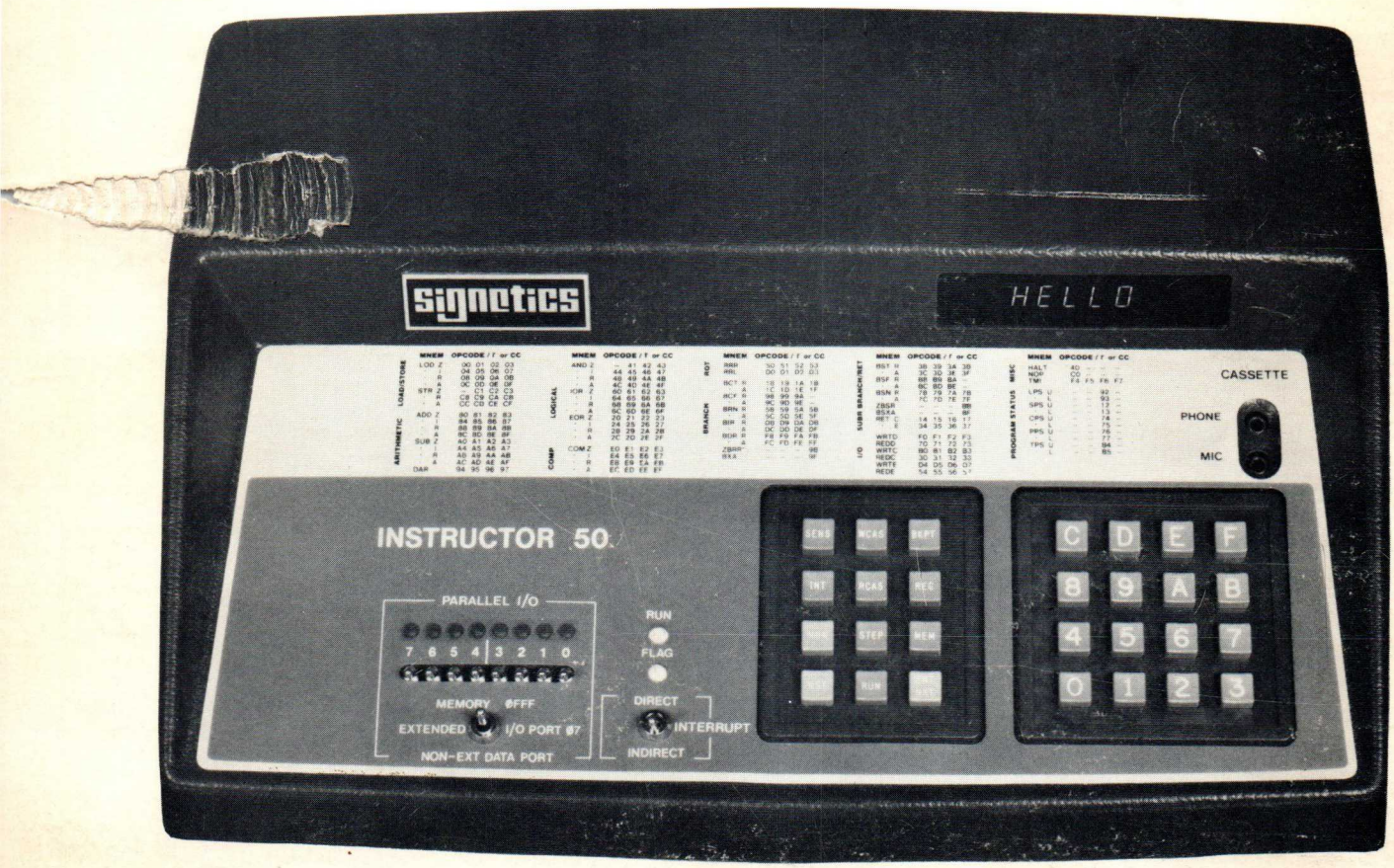
SIGNETICS INSTRUCTOR 50™ DESKTOP COMPUTER SOFTWARE APPLICATIONS MANUAL



SIGNETICS INSTRUCTOR 50™ DESKTOP COMPUTER SOFTWARE APPLICATIONS MANUAL

by

John C. Garceau

The applications contained in this manual are for illustration purposes only. SIGNETICS CORPORATION makes no representation or warranty that such applications are suitable for the use specified without further testing and modification.

SIGNETICS CORPORATION is a wholly-owned subsidiary of U. S. Philips Corporation.

INSTRUCTOR 50 PROGRAM APPLICATIONS

INTRODUCTION:

Originally intended as a separate part (Chapter 6) of the INSTRUCTOR 50 INTRODUCTORY MANUAL, this manual is now released as a "stand-alone" document for use by INSTRUCTOR 50 owners. This manual provides a wealth of detailed information about the various programs contained in the INSTRUCTOR 50 INTRODUCTORY CASSETTE. The following outline is provided for your convenience. Use it to index quickly to any point of interest you may desire to examine.

	PAGE
INTRODUCTION TO ASSEMBLERS .....*(a commentary)*	2
INDEX TO THE PROGRAMS.....*(including file numbers)*	5
INSTRUCTOR 50 USER OPERATIONS	
Memory Display & Alter Mode	6
Memory Fast Patch Mode	8
Write Cassette Mode	10
Tape Recorder Level Adjust Mode	13
Read Cassette Mode	14
HOW TO USE THIS DOCUMENTATION	15
THE PROGRAMS	
"BILLBOARD"	21
"DESK CLOCK"	24
"STOP WATCH"	34
"CRAPGAME"	45
"BEAT THE ODDS"	62
"SLOT MACHINE"	68
"TRAIN"	85
"INSTRUCTOR 50 MUSIC THEME"	94
2650 PROGRAMMING FORM .....*(left blank for your duplication purposes)*	97

NOTE:

As much as possible, these pages are designed with your use in mind. There's plenty of space for your notes and observations. Take time to study and practice the use of the INSTRUCTOR 50 and the 2650 microprocessor which it hosts. You'll find each minute devoted to this will valuably increase your knowledge.

COMMENTARY:

When we entered information in the coding forms for examples provided in Chapter 5, we first wrote each program in MNEMONIC form. The MNEMONIC representation of a program is called assembly language. Each instruction written in ASSEMBLY LANGUAGE has a corresponding HEXADECIMAL CODE which can be executed by the INSTRUCTOR 50. However, it is much easier for us as humans to interpret instructions written in assembly language rather than in hexadecimal code. Hexadecimal (HEX) code, with each digit representing 4 bits, is also known as MACHINE LANGUAGE. Once a program has been written in assembly language, it is a straight-forward task to translate it into machine language.

This translation task can be accomplished by a computer program. Such a program is called an ASSEMBLER PROGRAM. The purpose of the assembler program is to take information appearing on the right side of the CODING FORM and translate it into information for the left side of the coding form. The ASSEMBLER PROGRAM then produces a PRINTOUT which is similar to the coding form filled out in its entirety. This printout is called a LISTING of the program.

The listings for all of the programs recorded on the INSTRUCTOR 50 INTRODUCTORY CASSETTE are contained in the remainder of this chapter. Two of the listings are printouts from an assembler program run on the "TWIN", a Signetics SOFTWARE DEVELOPMENT SYSTEM. The other listings were HANDCODED and then reproduced on the Signetics 2650 PROGRAMMING FORM.

The information which appears on the right-hand side of the coding form is called SOURCE CODE . Simply, source code is information which the assembler (or HANDCODING programmer) must have in order to generate MACHINE LANGUAGE code. The HEX-CODES for the instructions which are generated by the assembler are known also as OBJECT CODES. After all, generation of valid machine language code is the OBJECT of the assembler.

Before the assembler can produce object code, it must be provided certain key information in addition to that appearing in the right hand columns of the coding form. This information is provided in the form of DIRECTIVES. For example, if you wish the start address for a routine to be at location H'100' in memory, you would input the directive ROUT ORG H'100' to the assembler. "ROUT", in the coding form's LABEL field, defines the name of the routine. It could be any combination of letters and numbers. "ORG", in the coding form's OPCODE field, directs the assembler to recognize this as a specific start address; a PROGRAM ORIGIN!! "H'100'", in the OPERAND field, defines the memory address. The ORG statement is not translated into OBJECT CODE by the assembler. For this reason, it and other assembler directives are also called "PSEUDO OP"s.

There are several other assembler directives which are used in preparing source code for the assembler's use, These include the EQUate and REServe statements. EQU directs the assembler to assign a specific

COMMENTARY: (CONTINUED)

numeric (hex code) value to a symbolic label. Once defined, the label will be translated by the assembler to its hex code equivalent each time it is encountered in the source code. RES directs the assembler to REServe 1 or more memory locations for some specific function defined in the LABEL field.

When writing programs for the assembler, it is very useful to insert COMMENTS before or after various blocks of source code. The asterisk symbol (*) is used to define a line of source code as a comment. Upon detecting an asterisk, the assembler makes no attempt to translate succeeding source code on the same line into object code.

In operation, the assembler sequentially SCANS the mnemonic code you have written for each instruction. It generates the hex-code for each instruction, storing these codes in its memory. In some cases, such as in translating BRANCH instructions, the complete hexadecimal code for the instruction may be unknown. For example, assume that a BCTA (Branch on Condition True - Absolute addressed) instruction is encountered in line 40 of a listing. Further, that the branch location (in memory) is not defined until the assembler encounters line 65. During the FIRST PASS, the assembler will "save" enough memory for eventual storage of the destination address, in this case, 2 bytes. Eventually, the address label, in line 65, is defined; the assembler saves this information in a data table of its own. Then, during a SECOND PASS, the assembler scans this table for missing addresses, inserting them into the saved memory locations as required. After the assembler's second pass is complete, it will generate, on command, a PRINTOUT of the entire listing, including all directives, source code, object code, and its own label tables.

NOTES:

THIS PAGE IS LEFT BLANK INTENTIONALLY.

The following chart specifies designated file number(s), and a short description of each program digitally recorded on the INSTRUCTOR 50 INTRODUCTORY CASSETTE tape. Details and variations of each program are provided later in this chapter. The first page of each program's documentation is identified immediately below its name in the table below.

PROGRAM	FILE	DESCRIPTION
BILLBOARD -21-	1	BILLBOARD provides a capability of viewing any user-defined message in the Hex display. Message length is programmable to a maximum of 254 characters. The entire message moves from right to left across the display, and repeats automatically.
DESKCLOCK -24-	2	Using your local AC line frequency as a time source, DESKCLOCK transforms the INSTRUCTOR 50 into a 4-function desk clock. The program permits the user to preset hours, minutes, seconds, and a unique AM/PM indication, then to start the clock, all from the keyboards.
STOPWATCH -34-	3	Accurate to within 1/10th of a second, STOPWATCH extends the INSTRUCTOR 50's capability to act as an event or elapsed time counter. Fully programmable, STOPWATCH permits its user to "freeze" (and display as well) up to 4 separate times while it is running, all without any alteration of its accuracy.
CRAPGAME -45-	4 4 A	Programmed to conform to a modified set of Nevada Casino Pass Line rules, CRAPGAME may be played by the single user or by several players as well. Following the user's "buy into the game" and "bet", the "dice are rolled". After the roll, the program determines whether the player has won, lost, or must roll again. Winnings (and losses) are added to (subtracted from) the player's current chips automatically.
BEAT THE ODDS VERSION 2 -62-	5	BEAT THE ODDS is a numbers game in which one or more players bet against the odds of a specific hex-digit pair or binary pattern being generated by the INSTRUCTOR 50. A convenient ODDS TABLE diagram is provided in the documentation associated with this program.
SLOTMACHINE -68-	6 6 A	SLOTMACHINE features 3 independently rotated "image wheels". After the wheels stop, the program tells the player whether he lost, or if he won. For winning combinations, the INSTRUCTOR 50 calculates and displays the odds for payoff by "the House", and goes absolutely berserk when the player "hits the JACKPOT!"
TRAIN -85-	7	While executing program TRAIN, the operator may select any 1 of 16 different "train" character sequences for display in the "SWITCHYARD. Further, he may run the selected train forwards or backwards at any 1 Of 7 speeds.
INSTRUCTOR 50 MUSIC THEME -94-	8	The music you heard on the Introductory Tape was generated by this program as executed by the INSTRUCTOR 50. You may create your own tunes, simply by modifying the pitch and duration data tables associated with INSTRUCTOR 50 MUSIC THEME.

MEMORY DISPLAY & ALTER MODE

INTRODUCTION:

There will be many times when you'll want to alter or display the contents of specific memory locations. In running the programs supplied in this chapter, you'll have occasion to change the display time of various messages. In other words, you'll want to "tailor" each program to your acceptance level. When you write your own routines and programs , you'll also debug them by changing the value of a byte, altering memory as required by your application's software. You'll use MEMORY DISPLAY & ALTER mode to accomplish these changes, in addition to other deletions, additions, and substitutions your program requires. You'll also employ this mode to VERIFY the contents of memory you may load in FAST PATCH mode.

PROCEDURE:

1. Depress MON, MEM, and 1 to 4 HEX keys. ..... *to define the specific address of the byte of interest. The selected address is displayed, and may be changed until you depress ENT/NXT.*

2. Depress ENT/NXT ..... *to set the desired address. The INSTRUCTOR 50 responds by displaying the current data contained in the selected memory address.*

```
**********************************************
*                    NOTE:                   *
*                                            *
*  AT THIS POINT, YOU MAY EXERCISE ANY       *
*  OF THE FOLLOWING OPTIONS, DEPENDENT       *
*  UPON YOUR REQUIREMENTS.                   *
*                                            *
**********************************************
```

3A. Depress ENT/NXT ..... *To INCREMENT the selected address to the next memory location, with NO DATA CHANGE to the current address's contents.*

USE THIS OPTION TO VERIFY MEMORY CONTENTS WITHOUT CHANGE.

3B. Depress 1 HEX key and ENT/NXT. ..... *to change the selected byte's least significant digit (bits 3-Ø). The most significant digit is ZEROED.*

3C. Depress 2 HEX keys and ENT/NXT. ..... *to alter the entire contents of the selected memory location. Current data is SET and the address is INCREMENTED when you depress ENT/NXT.*

EXAMPLE:

In routine "DISMES" (below), the display time of messages included in program "CRAPGAME" is controlled by data contained in location H'Ø13D' of memory. The value may be varied between H'01' and H'0A', to provide message display times equal to 0.25 and 2.5 seconds respectively (in 0.25 second increments).

Modify the contents of location H'013D' as you wish in order to provide a desired message display time:

Ø13C 05 02

ROUTINE DISMES	START ADDR 13A	PART OF PROGRAM CRAPGAME	26
DESCRIPTION 'DISMES' CONTROLS ACCESS AND DISPLAY TIME OF 8 MESSAG TED IN SMI MEMORY: LOCATIONS '1780' TO'17BE'			

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	
4								
5	Ø13A	CA	7B		DISMES	STRR,R2	>MESLOC-1	ON ENTRY
6	C	05	02			LODI,R1	2	INDEX TC
7	13E	C9	79		REPEATM	STRR,R1	RUNDSPX	NOW SET
8	140	CB	76		DSPAGN	STRR,R3	DSPLDLY	OUT & ST
								STATIC/D

1. Depress RST ..... *To execute the program: "CRAPGAME".*

2. Depress MON ..... *To stop CRAPGAME's operation after a few message sequences have been displayed.*

3. Depress MEM, 1, 3, D, ... *to select, then set the address of the* and ENT/NXT. *desired byte. The message:*

   "013d. 02" is displayed.

4. Depress '4', and ..... *to DOUBLE the display time of "CRAP-*
   ENT/NXT. *GAME"'s messages.*

5. Repeat steps 1 through ..... *keying in different values for location*
   4. *H'013D', until the value which represents your desired display time is stored in memory.*

****
*
* NOTE:
*
* YOU'LL BE ABLE TO PERFORM THE ABOVE PROCEDURE
* AFTER YOU HAVE LOADED PROGRAM 'CRAPGAME' INTO
* THE INSTRUCTOR 50'S MEMORY, EITHER FROM TAPE
* OR IN FAST PATCH MODE.
*
****

NOTES:

MEMORY FAST PATCH OPERATIONS

INTRODUCTION:

Within the listings contained in this chapter, there are many variations to each application, which may be accomplished by changing the contents of a BLOCK of CONTROL PROGRAM memory. You'll use MEMORY FAST PATCH, one of several INSTRUCTOR 50 operational modes, to load and modify these blocks of memory. FAST PATCH is also employed in restoring smaller blocks of memory, such as 8-byte messages, to their original data values. This is illustrated in the example on the opposite page.

PROCEDURE:

1. Depress MON, REG, and "F" .....*To enter MEMORY FAST PATCH mode. The message: ".Ad. = " is displayed.*

2. Depress 1, 2, 3, or 4 HEX keys .....*To select the address of the 1st byte you'll be loading into this memory block.*

3. Depress ENT/NXT .....*to set the address you just keyed in. The message: ".XXXX " is displayed.***

4. Depress TWO Hex keys for each byte of data to be loaded into memory. .....*Start with the data contained in the selected start address. As each memory location is loaded, the displayed address will be INCREMENTED by 1. Repeat Step 4 until the entire block of specified memory is loaded.*

****
* NOTE: *
* *
* WHILE YOU ARE KEYING DATA IN FAST PATCH *
* MODE, NO DATA CHANGE IS POSSIBLE. IF YOU *
* KEY AN UNDESIRED VALUE, EXECUTE ONE OF *
* THE FOLLOWING KEY SEQUENCES, DEPENDING *
* ON THE CURRENT HEX DISPLAY READOUT. *
****

(1) Depress REG,'F', ENT/NXT .....*if the address of the data error is still displayed. The INSTRUCTOR 50 will display the message: ".XXXX " where 'XXXX' is the address of the byte requiring change*

(2) Depress REG, 'F', then 1 to 4 HEX keys, then ENT/NXT. .....*for the desired address of the byte-in-error, if the desired address is no longer displayed.*

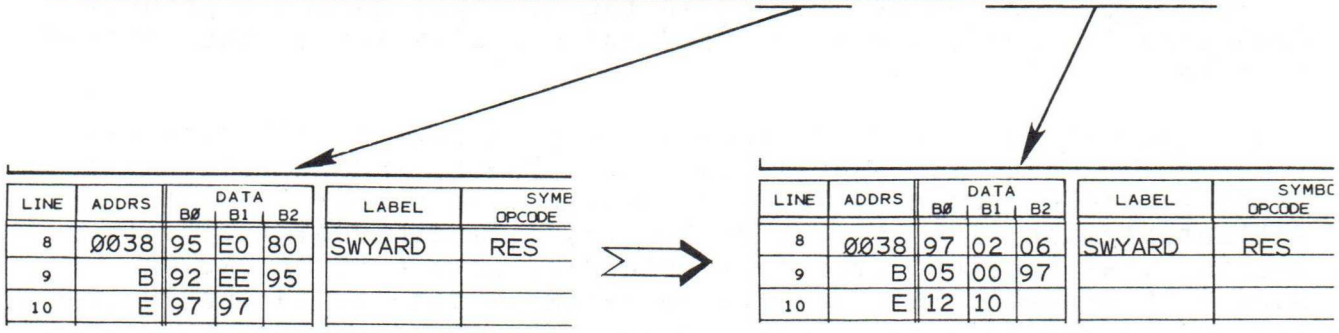
** 'XXXX' REPRESENTS THE ADDRESS YOU KEYED IN STEP 2.

Then, key in the desired data as defined in step 4.

5. Depress ENT/NXT .....*To terminate FAST PATCH mode.*

EXAMPLE:

CHANGE DATA AS SHOWN IN THE LISTING (BELOW)...TO THESE VALUES:



LINE	ADDRS	DATA B0	B1	B2	LABEL	SYME OPCODE
8	0038	95	E0	80	SWYARD	RES
9	B	92	EE	95		
10	E	97	97			

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBC OPCODE
8	0038	97	02	06	SWYARD	RES
9	B	05	00	97		
10	E	12	10			

1. Depress REG, F, 3, 8, ENT/NXT.....*Start address is H'0038'.*

2. Depress HEX keys:

9 7 0 2 0 6 0 5 0 0 9 7 1 2 1 0

3. Depress ENT/NXT .....*To terminate FAST PATCH mode.*

NOTES:

WRITE CASSETTE OPERATIONS

INTRODUCTION:

As you become more and more proficient in the art of writing programs for the INSTRUCTOR 50, you'll want to save them for future recall on tape. Once your program is stored on tape, you need only to perform the easy READ CASSETTE operational procedure to reload the INSTRUCTOR 50's memory. It is particularly useful to store "programs in process" as a means of recalling their historical content during debug. Many programmers also store specific routines or program segments. If a subsequent application requires use of one or more of these segments, each is easily recalled, then located as desired in the new application's program. Much time is saved, since you need not manually key in the desired routine.

Your selection of recording tape is very important. All tape has imperfections in its magnetic surfaces. For the music buff, this makes little difference; he'll probably never detect the intermittant blemish. The INSTRUCTOR 50, like all other digital data machines, will detect imperfect recording as a LOSS OF ONE OR MORE BITS OF DATA. This results in an ERROR display on the INSTRUCTOR 50. To minimize data loss problems, use good quality low-noise tape. Also, record a safety copy of each of your programs. When purchasing blank tape cassettes, select those which are biased for your recorder's operation. We recommend use of short-time tapes (e.g.: C-30, or C-45 cassettes) in order to minimize "hunt time" during your file searches. NOTE: SHORT-TIME TAPES ARE MANUFACTURED WITH A THICKER BASE, THUS, THEY ARE STRONGER AND LESS SUBJECT TO MAGNETIC "PRINT-THROUGH". Before recording an actual file on tape, we recommend that you vocalize a brief introduction to it. The introduction should include:

* the desired File number
* the name of the program or routine
* lower and upper memory addresses of the memory block to be stored as a file.
* the start address for program execution, especially if it is not address '0000'.

The introduction will probably require no more than 20 seconds of tape, and serves as a convenient "leader" between recorded segments of digital data. It is useful as well to keep a written record of the files stored on tape. In addition to the information contained in the verbal file leader, you may wish to include a few notes regarding INSTRUCTOR 50 control switch settings or other operator activities. If your recorder is equipped with a TAPE COUNTER, identify the approximate count (from 000) where each file is contained on tape. You'll simplify your file search operations later.

Finally, insure that your recorder's tapeheads and mechanical components are CLEAN, lubricated, and operate smoothly. The operations booklet for your recorder tells exactly what's necessary.

WRITE CASSETTE OPERATIONS
CONTINUED

PROCEDURE:

	ACTIVITY	COMMENTARY
1.	Install the cassette in your recorder, and vocalize the introduction to the next file.	*Page 6-10 provides a short list of items you may wish to include in your introduction.*
2.	Connect the audio cable between the INSTRUCTOR 50's MIC jack & the MICrophone input to your recorder.	*Do not use your recorder's "AUX" or "LINE" input unless your recorder's MIC input overloads or distorts.*
3.	Depress MON and WCAS	*To enter "WRITE CASSETTE mode. The message: "LAd = " is displayed.*
4.	Depress 1, 2, 3, or 4 HEX keys .. to define the LOWER address of the memory block to be stored on tape.	*Display: "LAd = XXXX" where 'x' is the value of each hex key.*
5.	Depress ENT/NXT	*to set the lower address. The message: "UAd = " is displayed.*
6.	Depress 1, 2, 3, or 4 HEX keys ..	*to define the UPPER address of the memory block to be stored. The message: "UAD = XXXX" is displayed where 'x' is the value of each hex key.*
7.	Depress ENT/NXT	*to set the upper address. The message: "SAd = " is displayed.*
8.	Depress 1, 2, 3, Or 4 HEX keys ..	*to define the program execution start address of the memory block represented by this file. Display: "SAd = XXXX".*
9.	Depress ENT/NXT	*to set the start address. The message: "F = " is displayed.*
10.	Depress 1 or 2 HEX keys	*to define the file number of the memory block to be stored. Display: "F = XX ". 'xx' = file number.*

WRITE CASSETTE OPERATIONS
CONTINUED

PROCEDURE: (CONTINUED)

11.	Depress ENT/NXT and immediately place your recorder in RECORD mode.	*You have about 6 seconds to do this, after which your program's digital data transfer to tape is initiated. The Extended I/O LEDs flash for the duration of the transfer.*
12.	STOP your recorder.....	*when the message "HELLO" is displayed. "HELLO" indicates that the data transfer is completed.*

If an additional file is to be stored on tape, repeat step 1, and steps 3 through 12. At your option, record a "SAFETY COPY" of each file stored on tape. In your verbal introduction, mention the file number, and the fact that this is a safety copy.

NOTES:

TAPE RECORDER LEVEL ADJUST

INTRODUCTION:

The following procedure is provided as a practical means of calibrating your tape recorder's output level (volume) and tone controls for loading digital data from tape to the INSTRUCTOR 50. By following this practical procedure, you'll achieve a very high rate of success in subsequent program recall.

1. Ensure that your recorder's tapeheads, capstan, pinch roller, and other mechanical assemblies are clean and operate smoothly. Your recorder's Operations Booklet will inform you of the proper sequence you must follow.

2. Use your recorder's FAST FORWARD or REVERSE controls to position the desired source program file over the playback head. You can do this with the volume control reduced for comfortable listening level.

3. Connect an audio cable between the INSTRUCTOR 50's PHONE jack, and your recorder's PHONE or MONITOR jack.

4. Set your recorder's TONE control (if present) in the high TREBLE range. Set the BASS tone control (if present) to its most attenuated value.

5. On the INSTRUCTOR 50, depress MON, REG, and 'A'. The Hex display will go blank.

6. Start your recorder in PLAYBACK mode. Then adjust its output volume control for higher output. The INSTRUCTOR 50's Hex display:

   * will be blank
   * will display a 'U'
   * will display a '-' (dash)
   * will display a 'd'

   } IN SEQUENCE AS YOU INCREASE VOLUME

7. As soon as the 'd' is displayed, DECREASE volume until the '-' (dash) reappears in the display. Do not adjust the volume control further.

8. Alternately REWIND the tape for a few turns, and PLAYBACK for a few seconds. During playback, the "dash" will reappear in the display until you have rewound the tape to the start of the desired program's data. Ignore display indications during the short rewind operations. When the "dash" no longer appears in the display, stop the recorder, and go right on to perform the READ CASSETTE procedure.

NOTE: PERFORMANCE OF STEP 8 ELIMINATES ANY NEED TO LISTEN FOR THE START OF DIGITAL DATA ONCE THE LEVEL IS ADJUSTED. IN ITS OWN WAY THE INSTRUCTOR 50 DOES THIS FOR YOU.

READ CASSETTE OPERATIONS

INTRODUCTION:

You'll use this procedure over and over again as you play back digital programs from the Introductory cassette, or from your own program "library" into the INSTRUCTOR 50.

NOTE: NO MAGNETIC RECORDING TAPE IS MANUFACTURED PERFECTLY. DESPITE STRINGENT PROCESSES, THE TAPE'S ABILITY TO STORE DATA EVENLY MAY VARY BECAUSE OF THE TAPE'S OWN MAGNETIC STRENGTH CHARACTERISTICS. FOR THIS REASON, YOU'LL FIND THAT IT IS YOUR ADVANTAGE TO PERFORM THE TAPE RECORDER LEVEL ADJUST SEQUENCE EACH TIME BEFORE SEQUENCING ANY READ CASSETTE OPERATION.

PROCEDURE:

1. VERIFY THE FOLLOWING:

   (a) The audio cable is connected between your recorder's PHONE or MONITOR jack, and the INSTRUCTOR 50's PHONE input jack.

   (b) Your recorder's VOLUME (Output Level) and TONE controls are properly calibrated.

   (c) Your recorder's PLAYBACK head is properly positioned over the start of the desired file's digital data.

2. DEPRESS THE FOLLOWING KEYS ON THE INSTRUCTOR 50:

   MON, RCAS, 1 or 2 Hex keys, ENT/NXT. ..... *to identify the desired file you wish to load from tape into the INSTRUCTOR 50's memory. ENT/NXT initiates the file search.*

3. START YOUR RECORDER IN "PLAYBACK" MODE. ..... *After 5 seconds, the Parallel I/O LEDs will flash to indicate that your program is entering the INSTRUCTOR 50's memory.*

   NOTE: THE I/O MODE SELECTION SWITCH MUST BE IN THE 'EXTENDED I/O PORT 07' POSITION FOR YOU TO VIEW THE I/O LEDS.

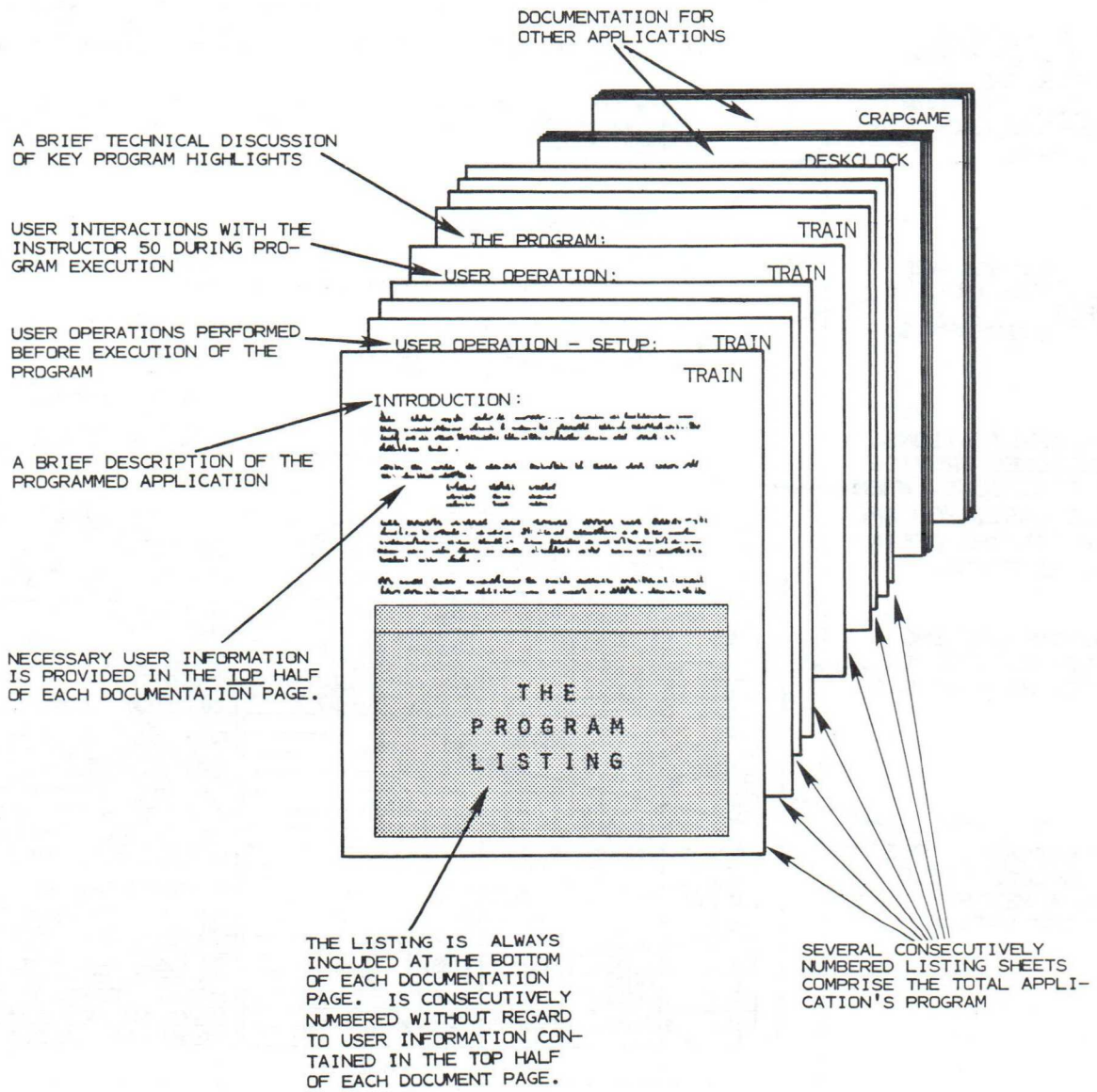
4. STOP YOUR RECORDER WHEN THE INSTRUCTOR 50 DISPLAYS: "HELLO".

   When the INSTRUCTOR 50 displays the message: "HELLO", the memory load from tape is complete. Stop (and store your recorder, and execute the desired program. If the program is contained in 2 files, repeat steps 1 through 4 (above). Do NOT change the volume or tone control settings on your recorder.

HOW TO USE THIS DOCUMENTATION

Documentation which defines any microprocessor-controlled application is every bit as important as the software and hardware which together provide realization of the application. At first glance, good technical documentation can be rather intimidating in its complexity, especially to the novice.

There are as many different formats for preparing documentation as there are application engineers and software specialists. Two essential formats are illustrated. The first, illustrated by programs "BILLBOARD" and "INSTRUCTOR 50 MUSIC THEME", is generated by a SOFTWARE DEVELOPMENT SYSTEM manufactured by Signetics Corporation. Typically, this method supports its application with a few terse comments and flowcharts which summarize the program sequence.



The second method, although more complex, is particularly useful when you are preparing the software and user-interactive parts of an application without the assistance of an ASSEMBLER or other software development system. To reduce its complexity, this format is diagrammed in these 6 pages. Practical use and flexibility of the format are then provided in the descriptions of the TIME and GAMES applications found later in this chapter.

AN OVERALL DESCRIPTION OF THE APPLICATION IS INCLUDED IN THE INTRODUCTION.

THE TITLE OF EACH PROGRAMMED APPLICATION IS HIGHLIGHTED AT THE TOP-RIGHT OF EACH PAGE.

TRAIN

INTRODUCTION: (CONTINUED)

the RST key is depressed. If a change of 'consists' is required, the operator will toggle the appropriate I/O input swiches to select the desired consist and running speed, depress RST, and then depress INT in that order!!!!. If no consist change is necessary, the operator merely selects a running speed, and depresses RST.

If the train is operating at one of its running speeds, the consist may be changed as described in the first paragraph of this introduction.

SET-UP OPERATIONS ARE TYPICALLY FOUND JUST AFTER THE INTRODUCTION. PERFORM THESE BEFORE EXECUTING THE PROGRAM.

USER OPERATION - SET UP:

After loading program "TRAIN" into the INSTRUCTOR 50's memory, the switches should be positioned as follows:

CONTROL SWITCHES ARE PREPOSITIONED IN LINE WITH APPLICATION & PROGRAM REQUIREMENTS.

I/O Mode Selection Switch	EXTENDED I/O PORT Ø7
Realtime/Keyboard Interrupt Selection Slide Switch	KEYBOARD
Interrupt Toggle Switch	INDIRECT (address)
Parallel I/O Input Switches	SEE DEFINITION ON NEXT SHEET OF DESCRIPTION.

IN CERTAIN APPLICATIONS, THE I/O SWITCHES PROVIDE 2 OR MORE FUNCTIONS. WHEN THIS IS THE CASE, YOU ARE INFORMED WHERE THE DEFINITIONS MAY BE FOUND.

ROUTINE AS SHOWN | START ADDR XXXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION SHIFT BYTE TRANSFERS FOR TRAIN MOTION, ALSO DIRECTION IDENT VIA SENS KEY INTERP. | SHEET 2

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø026	0D	60	3F	REPEAT2	LODA,R0	TEMP-1,R1	SHIFTING 1 BYTE LEFT (FWD)
2	9	CD	40	37		STRA,R0	SWYARD-1,R1,-	DONE? NO. FINISH. YES.
3	C	59	78			BRNR,R1	REPEAT2	MOVE MS BYTE OF TEMP TO
4	2E	08	10			LODR,R0	TEMP	LS BYTE OF SWYARD, THEN
5	3Ø	C8	0D			STRR,R0	SWYARD+7	GO BACK
6	2	1B	4C			BCTR,UN	DISPLAY	AND DISPLAY THE RESULT.
7								
8	Ø038	97	02	06	SWYARD	RES	8	SWITCHYARD IS AN 8-BYTE
9	B	05	00	97				TABLE FOR CURRENT TRAIN
10	3E	12	10					ORIG. ENTRY MESSAGE IS:
11								''2650 UP''
12	Ø040	XX	XX	XX	TEMP	RES	8	TEMP IS TEMPORARY STORAGE
13	3	XX	XX	XX				FOR TRAINS BEING MOTION
14	6	XX	XX					-SHIFTED (LEFT OR RIGHT)
15								
16	Ø048	B4	80		CHKSENS	TPSU	SENS	TEST FOR REVERSE (SENSE
17	A	18	04			BCTR,EQ	DELAY	KEY DOWN) COULD BE: GO FIND
18	C	74	40			CPSU	FLAG	OUT. IT WAS NOT. CLR FLAG
19	4E	1B	48			BCTR,UN	ROLLON	LED, AND EXIT TO ROLLON.
20	5Ø	D8	7E		DELAY	BIRR,R0	$	SET 1 MSEC DELAY & CHK IF
21	2	B4	80			TPSU	SENS	SENS KEY DELIB. DEPR.
22	4	98	42			BCFR,EQ	ROLLON	IT WASN'T. GO FORWARD.
23	6	76	40		REVERSE	PPSU	FLAG	IT WAS. SET THE FLAG LED,
24	Ø058	C1				STRZ	R1	AND SET CONST'S TO REVERSE

THE PROGRAM LISTING IS ALWAYS LOCATED IN THE LOWER HALF OF EACH PAGE.

FORMAT FOR A TYPICAL 8-BYTE MESSAGE. 'RES' IN THE OPCODE FIELD DEFINES A MEMORY BLOCK RESERVATION.

PAGE 2

THE SHEET NUMBER CORRESPONDS TO THE APPLICATION'S DOCUMENTATION

NOTES:

PROGRAMMING NOTE

On some of the listing pages for the various programs, you'll find that NOT all sequential memory locations are programmed. If you use the INSTRUCTOR 50's FAST PATCH or MEMORY DISPLAY & ALTER modes to load user storage from the listing, PROGRAM "FILLER CODES"; H'ØØ', in undefined addresses.

GENERAL CONDITIONS FOR I/O SWITCH POSITIONING ARE INTRODUCED.

DIAGRAMS ARE USED TO CLARIFY COMPLEX OPERATIONS.

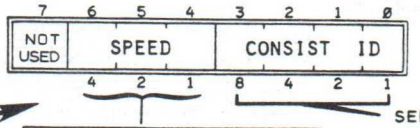
SPECIFIC DIRECTIONS FOR INITIALIZING THE I/O SWITCHES ARE PROVIDED.

TRAIN

USER OPERATION (CONTINUED)

Parallel I/O Input Switches:

The 8 Parallel I/O input switches are position-defined int 2 fields as shown in the following diagram:



SEE TABLE ON NEXT PAGE

#	SPEED	SWI'S ON
7	10 MPH	6 5 4
6	12 "	6 5
5	15 "	6 4
4	19 "	6
3	27 "	5 4
2	40 "	5
1	80 "	4
Ø	STOPPED IN CURRENT POSITION.	ALL OFF

** Toggle speed switches 6, 5, and 4 ON = Speed 7. **

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION CONT. OF CONTROL PROGR; REVERSE CONTROL, ALSO START OF 'SWYRDINT' INTERRUPT SERVICE, INCL. REGISTER VALUE SAVES. | SHEET 3

LINE	ADDRS	DATA			LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
		BØ	B1	B2				
1	ØØ59	07	F9			LODI,R3	H'F9'	DIRECTION. NOW MOVE SWYARD
2	5B	0D	60	38	LOOP1	LODA,R0	SWYARD,R1	TO TEMP; THIS TIME WITH A
3	5E	CD	20	40		STRA,R0	TEMP,R1,+	RIGHT BYTE-SHIFT. 7 BYTES
4	61	DB	78			BIRR,R3	LOOP1	MOVED? NO. FINISH. YES!!!
5	3	08	5A			LODR,R0	SWYARD+7	MOVE LS SWITCHYARD INTO

NOTES:

USER SET-UP PROCEDURES ARE FOLLOWED BY PRECISE DEFINITION OF USER ACTIVITY WHEN THE PROGRAM IS EXECUTED.

A SHORT DESCRIPTION OF THE PROGRAM'S KEY HIGHLIGHTS FOLLOWS THE USER OPERATIONS SEQUENCE. THESE REMARKS SUPPLEMENT THOSE PROVIDED IN THE 'COMMENT' FIELD OF THE LISTING. EACH HIGHLIGHT TYPICALLY DEMONSTRATES SOME UNIQUE TECHNICAL POINT IN SOFTWARE USE.

TRAIN

USER OPERATION: (CONTINUED)

5. Depress RST — If the train consist is stopped.

NOTE: TURN ON AT LEAST ONE OF THE SPEED CONTROL SWITCHES PRIOR TO DEPRESSING THE RST KEY.

6. Toggle input switches 3, 2, 1, or Ø — To select any other consist. Try them all, at different speeds.

7. Depress INT — to move the selected consist into the switchyard.

THE PROGRAM:

There are 2 major points of interest in program "TRAIN"; these include:

* Forward and Reverse Motion Control, and
* Selection & Transfer of the desired Consist into the Switchyard.

FORWARD AND REVERSE MOTION:

Refer to the diagram on the next page, also to the highlights pointed out in the Listing. The description is provided on sheet 8.

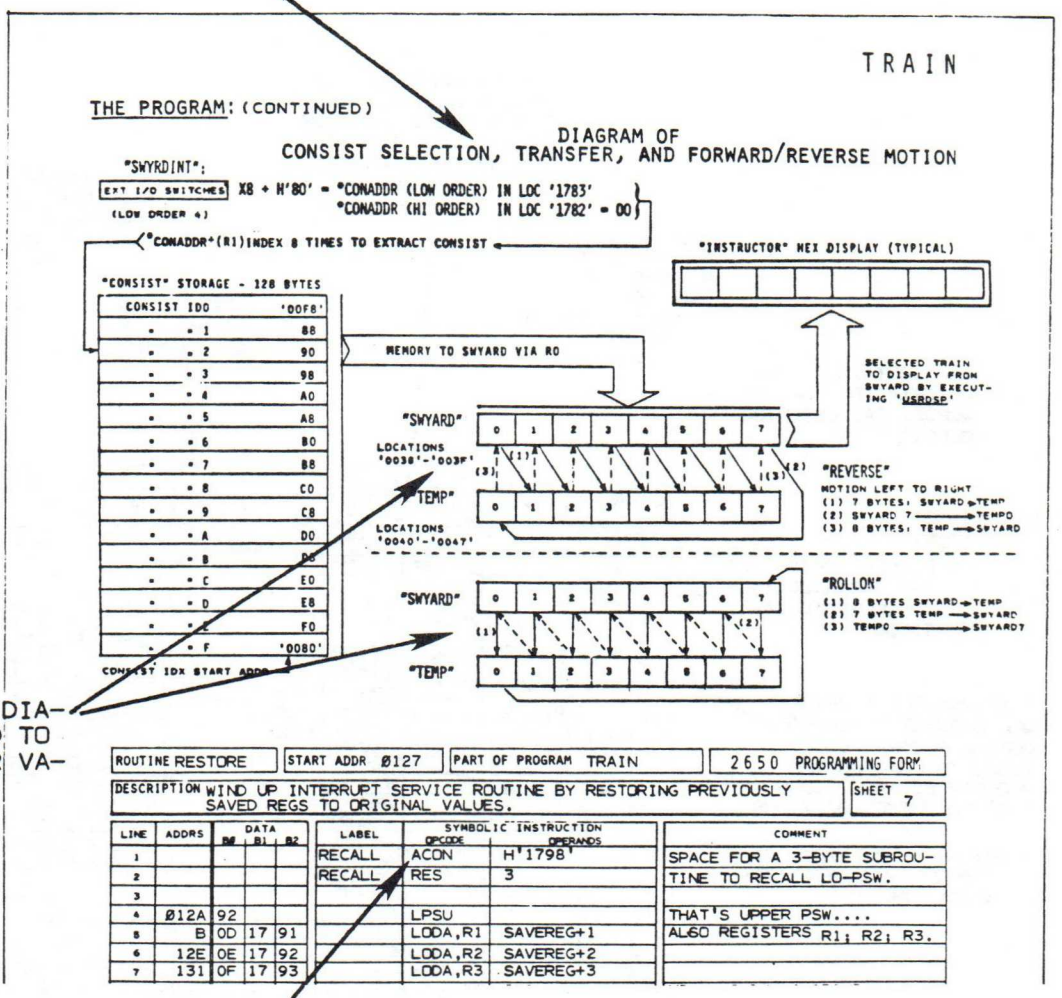
ROUTINE CONTINT | START ADDR Ø1ØØ | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF THE INTERRUPT SERVICE ROUTINE. | SHEET 6

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø1ØØ	13			CONTINT	SPSL		SAVE THE LOWER PSL AT LOC'N
2	1	CC	17	94		STRA,R0	SAVEREG+4	'1794'; AND UPPER PSW AT
3	4	12				SPSU		LOC'N '1795'
4	5	CC	17	95		STRA,R0	SAVEREG+5	
5	8	75	08			CPSL	WC	CLR WC BIT, THEN READ UP
6	A	54	07			REDE,R0	PORT7	MANUAL CONTROL & ISOLATE
7	C	44	0F			ANDI,R0	H'0F'	THE CONSIST I.D.. NOW, MUL-
8	E	D0				RRL	R0	TIPLY THE I.D. BY 8 TO GET
9	1ØF	D0				RRL	R0	AN 8-BYTE INDEX
10	11Ø	D0				RRL	R0	THEN RESTORE THE WC BIT
11	1	77	08			PPSL	WC	FOR FUTURE USE.
12								
13					CONADDR	ACON	H'1782'	

NOTES:

PROGRAM HIGHLIGHTS OF COMPLEX NATURE ARE SUPPORTED BY DIAGRAMS AND OTHER USEFUL TABLES TO MAKE THE LEARNING PROCESS EASIER. YOU ARE URGED TO USE THESE DIAGRAMS WHILE TRACING THE FLOW OF THE PROGRAM IN THE LISTING. BY DOING THIS, YOU'LL 'SEE' PROGRAM EXECUTION AS THE COMPUTER DOES!!



KEY PARTS OF THE DIAGRAM ARE REPEATED TO DEMONSTRATE MAJOR VARIATIONS.

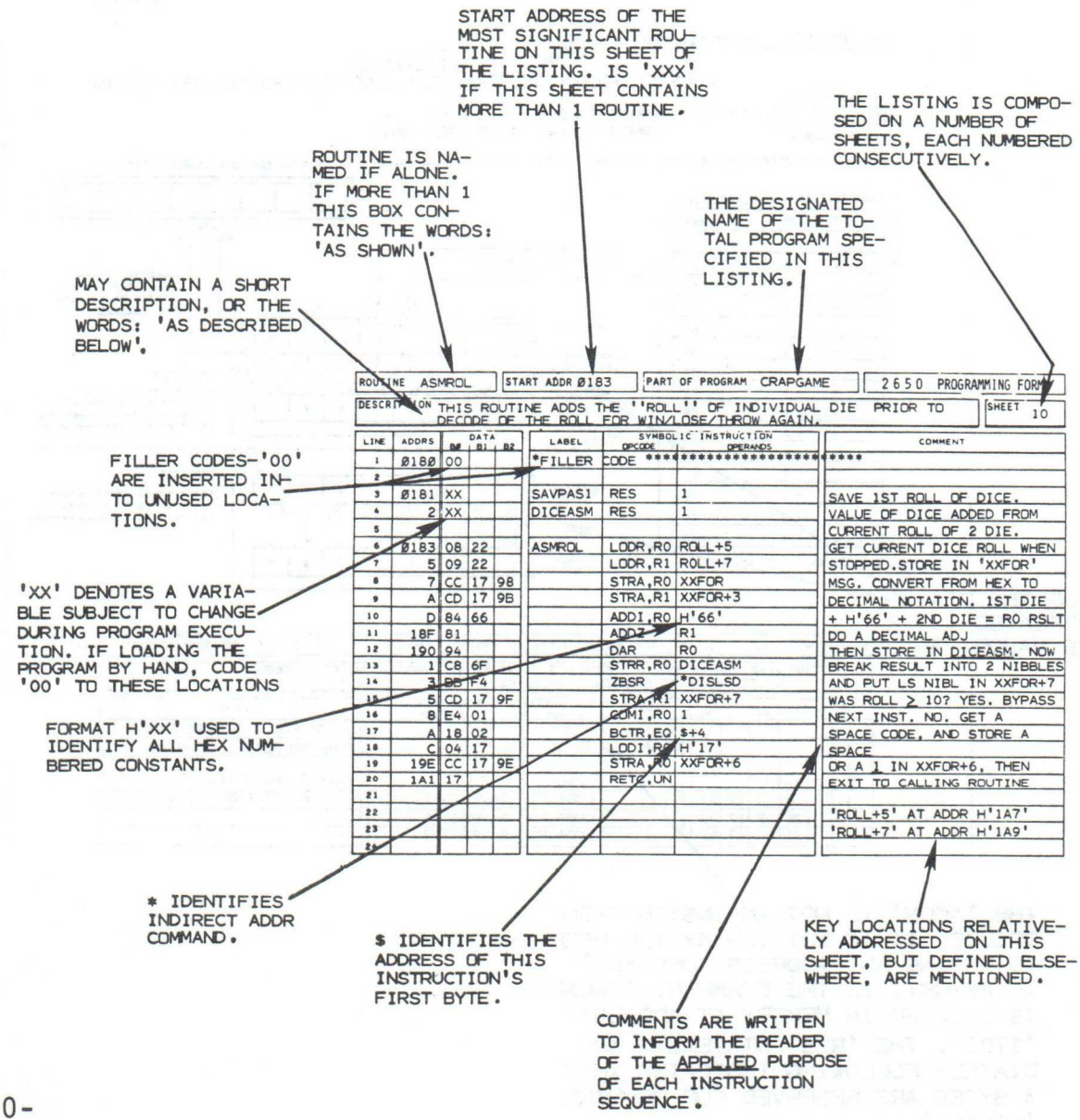
ROUTINE RESTORE | START ADDR Ø127 | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION WIND UP INTERRUPT SERVICE ROUTINE BY RESTORING PREVIOUSLY SAVED REGS TO ORIGINAL VALUES. | SHEET 7

LINE	ADDRS	DATA			LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
		R0	R1	R2				
1					RECALL	ACON	H'1798'	SPACE FOR A 3-BYTE SUBROU-
2					RECALL	RES	3	TINE TO RECALL LO-PSW.
3								
4	Ø12A	92				LPSU		THAT'S UPPER PSW....
5	B	0D	17	91		LODA,R1	SAVEREG+1	AL60 REGISTERS R1; R2; R3.
6	12E	0E	17	92		LODA,R2	SAVEREG+2	
7	131	0F	17	93		LODA,R3	SAVEREG+3	

THE 'ACON' IS NOT AN INSTRUCTION OPCODE. IT IS WRITTEN BY THE USER TO DEFINE AN 'ADDRESS CONSTANT' IN MEMORY. IN THE EXAMPLE, 'RECALL' IS LOCATED IN MEMORY AT ADDRESS '1798'. THE 'RES' STATEMENT IMDIATELY FOLLOWING INDICATES THAT 3 BYTES ARE RESERVED FOR ROUTINE 'RECALL'.

NOTES:



LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø18Ø	00			*FILLER	CODE	**********	**********
2								
3	Ø181	XX			SAVPAS1	RES	1	SAVE 1ST ROLL OF DICE.
4	2	XX			DICEASM	RES	1	VALUE OF DICE ADDED FROM
5								CURRENT ROLL OF 2 DIE.
6	Ø183	08	22		ASMROL	LODR,R0	ROLL+5	GET CURRENT DICE ROLL WHEN
7	5	09	22			LODR,R1	ROLL+7	STOPPED.STORE IN 'XXFOR'
8	7	CC	17	98		STRA,R0	XXFOR	MSG. CONVERT FROM HEX TO
9	A	CD	17	9B		STRA,R1	XXFOR+3	DECIMAL NOTATION. 1ST DIE
10	D	84	66			ADDI,R0	H'66'	+ H'66' + 2ND DIE = R0 RSLT
11	18F	81				ADDZ	R1	DO A DECIMAL ADJ
12	190	94				DAR	R0	THEN STORE IN DICEASM. NOW
13	1	C8	6F			STRR,R0	DICEASM	BREAK RESULT INTO 2 NIBBLES
14	3	BB	F4			ZBSR	*DISLSD	AND PUT LS NIBL IN XXFOR+7
15	5	CD	17	9F		STRA,R1	XXFOR+7	WAS ROLL ≥ 10? YES. BYPASS
16	8	E4	01			COMI,R0	1	NEXT INST. NO. GET A
17	A	18	02			BCTR,EQ	$+4	SPACE CODE, AND STORE A
18	C	04	17			LODI,R0	H'17'	SPACE
19	19E	CC	17	9E		STRA,R0	XXFOR+6	OR A 1 IN XXFOR+6, THEN
20	1A1	17				RETC,UN		EXIT TO CALLING ROUTINE
21								
22								'ROLL+5' AT ADDR H'1A7'
23								'ROLL+7' AT ADDR H'1A9'
24								

PROGRAMMABLE BILLBOARD

INTRODUCTION:

The first of a variety of programs provided on the INSTRUCTOR 50 INTRODUCTORY CASSETTE tape, "BILLBOARD" provides a means for you to prepare, then display your own messages on the INSTRUCTOR 50. The maximum message length is 254 separate characters, including space codes (Blanks), and other graphics. In it's taped version, the message: "HI, THIS IS THE 2650." is displayed. All messages move from right to left across the display window.

The following table provides the code for each character of the display "font". Use it to prepare your messages. In addition, you may experiment with any unlisted codes in order that you may generate "special graphics". In program "TRAIN" (located elsewhere in this chapter), special graphics are used to define the rolling stock of each train.

CHARACTER FONT:

SYMBOL	CODE	SYMBOL	CODE	SYMBOL	CODE
Ø,O	00	B	0B	0	15
1,I	01	C	0C	=	16
2,Z	02	D	0D	blank	17
3	03	E	0E	J	18
4	04	F	0F	- (DASH OR MINUS)	19
5,S	05	P	10	*	1A
6,G	06	L	11	Y	1B
7,T	07	U	12	▭ (SMALL BOX)	95
8	08	R	13	N	1C
9	09	H	14	T	6C
A	0A				

NOTE: Add H'80' to the character code if you desire a RIGHT-JUSTIFIED DECIMAL POINT. Terminate each message with the code: H'FF'.

USER OPERATION:

Depress RST to start "BILLBOARD". None of the INSTRUCTOR 50 control switches are used; no special positioning is required.

THE PROGRAM:

The listing for program "BILLBOARD" is provided on the next 2 pages. The first sheet of listing is given over to a brief description of the program, followed by definition of various symbols in terms of their HEX CODE equivalents. EQUate statements are input to the ASSEMBLER to perform this function. Before any symbol can be used by an assembler, it must be defined.

On the following sheet of listing, ORG (Program ORiGin) statements define both "BILLBOARD"'s start address, and the location of the MESSAGE's first byte. You may insert the message of your choice, starting at location H'101'. In line 78, a memory REServe statement dedicates a single byte, at address H'100', for use as a 'message pointer. The current 8 byte message is displayed, then the pointer is incremented. The 8 code bytes immediately following the memory location defined by the pointer are displayed, with the pointer being used as the LEAST SIGNIFICANT byte of address.

```
TWIN ASSEMBLER VER 2.0    BILLBOARD PROGRAM 17 APR 78        PAGE 0001

LINE ADDR  OBJECT  E SOURCE

0002              ********************************************************  ********
0003              *
0004              *PROGRAM WRITTEN BY JOHN KEENAN
0005              *
0006              *THIS PROGRAM IS WRITTEN FOR THE INSTRUCTOR 50
0007              *
0008              *THIS PROGRAM DISPLAYS THE MESSAGE IN THE DISPLAY BUFFER
0009              *
0010              *THE MESSAGE WILL WILL REAPPEAR ON THE DISPLAY PANNEL
0011              *AT REGULAR INTERVALS TO GIVE THE EFFECT OF A ROTATING
0012              *BILLBOARD.
0013              *
0014              *THE MAXIMUM MESSAGE IS 254 CHARACTERS
0015              *THE MESSAGE IS ENTERED STARTING AT LOCATION H'101'. PROGRAM LABEL 'MSG'
0016              *THE END OF MESSAGE IS INDICATED BY THE VALUE OF H'FF' AS THE LAST
0017              *CHARACTER OF THE MESSAGE.
0018              ************************************************************************
0019              * STANDARD SYMBOL DEFINITION - THIS FILE MAY BE APPENDED TO THE
0020              *                              FRONT OF ANY USER'S SOURCE DECK
0021                                  *   REGISTER EQUATES
0022 0000                            R0      EQU     0       REGISTER 0
0023 0001                            R1      EQU     1       REGISTER 1
0024 0002                            R2      EQU     2       REGISTER 2
0025 0003                            R3      EQU     3       REGISTER 3
0026                                 *  CONDITION CODES
0027 0001                            P       EQU     1       POSITIVE RESULT
0028 0000                            Z       EQU     0       ZERO RESULT
0029 0002                            N       EQU     2       NEGATIVE RESULT
0030 0002                            LT      EQU     2       LESS THAN
0031 0000                            EQ      EQU     0       EQUAL TO
0032 0001                            GT      EQU     1       GREATER THAN
0033 0003                            UN      EQU     3       UNCONDITIONAL
0034                                 *  PSW LOWER EQUATES
0035 0000                            CC      EQU     H'00'   CONDITIONAL CODES
0036 0020                            IDC     EQU     H'20'   INTERDIGIT CARRY
0037 0010                            RS      EQU     H'10'   REGISTER BANK
0038 0008                            WC      EQU     H'08'   1=WITH 0=WITHOUT CARRY
0039 0004                            OVF     EQU     H'04'   OVERFLOW
0040 0002                            COM     EQU     H'02'   1=LOGIC 0=ARITHMETIC COMPARE
0041 0001                            C       EQU     H'01'   CARRY/BORROW
0042                                 *  PSW UPPER EQUATES
0043 0080                            SENS    EQU     H'80'   SENSE BIT
0044 0040                            FLAG    EQU     H'40'   FLAG BIT
0045 0020                            II      EQU     H'20'   INTERRUPT INHIBIT
0046 0007                            SP      EQU     H'07'   STACK POINTER
0047                                 * END OF EQUATES
0048                                 ************************************************
```

```
TWIN ASSEMBLER VER 2.0   BILLBOARD PROGRAM 17 APR 78        PAGE 0002

LINE ADDR  OBJECT  E SOURCE

0050 0000                    ORG    0       SET THE BEGINNING OF PROGRAM TO LOCATION 0
0051                    *
0052 0000 1B22          BEGIN BCTR,UN INIT   SET LOWER REGISTER BANK
0053 0002 0560          START LODI,R1 H'60'  LOAD THE DELAY COUNTER
0054 0004 7710          DISPL PPSL    RS     SET THE UPPER REGISTER BANK
0055 0006 0501                LODI,R1 <MSG   LOAD THE UPPER BYTE OF MSG POINTER ADDRESS
0056 0008 0E0100              LODA,R2 PNTR   LOAD THE LOWER BYTE OF MESSAGE POINTER ADDRESS-1
0057 000B 0701                LODI,R3 01     LOAD THE 1 PASS COMMAND PARAMETER TO THE DISPLAY
0058 000D BBE6                ZBSR    *USRDSP EXIT TO THE DISPLAY ROUTINE FOR 1 PASS
0059 000F 7510                CPSL    RS     SELECT THE LOWER REGISTER BANK
0060 0011 FD0004              BDRA,R1 DISPL  DECREMENT THE DELAY COUNTER AND CHECK FOR END
0061                    *                    IF NOT AT END DISPLAY THE SAME MESSAGE TILL
0062                    *                    COUNT = 0
0063 0014 0E0100              LODA,R2 PNTR   LOAD POINTER TO MESSAGE
0064 0017 8601                ADDI,R2 1      INCREMENT IT
0065 0019 CE0100              STRA,R2 PNTR   SAVE THE NEW POINTER VALUE
0066 001C 0E6109              LODA,R0 MSG+8,R2       LOAD THE NEXT CHARACER TO BE DISPLAYED
0067 001F E4FF                COMI,R0 H'FF'  IS IT THE END OF MESSAGE CHARACTER
0068 0021 9C0002              BCFA,EQ START   IF NO GO DISPLAY THE MESSAGE ROTATED LEFT 1
0069                    *                    CHARACTER
0070 0024 20            INIT  EORZ   R0      IF YES, RESET MESSAGE POINTER TO BEGINNING OF
0071 0025 93                  LPSL           MESSAGE
0072 0026 CC0100              STRA,R0 PNTR   SET POINTER TO BEGINNING
0073 0029 1F0002              BCTA,UN START  GO DISPLAY THE MESSAGE FROM THE BEGINNING
0074                    *
0075                    *
0076 002C                    ORG    H'100'  THIS IS THE DATA AREA FOR THE PROGRAM
0077                    *
0078 0100               PNTR  RES    1       1 LOCATION TO SAVE THE LEAST SIGNIFICANT BYTE OF
0079                    *                    MESSAGE POINTER
0080                    *
0081                    *
0082                    *THIS IS THE INITIAL MESSAGE IN THE BUFFER
0083                    *
0084                    *THE MESSAGE IS 'HI THIS IS THE 2650....HI THIS'
0085                    *
0086                    *
0087 0101 14011707      MSG   DATA   H'14,01,17,07,14,01,05,17,01,05,17' HI THIS IS
     0105 14010517
     0109 010517
0088 010C 07140E17            DATA   H'07,14,0E,17,02,06,05,00'          THE 2650
     0110 02060500
0089 0114 1A1A1A1A            DATA   H'1A,1A,1A,1A,14,01,17,07,14,01,05,17' ....HI THIS
     0118 14011707
     011C 14010517
0090 0120 FF                  DATA   H'FF'                               END OF MESSAGE FLAG
0091                    *
0092                    *
0093 0121                    ORG    H'1FE6' LOCATION OF POINTER IN MONITOR TO DISPLAY ROUTINE
0094 1FE6               USRDSP EQU    $
0095                    *
0096                    *
0097 0000                    END    BEGIN
```

4 - FUNCTION DESK CLOCK

INTRODUCTION:

A full 4-function desk clock is implemented via the facilities of the INSTRUCTOR 50 when program "DESCLK" is loaded and executed. These functions include HOURS, MINUTES, SECONDS, and a distinctive AM/PM indication. All four time functions are preset from the HEX KEYBOARD; there is no need to use the INSTRUCTOR 50's "Display and Alter Memory" mode once the program is loaded. Special command requests are asserted in sequence to prompt the user to enter expected time parameters.

Once the time functions are entered, the set time is displayed in static mode. When the operator determines that the displayed time equals the exact time-of-day, he depresses the SENS key to start the clock running.

The desk clock is equally at home in a 50 Hz. environment. The operator needs only to modify one byte of data in the control program to change the clock's activity from 60 to 50 Hz. Minor modification of the program also permits usage of the desk clock as an interval timer, capable of displaying accurately in 1/10th second increments.

Typically, the operator will preset the desk clock a minute

ROUTINE AS SHOWN	START ADDR ØØØØ	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM

DESCRIPTION A PROGRAM TO SET AND SYNCHRONIZE A 4-FUNCTION ''DESK CLOCK'' TO ANY DESIRED TIME OF DAY, AND DISPLAY FOREVER. SHEET 1

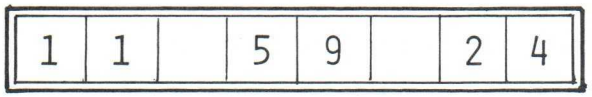
LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØØØ	76	20		DESCLK	PPSU	II	ON ENTRY, DISABLE INTPTS.
2	2	1B	07			BCTR,UN	READY	& GET READY TO SET TIME.
3	4	01	40		AMORPM	ACON	AORPM	IND ADDR TO SET AM OR PM
4	6	ØØ			*FILLER	CODE		
5	7	1B	30		IACLK1	BCTR,UN	CLK1	INTPT ADDR VECTOR TO CLOCK
6								
7	9	01	90		SETIME	ACON	SETIME1	IND ADDR TO SET TIME OF DAY
8								
9	B	74	40		READY	CPSU	FLAG	CLOCK IN 'SET' MODE. TURN
10	D	BB	84			ZBSR	*AMORPM	OFF RUNNING LIGHT. NOW,GO
11	ØF	BB	89			ZBSR	SETIME	SET AM OR PM. ON RTN, GO
12	11	75	FF			CPSL	ALL	SET TIME-OF-DAY. CLR PSL
13	3	05	00		CONT	LODI.R1	<DISCLK-1	ON RTN, THEN INIT. TO SHOW
14	5	06	FF			LODI,R2	>DISCLK-1	SET TIME (STATIC) OR ACTUAL
15	7	07	01			LODI,R3	1	TIME(DYNAM.) DISPL FOR ONE
16	9	BB	E6			ZBSR	*USRDSP	PASS. IS TIME SET BUT NOT
17	B	B4	40			TPSU	FLAG	RUNNING? YES. GO SYNCH TO
18	D	1A	07			BCTR,LT	CHKSENS	TRUE TIME. NO! ITS RUNNING
19	1F	74	20			CPSU	II	GO OPEN THE INTPT WINDOW
20	21	CO				NOP		FOR 6 USEC. IF NO INTPT
21	2	76	20			PPSU	II	SHUT OFF, AND DISPLAY THE
22	4	1B	6D			BCTR,UN	CONT	CURRENT TIME. SYNCH BY DE-
23	6	B4	80		CHKSENS	TPSU	SENSE	PRESSING SENS. NO SYNC YET
24	ØØ28	98	69			BCFR,EQ	CONT	DISPLAY THE TIME O'DAY

A JOHN GARCEAU PROGRAM

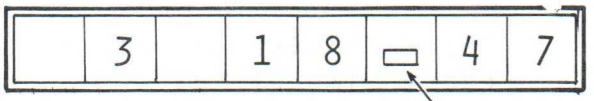
INTRODUCTION: (CONTINUED)

or so fast, then call the local telephone company's time check service to synchronize the clock. Since this clock uses the AC line frequency as a time source, it can not lose (or gain) time unless power is lost.

The leading hours digit is blanked automatically when the time cycles from 12:59 to 1:00, remaining blanked until 10:00 o'clock. All times between 12:00 noon and 11:59::59 PM are identified by the PM indicator, a small box located between the minutes and seconds display. Absence of the small box defines AM times. Typical time displays are shown below:



"11:59::24 AM"



"3:18::47 PM"

User operation is defined on the next page.

ROUTINE CLK | START ADDR AS IND. | PART OF PROGRAM DESCLK | 2650 PROGRAMMING FORM

DESCRIPTION 'CLK' IS THE REAL-TIME INPT SEVICE ROUTINE WHICH INCREMENTS SECONDS, MINUTES, & HOURS TO PROVIDE TRUE TIME OF DAY (OR NIGHT) | SHEET 2

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	002A	76	40			PPSU	FLAG	ITS SYNC'D CLOCK IS RUNN'G
2	2C	1B	65			BCTR,UN	CONT	SO GO SHOW CURRENT TIME OF
3								DAY.
4	30	C0	C0	C0	CLK1	NOP		ENTRY ONLY TO 'CLK'ROUTINE
5	3	C0	C0	C0		NOP		WHEN 60 (50)HZ AC INTPT
6	6	C0	C0	C0		NOP		TAKES PLACE. NOTE LOC '3F'
7	39	0D	01	10	CLK	LODA,R1	FREQCT	COUNT XXHZ/SEC BY INCR
8	C	85	01		INCFRQ	ADDI,R1	1	FREQ CTR TO ALTERN.
9	003E	E5	3C			COMI,R1	60	ALTERN: IF 60 HZ LOCAL
10	003E	E5	32			COMI,R1	50	ALTERN: IF 50 HZ LOCAL
11	40	18	03			BCTR,EQ	INCSEC	COUNT HZ FOR 1 SEC. DONE?
12	42	C9	F6			STRR,R1	*CLK+1	GO INCR SECS. NO, STORE
13	4	17				RETE,UN		CURRENT COUNT AWAY & GO
14								WAIT FOR NXT INTERRUPT!!!
15	45	20			INCSEC	EORZ	R0	1ST, RESET FREQCT,THEN
16	6	CC	01	10		STRA,R0	FREQCT	FETCH
17	9	0C	01	07		LODA,R0	LSSEC	AND INCREMENT THE
18	C	84	01			ADDI,R0	1	CURRENT LEAST SIG SES.
19	4E	E4	0A			COMI,R0	10	COUNT 10 SECS YET? NO.
20	50	18	04			BCTR,EQ	$+6	STORE LS SECS
21	2	CC	01	07		STRA,R0	LSSEC	AWAY, AND
22	5	37	.			RETE,UN		RETURN TO DISPLAY.? YES:
23	6	20				EORZ	R0	GO CALCUL. MS SEC, BUT 1ST
24	0057	CC	01	07		STRA,R0	LSSEC	ZERO AND STORE LSSEC.

USER OPERATION - SETUP:

After loading program "DESCLK" into the INSTRUCTOR 50's memory, the switches should be positioned as follows:

I/O Mode Selection Switch	EXTENDED I/O PORT Ø7
Parallel I/O Input Switches (8)	any position
AC Line / Keyboard Interrupt Selection Slide Switch	AC LINE
Interrupt Address Toggle Switch	DIRECT

USER OPERATION - PRESET TIME OF DAY

The following operations are performed in order to set the exact time of day:

1. Depress RST.....*The FLAG LED, if on, turns off. The Hex Display provides the following message for 1½ seconds:*

"SET CLOC" .....

ROUTINE CLK | START ADDR BELOW | PART OF PROGRAM DESCLK | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUE AS DESCRIBED BELOW | SHEET 3

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ5A	0C	01	06		LODA,R0	MSSEC	THEN FETCH & INCREMENT
2	D	84	01			ADDI,R0	1	THE MOST SIG SECS
3	5F	E4	06			COMI,R0	6	HAVE WE CLOCKED A FULL MI-
4	61	18	04			BCTR,EQ	INCMIN	NUTE YET? YES: GO INCREM.
5	3	CC	01	06		STRA,R0	MSSEC	MINUTES. NO: STORE MS SECS
6	66	37				RETE,UN		AND RETURN TO DISPLAY TO
7								WAIT FOR INTPT.
8	67	20			INCMIN	EORZ	R0	COMMENTS SAME AS FOR INCSEC
9	8	CC	01	06		STRA,R0	MSSEC	THE FOLLOWING COMMENTS DE-
10	B	0C	01	04		LODA,R0	LSMIN	SCRIBE VARIABLES TO BE A-
11	6E	84	01			ADDI,R0	1	NALYZED IN CONNECTION WITH
12	7Ø	E4	0A			COMI,R0	10	HOURS INCR & RESET, ALSO
13	2	18	04			BCTR,EQ	$+6	'AM/PM' INDICATIONS: FROM
14	4	CC	01	04		STRA,R0	LSMIN	12 NOON TO 12 MIDNITE, PM
15	7	37				RETE,UN		IND IS ON. OFF FOR AM HOURS
16	8	20				EORZ	R0	LS HRS MUST INCR. TO 9
17	9	CC	01	04		STRA,R0	LSMIN	IF MS HRS (1) IS OFF. IF
18	C	0C	01	03		LODA,R0	MSMIN	MS HRS IS 1, LS HRS INCR'S
19	7F	84	01			ADDI,R0	1	TO 2. EG:TO 12:00.
20	81	E4	06			COMI,R0	6	MS DIGITS OF HOURS IS BLAN-
21	3	18	04			BCTR,EQ	TESTHRS	KED ON TIME FROM 12:59::59
22	5	CC	01	03		STRA,R0	MSMIN	TO 1:00 0'CLK.
23	7	37				RETE,UN		CARRY FROM 10:59::59 TO
24								11:00 0'CLOCK IS A SPECIAL

USER OPERATION (CONTINUED)

.....*followed by the message:*
"A OR P ="

2. * Depress 'A' .....*If it is morning.*
   * Depress any other HEX key..... .....*If after noon, and before midnight.*

3. Depress ENT/NXT .....*to set the AM/PM parameter. The message: "HOURS =" is displayed.*

4. Depress 1 or 2 HEX digits .....*Enter the correct time in hours.*
   LIMITS: BETWEEN 1 AND 9 (1 DIGIT) OR 10 TO 12 (2 DIGITS).

5. Depress ENT/NXT .....*To set the HOURS parameter. The message: "FRACT =" is displayed.*

6. Depress 1 or 2 HEX digits .....*Enter the correct time in minutes. (This is fraction of an hour.)*
   LIMITS: BETWEEN 0 AND 59.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM DESCLK | 2650 PROGRAMMING FORM

DESCRIPTION COMMENTS ON PREVIOUS SHEET OF LISTING ARE NOW IMPLEMENTED. | SHEET 4

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØ89	20			TESTHRS	EORZ	R0	1ST, ZERO MS MIN IN PREP TO
2	A	CC	01	03		STRA,R0	MSMIN	CHANGE HRS. FETCH ALL HRS.
3	8D	0C	01	00		LODA,R0	MSHRS	SEE IF ITS 10:00 OR LATER
4	90	0D	01	01		LODA,R1	LSHRS	NO: ITS EARLIER, SO INCRE-
5	3	F4	17			TMI,R0	H'17'	MENT THE LS HR.
6	95	18	09			BCTR,EQ	INCHRS	ROUTINE 'HIHRS' HANDLES
7								FROM 10:00 ON:
8	97	85	01		HIHRS	ADDI,R1	1	NOW INCR LSHRS
9	9	E5	02			COMI,R1	2	IS IT 12:00 YET?
10	B	38	1B			BSTR,EQ	AMPM	YES: GO SWITCH THE AM/PM
11	9D	1F	00	E0		BCTR,UN	TENELEV	INDICATOR. NO, MUST BE 10:
12								GOING ON 11:00 SO SERVICE
13								THAT UNIQUE CASE.....
14	AØ	85	01		INCHRS	ADDI,R1	1	ROUTINE 'INCHRS' INCRE-
15	2	E5	0A			COMI,R1	10	MENTS THE HOURS
16	4	18	04			BCTR,EQ	$+6	WHEN ACCESSED BETWEEN
17	6	CD	01	01		STRA,R1	LSHRS	1:00 AND 10:00. SAME COM-
18	9	37				RETE,UN		MENT AS GIVEN FOR
19	A	05	00			LODI,R1	Ø	DESCRIBING SECONDS
20	C	CD	01	01		STRA,R1	LSHRS	INCREMENTING.
21	AF	04	01			LODI,R0	1	
22	B1	CC	01	00		STRA,R0	MSHRS	
23	4	37				RETE,UN		
24	00B5	C0	C0	C0		NOP		3 NOPS.

USER OPERATION (CONTINUED)

7. Depress ENT/NXT .....*To set the MINUTES parameter. The message:*

" S E C S = "

...is displayed.

8. Depress 1 or 2 HEX digits .....*enter the anticipated value for time synchronization in seconds.*

LIMITS: BETWEEN 0 AND 59.

NOTE: BEFORE ENTERING SECONDS, DIAL YOUR LOCAL TIME CHECK NUMBER (767-1111; MOST PLACES) TO DETERMINE THE PRECISE TIME IF DESIRED. THE 'SECONDS' VALUE ENTERED IN STEP 8 MAY PRECEED SYNCH TIME BY 10 TO 20 SECONDS.

9. Depress ENT/NXT.....*to set the anticipated value of time in seconds.*

The message: AM/PM INDIC.

" H H M M x S S "

...is displayed.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM DESCLK | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF TESTING FOR DEFINED CONDITIONS IN TIME OF DAY | SHEET 5

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØB8	0E	01	05	AMPM	LODA,R2	PMIND	ON ENTRY, ITS 12:00, SO
2	B	E6	95			COMI,R2	H'95'	GET THE AMPM INDICATOR, &
3	D	18	06			BCTR,EQ	SETAM	SEE IF ITS PM. YES? SET
4	BF	06	95			LODI,R2	H'95'	THE AM (SPACE) IND. NO?
5	C1	CE	01	05	STORIT	STRA,R2	PMIND	SET THE PM INDIC. &
6	4	17				RETC,UN		RETURN TO HI HOURS
7	5	06	17		SETAM	LODI,R2	SPACE	TO SET AM, GET A SPACE
8	7	1B	78			BCTR,UN	STORIT	CODE, AND PUT IN THE AM/PM
9	9	00	00	00				FIELD OF THE DISPLAY MSG.
10	C	00	00		*FILLER	CODE		
11	CE	04	17		END	LODI,R0	H'17'	ON ENTRY, INIT. A SPACE
12	DØ	CC	01	00		STRA,R0	MSHRS	FOR MS HRS, AND A 1 FOR
13	3	04	01			LODI,R0	1	LS HRS. ITS 1:00::00 FOR
14	5	CC	01	01		STRA,R0	LSHRS	NEXT DISPL, SO SHOW IT.
15	8	37				RETE,UN		
16	D9-DF	00	00	00	*FILLER	CODE		ON ENTRY, STORE THE PREVI-
17	EØ	CD	01	01	TENELEV	STRA,R1	LSHRS	OUSLY INCREM LS HRS. IS
18	3	F5	03			TMI,R1	3	IT 12:00 GOING ON 1? NO:
19	5	36				RETE,LT		RETURN; IT IS 11:59, NOT
20	E6	1F	00	CE		BCTA,UN	END	12:59. YES: ITS 1:00, SO
21	E9-FF	00	00	00	*FILLER	CODE		RESET LS HRS TO 1 !!!!
22	Ø1ØØ	XX	XX	17	DISCLK	RES	8	MESSAGE TABLE TO DISPLAY
23	3	XX	XX	XX				THE CURRENT TIME:
24	1Ø6	XX	XX					HRS - MIN - AM/PM IND -SEC

USER OPERATION (CONTINUED)

10. Depress SENS .....*At the exact time of synchronization.*
The FLAG LED turns on to indicate the clock is running.
The Hex Display provides a continuous indication of the true 'time of day'.

THE PROGRAM:

There are 3 major functions included in program "DESCLK"; these include:

1. a time preset sequence, during which the INSTRUCTOR 50's User Display and Get Numbers subroutines are accessed in order for the user to set the clock.
2. a REAL-TIME Interrupt sequence, in which the actual high-speed incrementation of time takes place, with resulting time values stored in a special clock display buffer (at address h'1ØØ').

ROUTINE AORPM	START ADDR Ø14Ø	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM
DESCRIPTION 'AORPM' IS ACCESSED DURING SET CLOCK TIME TO IDENTIFY AM OR PM			SHEET 6

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø11Ø	XX			FREQCT	RES	1	TEMP STORAGE FOR FREQ COUNT
2					*UNASSIGNED ADDRESSES CONTAIN FILLER CODES = H'00'.			
3	Ø13D	01	10		FREQZ	ACON	H'Ø11Ø	IND ADDR TO FREQCT
4								
5	Ø14Ø	3F	01	C0	AORPM	BSTA,UN	ORDER	1ST, TELL USER TO SET CLOCK
6	3	05	01			LODI,R1	<AMPMES-1	ON RETURN, SET & DISPLAY
7	5	06	60			LODI,R2	>AMPMES-1	A PROMPT FOR HIM TO SET THE
8	7	BB	FE			ZBSR	*MOV	AM OR PM INDICATOR. HE'LL
9	9	C0				NOP		DEPR. 'A' FOR AM & ANY
10	A	04	05			LODI.R0	5	OTHER HEX KEY FOR PM. AFTER
11	C	BB	FC			ZBSR	*GNPA	HE MAKES UP HIS MIND,SEE
12	E	C0				NOP		IF ITS MORNING? YES! GO SET
13	4F	E4	0A			COMI,R0	H'0A'	THE AM INDICATOR.
14	151	18	04			BCTR,EQ	MORNING	AM IS A SPACE BETWEEN MIN'S
15	3	04	95		EVENING	LODI,R0	H'95'	AND SECS. NO!, ITS PM! PM
16	5	1B	02			BCTR,UN	$+4	IS A SMALL BOX BETWEEN
17	7	04	17		MORNING	LODI,R0	H'17'	MIN'S & SECS IN TIME DSPLAY
18	9	CC	01	05		STRA,R0	PMIND	NOW INIT. THE FREQ. CTR.,
19	C	20				EORZ	R0	TO ZERO, THEN BRANCH TO
20	D	C8	DE			STRR,R0	*FREQZ	COMPLETE THE CLK MESSAGE
21	15F	1B	08			BCTR,UN	SPCKLK	
22	161	0A	17	95	AMPMES	RES	8	'A OR P= XX'
23	4	13	17	10				
24	7	16	17					

THE PROGRAM (CONTINUED)

3. a series of routines designed to handle anomalies and exceptions from the strict increment method used in incrementing the time functions. These include:

   * "HIHRS" .....*a routine which tests upper hour changes to see if it is time to change the AM/PM indicator.*

   * "TESTHRS".....*This routine checks to see if it is 10:00 yet. If so, TWO hour-digit changes are necessary.*

   * "AMPM".....*Controls set or reset of the AM/PM indication at 12:00.*

   * "END".....*Controls initialization of hours at 1:00 0'clock.*

In the listing, sheet 1, the routine "CONT" is employed in static time mode to test for activity of the SENS input after the clock is preset. This test is determined by FLAG (not active), and branch to subroutine "CHKSENS". Once SENS is depressed, the FLAG bit is set. Thereafter, routine "CONT alternately displays the 'Time of day', and tests for activity of the 60 (50) Hz AC LINE input. Until SENS is depressed, all interrupts are inhibited, and the real-time interrupt is not tested. Tight control of interrupt recognition is maintained by opening an 'interrupt window' only during a precise time in the sequence.

ROUTINE AS SHOWN	START ADDR XXX	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM
DESCRIPTION THE FOLLOWING ROUTINE, DESCRIBED AS SHOWN, ARE USED IN SEQUENCING INITIALIZATION AND PRECISE SET OF THE DESK CLOCK 4-FUNCS			SHEET 7

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø17Ø	06	03		BLNKZ	LODI,R2	3	ON ENTRY, TIME READY TO BE
2	2	EE	01	BB		COMA,R2	EVENTS	PUT INTO CLOCK MESSAGE.
3	5	98	04			BCFR,EQ	NOBLNK	IS IT HOURS? IF NOT, EXIT.
4	7	58	02			BRNR,R0	NOBLNK	YES. IS IT BETWEEN 1 & 9PM
5	9	04	17			LODI,R0	SPACE	??? NO; LEAVE MS HOUR AS IS
6	17B	17			NOBLNK	RETC,UN		YES; GET A SPACE CODE FOR
7					*NOTE CHANGE IN ORDER ********			MS HOUR, & EXIT.
8					*OF CODE ADDRESSES **********			
9	Ø169	04	17		SPCKLK	LODI,R0	H'17'	TO INIT. CLOCK MESSAGE, SET
10	B	CC	01	02		STRA,R0	DISCLK+2	A SPACE CODE IN ITS 3RD DI-
11	E	17				RETC,UN		GIT, THEN RETURN.
12								
13								ON ENTRY, SET UP MS BYTE
14	Ø18Ø	05	01		NOWSET	LODI,R1	<SELMES-1	OF ADDR, THEN MOVE THE SE-
15	2	BB	FE			ZBSR	*MOV	LECTED MESG INTO THE DSPLY
16	4	C0				NOP		BUFFER.
17	5	04	01			LODI,R0	1	NOW SET UP FOR 2-DIGIT TIME
18	7	BB	FC			ZBSR	*GNPA	ENTRY (HRS, MINS, SECS)
19	9	C0				NOP		AFTER USER SETS THIS TIME,
20	A	BB	F4			ZBSR	*DISLSD	BREAK THE SET TIME INTO 2
21	C	3B	62			BSTR,UN	BLNKZ	DIGITS, & IF HORS, GO SEE
22	18E	17				RETC,UN		IF BLANK IS REQU. ON RETURN
23								GO STORE THE TIME.
24								

THE PROGRAM (CONTINUED)

In sheet 2, a modulo 60 (50) counter is established by the ADD and COMI instructions (LINES 8 &9). The frequency counter (FREQCT) is reinitialized to zero in routine "INCSEC" (LINES 15 AND 16)

Modulo 10 and modulo 6 counters are employed in the processes "INCSEC", and "INCMIN", thus providing for orderly incrementation of seconds and minutes from Ø to 60. These routines are streamlined by handling their calculations in hex, thus eliminating a requirement for repetitive 'Decimal Adjust' instruction usage. Hours increment from 1:00 to 10:00 is also handled the same way.

AM/PM Indication is handled simply by swapping the codes for the PM Indicator 'box' and 'space' each time the clock has incremented to 12:00 (SHEET 5; LINES 1 THROUGH 8).

Clock preset is initiated by setting the AM/PM indicator code in subroutine "AORPM" (SHEET 6). Here, the HEX KEY code depressed by the operator in response to the "A OR P =" prompt, is compared with H'ØA'. If the compare is valid, it's AM, so a spacecode is inserted into the PM Indicator location in the clock display message. If there is no valid compare, the PM indicator code (H'95') is inserted into the same location.

ROUTINE AS SHOWN	START ADDR XXX	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM
DESCRIPTION CONTINUATION OF THE PRECISE SEQUENCE REQUIRED TO SET THE TIME IN ORDER: HOURS; MINUTES; AND SECS. 2-DIGIT EACH.			SHEET 8

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1								TIME TO PRESET THE CLOCK.
2	Ø19Ø	20			SETIME	EORZ	R0	SO INIT. THE INDEX FOR
3	1	C8	2A			STRR,R0	MESDX	STORAGE IN THE CLOCK DSPLY.
4	3	06	DF			LODI,R2	H'DF'	ALSO A 'FETCH INDEX' FOR
5	5	CA	25			STRR,R2	>SELMX	HRS, MINS, SECS. THEN INI-
6	7	05	03			LODI,R1	3	TIAL THE # OF ENTRIES (3)
7	9	C9	20		NEXT	STRR,R1	EVENTS	THIS IS A LOOP PRGRM !!!!!
8	B	0A	1F			LODR,R2	>SELMX	ON ENTRY, STORE UPDATED E-
9	D	86	08			ADDI,R2	8	VENTS, &INCR. THE SRC MSG
10	9F	CA	1B			STRR,R2	>SELMX	STORE IT AWAY, THEN GO SET
11	A1	3F	01	80		BSTA,UN	NOWSET	THE TIME. ON RETURN, GET
12	4	0B	17		STRTIME	LODR,R3	MESDX	THE DEST. INDX, & STORE THE
13	6	CF	61	00		STRA,R0	DISCLK,MESDX	CURRENT TIME (HRS,M, OR SEC
14	9	01				LODZ	R1	LS DIGIT IN R1 TO R0, AND
15	A	CF	21	00		STRA,R0	DISCLK,MESDX,+	STORE IT. NOW UPDATE THE
16	D	87	02			ADDI,R3	2	CLOCK MSG INDEX, & STORE
17	AF	CB	OC			STRR,R3	MESDX	IT UNTIL NEEDED.
18	1B1	09	08			LODR,R1	EVENTS	FETCH EVENTS. HAVE HRS,
19	3	F9	64			BDRR,R1	NEXT	MINUTES,& SECS BEEN KEYED?
20	1B5	17				RETC,UN		NO! FINISH UP. YES!! EXIT.
21								
22	1BB	XX			EVENTS	RES	1	ENTRY OF H,M,S; (3 MAX)
23	BC	XX			>SELMX	RES	1	H,M, &S ENTER MSG INDEX
24	BD	XX			MESDX	RES	1	DISPLAY CLOCK INDEX

THE PROGRAM (CONTINUED)

Routine "SETIME" is the control program required for hours, minutes, and seconds preset sequence (SHEET 8).An EVENT COUNTER (R1) is setup in line 6. Count is 3, there being 3 major sequences for data entry (hours, minutes, and seconds. Indexed message control (in 8-byte increments) is maintained to provide display prompts to the operator for time entry.

Actual time value entry is handled by routine "NOWSET"(SHEET 7). "NOWSET" calls Monitor subroutine "GNPA" in order to permit the operator to enter the time from the HEX KEYBOARD after the appropriate prompt message is placed in the Display Buffer of the Monitor. After the numbers are selected by the operator, they are split into contiguous nibbles. The most significant nibble is checked for a zero value, in which case subroutine "BLNKZ" insets a space code into the M.S. HOURS position of the clock display message.

Routine "ORDER" controls the display time (1½ seconds) of the message "SET CLOC", viewed once when the operator depresses the RST key. "ORDER" is located in sheet 9 of the listing. A block of four 8-byte messages is found in sheet 10.

This concludes the explanation of program "DESCLK".

ROUTINE AS SHOWN	START ADDR XXX	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM
DESCRIPTION CONTROL ROUTINE FOR DISPLAY OF CLOCK PRESET ENTRY COMMANDS (HOURS; MINUTES, AND SECS)			SHEET 9

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø1BE	XX			RUNDSPX	RES	1	ORDER RUN TIME CONSTANT.
2	1BF	XX			DSPDLY	RES	1	ORDER DISPLAY DELAY
3								
4	1CØ	05	03		ORDER	LODI,R1	3	ON ENTRY, SET UP THE CONSTS
5	2	07	7F			LODI,R3	H'7F'	FOR 2-SEC DISPLAY OF THE
6	4	C9	78		REPEAT	STRR,R1	RUNDSPX	MESSAGE: 'SET CLOCK'
7	6	CB	77		DSPAGN	STRR,R3	DSPDLY	STORE THE CONSTANTS, AND
8	8	05	01			LODI,R1	<SETCLK-1	INIT THE 'SETCLOC' MSG INDX
9	A	06	DF			LODI,R2	>SETCLK-1	AND DISPLAY IT.
10	C	BB	E6			ZBSR	*USRDSP	ON RETURN, SEE IF THE DIS-
11	CE	0B	6F			LODR,R3	DSPDLY	PLAY DELAY IS COMPLETE.
12	1DØ	E7	01			COMI,R3	1	IF IT IS, RE-INIT. IF NOT,
13	2	18	02			BCTR,EQ	CONTDSP	GO DISPLAY AGAIN
14	4	FB	70			BDRR,R3	DSPAGN	
15	6	09	66		CONTDSP	LODR,R1	RUNDSPX	IS DISPLAY RUN TIME OVER?
16	8	F9	01			BDRR,R1	$+3	NO!! GO REPEAT IT. YES!!!
17	A	17				RETC,UN		GO SET THE TIME OF DAY.
18	B	07	7F			LODI,R3	H'7F'	TO REPEAT, RE-INIT DISPLY
19	Ø1DD	1B	65			BCTR,UN	REPEAT	DELAY & LOOP.
20								
21								THE MESSAGE TABLE CONTAIN'G
22								THE CLOCK PRESET COMMANDS
23								FOLLOWS ON THE NEXT SHEET
24								OF THE LISTING.

ROUTINE SELMES	START ADDR Ø1EØ	PART OF PROGRAM DESCLK	2650 PROGRAMMING FORM
DESCRIPTION MESSAGES FOR CLOCK PRESET COMMANDS			SHEET 10

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1								
2	Ø1EØ	05	0E	07	SELMES	RES	32	MESSAGE: 'SET CLOC'
3	3	17	0C	11				ACCESS FROM ROUTINE 'ORDER'
4	6	00	0C					ONLY.
5	E8	14	00	12				MESSAGE: 'HOURS'
6	B	13	05	17				ACCESS ON EVENT 3 (1ST)
7	E	16	17					
8	1FØ	0F	13	0A				MESSAGE: 'FRACT' FRACTION
9	3	0C	07	97				OF HOURS = MINUTES.
10	6	16	17					ACCESS ON EVENT 2 (2ND)
11	8	05	0E	0C				MESSAGE: SECS'
12	B	05	17	16				ACCESS ON EVENT 1 (3RD)
13	Ø1FE	17	17					
14								

********** End of Program DESKCLOCK; original version **********

"TIC-TOC" SIMULATION PROGRAM MODIFICATIONS:

The following program changes and additions cause the I/O LEDs to pulse on and off once per second. This provides further indication that "DESKCLOCK" is running. The clock's accuracy is maintained.

ROUTINE TICTOC	START ADDR 114	PART OF PROGRAM DESKCLK MOD	1 2650 PROGRAMMING FORM
DESCRIPTION THIS MODIFICATION SWITCHES THE I/O LEDS ON & OFF ONCE PER SE-COND TO SIMULATE A ''TICKING'' CLOCK			SHEET 11

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1					*MAKE THE FOLLOWING PROGRAM CHANGES AS SHOWN. **********			
2	ØØØ7	1B	28			BCTR,UN	CLK-3	NEW INTPT ADDRESS.
3	ØØ36	3F	01	14	CLK-3	BSTA,UN	TICTOC	JUMP TO TICTOC TEST ROUTINE
4	Ø14Ø	3F	01	08	AORPM	BSTA,UN	INITIO	PATCH TO INIT I/O LEDS.
5	Ø1ØB	77	10		INITIO	PPSL	RS	ON ENTRY, SET H'FF' LED
6	A	05	FF			LODI,R1	H'FF'	DRIVER PATTERN INTO R1'
7	C	75	10			CPSL	RS	RESELECT NON' REGS, THEN
8	1ØE	1B	01			BCTR,UN	CINITIO	BYPASS ASSIGNED FREQCT LOCN
9								
10	111	1F	01	C0	CINITIO	BCTA,UN	ORDER	..AND GO PRESET TIME OF DAY
11								
12	114	77	10		TICTOC	PPSL	RS	ON ENTRY, RESELECT REGS',&
13	6	20				EORZ	R0	SEE IF FREQCT IS INITIALI-
14	7	EC	01	10		COMA,R0	FREQCT	ZED.
15	A	98	05			BCFR,EQ	TEST2	YES: WRTE LED DRIVER TO LEDS
16	C	D5	07			WRTE,R1	PORT7	THEN RESELECT NON' REGS &
17	11E	75	10		RETN	CPSL	RS	RETURN TO CLK ROUTINE
18	120	17				RETC,UN		NO: WELL SEE IF THE FREQCT
19	1	04	28		TEST2	LODI,R0	H'28'	IS EQUAL TO H'28' YET.
20	3	EC	01	10		COMA,R0	FREQCT	YES: CLEAR THE I/O LEDS,
21	6	98	76			BCFR,EQ	RETN	SINCE THE LAST PART OF A SEC
22	8	20				EORZ	R0	IS GOING ON.
23	9	D4	07			WRTE,R0	PORT7	NO, GO RETURN TO CLK ROU-
24	011B	1B	71			BCTR,UN	RETN	TINE.

PROGRAMMABLE STOPWATCH

INTRODUCTION:

The INSTRUCTOR 50's REAL TIME interrupt facilities are featured in the program for an extremely accurate STOPWATCH application. This is no ordinary stopwatch, however; you have the capability to "freeze" up to 4 different times while "STOPWATCH" is running, all without affecting its accuracy. In addition, you may "review" each frozen time, simply by depressing the appropriate hex key. Again, the internal clock 'mechanism' keeps running at designed accuracy.

Practical use of this "STOPWATCH" can take a myriad of forms. Think of its application at track and swim meets, where precise lap times must be determined by the coaches and participants, both to improve their performance, and in formal competition. The "STOPWATCH"'s 1/10th second digital readout will literally stop arguments over true time in all but the fastest races. At the horse or dog races, you may have the need to freeze times at the quarter mile, half pole, and at the head of the stretch. As an aid to your timing televised events, such as the Kentucky Derby, or the Olympics, its unbeatable!

In its original form, "STOPWATCH"'s "running clock" times from zero to 1:59::59.9 in tenth of a second increments. For easy readability, an "odd hour" indicator is blanked until timeout of the first hour (0:59::59.9) is completed. At 1:00::00.0, the

ROUTINE AS SHOWN	START ADDR ØØØØ	PART OF PROGRAM STOPWATCH	2650 PROGRAMMING FORM
DESCRIPTION 'STOPWATCH' EXTENDS THE CAPABILITY OF THE I-50 TO PERFORM AS A TIMER ACCURATE TO 1/10 SEC OVER 2 HOURS. CAN FREEZE & DISPL. 4 TIMES			SHEET 1

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØØØ	76	20		STPWTCH	PPSU	II	TURN ON INTPT INHIBIT, &
2	2	1F	00	90		BCTA,UN	INITIAL	GO INIT'ZE CLK MESSG. ON
3	5	00	19		MAINPRO	ACON	H'0019'	RETURN, JUMP TO MAIN PROG'M
4	7	00	B8		CLOCK	ACON	H'00B8'	INDIRECT ADDRESS TABLE:
5	9	01	28		INCTENM	ACON	H'0128'	CLOCK ENTRY ON R.T. INPT.
6	B	01	45		RUNCLKD	ACON	H'0145'	INCTENM CONT. CLK INCREM
7	D	01	4E		STACLKD	ACON	H'014E'	RUNN'G & STATIC CLK DSPLAYS
8	ØF	00	78		SETINDX	ACON	H'0078'	CONVERT KEY TO MESSG INDX.
9	11	00	70		WINDOW	ACON	H'0070'	ENBL INPT. AT SPEC. TIMES.
10	3	00	00	00	*FILLER	CODE		
11	6	00	00	00	*FILLER	CODE		AVAILABLE FOR EXPANSION:
12								
13	ØØ19	74	40		MAINPRO	CPSU	FLAG	AFTER INIT., CLR THE FLAG
14	B	20				EORZ	R0	ALSO, ZERO THE PSL, AND THE
15	C	93				LPSL		FREQUENCY CTR. FOR THE UP-
16	1D	CC	00	B7		STRA,R0	FREQCT	COMING CLOCK. NOW, SHOW THE
17	2Ø	BB	8B		INITCLK	ZBSR	*RUNCLKD	RESET CLOCK. WHEN OPERATOR
18	2	B4	80			TPSU	SENS	DEPR. SENS KEY, TURN FLAG
19	4	98	7A			BCFR,EQ	INITCLK	ON, & OPEN THE REAL TIME
20	6	76	40			PPSU	FLAG	INTP. WINDOW FOR 6 USEC.
21	8	BB	91		START	ZBSR	*WINDOW	THEN DISABLE IT.
22	A	C0				NOP		
23	B	C0				NOP		
24	C	C0				NOP		ON RETURN, GO SEE IF THE

A JOHN GARCEAU PROGRAM

INTRODUCTION: (CONTINUED)

the "odd hour" indicator, a small box in the left-most display digit location, is turned on. It remains on until "STOPWATCH" cycles automatically into the next (even) hour.

Two variations (described later) provide you options to stop the "STOPWATCH". The first, programmed in the tape you've received, stops the clock when the AC Line/Keyboard Interrupt Slide switch is moved to the KEYBOARD position. The second option stops the watch if any key in the second column (from the right) in the Hex Keyboard, is depressed.

NOTES:

Set up and operating procedures start on the next page.

ROUTINE MAINPRO | START ADDR XXX | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION THE MAIN PROGRAM, INCLUDING TEST FOR TIME FREEZE OR RETTRIEVAL FOR DISPLAY, IS CONTINUED | SHEET 2

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ2D	04	08		MAYSTOR	LODI,R0	STORECOL	CURRENT TIME IS TO BE FRO-
2	2F	D4	FA			WRTE,R0	PORTFA	ZEN. OPR. TELLS US BY DEP-
3	31	54	FE			REDE,R0	PORTFE	RESS'G 1 OF 4 HEX KEYS:
4	3	F4	0F			TMI,R0	NOKEY	'F', 'B', '7', OR '3'. IF
5	5	B8	10			BSFR,EQ	FREEZE	SO, FREEZE THE TIME. IF NOT
6	7	04	01		RECALL	LODI,R0	RETRVCOL	SEE IF HE WANTS PREVIOUSLY
7	9	D4	FA			WRTE,R0	PORTFA	FROZEN TIME. HE'LL DEPRESS
8	B	54	FE			REDE,R0	PORTFE	'C', '8', '4', OR 'Ø' TO
9	D	F4	0F			TMI,R0	NOKEY	TELL US THAT. IF SO, GO DIS-
10	3F	98	15			BCFR,EQ	RETRIEVE	PLAY DESIREDTIME. IF NOT,
11	41	BB	8B			ZBSR	*RUNCLKD	SHOW THE RUNNING TIME, THEN
12	3	1B	63			BCTR,UN	START	LOOP TO SCAN FOR MORE TIME
13								FREEZE OR STATIC DISPLAY.
14					*END OF	MAIN PROGRAM;	SUBROUTINES	FOLLOW. **********
15								
16	45	BB	91		FREEZE	ZBSR	*WINDOW	ON ENTRY, GO INCR CLOCK IF
17	7	BB	8F			ZBSR	*SETINDX	NECESS., THEN XLATE KEYCODE
18	9	05	00			LODI,R1	Ø	INTO MESSG INDX. ON RETURN
19	B	0D	20	FF	MOVE	LODA,R0	RUNCLK-1,R1,+	MOVE THE CURRENT TIME MES-
20	4E	CE	21	00		STRA,R0	RUNCLK,R2,+	SAGE TO DESIRED FREEZE LO-
21	51	E5	08			COMI,R1	8	CATION, THEN RETURN TO MAIN
22	3	98	76			BCFR,EQ	MOVE	LINE SCAN
23	5	17				RETC,UN		
24								

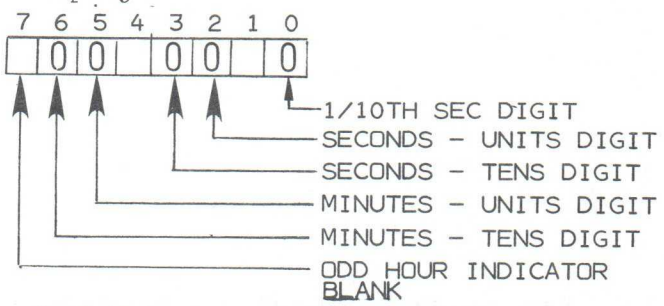
USER OPERATION - SET UP:

After loading program "STOPWATCH" into the INSTRUCTOR 50's memory from tape, position the control switches as follows:

AC Line / Keyboard Interrupt Selection Slide Switch:	AC LINE
Interrupt Address Select Sw:	INDIRECT
I/O Selection Switch:	EXTENDED I/O PORT Ø7
Parallel I/O input switches:	not used

REAL TIME USER OPERATION:

Depress RST .....*to initialize the STOPWATCH. The INSTRUCTOR 50 responds immediately with the display:*



The clock is not running. FLAG is off.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION INCLUDES SUBROUTINES RETRIEVE, WINDOW, AND SET INDEX AS DESCRIBED BELOW. | SHEET 3

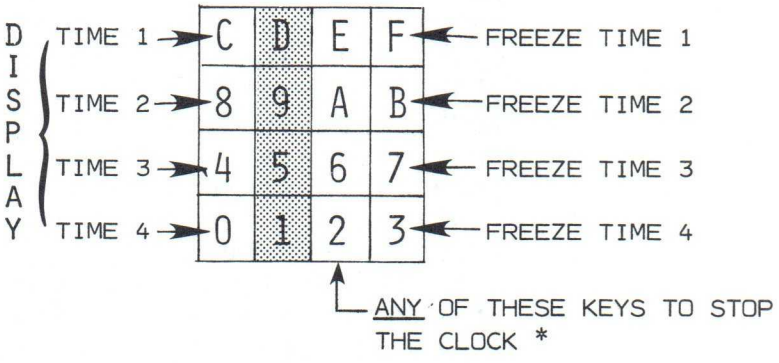
LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø056	BB	91		RETRIEVE	ZBSR	*WINDOW	ON ENTRY, INCREMENT RTCLK
2	8	BB	8F			ZBSR	*SETINDX	IF NECESS, THEN GO XLATE
3	A	CE	01	57		STRA,R2	CLK1234	THE DEPRESSED KEY TO THE
4	D	BB	8D			ZBSR	*STACLKD	DESIRED MESSAGE INDEX, NOW
5	5F	C0	C0			NOP		DISPLAY THE DESIRED FROZEN
6	61	1B	45			BCTR,UN	START	TIME ONCE, THEN RETURN TO
7								MAINLINE PROGRAM SCAN.
8	63-6F	00	00	00	*FILLER	CODE		
9								
10	Ø07Ø	74	20		WINDOW	CPSU	II	ON ENTRY, OPEN THE REALTIME
11	2	C0				NOP		INTERRUPT WINDOW FOR 6 USEC
12	3	76	20			PPSU	II	THEN SHUT IT, & RETURN
13	5	17				RETC,UN		
14								
15	76	00	00		*FILLER	CODE		
16								
17	Ø078	F4	07		SETINDX	TMI,R0	H'07'	SET UP THE INDEX FOR DESI-
18	A	18	0B			BCTR,EQ	INDXM1	RED STATIC TIME MESSAGE.
19	C	F4	0B			TMI,R0	H'0B'	EITHER FREEZE CURRENT TIME,
20	7E	18	0A			BCTR,EQ	INDXM2	OR FOR FUTURE DISPLAY OF
21	80	F4	0D			TMI,R0	H'0D'	PREV. FROZEN TIME
22	2	18	09			BCTR,EQ	INDXM3	STATIC TIME MESSG'S
23	4	06	1F		INDXM4	LODI,R2	H'1F'	INDEX TO >TIME -1 MESSAGE
24	86	17				RETC,UN		

REAL TIME USER OPERATION: (CONTINUED)

Depress SENS ..... *to start the STOPWATCH's run cycle. The FLAG LED is turned on, and the Hex Display continuously updates the current time from start.*

Using the handy chart at the right, depress appropriate HEX keys:

- to FREEZE desired times
- to display previously frozen times.
- to stop the clock*



* If this programmable option is chosen!!!!!

TO STOP THE CLOCK IN PLACE, WITH CAPABILITY TO READ "FROZEN" STATIC TIMES:

Slide the AC Line/Keyboard Interrupt Switch to the KEYBOARD position. THIS DIRECTION IS VALID FOR THE TAPED VERSION OF "STOPWATCH".

ROUTINE AS SHOWN	START ADDR XXX	PART OF PROGRAM STOPWATCH	2650 PROGRAMMING FORM
DESCRIPTION CONTIN. OF 'SETINDX' AND CLOCK INITIALIZATION ROUTINES:			SHEET 4

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	0087	06	07		INDXM1	LODI,R2	H'07'	INDEX TO TIME1-1
2	9	17				RETC,UN		
3	A	06	0F		INDXM2	LODI,R2	H'0F'	INDEX TO TIME2-1
4	C	17				RETC,UN		
5	D	06	17		INDXM3	LODI,R2	H'17'	INDEX TO TIME 3-1
6	8F	17				RETC,UN		
7								
8	90	20			INITIAL	EORZ	R0	MUST INITIALIZE STPWATCH
9	1	06	28			LODI,R2	40	READOUT TABLE & 4 STATIC
10	3	CE	41	00	CLEARIT	STRA,R0	RUNCLK,R2,-	TIME STORAGE MESSAGES, SO
11	6	5A	7B			BRNR,R2	CLEARIT	1ST CLEAR 40 BYTES FOR 5
12	8	07	05			LODI,R3	5	MESSAGES (8-BYTE WIDTH)
13	A	04	17			LODI,R0	SPACECODE	THEN PUT SPACE CODE IN EACH
14	C	3B	0A		INSERT	BSTR,UN	SPCD1	8-BYTE MESSAGE AS FOLLOWS:
15	9E	3B	08			BSTR,UN	SPCD1	'SP 0 0 SP 0 0 SP 0'
16	A0	3B	06			BSTR,UN	SPCD1	'RUNCLK' AT ADDR 100
17	2	A6	01			SUBI,R2	1	'TIME1' AT ADDR 108
18	4	FB	76			BDRR,R3	INSERT	'TIME2' AT ADDR 110
19	6	9B	85			ZBRR	*MAINPRO	'TIME3' AT ADDR 118
20								'TIME4' AT ADDR 120.
21	A8	CE	61	00	SPCD1	STRA,R0	RUNCLK,R2	ALGORITHM IS SPACE INSERT
22	B	86	03			ADDI,R2	3	THEN ADD 3; REPEAT TWICE
23	AD	17				RETC,UN		MORE, BUT SUBTRACT 1 TO
24	AE-B6	00	00	00	*FILLER CODE			JUSTIFY BEFORE NEXT INSERT.

THE PROGRAM:

In addition to the comments, provided in the listing, the following considerations are of interest:

"INITIAL" (REFER TO THE LISTING - SHEET 4)

The problem was to provide a data format of 8 bytes for each of 5 messages. This format involves zeroing bytes 6,5,3,2, & 0, and inserting space codes into bytes 7,4, and 1 of each message, prior to the time SENS is depressed to start the clock. The 40-byte memory block is first cleared, then routine "INSERT" provides an efficient mechanism to introduce space codes at desired locations. The key is provided in implementing the short subroutine "SPCD1" ((insert) one space code, then add 3 to the insertion index).This is executed 3 times. The resultant index is 1 higher than it should be after the 3rd "SPCD1" execution. To justify it to the next 8-byte message, the SUBtract instruction (LINE 17) is implemented.

R3, initialized to '05' in line 12, keeps track of the number of messages to be initialized.

"CLOCK" (REFER TO SHEETS 5, 6, AND 8 IN THE LISTING)

The choice of a precise location for the start address of this routine (at location ØØB8) was deliberate. On

ROUTINE CLOCK | START ADDR ØØB8 | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION ENTERED EACH TIME THERE IS A 60 (50) HZ AC LINE INTERRUPT, THIS ROUTINE INCREMENTS TENTHS OF SECS, SECS, MINUTES & HR. INDIC. | SHEET 5

LINE	ADDRS.	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØB7	XX			FREQCT	RES	1	FREQUENCY CT. VAR.:TEMP STR
2								
3	ØØB8	09	7D		CLOCK	LODR,R1	FREQCT	ON ENTRY, FETCH CURRENT LINE
4	A	85	01			ADDI,R1	1	FREQU. COUNT; INCR. IT &
5	C	E5	06			COMI,R1	6	SEE IF A 1/10TH SEC TIME:
6					*FOR 50 HZ, CHANGE CONTENTS OF ADDR 'BD' TO '05'.			
7	BE	18	03			BCTR,EQ	TENTHS	YES: GO INCR. TENTHS OF SEC
8	CØ	C9	75			STRR,R1	FREQCT	NO; STR CURR. FREQCT, AND
9	2	17				RETC,UN		RETURN TO MAINLINE PROGRAM
10								SCAN.
11	C3	20			TENTHS	EORZ	R0	ON ENTRY, ZERO FREQCT, THEN
12	4	C8	71			STRR,R0	FREQCT	INCREMENT TENTHS OF SECONDS
13	6	08	3F			LODR,R0	FRACSEC	BY 1. 10 TENTHS COUNTED
14	8	84	01			ADDI,R0	1	YET? NO: STORE TENTHS IN
15	A	E4	0A			COMI,R0	10	CURRENT TIME MESSG, & RE-
16	C	18	03			BCTR,EQ	INCUSEC	TURN. YES: GO INCREMENT
17	CE	C8	37			STRR,R0	FRACSEC	UNITS DIGIT OF SECONDS
18	DØ	17				RETC,UN		
19								
20	D1	20			INCUSEC	EORZ	R0	SAME FLOW AS ABOVE, SUBSTI-
21	2	C8	33			STRR,R0	FRACSEC	TUTING TENTHS FOR FREQUEN-
22	4	08	2F			LODR,R0	UNITSEC	CY COUNT, AND UNIT SECONDS
23	6	84	01			ADDI,R0	1	FOR TENTHS
24	8	E4	0A			COMI,R0	10	MODULO 10 COUNTER

THE PROGRAM: (CONTINUED)

Sheet 5; line 13, the LODR instruction uses a maximum relative displacement to access FRACSEC, the 1/10th second location in the running clock message. This displacement is positive with a magnitude of 63 (H'3F) bytes.

The running clock message "RUNCLK (at locations H'100' to '107') is bracketed by routine "CLOCK. On sheet 8 at line 14, the final store in RUNCLK message is accomplished by the STRR instruction. Here, a maximum NEGATIVE relative displacement (-64 bytes (H'40')) is programmed. By using relative-addressed instructions rather than absolute-addressing, considerable memory was saved. You can make a comparison of program lengths required to perform the SAME TASK, by inspection of program "DESKCLOCK"'s "CLK" routine. You'll see that this routine uses about 50 more bytes of storage.

"RUNCLKD"
"STACLKD" (REFER TO SHEETS 8 & 9)

At first glance, it does not appear to be useful to have 2 display routines; one for running clock, and 1 for static clock display. Both are "single pass" display routines, followed by RETC instructions. However, inspection shows that the LEAST SIGNIFICANT ADDRESS CONSTANT loaded into R2 is prepared differently in each routine. In "RUNCLKD" the constant is always H'FF'; this is why the LODImmediate instruction is used. IN "STACLKD", the Least sig. address constant, defining one of 4 possible time messages to be displayed,

ROUTINE CLOCK | START ADDR XXX | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF RUNNING CLOK MESSAGE INCREMENTATION IN REAL TIME. | SHEET 6

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØDA	18	03			BCTR,EQ	INCUTENS	
2	C	C8	27			STRR,R0	UNITSEC	
3	DE	17				RETC,UN		
4								
5	DF	20			INCUTENS	EORZ	R0	SAME FLOW AS DESCRIBED
6	EØ	C8	23			STRR,R0	UNITSEC	PREVIOUSLY. THIS TIME TO
7	2	08	20			LODR,R0	TENSEC	INCREMENT TENS OF SECONDS
8	4	84	01			ADDI,R0	1	
9	6	E4	06			COMI,R0	6	
10	8	18	03			BCTR,EQ	INCUMINS	
11	A	C8	18			STRR,R0	TENSEC	
12	C	17				RETC,UN		
13								
14	ED	20			INCUMINS	EORZ	R0	SAME FLOW: THIS TIME TO
15	EE	C8	14			STRR,R0	TENSEC	INCREMENT MINUTES UNIT
16	FØ	08	10			LODR,R0	UNITMIN	DIGIT.
17	2	84	01			ADDI,R0	1	
18	4	E4	OA			COMI,R0	10	
19	6	18	03			BCTR,EQ	SETUMØ	
20	8	C8	08			STRR,R0	UNITMIN	
21	A	17				RETC,UN		
22	B	20			SETUMØ	EORZ	R0	THIS PROCESS ZEROS UNIT
23	C	C8	04			STRR,R0	UNITMIN	DIGIT OF MINUTES, THEN
24	ØØFE	9B	89			ZBRR	*INCTENM	BRANCH TO INCR 10S OF MINS

THE PROGRAM: (CONTINUED)

is a VARIABLE. This variable,"CLK1234", is prepared by subroutine "SETINDX" just before "STACLKD" is executed. The LODRelative instruction (SHEET 9, LINE 3) fetches "CLK1234" into the display routine.

"SETINDX" (REFER TO SHEETS 3 & 4)

This subroutine is executed if the mainline program determines that the operator has depressed some Hex key to "freeze" or display a static time. The pattern read in from EXTENDED I/O PORT FE (KEY SENSE) is checked for binary patterns as shown at the right. If the desired pattern matches, an INDEX to the corresponding message is loaded into R1. Upon return from "SETINDX", R1's contents are :

- stored in "CLK1234" if this is a retrieve operation, or
- used as an index to provide a destination address for each of 8-bytes current clock in the desired "freeze" clock message.

H'0F'	● ● ● ●	NO KEY CODE PATTERN(1)
H'07'	0 ● ● ●	TIME1 'F' DEPRESSED
H'0B'	● 0 ● ●	TIME2 'B' DEPRESSED
H'0D'	● ● 0 ●	TIME3 '7' DEPRESSED
H'0E'	● ● ● 0	TIME4 '3' DEPRESSED(2)

NOTE 1: IF TRUE, ''SETINDX'' IS NOT ACCESSED

NOTE 2: THIS IS DEFAULT IF NONE OF THE OTHER TESTED PATTERNS HAVE A VALID DECODE. REFER TO LINES 22-24; SHEET 3

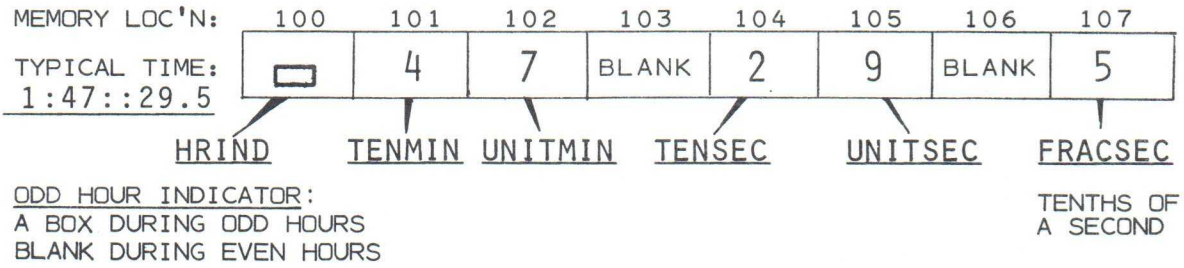
ROUTINE RUNCLK | START ADDR Ø1ØØ | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION 'RUNCLK' IS 5 ENTRY X 8-BYTE DATA TABLE CONTAINING THE RUN-CLOCK DISPLAY MESSAGE, ALSO 4 STATIC 'FREEZE' CLOCK MESSAGES | SHEET 7

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1					*ALTHOUGH NO CODE NEED BE ENTERED			IN THIS 40-BYTE TABLE BY
2					*THE USER DURING MEMORY LOAD OPERATIONS, THE CODE SHOWN DES-			
3					*CRIBES THE FORMAT FOR EACH TIME MESSAGE:			
4					*			
5					*		XX = DATA CHANGED BY INCR. CLOK.	
6					*REFER TO ROUTINE 'INITIAL' FOR SETUP BEFORE START.			
7	Ø1ØØ	XX	XX	XX	RUNCLK	ACON	H'0100'	RUNNING CLOCK MESSAGE LOC.
8		17	XX	XX	RUNCLK	RES	8	
9		17	XX					
10	8	XX	XX	XX	TIME1	ACON	H'0108'	STATIC TIME 1 MESSAGE LOC.
11		17	XX	XX	TIME1	RES	8	
12		17	XX					
13	10	XX	XX	XX	TIME2	ACON	H'0110'	STATIC TIME 2 MESSAGE LOC.
14		17	XX	XX	TIME2	RES	8	
15		17	XX					
16	18	XX	XX	XX	TIME3	ACON	H'0118'	STATIC TIME 3 MESSAGE LOC.
17		17	XX	XX	TIME3	ACON	8	
18		17	XX					
19	20	XX	XX	XX	TIME4	ACON	H'0120'	STATIC TIME 4 MESSAGE LOC.
20		17	XX	XX	TIME4	RES	8	
21	0126	17	XX					
22					*LAST ADDRESS OF MESSAGE DATA TABLE IS H'0127'			
23								
24					*CLOCK ROUTINE CONTINUES ON THE NEXT SHEET OF THE LISTING...			

THE PROGRAM: (CONTINUED)

Referring once again to subroutine "CLOCK" (sheets 5,6, & 8), there are several symbolic terms related to access of data from, or storage of data into, "RUNCLK", the running time message. The following diagram identifies the location and meaning of each of these terms:



This concludes the discussion of the program.

NOTES:

ROUTINE AS SHOWN | START ADDR Ø128 | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION CONTIN OF CLOCK INCREM; ALOS ODD HOUR INDIC SET, AND DISPLAY ROUTINES FOR RUNNING & STATIC TIMES. | SHEET 8

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø128	08	57		INCTENM	LODR,R0	TENMIN	MODULO 6 COUNT FOR TENS OF
2	A	84	01			ADDI,R0	1	MINUTES TEST; SAME FLOW
3	C	E4	06			COMI,R0	6	AS PREVIOUSLY DESCRIBED
4	2E	18	03			BCTR,EQ	SETODHR	
5	3Ø	C8	4F			STRR,R0	TENMIN	
6	2	17				RETC,UN		
7								
8	33	20			SETODHR	EORZ	R0	FIRST, ZERO TENS OF MINUTES
9	4	C8	4B			STRR,R0	TENMIN	THEN SEE IF ENDING HOUR IS
10	6	08	48			LODR,R0	HRIND	ODD. IF SO, GO BLANK THE
11	8	E4	95			COMI,R0	ODDBOX	ODD HOUR INDICATOR. IF NOT,
12	A	18	05			BCTR,EQ	SPACESET	SET A 'BOX' CODE ('95')
13	C	04	95			LODI,R0	ODDBOX	INTO THE HOUR INDICATOR
14	3E	C8	40		HOURSET	STRR,R0	HRIND	LOC'N OF RUNNING TIME MES-
15	40	17				RETC,UN		SAGE & EXIT.
16								
17	141	04	17		SPACESET	LODI,R0	SPACECODE	HOUR INDICATOR SPACECODE
18	3	1B	79			BCTR,UN	HOURSET	SETUP. THIS IS END OF REAL
19								TIME INPT. ROUTINE.
20	145	05	00		RUNCLKD	LODI,R1	<RUNCLK-1	ON ENTRY, SET UP CONSTANTS
21	7	06	FF			LODI,R2	>RUNCLK-1	FOR RUNNING CLOCK ACCESS
22	9	07	01			LODI,R3	ONEPASS	AND GO MAKE A SINGLE PASS
23	014B	BB	E6			ZBSR	*USRDSP	OF THE DISPLAY, THEN EXIT
24	D	17				RETC,UN		

NOTES:

Referring to the listing (sheet 9 - below), alter memory as shown BELOW line 10 to provide the operator with the capability to stop the clock by depressing 'E', 'A', '6', or '2' keys.

ROUTINE STACLKD | START ADDR Ø14E | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION STATIC MESSAGE DISPLAY ON CALL BY USER KEY DEPRESSION OF 'C', '8', '4', OR '0' HEX KEYS DURING CLOCK RUN. | SHEET 9

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø14E	05	01		STACLKD	LODI,R1	<RUNCLK	ON ENTRY, SET UP CONSTANTS
2	15Ø	0A	05			LODR,R2	CLK1234	FOR SINGLE PASS OF SELEC-
3	2	07	01			LODI,R3	ONEPASS	TED STATIC TIME MESSAGE,
4	4	BB	E6			ZBSR	*USRDSP	THEN EXIT AFTER DISPLAY.
5	6	17				RETC,UN		
6								
7	0157	XX			CLK1234	RES	1	COMPUTED INDEX FOR SELECTED
8					*END OF	PROGRAM	'STOPWATCH'**	TIME MESSAGE.
9	**********				**********	**********	**********	**********
10	**********				MODIFICATION TO STOP CLOCK BY DEPRESSING A HEX KEY !!!!!!			
11					*MAKE THE FOLLOWING CHANGES & ADDITIONS:			
12	ØØ13	01	60		STOP	ACON	H'0160'	IND. ADDR TO STOP CLK ROUT.
13	ØØ43	9B	93			ZBRR	*STOP	SCAN TO SEE IF STOP NOW.
14	Ø16Ø	08	04		STOP	LODI,R0	STOPCOL	*ON ENTRY, GO SEE IF THE
15	2	D4	FA			WRTE,R0	PORTFA	OPERATOR HAS DEPRESSED ANY
16	4	54	FE			REDE,R0	PORTFE	KEY IN THE ''E*A*6*2'' COL-
17	6	F4	0F			TMI,R0	NOKEY	UMN. IF NOT, REPEAT MAINLINE
18	8	98	02			BCFR,EQ	STOPTIME	SCAN. IF SO, STOP AND DIS-
19	A	1B	88			BCTR,UN	*START1	PLAY THE RUNNING CLOCK
20	C	74	40		STOPTIME	CPSU	FLAG	MESSAGE IN PLACE UNTIL THE
21	16E	76	20			PPSU	II	OPERATOR DEPRESSES RST.
22	17Ø	BB	8B			ZBSR	*RUNCLKD	FLAG LED IS TURNED OFF, &
23	2	1B	7C			BCTR,UN	$-2	ALL INTERRUPTS ARE BLOCK-
24	4	00	28		START1	ACON	START	ED.

CLOCK READOUT ACCURACY REFINEMENT -60 HZ

Minor modifications to program "STOPWATCH" make it possible for you to interpret a combined output of the Hex Display and I/O LEDs. This feature extends the readout accuracy of "STOPWATCH" to 1/60th (0.01667) seconds. Just program the changes to the program as shown in the listing - sheet 10; below. Operate the program as specified previously. Then, when you stop the clock, use the conversion chart - below - to interpret the result.

A 50 Hz version of this feature is provided in the listing; sheet 11 on the next page.

I/O LEDS** (7 6 5 4 3 2 1 Ø)	TENTH	TIME*	TENTH	TIME*	TENTH	TIME*	TENTH	TIME*	TENTH	TIME*
0 0 0 0 0 0 0 0	0	0.00000	2	0.20000	4	0.40000	6	0.60000	8	0.80000
0 0 0 0 0 0 0 ●		.01667		.21667		.41667		.61667		.81667
0 0 0 0 0 0 ● ●		.03333		.23333		.43333		.63333		.83333
0 0 0 0 0 ● ● ●		.05		.25		.45		.65		.85
0 0 0 0 ● ● ● ●		.06667		.26667		.46667		.66667		.86667
0 0 0 ● ● ● ● 0		.08333		.28333		.48333		.68333		.88333
0 0 ● ● ● ● 0 0	1	0.1	3	0.3	5	0.5	7	0.7	9	0.9
0 ● ● ● ● 0 0 0		.11667		.31667		.51667		.71667		.91667
● ● ● ● 0 0 0 0		.13333		.33333		.53333		.73333		.93333
● ● ● 0 0 0 0 0		.15		.35		.55		.75		.95
● ● 0 0 0 0 0 0		.16667		.36667		.56667		.76667		.96667
● 0 0 0 0 0 0 0		0.18333		0.38333		0.58333		0.78333		0.98333

**● = ON
0 = OFF

*IN SECONDS

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION THIS MODIFICATION PERMITS DIRECT INTERPRETATION OF THE DISPLAY & I/O LEDS FOR 1/60TH OF A SECOND ACCURACY USING THE CONV. TABLE | SHEET 10

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1					*MODIFY	OR MAKE	PROGRAM ADDITIONS	AS FOLLOWS: **********
2	ØØØ7	01	90		SHIFTIO	ACON	H'0190'	NEW INTPT INDIR. ADDRESS
3								
4	ØØ9Ø	3F	01	80	INITIAL	BCTA,UN	IOPREP	BRANCH TO IOPREP ROUTINE
5								
6	Ø18Ø	75	01		IOPREP	CPSL	CARRY	ON ENTRY, CLR CARRY, &
7	2	77	10			PPSL	RS	SELECT PRIME REG BANK.
8	4	20				EORZ	R0	CLEAR R0, ALSO ZERO R1, THE
9	5	C1				STRZ	R1	I/O WRT REG, AND INIT
10	6	06	F0			LODI,R2	H'F0'	R2 (PATTERN GEN). FOR 60 HZ
11	8	D5	07			WRTE,R1	PORT07	USE H'F0'. NOW, BLANK THE
12	A	75	10			CPSL	RS	IO LEDS FOR START, THEN RE-
13	C	06	28			LODI,R2	40	SELECT NON' REGS, AND INIT.
14	E	17				RETC,UN		R2 AS MEMORY BYTE CTR. AND
15	18F	00			*FILLER	CODE		RETURN TO CLR A MEM BLOCK.
16								
17	Ø19Ø	77	18		SHIFTIO	PPSL	RS,WC	60 HZ LINE INTPT, SO SELECT
18	2	D2				RRL	R2 .	REGS' & SET FOR CARRY. NOW
19	3	D1				RRL	R1	SHIFT PATTRN GEN, & IO WRT
20	4	D5	07			WRTE,R1	PORT07	REG, AND SHOW ON THE LEDS.
21	6	59	02			BRNR,R1	$+4	WRT REG EMPTY? YES: REINIT
22	8	06	F0			LODI,R2	H'F0'	PATTRN GEN (60HZ = H'F0')
23	A	75	18			CPSL	WC,RS	NO: SKIP ABOVE INST. RESEL.
24	19C	1F	00	B8		BCTA,UN	CLOCK	NON' REGS AND GO CLOCK.

50 HZ CLOCK READOUT ACCURACY REFINEMENT:

This modification is designed for use by those whose INSTRUCTOR 50s are operating in a 50 Hz environment. In addition to the instructions provided in the listing (sheet 11 - below), ensure that memory location '00BD' contains H'05'.

After stopping the clock, interpret the results by comparison with the conversion chart provided below.

I/O LEDS** (7 6 5 4 3 2 1 Ø)	TENTH	TIME*	TENTH	TIME*	TENTH	TIME*	TENTH	TIME*	TENTH	TIME
● 0 0 0 0 0 0 0		0.00		0.20		0.40		0.60		0.80
0 0 0 0 0 0 0 ●		0.02		0.22		0.42		0.62		0.82
0 0 0 0 0 0 ● ●	0	0.04	2	0.24	4	0.44	6	0.64	8	0.84
0 0 0 0 0 ● ● ●		0.06		0.26		0.46		0.66		0.86
0 0 0 0 ● ● ● 0		0.08		0.28		0.48		0.68		0.88
0 0 0 ● ● ● 0 0		0.10		0.30		0.50		0.70		0.90
0 0 ● ● ● 0 0 0		0.12		0.32		0.52		0.72		0.92
0 ● ● ● 0 0 0 0	1	0.14	3	0.34	5	0.54	7	0.74	9	0.94
● ● ● 0 0 0 0 0		0.16		0.36		0.56		0.76		0.96
● ● 0 0 0 0 0 0		0.18		0.38		0.58		0.78		0.98
7 6 5 4 3 2 1 Ø										

* TIME IN SECONDS. ** ● = ON; 0 = OFF.

NOTE: BEFORE SENS IS DEPRESSED TO START CLOCK, ALL LEDS ARE OFF.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM STOPWATCH | 2650 PROGRAMMING FORM

DESCRIPTION DISPLAY AND I/O LED INTERPRETATION FOR 1/50TH (0.02) SECOND ACCURACY - USE WITH LINE INPUT = 50 HZ. | SHEET 11

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1					*PROGRAM	MODIFICATIONS SHOULD REFLECT THOSE PROVIDED ON		
2					*SHEET 10,	EXCEPT AS PROVIDED BELOW.**********		
3					*			
4	Ø186	06	E0			LODI,R2	H'E0'	50 HZ INTERP. FACTOR IS
5					*			H'E0'.
6					*PROGRAM	(LOC'NS 0190-0195) IS UNCHANGED		
7					*MODIFICATION	OF SUBROUTINE ''SHIFTIO''...: **********		
8	Ø196	E5	80			COMI,R1	H'80'	AFTER 10TH LINE INTP (2/10
9	8	18	05			BCTR,EQ	RSTI/O	SEC), IOLED # 7 ON ONLY.
10	A	75	18			CPSL	RS,WC	IF SO, MUST REINIT PATTRN
11	C	1F	00	B8		BCTA,UN	CLOCK	GEN & I/O WRT REG. IF NOT
12	19F	05	00		RSTI/O	LODI,R1	0	RESELECT NON' REGS AND GO
13	1A1	06	E0			LODI,R2	H'E0'	EXECUTE CLOCK RTN.
14	1A3	1B	75			BCTR,UN	$-11	TO REINIT IO LED DRIVERS,
15								CLEAR I/O WRT REG, AND SET
16								H'E0' INTO PATTERN GEN,
17								THEN GO BACK TO RESELECT
18								NON' REGS; ETC.
19					**********	END OF MODIFICATION**********		
20								
21								
22								
23								
24								

ELECTRONIC CRAPGAME

INTRODUCTION:

The flexible facilities of the INSTRUCTOR 50 are dedicated this time to a highly interactive "CRAPGAME" application, one in which you and you friends will enjoy time and again.

CRAPGAME operates under a modified set of Nevada casino "Pass Line" rules. After the player 'buys into the game' and places his first 'bet', the 'dice are rolled'. The 'roll' is terminated by someone depressing the SENS key.

At this point, the INSTRUCTOR 50 determines whether this is the first dice roll (PASS 1), or a roll following the first pass. Based on this condition, messages indicate whether the player won, lost, or must roll again. The following table indicates Nevada Casino Pass Line Play:

ROLL OF DICE :	2	3	4	5	6	7	8	9	10	11	12
FIRST PASS :	CRAP OUT LOSES BET		SETS POINT ROLLS AGAIN			WINS. HOUSE PAYS	SETS POINT ROLLS AGAIN			WINS. HOUSE PAYS	CRAPS. LOSES BET
SUBSEQUENT PASSES :	ROLL AGAIN NO MATCH POSSIBLE		MATCH WINS. ROLL AGAIN IF NO MATCH			LOSES PAY HOUSE	MATCH WINS ROLL AGAIN IF NO MATCH			ROLL AGAIN. NO MATCH POSSIBLE	

ROUTINE CRAPGAME | START ADDR ØØØØ | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION THIS SHEET CONTAINS PROGR. INDIRECT ADDR TABLES, AND INITIALIZE THE GAME | SHEET 1

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØØØ	1B	88		CRPGAME	BCTR,UN	*CHIPSET	BRCH TO STRT OF CRAPGAME
2	2	01	B1		RLDICE	ACON	H'01B1'	INDIRECT ADDR TABLE TO THE
3	4	01	3A		DISMES	ACON	H'013A'	FOLLOWING SUBROUTINES:
4	6	01	83		ASMROL	ACON	H'0183'	ROLL THE DICE, DISPLAY THE
5	8	01	5C		DCDROL	ACON	H'015C'	MESSAGE, ASSEMBLE THE ROLL
6	A	01	0D		CHIPSET	ACON	H'010C'	OF DICE, DECODE THE ROLL,
7	C	00	D0		WINNINGS	ACON	H'00D0'	BUY INTO THE GAME, AND CALC
8								WINNINGS OR LOSSES.
9	0E	XX			BET	RES	1	CURRENT BET , AND
10	0F	XX			CHIPS	RES	1	CHIPS ON HAND
11								
12	10	77	0A		NEWGAME	PPSL	WC,COM	SET WC & LOGICAL COMPR, &
13	2	74	60			CPSU	FLAG,II	CLEAR FLG & INTPT INHIBIT
14	4	05	DC			LODI,R1	H'DC'	NOW SET DICE TABL CONST, &
15	6	CD	01	AF		STRA,R1	DICEXT	AND STORE IT FOR LATER USE
16	9	08	74			LODR,R0	CHIPS	NOW, GET CURRENT CHIPS, &
17	B	BB	F4			ZBSR	*DISLSD	BREAK INTO MS & LS NIBBLES
18	1D	CC	17	88		STRA,R0	XXBET=	THEN STORE THESE IN THE
19	20	CD	17	89		STRA,R1	XXBET=+1	XXBET= TABLE.
20	3	04	7F			LODI,R0	H'7F'	NOW SET UP RUNNING DISPLAY
21	5	C2				STRZ	R2	TO 'PLACE BET'
22	6	C3				STRZ	R3	AND GO
23	7	BB	84			ZBSR	*DISMES	DISPLAY THAT MESSAGE.
24	29	05	17			LODI,R1	<XXBET=-1	NOW SET CONSTANTS TO PLACE

INTRODUCTION: (CONTINUED)

If the player has won or lost on this roll, the computer calculates his winnings (or losses) by adding his current bet (or subtracting it) to the current number of chips he possesses. He purchased these "chips" when he 'bought into the game'. Due to the INSTRUCTOR 50's limited user memory and complexity of CRAPGAME play, it can only track one better's current holdings. Odds for payoff are fixed; they are EVEN, paying at a 1 : 1 rate.

After the current game has resulted in a win (or loss), the players are requested to place their bets once again. If the INSTRUCTOR 50 determines that the 'on-board player' has accrued more than 100 chips (or lost more chips than he has available), the CRAPGAME in progress is over, and the I/O LEDs turn on. A new game is started by depressing RST, and the 'on-board player must 'buy a fresh number of chips.

At the start of the game, the 'on-board player' may purchase up to 99 chips. Typically, he purchases about 50, to allow for rise and fall of his fortunes. Subsequently, the player may bet up to 9 chips on the current first pass of dice. He may not change his bet once the dice are rolled.

ROUTINE NEWGAME | START ADDR Ø1Ø | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION COMPLETE NEWGAME SETUP, THEN EXECUTE THE FIRST ROLL OF DICE | SHEET 2

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ2B	06	87			LODI,R2	>XXBET=-1	THE ACTUAL BET. IT IS EN-
2	D	BB	FE			ZBSR	*MOV	TERED VIA THE KEYBOARD
3	2F	04	05			LODI,R0	5	ONE DIGIT LIMITS; NO CHANGE
4	31	BB	FC			ZBSR	*GNPA	THE NEXT DIGIT EXITS FROM
5								THE SUBR. NOW, BRANCH TO
6	33	3F	01	2A		BSTA,UN	CORRSUB	CORRECT A MINOR ERROR. PRO-
7								GRAM EXPANSION WAS NECESS.
8	36	00	C1		DCDCOR	ACON	H'00C1'	SAME FOR DECODE ROUTINE.
9	8	C0				NOP		ON RETURN, SET INDEX FOR
10	9	06	8F		PASS1	LODI,R2	H'8F'	'PASS 1 = ' MESSAGE
11	B	BB	84			ZBSR	*DISMES	AND SHOW IT. ON RETURN, AS-
12	D	BB	86			ZBSR	*ASMROL	THE 1ST ROLL OF DICE.
13	3F	1B	01			BCTR,UN	$+3	BRCH AROUND PASS1 CONSTANT
14								
15	41	XX			PASS1X	RES	1	PASS1 IDENT CONST. LOC'N.
16								
17	42	05	01			LODI,R1	1	NOW SET PASS1 I.D., AND
18	4	C9	7B			STRR,R1	PASS1X	STORE IT
19	6	20				EORZ	R0	NOW, CLEAR THE PREVIOUSLY
20	7	CC	01	81		STRA,R1	SAVPAS1	SAVED DICE ROLL.
21	A	06	97			LODI,R2	H'97'	THEN SET THE CONSTANTS FOR
22	C	07	7F			LODI,R3	H'7F'	DISPLAY OF 'X--X = YY' &
23	4E	BB	84			ZBSR	*DISMES	RUN A TIME DISPLAY. THEN
24	Ø5Ø	BB	88			ZBSR	*DCDROL	DECODE THIS DICE ROLL AND

INTRODUCTION: (CONTINUED)

The INSTRUCTOR 50 provides a variety of indications to keep the game as fast-moving as possible. These include:

a. telling the player whether this is the 1st pass of dice.
b. indicating the player's "win point" if he is on 2nd and subsequent dice throws.
c. prompting the player as to his current bet and chips total after each throw of the dice.

There is nothing to prevent you (and your friends) from making side bets and raising the odds 'against the house'. CRAPGAME provides plenty of time for this.

Poker chips or play-money are useful for multi-player CRAPGAMES. You can raise the excitement level by designating one player as 'The House'. The other players are then provided the opportunity to control the dice roll by depressing the SENS key. The 'House' handles all other interactive functions.

ROUTINE SEE BELOW | START ADDR XXX | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION PASS 1 OF DICE: CONTINUED. ALSO, CONTINUATION OF DICE ROLL IF NO WIN/LOSS ON 1ST PASS | SHEET 3

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØ52	89	6D			ADDR,R1	PASS1X	IDENTIFY AS PASS1, THEN
2	4	06	00			LODI,R2	Ø	CLEAR THE PASS1 I.D. FOR
3	6	CA	69			STRR,R2	PASS1X	SUBSEQUENT ROLLS. WAS THE
4	8	E5	03			COMI,R1	3	1ST PASS A '7'? YES!!! THE
5	A	1C	00	9E		BCTA,EQ	BEATP1	PLAYER WON. EXIT.... NO!
6	D	E5	09			COMI,R1	9	WAS IT CRAPS (2 OR 3) ?
7	5F	1C	00	B1		BCTA,EQ	BOMBOUT	YES. PLAYER LOST!! EXIT
8	62	1B	D2			BCTR,UN	*DCDCOR	NO!! CHECK OTHER DECODES
9								VIA EXIT TO PASS 1 EXPAN-
10	64	C0	C0		PASCON	NOP		SION AT LOC '36' . ON RETRN
11								WIND UP PASS 1. NO. ROLL
12	66	CC	01	81		STRA,R0	SAVPAS1	WAS NOT 2,3 7,11, OR 12 SO
13	9	0C	17	9E		LODA,R0	XXFOR+6	PLAYER MUST CONTINUE ROLL;
14	C	CC	17	BE		STRA,R0	SHOOT+6	SAVE THE 1ST ROLL THEN SET
15	6F	0C	17	9F		LODA,R0	XXFOR+7	UP THE DISPLAY TABLE TO
16	72	CC	17	BF		STRA,R0	SHOOT+7	TEL WHAT POINT HE MUST ROLL
17								THEN PROMPT HIS MEMORY A-
18	75	07	7F		CONROL	LODI,R3	H'7F'	BOUT CURRENT CHIPS & BET BY
19	7	06	87			LODI,R2	H'87'	DISPLAYING 'XX BET Y'.
20	9	BB	84			ZBSR	*DISMES	ON RETURN,
21	B	07	7F			LODI,R3	H'7F'	TELL HIM THE POINT TO
22	D	06	B7			LODI,R2	H'B7	SHOOT FOR BY DISP:'SHOOTZZ
23	7F	BB	84			ZBSR	*DISMES	ON RETURN, ROLL, THE DICE
24	081	BB	82			ZBSR	*RLDICE	WHEN HE HITS SENS, STOP &

USER OPERATION - SET UP:

If you are loading the INSTRUCTOR 50's memory from tape, you must access 2 files. This is required since all available user memory (including the 64 bytes available in the SMI) are required by CRAPGAME.

Set the INSTRUCTOR 50's switches as follows:

I/O Selection Toggle Switch	EXTENDED I/O PORT Ø7
I/O Input Toggle Switches (8)	any position
Interrupt Address Toggle Switch	DIRECT
AC Line /Keyboard Interrupt Selection Slide Switch	KEYBOARD

Timed message display is controlled by the value located at address '013D' in the INSTRUCTOR 50's memory. Change the contents of this location from H'02' to H'04' or '05'. This change will lengthen the time each message is displayed while you are getting used to CRAPGAME's operation. Later, you can restore the contents of address '013D' for faster playing action.

ROUTINE AS BELOW | START ADDR XXX | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION 'CONROL' CONTINUED. ALSO 'BEATP1' IN WHICH THE PLAYER IS TOLD THAT HE BEAT THE HOUSE WITH '11' OR '7' ON HIS 1ST PASS OF DICE | SHEET 4

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ83	BB	86			ZBSR	*ASMROL	ASSEMBLE, THEN SHOW
2	5	07	7F			LODI,R3	H'7F'	HIM THE ROLL OF DICE
3	7	06	97			LODI,R2	H'97'	BY DISPLAYING 'X--X=YY.
4	9	BB	84			ZBSR	*DISMES	THEN, DECODE THE
5	B	BB	88			ZBSR	*DCDROL	LATEST ROLL OF DICE
6	D	E5	02			COMI,R1	2	WAS THIS ROLL A '7'? YES,
7	8F	1C	00	B1		BCTA,EQ	BOMBOUT	HE LOST THIS PASS. EXIT TO
8	92	C0	C0	C0		NOP		TELL HIM SO (A FEW TIME
9	5	C0				NOP		WASTE INSTRUCS). NO!! THEN
10	6	E5	04			COMI,R1	4	DID HE MATCH HIS POINT?
11	8	18	02			BCTR,EQ	BEATP1-2	YES, HE WON. TELL HIM !!!
12	A	1B	59			BCTR,UN	CONROL	NO, NO MATCH, ROLL AGAIN!!
13								*
14	9C	06	00			LODI,R2	0	HE WON !!!!!!! SO CLEAR
15	9E	CE	01	81	BEATP1	STRA,R2	SAVPAS1	OUT THE 1ST DICE ROLL.
16	A1	07	7F			LODI,R3	H'7F'	THEN SHOW A TIMED MESSAGE:
17	3	06	9F			LODI,R2	H'9F'	'YOU BEAT'
18	5	BB	84			ZBSR	*DISMES	'THE HOUSE'
19	7	07	7F			LODI,R3	H'7F'	WHILE HE'S STILL IN A
20	9	06	A7			LODI,R2	H'A7'	STATE OF EUPHORIA, GET A
21	B	BB	84			ZBSR	*DISMES	'UWIN' CONSTANT LOADED TO
22	D	05	9F			LODI,R1	UWIN	R1, THEN GO CALCULATE HIS
23	AF	9B	8C			ZBRR	*WINNINGS	WINNINGS
24								

USER OPERATION - SET UP: (CONTINUED)

Dice roll rate is controlled by the value contained in memory at address '01C9'. Roll rate may be slowed down by increasing the value greater than H'10'. Decrease of this location's contents speeds up the roll rate.
The new value should not be greater than H'7F', nor lower than H'01'. Otherwise, the roll will halt in place, and repeat forever. The contents of this location should be adjusted to a value such that you and your friends can not see any single dice pair clearly until the SENS key is depressed (too slow); also that you can clearly "see" the action of the rolling dice. H'10' is a useful value for roll rate display control.

USER OPERATION - PLAYING SEQUENCE:

The sequence for playing CRAPGAME follows on the next and succeeding pages. For initial play, set the TIMED MESSAGE DISPLAY control to H'04', as described in "User set-up operations", on the opposite page.

ROUTINE AS BELOW | START ADDR XXX | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION 'BOMBOUT' RCOGNIZES THAT THE PLAYER LOST. 'PASS1XP' FIXES PASS1 DICE DECODES; OTHER ROUTINES AS SPECIFIED | SHEET 5

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØB1	0C	00	0E	BOMBOUT	LODA,R0	BET	DICE WERE COLD!!! SO FETCH
2	4	CC	17	B7		STRA,R0	UPAYUP+7	HIS BET AND PUT IT IN THE
3	7	07	7F			LODI,R3	H'7F'	YOU PAY MESSAGE TO TELL HIM
4	9	06	AF			LODI,R2	H'AF'	THE BAD NEWS. NOW, SHOW HIS
5	B	BB	84			ZBSR	*DISMES	LOSS TO THE WHOLE WORLD,
6	D	05	AF			LODI,R1	UPAY	THEN SET A 'YOU PAY' XTNT
7	BF	9B	8C			ZBRR	*WINNINGS	IN R1, & EXIT TO CALCULATE
8								HIS LOSSES.
9	C1	E5	21		PAS1XP	COMI,R1	H'21'	PASS 1 EXPANSION:
10	3	18	5A			BCTR,EQ	BEATP1	WAS THIS PASS AN '11'? YES
11	5	E5	41			COMI,R1	H'41'	HE WON!!! EXIT. NO: WAS
12	7	18	68			BCTR,EQ	BOMBOUT	THIS PASS A '12'? YES. HE
13	9	1F	00	64		BCTA,UN	PASCON	CRAPPED OUT ON PASS1. TELL
14								HIM. NO: GO CONTIN. PASS1
15	CC	00			*FILLER	CODE		
16	CD	00	10		NEWGAME	ACON	H'0010'	IND ADDR TO NEW GAME START.
17	CF	00			*FILLER	CODE		
18								FIRST, FIND OUT IF THE
19	DØ	E5	AF		WINNINGS	COMI,R1	UPAY	PLAYER WON OR LOST. IF HE
20	2	18	25			BCTR,EQ	CALCLOS	WON, CONST = '9F' + NO COMP
21	4	0C	00	0F	WON	LODA,R0	CHIPS	IF HE LOST, THE COMP MATCHES
22	7	84	66			ADDI,R0	H'66'	THE 'AF' CONST, SO EXIT TO
23	9	8C	00	0E		ADDA,R0	BET	CALC LOSS. HE WON. FETCH
24	DC	B5	01			TPSL	CARRY	CHIPS & BET AND DECIMAL ADD.

USER OPERATION - PLAYING SEQUENCE: (CONTINUED)

Depress RST .....*to start the game. The INSTRUCTOR 50 responds with the prompt:*

" U BUY = "

Depress 1 or 2 hex keys .....*One or both hex keys are NUMERICAL digits, representing the number of chips the 'on-board player' must purchase to buy into the game.*

LIMITS: MORE THAN 1 CHIP
LESS THAN 99 CHIPS
TYPICALLY 50 CHIPS.

Depress ENT/NXT .....*to 'close the purchase from the house'. The INSTRUCTOR 50 responds with the message:*

" P L A C E B E T " followed by:
" X X BET Y

where XX = value of chips just purchased, &
Y = some previous (meaningless) number.

ROUTINE AS BELOW | START ADDR XXX | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION 'WINNINGS' CONT. ALSO CALC LOSS, AND 'WINDUP FOR GAME BEING OVER. | SHEET 6

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØDE	18	09			BCTR,EQ	GAMEOVER	IS RESULT >100?. YES!! EXIT
2	EØ	94			WINDUP	DAR	R0	TO FINISH THIS GAME. NO.
3	1	C8	F2			STRR,R0	*CHIPS	ADJUST CHIPS & BET & STORE
4	3	20				EORZ	R0	AS NEW CHIPS. HE'S FINISHED
5	4	CC	17	8F		STRA,R0	XXBET=+7	WITH CURR. BET, SO ZERO IT
6	E7	1B	E4			BCTR,UN	*NEWGAME	AND SET UP FOR NEXT BET &
7								ROLL OF DICE.
8								
9	E9	20			GAMEOVER	EORZ	R0	GAME'S OVER, SO ZERO RESIDU
10	A	C8	E9			STRR,R0	*WON+1	CHIPS,BET, & BETXX DISPL.
11	C	C8	EC			STRR,R0	*WON+6	NOW SET THE PRELIM. CHIP
12	EE	C8	F5			STRR,R0	*WINDUP+5	ENTRY CUE, AND DISPLAY IN
13	FØ	04	FF			LODI,R0	H'FF'	LIGHTS . THEN, TURN OFF THE
14	2	D4	07			WRTE,R0	PORT7	CARRY BIT (IF SET), AND EX-
15	4	75	05			CPSL	CARRY	IT TO SET NEW CHIPS VALUE
16	F6	1F	01	0D		BCTA,UN	CHIPSET	
17								
18	F9	0C	00	0F	CALCLOS	LODA,R0	CHIPS	HE LOST, SO SUBTRACT BET
19	C	AC	00	0E		SUBA,R0	BET	FROM CHIPS TOTAL. DID HE
20	ØFF	E4	F8			COMI,R0	H'F8'	LOSE MORE THAN HE HAD AVAI-
21	1Ø1	9A	66			BCFR,LT	GAMEOVER	LABLE? YES. EXIT TO TERMIN-
22	1Ø3	1B	5B			BCTR,UN	WINDUP	ATE THIS GAME. NO. WINDUP
23								& GET SET TO PLACE NEXT BET
24								

USER OPERATION - PLAYING SEQUENCE: (CONTINUED)

Depress 1 hex key .....*to place a bet for the 'on-board' player. The bet may not exceed 9. It may be zero if the player elects to 'sit out this roll'. Once the bet is placed, no change is possible.*

NOTE: DUE TO PROGRAM LIMITATIONS, THE PREVIOUS BET 'MOVES OVER', RATHER THAN BEING BLANKED. IGNORE IT

Depress any other hex or function key. .....*The dice-roll commences with the display:*

" R O L L X_1 X_2 "

where X1 and X2 are numbers between 1 and 6, representing the faces of the rolling dice.

Depress SENS .....*To stop the roll of dice. The INSTRUCTOR 50 responds with the following messages:*

" P A S S 1 = " and
" X - - X = Y Y "

ROUTINE AS BELOW | START ADDR XXX | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION 'CHIPSET' PROVIDES MEANS FOR PLAYER TO BUY INTO THE GAME. 'CORRSUB' PROVIDES A REQU. FIX TO THE PASS1 ROUTINE | SHEET 7

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø1Ø5	12	17	0B	CHIPSXX	RES	8	CHIPS MESSAGE:
2	8	12	1B	17				'U BUY = '
3	B	16	17					
4								
5	1ØD	05	01		CHIPSET	LODI,R1	<CHIPSXX-1	PUT CHIPS MESSAGE INTO DIS-
6	1ØF	06	04			LODI,R2	>CHIPSXX-1	PLAY BUFFER, THEN SET 'GNP'
7	111	BB	FE			ZBSR	*MOV	PARAMETER FOR 2-DIGIT ENTRY
8	3	04	01			LODI,R0	1	OF CHIP °'BUY' INTO GAME.
9	5	BB	FC			ZBSR	*GNP	PLAYER CAN CHANGE HIS PUR-
10	7	CC	00	0F		STRA,R0	CHIPS	CHASE UNTIL A FUNC. KEY IS
11	A	1F	00	10		BCTA,UN	NEWGAME	DEPRESSED. STORE SELECTED
12								VALUE IN 'CHIPS', AND EXIT
13	11D	00			*FILLER	DATA		TO PLACE A BET
14					*ADDR'S 011E TO 0125 CONTAIN A FIX TO DECODE ROLL ROUTINE ***			
15	0126	00	00		*DOCUMENTED WITH THAT ROUTINE **********			
16	8	00	00		*FILLER CODE - LOCATIONS '0126' TO '0129'			
17	Ø12A	44	0F		CORRSUB	ANDI,R0	H'0F'	THIS SUBR. INSERTED TO COR-
18	C	CC	00	0E		STRA,R0	BET	RECT PROGRAM ERROR. MASK
19	12F	CC	17	8F		STRA,R0	XXBET=+7	OFF RESIDUAL MS NIBBLE, THEN
20	132	BB	82			ZBSR	*RLDICE	STORE THE ROLL OF DICE.
21	4	07	7F			LODI,R3	H'7F'	AFTER THE ROLL, SET UP A
22	136	17				RETC,UN		TIMED 'PASS1 =' PARAMETER,
23								& RETURN TO MAIN PROGRAM.
24								

USER OPERATION - PLAYING SEQUENCE: (CONTINUED)

where X & X represent the "face-up" positions of the stopped dice, and YY equals the combined total of the dice.

At this point, the message sequence displayed by the INSTRUCTOR 50 is determined by whether the player wins, loses, or must roll the dice again. No operator intervention is required....or desired !!!

PLAYER WINS: PASS 1 DICE EQUAL 7 OR 11:

"YOU BEAT" .....*tells everyone that you won on the first*
"THE HOUSE" *pass of dice. Collect your winnings, and*
"PLACE BET" *get ready to place your next bet. Meanwhile, the INSTRUCTOR 50 calculates your new (higher) total of chips!!!!!!*

"XX BET Ø" .....*The new (total) number of chips is displayed. The previous bet is zeroed. Place your bet (described previously) to continue the game.*

PLAYER LOSES: PASS 1 DICE EQUAL 2, 3, OR 12:

"U PAY X" .....*Tells the world that you CRAPPED OUT on the first roll of dice. 'X' equals the amount of the bet which the 'on-board*

ROUTINE DISMES | START ADDR 13A | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION 'DISMES' CONTROLS ACCESS AND DISPLAY TIME OF 8 MESSAGES LOCATED IN SMI MEMORY: LOCATIONS '1780' TO '17BF' | SHEET 8

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø137	XX			>MESLOC-1	RES	1	THESE 3 BYTES ARE CONSTANTS
2	8	XX			DSPLDLY	RES	1	USED DURING DYNAMIC DISPLAY
3	9	XX			RUNDSPX	RES	1	OF THE MESSAGE.
4								ON ENTRY, STORE CURRENT
5	Ø13A	CA	7B		DISMES	STRR,R2	>MESLOC-1	INDEX TO THE MESSG TABLE
6	C	05	02			LODI,R1	2	NOW SET RUNNING DISPL TIME-
7	13E	C9	79		REPEATM	STRR,R1	RUNDSPX	OUT & STORE. ALSO STORE
8	14Ø	CB	76		DSPAGN	STRR,R3	DSPLDLY	STATIC/DYNAMIC DISPLAY
9	2	05	17			LODI,R1	<MESLOC-1	PARAM. NOW SET INDX TO SPE-
10	4	0A	71			LODR,R2	>MESLOC-1	CIFIED MSG, AND GO
11	6	BB	E6			ZBSR	*USRDSP	DISPLAY IT. IF STATIC, PLYR
12	8	0B	6E			LODR,R3	DSPLDLY	DEPR ANY KEY TO EXIT. IF
13	A	E7	00			COMI,R3	Ø	DYNAMIC, IS DSPLDLY A '1'
14	C	14				RETC,EQ		YET? YES! EXIT TO CONTINUE
15	D	E7	01			COMI,R3	1	DISPLAY. NO! REPEAT SINGLE
16	14F	18	02			BCTR,EQ	CONTDSP	PASS DISPLAY.
17	151	FB	6D			BDRR,R3	DSPAGN	TO CONTINUE RUNNING TH
18	3	09	64		CONTDSP	LODR,R1	RUNDSPX	DISPLAY, GET ITS CONSTANT
19	5	F9	01			BDRR,R1	$+3	& DECREMENT. IF 'Ø', EXIT
20	7	17				RETC,UN		TO CALLING ROUTINE. IF NOT,
21	8	07	7F			LODR,R3	H'7F'	SET UP THE DYNAMIC PARAME-
22	15A	1B	62			BCTR,UN	REPEATM	TER, AND REPEAT.
23								
24								

USER OPERATION - PLAYING SEQUENCE: (CONTINUED)

player' must pay 'the house'. The 'house' also collects losing bets from the other players.

"PLACE BET".....*as previously described.*

"XX BET Ø".....*as described previously, except the new total of chips is now lower. Place your bets!!!!. Better luck next time.*

PLAYER MUST ROLL AGAIN: PASS 1 DICE EQUAL 4, 5, 6, 8, 9, OR 10:

"XX BET Y".....*prompts on-board player to remind him of the value of his current chips (XX) and bet (Y). The bet can't be changed.*

"SHOOT XX".....*This prompt tels all players the "point" which must be rolled in order for a win to take place on the Pass Line. XX equals the dice total made on the first pass.*

"ROLL X X".....*Automatic roll of dice. Stops when the SENS key is depressed.*

ROUTINE DCDROL	START ADDR Ø15C	PART OF PROGRAM CRAPGAME	2650 PROGRAMMING FORM
DESCRIPTION DECODES CURRENT DICE-ROLL FORWIN, LOSE, OR THROW AGAIN			SHEET 9

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø15C	08	24		DCDROL	LODR,R0	DICEASM	FETCH DICE. WAS THIS 'ROLL'
2	15E	E4	07			COMI,R0	7	OF DICE A '7'. IF SO,
3	16Ø	1C	01	1E		BCTA,EQ	SEVEN	EXIT TO SET EQUIV. CONST.
4	3	E4	11			COMI,R0	H'11'	IF NOT, WAS IT '11'?
5	5	1C	01	23		BCTA,EQ	ELEVEN	YES: EXIT TO 'ELEVEN SAME
6	8	E4	03			COMI,R0	3	PURPOSE. DID HE CRAP OUT?
7	A	99	11			BCFR,GT	CRAPOUT	YES. EXIT TO SET CRAPOUT XT
8	C	E4	12			COMI,R0	H'12'	WAS IT '12'? IF SO, SET 1ST
9	16E	18	07			BCTR,EQ	PAS112	PASS '12' CONST. DID THIS
10	17Ø	E8	0F			COMR,R0	SAVPAS1	ROLL MATCH PREVIOUS 1ST
11	2	18	06			BCTR,EQ	MATCH	PAS? YES. SET MATCH CONST.
12	4	05	10			LODI,R1	H'10'	NO. NO MATCH. SET NO MATCH
13	6	17				RETC,UN		DECODE & RETURN.
14	7	05	40		PAS112	LODI,R1	H'40'	SET PASS 1: '12' DECODE
15	9	17				RETC,UN		AND RETURN
16	A	05	04		MATCH	LODI,R1	4	SET 'MATCH' DECODE
17	C	17				RETC,UN		AND RETURN
18	D	05	08		CRAPOUT	LODI,R1	8	SET CRAPOUT DECODE,
19	Ø17F	17				RETC,UN		AND RETURN
20					*NOTE RELOCATED ADDR'S FROM PREV.			MEM. NEEDED MORE DECODES***
21	Ø11E	05	02		SEVEN	LODI,R1	2	SET '7' DECODE
22	12Ø	17	00	00		RETC,UN		AND RETURN /FILLER CODE
23	123	05	20		ELEVEN	LODI,R1	H'20'	SET '11' DECOSE
24	Ø125	17				RETC,UN		AND RETURN.

USER OPERATION - PLAYING SEQUENCE: (CONTINUED):

PLAYER WINS: SUBSEQUENT PASS EQUALS HIS "POINT":

Same message sequence as that described for winning with a '7' or '11' in Pass 1.

PLAYER LOSES: SUBSEQUENT PASS EQUALS "7":

Same message sequence as that described for losing (crap out) with a '2', '3', or '12' in Pass 1.

NO MATCH ON THIS ROLL - NO "7" EITHER:

Repeat message sequence for subsequent roll. Is the same as that described when player rolled '4, 5, 6, 8, 9, or 10' in Pass 1.

PLAYER EXCEEDS 100 CHIPS, OR LOSES MORE THAN HE HAS AVAILABLE:

Same message sequences as appropriate for winning or losing. However, after the message "YOU BEAT THE HOUSE" (U PAY XX), the 8 I/O LED's turn on and the message:

"U B U Y = "..... *prompts the player to buy into a new game.*

ROUTINE ASMROL	START ADDR Ø183	PART OF PROGRAM CRAPGAME	2650 PROGRAMMING FORM
DESCRIPTION THIS ROUTINE ADDS THE ''ROLL'' OF INDIVIDUAL DIE PRIOR TO DECODE OF THE ROLL FOR WIN/LOSE/THROW AGAIN.			SHEET 10

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø18Ø	00			*FILLER	CODE	**********	
2								
3	Ø181	XX			SAVPAS1	RES	1	SAVE 1ST ROLL OF DICE.
4	2	XX			DICEASM	RES	1	VALUE OF DICE ADDED FROM
5								CURRENT ROLL OF 2 DIE.
6	Ø183	08	22		ASMROL	LODR,R0	ROLL+5	GET CURRENT DICE ROLL WHEN
7	5	09	22			LODR,R1	ROLL+7	STOPPED.STORE IN 'XXFOR'
8	7	CC	17	98		STRA,R0	XXFOR	MSG. CONVERT FROM HEX TO
9	A	CD	17	9B		STRA,R1	XXFOR+3	DECIMAL NOTATION. 1ST DIE
10	D	84	66			ADDI,R0	H'66'	+ H'66' + 2ND DIE = R0 RSLT
11	18F	81				ADDZ	R1	DO A DECIMAL ADJ
12	190	94				DAR	R0	THEN STORE IN DICEASM. NOW
13	1	C8	6F			STRR,R0	DICEASM	BREAK RESULT INTO 2 NIBBLES
14	3	BB	F4			ZBSR	*DISLSD	AND PUT LS NIBL IN XXFOR+7
15	5	CD	17	9F		STRA,R1	XXFOR+7	WAS ROLL ≥ 10? YES. BYPASS
16	8	E4	01			COMI,R0	1	NEXT INST. NO. GET A
17	A	18	02			BCTR,EQ	$+4	SPACE CODE, AND STORE A
18	C	04	17			LODI,R0	H'17'	SPACE
19	19E	CC	17	9E		STRA,R0	XXFOR+6	OR A 1 IN XXFOR+6, THEN
20	1A1	17				RETC,UN		EXIT TO CALLING ROUTINE
21								
22								'ROLL+5' AT ADDR H'1A7'
23								'ROLL+7' AT ADDR H'1A9'
24								

THE PROGRAM:

Program "CRAPGAME" is organized around a sequenced mainline set of instructions, from which are called a series of dedicated subroutines as required. In sheet 1; lines 2 through 7, an indirect address table for access of supporting subroutines is listed. Typically, the mainline program calls one, or a series of more than 1, of these subroutines as it is sequenced. The major subroutines include:

"RLDICE" .....*control roll and stopping of the dice.*
"DISMES" .....*controls display of all messages except the rolling dice and chips purchase messages.*
"ASMROL" .....*assembles the total of the 2 die, once the roll has stopped.*
"DCDROL" .....*decodes for all win, lose, and roll again conditions in Pass 1 and subsequent passes.*
"CHIPSET" .....*sequences entry of the player's "Buy into the game".*
"WINNINGS" .....*provides steering to calculate all winnings and losses, and to adjust current chip total.*

The "Place Bet" routine is contained in the mainline program under the label "NEWGAME" (sheet 1; line 12). Here, the mes-

ROUTINE RLDICE	START ADDR Ø1B1	PART OF PROGRAM CRAPGAME	2650 PROGRAMMING FORM
DESCRIPTION THIS ROUTINE PROVIDES A RAPID TIMED SEQUENCE THRU ALL DICE COMBINATIONS TO SIMULATE DICE ROLL. EXIT WHEN PLYR DEPRESSES SENS			SHEET 11

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø1A2	13	15	01	ROLL	RES	8	'ROLL X X' MESSAGE ''INDI-
2	5	01	17	XX				VIDUAL DIE'' INTO ROLL+5
3	8	17	XX					& ROLL + 7.
4								
5	Ø1AA	05	DB		GETXT	LODI,R1	H'DB'	GET DICE TABLE INITIAL IN-
6	C	C9	01			STRR,R1	DICEXT	DEX CONST. STR IN 'DICEXT'
7	E	17				RETC,UN		AND RETURN.
8								
9	1AF	XX			DICEXT	RES	1	DICE TABLE CONSTANT
10	1BØ	XX			ROLDLY	RES	1	ROLL DICE DELAY CONST.
11								
12	Ø1B1	76	60		RLDICE	PPSU	FLAG,II	TURN ON FLAG LED & INTPT
13	3	09	7A		ROLLING	LODR,R1	DICEXT	INHIB. GET CURRENT DICEXT
14	5	85	01			ADDI,R1	1	& INCR. DICE TABLE READ
15	7	B5	01			TPSL	CARRY	COMPL? YES. REINIT. INDEX.
16	9	38	6F			BSTR,EQ	GETXT	NO. INDX INTO NXT ENTRY
17	B	0D	61	00		LODA,R0	DICETB,R1	& STORE UPDATED INDEX. NOW
18	1BE	C9	6F			STRR,R1	DICEXT	SHOW ENTRY IN LIGHTS,
19	1CØ	D4	07			WRTE,R0	PORT7	& THEN BREAK THE
20	2	BB	F4			ZBSR	*DISLSD	LATEST ENTRY INTO NIBBLES.
21	4	C8	61			STRR,R0	ROLL+5	THEN PUT IN 'ROLL X X' MSG.
22	6	C9	61			STRR,R1	ROLL+7	NOW, SET THE ROLL DISPLAY
23	8	07	10			LODI,R3	H'10'	CONST, & PUT IN 'ROLDLY'.
24	1CA	CB	64		AGAIN	STRR,R3	ROLDLY	THEN, FETCH THE ADDRESS

THE PROGRAM: (CONTINUED)

sages: "PLACE BET" & "XX BET Y" are sequenced, "PLACE BET" under the control of subroutine "DISMES", and "XX BET Y" controlled by the INSTRUCTOR 50's MONITOR subroutine "GNPA" (Get Numbers Parameters And display). "GNPA" is accessed indirectly by the ZBSR instruction in sheet 2; line 4. Until the operator places his bet, and subsequently depresses any other HEX or FUNCTION key, the message is displayed. This type of display is controlled by programming '05' in the LODI,R0 instruction; line 3.

The BSTA instruction (line 8) was necessary in order to permit additional programming for a bug found during proof of the program's operation.

In line 15, a constant (PASS1X) is conditioned to track whether this is Pass 1 of the dice, or whether a subsequent roll is taking place. The win/loss decode software depends on the condition of this constant in its decision making process.

Routine "PASS1", starting at line 9 ; sheet 2, controls all message sequencing for operations taking place during Pass 1 dice

ROUTINE RLDICE | START ADDR CONTIN | PART OF PROGRAM CRAPGAME | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF 'RLDICE', ALSO THE 36-BYTE DICE TABLE. | SHEET 12

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1								
2	Ø1CC	05	01			LODI,R1	<ROLL-1	CONSTANTS FOR THE 'ROLL X X
3	1CE	06	A1			LODI,R2	>ROLL-1	MESSAGE, AND DISPLAY THE
4	1DØ	BB	E6			ZBSR	*USRDSP	CURRENT ROLL. THEN RECALL
5	2	0B	5C			LODR,R3	ROLDLY	THE ROLL DSPLY DLY, AND SEE
6	4	FB	74			BDRR,R3	AGAIN	IF FINISHED. IF NOT, DO A-
7	6	B4	80			TPSU	SENS	GAIN. WHEN FINISHED, SEE
8	8	98	59			BCFR,EQ	ROLLING	IF THE PLAYER HAS STOPPED
9	Ø1DA	17				RETC,UN		THE ROLL. YES? EXIT TO CAL-
10								LING RTN. NO? KEEP ROLLING
11								
12	Ø1DB	34	41	13	DICETB	RES	37	'DICE TABLE' HOLDS ALL
13	1DE	36	24	51				POSSIBLE COMBINATIONS OF
14	1E1	25	66	21				2 6-SIDED DIE, PLUS
15	4	31	61	16				AN EXTRA '7' ('34' AT
16	7	12	56	46				ADDRESS '1DB' WHICH IS
17	A	45	53	64				ACCESSED ONLY ONCE AT
18	1ED	54	23	32				THE START OF EACH GAME.
19	1FØ	33	35	65				
20	3	11	14	63				
21	6	42	62	26				
22	9	43	34	55				
23	C	22	44	15				
24	Ø1FF	52						

THE PROGRAM: (CONTINUED)

roll. If subsequent rolls are required (e.g. no match; no loss or win.) the mainline program sequence continues with routine "CONROL". Any winning roll detected during Pass 1 or on a subsequent roll causes program control to execute routine "BEATP1" in sheet 4. A loss detection turns program control over to the routine "BOMBOUT", in sheet 5 of the main program.

"BEATP1" and "BOMBOUT" respectively set up message displays for 'You beat the House', and 'U pay XX', then set unique constants prior to transfer of program control to subroutine "WINNINGS".

"WINNINGS" (sheet 5) compares the unique constant with one of its own: "UPAY". If the constants match, this means that the player has lost, and process "CALCLOS" (sheet 6) is executed. Here, the current bet is subtracted from current chips, then displayed in routine "WINDUP", also found in sheet 6.

If the constants do not match, a WIN is indicated. Program execution falls through to routine "WON", where the current bet and chips are summed. "WINDUP" is then executed.

Both "CALCLOS" and "WON" test to see if the number of chips respectively exceed 100, or are less than zero. If so, routine "GAMEOVER" (sheet 6) is executed."GAMEOVER" zeros both current

ROUTINE MSGTBL	START ADDR 1780	PART OF PROGRAM CRAPGAME	2650 PROGRAMMING FORM

DESCRIPTION 'MSGTBL' IS ACCESSED ONLY BY ROUTINE 'DISMES'. CONSISTS OF EIGHT 8-BYTE MESSAGES IN SMI RAM LOCATIONS '1780'-'17BF'

SHEET 13

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	1780	10	11	0A	PLACEBET	RES	8	''PLACEBET''
2	3	0C	0E	0B				
3	6	0E	87					
4	1788	XX	XX	17	XXBET	RES	8	''XX BET Y''
5	B	0B	0E	07				XX = CHIPS ON HAND
6	8E	17	17					Y = CURRENT BET
7	1790	10	0A	05	PASS1=	RES	8	'' PASS 1 = ''
8	3	05	17	01				
9	6	17	16					
10	1798	XX	19	19	XXFOR	RES	8	''X--X = YY
11	B	XX	17	16				X & X = INDIV. DIE
12	E	YY	YY					YY = 2 DIGITS ADDED DICE
13	17A0	1B	00	12	YOUBEAT	RES	8	''YOU BEAT''
14	3	17	0B	0E				
15	6	0A	07					
16	17A8	07	14	0E	THEHOUS	RES	8	''THE HOUSE''
17	B	14	00	12				
18	D	05	0E					
19	17B0	12	17	10	UPAYUP	RES	8	''U PAY X''
20	3	0A	1B	17				X = BET WHICH PLR LOST.
21	6	17	XX					
22	17B8	05	14	00	SHOOT	RES	8	''SHOOT XX''
23	B	00	07	17				XX = PT PLAYER MUST SHOOT
24	17BE	17	XX					FOR TO WIN.

THE PROGRAM: (CONTINUED)

chips and bet indications, and turns on the I/O LEDs. The LEDs remain lighted until the next dice roll takes place in a new game.

Subroutine "CHIPSET" (sheet 7) accesses the MONITOR's subroutine "GNP" (Get Numbers Parameters). This differs from subroutine "GNPA" in that the least significant 3 or 4 digits are blanked when the first hex digit is depressed by the operator. This 'makes room' for data entered by the operator. The value loaded into R0 (line 8; sheet 7) determines whether 2 or 4 digits may be entered, and how many preceding display digits (least significant 3 or 4 digits are blanked. The '01' programmed in location '0114' conditions the routine to:

* blank the 3 least significant digits, and
* expect 2 digits entry with change permitted.

This is why the "EQUALS" sign goes out when you enter the first chips digit entry !!!!! Further, you've got to depress some FUNCTION key (e.g. MEM; REG, or ENT/NXT) to exit from the routine.

"DISMES" (sheet 8) provides controls for both static ('display until a key is depressed') and dynamic (automatically timed-out) message display. Your author will demonstrate how you can check out both static and dynamic (timed) display later on. The key lies in the contents of R3 prior to the time the MONITOR's User Display ("USRDSP") routine is executed. As long as R3 contains a number between '01' and '7F', "USRDSP" makes one display pass, then immediately exits. If R3 equals '00', "USRDSP" will display the current message until the operator depresses some HEX or FUNCTION key. This is the reason behind programming of the COMI R3, Ø instruction in line 13.If R3 = 0, the player will have just depressed a key to exit the display. The following instruction, RETC,EQ is satisfied, and program control returns to the main program. All this takes place for a static display.

The instructions in lines 15 through 22 provide control for timing a desired dynamic message. Decrement of R3, from '7F' to '01', takes about 1/2 second when combined with time required by subroutine "USRDSP". Message display time is controlled by programming location '013D'. Routine "DISMES" will run the display through a precise number of single passes. The formula is:

(R1) x H'7F' = SINGLE PASSES

WHERE (R1) = VALUE LOADED INTO R1 BY THE LODI INSTRUCTION AT LINE 6.
H'7F' IS THE CONSTANT INITIALIZED IN R3 BEFORE ENTRY TO ''DISMES'', OR BY THE LODR INSTRUCTION AT LINE 21.

A value of '04' initialized in line 6 provides a running display time equal to about 2 seconds. Take a few minutes to read the comments in lines 15 to 22 , then continue on the next page.

THE PROGRAM: (CONTINUED)

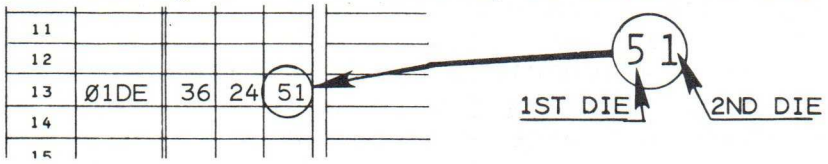
Routine "DCDROL" (sheet 9) presents some interesting problems. Basically, this routine tests the ASSEMBLED ROLL of dice for various number values and sets an appropriate constant (in R1) for each specific value detected. All general dice possibilities are coverd; these include:

CRAPOUT.....2 or 3 in dice - will be tested in PASS 1 only.
SEVEN.....7 rolled - will be tested for win(Pass1) or lose.
ELEVEN.....11 rolled - will be tested for a win in PASS 1 only.
PAS112.....12 rolled - will be tested for crapout in PASS 1.
MATCH.....rolled dice = previous PASS 1 on subseq. roll.
NO MATCH....If saved dice don't match this roll; main program will roll the dice on subsequent passes.

As indicated, the actual test for WIN, LOSE, or ROLL AGAIN is performed upon return to the main program from DCDROL.

Routine "ASMROL" (sheet 10) is accessed by the main program after the player depresses the SENS key to terminate the dice roll. "ASMROL"'s sole function is to set up the variables in the message "X--X =YY". After placing the values corresponding to the face-up values of the individual dice in the "X--X" portion of the message, "ASMROL" adds up the dice, and inserts the result into the "Y" part of the message. If the total equals less than 10, the leading "Y" digit is blanked.

Subroutine "RLDICE" is a special high-speed routine dedicated to controlling all throws of the dice. In lines 13 to 18, an index into a 36-entry data table is set up, then used to fetch a single 1-byte entry from "DICETB" (sheet 12; lines 12 and following). All dice combinations are contained in this table, each die face represented by a 4-bit nibble, as shown in the following diagram.



After fetch, the binary code of the "DICETB" entry is transmitted to the I/O LEDs. The entry is then split into 2 nibbles (line 20; sheet 11) and then placed in the "ROLL X X" message and displayed. The constant loaded into R3 (line 23; sheet 11) determines the display time for each current roll. After the roll display has timed out (sheet 12; line 6) the SENS key input is tested. Roll is repeated (or terminated) depending on the result of that test.

Subroutine "GETXT" (sheet 11; line 5) is accessed by routine "RLDICE" whenever the 'DICETB' index must be reinitialized. The TPSL instruction (line 15) tests whether the current index has carried through 'FF' to 'ØØ'. When it does, the CARRY bit is active, and

THE PROGRAM: (CONTINUED)

program control branches to "GETXT". Here, the index constant is re-initialized to H'DB', the least significant byte of the 'DICETB's start address. The LODA instruction (line 17) adds this index to a fixed base address (' 1ØØ') contained in its 2nd & 3rd bytes. The result provides the proper memory address into the table.

In "DICETB" (sheet 12) note that the code '34' is repeated twice, at locations '01DB', and at '01FA'(line 22). The code at location '01DB' is never accessed, however, since the initial index is incremented by 1 before the 1st access is made into DICETB.

The routine "DISMES" (sheet 8) accesses an 8-entry data message table; ("MESTBL") on sheet 13. Each message entry contains 8 bytes. Message variables are identified by 'XX' or 'YY' in the codes. "MESTBL" is located in the user-available memory section of the 2656 Systems Memory Interface (SMI) chip, at addresses '1780' through '17BF'.

This concludes the description of program "CRAPGAME".

STATIC / AUTO MESSAGE DISPLAY MODE CHECKOUT:

As presently coded, the mainline program does not exercise the STATIC message display mode feature of subroutine "DISMES". All "DISMES"-accessed messages are displayed for a short period of time only. The following modifications are provided in order to checkout STATIC DISPLAY mode:

NOTE: THIS MODIFICATION CAN WORK TO YOUR ADVANTAGE BY PERMITTING THE PLAYERS TIME TO PLACE INTERMEDIATE OR SIDE BETS AFTER THE FIRST PASS OF DICE OR EVEN AFTER EACH DICE ROLL IF THERE IS NO WIN OR LOSS ON THE FIRST PASS.

TO DISPLAY THE STOPPED ROLL OF DICE AND THEIR TOTAL UNTIL A PLAYER DEPRESSES A HEX OR FUNCTION KEY:

PASS 1 ONLY:

Depress MON MEM 4 D ENT/NXT ..... *to select a memory alter address of the constant programmed for timed display of the message:*

" X--X = YY "

Depress Ø ENT/NXT ..... *to condition the display to show the message until any key is depressed.*

STATIC / AUTO MESSAGE DISPLAY MODE CHECKOUT: (CONTINUED)

SUBSEQUENT ROLLS ONLY - NO STOP ON PASS 1 ROLL:

Depress MEM 8 6 ENT/NXT .....*address of memory change.*

Depress Ø ENT/NXT .....*dice will be displayed until player depresses Hex or Function key.*

Play as usual. Both changes may be incorporated if desired. To restore, change contents at affected addresses from 'ØØ' back to '7F', as shown in the program.

I/O DISPLAY WHEN 'ON-BOARD PLAYER EXCEEDS 100 CHIPS, OR LOSES MORE THAN HE HAS:

Play in normal fashion, except 'buy into the game' with 99 (1) chip(s), and attempt to win (lose) an appropriate bet. When the limits are exceeded, all 8 I/O LEDs light, remaining on until the next dice roll takes place.

NOTES:

BEAT THE ODDS-VARIATION 1**

INTRODUCTION AND PLAY:

This simple game will provide more "action" in an evening than you ever dreamed of, particularly if your friends enjoy some friendly wagering. Use of poker chips or play money provides a realistic excitement to the game. After you have loaded "BEAT THE ODDS" into the INSTRUCTOR 50, play as follows:

1. Depress RST then INT ...*When the FLAG LED turns on, your friends place their bets in the field of their choice. Then.....*

2. Depress SENS .....*to start the random generation of numbers. Then.....*

3. Depress SENS again .....*when any player calls for display of the generated number. Bets are paid "by the House" as indicated in the 'Tables' diagram on the next page.*

4. Depress INT .....*to call for placing new bets*

** Variation 2 is contained in the INTRODUCTORY CASSETTE tape.

GAME AND PROGRAM BY JOHN GARCEAU

ROUTINE AS SHOWN	START ADDR ØØØ	PART OF PROGRAM ODDS VER 1	2650 PROGRAMMING FORM

DESCRIPTION	SHEET
PROGRAM 'ODDS' IS AN ELECTRONIC LOTTERY, WITH FULL RANDOM NUMBER SELECTION. HOWEVER IT DOES NOT CALCULATE WINNINGS OR LOSSES	1

LINE	ADDRS.	DATA ØØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØØØ	1B	0E		ODDS	BCTR,UN	NEWGAME	ON RESET, GO START NEWGAME.
2	Ø2-6	00	00	00	*FILLER	CODE		
3	7	00	80		INTADD	ACON	H'0080'	ENTER ROUTINE TO PLACE BET
4	09-F	00	00	00	*FILLER	CODE		ON INT KEY DEPRESS ONLY
5								
6	Ø1Ø	20			NEWGAME	EORZ	R0	INIT R0=R3='ØØ' THEN
7	1	C3				STRZ	R3	CLEAR SAVECT (256 LOCATIONS
8	2	CF	21	00		STRA,R0	SAVECT,R3,+	STARTING AT LOC.'1ØØ')
9	5	5B	7B			BRNR,R3	$-3	FINISHED? YES! GO TURN OFF
10	7	74	40			CPSU	FLAG	FLAG LED, AND LOOP UNTIL
11	9	1B	7E		PLAY	BCTR,UN	PLAY	INT KEY DEPR. TO PLACE BETS
12	1B-1F	00	00	00	*FILLER	CODE		
13	2Ø	77	08			PPSL	WC	SET CARRY FOR CHAR. BUILDUP
14	2	0D	1E	70		LODA,R1	RIGID1	NOW, GET A BYTE FRØM RIGID
15	5	0D	5C	89		LODA,R0	RIGID2,R1,-	ADDRESS (TWICE, AND ENTER
16								INTO THE COMPUTATION,
17	8	07	80		ROLLING	LODI,R3	H'80'	1ST, SET ROLL RATE, THEN
18	A	04	01			LODI,R0	1	INCR. SAVECT,R1 INDEXED.
19	C	8D	61	00		ADDA,R0	SAVECT,R1	NOW TEST TO SEE IF THE LO-
20	2F	B5	01			TPSL	CARRY	CATION HAS CARRIED? YES!!
21	31	18	11			BCTR,EQ	GAMEOVER	EXIT TO GAME OVER. NO!!
22	3	CD	61	00		STAR,R0	SAVECT,R1	SAVE THE INCREM CT, AND DO
23	6	FB	7B			BDRR,R3	$-3	ROLL RATE. DONE?? YES!!
24	Ø38	8D	5D	00		ADDA,R0	NEWNBR,R1,-	FETCH ANOTHER RANDOM #, ADD

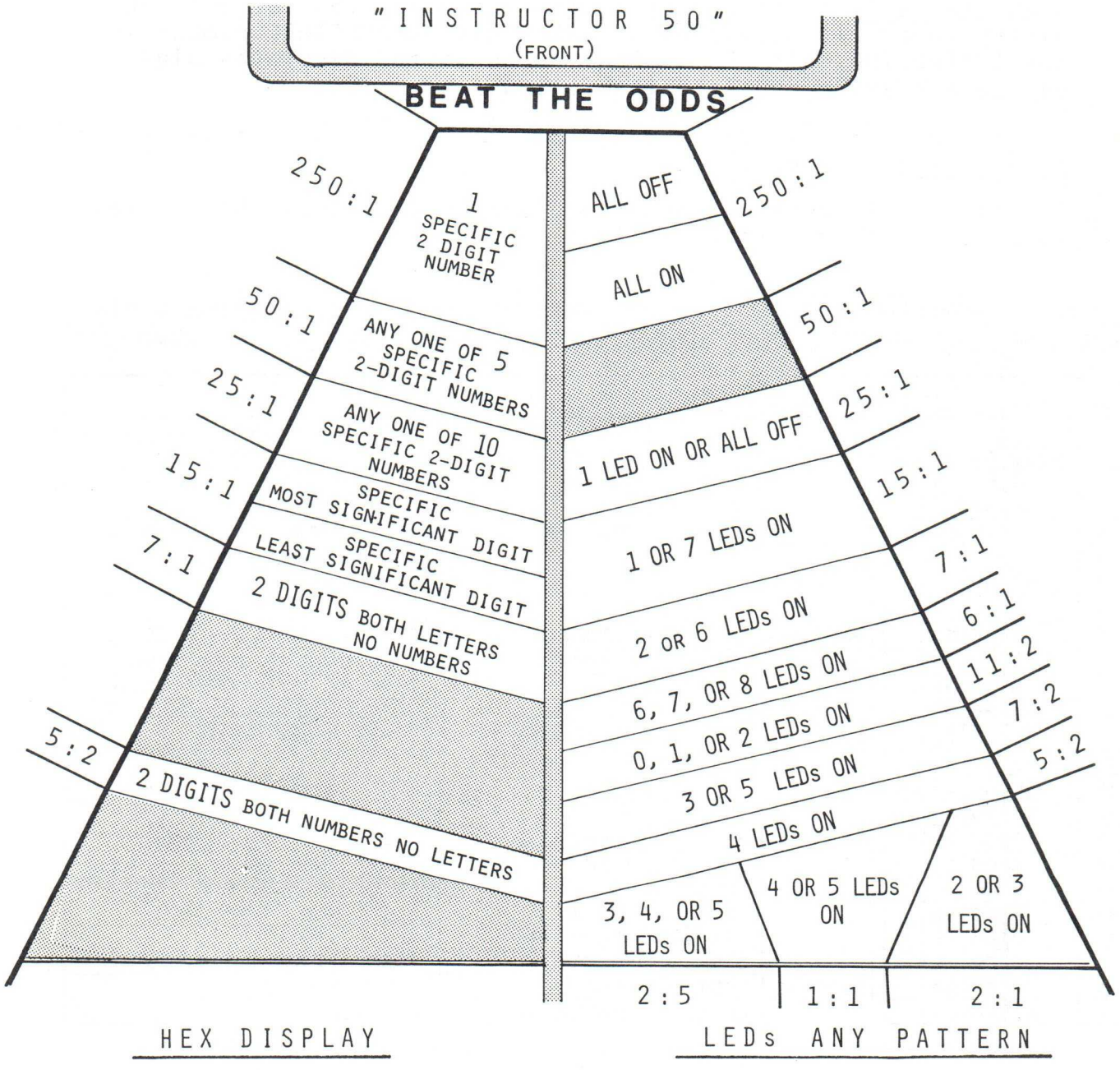
PLAYING 'TABLE'

Use a large piece of light-colored construction paper to duplicate the "Playing Table" model illustrated below. For play, place on a convenient card table facing the players. Locate the INSTRUCTOR 50 as shown.

If the player opts to select 5 or 10 specific 2-digit numbers, he should jot them down on a small piece of paper. Then, he must place the paper, together with his bet, on the table during the "PLACE BET" cycle. Any one of his 5 (or 10) numbers wins if matched.

Note: The "Table" is drawn to an approximate 1/3 or 1/4 scale.

Preparation for play is specified on page 6-65.



INTRODUCTION AND PLAY (CONTINUED)

5. Repeat steps 2, 3, & 4 .....*until the message "YOU HOPE" is displayed. At this time, the game is over. Repeat from step 1 to start a new game.*

THE PROGRAM:

Program "ODDS" consists of 2 major routines; these are "ROLLING" and "DISPLAY". "Rolling" provides for a high-speed random number generation after the SENS key is depressed the first time. Generated numbers are 8-bit binary codes, thus 256 possible combinations may be generated.

When the SENS key is depressed the 2nd time, the current 8-bit binary number is displayed as a 2-digit HEXADECIMAL number in the INSTRUCTOR 50's HEX DISPLAY. Each of the displayed digits may have a value equal to the following:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12) D(13), E(14), or F(15).

The display is blanked when the operator depresses the INT key to allow further betting to take place.

ROUTINE AS SHOWN | START ADDR XXXX | PART OF PROGRAM ODDS (VER1) | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATON OF CONTROL PROGRAM FOR ODDS GENERATION AND DISPLAY. | SHEET 2

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1								*TO R0 CONTENTS, THEN SHOW
2	0003B	D4	07			WRTE,R0	PORT7	THE HIGH SPEED ROLL IN
3	D	B4	80			TPSU	SENSE	LIGHTS. DOES THE PLAYER
4	3F	38	0D			BCTR,EQ	DISPLAY	WANT TO SEE HIS ROLL?
5	41	1B	65			BCTR,UN	ROLLING	YES!! SHOW IT TO HIM. NO:
6								NOT YET; KEEP ROLLING.
7	43	00			*FILLER	CODE		
8								
9	44	05	00		GAMEOVER	LODI,R1	<OVER-1	SET DISPLAY CONSTANTS TO
10	6	06	77			LODI,R2	>OVER-1	SHOW GAME IS COMPLETED,
11	8	07	01			LODI,R3	1	DISPLAY 'YOU HOPE' UNTIL
12	A	BB	E6			ZBSR	*USRDSP	PLAYER DEPRESSES 'RST' TO'
13	4C	1B	76			BCTR,UN	GAMEOVER	START A NEW GAME.
14								
15	4E	CC	17	D9	DISPLAY	STRA,R0	SAVEREG	ROLL IS STOPPED; HE WANTS
16	51	76	60			PPSU	FLAG,II	HIS COUNT, SO SAVE THE REGS
17	3	3F	1E	A9		BSTA,UN	SAVREG	AND SET FLAG LED AND INTPT
18	6	D5	07			WRTE,R1	PORT7	INHIB. NOW SPLIT HIS ROLL
19	8	01				LODZ	R1	INTO 2 NIBLS, AND STORE IN
20	9	BB	F4			ZBSR	*DISLSD	THE ''ROLL'' MESSAGE.
21	B	C8	19			STRR,R0	ROLMES+6	
22	D	C9	18			STRR,R1	ROLMES+7	THEN SET UP THE DISPLAY
23	5F	05	00		DISPLAGN	LODI,R1	<DATABL-1	CONSTANTS FOR
24	61	06	6F			LODI,R2	>DATABL-1	HIS ROLL

THE PROGRAM (CONTINUED)

The program particulars are further defined in the listing provided at the bottom of these pages.

PREPARATION FOR PLAY:

Place the INSTRUCTOR 50 at the rear-center of a 4 x 4-foot card table, facing towards the center of the table.

Place the reproduced (enlarged) version of the 'Table' in front of the INSTRUCTOR 50, facing the front of the card table. Power up the INSTRUCTOR 50.

In FAST MEMORY PATCH mode, load the code for "ODDS" into the INSTRUCTOR 50.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM ODDS (VER1) | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF DISPLAY ROUTINE, ALSO INTERRUPT SERVICE. | SHEET 3

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	0063	07	01			LODI,R3	1	AND SHOW IT
2	5	BB	E6			ZBSR	*USRDSP	TO HIM. WHEN HE DEPRESSES
3	7	74	20			CPSU	II	INTPT KEY, ALL PAYOFFS ARE
4	9	DB	7E			BIRR,R3	$	MADE, AND HE'S READY TO
5	B	76	20			PPSU	II	PLACE NEW BETS.
6	D	1B	70			BCTR,UN	DISPLAGN	UNTIL THAT TIME, DISPLAY
7								HIS ROLL.
8	6F	00			*FILLER	CODE		
9								
10	70	13	15	11	ROLMES	RES	8	MESSAGE: 'ROLL = XX'
11	3	11	16	17				
12	6	XX	XX					
13								
14	78	1B	00	12	OVER	RES	8	MESSAGE: 'YOU HOPE'
15	B	17	14	00				
16	7E	10	0E					
17								
18	80	12			INTSVC	SPSU		FETCH PSU, AND CLEAR THE
19	1	A4	61			SUBI,R0	H'61'	FLAG, II, AND SUBTRACT 1
20	3	92				LPSU		FROM THE STACK POINTER;
21	4	0C	17	D9		LODA,R0	SAVEREG	NOW RESTORE THE PREV. SAV'D
22	7	3F	1E	B3		BSTA,UN	RESTR0	REGS.
23	A	B4	80			TPSU	SENS	WAIT UNTIL PLAYER DEPRESSES
24	8C	98	7C			BCFR,EQ	$-2	SENSE, THEN DELAY 1/2 SEC
2								
3								
4	8E	FB	7E			BDRR,R3	$	(A SUITABLE TIME)
5	90	FA	7C			BDRR,R2	$-2	THEN GO ROLL THE
6	0092	1F	00	28		BCTA,UN	ROLLING	RANDOM #'S AGAIN.

Toggle the INSTRUCTOR 50's switches as follows:

I/O Mode Switch	EXTENDED I/O PORT 07
Interrupt Address Sw.	INDIRECT
AC Line / Keyboard Interrupt Selection slide switch	KEYBOARD
I/O input switches (8)	any position

Play as described in the introduction.

BEAT THE ODDS - VARIATION 2*

INTRODUCTION AND PLAY:

Variation 2 of "BEAT THE ODDS" provides for enhancement of the visual display effects associated with this game. The players are told when to place their bets on the playing 'table'. Also the actual computation of the random number is preceded by a ¼ second display: "ROLL =". This is followed immediately by the 8 I/O LEDs flashing at high speed to indicate number generation.

There are slight changes to the rules of play; these include:

1. Depress RST and INT .....*to start the game. The promt: "PLACE BET" is immediately displayed.*

2. Depress any HEX KEY.....*to close placing of bets. The message "ROLL =" is displayed for ¼ second, followed by random number calculation.*

3. Depress SENS .....*to stop the number calculation 'roll'. The current number is displayed:*
   * *in binary on the I/O LEDs.*
   * *in Hex code on the Hex Display.*

   Make bet payoffs at this time.

4. Depress INT .....*to prompt for placing new bets.*

5. Repeat steps 3, 4, & 5 ....*until the message "YOU HOPE" is displayed*

Control switch settings are unchanged.

* Contained as File 5 on the INTRODUCTORY CASSETTE.

BEAT THE ODDS - VARIATION 2

PROGRAM CHANGES:

The LODI,R3 Ø instruction (LINE 3; SHEET 1 OF 'ODDS' VERSION 2 -- BELOW) conditions the INSTRUCTOR 50's Monitor USER DISPLAY routine to display the message "PLACE BET" until the operator depresses a HEX or FUNCTION key. For ease of play, the rules specify depression of a HEX key only.

The LODI,R3 H'7F' instruction in line 18 conditions the Display routine for a 1/4 second timeout, with auto-exit to 'roll' for the random number.

VARIATION 2

ROUTINE INTSVC | START ADDR Ø8Ø | PART OF PROGRAM ODDS (VER2) | 2650 PROGRAMMING FORM

DESCRIPTION VARIATION TO PROVIDE 'PLACE BET' & 'ROLL' PROMPTS FROM LOCATION '8Ø'. OVERLAYS ORIGINAL ODDS PROGRAM WITH THE FOLLOWING. | SHEET 1

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1					*NOTE THE	ADDRESSES SHOWN FOR MODI	FICATION OF 'ODDS' VERSION 1	
2					**********	****		
3	ØØ8Ø	07	00		INTSVC	LODI,R3	Ø	RULES AND ODDS FOR THIS
4	2	05	00		PLACE	LODI,R1	<PLACEBET-1	VERSION HAVE MINOR CHANGES.
5	4	06	8D			LODI,R2	>PLACEBET-1	SEE EXPLANATION PROVIDED
6	6	BB	E6			ZBSR	*USRDSP	ABOVE THIS LISTING.
7	8	12				SPSU		
8	9	A4	61			SUBI,R0	H'61'	
9	B	92				LPSU		
10	8C	1B	09			BCTR,UN	ROLLBET	
11								
12	8E	10	11	0A	PLACEBET	RES	8	MESSAGE: 'PLACE BET'
13	91	0C	0E	0B				DEPRESS ANY HEX OR FUNCTION
14	4	0E	87					KEY TO START ROLL.
15								
16	96	XX			ROLLTIME	RES	1	HI-SPEED ROLL TIME CONST.
17								
18	97	07	7F		ROLLBET	LODI,R3	H'7F'	DISPLAY FOR HI-SPEED ROLL
19	9	CB	7B			STRR,R3	ROLLTIME	
20	B	05	00			LODI,R1	<ROLL=-1	SET UP CONSTANTS
21	D	06	AD			LODI,R2	>ROLL=-1	
22	9F	BB	E6			ZBSR	*USRDSP	& SHOW ROLL (CURRENT)
23	A1	0B	73			LODR,R3	ROLLTIME	PULL ROLLTIME, AND SEE IF
24	ØA3	FB	74			BDRR,R3	ROLLBET+2	CURRENT ROLL DSPL OVER.

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM ODDS (VER2) | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF MODIFIED HIGH SPEED ROLL & DISPLAY. | SHEET 2

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØA5	0C	17	D9		LODA,R0	SAVREG	WHEN COMPLETE, GET PREV.
2	8	3F	1E	B3		BSTA,UN	RESTR0	SAVED RØ & OTHER REGS,
3	ØAB	1F	00	28		BCTA,UN	ROLLING	THEN EXIT TO ROLL UP ANO-
4								THER RANDOM NUMBER.
5								
6	ØAE	13	15	11	ROLL=	RES	8	MESSAGE: 'ROLL = XX'
7	B1	11	16	97				
8	ØØB4	97	97					
9					*FLASHES	FOR 1/2	SECOND, THEN PROGRAM EXITS TO ROLL FOR ODDS.	

ELECTRONIC SLOTMACHINE

INTRODUCTION:

Program "SLOTMACHIN" provides a fastmoving means to 'play the slots', very much as a typical slotmachine is organized for use in Nevada, and now in New Jersey. The HEX DISPLAY facilities are used to sequence a series of messages as follows:

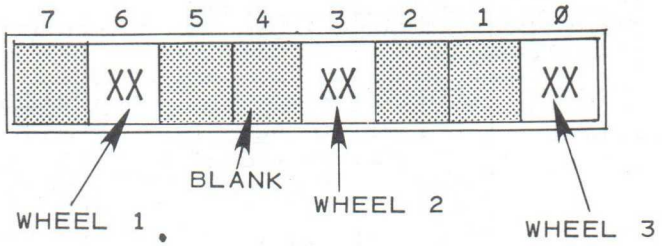
"R O T A T E"

"Y O U H O P E"

"I T I S A"

"J A C - P O T"

This is followed by rolling of the 3 slotmachine 'wheels' at an adjustable rate of speed. The 'wheels' are displayed as follows, and stop in sequence; wheel 1 first, followed by wheel 2, then with wheel 3.



XX = VARYING IMAGES FOR EACH WHEEL'S DISPLAY DEFINED ON PAGE 6-78.

ROUTINE INITIAL | START ADDR ØØØØ | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION IND ADDR CALLS & INIT. MESSAGE DISPLAY FOR SLOTMACHINE. | SHEET 1

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØØØ	1B	0E		SLOTMACH	BCTR,UN	INITIAL	BRCH TO PROGRM STRT ON RST
2	2	Ø1	4Ø		DISMES	ACON	H'0140'	
3	4	01	E9		DISROL	ACON	H'01E9'	
4	6	01	B0		ZERO	ACON	H'01B0'	
5	8	01	BA		ODDSDSP	ACON	H'01BA'	SUBROTINE IND.ADDR CALLS
6	A	01	70		SHOWJAC	ACON	H'0170'	
7	C	00	00		*FILLER	CODE		
8	ØE	01	30		EVENTSET	ACON	H'0130'	
9								
10	1Ø	20			INITIAL	EORZ	RØ	CLR RØ, THEN SET ALL PSW
11	1	92				LPSU		AT ZERO
12	2	93				LPSL		THEN GO INITIALIZE THE
13	3	76	20			PPSU	II	WHEEL INDEX CONSTANT
14	5	BB	8E		LOSER	ZBSR	*EVENTSET	ON RETURN, SET UP EVENTS, &
15	7	CF	01	A0		STRA,R3	EVENTS	MAKE A RAPID DSPLY OF THE
16	A	06	7F		MESSAGES	LODI,R2	>ROTATE-1	FOLLOWING MESSAGES:
17	C	05	01			LODI,R1	1	(RUNDSPX)X1 'ROTATE'
18	1E	BB	82			ZBSR	*DISMES	'YOU HOPE'
19	2Ø	06	AF			LODI,R2	>YOUHOPE-1	'IT IS A'
20	2	05	03			LODI,R1	3	(RUNDSPX)X3 'JACKPOT'
21	ØØ24	BB	82			ZBSR	*DISMES	YOU CAN ALTER MESSAGE SPEED
22								BY CHANGING RUNDSPX FOR
23								EACH MESSAGE.
24								

A JOHN GARCEAU PROGRAM

INTRODUCTION: (CONTINUED)

The display is fixed for 2 full seconds after all wheels have ceased rotation to permit the viewer to see the combination rolled. Depending on the combination rolled, the player will then see the following messages:

IF THERE IS A WINNING COMBINATION:

" P A Y - O F F " .....*a 1-second message to tell the player that he has won, and that the odds for payoff will be displayed.*

" X X X - T O 1 ".....*a 2-second message to indicate the exact payoff odds calculated by the INSTRUCTOR 50.*

IF THERE IS A LOSING COMBINATION:

" Y O U L O S T " .....*is displayed for about one second to inform the player that he lost this time.*

Then, depending on certain programmable functions (easily altered by the operator), SLOTMACHIN will wait for the operator to depress RST, or automatically display the message: "ROTATE, YOU HOPE IT IS A JAC-POT", as described previously.

ROUTINE ROTATE1 | START ADDR Ø32 | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION FINISH 'INITIAL' , THEN ROLL THE 'WHEELS' OF THE MACHINE | SHEET 2

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ26	06	B7			LODI,R2	>ITISA-1	
2	8	05	02			LODI,R1	2	(RUNDSPX)X2
3	A	BB	82			ZBSR	*DISMES	
4	C	06	A7			LODI,R2	*>JACKPOT-1	
5	2E	05	08			LODI,R1	8	(RUNDSPX)X8
6	30	BB	82			ZBSR	*DISMES	CONCL. INIT. MSG DISPLAY
7					*THIS NEXT ROUTINE CONTROLS SELECTION OF, AND ROTATION OF 3,			
8					*2, OR 1	WHEELS, OR EXIT TO SHOWW		3 WHEELS STOPPED.
9	32	0F	01	A0	ROTATE1	LODA,R3	EVENTS	GET ROLL SELECT. PTR, THEN
10	5	BF	00	3A	ROTATE3	BSXA	ROLL,EVENTS	INDX INTO ROLL CONTRL TABLE
11	8	1B	0C			BCTR,UN	STOPWHEEL	ON RTRN, GO SEE IF TIME TO
12	A	1F	00	60	ROLL	BCTA,UN	SHOW3	STOP. SHOW 3 WHLS STOPPED
13	3D	1F	01	0C		BCTA,UN	ROLL3	#S 1 & 2 STOPPED; 3 ROTATES
14	40	1F	01	06		BCTA,UN	ROLL23	# 1 STOPPED; 2&3 ROTATE.
15	3	1F	01	00		BCTA,UN	ROLL123	WHEELS 1,2, & 3 ROTATE.
16					*CODE FROM ADDR 3A - 43 IS A 4-ENTRY TABLE.....			
17								
18					*ON FOLLOWING SHEET, THERE ARE 2 ALTERNATIVES FOR ROUTINE			
19					*'STOPWHEEL', STARTING AT LOCATION '0046'. THE FIRST PROVIDES			
20					*A TIMED AUTOMATIC STOP OF EACH WHEEL. THE SECOND REQUIRES			
21					*THE OPERATOR TO DEPRESS SENS EACH TIME HE WISHES A WHEEL			
22					*TO STOP.			
23								
24								

INTRODUCTION: (CONTINUED)

A programmable option, found in the listing, permits the player to stop each 'wheel's' rotation by depressing the SENS key. Your author recommends this option as a means of further generating a "random number sequence" for each wheel's stopping point.

In addition, all message display times are individually adjustable; these will be precisely defined in the next section of this description.

If the winning combination is 3 bars (▭ ▭ ▭), the player wins the "JACKPOT". The INSTRUCTOR 50 'goes berserk' by displaying the following messages 4 times while simultaneously flashing the I/O LEDs in different patterns:

"J A C - P O T"

"P A Y - O F F"

"2 5 0 T O 1"

The 4-event JACKPOT sequence lasts approximatlely 14 seconds. The number of events may be increased (or decreased) by minor program change.

ROUTINESTOPWHEEL | START ADDR Ø46 | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION 2 ALTERNATIVES TO STOP WHEELS: # 1: AUTOMATIC
2: PLAYER DEPRESSES SENS KEY | SHEET 3

	LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
	1	Ø046	08	17		STOPWHEL	LODR,R0	TIMEOUT	ALTERNATIVE 1:
	2	8	A4	01			SUBI,R0	1	TAKE 1 FRM TIMEOUT XT. AND
	3	A	C8	13			STRR,R0	TIMEOUT	STORE IT. IS THE TIMEOUT
	4	C	E4	C0			COMI,R0	H'C0'	DOWN TO H'C0'? YES. DECR
ALT. 1 AUTO	5	4E	1C	01	A2	DECEV	BCTA,EQ	*DECEVENT	EVENT. NO! REPEAT MAYBE..
	6	51	E4	70			COMI,R0	H'70'	TIMEOUT 48? YES. DECR EVNT
	7	3	18	FA			BCTR,EQ	*DECEV+1	NO! GO REPEAT MAYBE. WELL,
	8	5	E4	00			COMI,R0	0	IS TIME = Ø YET? YES! DECR
	9	7	18	F6			BCTR,EQ	*DECEV+1	EVENT; TIME TO SHOW 3 WHLS.
	10	9	1B	57			BCTR,UN	ROTATE1	NO! NONE OF THESE; GO REPEAT
	11	5B-5E	00	00	00	*FILLER	CODE.		ROLL.
	12	Ø05F	XX			TIMEOUT	RES	1	VARIABLE 0 < XX < H'FC'
	13								ALTERNATIVE 2:
	14	Ø046	B4	80		STOPWHEL	TPSU	SENS	DID PLAYER DEPRESS SENS?
	15	8	18	02			BCTR,EQ	WAIT	NO! ROTATE WHEELS AGAIN.
	16	A	1B	66			BCTR,UN	ROTATE1	YES! WAIT TIL HE LIFTS FIN-
	17	C	12			WAIT	SPSU		GER OFF THE SENS KEY.....
ALT. 2 DEPRESS SENS	18	D	44	80			ANDI,R0	SENS	IF SENS IS STILL DOWN,
	19	4F	58	7B			BRNR,R0	WAIT	WAIT SOME MORE TIME. FIN-
	20	51	3F	01	A2		BSTA,UN	DECEVENT	GER'S UP! GO FIX SENS.
	21	4	A7	03			SUBI,R3	3	ON RETURN, DECR EVENTS AND
	22	6	CF	01	A0		STRA,R3	EVENTS	STORE, THEN GO RETURN TO
	23	59	1B	5A			BCTR,UN	ROTATE3	THE ROTATE TABLE.
	24	5B-5F	00	00	00	*FILLER	CODE.		

INTRODUCTION: (CONTINUED)

LIMITATION

IT IS ALMOST IMPOSSIBLE TO GENERATE A TOTALLY RANDOM RATE OF WHEEL ROLL, AND IMAGE SELECTION WITHIN PROGRAM ''SLOTMACHIN''. WITH THE SELECTIONS PROVIDED IN THIS PROGRAM, YOU'LL FIND THAT THE PAYOFF RATE IS ABOUT 1.7 TO 1 IN YOUR FAVOR. TO MAKE THE GAME MORE INTERESTING, AND ENSURE A MODERATE PAYOFF IN THE 'HOUSE'S' FAVOR, YOU MAY WISH TO MODIFY YOUR PROGRAM. WE WILL DESCRIBE THIS LATER. YOUR AUTHOR HAS ALSO DETERMINED, AFTER MANY HOURS OF OPERATION, AND BY STATISTICAL ANALYSIS GENERATED BY HIS INSTRUCTOR 50, THAT PAYOFFS ARE CONTROLLED IN REAL SLOT MACHINES BY HOUSE ADJUSTMENT OF THE MACHINE'S INTERNAL CAMS. THUS, A TRULY RANDOM SELECTION OF PAYOFF IMAGES APPEARS TO BE IMPOSSIBLE AT GAMING HOUSES. AND CERTAINLY, IT IS IMPOSSIBLE TO PROGRAM THE COMPUTER TO LIE DELIBERATELY IN THE 'HOUSE'S' FAVOR, SIMPLY BECAUSE THE COMPUTER ITSELF IS AN EMINENTLY LOGICAL MACHINE.

THIS ANALYSIS IS BASED, HOWEVER, UPON USE OF 16-IMAGE WHEELS, WITH ALL CHARACTERS IN THE SAME SEQUENCE. TYPICAL CASINO MACHINES CONTAIN 20-IMAGE WHEELS, WITH LESS CHERRIES IN WHEEL 1. THIS PRACTICE SUBSTANTIALLY DECREASES THE ODDS FOR PAYOFF.

ROUTINE SHOW3 | START ADDR Ø6Ø | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION THIS ROUTINE DISPLAYS ALL 3 WHEELS AFTER THEY HAVE STOPPED. 2 SEC SHOW; THEN START DECODE FOR WINNING OR LOSING COMBINATIONS | SHEET 4

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø06Ø	06	08		SHOW3	LODI,R2	8	SET XTANTS TO MOVE 8-BYTE
2	2	0E	60	F7		LODA,R0	WHMES-1,R2	WHL MESSG INTO 'BAR'LOC'NS
3	5	CE	57	A0		STRA,R0	BAR,R2,-	WHEN THE MOVE
4	8	5A	78			BRNR,R2	MOVIT	IS COMPLETED, DIS-
5	A	06	9F			LODI,R2	>BAR-1	PLAY THE 'BAR' MESSAGE FOR
6	C	05	08			LODI,R1	8	2 SECS, THE GO ON TO
7	6E	BB	82			ZBSR	*DISMES	COMPARE & DECODE
8								ROUTINE.
9	7Ø	0D	00	F9	COMPARE	LODA,R1	WHMES+1	ON ENTRY, FETCH WHEEL 1.
10	3	E5	0C			COMI,R1	H'0C'	IS IT A CHERRY? NO! GO ADD
11	5	9C	01	60		BCFA,EQ	ADD23	WH 2 + 3 TO WHL 1. YES!!
12	8	8D	00	FC		ADDA, R1	WHMES+4	ADD WHL 2 TO WHL 1.
13	B	E5	18			COMI,R1	H'18'	ARE THERE 2 CHERRIES? NO!
14	7D	9C	00	B3		BCFA,EQ	PAY2/1	GO PAY 2:1 ODDS. YES!! SO
15	8Ø	8D	00	FF		ADDA,R1	WHMES+7	GET WHL 3 TO SEE IF THERE
16	3	E5	24			COMI,R1	H'24'	ARE 3 CHERRIES. YES!! GO
17	5	1C	00	BE		BCTA,EQ	PAY10/1	PAY 10:1 ODDS. NO: JUST 2
18	8	1F	00	CA		BCTA,UN	PAY5/1	CHERRIES; PAY 5:1 ODDS.
19	B	E5	21		3BELLS	COMI,R1	H'21'	ON ENTRY, ALL 3 WHEELS ARE
20	8D	1C	00	D0	20REP	BCTA,EQ	PAY20/1	ADDED. ARE THERE 3 BELLS?
21	90	E5	AB			COMI,R1	H'AB'	YES! GO PAY 20:1 ODDS
22	92	18	FA			BCTR,EQ	*20REP+1	NO. ARE THERE 2 BELLS & BAR
23								YES!! GO PAY 20:1
24								

USER OPERATION - SET UP:

After loading the program (2 files if via Read Cassette mode), position the INSTRUCTOR 50's switches as follows:

I/O Mode Selection Switch	EXTENDED I/O PORT Ø7
I/O input Toggle switches	any position
Interrupt Selection switch	either position
AC Line /Keyboard Interrupt Selection Slide Switch	KEYBOARD

STOP WHEEL ROTATION SELECTION OPTIONS:

Refer to sheet 3 of the listing for program "SLOTMACHIN". If you loaded "SLOTMACHIN" from cassette tape, alternative 1 of routine 'STOPWHEL' is currently in the INSTRUCTOR 50's memory at addresses H'0046' through '005A'. Alternative 1 provides AUTOMATIC ROTATION AND STOP of each of the 3 'character wheels' used by the 'slotmachine'. If you desire MORE PLAYER ACTION, use the INSTRUCTOR 50's MEMORY FAST PATCH mode* to load the alternative version of routine 'STOPWHEL' into the same addresses. Alternative 2, starting at line 14 in the listing, requires the player to depress the SENS key to stop each wheel.

* page 6-8

ROUTINE COMPARE | START ADDR CONT. | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION FINISH UP THE VALID COMPARES FOR WINNINGS, THEN SET UP THE APPROPRIATE WINNING MESSAGE OR °YOU LOST° | SHEET 5

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ94	E5	33			COMI,R1	H'33'	ARE THERE 3 LEMONS?
2	6	1C	00	D6	18REP	BCTA,EQ	PAY18/1	YES! GO PAY 18:1
3	9	E5	B7			COMI,R1	H'B7'	NO. 2 LEMONS & A BAR?
4	B	18	FA			BCTR,EQ	*18REP+1	YES: PAY 18:1
5	D	E5	30			COMI,R1	H'30'	NO...3 PLUMS?
6	9F	1C	00	E3	14REP	BCTA,EQ	PAY14/1	YES!! GO PAY 14:1 ODDS
7	A2	E5	B5			COMI,R1	H'B5'	NO... 2 PLUMS & A BAR?
8	4	18	FA			BCTR,EQ	*14REP+1	YES: PAYOFF IS 14:1
9	6	E5	BF			COMI,R1	H'BF'	NO. ARE THERE 3 BARS FOR
10	8	1C	00	E9		BCTA,EQ	JACKPOTX	A JACKPOT? YES!!!!! TELL IT
11	B	06	87		HELOST	LODI,R2	>YOULOST-1	TO HIM NOW. NO; NO MORE WAY
12	D	05	08			LODI,R1	8	TO WIN. TELL HIM HE LOST.
13	AF	BB	82			ZBSR	*DISMES	SHOW FOR 2 SEC, THEN GO RE-
14	B1	9B	15			ZBRR	LOSER	INIT. EVENTS. LOTSA LUCK!!
15					*PAYOFF ROUTINES CONDITION PAYOFF ODDS MSGS PRIOR TO SHOW.			
16	ØØB3	BB	86		PAY2/1	ZBSR	*ZERO	ONENTRY, ENSURE ODDS=ØØ
17	5	04	02			LODI,R0	2	NOW SET 2:1 ODDS IN MSG
18	7	CC	17	9A		STRA,R0	XXXTO+1	AND GO SHOW IT. DO CLEANUP
19	A	BB	88		CLEANUP	ZBSR	*ODDSDSP	FUNCTIONS, THEN
20	C	9B	15			ZBRR	LOSER	GO RESTART THE GAME.
21	BE	BB	86		PAY10/1	ZBSR	*ZERO	GO ZERO THE ODDS MESSAGE
22	C0	04	01			LODI,R0	1	THEN SET A '1' FOR 10:1 &
23	2	CC	17	99	20X	STRA,R0	XXXTO+1	STORE IT IN THE 10S DIGIT
24								LOCATION

USER OPERATION - SET UP (CONTINUED)

MESSAGE DISPLAY TIME CONTROL - PROGRAMMABLE:*

The following single-byte program changes may be incorporated into the control program in order to vary display time of each of the following messages. In this way, you can "tailor" your game's operation to your requirements, or those with whom you are playing the game of SLOT MACHINE.

MESSAGE:	TIME CONTROL ADDRESS	CODE	CURRENT TIMEOUT	CHANGE COMMENT:
ROTATE	ØØ1D	'01'	¼ sec	change to '04' for 1 sec time
YOU HOPE	ØØ23	'03'	3/4 sec	
IT IS A	ØØ29	'02'	½ sec	
JACKPOT	ØØ2F	'08'	2 sec	
YOU LOST	ØØAE	'08'	2 sec	same comment as above.
PAY-OFF	Ø1BD	'04'	1 sec	
XXX TO 1	Ø1C3	'08'	2 sec	
JAC-POT	Ø17B	'04'	1 sec	"JACKPOT" MESSAGE
PAY-OFF	Ø185	'04'	1 sec	same comment as above but this is separate message control
250 TO 1	Ø18F	'06	1½ sec	used only when he hits JACKPOT.

* USE MEMORY DISPLAY & ALTER MODE.

ROUTINE PAYOFFS | START ADDR CONT. | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF PAYOFF MESSAGE SETUP BASED ON CONDITION OF THE 3 STOPPED WHEELS. | SHEET 6

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØC5	20				EORZ	R0	CLEAR R0,AND PUT THE Ø IN
2	6	C8	F0			STRR,R0	*CLEANUP-2	ODDS LSD. THEN CLEANUP.
3	8	1B	70			BCTR,UN	CLEANUP	
4	A	BB	86		PAY5/1	ZBSR	*ZERO	CLEAR THE ODDS MESSAGE. ON
5	C	04	05			LODI,R0	5	RTN, SET FOR 5:1 ODDS AND
6	CE	1B	67			BCTR,UN	CLEANUP-3	GO CLEANUP.
7	D0	BB	86		PAY20/1	ZBSR	*ZERO	CLEAR ODDS; THEN SET A 2
8	2	04	02			LODI,R0	2	FOR 10S DIGIT, AND GO CLEAN
9	4	1B	6C			BCTR,UN	20X	UP AND DISPLAY.
10	6	BB	86		PAY18/1	ZBSR	*ZERO	CLEAR ODDS, THEN SET LSD TO
11	8	04	08			LODI,R0	8	AN '8' AND THE 10S DIGIT
12	A	CC	17	9A	18X	STRA,R0	XXXTO+2	TO A '1' FOR 18:1 ODDS
13	D	04	01			LODI,R0	1	THEN STORE
14	DF	C8	90			STRR,R0	*$+18	THEN GO DISPLAY
15	E1	1B	57			BCTR,UN	CLEANUP	AND CLEANUP.
16	3	BB	86		PAY14/1	ZBSR	*ZERO	CLEAR ODDS DSPLAY. SET LSD
17	5	04	04			LODI,R0	4	TO '4' AND LOOP BACK TO 18X
18	7	1B	71			BCTR,UN	18X	FOR STORE & DISPLAY.
19	9	04	02		JACKPOTX	LODI,R0	2	HE'S WON THE JACKPOT !!!!!
20	B	CC	17	98		STRA,R0	XXXTO	SO SET UP THE DIGITS
21	EE	04	05			LODI,R0	5	FOR A 250:1 ODDS DISPLAY
22	FØ	CC	17	99		STRA,R0	XXXTO+1	THEN EXIT TO SHOW.....
23	3	20				EORZ	R0	THE JACKPOT
24								

USER OPERATION - SET UP: (CONTINUED)

ROLL DISPLAY CONTROL:

The 3-wheel roll display routine is programmed to run at high speed. The following controls are offered, and may be changed:

ØØ1EA '03' ...*current programmed wheel rotation permits distribution of 256 image changes among the 3 wheels in 5½ seconds. Increase of H'03' adds 5 seconds to wheel rotation time. Note: These times are predicated on being in automatic wheel rotation mode, described previously.*

ØØ4D 'C0' ...*first wheel stops after 64 images, distributed among 3 wheels, have been exercised.*

ØØ52 '70' ...*2nd wheel stops after 144 images, distributed among 2 wheels, have been exercised.*

NOTE: YOU MAY MODIFY THESE BYTES, PROVIDED CONTENT OF LOCATION '0052' IS LESS THAN CONTENTS OF LOCATION '004D' AUTO ROTATION ONLY!!!!!!!

ROUTINE ROLL123	START ADDR BELOW	PART OF PROGRAM SLOTMACHIN	2650 PROGRAMMING FORM
DESCRIPTION ROLL123 PROVIDES FOR SELECT & ROLL OF SLOTMACHINE WHEELS THE ENTRY POINT IS DETERMINED BY THE ROLL CONTROL ROUTINE AT ADDR '46'			SHEET 7

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØF4	C8	E5			STRR,R0	*$-25	AFTER SHOWING IT,
2	6	BB	8A			ZBSR	*SHOWJAC	THIS FINIHES ODDS CALCUL.
3					*THIS ROUTINE PROCESSES THE ROLL OF 3, 2, OR 1 WHEELS, AND			
4					*PSEUDO # GENERATION FOR INDEXING			INTO THE POSSIBLE COMPLE-
5					*MENT OF EACH WHEEL'S CHARACTERS.			ROTATING WHLS DISPLAY:
6	ØØF8	17	XX	17	WHMES	RES	8	1 4 7
7	B	17	XX	17				XX XX XX
8	ØØFE	17	XX					WHL1 WHL2 WHL3
9								
10	Ø1ØØ	3B	13		ROLL123	BSTR,UN	FETCHWH	ONENTRY, ROLL ALL 3 WHLS,
11	2	C8	75			STRR,R0	WHMES+1	THEN STR. 1ST INDXD WHEEL
12	4	BB	84			ZBSR	*DISROL	CHAR IN POSITION 1..SHOW!!
13	6	3B	0D		ROLL23	BSTR,UN	FETCHWH	ON ENTRY, ROLL WHEELS 2 & 3
14	8	C8	72			STRR,R0	WHMES+4	SAME COMMENT AS ABOVE.
15	A	BB	84			ZBSR	*DISROL	
16	C	3B	07		ROLL3	BSTR,UN	FETCHWH	ON ENTRY, ROLL LAST WHEEL.
17	1ØE	C8	6F			STRR,R0	WHMES+7	SAME COMMENT AS ABOVE
18	11Ø	BB	84			ZBSR	*DISROL	THEN RETURN TO ROLL CONTROL
19	2	17				RETC		ROUTINE.
20	3	XX			RANNUM	RES	1	SAVED RANDOM # FOR INDX GEN
21	4	XX			WTBLX	RES	1	WHEEL TABLE INDEX VARIABLE.
22								ONENTRY, EXIT TO PROVIDE
23	115	3B	A2		FETCHWH	BSTR,UN	*FIXNBR	BETTER RAND # GEN BY VARYNG
24	7	88	7A			ADDR,R0	RANNUM	THE R # CONST, ADD & SAVE.

USER OPERATION - SET UP: (CONTINUED)

After all the 'wheels' have stopped, the 3 resultant "images" are displayed for 2 seconds prior to testing for winning combinations. The timeout for this display is:

ØØ6C '08'.....*equals 2 seconds. Increase to lengthen time. Decrease to shorten time as previously described.*

AUTO/MANUAL NEW GAME OPTION:

You have the capability to select options for start of a new 'roll' of the wheels. These are provided in sheets 11 and 12 of the listing. The 1st alternative, included in the taped version of SLOTMACHINE, provides an automatic restart, once the payoff is displayed. The second alternative requires the player to depress the RST key to restart. The following table provides a listing of the necessary codes to accomplish these options:

ADDR	CODE	INSTRUCTION	MODE	COMMENT
019E	9B 00	ZBRR SLOTMACHIN	Auto	...*Restarts after Jackpot displ.*
019E	1B 50	BCTR,UN SHOWJAK	Man'l	...*Show Jackpot message until the player depresses the RST key.*

ROUTINE COMMENTED | START ADDR | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION FETCHWH (CONT.) ALSO EVENT SET FOR NEW ROLL, AND FIX FOR BETTER RANDOM NUMBER GENERATION | SHEET 8

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø119	C8	78			STRR,R0	RANNUM	AFTER SAVE, BREAK THE RE-
2	B	BB	F4			ZBSR	*DISLSD	SULT INTO NIBBLES, & ADD
3	D	81				ADDZ	R1	UP FOR NEW XTANT. ADD TO
4	1E	88	74			ADDR,R0	WTBLX	CURR. WHEEL TABL INDEX THEN
5	12Ø	44	0F			ANDI,R0	H'0F'	MASK OFF MS 4 BITS AND
6	2	C8	70			STRR,R0	WTBLX	STORE AWAY. NOW SET THE
7	4	C1				STRZ	R1	RESULT IN INDEX REG R1, AND
8	5	0D	61	D8		LODA,R0	WHTBL,R1	FETCH A NEW ENTRY FROM THE
9	28	17				RETC,UN		WHEEL TABLE, THEN RETURN
10								WITH WHL TABLE ENTRY IN R1.
11	Ø129	00	00	00	*			
12	C	00	00	00	* FILLER CODE			FILLER CODE
13	12F	00			*			
14								NEW ROLL COMING, SO SET EV-
15	130	07	09		EVENTSET	LODI,R3	9	ENTS FOR ROLL123 CONTROL.
16	2	06	FF			LODI,R2	H'FF'	THEN REINIT. ROLL TIMEOUT,
17	4	CE	00	5F		STRA,R2	TIMEOUT	(USED IN AUTO STOP ONLY) &
18	137	17				RETC,UN		RETURN TO CALLING PROG.
19	8	00						FILLER CODE
20	139	01	C8		FIXNBR	ACON	H'01C8'	IND. ADDR TOROUTINE WHICH
21								GENERATES A MORE RANDOM
22								INDEX CONSTANT
23								
24	13B	00	00					FILLER CODE

USER OPERATION: (CONTINUED)

```
01C6    1B 72  BCTR,UN ODDSDSP   Man'l ..show winning odds until the
                                          player depresses the RST key.

01C6    17     RETC,UN           }
01C7    00     *filler code      }  AUTO ...restart after winning odds
                                          are displayed
```

USER OPERATION - SLOT MACHINE PLAY:

If you chose the AUTO mode for SLOT MACHINE operation, you need only to depress the RST key, and the INSTRUCTOR 50 will perform its programmed sequence for hours on end (if you let it). AUTO mode is the best for single player operation. Incidently, this mode provides a useful vehicle for your setup and program tailoring operations previously described.

MANUAL mode configures the program so that you, and your acquaintences, can place bets "against the house"(INSTRUCTOR 50) before the 'wheels are rolled'. In MANUAL mode, you need only depress RST once each time you wish the wheels to spin. The INSTRUCTOR 50 completes one entire sequence, then displays the latest winning odds until you depress RST once again. However, if a LOSING combination turns up, the sequence loops back to commence a new roll. This also may be changed as shown on the next page.

ROUTINE DISMES	START ADDR 0140	PART OF PROGRAM SLOTMACHIN	2650 PROGRAMMING FORM
DESCRIPTION THIS SUB CONTROLS DISPLAY TIME OF ALL MESSAGES EXCEPT THE DECR WHEEL ROLL -- ACCESSES 8X8 BYTE MESSAGE TABL IN SMI RAM: 1780-17BF.			SHEET 9

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø13D	XX			>MESLOC-1	RES	1	CURRENT MSG LSBYTE OF ADDR
2	E	XX			DSPLDLY	RES	1	DISPLAY DELAY CONSTANT
3	F	XX			RUNDSPX	RES	1	RUN TIME CONST. X 1/4 SEC
4								EQUALS RUN TIME OF DISPLAY.
5	Ø14Ø	CA	7B		DISMES	STRR,R2	>MESLOC-1	ON ENTRY, STORE >MESLOC AD-
6	2	07	50			LODI,R3	H'50'	DRS, & INIT DSPLDLY
7	4	C9	79		REPEATM	STRR,R1	RUNDSPX	STORE RUN TIME CONSTANT, &
8	6	CB	76		DSPAGN	STRR,R3	DSPLDLY	DECREMENTED DISPLAY DELAY
9	8	05	17			LODI,R1	<MESLOC-1	SET UP MESSAGE ADDR -- HIGH
10	A	0A	71			LODI,R2	>MESLOC-1	BYTE, THEN LOW BYTE, THEN
11	C	BB	E6			ZBSR	*USRDSP	GO SHOW SELECTED MESSAGE.
12	4E	0B	6E			LODR,R3	DSPLDLY	ON RET, GET DISPL DLY. IS
13	5Ø	E7	01			COMI,R3	1	IT DECR. TO 1 YET? YES!! GO
14	2	18	02			BCTR,EQ	CONTDSP	SEE IF DSPL TO CONTIN. NO!
15	4	FB	70			BDRR,R3	DSPAGN	DECR. & GO DISPLAY AGAIN.
16	6	09	67		CONTDSP	LODR,R1	RUNDSPX	GET RUN TIME CONST. & DECR-
17	8	F9	01			BDRR,R1	$+3	EMENT. IS IT ZERO YET? YES!
18	A	17				RETC,UN		MSG DSPLAY FINIHED; EXIT!!!
19	B	0B	66			LODR,R3	REPEATM-1	NO! REINIT. DSPLDLY, THEN
20	15D	1B	65			BCTR,UN	REPEATM	LOOP TO DISPLAY SAME MES-
21								SAGE AGAIN.
22	15F	00			*FILLER	CODE		FILLER CODE
23								
24								

USER OPERATION - SLOT MACHINE PLAY: (CONTINUED

To stop the sequence after a losing combination is encountered, make the following modification:

ADDR. 'ØØB1': CHANGE '9B' TO '40' . A HALT INSTRUCTION

After collecting the losses for 'the house', have all players place new bets, then depress RST for the next sequence.

You may also wish to select the MANUAL STOP WHEEL ROTATION option involving use of the SENS key. The modifying instructions are contained in sheet 3 of the listing, and this procedure is described on page 5.

SEQUENCE:

Once started, the sequence is automatic. In order, the following messages are displayed:

"ROTATE" "YOU HOPE" "IT IS A" "JAC-POT"

ROUTINE AS SHOWN | START ADDR | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION 'ADD23' IS FIX TO ADD WHLS 2 & 3 TO WHL 1 IF NO CHERRY IN WHL 1. 'SHOWJAC' IS SPECIAL SUBR TO SHOW THE WINNER HIS GIANT JACKPOT, | SHEET 10

LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	Ø16Ø	8D	00	FC	ADD23	ADDA,R1	WHMES+4	ADD WHEELS 2 AND 3 TO WHEEL
2	3	8D	00	FF		ADDA,R1	WHMES+7	1; RESULT IN R1; THEN GO
3	6	1F	00	8B		BCTA,UN	3BELLS	SEE IF 3 BELLS, ETC.
4	169-E	00	00	00	*FILLER	CODE - 6	ADJACENT LOCATIONS.....	
5	16F	XX			SCREAM4	RES	1	LOOP CNT CNTRL - 4 TIMES
6	17Ø	04	04		SHOWJAK	LODI,R0	4	COMMENT SAME AS FOR MESSAGE
7	2	C8	7B		AGAIN	STRR,R0	SCREAM4	SETUP PREVIOUSLY DESCRIBED.
8	4	04	55			LODI,R0	H'55'	ALSO WRITE SELECTED # TO
9	6	D4	07			WRTE,R0	PORT7	EXT. I/O PORT 7 TO SIMULATE
10	8	06	A7			LODI,R2	>JACKPOT-1	INSTRUCTOR SO GOING BERSERK
11	A	05	04			LODI,R1	4	WITH JOY WHEN THE WINNER
12	C	BB	82			ZBSR	*DISMES	HITS THE JACKPOT
13	17E	04	AA			LODI,R0	H'AA'	
14	18Ø	D4	07			WRTE,R0	PORT7	NO FURTHER COMMENT ON THIS
15	2	06	8F			LODI,R2	>PAYOFF-1	
16	4	05	03			LODI,R1	3	
17	6	BB	82			ZBSR	*DISMES	
18	8	04	D2			LODI,R0	H'D2'	
19	A	D4	07			WRTE,R0	PORT7	
20	C	06	97			LODI,R2	>XXXTO-1	
21	18E	05	06			LODI,R1	6	
22	19Ø	BB	82			ZBSR	*DISMES	
23	192	04	2D			LODI,R0	H'2D'	
24	194	D4	07			WRTE,R0	PORT7	

SEQUENCE: (CONTINUED)

The 3 'wheels' are then rotated. The wheels, located in positions 6, 3, and Ø of the HEX DISPLAY, provide the following IMAGES, representing the characters you would view when a mechanical slot machine is operating:

CHARACTER:

CHERRY
BELL
PLUM
CHERRY
BAR
LEMON
CHERRY
PLUM
CHERRY
LEMON
PLUM
CHERRY
PLUM
BELL
CHERRY
LEMON

16 IMAGES IN ORDER ON EACH WHEEL.

IMAGE VIEWED:

BELL =
CHERRY =
LEMON =
PLUM =
BAR =

*** NOTE ALTERNATIVES ***

ROUTINE SEE BELOW | START ADDR | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION CONCLUSION OF 'SHOWJAK'. EVENT DECREM. IN AUTO STOP VERSION ONLY.... ALSO SUBR. 'ZERO' TO CLEAR ODDS DISPLAY | SHEET 11

	LINE	ADDRS	DATA BØ	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
	1	Ø196	FA	7E			BDRR,R2	$	PRECEDING SHEET HAS EXPLA-
	2	8	F8	7C			BDRR,R0	$-2	NATION.
	3	A	08	53			LODR,R0	SCREAM4	
	4	C	F8	54			BDRR,R0	AGAIN	
	5					**NOTE COMMENT:		ALTERNATIVE A:	LOOP UNTIL PLAYER DEPR. RST
	6					*	ALTERNATIVE B: AUTO RETURN		FOR NEW GAME.**********
***	7	19E	1B	50			BCTR,UN	SHOWJAK	ALTERNATIVE A... MANUAL
***	8	19E	9B	00			ZBRR	SLOTMACHIN	'' B AUTO
	9								
	10	Ø1AØ	XX			EVENTS	RES	1	ROLL TABL CONTRL 0<N<9
	11	1	00			*FILLER	CODE		
	12	1A2	0B	7C		DECEVENT	LODR,R3	EVENTS	ON ENTRY, SUBTRCT 3 FROM
	13	4	A7	03			SUBI,R3	3	CURRENT EVENT (ROLL TABLE
	14	6	CB	78			STRR,R3	EVENTS	INDEX). STORE NEW VALUE, &
	15	8	1F	00	35		BCTA,UN	ROTATE3	EXIT TO ACCESS ROLL CONTROL
	16								TABLE. (LOC. 46, ETC.)
	17	1AB-F	00	00	00	*FILLER	CODE		
	18								
	19	Ø1BØ	04	17		ZERO	LODI,R0	SPACECODE	MUST BLANK ODDS DISPLAY, SO
	20	2	06	03			LODI,R2	3	GET A SPACE CODE, AND SET
	21	4	CE	57	98	ANOTHER	STRA,R0	XXXTO,R2,-	IT INTO XXXTO THROUGH
	22	7	5A	7B			BRNR,R2	ANOTHER	XXXTO+2 . WHEN FINISHED,
	23	Ø1B9	17				RETC,UN		EXIT TO CALLING PROGRAM
	24								

SEQUENCE: (CONTINUED)

After all three wheels have stopped, the 3 characters are checked for winning combinations. If a winner is detected, the appropriate payoff odds are displayed as follows:

COMBINATION:(1)	MESSAGE DISPLAY:
CHERRY IN WHEEL 1	"PAY-OFF" " 2 TO 1"
CHERRIES IN WHEELS 1 & 2	"PAY-OFF" " 5 TO 1"
3 CHERRIES	"PAY-OFF" "10 TO 1"
3 BELLS	"PAY-OFF" "20 TO 1"
3 LEMONS	"PAY-OFF" "18 TO 1"
3 PLUMS	"PAY-OFF" "14 TO 1"
2 BELLS AND 1 BAR	"PAY-OFF" "20 TO 1"
2 LEMONS AND 1 BAR	"PAY-OFF" "18 TO 1"
2 PLUMS AND 1 BAR	"PAY-OFF" "14 TO 1"
3 BARS JACKPOT	"JAC-POT" "PAY-OFF" "250 TO 1"

If there is no winning combination, the message:

"YOU LOST" is displayed.

(1) DECREASE THE ODDS TO DECREASE THE POTENTIAL PAYOFF, IF DESIRED.

ROUTINE SEE BELOW | START ADDR | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION 'ODDSDSP' SETS UP CONST'S FOR PAYOFF DISPLAY: ''XXX TO 1'' 'FIXNBR' SETS A FLOATING XTANT FOR PART OF RANDOM # INDEX CALCULATION | SHEET 12

	LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
	1	01BA	06	8F		ODDSDSP	LODI,R2	>PAYOFF-1	
	2	C	05	04			LODI,R1	4	1 SEC RUN TIME
	3	BE	BB	82			ZBSR	*DISMES	
	4	CØ	06	97			LODI,R2	>XXXTO-1	
	5	2	05	08			LODI,R1	8	2 SEC RUNTIME
	6	1C4	BB	82			ZBSR	*DISMES	
	7					*NOTE COMMENT FOR ALTERNATIVE PROGRAMMING SELECTION			
***	8	1C6	1B	72			BCTR,UN	ODDSDSP	1: LOOP UNTIL RST DEPRESS.
***	9	1C6	17	00			RETC,UN	*FILLER CODE	2: AUTO RETURN FOR NEW GAME
	10								
	11	Ø1C8	08	0C		FIXNBR	LODR,R0	FIXIT	SUBTRACT 9 FROM CURRENT
	12	A	A4	09			SUBI,R0	9	'FIXIT CONST'. IF IT IS
	13	C	18	03			BCTR,EQ	NEWCON	ZERO, GO GET ANOTHER, THEN
	14	CE	C8	06			STRR,R0	FIXIT	STORE AWAY
	15	DØ	17				RETC,UN		AND RETURN TO CALL PROGRAM
	16								NOW, GET THE CURRENT RANDOM
	17	1D1	0C	01	13	NEWCON	LODA,R0	RANNUM	NUMBER -- ITS ALWAYS DIF-
	18	4	1B	78			BCTR,UN	NEWCON-3	FERRENT, THEN STORE AS
	19								'FIXIT' AND RETURN.
	20	1D6	XX			FIXIT	RES	1	VARIABLE
	21								
	22	1D7	00			*FILLER	CODE		
	23								
	24								

*** NOTE ALTERNATIVES ***

SEQUENCE: (CONTINUED)

The sequence is repeated automatically, or when the player depresses RST, dependant upon the mode chosen.

THE PROGRAM:

In describing the many options available to customize program "SLOTMACHIN" to your own use, much of the applications thrust of the program has also been introduced. In addition to the comments offered in the listing, the following program considerations are of special interest:

In sheets 1 and 2, a classic method for "stringing" a sequence of messages is described. Once the constants required for each message are set up, program control is turned over to subroutine "DISMES" for actual message display. Note that the TIME for each segment of the message is individually programmable.

Note the use of the BSXA instruction in line 10 of sheet 2. In this case, sequential "wheel rotation" is organized as a series of 4 events; these are:

Event 3: .....*rotation of all 3 wheels.*
Event 2: .....*rotation of wheels 2 and 3; wheel 1 stopped.*
Event 1: .....*rotation of wheel 3; wheels 1 and 2 stopped.*
Event Ø: .....*all wheels stopped; Show the 3-image pattern.*

ROUTINE DISROL | START ADDR Ø1E8 | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION 'DISROL' IS A SPECIAL HIGH SPEED DISPLAY CONTROL TO SHOW THE ROTATION OF THE 3 WHEELS TAKING PLACE. | SHEET 13

LINE	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1								
2	Ø1D8	0B	0C	10	WHTBL	RES	16	CODES CONFORM TO IMAGES FOR
3	B	0C	95	11				SLOT MACHINE WHEELS :
4	DE	0C	10	0C				C = CHERRY; B = BELL;
5	E1	11	10	0C				P = PLUM; L = LEMON.
6	4	10	0B	0C				= = BAR...CODE'95'
7	E7	11						
8								
9	Ø1E8	XX			DRC	RES	1	WHEEL ROLL DISPLAY CONSTANT
10					*LIMITS:	1 < N <	XX; MODIFY LOC.	'1EA' FOR DESIRED SPEED.
11								
12	Ø1E9	07	03		DISROL	LODI,R3	3	ORIG SPEED IS '3' IN '1EA'
13	B	CB	7B		DRAGAIN	STRR,R3	DRC	STORE THE CONSTANT. NOW
14	D	05	00			LODI,R1	<WHMES-1	SET WHL MESSAGE ADDR CONSTS
15	EF	06	F7			LODI,R2	>WHMES-1	AND GO DISPLAY.
16	F1	BB	E6			ZBSR	*USRDSP	ON EXIT, GET THE
17	3	0B	73			LODR,R3	DRC	WHEEL ROLL DISPLAY CONST.,
18	5	FB	74			BDRR,R3	DRAGAIN	& DECR. ZERO YET? YES!
19	Ø1F7	17				RETC,UN		RETURN TO CALL. NO! GO DIS-
20								PLAY AGAIN.
21								
22					* DISPLAYABLE MESSAGES FOLLOW ON			NEXT SHEET OF LISTING
23								
24								

THE PROGRAM: (CONTINUED)

During initialization (lines 14 & 15; sheet 1) the EVENTS COUNTER is set to 9 during execution of subroutine "EVENTSET" at address 'Ø13Ø'. This number is stored in R3, then used subsequently in routine "ROTATE1" as an index into a 4-entry table, "ROLL", in sheet 2. Each entry is 3 bytes long. The first time the BSXA instruction is executed, EVENTS equals 9. Therefore, the last entry in data table "ROLL" is accessed, and program control exits to routine "ROLL123", at address ØlØØ. Upon return from this routine, routine "STOPWHEL" (sheet 3 --- either version) is executed, in which the stored EVENTS are recalled and decremented by 3, the length of a single entry in table "ROLL". Thus, when execution returns to routine "ROTATE1" (or ROTATE3), the BSXA instruction accesses the next higher entry in table "ROLL, at address ØØ4Ø. This time, subroutine "ROLL23" is accessed, and only the 2nd & 3rd wheels are 'rotated'. This is repeated until all 3 wheels are 'stopped'.

The use of BCTA instructions in table "ROLL" is very important. Even though these are not subroutine 'call' instructions, they all access subroutines which are terminated with a RETC instruction. This causes program control to return to the BCTR instruction following the BSXA at line 10 on sheet 2. An unconditional branch to the STOP WHEEL COUNTER routine takes place. If the count is finished, the odds will be displayed. If not, further rotation ensues.

ROUTINE MESTBL | START ADDR 178Ø | PART OF PROGRAM SLOTMACHIN | 2650 PROGRAMMING FORM

DESCRIPTION THESE MESSAGES ACCESSED ONLY BY 'DISMES' SUBROUTINE. | SHEET 14

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	178Ø	17	13	00	ROTATE	RES 8		''_ROTATE_'' MESSAGE
2	3	07	0A	07				
3	6	0E	17					
4	1788	1B	00	12	YOULOST	RES 8		''YOU_LOST'' MESSAGE
5	B	17	11	00				
6	E	05	07					
7	1790	10	0A	1B	PAYOFF	RES 8		''_PAY_OFF''
8	3	19	00	0F				
9	6	0F	17					
10	1798	17	XX	XX	XXXTO	RES	8	''XXX_TO_1'' MESSAGE
11	B	17	07	00				
12	E	17	01					
13	17A0	17	XX	17	BAR	RES	8	STOPPED WHEELS DSPLY MSG
14	3	17	XX	17				
15	6	17	XX					
16	17A8	18	0A	0C	JACKPOT	RES	8	''JAC-POT_'' MESSAGE
17	B	19	10	00				
18	E	07	97					
19	17BØ	1B	00	12	YOUHOPE	RES	8	''YOU_HOPE'' MESSAGE
20	3	17	14	00				
21	6	10	0E					
22	17B8	01	07	17	ITIS	RES	8	''IT_IS_A_'' MESSAGE
23	B	01	05	17				
24	17BE	0A	17		*********	THIS CONCLUDES LISTING		OF PROGRAM SLOTMACHIN******

THE PROGRAM: (CONTINUED)

As indicated previously, there are 2 stop wheel alternatives; these are for automatic and manual (SENS key depression) stop. Both have start addresses at location 'ØØ46', and both are listed on sheet 3.

In alternative 1 (AUTO mode), a timeout byte is accessed, decremented, then compared with each of 3 numbers; these are:

H'C0' ....*for stop of Wheel 1 after 64 images, distributed among 3 wheels, have been displayed.*

H'70' ....*for stop of Wheel 2, after an additional 80 images, distributed among wheels 2 and 3, have been displayed.*

H'00' ....*for stop of the third wheel.*

If none of these numbers compare with the timeout byte's current value, wheel rotation & display continues

NOTE: YOU MAY WISH TO EXPERIMENT WITH THESE NUMBERS IN ORDER TO INCREASE OR DECREASE THE DELAY (AND SEQUENCE FOR EACH WHEEL STOP.

Upon getting a valid compare (lines 5, 7, or 9 --sheet 3) program control exits to decrements EVENTS, as described previously.

In alternative 2 (MANUAL mode), a double test of SENS key depression takes place. This ensures that SENS is pressed deliberately. The double test is made to illustrate possible software elimination of electrical noise that might be present in your future application's environment. The same technique is applicable to key debounce procedures you may desire. The ANDI instruction (line 18) is exercised repeatedly until the operator lifts his finger from the SENS key. It is only after that, that EVENTS are decremented (by 3), and a new entry in data table ROLL (sheet 2) is accessed.

Routine "COMPARE" (sheets 4 & 5) required a certain level of sophistication. Your author finally decided to test first for a "cherry" in wheel 1; continuing for test of wheels 2 (and 3) for cherries only if wheel 1's compare was valid. If the cherry compare failed, the only alternative was to exit, and add the hex codes of the remaining wheel's images to that of wheel 1. Code designation is nominal, since each image corresponds to a given hex code. From label "3BELLS" on, programming was simple, requiring only a sequence in which all possible winning combinations would be tested, with appropriate exits to set up the winning odds for display. The image codes are located in "WHTBL"; addr: 'Ø1D8'.

In sheet 5, note the use of RELATIVE INDIRECT ADDRESSING wherever possible. In a tightly organized program, this practice can save considerable user memory.

Starting at line 16 on sheet 5, and on both sheets 6 & 7, appropriate odds numbers are preset into the (winning) odds display message. Once again, the program is tightened in its use of memory, there being no serious time constraints to worry about.

THE PROGRAM: (CONTINUED)

Multiple entry points to (essentially) the same routine are demonstrated on sheet 7, starting at line 10. Routines "ROLL123", "ROLL23", and "ROLL3" are actually a single routine, terminated with a common RETC instruction in line 19. Entry, at addresses 'Ø1ØØ', 'Ø1Ø6', and 'Ø1ØC', is determined by the current "ROLL" table entry, described previously on page 6-81.
The 'rotating wheels" display is controlled by subroutine "DISROL", accessed after each new image is fetched. "DISROL" is located in memory at address '01E9'; its access is via indirect addressed ZBSR instructions. The code '84' in locations '0105','010B', and '0111' points to a 2-byte absolute address ('01E9') at location '0004' (sheet 1). "DISROL" is described near the end of this page.

Subroutine "FETCHWH", accessed by the ROLL routine, generates a random number for indexing into the 16-entry WHEEL TABLE which contains the hex codes for all images in order. Random number generation adds a degree of authenticity to the game, but still remains one of the hardest tasks for a (logical) computer to perform.

Routine "DISMES" is the common subroutine dedicated to all message displays except actual 3-WHEEL ROLL. Timeout and low-order message address constants are preset by the calling program prior to access of this routine. The constant ('01') contained in the COMI instruction (line 13; sheet 9) is deliberate. If the contents of R3 (DSPLDLY) were permitted to decrement to zero, the user would have to depress a HEX or FUNCTION key in order to exit from the display routine.

Routine "SHOWJAK" (sheet 10) is straight forward; it is adequately described in the introduction (above sheet 3 of the listing).

Routine "ZERO" (sheet 11) is necessary in order that UNUSED digits in the current odds display message will be blank,thus that they will not show a residual odds left over from a previous roll sequence.

Routine "DISROL" requires a higher rate of display change (to simulate the rotating wheels) than the other message routine, "DISMES". For this reason, it is streamlined for maximum efficiency, thus permitting the user to exercise considerable program control over the rate at which the wheels are cycled and displayed.

The message table accessed by routine "DISMES" is contained in the INSTRUCTOR 50's SMI memory, starting at location '178Ø' (sheet 14). This data table (MESTBL) is organized in 8 entries, each 8 bytes in length.

This concludes the comment on program "SLOTMACHIN", A short DIAGNOSTIC procedure for checkout of the WINNING JACKPOT message is provided on the next page.

JACKPOT MESSAGE DIAGNOSTIC PROCEDURE:

There is only ONE chance in over 65000 that your INSTRUCTOR 50 slot machine game will ever "HIT THE JACKPOT". In order for you to checkout and view the WINNING JACKPOT MESSAGE sequence, perform the following steps:

1. Depress MON, BKPT, 6, C, and ENT/NXT. ..... *To set a BREAKPOINT (Stop) address at location H'006C' in memory.*

2. Depress REG, C, 0, and ENT/NXT. ..... *To set a START address at location H'0000' in memory.*

3. Depress RUN ..... *To execute program "SLOTMACHINE" from address H'0000' to the Breakpoint.*

4. Load H'95' codes into the following locations: ..... *Use the INSTRUCTOR 50's "MEMORY DISPLAY & ALTER" mode to load these "Bar" codes. Three Bars make a jackpot. You are replacing the current "stopped wheel" codes at this time.*

   (1) H'00F9'
   (2) H'00FC'
   (3) H'00FF'

5. Depress BKPT twice..... *to cancel the Breakpoint address.*

6. Depress RUN. ..... *to execute the program from the Breakpoint without interruption.*

   The program will display the previous combination of wheels, but then will decode the "winning jackpot" codes due to the alteration performed in step 4. The winning jackpot message, described previously, will be displayed.

NOTES:

ELECTRONIC TRAIN

INTRODUCTION:

The flexible facilities of the INSTRUCTOR 50 are configured this time as an electronic TRAIN. Each combination of "engine", "freight cars", and "caboose" is known as a CONSIST. The operator may select any one of 16 consists for display at any given time, simply by toggling a selection ID into the 4 low-order I/O input switches, then depressing the INT key.

Speed is controlled automatically, under control of the code toggled into the 2nd, 3rd, and 4th high-order I/O input switches. Seven "speeds" may be simulated, from 10 to 80 MPH, in addition to a "stop train" selection.

Direction is controlled via SENS key depression. When the SENS key is depressed, the "train" runs in reverse (left to right) across the HEX DISPLAY. The speed is that determined by the controlling I/O input switches previously described. The "train" runs forward (right to left) when the operator's finger is removed from the SENS key. The FLAG LED is turned on whenever the reverse command is given.

There is one minor "bug" in the software. If the "train" is stopped, it may not be restarted in either direction unless

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION THIS PROGRAM PERMITS SELECTION OF ANY 1 OF 16 TRAINS, & ANY 1 OF 7 SPEEDS; FORWARD OR REVERSE, PLUS STOP AND TRAIN RESELECTION | SHEET 1

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	0000	57	07		DISPLAY	REDE,R3	PORT7	GET MANUAL CTRL BYTE, & I-
2	2	47	70			ANDI,R3	H'70'	SOLATE SPEED FACTOR, THEN
3	4	1B	03			BCTR,UN	INTFREE	BRANCH PAST INTPT ADDR
4								
5	6	XX			HOLD	RES	1	TEMP. SPEED STORAGE CODE
6								
7	7	00	71		SWYDINT	ACON	H'0071'	SWITCH YARD INTPT ADDRESS
8								
9	9	CB	7B		INTFREE	STRR,R3	HOLD	PUT SPEED FACTOR IN 'HOLD'
10	B	05	00			LODI,R1	<SWYARD-1	THEN SETUP CONSTANTS FOR
11	D	06	37			LODI,R2	>SWYARD-1	SWITCHYARD DISPLAY OF CUR-
12	0F	BB	E6			ZBSR	*USRDSP	RENTLY SELECTED ''TRAIN''.
13	11	0B	73			LODR,R3	HOLD	NOW, GET SPEED FACTOR, &
14	3	FB	74			BDRR,R3	INTFREE	DECR. DISPLAY OVER? NO!! DO
15	015	1B	31			BCTR,UN	CHKSENS	OVER. YES!! GO READ DIREC-
16								TION.
17	7	C0				NOP		FILLER CODE.
18								
19	018	05	00		ROLLON	LODI,R1	0	INIT. SWYARD INDEX, AND
20	A	07	08			LODI,R3	8	COUNTER, THEN PUT SWYARD IN
21	C	0D	60	38	REPEAT1	LODA,R0	SWYARD,R1	TEMP. DONE YET? NO! MOVE
22	1F	CD	20	3F		STRA,R0	TEMP-1,R1,+	REST OF TRAIN. YES!! SET UP
23	22	FB	78			BDRR,R3	REPEAT1	FOR BYTE SHIFT AND MOVE
24	024	05	08			LODI,R1	8	TEMP BACK INTO SWITCHYARD

A JOHN GARCEAU PROGRAM

INTRODUCTION: (CONTINUED)

the RST key is depressed. If a change of 'consists' is required, the operator will toggle the appropriate I/O input swiches to select the desired consist and running speed, depress RST, and then depress INT in that order!!!!. If no consist change is necessary, the operator merely selects a running speed, and depresses RST.

If the train is operating at one of its running speeds, the consist may be changed as described in the first paragraph of this introduction.

USER OPERATION - SET UP:

After loading program "TRAIN" into the INSTRUCTOR 50's memory, the switches should be positioned as follows:

I/O Mode Selection Switch	EXTENDED I/O PORT Ø7
AC Line /Keyboard Interrupt Selection Slide Switch	KEYBOARD
Interrupt Toggle Switch	INDIRECT (address)
Parallel I/O Input Switches	SEE DEFINITION ON NEXT SHEET OF DESCRIPTION.

ROUTINE AS SHOWN | START ADDR XXXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

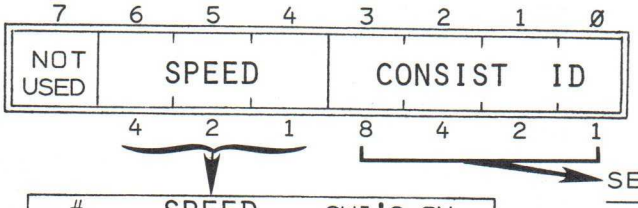
DESCRIPTION SHIFT BYTE TRANSFERS FOR TRAIN MOTION, ALSO DIRECTION IDENT VIA SENS KEY INTERP. | SHEET 2

LINE	ADDRS.	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	ØØ26	0D	60	3F	REPEAT2	LODA,R0	TEMP-1,R1	SHIFTING 1 BYTE LEFT (FWD)
2	9	CD	40	37		STRA,R0	SWYARD-1,R1,-	DONE? NO. FINISH. YES.
3	C	59	78			BRNR,R1	REPEAT2	MOVE MS BYTE OF TEMP TO
4	2E	08	10			LODR,R0	TEMP	LS BYTE OF SWYARD, THEN
5	3Ø	C8	0D			STRR,R0	SWYARD+7	GO BACK
6	2	1B	4C			BCTR,UN	DISPLAY	AND DISPLAY THE RESULT.
7	34-37	00	00	00	*FILLER	CODE		
8	ØØ38	97	02	06	SWYARD	RES	8	SWITCHYARD IS AN 8-BYTE
9	B	05	00	97				TABLE FOR CURRENT TRAIN
10	3E	12	10					ORIG. ENTRY MESSAGE IS:
11								''2650 UP''
12	ØØ4Ø	XX	XX	XX	TEMP	RES	8	TEMP IS TEMPORARY STORAGE
13	3	XX	XX	XX				FOR TRAINS BEING MOTION
14	6	XX	XX					-SHIFTED (LEFT OR RIGHT)
15								
16	ØØ48	B4	80		CHKSENS	TPSU	SENS	TEST FOR REVERSE (SENSE
17	A	18	04			BCTR,EQ	DELAY	KEY DOWN) COULD BE: GO FIND
18	C	74	40			CPSU	FLAG	OUT. IT WAS NOT. CLR FLAG
19	4E	1B	48			BCTR,UN	ROLLON	LED, AND EXIT TO ROLLON.
20	5Ø	D8	7E		DELAY	BIRR,R0	$	SET 1 MSEC DELAY & CHK IF
21	2	B4	80			TPSU	SENS	SENS KEY DELIB. DEPR.
22	4	98	42			BCFR,EQ	ROLLON	IT WASN'T. GO FORWARD.
23	6	76	40		REVERSE	PPSU	FLAG	IT WAS. SET THE FLAG LED,
24	Ø58	C1				STRZ	R1	AND SET CONST'S TO REVERSE

USER OPERATION (CONTINUED)

Parallel I/O Input Switches:

The 8 Parallel I/O input switches are position-defined into 2 fields as shown in the following diagram:



SEE TABLE ON NEXT PAGE

#	SPEED	SWI'S ON
7	10 MPH	6 5 4
6	12 "	6 5
5	15 "	6 4
4	19 "	6
3	27 "	5 4
2	40 "	5
1	80 "	4
Ø	STOPPED IN CURRENT POSITION.	ALL OFF

** Toggle speed switches 6, 5, and 4 ON = Speed 7. **

ROUTINE AS SHOWN | START ADDR XXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION CONT. OF CONTROL PROGR; REVERSE CONTROL, ALSO START OF 'SWYRDINT' INTERRUPT SERVICE, INCL. REGISTER VALUE SAVES. | SHEET 3

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØ59	07	F9			LODI,R3	H'F9'	DIRECTION. NOW MOVE SWYARD
2	5B	0D	60	38	LOOP1	LODA,R0	SWYARD,R1	TO TEMP; THIS TIME WITH A
3	5E	CD	20	40		STRA,R0	TEMP,R1,+	RIGHT BYTE-SHIFT. 7 BYTES
4	61	DB	78			BIRR,R3	LOOP1	MOVED? NO. FINISH. YES!!!
5	3	08	5A			LODR,R0	SWYARD+7	MOVE LS SWITCHYARD INTO
6	5	C8	51			STRR,R0	SWYARD	MS POSITION, THEN
7	7	0D	60	40	LOOP2	LODA,R0	TEMP,R1	MOVE ALL OF TEMP BACK INTO
8	A	CD	60	38		STRA,R0	SWYARD,R1	THE SWITCHYARD - NO SHFT.
9	D	F9	78			BDRR,R1	LOOP2	WHEN FINISHED, EXIT
10	6F	9B	00			ZBRR	DISPLAY	TO SHOW THE TRAIN IN ITS
11								CURRENT POSITION.
12					*			
13					*1ST PART OF SWYRDINT IS ROUTINE TO SAVE R0-R3, ALSO THE PSW			
14					*(UPPER & LOWER) IN CASE THEY ARE			NEEDED WHEN INTERRUPT SER-
15					*VICE IS COMPLETE.			
16					*			
17	Ø71	CC	17	90	SWYRDINT	STRA,R0	SAVEREG	SAVE REGS Ø THRU 3 IN
18	4	CD	17	91		STRA,R1	SAVEREG+1	LOCATIONS H'1790' THRU
19	7	CE	17	92		STRA,R2	SAVEREG+2	H'1793'. THEN BRANCH
20	A	CF	17	93		STRA,R3	SAVEREG+3	AROND 128-BYTE CONSIST TABL
21	Ø7D	1F	01	00		BCTA,UN	CONTINT	TO CONTINUE THE SAVE.
22					*			
23					*NEXT SHEETS OF LISTING PROVIDE A			128-BYTE TABLE OF TRAIN
24					*CONSISTS; 16X8BYTE ORGANIZATION			

USER OPERATION (CONTINUED)

The 4 low-order I/O Input switches control train "CONSIST" selection as shown in the following table:

SWITCHES ON:	I.D.:	CONSIST:
ALL OFF	Ø	HSE (alone)
Ø	1	HSE + (3) FC
1	2	HSE + FC + (2) EC
1 Ø	3	HSE + FC + EC + ½FC + C
2	4	HSE + (2) FC + C
2 Ø	5	HSE + (3) C
2 1	6	HSE + (3) FC + C
2 1 Ø	7	LSE + (3) C
3	8	LSE + (2) EC + C
3 Ø	9	LSE + (2) ½FC + C
3 1	A	LSE + (2) EC + C
3 1 Ø	B	LSE + (3) FC + C
3 2	C	LSE (alone)
3 2 Ø	D	MT (alone)
3 2 1	E	EM + (4) C
3 2 1 Ø	F	MT + (3) C

NUMBERS IN (PARENTHESES) DEFINE NUMBER OF CARS IN ROLLING STOCK.

DEFINITION:

HSE	HIGH SPEED ENGINE
LSE	LOW-SPEED ENGINE
MT	BATTERY-POWERED MINE TRACTOR
EM	OVERHEAD ELECTRIC POWER MINE ENGINE
FC	FULL FREIGHT CAR
½FC	ONE HALF FULL FREIGHT CAR
EC	EMPTY CAR
C	IS CABOOSE IN ALL CONSISTS EXCEPT #'S 7, E, & F, WHERE IT IS A COAL CAR

** Select CONSIST 2 by setting input switch 1 ON; the others OFF **

ROUTINE CONSIST | START ADDR ØØ8Ø | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION 'CONSIST' IS A 16 X 8 DATA TABLE, CONTAINING EACH OF 16 POSSIBLE 'TRAINS' WHICH MAY BE SELECTED. | SHEET 4

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØ8Ø	95	E0	97	IDØ	RES	8	CONSIST Ø
2	3	97	97	97				
3	6	97	97					
4	88	95	E0	80	ID1	RES	8	CONSIST 1
5	B	80	80	97				
6	E	97	97					
7	9Ø	95	E0	80	ID2	RES	8	CONSIST 2
8	3	92	92	97				
9	6	97	97					
10	98	95	E0	80	ID3	RES	8	CONSIST 3
11	B	92	EE	95				
12	E	97	97					
13	AØ	95	E0	80	ID4	RES	8	CONSIST 4
14	3	80	95	97				
15	6	97	97					
16	A8	95	E0	95	ID5	RES	8	CONSIST 5
17	B	95	95	97				
18	E	97	97					
19	BØ	95	E0	80	ID6	RES	8	CONSIST 6
20	3	80	80	95				
21	6	97	97					
22	B8	95	96	95	ID7	RES	8	CONSIST 7
23	B	95	95	97				
24	ØØBE	97	97					

USER OPERATION - RUNNING THE TRAIN:

1. Depress RST .....*The message: "2650 UP." is displayed immediately.*

   NOTE: THE FIRST TIME THE INT KEY IS DE PRESSED AFTER THE PROGRAM IS LOAD- ED INTO MEMORY, MESSAGE '.2650 UP.' IS LOST.*

2. Depress INT .....*To route "CONSIST 2" into the Hex Display 'switchyard'. The following image is displayed, moving forward (right to left) at 10 MPH (Speed 7).*

3. Toggle input switches 6, 5, and 4 ..*To run Consist 2 at different speeds (see page 6-87).*

4. Depress the SENS key ...*To move the train in reverse.*

* A 'RECOVERY PROCEDURE' IS PROVIDED ON PAGE 6-9.

ROUTINE CONSIST | START ADDR XXX | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION 'CONSIST' OF ''TRAIN'' (CONTINUED) | SHEET 5

LINE	ADDRS	DATA B0	B1	B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	OPERANDS	COMMENT
1	ØØCØ	95	96	92	ID8	RES	8	CONSIST 8
2	3	92	95	97				
3	6	97	97					
4	C8	95	96	EE	ID9	RES	8	CONSIST 9
5	B	EE	95	97				
6	E	97	97					
7	DØ	95	96	92	IDA	RES	8	CONSIST A
8	3	92	95	97				
9	6	97	97					
10	D8	95	96	80	IDB	RES	8	CONSIST B
11	B	80	80	95				
12	E	97	97					
13	EØ	95	96	97	IDC	RES	8	CONSIST C
14	3	97	97	97				
15	6	97	97					
16	E8	C7	97	97	IDD	RES	8	CONSIST D
17	B	97	97	97				
18	E	97	97					
19	FØ	8D	95	95	IDE	RES	8	CONSIST E
20	3	95	95	97				
21	6	97	97					
22	F8	C7	95	95	IDF	RES	8	CONSIST F
23	B	95	97	97				
24	ØØFE	97	97					END OF 'CONSIST TABLE'.

USER OPERATION: (CONTINUED)

5. Depress RST .....*if the train consist is stopped.*

NOTE: TURN ON AT LEAST ONE OF THE SPEED CONTROL SWITCHES PRIOR TO DEPRESSING THE RST KEY.

6. Toggle Input Switches 3, 2, 1, or Ø .....*To select any other consist. Try them all, at different speeds.*

7. Depress INT.....*To move the selected consist into the Switchyard.*

THE PROGRAM:

There are 2 major points of interest in program "TRAIN"; these include:

* Forward and Reverse Motion Control, and
* Selection & Transfer of the desired Consist into the Switchyard.

FORWARD AND REVERSE MOTION:

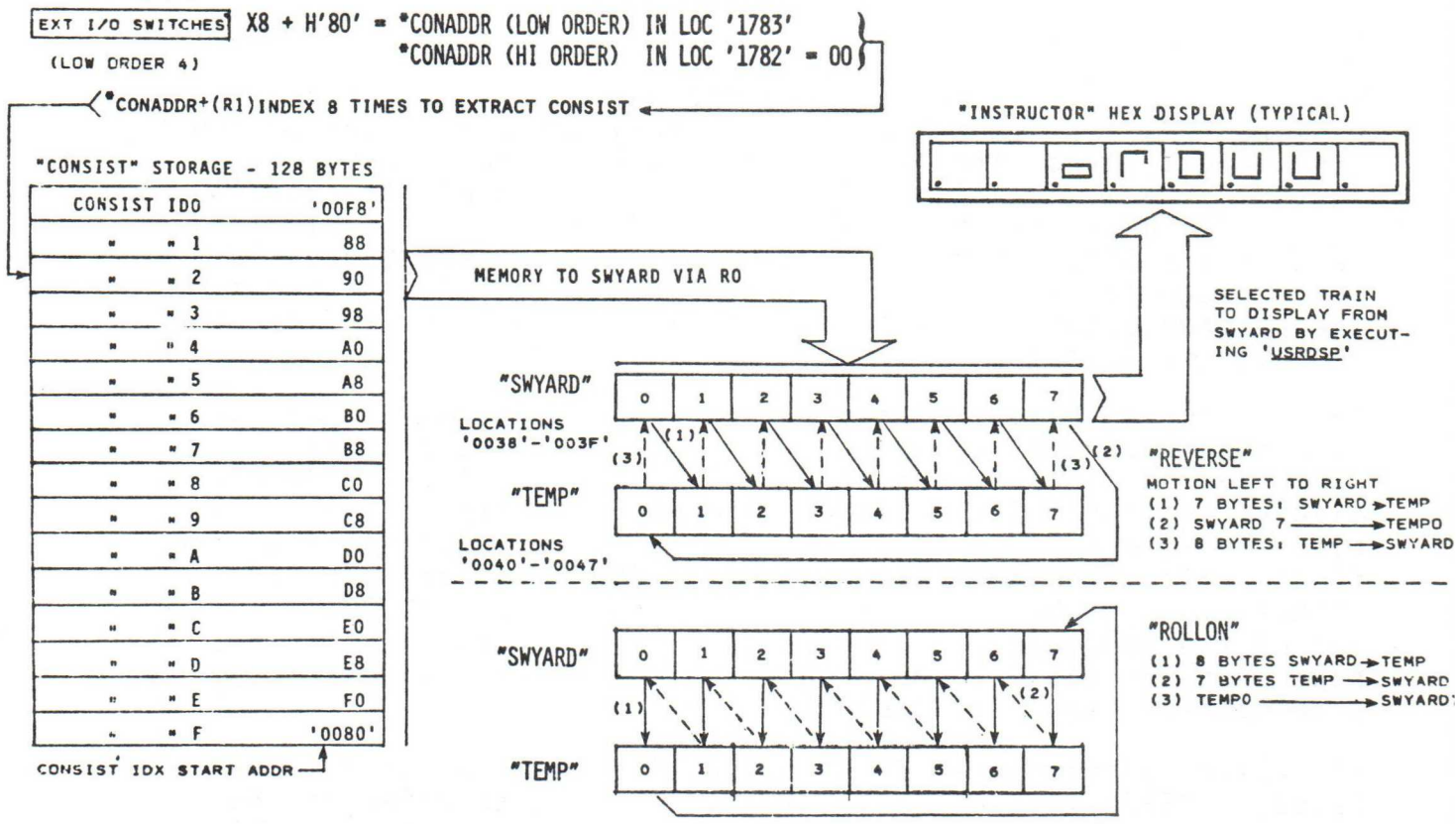
Refer to the diagram on the next page, also to the highlights pointed out in the Listing. The description is provided on sheet 8.

ROUTINE CONTINT | START ADDR Ø1ØØ | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION CONTINUATION OF THE INTERRUPT SERVICE ROUTINE. | SHEET 6

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1	Ø1ØØ	13			CONTINT	SPSL		SAVE THE LOWER PSL AT LOC'N
2	1	CC	17	94		STRA,R0	SAVEREG+4	'1794'; AND UPPER PSW AT
3	4	12				SPSU		LOC'N '1795'
4	5	CC	17	95		STRA,R0	SAVEREG+5	
5	8	75	08			CPSL	WC	CLR WC BIT, THEN READ UP
6	A	54	07			REDE,R0	PORT7	MANUAL CONTROL & ISOLATE
7	C	44	0F			ANDI,R0	H'0F'	THE CONSIST I.D.. NOW, MUL-
8	E	DO				RRL	R0	TIPLY THE I.D. BY 8 TO GET
9	1ØF	DO				RRL	R0	AN 8-BYTE INDEX
10	11Ø	DO				RRL	R0	THEN RESTORE THE WC BIT
11	1	77	08			PPSL	WC	FOR FUTURE USE.
12								
13					CONADDR	ACON	H'1782'	
14								
15	113	84	80			ADDI,R0	H'80'	NOW ADD CONSIST BASE ADDR
16	5	CC	17	83		STRA,R0	CONADDR+1	TO INDX (LO BYTE) & STORE
17	8	20				EORZ	R0	THEN CLR R0 AND STORE AS
18	9	CC	17	82		STRA,R0	CONADDR	HI BYTE OF ADDR. SET CONSTS
19	C	C1				STRZ	R1	TO MOVE SELECTED CONSIST
20	D	07	Ø8			LODI,R3	8	INTO THE SWITCHYARD
21	11F	0D	F7	82	LOOP3	LODA,R0	*CONADDR,R1	AND MOVE IT.
22	122	CD	2Q	37		STRA,R0	SWYARD-1,R1,+	
23	5	FB	78			BDRR,R3	LOOP3	DONE YET? NO; FINISH. YES:
24	Ø127	0C	17	95	RESTORE	LODA,R0	SAVEREG+5	RESTORE PREV SAV'D P S W.

DIAGRAM OF CONSIST SELECTION, TRANSFER, AND FORWARD/REVERSE MOTION



ROUTINE RESTORE | START ADDR Ø127 | PART OF PROGRAM TRAIN | 2650 PROGRAMMING FORM

DESCRIPTION WIND UP INTERRUPT SERVICE ROUTINE BY RESTORING PREVIOUSLY SAVED REGS TO ORIGINAL VALUES. | SHEET 7

LIN	ADDRS	DATA BØ	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1					RECALL	ACON	H'1798'	SPACE FOR A 3-BYTE SUBROU-
2					RECALL	RES	3	TINE TO RECALL LO-PSW.
3								
4	Ø12A	92				LPSU		THAT'S UPPER PSW....
5	B	0D	17	91		LODA,R1	SAVEREG+1	ALSO REGISTERS R1; R2; R3.
6	12E	0E	17	92		LODA,R2	SAVEREG+2	
7	131	0F	17	93		LODA,R3	SAVEREG+3	
8	4	04	37			LODI,R0	RETE	NOW BUILD A SUBROUTINE TO
9	6	CC	17	9A		STRA,R0	RECALL+2	RESTORE THE PREVIOUSLY SA-
10	9	0C	17	94		LODA,R0	SAVEREG+4	VED PSL. IS ROUTINE 'RECALL
11	C	CC	17	99		STRA,R0	RECALL+1	ADDR: INSTRUCTION:
12	13F	04	77			LODI,R0	PPSL	1798 PPSL PSL(SAVED)
13	141	CC	17	98		STRA,R0	RECALL	179A RETE
14	4	0C	17	90		LODA,R0	SAVEREG	
15	7	75	FF			CPSL	H'FF'	NOW CLEAR THE CURRENT PSL
16	Ø149	1F	17	98		BCTA,UN	RECALL	AND EXECUTE 'RECALL' TO
17					*			RETURN FROM INTERRUPT SVC.
18					*FOR DIFFERENT MESSAGE DISPLAY, CHANGE ANY CONSIST 8-BYTE			
19					*CODES TO THOSE OF YOUR OWN CHOOSING, THEN SELECT THE ORDER			
20					*OF DESIRED 'WORDS' VIA APPROPRIATE 'CONSIST ID' SWITCHES.			
21					*'2650 UP' MESSAGE IS LOST AFTER 1ST SELECTION OF CONSIST			
22					* WHEN THE 'INT' KEY IS DEPRESSED.			
23								
24								

THE PROGRAM (CONTINUED) Sheet 8

The steering process for forward and reverse motion control is routine "CHKSENS" (SHEET 2). Immediately after the current consist is displayed, the logical condition of the SENS input is tested. If SENS is active (key down), the BCTR,EQ instruction (line 17) passes execution to a short delay, followed by a 2nd test of the SENS input. This ensures that SENS key depression was deliberate. If the test for SENS activity fails either time, the program assumes that forward motion is desired, and routine "ROLLON" is accessed.

Routine "REVERSE" is initiated with FLAG LED turn-on, followed by a sequenced shift-right of the 8-byte selected consist message. The sequence is graphically illustrated at the middle of the diagram on page 6-91. 8 bytes of memory, labeled "TEMP" provide sufficient space to accomplish the byte shift desired. The first shift, from the Switchyard to 'TEMP', and transfer of 'TEMP7' to 'SWYARD1', is accomplished by process "LOOP1". Transfer of the remaining 7 bytes in 'TEMP', now right shifted, is handled by process "LOOP2". Upon completion of the transfer, program control exits to display the result.

If desired direction is forward, routine "ROLLON" is accessed. Process "REPEAT1" handles the initial 8-byte transfer of the current consist from 'SWYARD' to 'TEMP'. In process 'REPEAT2', the least significant 7 bytes are left-shifted into the 'SWYARD' from 'TEMP'. The remaining byte in TEMP', at location TEMPØ, is then transferred to the least significant location in the 'SWYARD'; this is SWYARD+7. SWYARD+7 corresponds to the rear-most train position in the forward direction. When the transfer is complete, program control exits to display the result.

SELECTION AND TRANSFER OF THE DESIRED CONSIST INTO THE SWITCHYARD:

As noted previously, there are 16 possible train consists which may be selected. From a programmer's viewpoint, these consists are organized in a 16-entry table (SHEETS 4 & 5). Each entry is 8-bytes in length, this being the number of bytes usable in any displayable message.

To select any given entry, the appropriate I/O input switches are positioned to provide a selection I.D., as outlined at the top of sheet 4.

Routine "SWYRDINT" (Switchyard Interrupt) is accessed only when the INT key is depressed. Following an initial process of storing the contents of the current registers and PSW in mainline program use (SHEET 3; LINE 17 AND FOLLOWING), program control transfers to the main part of the interrupt service routine at location 'Ø1ØØ' (SHEET 6). The instructions at addresses '010A'

THE PROGRAM (CONTINUED)

through '0113' convert the input switch pattern into a precise index to the desired entry in the Consist Table. This index is then posted as the least significant byte in a 2-byte indirect address field: 'CONADDR', in the INSTRUCTOR 50's SMI memory. Thus, process "LOOP3" can then be executed to make an orderly transfer of the desired 8-byte consist message.

Routine "SWYRDINT" concludes with restoration of previously stored register and PSW contents. The instructions at locations '0134' to '0141' create a 2-instruction subroutine, "RECALL". The purpose of "RECALL" is to permit return to the main program without altering the previously stored condition code. This code, which is used to determine the direction of many branch instructions, can be altered during register restoration. The necessary instructions are found in sheet 7.

Refer to sheet 1 for a description of "speed control".

In lines 1 and 2, the speed input switch pattern is isolated by the microprocessor. Th resultant value is used to condition the Monitor's USRDSP display routine running time constant. When all speed switches are off, the value is zero; this is why the selected consist remains stopped. If 1 or more switches are on, the display "times out" after a short delay. The delay is directly proportional to the value of the switch pattern, as described in the User Operation procedures on sheet 3. After display timeout, program control exits to check direction demands. The consist message is shifted, and control loops back to display the result once again.

The listing of this program is particularly informative in that it provides precise examples of:

- I/O input pattern detection, isolation, and use.(SHEET 1)
- SENSE bit testing and control usage (SHEET 2)
- Multi-byte transfer and shifting techniques (SHEETS 1,2, & 6)
- Indirect address construction (SHEET 6)
- Data Table organization by multi-byte entry (SHEETS 4 & 5)
- Index preparation and usage (SHEET 6)
- Register save & restore techniques, including pseudo subroutine construction for Condition Code protection. (SHEETS 3,6, & 7)

This concludes the description of program "TRAIN".

INTRODUCTION:

When you listened to the INSTRUCTOR 50 INTRODUCTORY CASSETTE tape, the "electronic music" you heard was generated by executing the program "INSTRUCTOR 50 MUSIC THEME". It is fully documented on these next 3 pages.

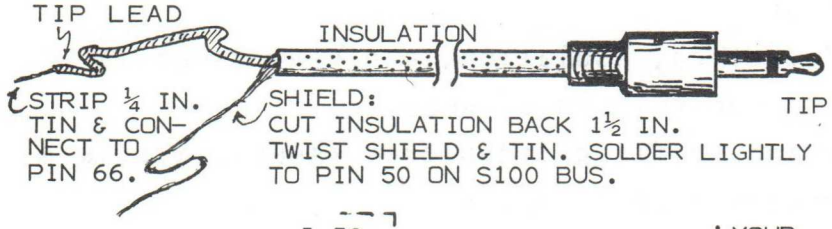
When "source code" is input to an ASSEMBLER, it is necessary to declare the (hex code) value of each symbolic address or operand, PRIOR to its use in the program itself. As explained earlier in this chapter, these declarations are made by implementing EQUATE, RES, and ORG statements. In line 55 of the listing on the next page, an ORG (address origin) statement defines memory location H'0000' as the program's start address. On the following page, data table "NTAB" is origined at address H'100'.

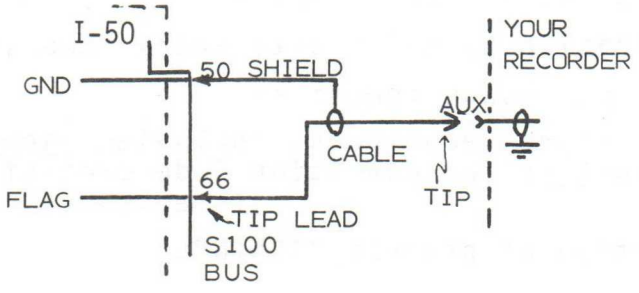
Most of the following page is dedicated to definition of symbolic EQUates in terms of their hex code values.

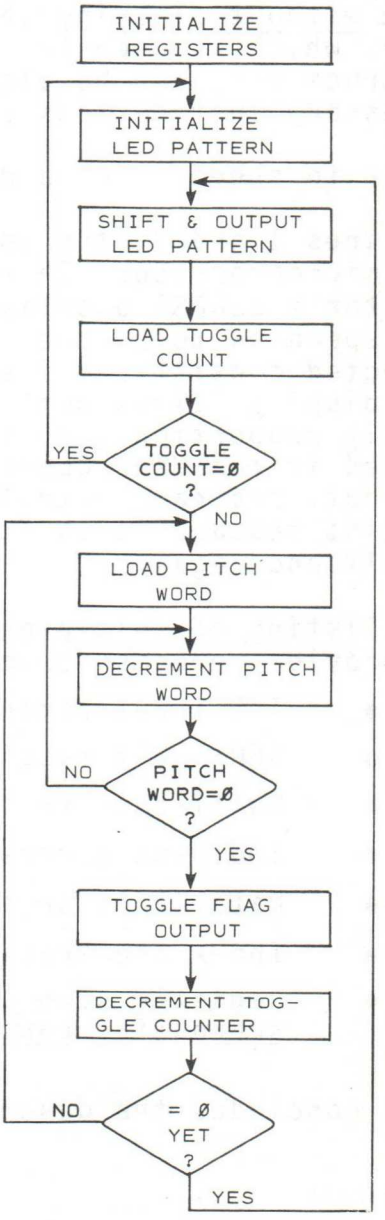
In line 83, a RES statement reserves 1 byte of user memory (at address H'0032') for a symbolic function called "LEDWRD".

A FLOWCHART (to the right) illustrates the overall sequencing of program "INSTRUCTOR 50 MUSIC THEME" as it is executed to play the desired music.

Prior to executing the "music theme" program, prepare an AUDIO CABLE as shown in the diagram below. Then, connect it between the INSTRUCTOR 50 and your cassette recorder or, at your option, to the AUX input of an available hi-fi amplifier.







The source and object codes for program "INSTRUCTOR 50 MUSIC THEME" are contained on the following pages.

```
LINE ADDR  OBJECT  E SOURCE

0002                ********************************************************
0003                *                                                      *
0004                *  INSTRUCTOR 50 THEME                                 *
0005                *                                                      *
0006                *  THIS PROGRAM PLAYS A SHORT MUSICAL                  *
0007                *  THEME ON THE INSTRUCTOR 50                          *
0008                *                                                      *
0009                *  THE DATA FOR THE MUSIC IS STORED IN A               *
0010                *  TABLE AT ADDRESS 100H.  EACH NOTE IS                *
0011                *  SPECIFIED BY A TWO-BYTE PAIR.  THE FIRST            *
0012                *  BYTE SPECIFIES THE DURATION OF THE TONE             *
0013                *  IN HALF CYCLES.  THE SECOND BYTE IS USED            *
0014                *  TO DETERMINE THE LENGTH OF EACH HALF                *
0015                *  CYCLE.  A DURATION CODE OF 00 CAUSES THE            *
0016                *  TUNE TO START OVER AGAIN.                           *
0017                *                                                      *
0018                * PROGRAM BY JOE DOLL                                  *
0019                * MUSIC BY DENNIS NYRIN                                *
0020                *                                                      *
0021                ********************************************************
0022                *
0023                * EQUATES
0024                *
0025 0000           R0      EQU     0               REGISTER EQUATES
0026 0001           R1      EQU     1
0027 0002           R2      EQU     2
0028 0003           R3      EQU     3
0029 0000           Z       EQU     0               CONDITION EQUATES
0030 0003           UN      EQU     3
0031                *
0032                * NOTE AND DURATION VALUES
0033                *
0034 0030           B1      EQU     H'30'
0035 0066           B1P     EQU     H'66'
0036 0033           C2      EQU     H'33'
0037 0060           C2P     EQU     H'60'
0038 0039           D2      EQU     H'39'
0039 0055           D2P     EQU     H'55'
0040 0040           E2      EQU     H'40'
0041 004C           E2P     EQU     H'4C'
0042 0044           F2      EQU     H'44'
0043 0048           F2P     EQU     H'48'
0044 004D           G2      EQU     H'4D'
0045 0040           G2P     EQU     H'40'
0046 0056           A2      EQU     H'56'
0047 0039           A2P     EQU     H'39'
0048 0060           B2      EQU     H'60'
0049 0032           B2P     EQU     H'32'
0050 0066           C3      EQU     H'66'
0051 002F           C3P     EQU     H'2F'
0052                *
0053                * PROGRAM
0054                *
0055 0000                   ORG     0               PROG START AT H'000'
0056 0000 0420      INIT    LODI,R0 H'20'           INHIBIT INTERRUPTS,
```

```
LINE ADDR  OBJECT  E SOURCE

0057 0002 92              LPSU              ZERO SP,S,F
0058 0003 0400            LODI,R0 0         SEL BANK 0,CLEAR CC,WC
0059 0005 93              LPSL
0060 0006 0401      BEGIN LODI,R0 H'01'
0061 0008 CC0032          STRA,R0 LEDWRD    INITIALIZE LEDWRD
0062 000B 0600            LODI,R2 0         INIT NOTE COMMAND INDEX
0063 000D 0C0032    GETOG LODA,R0 LEDWRD    GET LED WORD
0064 0010 50              RRR,R0            SHIFT RIGHT LEDS
0065 0011 CC0032          STRA,R0 LEDWRD    SAVE NEW VALUE
0066 0014 D407            WRTE,R0 H'07'     WRITE TO LED PORT
0067 0016 0E20FF          LODA,R0 NTAB-1,R2,+ LOAD R0 WITH TOGGLE COUNT
0068 0019 186B            BCTR,Z  BEGIN     RESTART IF COUNT=0
0069 001B C1              STRZ    R1        STORE TOGGLE COUNT IN R1
0070 001C 0E20FF          LODA,R0 NTAB-1,R2,+ GET PITCH WORD
0071 001F C3              STRZ    R3        AND STORE IN R3
0072 0020 03        HALFCY LODZ   R3        RETRIEVE PITCH WORD TO R0
0073 0021 F87E      PCHDLY BDRR,R0 PCHDLY   DELAY FOR 1/2 CYC TIME OF NOTE
0074 0023 03              LODZ    R3
0075 0024 F87E      PCHDL2 BDRR,R0 PCHDL2
0076 0026 12              SPSU              GET PSU
0077 0027 2440            EORI,R0 H'40'     TOGGLE FLAG OUTPUT
0078 0029 92              LPSU
0079 002A C0              NOP               NO OPS TO DELAY
0080 002B C0              NOP               *
0081 002C C0              NOP               *
0082 002D C0              NOP               *
0083 002E F970            BDRR,R1 HALFCY    DECREMENT TOGGLE COUNT
0084                *                       AND REPEAT IF NOT ZERO
0085 0030 1B5B            BCTR,UN GETOG     GO BACK AND START NEXT NOTE
0086               *
0087 0032          LEDWRD RES    1          LED WORD BYTE
0088               *
0089               * NOTE TABLE - DURATION,PITCH
0090               *
0091 0033                 ORG    H'100'
0092 0100 33604D40  NTAB  DATA   C2,C2P,G2,G2P,E2,E2P,C3,C3P    32 NOTES IN THEME
     0104 404C662F
0093 0108 4D403360        DATA   G2,G2P,C2,C2P,E2,E2P,G2,G2P
     010C 404C4D40
0094 0110 56393360        DATA   A2,A2P,C2,C2P,E2,E2P,A2,A2P
     0114 404C5639
0095 0118 404C5639        DATA   E2,E2P,A2,A2P,C2,C2P,C3,C3P
     011C 3360662F
0096 0120 39555639        DATA   D2,D2P,A2,A2P,F2,F2P,D2,D2P
     0124 44483955
0097 0128 44483955        DATA   F2,F2P,D2,D2P,A2,A2P,F2,F2P
     012C 56394448
0098 0130 60324D40        DATA   B2,B2P,G2,G2P,D2,D2P,F2,F2P
     0134 39554448
0099 0138 30664448        DATA   B1,B1P,F2,F2P,B2,B2P,G2,G2P
     013C 60324D40
0100 0140 00              DATA   0
0101 0000                 END    0
```

ROUTINE	START ADDR	PART OF PROGRAM	2650 PROGRAMMING FORM

DESCRIPTION	SHEET

LINE	ADDRS	DATA B0	DATA B1	DATA B2	LABEL	SYMBOLIC INSTRUCTION OPCODE	SYMBOLIC INSTRUCTION OPERANDS	COMMENT
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								
22								
23								
24								

DIRECT RELATIVE ADDRS: SECOND BYTE

ADD H'80' TO DISPLACEMENT FOR INDIRECT ADDRESS DEFINITION

+	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
N	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
−		7F	7E	7D	7C	7B	7A	79	78	77	76	75	74	73	72	71	70	6F	6E	6D	6C	6B	6A	69	68	67	66	65	64	63	62	61
+	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
N	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
−	60	5F	5E	5D	5C	5B	5A	59	58	57	56	55	54	53	52	51	50	4F	4E	4D	4C	4B	4A	49	48	47	46	45	44	43	42	41	40

COMMENT SHEET

TITLE:

REVISION:

This form is not intended to be used as an order blank. Signetics Corporation solicits your comments about this manual with a view to improving its usefulness in later editions.

Applications for which you use this manual.

Do you find it adequate for your purpose?

What improvements to this manual do you recommend to better serve your purpose?

Note specific errors discovered (please include page number reference).

General comments;

FROM NAME:_____ POSITION:_____

COMPANY NAME:_____

ADDRESS:_____

Tear off and return

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 166
SUNNYVALE, CALIF.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

signetics

a subsidiary of **U.S. Philips Corporation**

Signetics Corporation
P.O. Box 9052
811 East Arques Avenue
Sunnyvale, California 94086

Bin No. 038 MOS Microprocessor Division

FOLD

FOLD

signetics

a subsidiary of **U.S. Philips Corporation**

Signetics Corporation
P.O. Box 9052
811 East Arques Avenue
Sunnyvale, California 94086
Telephone 408/739-7700