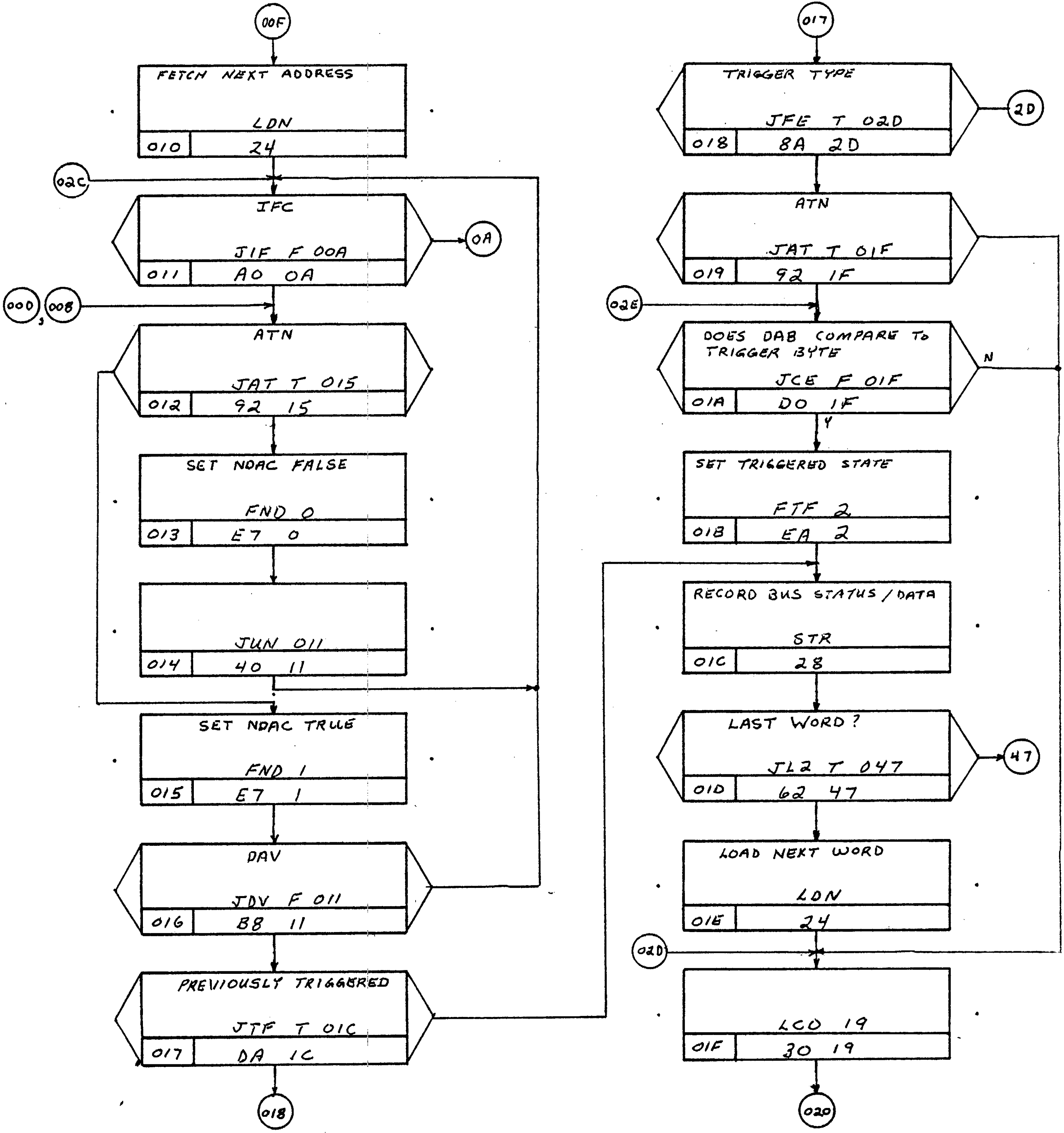
FLOWCHARTS
FOR
MONITOR MODE PROGRAM

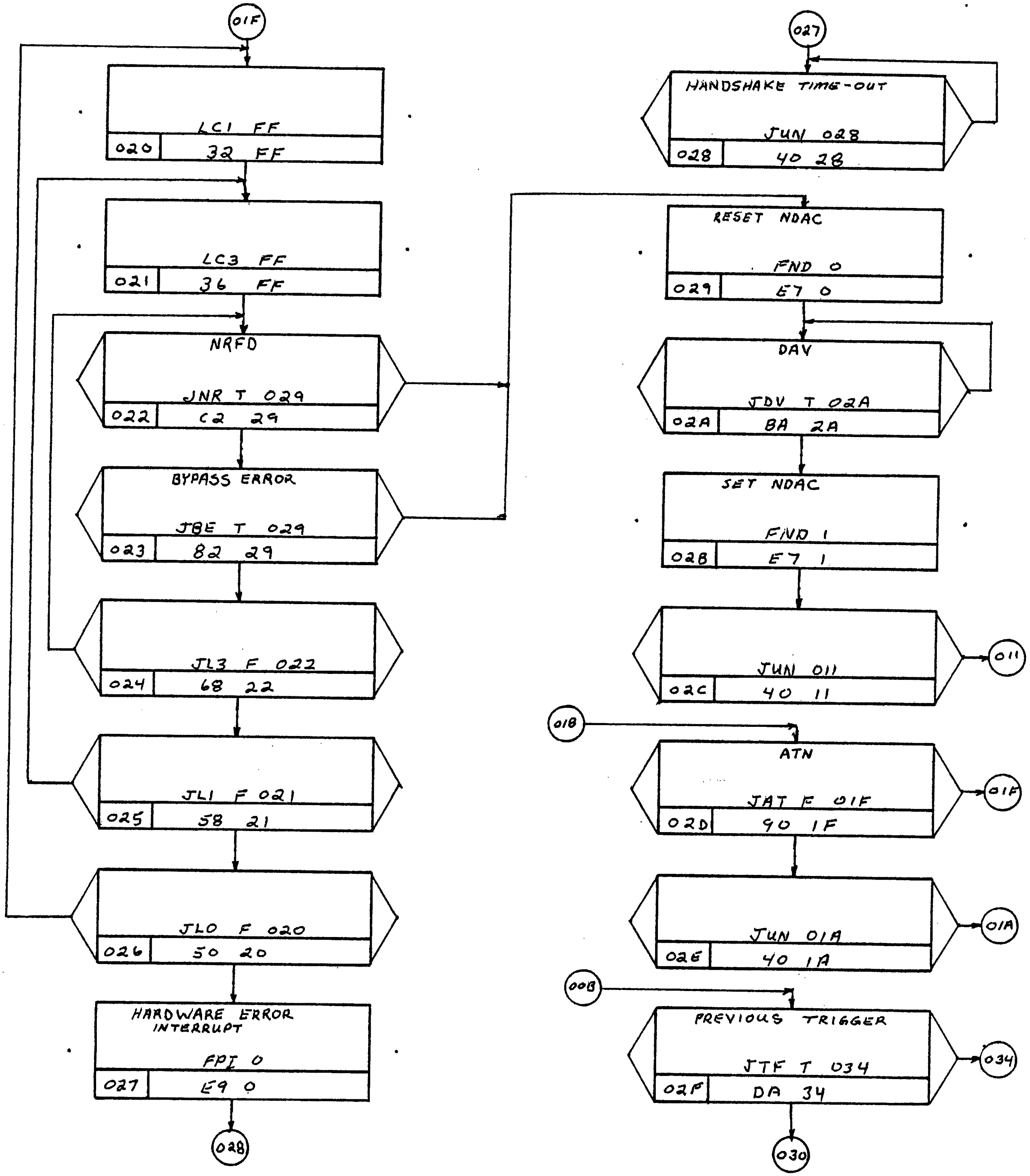
MODEL 488

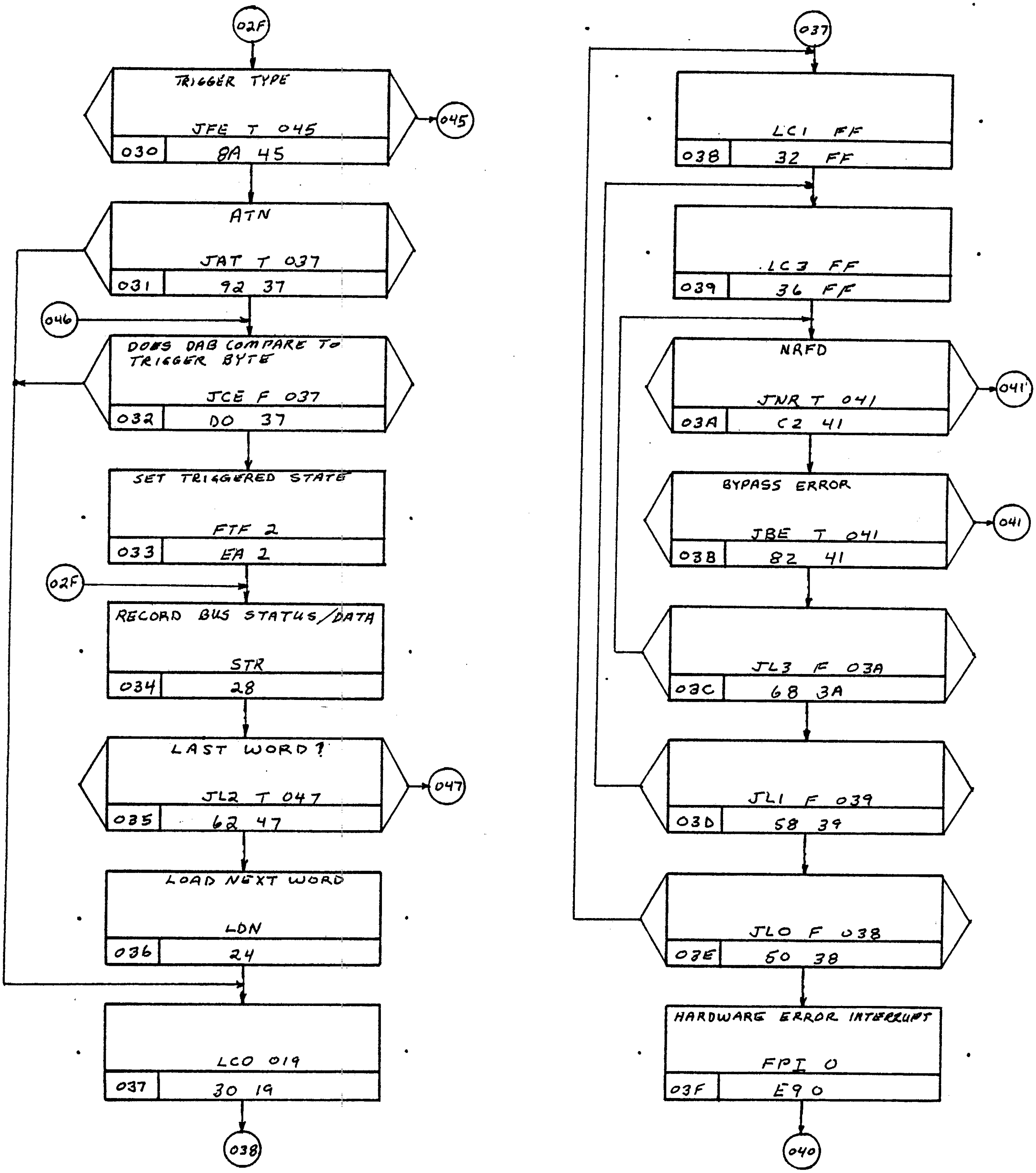
10-13-78 MONITOR SUBROULTINE PAGE 1 of 5

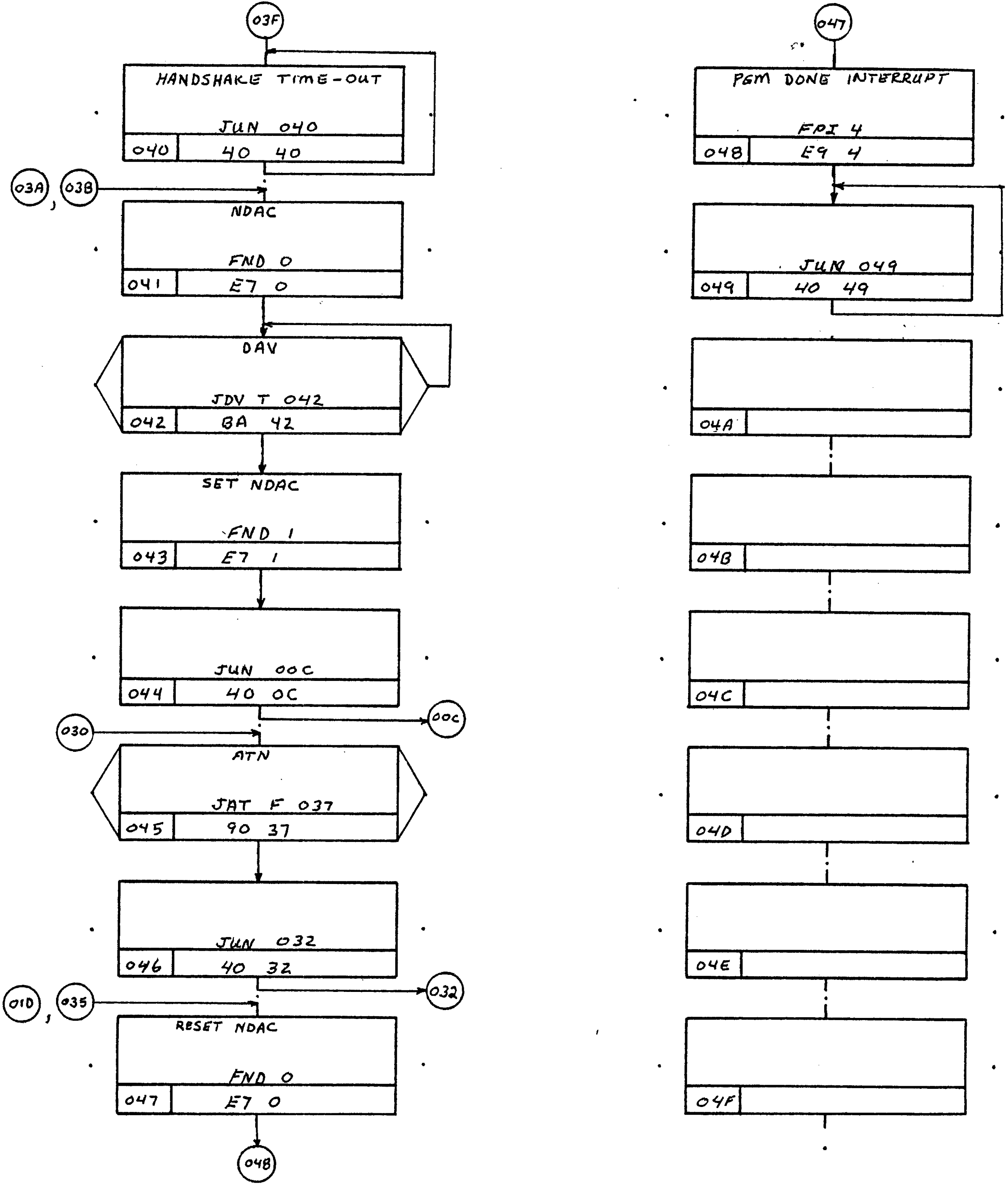


* INSTRUCTIONS UNDER 6800 SOLFTWARE CONTROL


00F
FETCH NEXT ADDRESS
LDN
010
24
02C
IFC
JIF F 00A
011
A0 0A
0A
000
008
ATN
JAT T 015
012
92 15
SET NDAC FALSE
FND 0
013
E7 0
JUN 011
014
40 11
SET NDAC TRUE
FND 1
015
E7 1
DAV
JDV F 011
016
B8 11
PREVIOUSLY TRIGGERED
JTF T 01C
017
DA 1C
018
017
TRIGGER TYPE
JFE T 02D
018
8A 2D
2D
ATN
JAT T 01F
019
92 1F
02E
DOES DAB COMPARE TO TRIGGER BYTE
JCE F 01F
01A
D0 1F
N
Y
SET TRIGGERED STATE
FTF 2
01B
EA 2
RECORD BUS STATUS / DATA
STR
01C
28
LAST WORD?
JL2 T 047
01D
62 47
47
LOAD NEXT WORD
LDN
01E
24
02D
LCO 19
01F
30 19
020


01F
LC1 FF
020
32 FF
LC3 FF
021
36 FF
NRFD
JNR T 029
022
C2 29
BYPASS ERROR
JBE T 029
023
82 29
JL3 F 022
024
68 22
JL1 F 021
025
58 21
JLO F 020
026
50 20
HARDWARE ERROR INTERRUPT
FPI 0
027
E9 0
028
027
HANDSHAKE TIME-OUT
JUN 028
028
40 28
RESET NDAC
FND 0
029
E7 0
DAV
JDV T 02A
02A
BA 2A
SET NDAC
FND 1
02B
E7 1
JUN 011
02C
40 11
011
018
ATN
JAT F 01F
02D
90 1F
01F
JUN 01A
02E
40 1A
01A
00B
PREVIOUS TRIGGER
JTF T 034
02F
DA 34
034
030


02F
TRIGGER TYPE
JFE T 045
030
8A 45
045
ATN
JAT T 037
031
92 37
046
DOES DAB COMPARE TO TRIGGER BYTE
JCE F 037
032
D0 37
SET TRIGGERED STATE
FTF 2
033
EA 2
02F
RECORD BUS STATUS/DATA
STR
034
28
LAST WORD?
JL2 T 047
035
62 47
047
LOAD NEXT WORD
LDN
036
24
LC0 019
037
30 19
038
037
LC1 FF
038
32 FF
LC3 FF
039
36 FF
NRFD
JNR T 041
03A
C2 41
041
BYPASS ERROR
JBE T 041
03B
82 41
041
JL3 F 03A
03C
68 3A
JL1 F 039
03D
58 39
JL0 F 038
03E
50 38
HARDWARE ERROR INTERRUPT
FPI 0
03F
E9 0
040


03F
HANDSHAKE TIME-OUT
JUN 040
040
40 40
03A
03B
NDAC
FND 0
041
E7 0
DAV
JDV T 042
042
BA 42
SET NDAC
FND 1
043
E7 1
JUN 00C
044
40 0C
00C
030
ATN
JAT F 037
045
90 37
JUN 032
046
40 32
032
01D
035
RESET NDAC
FND 0
047
E7 0
04B
047
PGM DONE INTERRUPT
FPI 4
048
E9 4
JUN 049
049
40 49
04A
04B
04C
04D
04E
04F

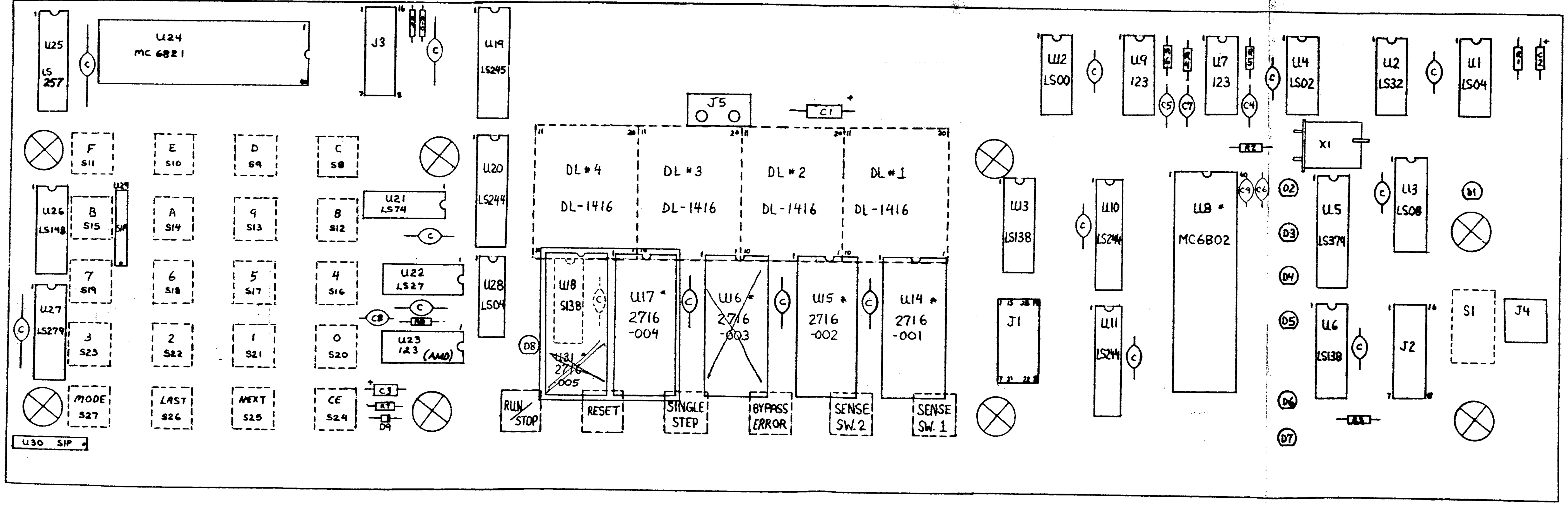
APPENDIX M

MODEL 488 DRAWINGS

Title	Drawing No.
Top Assembly	1001 1126
Front Panel Assy/LBD	1001 1125
Lower Panel Assy/LBD	1001 1128
Stored Program Card Assy/LBD	1001 1130
IEEE Interface Card Assy/LBD	1001 1134
Serial Interface Card Assy/LBD	1001 1132
Power Supply Schematic	3024 S01

MARK-UP COPY FOR 2732 EPROM PROGRAM STORAGE

REV	REVISION	APPROVED	DATE
A			10-11-78
B	SH. 2: U7-05 FROM [illegible] TO PULL+ SH.4: ADD U31 & U18-01 TO U31-20 & U18-05 FROM [illegible] TO [illegible] & U18-08 FROM [illegible] TO A13.	EKC	1-19-79
C	PER C.O. 0046	RK	6-6-80



COMPONENT SIDE

NOTES:
1. ALL ITEMS SHOWN WITH BROKEN LINES MOUNT ON FAR SIDE OF BOARD
2. RESISTORS ARE 1/4W CARBON WITH OHM VALUES AS FOLLOWS:

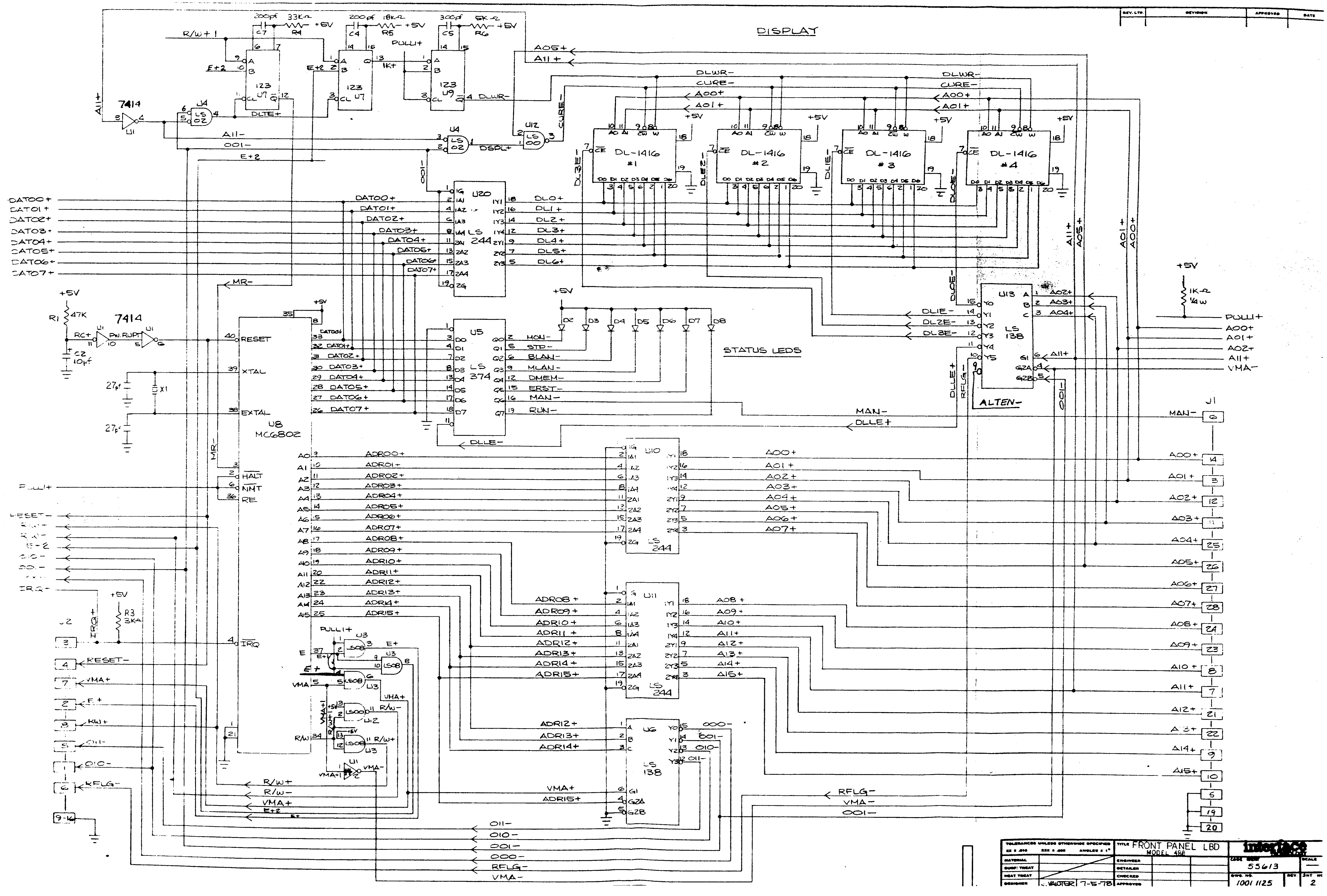
R2 1K	R4 33K	R6 5.1K	R8 15K	R10 10K	
R1 47K	R3 3K	R5 18K	R7 33K	R9 10K	SIPS 10K

3. ALL CAPACITORS, EXCEPT C1-C3, ARE 50V CERAMIC

C2 10uf	C4 200pf	C6 27pf	C8 200pf	C 0.05uf
C1 10uf	C3 1uf	C5 300pf	C7 200pf	C9 27pf

4. CAPACITORS C1-C3 ARE 10V ELECTROLYTIC
5. ALL IC'S ARE SN74XX UNLESS MARKED WITH AN ASTERISK (*)
6. DIODE D9 IS A IN914
7. REF. 10011124 FOR PCB

COVINA	interface TECHNOLOGY	CA91724
SCALE: NTS DATE: 12-18-78	APPROVED BY: EKC	DRAWN BY: GTHW REV:
FRONT LOGIC ASSY 488		
SHEET 1 OF 4	55613	DRAWING NO. 10011125



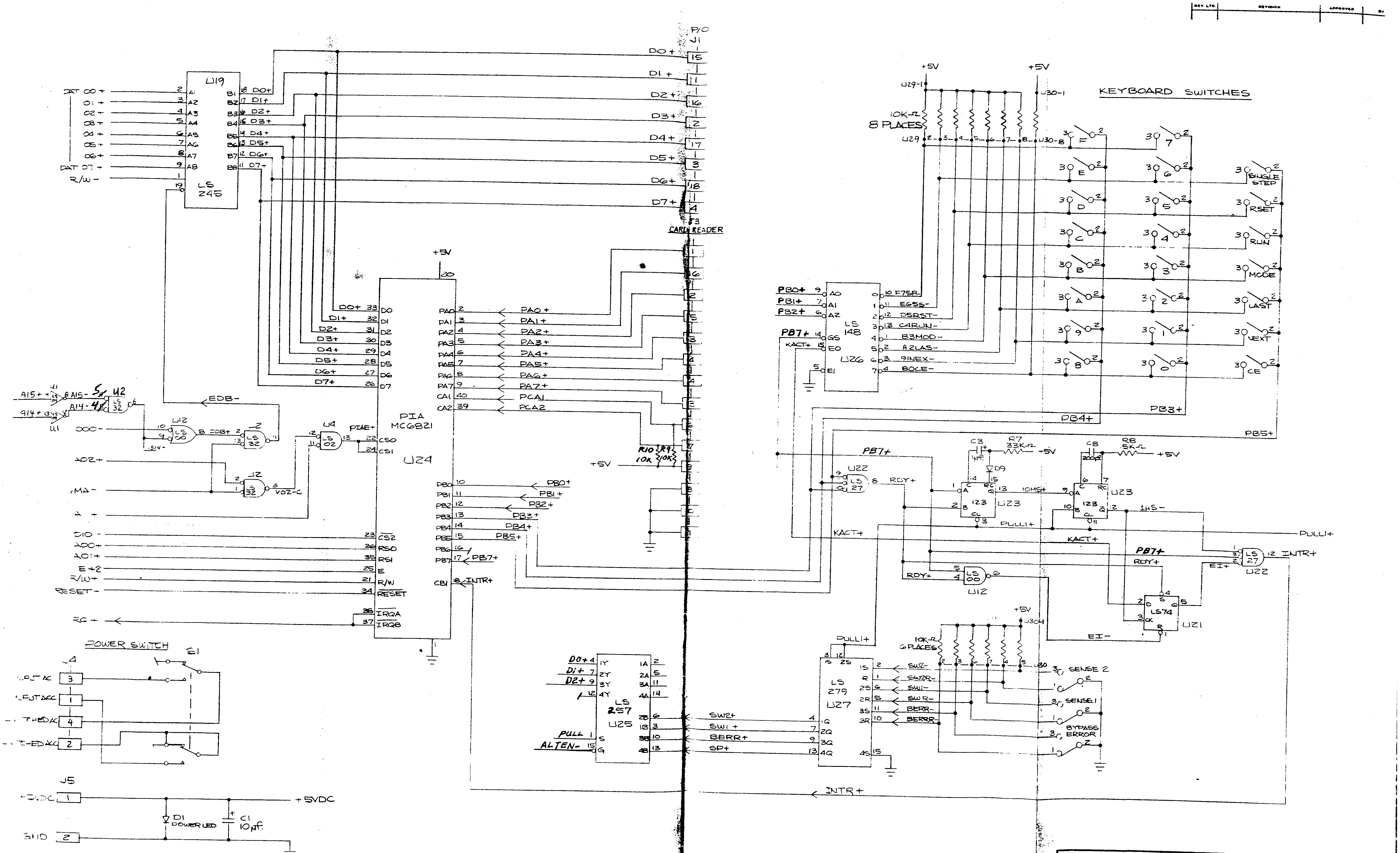
TOLERANCES UNLESS OTHERWISE SPECIFIED
XX ± .016 XXX ± .005 ANGLES ± 1°

TITLE FRONT PANEL LBD
MODEL 486

interface

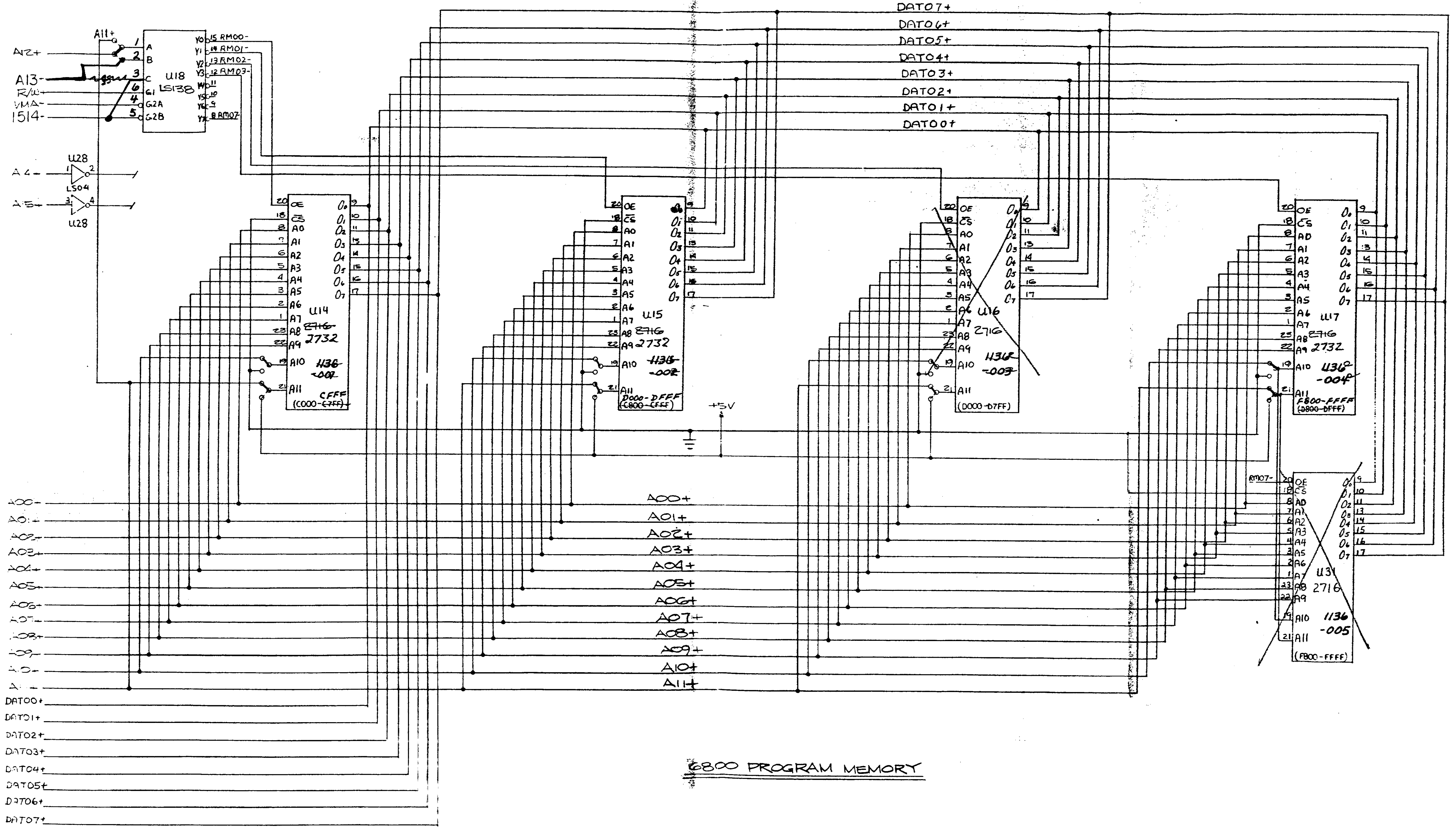
MATERIAL		ENGINEER	
SURF. TREAT		DETAILER	
HEAT TREAT		CHECKED	
DESIGNER	J. WALTER 7-5-78	APPROVED	

CODE IDENT 55613
SCALE —
DWG. NO. 1001 1125
REV.
SHT NO 2



TOLERANCES UNLESS OTHERWISE SPECIFIED XX ± .010 XXX ± .005 ANGLES ± 1°	TITLE FRONT PANEL LBD MODEL 486			interface
MATERIAL		ENGINEER		CODE IDENT 55613
SURF. TREAT		DETAILER		SCAL —
HEAT TREAT		CHECKED		DWG. NO. 1001 1125
DESIGNER	J.VAUTER 7-5-78	APPROVED		REV.

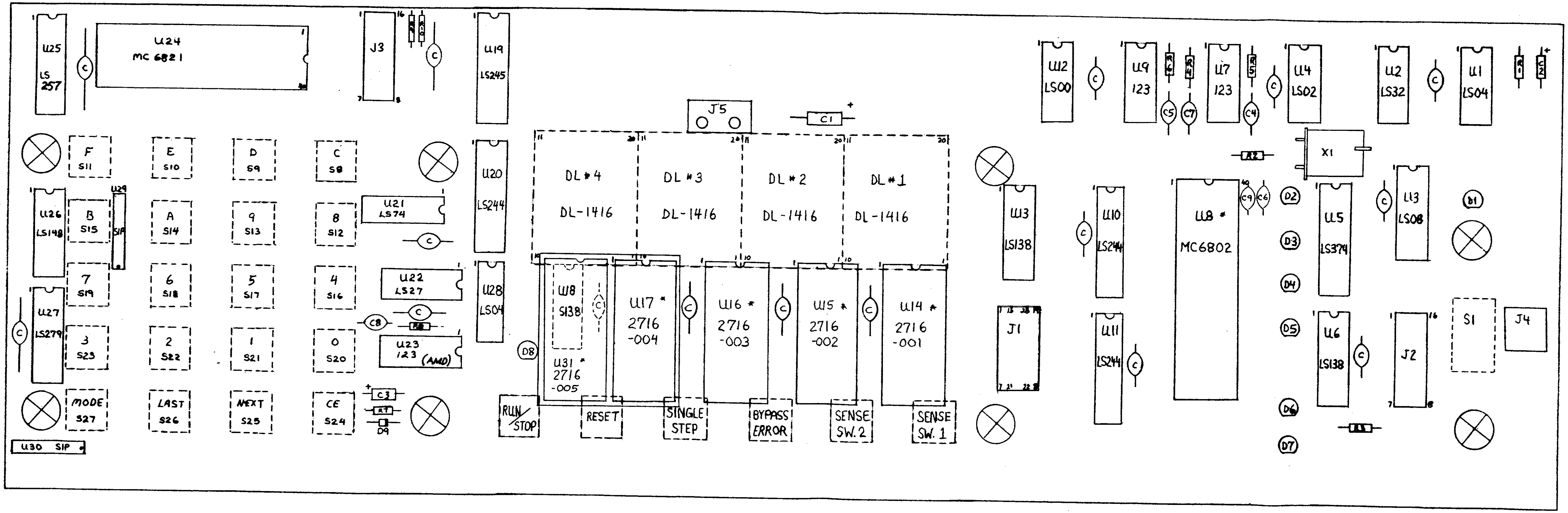
REV. LTR.	REVISION	APPROVED	DATE



6800 PROGRAM MEMORY

TOLERANCES UNLESS OTHERWISE SPECIFIED .XX ± .010 .XXX ± .005 ANGLES ± 1°	TITLE FRONT PANEL LBD MODEL 488	interface
MATERIAL	ENGINEER	CODE IDENT 55613
SURF. TREAT	DETAILER	SCALE
HEAT TREAT	CHECKED	DWG. NO. 1001125
DRAWN WAITE 7-30-79		SHT 4

REV	REVISION	APPROVED	DATE
A			10-11-78
B	SH. 2: U7-05 FROM DLTE+ TO PULL1+ SH.4: ADD U31 & U18-07 TO U31-20 & U18-05 FROM [illegible] TO [illegible]; U18-03 FROM [illegible] TO A15.	EKK	1-19-79
C	PER C.O. 0046	RK	6-6-80



COMPONENT SIDE

NOTES:

1. ALL ITEMS SHOWN WITH BROKEN LINES MOUNT ON FAR SIDE OF BOARD
2. RESISTORS ARE 1/4W CARBON WITH OHM VALUES AS FOLLOWS:

R2	1K	R4	33K	R6	5.1K	R8	15K	R10	10K	
R1	47K	R3	3K	R5	18K	R7	33K	R9	10K	SIPS 10K

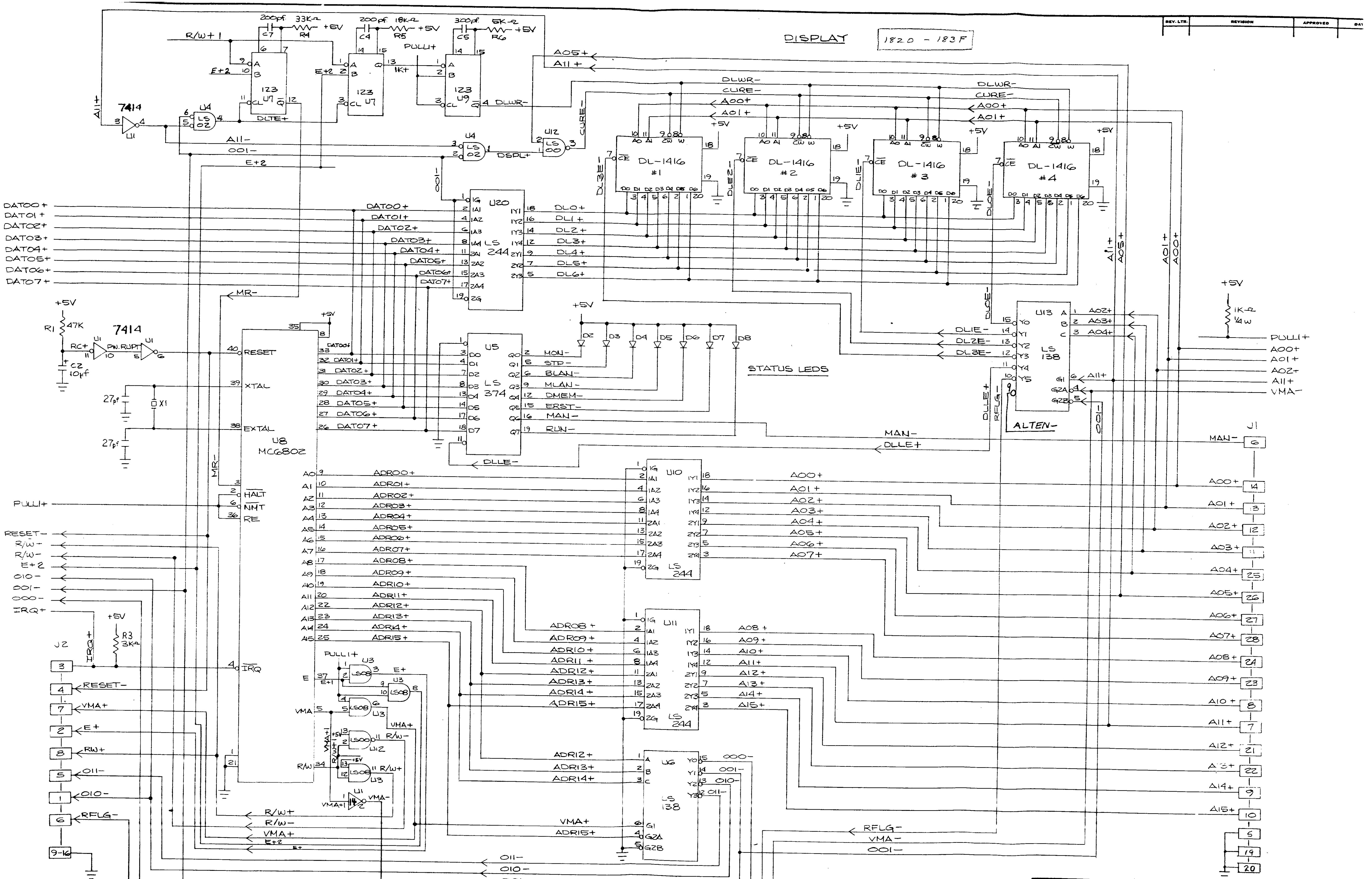
3. ALL CAPACITORS, EXCEPT C1-C3, ARE 50V CERAMIC

C2	10uf	C4	200pf	C6	27pf	C8	200pf	C	0.05uf
C1	10uf	C3	1uf	C5	300pf	C7	200pf	C9	27pf

4. CAPACITORS C1-C3 ARE 10V ELECTROLYTIC
5. ALL IC'S ARE SN74XX UNLESS MARKED WITH AN ASTERISK (*)
6. DIODE D9 IS A IN914
7. REF. 10011124 FOR PCB

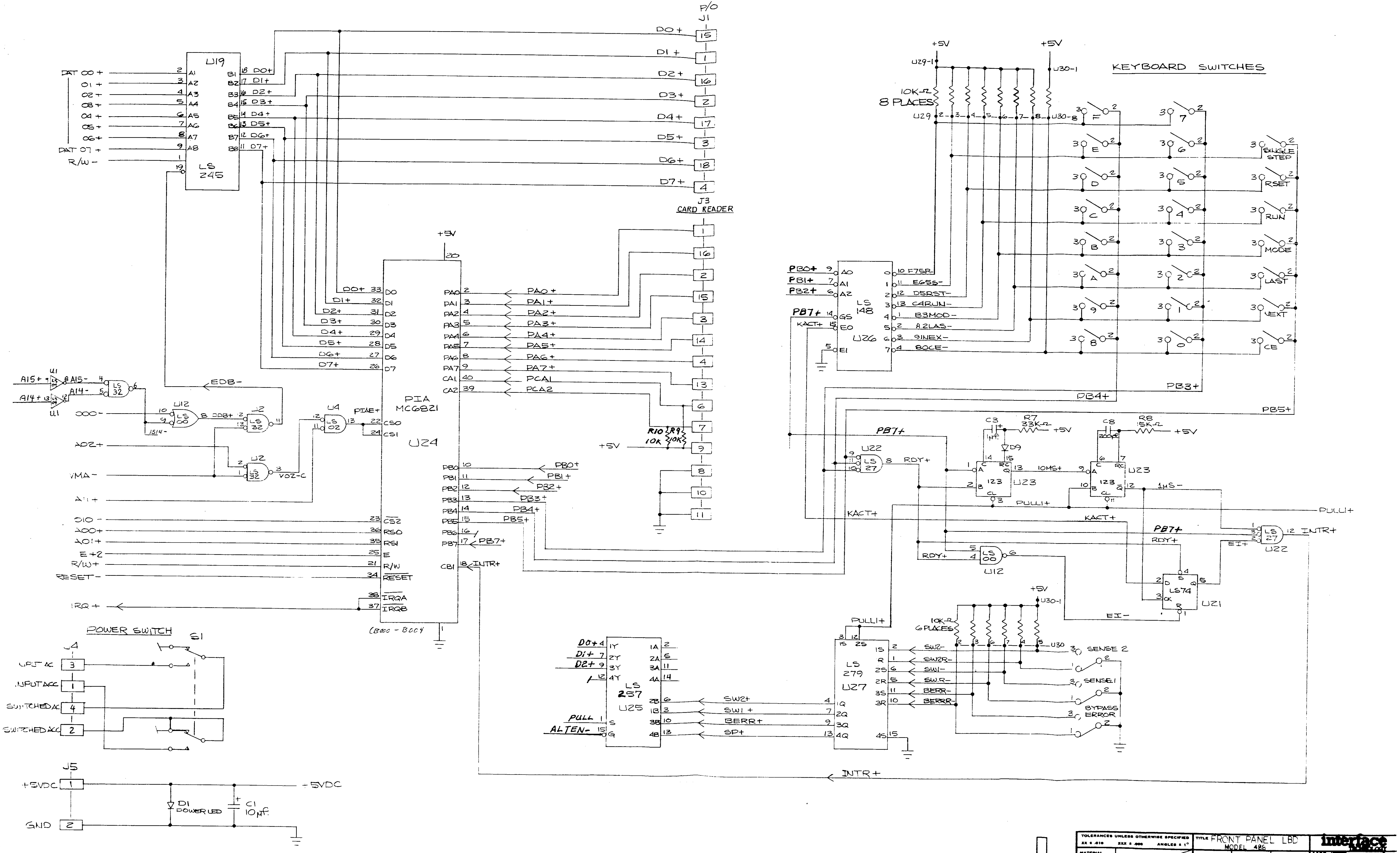
COVINA	interface TECHNOLOGY	CA91724	
SCALE: NTS	APPROVED BY: EKK	DRAWN BY: JHW	
DATE: 12-18-78		REV:	
FRONT LOGIC ASSY 488			
SHEET 1 OF 4	55613	DRAWING NO. 10011125	

REV. LTR.	REVISION	APPROVED	DA



TOLERANCES UNLESS OTHERWISE SPECIFIED XX ± .010 XXX ± .005 ANGLES ± 1°	TITLE FRONT PANEL LBD MODEL 488	interface
MATERIAL	ENGINEER	CODE IDENT 55613
SURF. TREAT	DETAILER	SCAL —
HEAT TREAT	CHECKED	DWG. NO. 1001 1125
DESIGNER J. VAUTER 7-5-78	APPROVED	REV. SHT 2

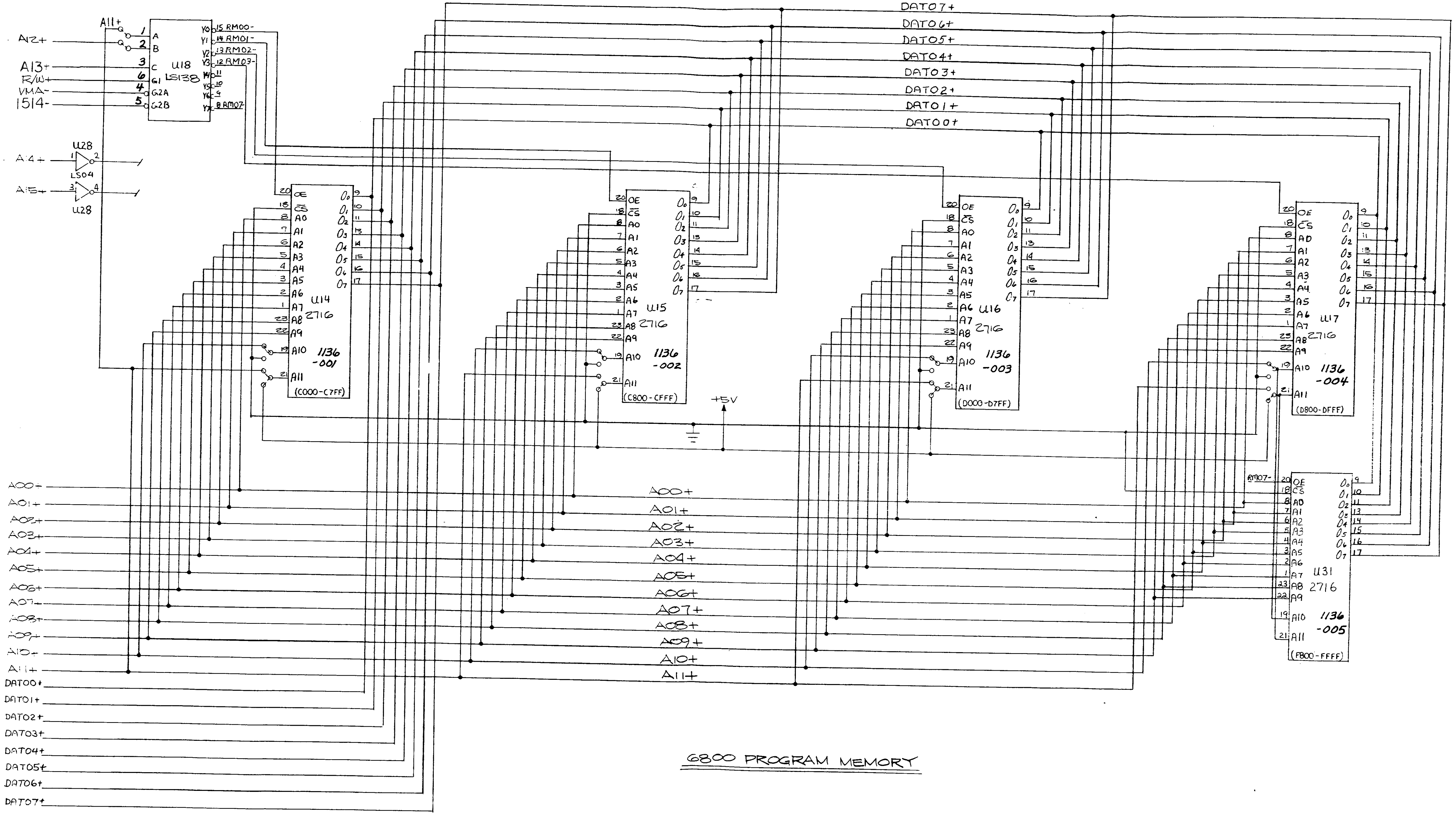
REV. LTR.	REVISION	APPROVED



TOLERANCES UNLESS OTHERWISE SPECIFIED XX ± .010 XXX ± .005 ANGLES ± 1°		TITLE FRONT PANEL LBD MODEL 486			interface
MATERIAL		ENGINEER			CODE IDENT 55613
SURF: TREAT		DETAILER			
HEAT TREAT		CHECKED			DWG. NO. 1001 1125
DESIGNER	J.VAUTER 7-5-78	APPROVED			

SCALE — REV. SHT 3

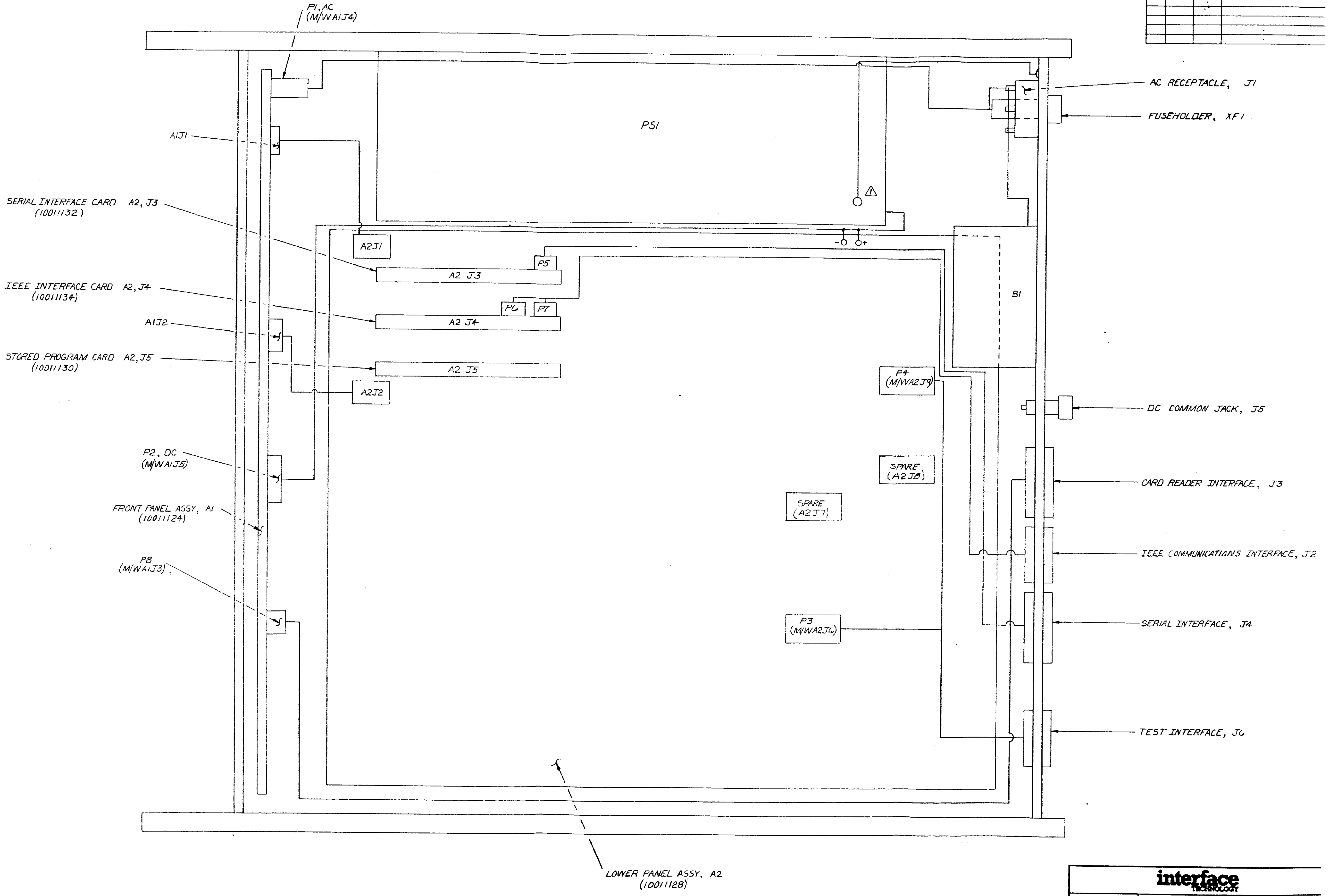
REV. LTR.	REVISION	APPROVED	DATE



6800 PROGRAM MEMORY

TOLERANCES UNLESS OTHERWISE SPECIFIED		TITLE FRONT PANEL LBD MODEL 488			interface
XX ± .010 XXX ± .005 ANGLES ± 1°					
MATERIAL		ENGINEER			CODE IDENT 55613
SURF. TREAT		DETAILER			
HEAT TREAT		CHECKED			DWG. NO. 1001 1125
DESIGNER	J. VAUTER 7-30-78	APPROVED			REV.

REVISIONS			
LTR	DATE	APPR	DESC



NOTES: ⚠1 CHASSIS GND. TO SIGNAL GND. JUMPER.

interface TECHNOLOGY

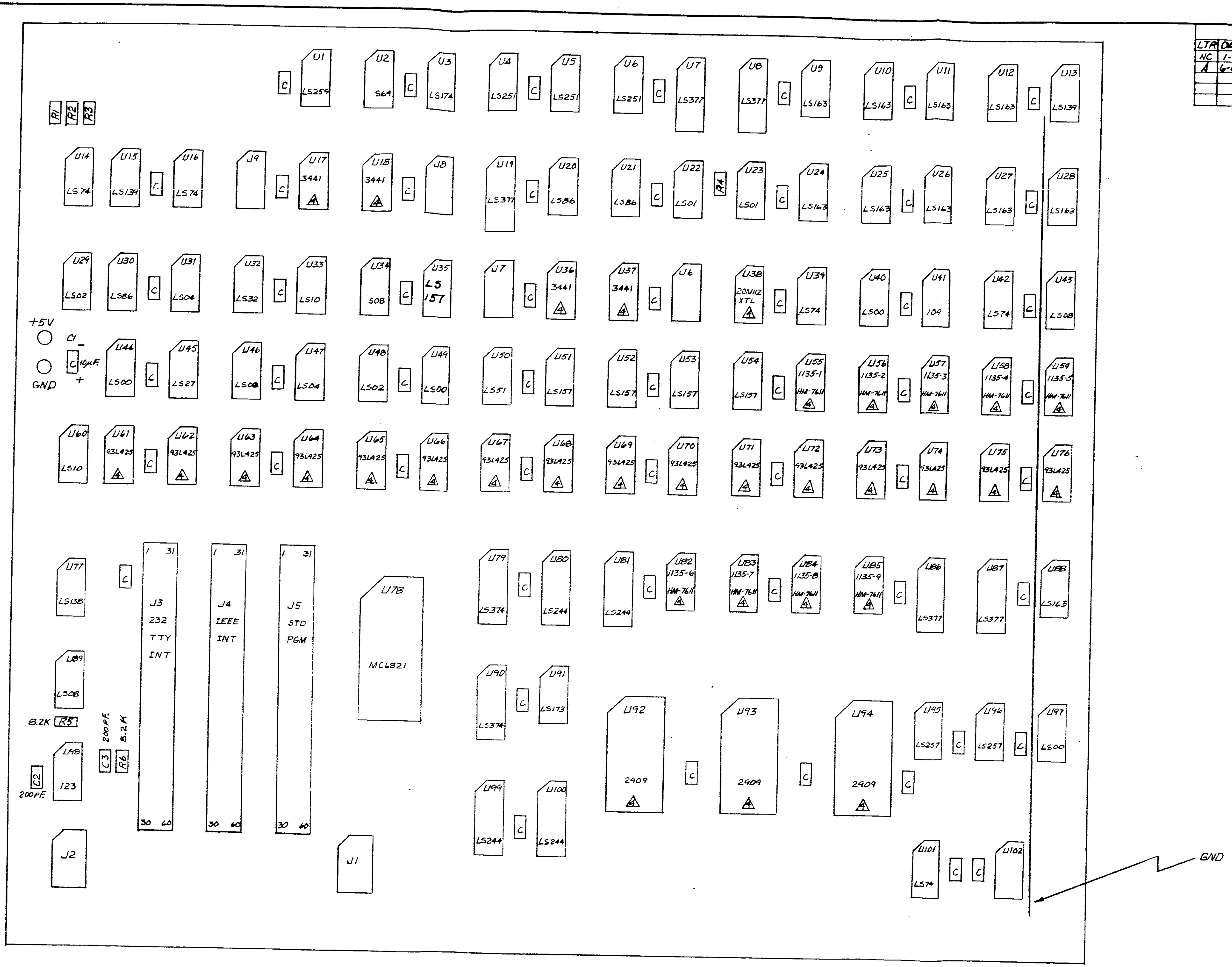
SCALE: NTS
DATE: 2-6-79
APPROVED BY:
DRAWN BY R.J.K.
REVISED

TOP ASSY, MODEL 488 1.

SHT 1 OF 1
55613
DRAWING NUMBER 10011126

PRINTED ON NO. 1000H-10 CLEARPRINT FADE-OUT

REVISIONS			
LTR	DATE	APPR	DESC
NC	1-3-9	CHK	RELEASE
A	6-6-80	RK	PER. C.O. 0032



△4 ALL INTEGRATED CIRCUITS ARE SN74XX UNLESS NOTED.

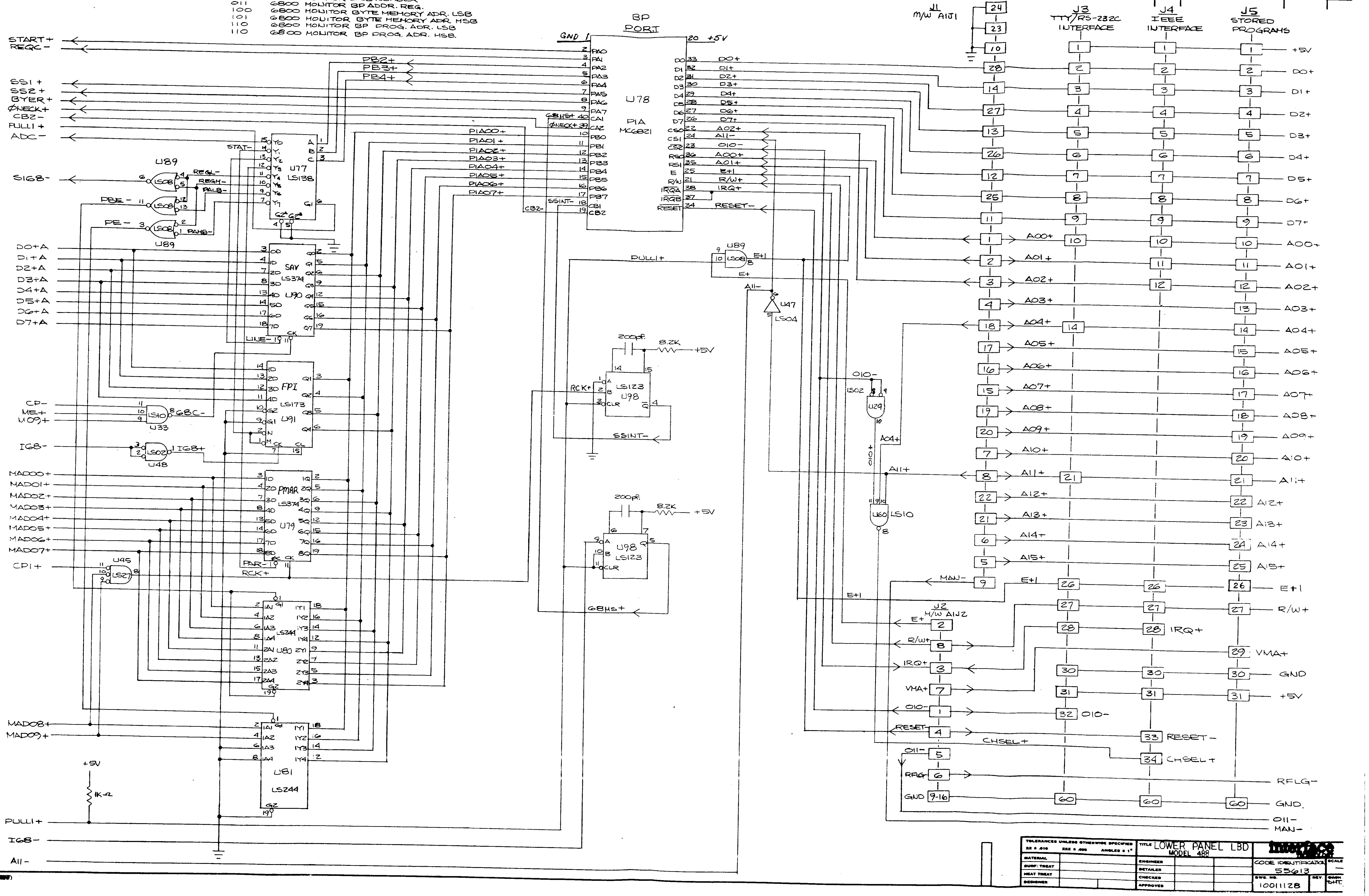
3. REF. 10011127 FOR PCB.

2. ALL CAPACITORS ARE 0.05µF. 50V. CERAMIC UNLESS NOTED.

NOTES: 1. ALL RESISTORS ARE 1K-¼W CARBON UNLESS NOTED.

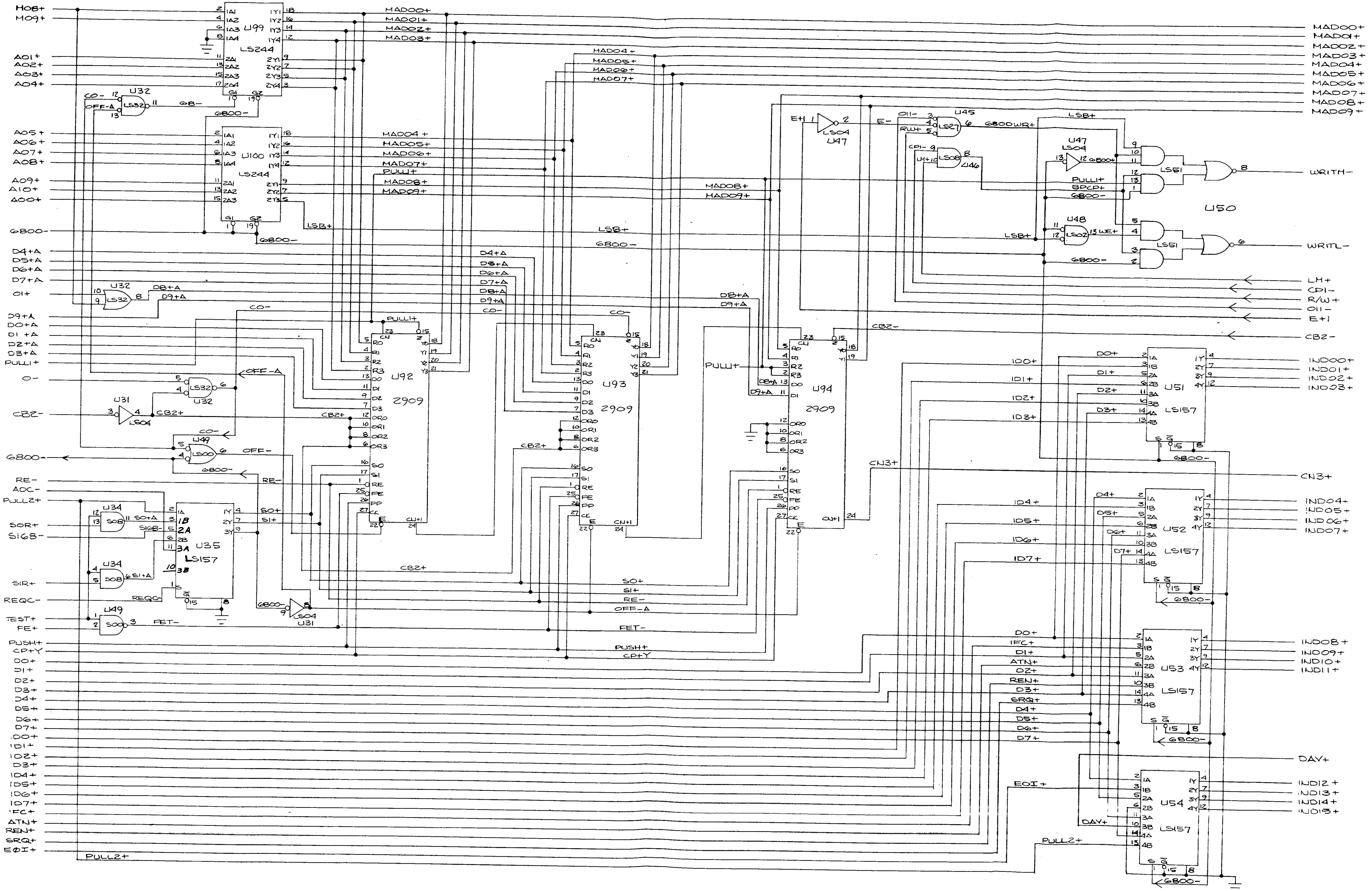
COVINA	interface TECHNOLOGY	CA. 917
SCALE: NTS DATE: 1-3-79	APPROVED BY: CHK	DRAWN BY RJK REVISED
ASSY/LBD LWR LOGIC PANEL MODEL 488		
SHEET 1 OF 7	55613	DRAWING NUMBER 10011128

14 13 12		
000	6800	CONTROL OF BP MEM.ADDR.
001	6800	MONITOR INTERRUPT STATUS.
010	6800	MONITOR LINE NUMBER
011	6800	MONITOR BP ADDR. REG.
100	6800	MONITOR BYTE MEMORY ADR. LSB
101	6800	MONITOR BYTE MEMORY ADR. HSB
110	6800	MONITOR BP PROG. ADR. LSB
110	6800	MONITOR BP PROG. ADR. HSB.



TOLERANCES UNLESS OTHERWISE SPECIFIED		TITLE LOWER PANEL LBD MODEL 488			Interface
XX ± .010 XXX ± .005 ANGLES ± 1°					
MATERIAL		ENGINEER			CODE IDENTIFICATION 55613
SURF. TREAT		DETAILER			SCALE —
HEAT TREAT		CHECKED			DWG. NO. 10011128
DESIGNER		APPROVED			REV.
					SHEET

REV. LTR.	REVISION	APPROVED	DATE



BUS PROCESSOR PROGRAM SEQUENCER

TOLERANCES UNLESS OTHERWISE SPECIFIED
XX ± .010 XXX ± .005 ANGLES ± 1°

MATERIAL
SURF. TREAT
HEAT TREAT
DESIGNER

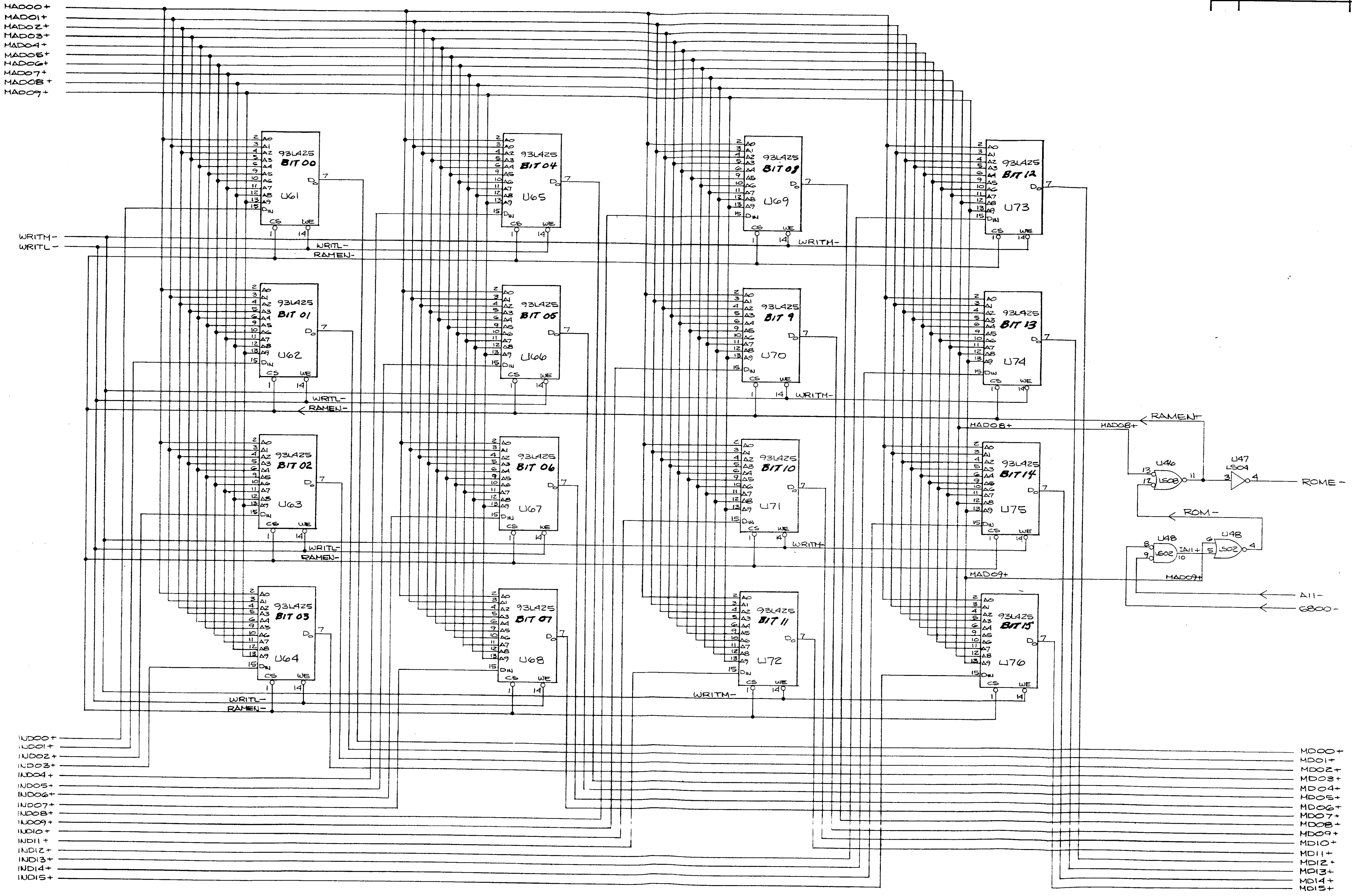
TITLE LOWER PANEL LBD
MODEL 488

ENGINEER
DETAILER
CHECKED
APPROVED

interface
TECHNOLOGY

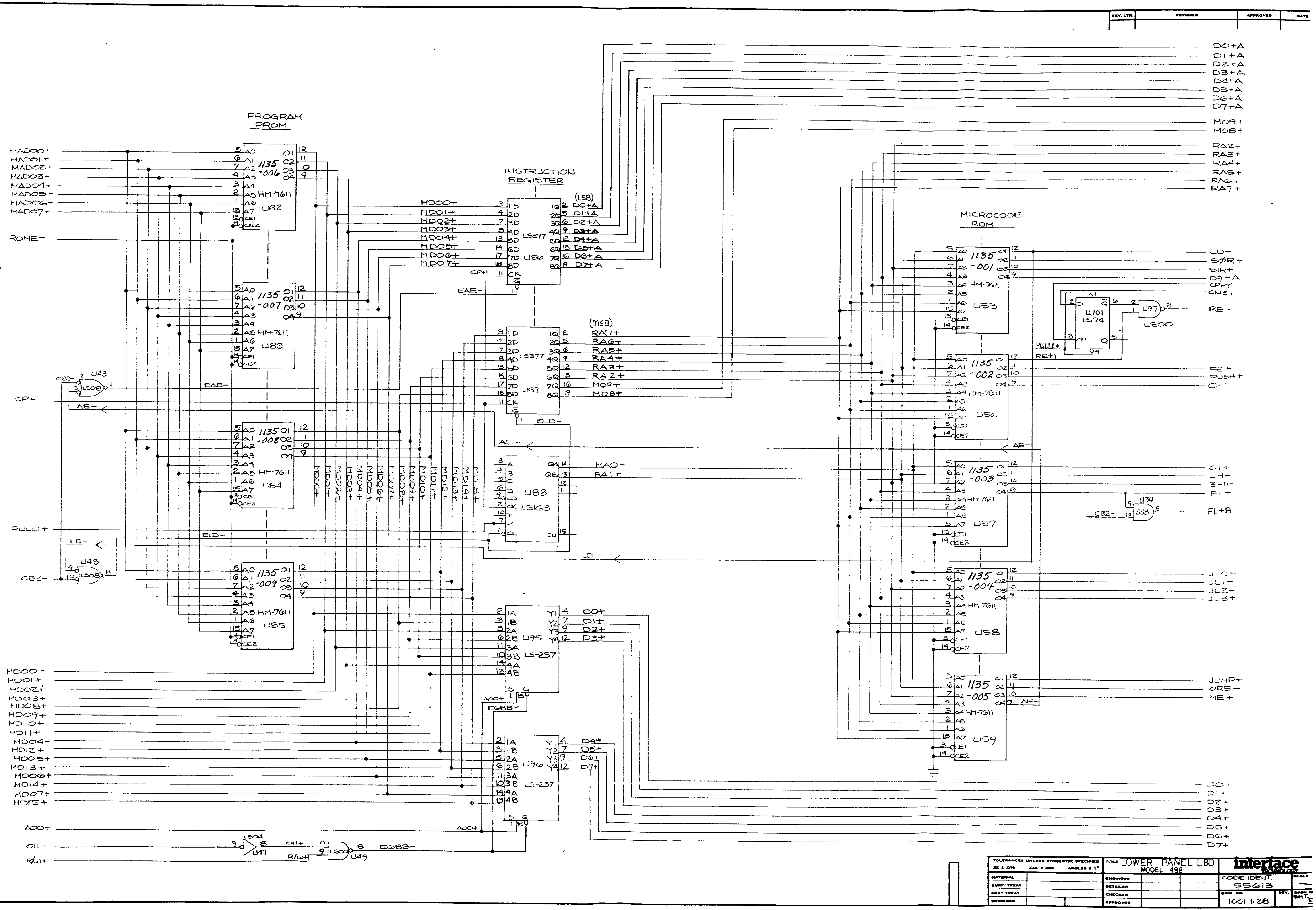
CODE IDENT. 55613
SCALE —
DWG. NO. 10011128
REV.
SHEET

REV. LTR.	REVISION	APPROVED	DATE



PROGRAM RAM

TOLERANCES UNLESS OTHERWISE SPECIFIED	TITLE LOWER PANEL LBD MODEL 488	Interface Technology
XX ± .010 XXX ± .005 ANGLES ± 1°	ENGINEER	CODE IDENT. 55613
MATERIAL	DETAILER	DWG. NO. 1001 1128
SURF. TREAT	CHECKED	REV.
HEAT TREAT	APPROVED	SCALE
DESIGNER		SHT.



TOLERANCES UNLESS OTHERWISE SPECIFIED
XX ± .010 XXX ± .005 ANGLES ± 1°

TITLE LOWER PANEL LBD
MODEL 488

interface TECHNOLOGY

MATERIAL	
SURF. TREAT	
HEAT TREAT	
DESIGNER	

ENGINEER		
DETAILER		
CHECKED		
APPROVED		

CODE IDENT. 55613

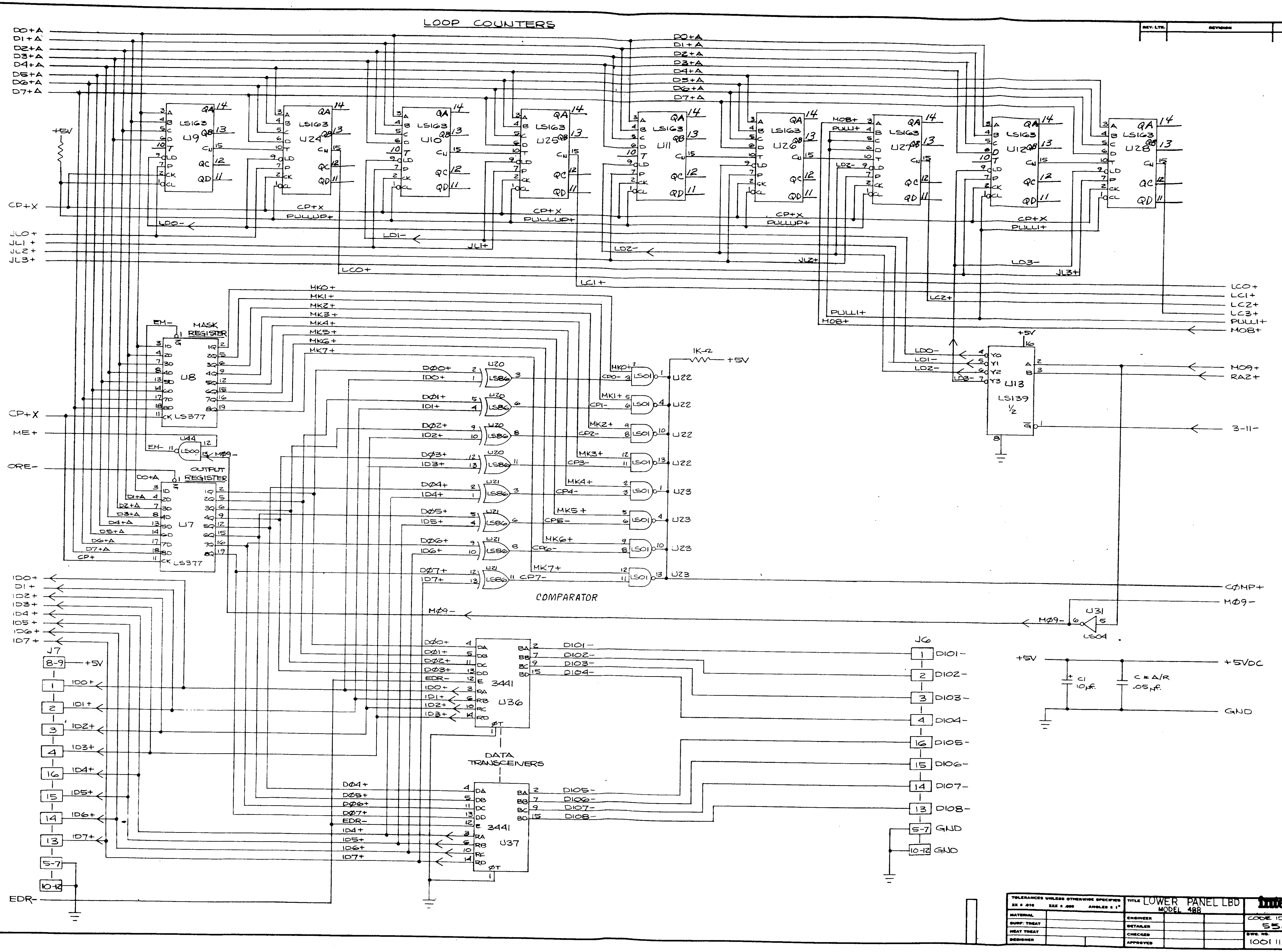
SCALE —

DWG. NO. 1001 1128

REV.

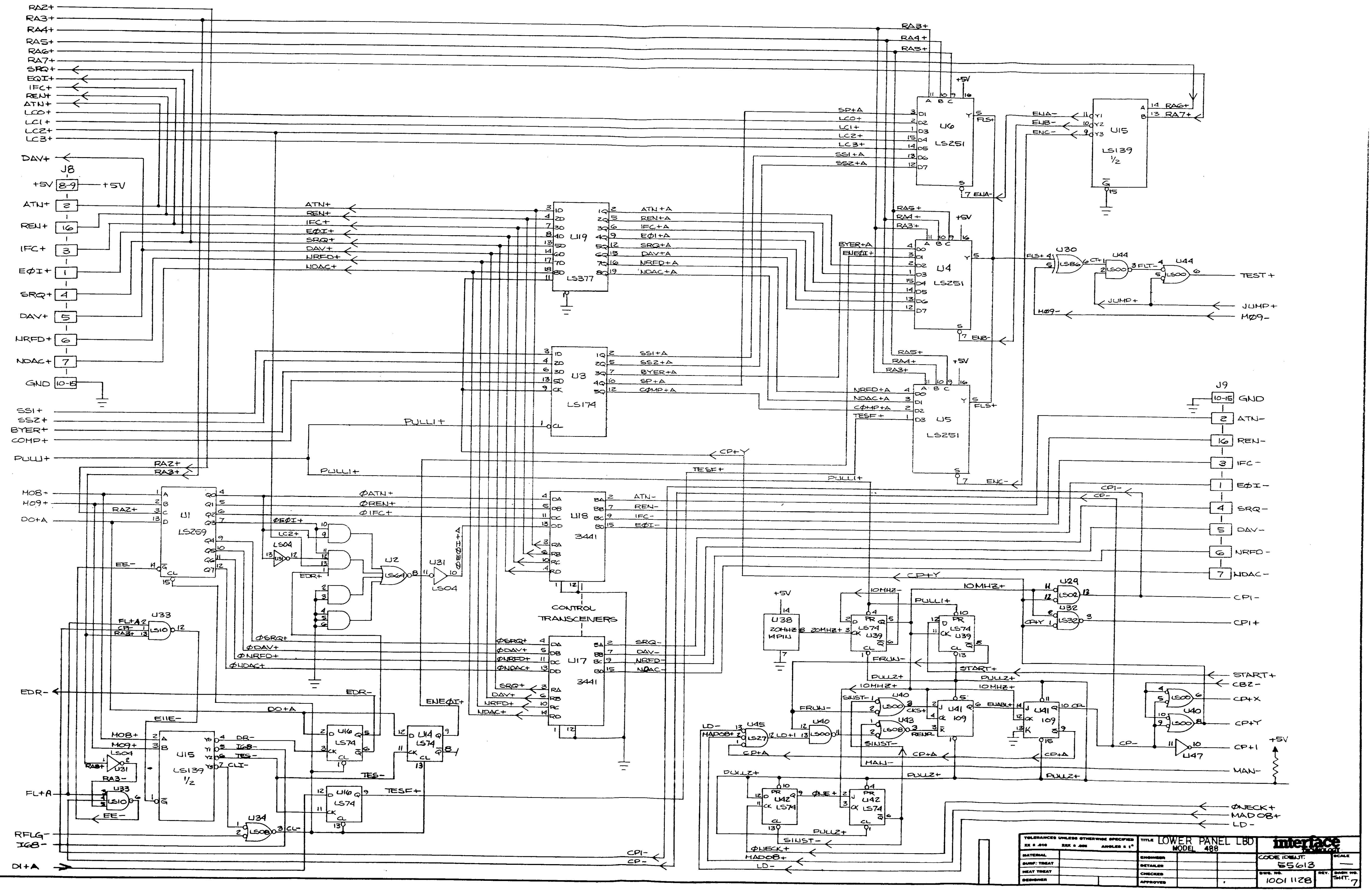
DASH NO. SHT. 5

REV. LTR.	REVISION	APPROVED	DATE



TOLERANCES UNLESS OTHERWISE SPECIFIED XX ± .010 XXX ± .005 ANGLES ± 1°	TITLE LOWER PANEL LBD MODEL 488	Interface Technology
MATERIAL	ENGINEER	CODE IDENT. 55613
SURF. TREAT	DETAILER	SCALE —
HEAT TREAT	CHECKED	DWG. NO. 1001 1128
DESIGNER	APPROVED	REV. DASH NO. SHT. 6

REV. LTR.	REVISION	APPROVED	DATE



TOLERANCES UNLESS OTHERWISE SPECIFIED XX ± .010 XXX ± .005 ANGLES ± 1°

MATERIAL	
SURF. TREAT	
HEAT TREAT	
DESIGNER	

TITLE LOWER PANEL LBD MODEL 488

ENGINEER		
DETAILER		
CHECKED		
APPROVED		

interface technology

CODE IDENT. 55613

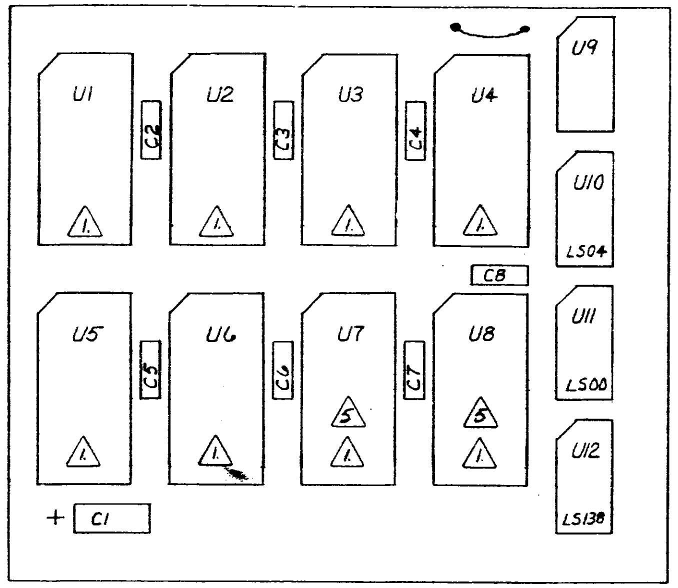
SCALE —

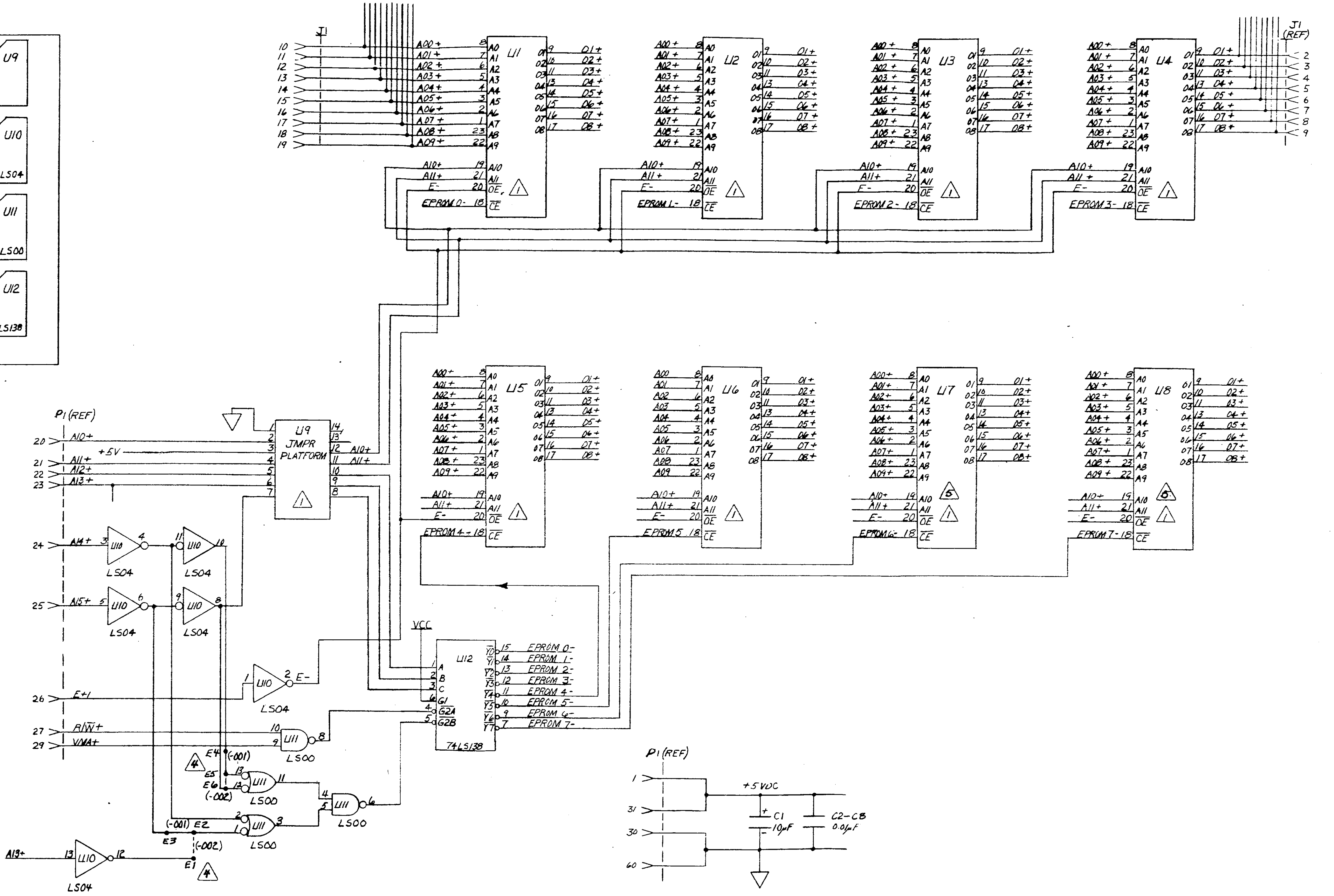
DWG. NO. 10011128

REV.

DASH NO. SHT. 7

REVISIONS			
LTR	DATE	APPR	DESC
A	8-17-79	GWK	ADDED -002
B	6-6-80	RK	PER C.O. 0003





6. REF. 10011129 FOR PCB.

5 OMIT FOR -002

4 •- - -• -002 (RS-660)
•——• -001 (MODEL 488)

3 CAPACITORS C2-C8 ARE 0.01 μF CERAMIC

2. CAPACITOR C1 IS 10μF 25 V.

TO U9 PIN NO. / EPROM TYPE	FROM U9 PIN NO.				
	12	11	10	9	8
2758	1	3	2	4	5
2716	2	3	4	5	6
2732	2	4	5	6	7
2516	2	3	4	5	6

1 U1-U8 MAY USE INTEL 2758, 2716, 2732 OR TI 2516;
INSTALL FOLLOWING JUMPERS ON U9.

NOTES:

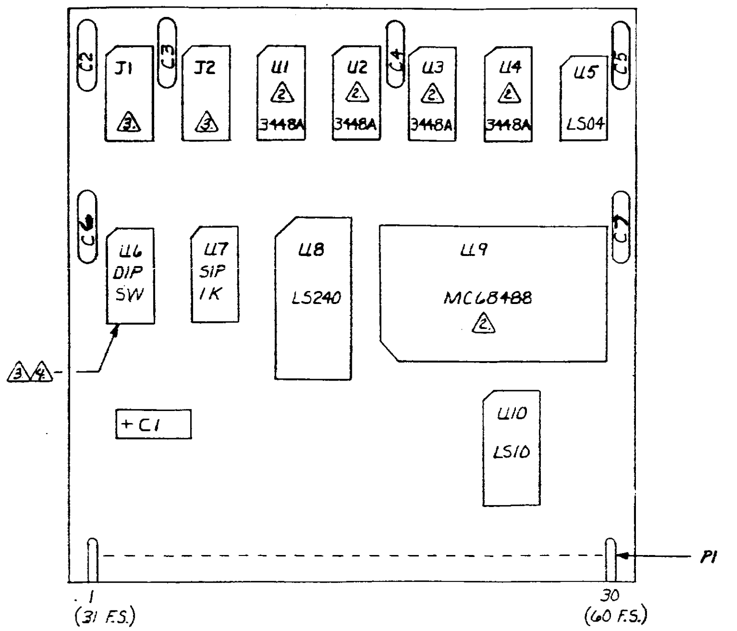
interface TECHNOLOGY

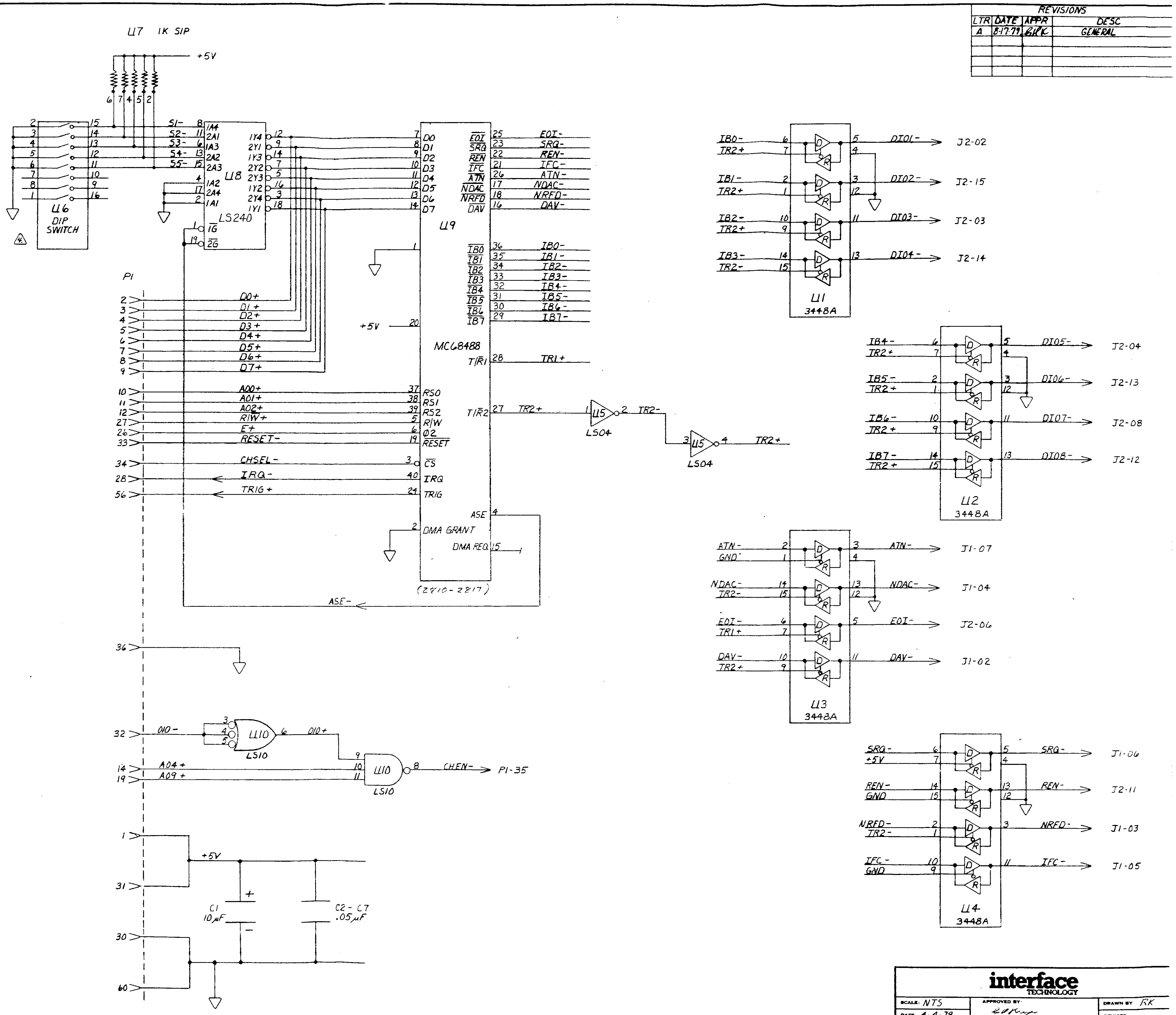
SCALE: N.T.S
APPROVED BY: 2-7-79
DRAWN BY R.J.K
DATE: 1-8-79
REVISED

ASSY/LBD, STORED PGM. CARD

SHT. 1 OF 1 55613 DRAWING NUMBER 10011139

REVISIONS			
LTR	DATE	APPR	DESC
A	8-17-79	BKK	GENERAL





4 FOR MODEL 488, INSTALL 5 POSITION DIPSWITCH WITH SWITCH NO. 1 IN SOCKET PIN 2. FOR RS-660, INSTALL CABLE TO REAR PANEL.

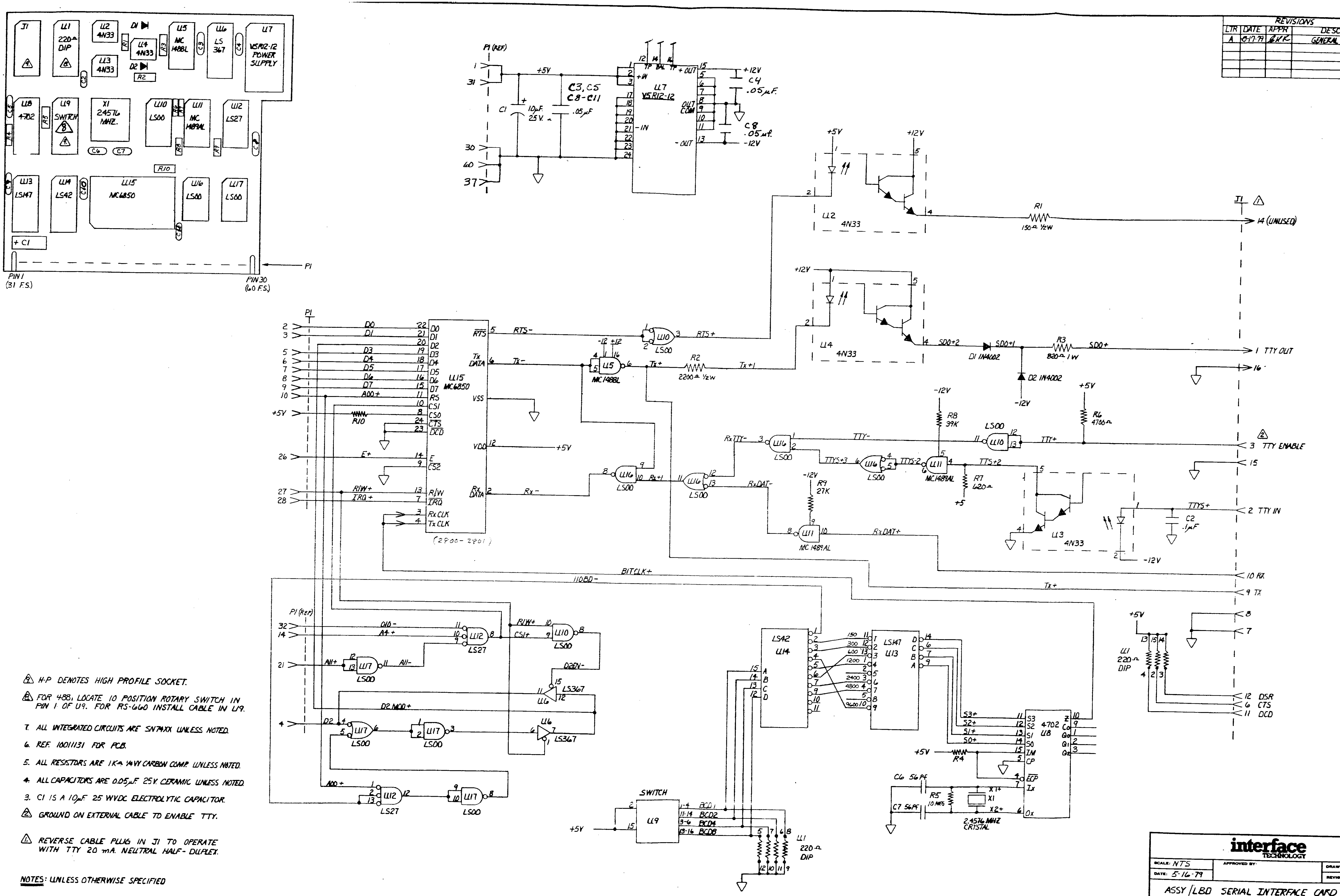
3 H-P DENOTES HIGH PROFILE SOCKET.

2 ALL INTEGRATED CIRCUITS ARE SN74XX UNLESS NOTED.

1. REFER TO 10011133 FOR PCB.

NOTES: UNLESS OTHERWISE SPECIFIED

interface TECHNOLOGY		
SCALE: NTS	APPROVED BY:	DRAWN BY RK
DATE: 4-4-79		REVISED
ASSY/LBD IEEE INTERFACE CARD		
SHT. 1 OF 1	55613	DRAWING NUMBER 10011134



⑨ H-P DENOTES HIGH PROFILE SOCKET.

⑧ FOR 488, LOCATE 10 POSITION ROTARY SWITCH IN PIN 1 OF U9. FOR RS-660 INSTALL CABLE IN U9.

7. ALL INTEGRATED CIRCUITS ARE SN74XX UNLESS NOTED.

6. REF. 10011131 FOR PCB.

5. ALL RESISTORS ARE 1KΩ ¼W CARBON COMP. UNLESS NOTED.

4. ALL CAPACITORS ARE 0.05μF 25V. CERAMIC UNLESS NOTED.

3. C1 IS A 10μF 25 WVDC ELECTROLYTIC CAPACITOR.

② GROUND ON EXTERNAL CABLE TO ENABLE TTY.

① REVERSE CABLE PLUG IN J1 TO OPERATE WITH TTY 20 mA. NEUTRAL HALF-DUPLEX.

NOTES: UNLESS OTHERWISE SPECIFIED

REVISIONS

LTR	DATE	APPR	DESC
A	8-17-79	BKK	GENERAL

interface TECHNOLOGY

SCALE: NTS	APPROVED BY:	DRAWN BY RK
DATE: 5-16-79		REVISED

ASSY/LBD SERIAL INTERFACE CARD

SHT 1 OF 1	55613	DRAWING NUMBER 10011132

File: 6

INTERFACE TECHNOLOGY

INTEROFFICE MEMORANDUM

#M9-466

DATE: 8 July 1983

TO: J. A. Stroot

FROM: D. K. Hadley

SUBJECT: Model 488 Remote Control Problem

cc: B. Hironaka
D. Johnson
S. Kubota

Steve Coan of Tektronics encountered some problems controlling our Model 488 from a Remote Controller as follows:

- An attempt to set/reset the front panel alternate action switches (SS1, SS2, or bypass error) via a remote controller following the entry of the Model 488 into Machine Language mode via the front panel keys would cause the loss of program control.
- Interrogation of the Model 488 unit status following a hardware error halt at program memory address 040 resulted in the transmission of an incorrect error code and class.

The following program patches are required which will upgrade the software to Revision 3.2:

PROGRAM LOCATION	EPROM	EPROM LOCATION	DATA WAS	CHANGED TO
CA4D	1136-21	0A4D	CC	D6
CB52	1136-21	0B52	CC	D6
DACE,F,D0	1136-22	0ACE,F,D0	30,30,33	33,2E,32
D913	1136-22	0913	11	12
D98F	1136-22	098F	11	12
~~FD81~~	~~1136-24~~	~~0D81~~	~~CC~~	~~D6~~ no change
FECE,F	1136-24	0ECE,F	97,1F	01,01
~~FD57~~	~~1136-24~~	~~0D57~~	~~CC~~	~~D6~~ no change
FDF4	1136-24	0DF4	11	12

DKH:mmb