MAINDEC 1

INSTRUCTION TEST

Abstract: Instruction Test is a sequence of sixteen programs which tests the operation of all PDP-1 instructions except the iot group. For deferrable instructions, indirect addressing is checked. The augmented instructions are checked with the defer bit both 1 and 0.

The programs are numbered octally (1-20). Program 1 clears memory locations 0000-7766, but does not test any instructions. Programs 2-20 test every instruction at least once before it is used; in general, an instruction is not used within the program which tests it.

A RIM loader is read in together with Program 1 and remains in locations 7772-7777 throughout the entire Instruction Test. This short loading routine is used to read in Programs 2-20. However, if the loader fails to operate properly, read in mode may be used instead. Sense switches 1 and 2 control the execution of Programs 3-20. With SS1 on, the program halts after read in. With SS2 on, the program iterates. With both switches off, each program is read in and executed once.

CHAPTER 1

CONSOLE OPERATING PROCEDURE

The tables below describe the console operating procedure to be used when running the Instruction Test program.

TABLE 1-1 TAPES REQUIRED FOR TEST

Instruction Test program tape.

TABLE 1-2 SWITCHES

Switch	Setting	Function
SENSE SWITCH 1 (Used in Programs 3 through 20)	0	Program is executed after read in.
	1	Program halts after completion of read in.
SENSE SWITCH 2 (Used in Programs 3 through 20)	0	Program is executed once and the next program is read in.
	1	Program iterates until this switch is turned off.
TEST WORD (Used in Program 3)	777777	If the contents of the TEST WORD switches are not shown, the computer halts at Errhlt 3 of Program 3.

TABLE 1-3 LOAD SEQUENCE

a) Load the Instruction Test tape into reader.

b) Turn off all SENSE SWITCHES.

c) Begin reading the tape using read-in mode, i.e. push down on the READ IN switch (refer to PDP-1 Maintenance Manual, paragraph 5-6a). The computer should read in Program 1, execute it, then read in Program 2 and halt with MA equal to 0001.

d) Push CONTINUE switch down. The jmp and szs test runs until operator intervenes or until an error halt occurs (running time equals 100 μsec per iteration).

e) Push STOP switch down.

f) Turn on all SENSE SWITCHES.

g) Set ADDRESS switches to 0032.

h) Push down on START switch. The szs test runs until operator intervenes or until an error halt occurs (running time equals 75 μsec per iteration).

i) Push STOP switch down.

j) Set the TEST WORD switches to 777777 (all on).

k) Set SS1 (SENSE SWITCH 1) to desired position. If on, each program halts upon completion of read-in; if off, program is executed after read-in.

l) Set SS2 to desired position. If on, the next program which is read in iterates until SS2 is turned off. If off, each program is executed once and then the next program is read in.

m) Read in Program 3: (1) Use read-in mode, i.e. push READ IN switch down; or, alternatively, (1) Set ADDRESS switches to 7772 and (2) Push START switch down.

n) If SS1 and SS2 are both off, the remaining programs are read in and executed in sequence. Upon completion of the Instruction Test the computer halts with: PC equal 0001; MA equal 0000; MB equal 0020 (number of last program); AC equal 000777; IO equal 777000 and all program flags on.

TABLE 1-4 PROGRAM 1 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
Anyhlt	not relevant	Not a programmed halt.

TABLE 1-5 PROGRAM 1 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
Anyhlt	If cause of halt is not apparent, skip this program. (The remaining programs may have to be read in by means of the READ IN switch.)

TABLE 1-6 PROGRAM 2 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
Halt	0001	Not an error halt. This is the test for hlt instruction
Errhlt 1	0003	The jmp instruction located in 0002 failed to execute the jump.
Errhlt 2	0006	szs 10 or szs ' 10 error with SS1 off.
Errhlt 3	0011	szs 20 or szs ' 20 error with SS2 off.
Errhlt 4	0014	szs 30 or szs ' 30 error with SS3 off.

TABLE 1-6 PROGRAM 2 ERROR HALTS

(continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 5	0017	szs 40 or szs ' 40 error with SS4 off.
Errhlt 6	0022	szs 50 or szs ' 50 error with SS5 off.
Errhlt 7	0025	szs 60 or szs ' 60 error with SS6 off.
Errhlt 8	0030	szs 70 or szs ' 70 error with all SS's off.
Errhlt 9	0034	szs ' 10 or szs 10 error with SS1 on.
Errhlt 10	0037	szs ' 20 or szs 20 error with SS2 on.
Errhlt 11	0042	szs ' 30 or szs 30 error with SS3 on.
Errhlt 12	0045	szs ' 40 or szs 40 error with SS4 on.
Errhlt 13	0050	szs ' 50 or szs 50 error with SS5 on.
Errhlt 14	0053	szs ' 60 or szs 60 error with SS6 on.
Errhlt 15	0056	szs ' 70 or szs 70 error with all SS's on.
Errhlt 16	3001	The jmp instruction located in 3000 failed to execute the jump.
Errhlt 17	5000	The jmp instruction located in 4777 failed to execute the jump.
Errhlt 18	6001	The jmp instruction located in 6000 failed to execute the jump.
Other	any other	If all the sense switches are off, this halt was probably caused by incorrect execution of a jmp instruction, i.e., by a jump to the wrong address. If all the sense switches are on, then this is not a programmed halt.

TABLE 1-7 PROGRAM 2 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
Halt	The test for jmp is ready to start. Push CONTINUE.
Errhlt 1	Record the contents of PC. To restart program set ADDRESS switches to 0002. Push START.
Errhlt 2-8	Check to make sure that all sense switches are off. Set the ADDRESS switches to 0002. Push START.
Errhlt 9-15	Check to make sure that all sense switches are on. Set the ADDRESS switches to 0032. Push START.
Errhlt 16-18	Set the ADDRESS switches to 0002. Turn on the SINGLE INST. switch. Push START and trace the program. The first four instructions executed constitute the jmp test.
Other	If sense switches off -- (same as Errhlt 16-18 above). If sense switches on -- Try restarting the program at 0032 using START. If this doesn't work, try reloading the program.

TABLE 1-8 PROGRAM 3 ERROR HALTS

Error No.	Contents of MA	Contents of AC	Cause of Error Halt
SS1	0002		Not an error halt. The test for skp is ready to start. Location 0000 contains program number.
Errhlt 1	0004		The 650000 instruction failed to skip.
Errhlt 2	0006		The 654000 instruction failed to skip.
Errhlt 3	0011		The 640000 instruction skipped.
Errhlt 4	0014		The 644000 instruction skipped.
Errhlt 5	0020	000000 not 000000	The sza instruction failed to skip. The cla instruction failed to clear AC.
Errhlt 6	0024	000000 not 000000	The sza instruction failed to skip. The lat instruction failed to load AC with all 1s (are all TEST WORD switches on?); or the cma instruction failed to complement whichever bits are not zero.
Errhlt 7	0026	000000	The spa instruction failed to skip.
Errhlt 8	0030	000000	The sma ' instruction failed to skip.
Errhlt 9	0033	*777777* not *777777*	The sza ' instruction failed to skip. The cma instruction failed to complement properly.
Errhlt 10	0035	*777777*	The spa ' instruction failed to skip.
Errhlt 11	0037	*777777*	The sma instruction failed to skip.
Other	any other		Not a programmed halt.

TABLE 1-9 PROGRAM 3 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	Program is ready to start. Make sure all TEST WORD switches are on. Push CONTINUE.
Errhlt 1-4	Set ADDRESS switches to 0003 and push START. Program restarts.
Errhlt 5-11	Record contents of AC. Set ADDRESS switches to 0003. Push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-10 PROGRAM 4 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	Not an error halt. The test for xor is ready to start.
Errhlt 1	0006	Incorrect execution of xor. The exclusive-OR of 777777 with 777777 was attempted.
Errhlt 2	0011	Incorrect execution of xor. The exclusive-OR of 000000 with 000000 was attempted.
Errhlt 3	0015	Incorrect execution of xor. The exclusive-OR of 000000 (in the AC) with 777777 was attempted.
Errhlt 4	0022	Incorrect execution of xor. The exclusive-OR of 777777 (in the AC) with 000000 was attempted.
Errhlt 5	0024	The sas failed to skip when comparing equal numbers (000000).
Errhlt 6	0026	The sas skipped properly but failed to replace the contents of the AC. The AC initially contained 000000.
Errhlt 7	0030	The sad failed to skip when comparing unequal numbers.
Errhlt 8	0032	The sad skipped properly but failed to replace the contents of the AC. The AC initially contained 000000.
Errhlt 9	0036	The sas skipped when comparing unequal numbers.
Errhtl 10	0041	The sas rightly did not skip, but then failed to replace the contents of the AC. The AC initially contained 777777.
Errhlt 11	0045	The sad skipped when comparing equal numbers.
Errhlt 12	0050	The sad rightly did not skip, but then failed to replace the contents of the AC. The AC initially contained 777777.
Other	any other	Not a programmed halt.

TABLE 1-11 PROGRAM 4 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1-12	Record contents of the AC. Turn on SS2. Set ADDRESS switches to 0003. Push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-12 PROGRAM 5 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	Not an error halt. The test for dac, dap, dip is ready to start.
Errhlt 1	0007	Incorrect execution of dap and/or dip. All 0s should have been deposited into location 0063. Any 1s in bits 0-5 imply a dip error; any 1s in bits 6-17 imply a dap error.
Errhlt 2	0012	Incorrect execution dac. All 0s should have been deposited into location 0064.
Errhlt 3	0017	Incorrect execution of dap and/or dip. All 1s should have been deposited into location 0063. Any 0s in bits 0-5 imply a dip error; any 0s in bits 6-17 imply a dap error.
Errhlt 4	0022	Incorrect execution of dac. All 1s should have been deposited into location 0064.
Errhlt 5	0026	(Same as Errhlt 3)
Errhlt 6	0031	(Same as Errhlt 4)
Errhlt 7	0036	(Same as Errhlt 1)
Errhlt 8	0041	(Same as Errhlt 2)
Errhlt 9	0044	Incorrect execution of lac. All 0s should have been loaded into the AC.
Errhlt 10	0047	Incorrect execution of lac. All 1s should have been loaded into the AC.
Errhlt 11	0052	(Same as Errhlt 10)
Errhlt 12	0055	(Same as Errhlt 9)
Other	any other	Not a programmed halt.

TABLE 1-13 PROGRAM 5 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1, 3, 5,7	Record the contents of location 0063. Turn on SS2. Push CONTINUE. If the trouble is in the jam transfer then error halts also occur for dac and lac.
Errhlt 2, 5, 6, 8	Record the contents of location 0064. Turn on SS2. Push CONTINUE. If the trouble is in the jam transfer then error halts also occur for dip, dap and lac.
Errhlt 9-12	Record the contents of the AC. Turn on SS2. Push CONTINUE. If the trouble is in the jam transfer then error halts also occur for dac, dap and dip.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-14 PROGRAM 6 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	Not an error halt. The test for dzm is ready to start.
Errhlt 1	0010	The dzm instruction failed to clear location 0166 which contained all 1s.
Errhlt 2	0014	The dzm instruction failed to clear location 0166 which contained all 0s.
Errhlt 3	0021	Carry chain in the AC failed to propagate the length of the register. AC should contain 000001.
Errhlt 4	0023	End-around carry in the AC failed to operate. AC should contain 000001.
Errhlt 5	0030	Contents of AC incorrect after indexing. AC should contain 000000.
Errhlt 6	0035	Carry chain did not stop at bit 17. AC should contain 377777.
Errhlt 7	0042	Carry chain did not stop at bit 16. AC should contain 377776.
Errhlt 8	0047	Carry chain did not stop at bit 15. AC should contain 377774.
Errhlt 9	0054	Carry chain did not stop at bit 14. AC should contain 377770.
Errhlt 10	0061	Carry chain did not stop at bit 13. AC should contain 377760.
Errhlt 11	0066	Carry chain did not stop at bit 12. AC should contain 377740.
Errhlt 12	0073	Carry chain did not stop at bit 11. AC should contain 377700.
Errhlt 13	0100	Carry chain did not stop at bit 10. AC should contain 377600.
Errhlt 14	0105	Carry chain did not stop at bit 9. AC should contain 377400.
Errhlt 15	0112	Carry chain did not stop at bit 8. AC should contain 377000.

TABLE 1-14 PROGRAM 6 ERROR HALTS

(continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 16	0117	Carry chain did not stop at bit 7. AC should contain 376000.
Errhlt 17	0124	Carry chain did not stop at bit 6. AC should contain 374000.
Errhlt 18	0131	Carry chain did not stop at bit 5. AC should contain 370000.
Errhlt 19	0136	Carry chain did not stop at bit 4. AC should contain 360000.
Errhlt 20	0143	Carry chain did not stop at bit 3. AC should contain 340000.
Errhlt 21	0150	Carry chain did not stop at bit 2. AC should contain 700000.
Errhlt 22	0155	Carry chain did not stop at bit 1. AC should contain 600000.
Errhlt 23	0162	Carry chain did not stop at bit 0. AC should contain 400000.
Other	any other	Not a programmed halt.

TABLE 1-15 PROGRAM 6 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	Check SS2 for desired setting. Push CONTINUE.
Errhlt 1-2	Record the contents of location 0166. Turn on SS2. Set the ADDRESS switches to 0003. Push START.
Errhlt 3-23	Record the contents of the AC. Turn on SS2. Set the ADDRESS switches to 0003. Push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-16 PROGRAM 7 ERROR HALTS

Error No.	Contents of MA	Contents of Memory Location	Cause of Error Halt
SS1	0002		Not an error halt. The test for isp is ready to start.
Errhlt 1	0012	0050 = +	The isp failed to skip on a positive number.
		0050 = -	The isp incorrectly indexed location 0050. The correct number equals 1 more than the contents of location 0046.
Errhlt 2	0015		The isp incorrectly indexed location 0050. The AC and location 0046 both contain the correct number.
Errhlt 3	0022	0047 = +	The isp incorrectly indexed location 0046. The correct number equals 1 more than the contents of location 0044.
		0047 = -	The isp skipped on a negative number.
Errhlt 4	0025		The isp incorrectly indexed location 0047. The AC and location 0045 both contain the correct number.
Errhlt 5	0030	0047 = +	The isp failed to skip on a positive number.
		0047 = -	The isp incorrectly indexed the number 777776 (contained in location 0047).
Errhlt 6	0032		The isp incorrectly indexed the number 777776 (contained in location 0047).
Errhlt 7	0035	0050 = +	The isp incorrectly indexed the number 377777 (contained in location 0050).
		0050 = -	The isp skipped on a negative number.

TABLE 1-16 PROGRAM 7 ERROR HALTS

(continued)

Error No.	Contents of MA	Contents of Memory Location	Cause of Error Halt
Errhlt 8	0040		The isp incorrectly indexed the number 377777 (contained in location 0050).
Other	any other		Not a programmed halt.

TABLE 1-17 PROGRAM 7 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1, 2, 7, 8	Record the contents of locations 0046 and 0050. Turn on SS2. Set the ADDRESS switches to 0003. Push START. This restarts the program.
Errhlt 3, 4, 5, 6	Record the contents of locations 0045 and 0047. Turn on SS2. Set the ADDRESS switches to 0003. Push START. This restarts the program.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-18 PROGRAM 10 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	This is not an error halt. The test for and is ready to start.
Errhlt 1	0006	Incorrect execution of 777777 and 777777. The correct contents of the AC equal 777777.
Errhlt 2	0012	Incorrect execution of 000000 and 777777. The correct contents of the AC equal 000000.
Errhlt 3	0016	Incorrect execution of 000000 and 000000. The correct contents of the AC equal 000000.
Errhlt 4	0022	Incorrect execution of 777777 and 000000. The correct contents of the AC equal 000000.
Errhlt 5	0026	Incorrect execution of 000000 ior 000000. The correct contents of the AC equal 000000.
Errhlt 6	0032	Incorrect execution of 000000 ior 777777. The correct contents of the AC equal 777777.
Errhlt 7	0036	Incorrect execution of 777777 ior 777777. The correct contents of the AC equal 777777.
Errhlt 8	0042	Incorrect execution of 777777 ior 000000. The correct contents of the AC equal 777777.
Other	any other	Not a programmed halt.

TABLE 1-19 PROGRAM 10 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start push CONTINUE. (If SS1 is left on, the program stops at this same location after read-in.)
Errhlt 1-8	Record the contents of the AC. Turn on SS2. Push CONTINUE. (Note that the and effects an $MB \xrightarrow{0} AC$ while the ior effects $MB \xrightarrow{1} AC$.)
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-20 PROGRAM 11 ERROR HALTS

Error No.	Contents of MA	Contents of Register	Cause of Error Halt
SS1	0002		Not an error halt. The test for lio, dio and spi is ready to start.
Errhlt 1	0010	C(AC) = C(IO)	Incorrect execution of dio. Should have deposited the IO in location 0034.
		C(AC) ≠ C(IO)	Incorrect execution of lio. Should have loaded IO with the contents of location 0033 (which is equal to the AC).
Errhlt 2	0014		The spi failed to skip on a positive IO.
Errhlt 3	0020	C(AC) = C(IO)	(Same as Errhlt 1)
		C(AC) ≠ C(IO)	(Same as Errhlt 1)
Errhlt 4	0025		The spi ' failed to skip on a negative IO.
Other			Not a programmed halt.

TABLE 1-21 PROGRAM 11 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1, 3	Record the contents of the AC and the IO. Turn on SS2. Set the ADDRESS switches to 0003. Push START. (CONTINUE may be used instead of the restart procedure described above. However, an invalid error halt may occur for spi if the lio loaded in the wrong sign.)
Errhlt 2, 4	Record the contents of the AC. Turn on SS2. Push CONTINUE.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-22 PROGRAM 12 ERROR HALTS

Error No.	Contents of MA	Overflow	Cause of Error Halt
SS1	0002		Not an error halt. The test for add and szo is ready to start.
Errhlt 1	0006	off	The szo instruction failed to skip on no overflow.
		on	The szo instruction failed to clear OVERFLOW.
Errhlt 2	0011	off	The szo ' instruction skipped on no overflow.
		on	The szo and szo ' instructions failed to clear OVERFLOW.
Errhlt 3	0015		Incorrect execution of 000000 add 000000. Correct contents of AC equal 000000.
Errhlt 4	0020		Incorrect execution of 000000 add 377777. Correct contents of AC equal 377777.
Errhlt 5	0023		Incorrect execution of 377777 add 000000. Correct contents of AC equal 377777.
Errhlt 6	0027		Incorrect execution of 000000 add 400000. Correct contents of AC equal 400000.
Errhlt 7	0032		Incorrect execution of 400000 add 000000. Correct contents of AC equal 400000.
Errhlt 8	0034	off	OVERFLOW was incorrectly set or szo failed to skip with no overflow.
		on	OVERFLOW was incorrectly set and szo also failed to clear it.
Errhlt 9	0037		Incorrect execution of 400000 add 377777. Correct contents of AC equal 000000.

TABLE 1-22 PROGRAM 12 ERROR HALTS

(continued)

Error No.	Contents of MA	Overflow	Cause of Error Halt
Halt 10	0043		Full-register carry failed. The number 252525 was added to itself. The correct contents of AC equal 525252.
Halt 11	0045	0	OVERFLOW was not set and/or szo ' failed to skip on overflow.
		1	The szo ' failed to skip on overflow and in addition failed to clear OVERFLOW.
Halt 12	0051		Full-register carry failed. The number 125252 was added to itself. The correct contents of the AC equal 252524.
Halt 13	0055		Clear-AC-on-minus-zero failed. The correct contents of the AC equal 1000000.
Halt 14	0057	0	OVERFLOW incorrectly set or szo failed to skip on no overflow.
		1	OVERFLOW incorrectly set and in addition the szo failed to clear it.
Halt 15	0063		Ripple carry failed to propagate properly. Correct contents of AC equal 000001.
Halt 16	0065	0	OVERFLOW incorrectly set or szo failed to skip on no overflow.
		1	OVERFLOW incorrectly set and in addition szo failed to clear it.
Halt 17	0071		Ripple carry failed to initiate properly in some bit. Correct contents of AC equal 252525.

TABLE 1-22 PROGRAM 12 ERROR HALTS

(continued)

Error No.	Contents of MA	Overflow	Cause of Error Halt
Errhlt 18	0075		Ripple carry failed to initiate properly in some bit. Correct contents of AC equal 525252.
Errhlt 19	0077	0	OVERFLOW incorrectly set or szo failed to skip on no overflow.
		1	OVERFLOW incorrect set and in addition szo failed to clear it.
Errhlt 20	0107		Incorrect execution of sub. Location 0127 contains the correct result of the operation. (Minus 1 was subtracted from a number, which equalled 1 less than the contents of location 0127.)
Other	any other		Not a programmed error halt.

TABLE 1-23 PROGRAM 12 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1, 2, 8, 11, 14, 16-19	On any overflow error, first make sure that the szo instruction operates correctly (see a below) then start the program at the appropriate address (see b below) and step through it, with the SINGLE INST. switch on, until the PC equals the address of the Errhlt. a Turn on SS2; turn on the SINGLE INST. switch; set the overflow flip-flop. (To turn on the overflow flip-flop: set the ADDRESS switches to 0042, leave the SINGLE INST. switch on, push START, push CONTINUE. At this point the OVERFLOW light should be on.) Set the ADDRESS switches to 0002;

TABLE 1-23 PROGRAM 12 POST-ERROR RESTART PROCEDURE (continued)

<table>
<tr><th>Error No.</th><th>Procedure</th></tr>
<tr><td></td><td>push START. Using CONTINUE step through the program until the MEMORY ADDRESS reads 0012.

The correct sequence is:
START--MA = 0002, OVERFLOW is on; CONTINUE--MA = 0003, OVERFLOW is off; CONTINUE through MA = 0004, 0005, 0007, 0010 and 0012. (MA = 0006 or 0011 denotes errors in the instruction.)

b Set the ADDRESS switches. Push START.

<table>
<tr><th>Errhlt</th><th>MA</th><th>Remarks</th></tr>
<tr><td>1,2</td><td></td><td>Nothing but overflow check of a.</td></tr>
<tr><td>8</td><td>0012</td><td>The OVERFLOW should not turn on.</td></tr>
<tr><td>11</td><td>0035</td><td>OVERFLOW turns on when MA equals 0043.</td></tr>
<tr><td>14</td><td>0046</td><td>The OVERFLOW should not turn on.</td></tr>
<tr><td>16</td><td>0060</td><td>OVERFLOW should not turn on.</td></tr>
<tr><td>19</td><td>0066</td><td>OVERFLOW should not turn on.</td></tr>
</table>
</td></tr>
<tr><td>Errhlt 3, 4, 5, 6, 7, 9, 10, 12, 13, 15, 17, 18</td><td>Record the contents of the AC. Turn on SS2; set ADDRESS switches to 0003; push START.</td></tr>
<tr><td>Other</td><td>Location 0000 contains the number of the program that is in the com-computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.</td></tr>
</table>

TABLE 1-24 PROGRAM 13 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	This is not an error halt. The test for cli is ready to start.
Errhlt 1	0006	The law instruction failed to clear the AC. With the AC all 1s, 0 was loaded (law) into it. The correct contents of the AC equal 000000.
Errhlt 2	0011	The law incorrectly executed the MB_{6-17} AC. The correct contents of the AC equal 007777.
Errhlt 3	0015	The law-0 instruction was incorrectly executed. The correct contents of the AC equal 777777. If the contents of the AC equal 000000 then law ' is not complementing the AC, i.e. not sensing the defer bit correctly.
Errhlt 4	0020	The law -7777 instruction was incorrectly executed. The correct contents of the AC equal 770000.
Errhlt 5	0030	The cli instruction failed to clear the IO (which was all 1s).
Errhlt 6	0035	The cli instruction failed to clear the IO (which was all 0s).
Other	any other	This is not a programmed halt.

TABLE 1-25 PROGRAM 13 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1	Record the contents of the AC. Turn on SS2; set the ADDRESS switches to 0003; push START.
Errhlt 2	Record the contents of the AC. Turn on SS2; push CONTINUE. If the computer skips Errhlt 3 and stops at Errhlt 4, then the trouble is definitely in the MB^{1}_{6-17} AC transfer.
Errhlt 3, 4	Record the contents of the AC. Turn on SS2; set the ADDRESS switches to 0003; push START.
Errhlt 5, 6	Record the contents of the IO and check to see that IO equals AC (since it is actually the AC that is checked for zero). Turn on SS2; set the ADDRESS switches to 0003; push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-26 PROGRAM 14 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	This is not an error halt. The test for the clf, szf, stf is ready to start.
Errhlt 1	0005	All program flags off: szf 7 failed to skip. Any flag on: clf 7 failed to clear all flags.
Errhlt 2	0022	Bits 15-17 of location 0021 contain the number f (f is octal). Flag f off: stf f failed to set the flag. Flag f on: szf ' f failed to skip.
Errhlt 3	0033	(Same as Errhlt 1).
Errhlt 4	0040	Any program flag off: stf 7 failed to set all flags. Any program flag on: szf ' 7 failed to skip.
Errhlt 5	0047	Bits 15-17 of location 0046 contain the number f (f is octal). Flag f off: szf f failed to skip. Flaf f on: clf f failed to clear flag.
Other	any other	Not a programmed halt.

TABLE 1-27 PROGRAM 14 POST-ERROR REST .RT PROCEDURE

Error No.	Proce .re
SS1	To start test push CONTINUE. (If S1 is left on, the next program stops at this same location aft read-in.)
ErrHlt 1-5	Record which instruction caused the error. Turn on SS2; set the ADDRESS switches to 0003; push ST RT. (To change the speed of the program, alter the contents of l cation 0070. The number in this location is indexed until positi .)
Other	Location 0000 contains the number the program that is in the computer. Make sure that this num er corresponds to the program number of the error table that you c ecked. -- Set the ADDRESS switches to 0003 and push START dc n. If the halt persists, try reloading the program.

TABLE 1-28A PROGRAM 15 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	Not an error halt. Test for jsp is ready to start.
Errhlt 1	0105	The cal instruction failed to save the contents of PC. (See Table 1-28D).
Errhlt 2	0110	The cal instruction failed to save the contents of AC in 0100. (See Table 1-28D).
Errhlt 3	0744	The first jda instruction failed to save the contents of PC. (See Table 1-28C).
Errhlt 4	0747	The first jda instruction failed to save the AC. (See Table 1-28C).
Errhlt 5	0763	The first jsp instruction failed to save the PC. (See Table 1-28B).
Errhlt 6	1005	The third jda instruction failed to save the PC. (See Table 1-28C).
Errhlt 7	1010	The third jda instruction failed to save the AC. (See Table 1-28C).
Errhlt 8	1104	The third jsp instruction failed to save the PC. (See Table 1-28B).
Errhlt 9	6024	The second jsp instruction failed to save the PC. (See Table 1-28B).

TABLE 1-28A PROGRAM 15 ERROR HALTS (continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 10	6046	The second jda instruction failed to save the PC and/or failed to save the contents of OVERFLOW in AC_0. (See Table 1-28C).
Errhlt 11	6051	The second jda instruction failed to save the AC. (See Table 1-28C).
Other	any other	Probably due to incorrect execution of cal, jsp or jda, i.e. a jump to the wrong address.

TABLE 1-28B PROGRAM 15 - LOCATIONS RELEVANT TO jsp TRANSFERS

	First jsp		Second jsp		Third jsp	
	AC	PC	AC	PC	AC	PC
before jsp	777757	0020	776020	1757	771677	6100
during jsp:						
$\xrightarrow{0}$ AC	000000		000000		000000	
PC $\xrightarrow{1}$ AC	000020		001757		006100	
$\xrightarrow{0}$ PC		0000		0000		0000
MB $\xrightarrow{1}$ PC		0757		6020		1100
after jsp	000020	0760	001757	6021	006100	1101

TABLE 1-28C PROGRAM 15 - LOCATIONS RELEVANT TO jda TRANSFERS

	First jda			Second jda		
	AC	Location 0737	PC	AC	Location 6040	PC
before jda	771737	000040	0040	006040	771737	1737
during jda:						
AC $\xrightarrow{i}$ MB		771737			006040	
$\xrightarrow{0}$ AC	000000			000000		
PC $\xrightarrow{1}$ AC	000040			401737		
$\xrightarrow{0}$ PC			0000			0000
MA $\xrightarrow{1}$ PC			0737			6040
$\xrightarrow{+1}$ PC			0740			6041
after jda	000040	771737	0741	401737	6040	6042

TABLE 1-28C PROGRAM 15 - LOCATIONS RELEVANT TO jda TRANSFERS

(continued)

	Third jda		
	AC	Location 1000	PC
before jda	771777	006000	6000
during jda:			
AC → MB		771777	
0 → AC	000000		
PC 1 → AC	006000		
0 → PC			0000
MA 1 → PC			1000
+1 → PC			1001
after jda	006000	771777	1002

TABLE 1-28D PROGRAM 15 - LOCATIONS RELEVANT TO cal TRANSFERS

	cal			
	AC	Location 0100	PC	MA
before cal	777777	------	3002	------
during cal:				
100 $\xrightarrow{1}$ MA				000100
AC $\xrightarrow{i}$ MB		777777		
$\xrightarrow{0}$ AC	000000			
PC $\xrightarrow{1}$ AC	003002			
$\xrightarrow{0}$ PC			0000	
MA $\xrightarrow{1}$ PC			0100	
$\xrightarrow{+1}$ PC			0101	
after cal	003002	777777	0102	000100

TABLE 1-29 PROGRAM 15 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	The test for jsp is ready to start. Push CONTINUE.
Errhlts 1, 3, 5, 6, 8, 9, 10	Record the contents of AC. Turn on SS2 and push CONTINUE. Use Tables 1-28B, 1-28C and 1-28D (Program 15 error halts) to determine which transfers are not operating properly.
Errhlt 2	Record the contents of 0100. Turn on SS2 and push CONTINUE. Use Table 1-28D (Program 15 error halts) to determine which transfers are not operating properly.
Errhlt 4	Record the contents of 0737. Turn on SS2 and push CONTINUE. Use Table 1-28C (Program 15 error halts) to determine which transfers are not operating properly.
Errhlt 7	Record the contents of 1000. Turn on SS2 and push CONTINUE. Use Table 1-28C (Program 15 error halts) to determine which transfers are not operating properly.
Errhlt 11	Record the contents of 6040. Turn on SS2 and push CONTINUE. Use Table 1-28C (Program 15 error halts) to determine which transfers are not operating properly.
Other	Record the contents of PC and AC. Turn on SS2; set the ADDRESS switches to 0003 and push START. Use Tables 1-28B, 1-28C and 1-28D (Program 15 error halts) to determine which of the jump instructions (jsp, jda, cal) failed to operate properly.

TABLE 1-30 PROGRAM 16 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
SS1	0002	Not an error halt. The test for nop is ready to start.
Errhlt 1	0003	The nop was not executed, i.e. the computer stopped.
Errhlt 2	0005	The xct instruction was incorrectly executed.
Errhlt 3	0020	The indirect addressing was not executed correctly. Make sure you were not in the extend mode. (The test uses 5 levels of indirect addressing, but the extend mode allows only 1 level.)
Other	any other	Not a programmed halt.

TABLE 1-31 PROGRAM 16 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	To start test push CONTINUE. (If SS1 is left on, the next program stops at this same location after read-in.)
Errhlt 1	Turn on SS2; set the ADDRESS switches to 0003; push START.
Errhlt 2	Turn on SS2; turn on the SINGLE INST. switch; set the ADDRESS switches to 0004; push START. The program starts by executing the xct instruction.
Errhlt 3	Turn on SS2; turn on the SINGLE INST switch; set the ADDRESS switches to 0015; push START. The first instruction clears AC. The instruction is the lac ' (defer bit on) which caused the halt.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-32 PROGRAM 17 ERROR HALTS

Error No.	Contents of MA	Cause of Error
SS1	0002	Not an error halt. Test for ral and ril is ready to start.
Errhlt.1	0017	The ral s1 instruction (in location 0014) failed. The correct contents of the AC are in location specified by the address of the sas instruction in 0016.
Errhlt 2	0022	The ril s1 instruction (in location 0015) failed. The correct contents of the IO are in the AC.
Errhlt 3	0053	The rar s1 instruction (in location 0050) failed. The correct contents of the AC are in the location specified by address of the sas instruction in 0052.
Errhlt 4	0056	The rir s1 instruction (in location 0051) failed. The correct contents of the IO are in the AC.
Errhlt 5	0102	The ral s9 instruction failed to rotate the AC correctly. The correct contents of the AC equal 070777.
Errhlt 6	0105	The ril s9 instruction failed to rotate the IO nine bits. The correct contents of the IO equal 070777.
Errhlt 7	0111	The rar instruction failed to rotate the AC nine bits. The correct contents of the AC equal 777070.
Errhlt 8	0114	The rir instruction failed to rotate the IO nine bits. The correct contents of the IO equal 777070.

TABLE 1-32 PROGRAM 17 ERROR HALTS (continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 9	0120	The rcl instruction failed to rotate the combined registers nine bits. The contents of the AC are incorrect. The AC should contain 777777.
Errhlt 10	0124	The rcl instruction failed to rotate the combined registers nine bits. The contents of the IO are incorrect. The IO should contain 070070.
Errhlt 11	0130	The rcr instruction failed to rotate the combined registers nine bits. The contents of the AC are incorrect. The AC should contain 070777.
Errhlt 12	0134	The rcr instruction failed to rotate the combined registers nine bits. The contents of the IO are incorrect. The IO should contain 777070.
Errhlt 13	0140	The ral instruction rotated the AC although no rotation was specified. The correct contents of the AC equal 777070.
Errhlt 14	0143	The ril instruction rotated the IO although no rotation was specified. The correct contents of the IO equal 777070.
Errhlt 15	0147	The rar instruction rotated the AC although no rotation was specified. The correct contents of the AC equal 777070.
Errhlt 16	0152	The rir instruction rotated the IO although no rotation was specified. The correct contents of the IO equal 777070.

TABLE 1-32 PROGRAM 17 ERROR HALTS (continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 17	0155	The rcl instruction changed the contents of the AC although no rotation was specified. The correct contents of the AC equal 777070.
Errhlt 18	0160	The rcl instruction changed the contents of the IO although no rotation was specified. The correct contents of the IO equal 777070.
Errhlt 19	0163	The rcr instruction changed the contents of the AC although no rotation was specified. The correct contents of the AC equal 777070.
Errhlt 20	0166	The rcr instruction changed the contents of the IO although no rotation was specified. The correct contents of the IO equal 777070.
Errhlt 21	0203	The rcl instruction failed during execution of a 36-bit rotation. The contents of the AC are incorrect. The correct contents of AC are in the location specified by the address of the lio instruction in 0173.
Errhlt 22	0206	The rcl instruction failed during execution of a 36-bit rotation. The contents of IO are incorrect. The correct contents of IO are in the location specified by the address of the lio instruction in 0173 (the correct contents of IO are also in AC unless Errhlt 21 occurred).

TABLE 1-32 PROGRAM 17 ERROR HALTS (continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 23	0222	The rcr instruction failed during execution of a 36-bit rotation. The contents of the AC are incorrect. The correct contents of the AC are in the location specified by the address of the lio instruction in 0173.
Errhlt 24	0225	The rcr instruction failed during execution of a 36-bit rotation. The contents of IO are incorrect. The correct contents of IO are in the location specified by the address of the lio instruction in 0173 (the correct contents of IO are also in AC unless Errhlt 23 occurred).
Errhlt 25	0245	Failure to execute a series of eight rcr and rcl instructions, for a total of 72 bits rotation. The contents of the AC are incorrect. The correct contents of the AC are in the location specified by the address of the lio instruction in 0173.
Errhlt 26	0250	Failure to execute a series of eight rcr and rcl instructions for a total of 72 bits rotation. The contents of the IO are incorrect. The correct contents of the IO are in the location specified by the address of the lio instruction in 0173.
Other	any other	Not a programmed halt.

TABLE 1-33 PROGRAM 17 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
SS1	The test for ral and ril is ready to start. Push CONTINUE.
Errhlts (all)	Record the contents of AC and/or IO (whichever is appropriate). Turn on SS2, set the ADDRESS switches to 0003 and push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

TABLE 1-34 PROGRAM 20 ERROR HALTS

Error No.	Contents of MA	Cause of Error Halt
EOT	0000	If PC equals 0001, MA equals 0000, MB equals 000020, AC equals 000777, IO equals 777000, and all program flags are on, then the Instruction Test is completed. If not, then refer to Other under the Error Halts.
SS1	0002	Not an error halt. Test for sal and sil is ready to start.
Errhlt 1	0012	The sal instruction failed to shift the AC correctly 17 bits. The initial contents of AC were 377777; the correct contents of AC are 000000.
Errhlt 2	0015	The sil instruction failed to shift the IO correctly 17 bits. The correct contents of the IO are 000000 (the initial contents of the IO were 377777).
Errhlt 3	0025	The sar instruction failed to shift the AC correctly 17 bits. The correct contents of the AC equal 000000 (the initial contents of the AC were 377777).
Errhlt 4	0030	The sir instruction failed to shift the IO correctly 17 bits. The correct contents of the IO equal 000000 (the initial contents of the IO were 377777).
Errhlt 5	0041	The sar instruction failed to shift the AC correctly 17 bits. The correct contents of the AC equal 777777 (the initial contents of the AC were 400000).

TABLE 1-34 PROGRAM 20 ERROR HALTS (continued)

Error No.	Contents of MA	Cause of Error Halt
Errhlt 6	0045	The sir instructions failed to shift the IO correctly 17 bits. The correct contents of the IO equal 777777 (the initial contents of the IO were 400000).
Errhlt 7	0052	The scl instruction failed to shift the registers correctly. The contents of the AC are incorrect. The correct contents of the AC equal 377776. The initial contents of the AC and the IO were 377777.
Errhlt 8	0056	The scl instruction failed to shift the registers correctly. The contents of the IO are incorrect. The correct contents of the IO equal 777776. The initial contents of the AC and the IO were 377777.
Errhlt 9	0063	The scr instruction failed to shift the registers correctly. The contents of AC are incorrect. The correct contents of the AC equal 177777. The initial contents of the AC and the IO were 377777.
Errhlt 10	0067	The scr instruction failed to shift the registers correctly. The contents of IO are incorrect. The correct contents of IO equal 577777. The initial contents of AC and IO were 377777.
Other	any other	Not a programmed halt.

TABLE 1-35 PROGRAM 20 POST-ERROR RESTART PROCEDURE

Error No.	Procedure
EOT	End of the Instruction Test
SS1	The test for sal and sil is ready to start. Push CONTINUE.
Errhlts (all)	Record the contents of AC and/or IO (whichever is appropriate). Turn on SS2, set the ADDRESS switches to 0003 and push START.
Other	Location 0000 contains the number of the program that is in the computer. Make sure that this number corresponds to the program number of the error table that you checked. -- Set the ADDRESS switches to 0003 and push START down. If the halt persists, try reloading the program.

CHAPTER 2

SUGGESTED APPLICATION OF THE INSTRUCTION TEST PROGRAM

The four procedures described below provide useful methods for testing the PDP-1 instructions.

a FULL TEST PROCEDURE

1) Execute Programs 1 and 2.

2) Turn on SS2. Read in Program 3 and allow it to iterate.

3) Turn on SS1 and use the following sequence for the remaining programs (Programs 4 through 20):

i) Turn off SS2 (program currently executing, completes execution; next program is read in and the computer halts with MA equal 0002). Note that the number of the program just read is in location 0000.

ii) Turn on SS2.

iii) Push CONTINUE (the program executes and iterates until SS2 is turned off).

or

If the state of the computer has been changed after the halt at MA equal to 0002, then set the ADDRESS switches to 0003 and push down on START.

iv) Allow the program to iterate (Program iterates until SS2 is turned off).

v) Repeat steps (i) through (v) until the Instruction Test is complete.

b DAILY TEST PROCEDURE - Before beginning normal operation of the computer, ensure that the instructions are working correctly by executing the Instruction Test once (SS1 and SS2 both off for Programs 3-20).

c COMPUTER MALFUNCTIONS - Attempt to perform the full checkout (a above). If Program 1 fails to execute, skip it since it does not test any instructions. However, if the RIM loader fails to execute correctly then read in the remaining programs by means of the following sequence:

1) Read in Program 2 using the READ IN switch.

2) After executing Program 2, turn on SS2 and leave it on.

3) Push down STOP.

4) Push down on the READ IN switch (the next program is read in and executes repeatedly).

5) Allow the program to iterate.

6) Repeat steps 3) through 6) until the Instruction Test is completed.

d MARGIN CHECKS - Perform margin checks using the full test procedure (a above).

Detailed methods for checking margins are presented in paragraph 11-7b of the PDP-1 Maintenance Manual.

CHAPTER 3

PROGRAM DESCRIPTION

3-1 GENERAL

Instruction Test is a sequence of programs designed to test the operation of PDP-1 instructions. The test checks the following instructions: from the arithmetic group - add, sub, idx, isp; all the logical instructions; all the data handling instructions; all the shift/rotate group; all the skip group; and all the operate group (except lat, which is only partly tested).

The Instruction Test comprises a series of sixteen programs (octally numbered 1 through 20). The first program clears memory and locates a RIM loader in the high end of core. (The RIM loader is a short sequence which simulates the read-in mode normally controlled by computer hardware; the loader reads a paper tape in read-in mode format.) This first program of the instruction test does not have any error halts; the program assumes that the few instructions it uses are working properly. However, this program is not an essential part of the Instruction Test, and may be skipped if it doesn't run properly.

The remaining fifteen programs of the Instruction Test check the instructions listed above (refer to table 3-1). In general, a given instruction is not used within the same program that tests it; the few instructions used within the same program are used only after the program has already checked them. Every instruction is checked at least once before it is assumed to be working (refer to table 3-2).

TABLE 3-1

LIST OF INSTRUCTIONS TESTED BY MAINDEC 1 - INSTRUCTION TEST

(Each instruction is listed alphabetically with the number of the program which tests it.)

Instruction	Tested by Program	Instruction	Tested by Program	Instruction	Tested by Program
add	12	jmp	2	scl	20
and	10	jsp	15	scr	20
cal	15	lac	5	sil	20
cla	3	lat*	3	sir	20
clf	14	law	13	sma	3
cli	13	lio	11	spa	3
cma	3	nop	16	spi	11
dac	5	ral	17	stf	14
dap	5	rar	17	sub	12
dio	11	rcl	17	sza	3
dip	5	rcr	17	szf	14
dzm	6	ril	17	szo	12
hlt	2	rir	17	szs	2
idx	6	sad	4	xct	16
ior	10	sal	20	xor	4
isp	7	sar	20	650000	3
jda	15	sas	4	654000	3

* only partially tested

TABLE 3-2 INSTRUCTIONS TESTED OR USED WITHIN EACH PROGRAM OF MAINDEC 1 - INSTRUCTION TEST

(Programs are listed in numerical order. Each instruction is listed in the order of test under the program which tests it.)

Inst.	Checked by Prog.	Instructions Used Within Program																
		RIM	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	20
hlt	2			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
jmp	2	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
szs	2				x	x	x	x	x	x	x	x	x	x	x	x	x	x
cla	3					x	x	x			x	x	x			x		
cma	3					x	x	x				x	x					x
sma	3							x										
spa	3							x			x							
sza	3					x	x	x	x	x	x	x	x					x
654000	3																	
650000	3								x			x						
lat	3*				x													
xor	4																	
sas	4						x		x	x	x	x	x		x	x	x	x
sad	4								x								x	
dip	5																	
dap	5													x			x	
dac	5							x	x			x	x	x	x		x	
lac	5							x	x	x		x	x	x	x	x	x	x

*only partially tested

TABLE 3-2 INSTRUCTIONS TESTED OR USED WITHIN EACH PROGRAM OF MAINDEC 1 - INSTRUCTION TEST

(continued)

Inst.	Checked by Prog.	Instructions Used Within Program															
		RIM	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	20
dzm	6		x								x							
idx	6		x						x		x	x		x			x	
isp	7		x											x			x	
and	10														x		x	
ior	10														x			
lio	11												x				x	
dio	11	x											x				x	x
spi	11	x																
add	12																x	
szo	12														x			
sub	12																x	
law	13													x	x		x	
cli	13																	
clf	14																	
stf	14																	x
szf	14																	
jsp	15																	
jda	15																	
cal	15																	
nop	16																	
xct	16																	
ral	17																	

TABLE 3-2 INSTRUCTIONS TESTED OR USED WITHIN EACH PROGRAM OF MAINDEC 1 - INSTRUCTION TEST

(continued)

Inst.	Checked by Prog.	Instructions Used Within Program																
		RIM	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	20
ril	17																	
rar	17																	
rir	17																	
rcl	17																	
rcr	17																	
sal	20																	
sil	20																	
sar	20																	
sir	20																	
scl	20																	
scr	20																	
rpb	not tested	x																

3-2 RIM LOADER

The RIM loader, read in with Program 1, reads a tape in read-in mode format, thereby simulating computer read-in mode. The RIM loader occupies locations 7772 through 7777. The loader remains in core throughout the entire Instruction Test. It may be used to read in any or all of the other programs in the test. Sense switch 2 controls this option.

A tape in read-in mode format contains a series of instructions; the even numbered instructions make up the program to be stored in core; the odd numbered instructions are dio instructions, each having an address that determines the storage location for the next (even numbered) instruction. The last even numbered instruction of the tape may be followed by a jmp command. This jmp specifies the starting address for the stored program and, after read-in is completed, causes program execution to begin.

The RIM loader reads in an odd numbered instruction and, by sensing the sign of the IO, checks to determine if the instruction is a dio or a jmp. A +IO indicates that the instruction is dio (since the dio op code = 32); conversely, -IO indicates jmp (op code = 60). If the instruction is a dio, the loader reads in the next instruction and executes the dio. On the other hand, if the instruction is a jmp, the computer executes the jmp thereby leaving the RIM loader and beginning operations at the address specified by the jmp. In this way, the RIM loader simulates the computer read-in mode.

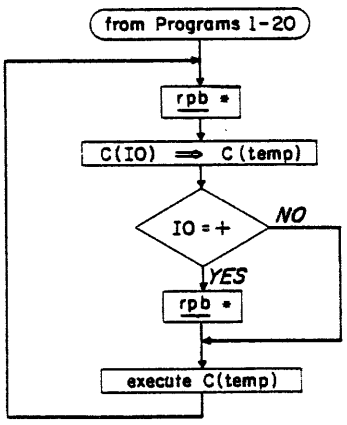


Figure 1 Instruction Test RIM Loader

3-3 PROGRAM 1

Program 1 clears memory locations 0000 through 7766 by means of the dzm instruction. Two of the principal reasons for clearing memory are: (1) If memory cannot be cleared, this may indicate extensive troubles within the computer, rather than merely one defective instruction; (2) the test for the jmp instruction (part of Program 2) follows Program 1, and clearing memory increases the probability that a jmp, executed to the wrong address, will result in an immediate computer halt.

Program 1 occupies locations 7766 through 7771, and uses location 7776 (location 7776 is also used by the RIM loader). Program 1 first clears location 0000, and then clears successive locations up to and including 7766. However, location 7766 is subsequently indexed so that its final contents equal 000001. This fact can be useful in the event that the computer should halt before reading in Program 2 of the Instruction Test. Should the computer halt with the contents of 7766 not equal to 000001, then memory was not properly cleared by Program 1. However, if the contents of 7766 do equal 000001 when the computer halts, then a computer malfunction probably prevented proper execution of the RIM loader. In the latter case, the remaining programs of the Instruction Test can be loaded by using the computer read-in mode.

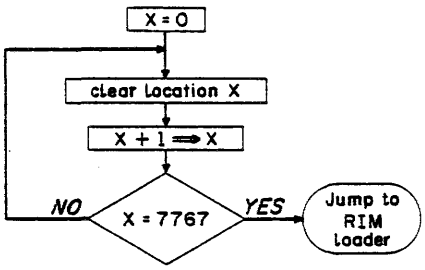


Figure 2 Instruction Test Program 1

3-4 PROGRAM 2

Program 2 test the hlt instruction, the jmp instruction, and all the szs instructions. It has three routines: the first is a single-instruction test for hlt; the second tests jmp and the szs instructions with the sense switches off; the third checks the szs instructions with all sense switches on. Operator intervention is required to start the second and third routines as well as to read in Program 3.

Program 2 is automatically read in and initiated at the completion of Program 1. The first instruction in Program 2 is a hlt at location 0001. This is the only time the hlt instruction is checked. After the computer executes this halt, the operator should ensure that all sense switches are turned off. At this time, the operator may also wish to check for the proper execution of Program 1. With all sense switches off, the computer is ready to execute the second routine of Program 2. This routine is initiated by pressing CONTINUE.

The routine begins by checking the jmp instruction. This instruction is tested by requiring successive jumps to locations 3000, 4777, 6000 and 0004. These particular four addresses were chosen so that every bit in the PC is both cleared to 0 from a 1, and set to 1 from a 0. Upon completing the jump sequence, the computer begins the sense switch check.

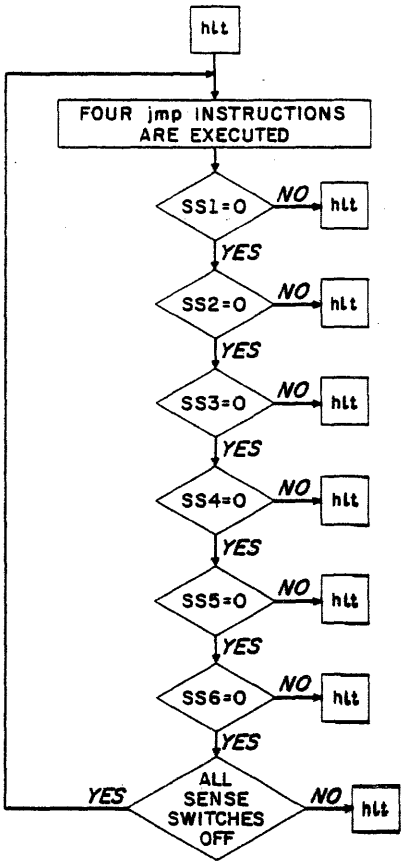
The sense switch check comprises a series of szs and hlt instructions which test each of the szs instructions beginning with szs 10 and running through szs 70. The routine checks both the normal and deferred condition of each sense switch when it is in the OFF position. The routine then loops back to the start of the jmp check. The second routine continues to iterate until an error halt is encountered or until the operator intervenes by stopping the computer.

If the computer halts with the contents of the MA between 0006 and 0030, then the hlt is a result of an szs error. The particular szs error which caused the halt can be obtained from TABLE 1-6 (Program 2 - Error Halts). A halt at any other location is probably a jmp error. After allowing the second routine to run, the operator stops the computer, turns on all sense switches and sets the ADDRESS switches to 0032. The operator then pushes the START to cause the computer to begin executing the third routine of Program 2.

The third routine checks the sense switches in the UP position. This check is identical to the sense switch test of the second routine, however, error halts are positioned in MA locations 0034-0056. See TABLE 1-6 (Program 2 - Error Halts) to define error. A halt in any other location is not a programmed halt. The third routine continues to iterate until the operator intervenes by stopping the computer.

Before read-in of Program 3, the desired settings for SS1 and SS2 must be selected. SS1 determines whether the program is executed upon completion of read-in (SS1 off) or whether the computer halts after read-in (SS1 on). Whenever execution of a program is completed, SS2 determines whether the next program is read in (SS2 off) or whether the same program iterates (SS2 on). The setting of SS1 has no effect on the iteration of a program.

After setting SS1 and SS2, the operator reads in Program 3. This may be accomplished either by pressing the READ IN switch or by setting the ADDRESS switches to 7772 and pushing down on START.



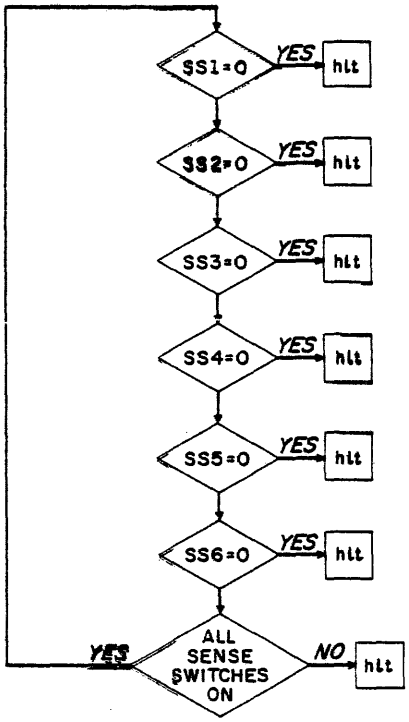


Figure 3 Instruction Test Program 2

3-5 PROGRAM 3

Program 3 tests two instructions from the operate group (cla, cma) and five instructions from the skip group (sma, spa, sza, 654000, 650000). The lat instruction is also partially tested. The instructions from the skip group are checked both with the defer bit equal to 0 and with the defer bit equal to 1. The instructions 654000 and 650000 are unconditional skips but when the defer bit is 0 (i.e., 644000 and 640000) the instructions are equivalent to the nop instruction.

The skip instructions are tested by asserting the condition for skip and placing a hlt after each skip instruction. The 644000 and 640000 instructions are checked to make sure they do not skip. Since, so far, only hlt, jmp and szs have been tested, some of the operate instructions are paired with the skip instructions for testing. Therefore, to determine which of two instructions caused the halt, the AC must be checked.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The program first tests the unconditional skips. A hlt is placed after each skip instruction. Then 644000 and 640000 are checked to see that they do not skip. Next, the AC is loaded with 1s using the TEST WORD switches and the lat instruction; the AC is then cleared (cla) and checked for zero (sza). If the computer halts here (MA equal to 0020), the AC must be checked to determine whether the cla instruction failed to clear it or whether the sza failed to skip properly.

The AC is again loaded with 1s using lat, complemented (cma), and checked for +0 (sza). If the computer halts here (MA equal to 0024), the AC must be checked to determine whether the halt was caused by a failure to skip or whether lat or cma failed to operate properly. The lat is used to ensure that all bits are complemented properly. (Clearing the AC, complementing twice, and checking for all zeros would not detect any bits which failed to complement; this procedure could not even indicate that the cma failed to operate.)

After being complemented, the AC equals +0. Next the spa and sma ' are checked by placing hlt after each. Then the AC is again complemented and sza ', spa ' and sma are checked by placing hlt after each.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

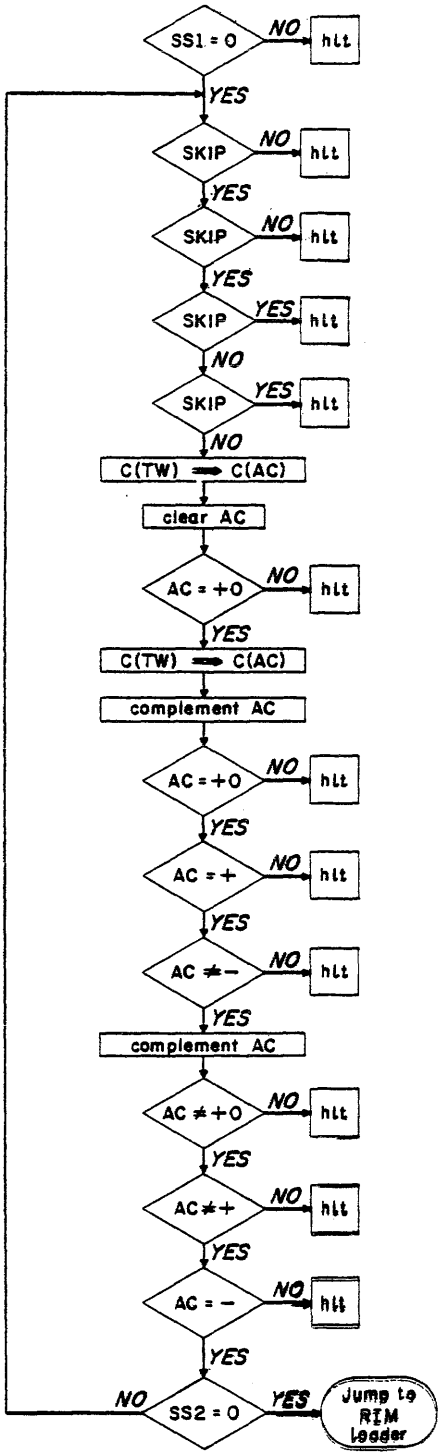


Figure 4 Instruction Test Program 3

3-6 PROGRAM 4

Program 4 tests three instructions (xor, sas, sad). The exclusive-OR is formed four times to include all possible combinations. The sas and sad instructions are checked to ensure that they skip when the condition is asserted and also to ensure that they don't skip if the skip condition is not asserted. The AC is used during the execution of both sas and sad. Therefore, a check is performed to see if the contents of the AC were properly replaced.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The xor test follows the sensing of switch 1. This part of the program occupies locations 0003 through 0022 and has four error halts. The exclusive-OR is formed as follows: AC all 1s with a core location containing all 1s; AC all 0s with a location containing all 0s; AC all 0s with all 1s; and the AC all 1s with all 0s. This series of operations forms all possible combinations of the exclusive-OR for each bit of the AC.

The AC is loaded with all 1s by being cleared and complemented. Next, the exclusive-OR is formed with a location (0054) containing all 1s. The AC is checked; if it is not +0, an error halt occurs. If no halt occurs, and exclusive-OR is formed with a location (0055) containing all 0s, and the AC is checked for +0. After this check, with the AC still +0, the exclusive-OR is again formed with the location containing all 1s. The AC contains all 1s, and is therefore complemented before checking for +0. When this check is completed, the AC is once more complemented (so that it contains all 1s) and the exclusive-OR is formed with the location containing all 0s. The AC is then complemented for the last time and checked for +0. At the end of this check, the program begins the sas and sad test.

The sas and sad test occupies locations 0023 through 0050; it contains eight error halts. Two test numbers (000000 and 777777) are used to check the compare instructions. The test has four parts: (1) The sas instruction is used to compare two equal numbers (000000). If the computer

skips, the contents of the AC are checked to ensure that they were properly replaced after the sas instruction. However, if the computer fails to skip, an error halt occurs. (2) The same test is then made for sad, comparing unequal numbers. (3) Next, sas is used to compare unequal numbers, and the AC is checked to ensure that its contents were properly replaced. If the computer skips, an error halt occurs. (4) Finally, the same test is made for sad comparing equal numbers (777777). This completes the test of the compare instructions, and switch 2 is sensed.

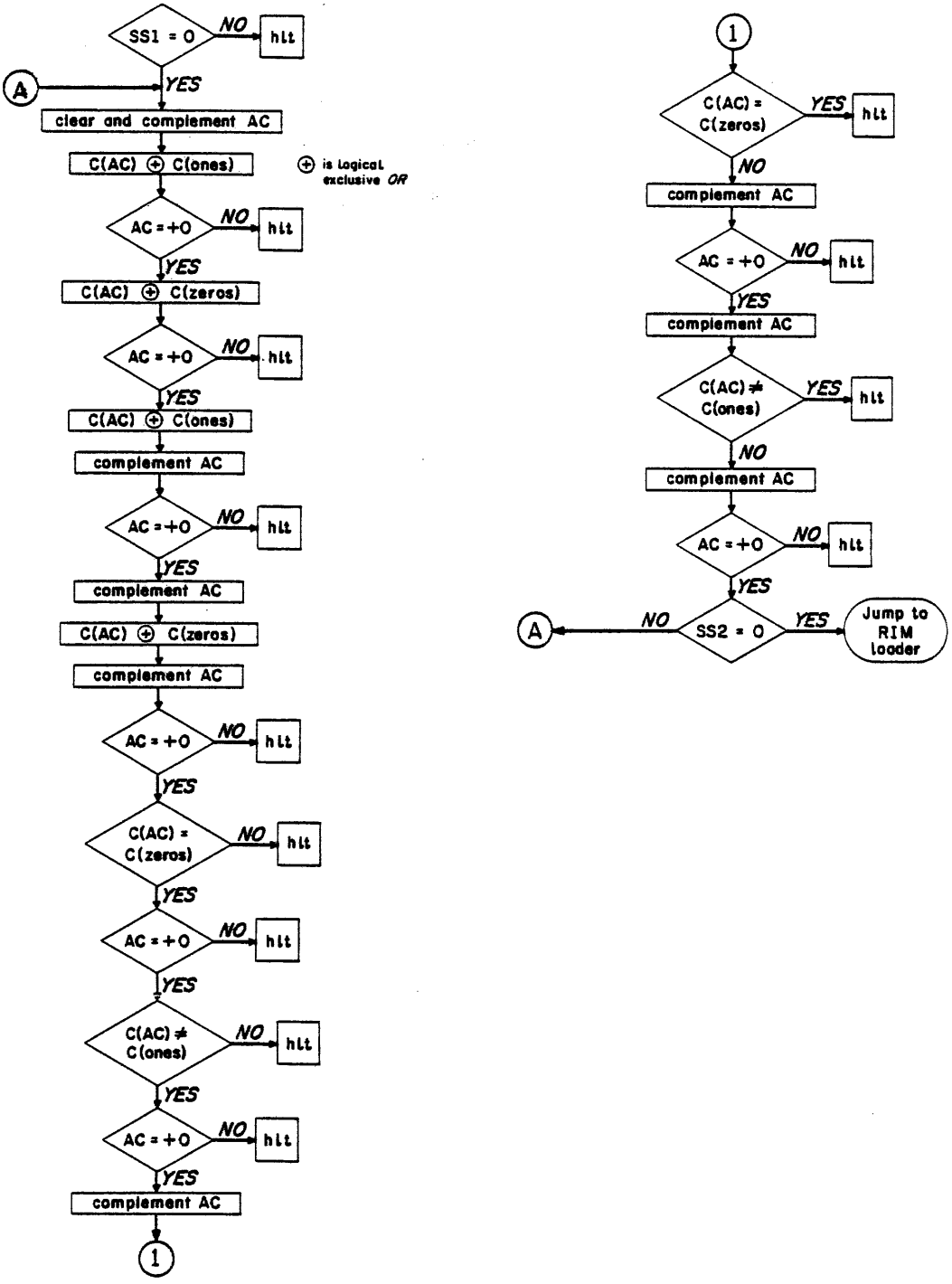


Figure 5 Instruction Test Program 4

3-7 PROGRAM 5

Program 5 tests four instructions (dip, dap, dac, lac). Some test numbers are transferred between two memory locations and the AC. To ensure that the transfers were properly executed, the AC and the memory locations are compared after each eighteen-bit transfer. The dip, dap and dac instructions are used to deposit 0s into memory locations containing 0s, to deposit 1s into locations containing 0s, 1s into locations containing 1s, and 0s into locations containing 1s. Similarly, the lac instruction is used to load the AC with 1s and 0s.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The dip, dap and dac test follows the sensing of switch 1. This part of the program occupies locations 0003 through 0041 and has eight error halts. Four halts are dip and dap errors; the remaining four are used in the dac check.

The contents of the AC are deposited into location 0063 using dip and dap. Then the AC is compared against 0063. If they differ, an error halt occurs. Next, the contents of the AC are deposited into location 0064 using dac. The AC is compared against 0064 and, if they differ, and error halt occurs. This procedure is repeated four times. The first and fourth times, the initial contents of the AC are all 0s; the second and third times, the initial contents are all 1s. This sequence effects a transfer into memory of all 0s into all 0s, all 1s into all 0s, all 1s into all 1s, and all 0s into all 1s. Then the program proceeds with the lac test.

The lac test occupies locations 0042 through 0055; it has four error halts. This test is quite similar to the test for dip, dap and dac. The AC is successively loaded with all 0s, all 1s, all 1s, and all 0s. After loading with 0s the AC is checked using the sza instruction; after loading with 1s the AC is compared against a core location (0062) which contains all 1s. This completes the lac test and switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

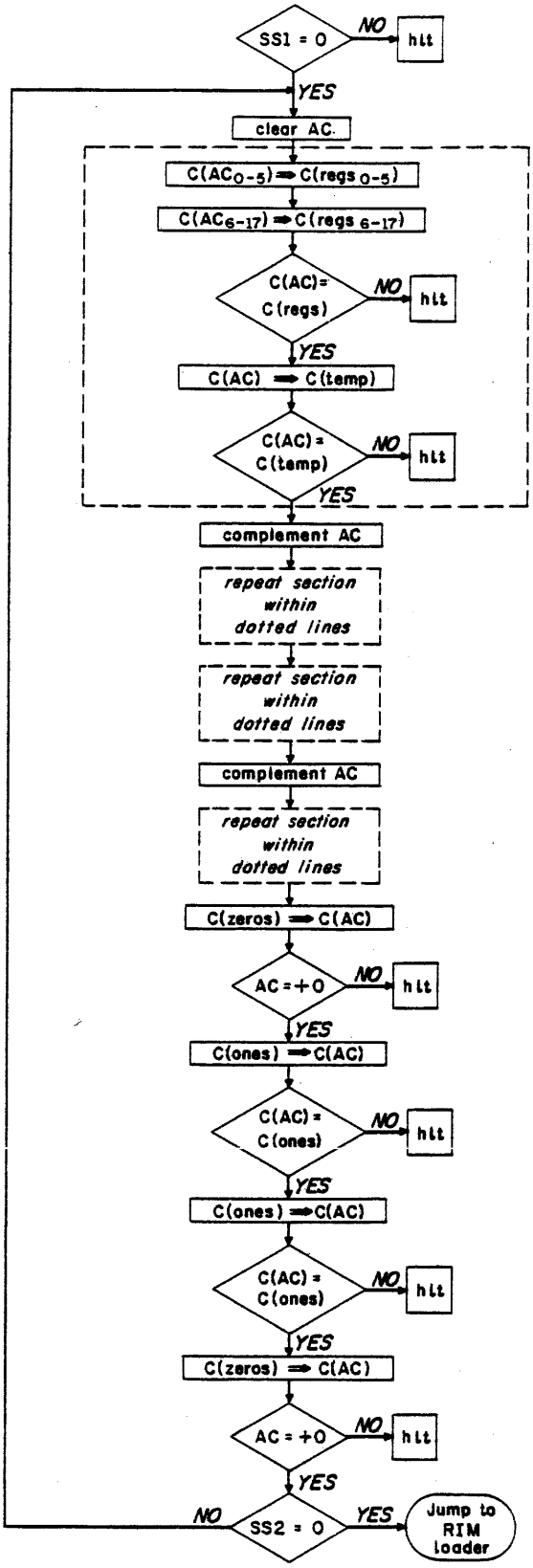


Figure 6 Instruction Test Program 5

3-8 PROGRAM 6

Program 6 tests two instructions -- dzm and idx. The program is designed to facilitate the diagnosis of troubles in the AC. For this reason, there are more error halts than are required to merely check the idx instruction.

The dzm instruction is tested to ensure that it clears a memory location containing all 1s as well as a location containing all 0s. The idx instruction is tested by checking for the proper execution of various AC operations. Four checks are performed: (1) Checks that the carry propagates the full length of the AC; (2) checks that the end-around carry functions properly; (3) checks that the contents of the AC are correct after an idx; (4) checks that the carry terminates properly in each bit of the AC.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The dzm test follows the sensing of switch 1. This test occupies locations 0003 through 0014. It has two error halts. The first results from failure to clear a memory location containing all 1s; the second results from failure to clear a memory location containing all 0s. The program deposits 1s into memory location 0166. This is accomplished by clearing and complementing the AC and depositing its contents into the memory location. Then location 0166 is cleared with a dzm and the results are checked by loading the contents into the AC and checking for +0. If the AC is +0, the first halt is skipped and the same location is once more cleared with a dzm. The same check is again performed, and if the second halt is skipped, the program proceeds to the idx test.

The idx test occupies locations 0015 through 0165 and has 21 error halts. The first three halts test the AC for the propagation of the carry chain, the end-around carry, and the contents after an idx. The remaining 18 halts check for termination of the carry chain in each of the AC bits beginning with the last, AC_{17}.

The idx test uses memory location 0166 as a temporary buffer. The program deposits 1s into the buffer by clearing and complementing the AC and transferring its contents. Then the buffer is

indexed once. Since the number indexed is all 1s, the carry clears the AC and the end-around carry makes AC_{17} equal 1. The AC is checked for a 0 in the sign bit to ensure that the carry propagated the length of the register. If the AC is positive (AC_s=1), the first halt is skipped. The AC is then checked for nonzero to test the end-around carry. If the AC is not zero, the second halt is skipped.

The final contents of the AC should be 000001. Note, however, that the first two halts would be skipped if the AC contained any positive number greater than zero. Therefore, the AC must be checked to ensure that its contents are 000001. This is accomplished by complementing the AC, depositing its contents into the buffer, indexing the buffer once, and checking the AC for +0. Complementing the AC produces the number 777776 which is then deposited in the buffer; attempting to index this number causes the computer to clear the AC. The AC is checked for +0. If this condition is asserted, the third halt is skipped and the test for termination of the carry chain commences.

To check the last fifteen bits of the accumulator (AC_{3-17}) for proper carry chain termination, a number is deposited into the buffer. This number consists of 1s in all bits except for a 0 in the sign bit and a 0 in the bit which is being checked. If the carry chain does not terminate properly, it continues the length of the AC and causes a sign change. Note that this test was preceded by the check which ensures that the carry can propagate the full length of the register. The AC is checked for sign; if positive, the error halt is skipped. The check for each bit has a separate error halt. The check for the first three bits (AC_{0-2}) is similar, except that the test number also has the sign bit equal to 1 and the AC is checked for a minus sign. This completes the idx test and the program senses switch 2.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

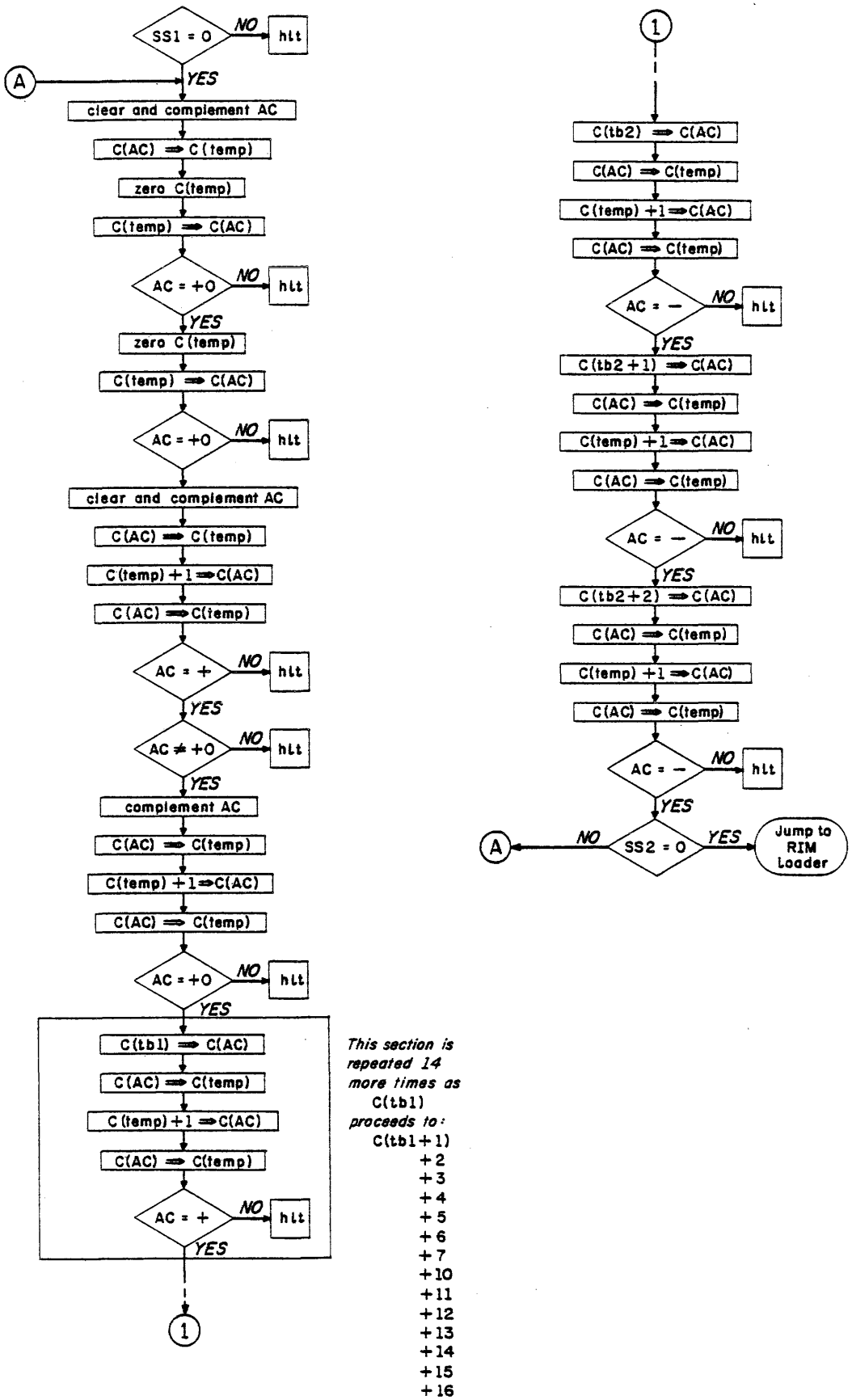


Figure 7 Instruction Test Program 6

3-9 PROGRAM 7

Program 7 tests the isp instruction. One location is indexed (isp) through all the positive numbers. A check is made after each indexing to ensure that the skip occurred and that the location was incremented by exactly 1. Another location is indexed (isp) through all the negative numbers. A similar check is made after each indexing to ensure that no skip occurred and that the location was correctly incremented.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The isp test commences after the sensing of switch 1. This part of the program occupies locations 0003 through 0040. Locations 0045 through 0050 are used for indexing. The program has eight error halts. First, the two locations used for indexing are initialized. Next, the program loops as the locations are indexed until their contents equal 777776 and 377777. Finally, after leaving the loop, the locations are indexed until a sign change occurs. The idx instruction is used to check the results after each isp. In the first part of the test, locations 0045 and 0047 are each initialized with 377777; locations 0046 and 0050 are each initialized with 777776.

The second part of the test, the loop, contains four indexing instructions. It begins by indexing (isp) location 0050 which contains 777776. Indexing this number once produces 000000. Therefore, an error halt occurs if the computer does not skip. Then, location 0046, which also conta' 777776, is indexed (idx) and the contents of locations 0050 and 0046 are compared. If their contents differ, an error halt results.

Similarly, location 0047, which contains 377777, is indexed (isp) to 400000; then it is checked to make sure it produces no skip. Location 0045 is indexed (idx) and compared against 0047. The program iterates this sequence and does not leave the loop until locations 0045 and 0047 contain 777776, and locations 0046 and 0050 contain 377777. Notice that subsequent to the first indexing, no sign change has occurred. After leaving the loop, the program enters part three of the test.

The last part of the test checks that, on indexing (isp) 777776 once, the resulting number is 000000; it also checks that a skip takes place. Similarly, it checks that, on indexing (isp)

377777 once, the resulting number is 400000 and also that no skip results. This ends the test and switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

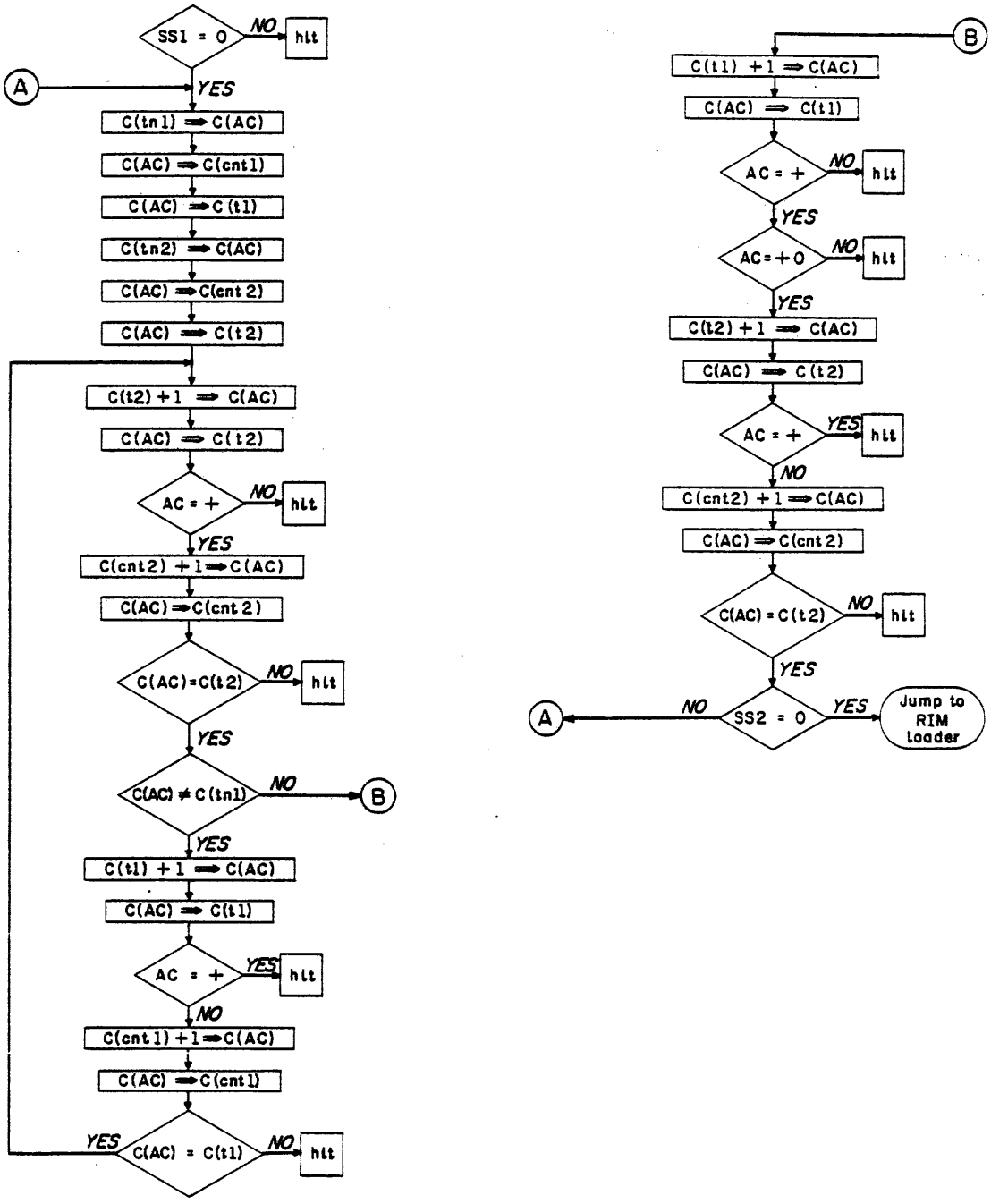


Figure 8 Instruction Test Program 7

3-10 PROGRAM 10

Program 10 tests two instructions (and, ior). For each bit, all possible combinations of the AND are formed. Similarly, all possible combinations of the inclusive-OR are formed. A check is made to ensure that each operation was correctly executed. After any error halt the program may be restarted simply by pushing CONTINUE.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The and test follows the sensing of switch 1. This part of the program occupies locations 0003 through 0022 and has four error halts. The two test numbers used (000000 and 777777) are in locations 0046 and 0047, respectively.

The AC is loaded with all 1s, ANDed (and) with all 1s; then the contents of the AC are checked. The procedure is repeated, ANDing 1s to 0s, 0s to 0s, 0s to 1s; each and being followed by a check of the AC. This sequence forms all possible combinations for each bit of the and instruction. The program then executes the ior test.

The ior test occupies locations 0023 through 0042 and has four error halts. This test is identical to the and test except that instead of ANDing two numbers, the inclusive-OR is formed. After the ior test, switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

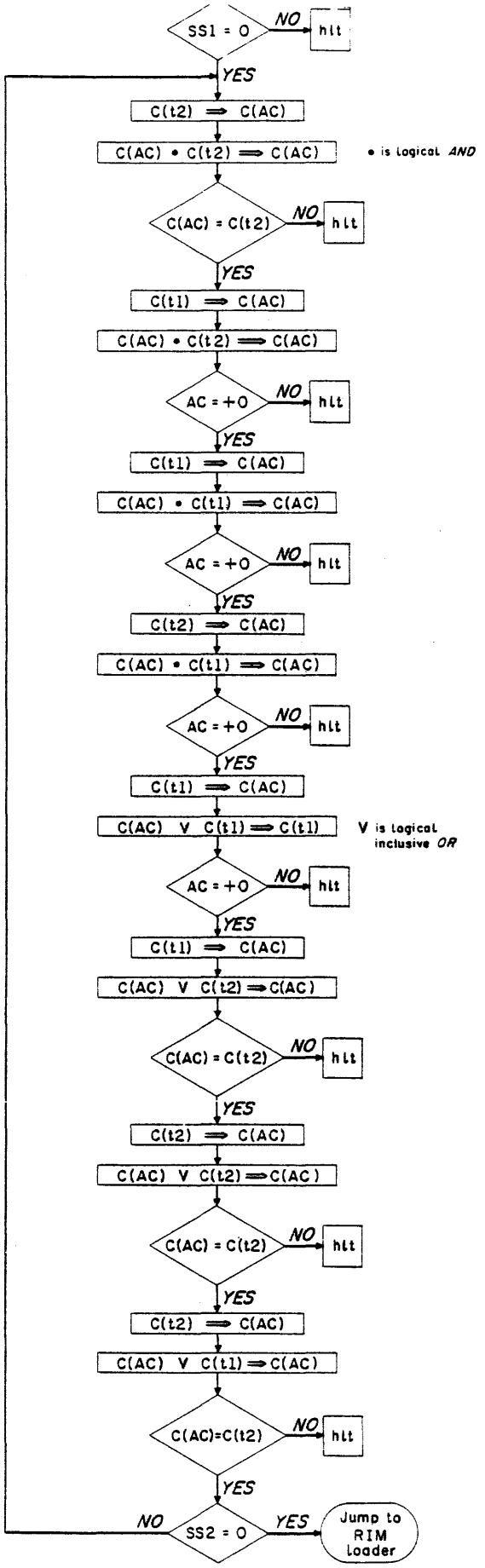


Figure 9 Instruction Test Program 10

3-11 PROGRAM 11

Program 11 tests three instructions (lio, dio, spi). A test number is loaded into the IO and deposited from the IO into a temporary buffer. The AC is used to check that the number was correctly transferred. The spi instruction is then checked. The test number takes on all possible values.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The lio, dio and spi test follows the sensing of switch 1. The test has two parts. In the first part, the test number ranges over all positive values; in the second, the test number ranges over all negative values. Location 0034 is used as a temporary buffer and location 0033 contains the test number. The first portion of the test occupies locations 0003 through 0016 and has two error halts.

At the start of the test, the location containing the test number and the AC are both cleared. Then a loop, which comprises the test for all positive numbers, is entered. The IO is loaded with the test number, and its contents are deposited in the temporary buffer. The contents of the buffer are compared against the AC; if they differ, an error halt occurs. On a halt, the contents of the IO must be checked to determine whether the lio or dio caused the error. Next, the spi is checked to ensure that it skips on a positive IO. Then the program indexes the test number and jumps to the beginning of the loop. (Note that execution of idx leaves the test number in the AC, in preparation for the comparison after the IO transfer.) When the test number becomes negative, the program leaves this loop and enters the second part of the test.

The second part occupies locations 0017 through 0027 and has two error halts. It consists of a loop which is similar to the first loop except that spi ' is checked to ensure that it skips on negative IO. When the test number is indexed to +0, the program leaves this loop and senses switch 2.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

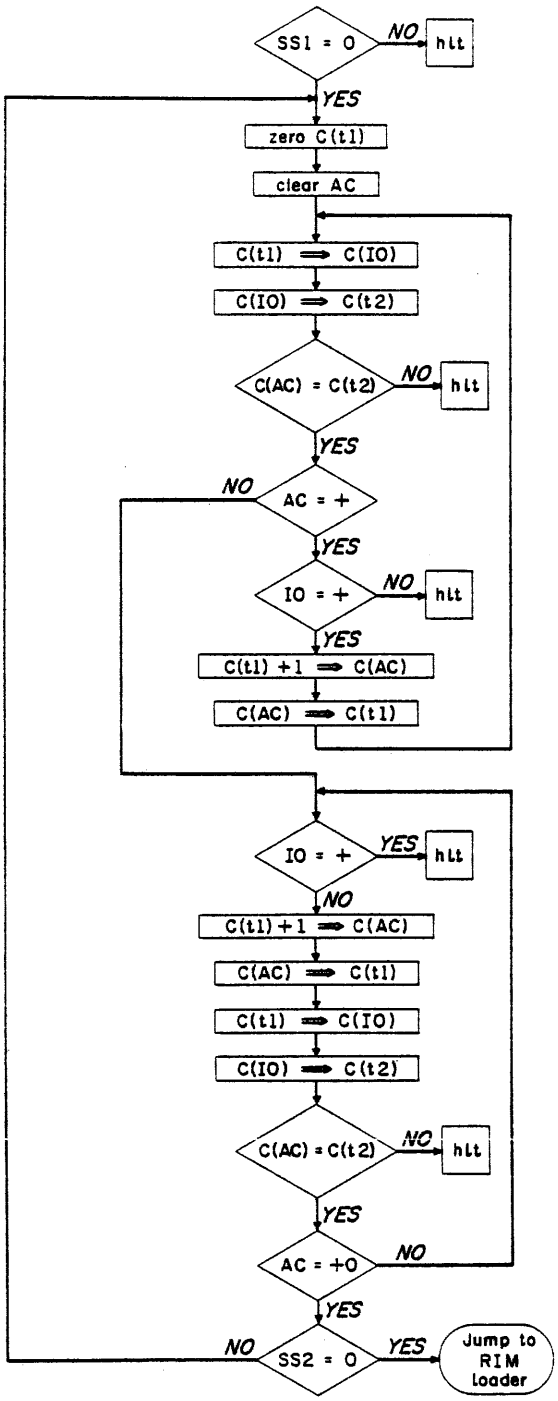


Figure 10 Instruction Test Program 11

3-12 PROGRAM 12

Program 12 tests three instructions (add, szo, sub). To facilitate trouble diagnosis, each of the operations which make up the add instruction are checked separately. Because of this feature, however, there are more error halts in the program than would be required merely to test this instruction. The capability of testing the component operations of add should prove equally useful for troubleshooting the sub instruction, since the two instructions differ only in that sub includes a complement operation which is not used in add. The szo instruction is checked to ensure that it skips on overflow, that it does not skip on no overflow, and that it correctly sets and clears the overflow flip-flop.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The add and szo test follows the sensing of switch 1. This test occupies locations 0003 through 0077 and contains 19 error halts. Seven halts are used for the szo, and the remaining twelve halts are used for add.

The add test checks for correct execution of the partial add, the end-around carry, the full-register carry, the clear-on-minus-zero, and the ripple carry. The ripple carry is checked to ensure that it propagates through the entire register and also to ensure that it is initiated properly at each bit of the AC. Interspersed within the add test are checks on the szo instruction.

The test checks that szo skips on zero overflow. This is done by executing szo, szo ' and szo followed by a halt, and szo ', and another halt. If the overflow flip-flop is set, the first szo clears it. Then the szo ' is executed, followed by another szo which skips the first halt. However, if the flip-flop was zero, the first szo skips the szo ' and the next szo skips the first halt. In either case, the overflow flip-flop is cleared and the first halt is skipped. Note that the only other way the first halt can be skipped is if the szo fails to clear the flip-flop and if, furthermore, the szo ' fails to skip with the flip-flop set. In this case, the szo ' which follows the first halt detects the error. After completing this check, the program begins the test for the partial add.

The partial add is tested by performing a sequence of additions which form every combination of the partial add for each bit in the AC. First the AC is cleared and the number 000000 is ad to it. Then 377777 is added to 000000, 000000 to 377777, 400000 to 000000, and 000000 to 400000. After each addition, the contents of the AC are checked. Then a check is made to see that the overflow flip-flop was not set during this sequence.

The test for the full-register carry follows the overflow check. The AC is loaded with the number 252525 (alternate 0s and 1s) and the same number is added to it, producing the number 525252 and AC overflow. The contents of the AC are checked. This tests the full-register carry for the odd bits in the AC. Then the overflow is tested and cleared. Next, the even bits are tested for the full-register carry by adding 125252 to itself. Again the AC is checked, followed by a short test for clear-on-AC-minus-zero.

This test is performed by loading the AC with -0 and adding +0 to it. The AC is checked for +0, and the overflow flip-flop is sensed to ensure that it was not set. The ripple-carry test follows this overflow check.

The ripple carry is tested to ensure that it travels the entire length of the AC; it is also tested to ensure that it is correctly initiated in every bit. The AC is loaded with all 1s, and +1 is added to it. The carry ripples the entire length of the register and around the end, leaving the final contents of the AC equal to +1. As before, the contents of AC and of the overflow flip-flop are checked.

To test that the ripple carry initiates properly at each bit, the AC is loaded with the number 252525, to which the number 777777 is then added. Unless the carry initiates correctly in al. even bits of the AC, the contents of AC are incorrect and, upon checking, cause a halt. Similarly, the odd bits are tested using the number 525252 and adding 777777. Overflow is checked for the last time, concluding the test for add and szo. The program then proceeds to the test for sub.

The sub test occupies locations 0100 through 0111; it has only one error halt. If the add instruction is working correctly, it is necessary to check only the execution of sub once to ensure that it, too, is operating correctly. However, Program 12 tests the sub instruction 262,144 (2^{18}) times. This large number of tests is desirable because sometimes an instruction operates incorrectly only when executed repeatedly. As previously mentioned, the add and sub instructions

are almost identical in operation; consequently, it suffices to test one of the two for repeated execution.

The repetitive sub test is effected by subtracting -1 (equivalent to adding +1) and checking the results against a location which is being indexed. This procedure is followed, beginning with +0 and continuing through the positive numbers, the negative numbers, and back to zero. No check is made for overflow (which should occur). At this point the test is complete and switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

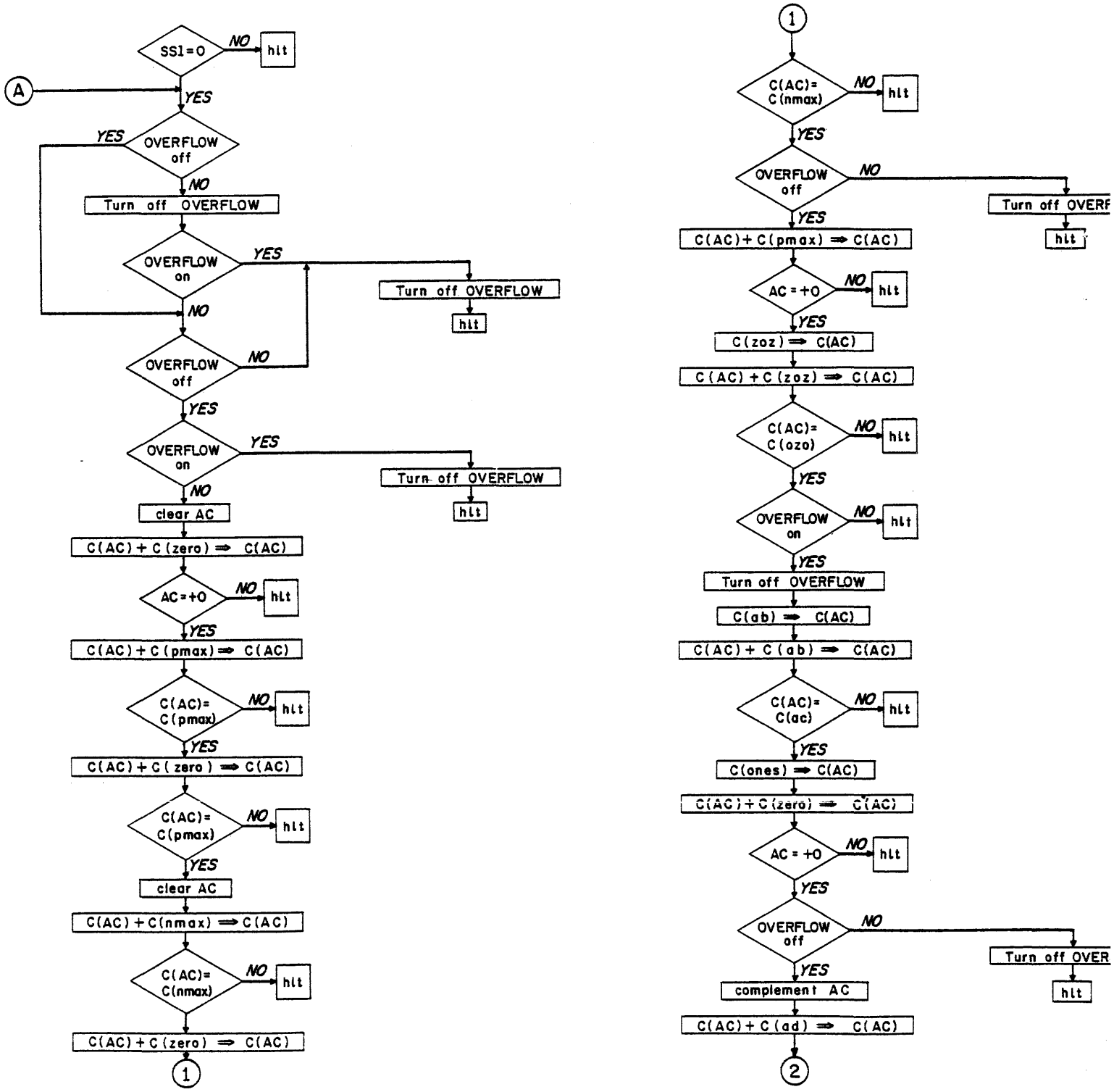


Figure 11 Instruction Test Program 12

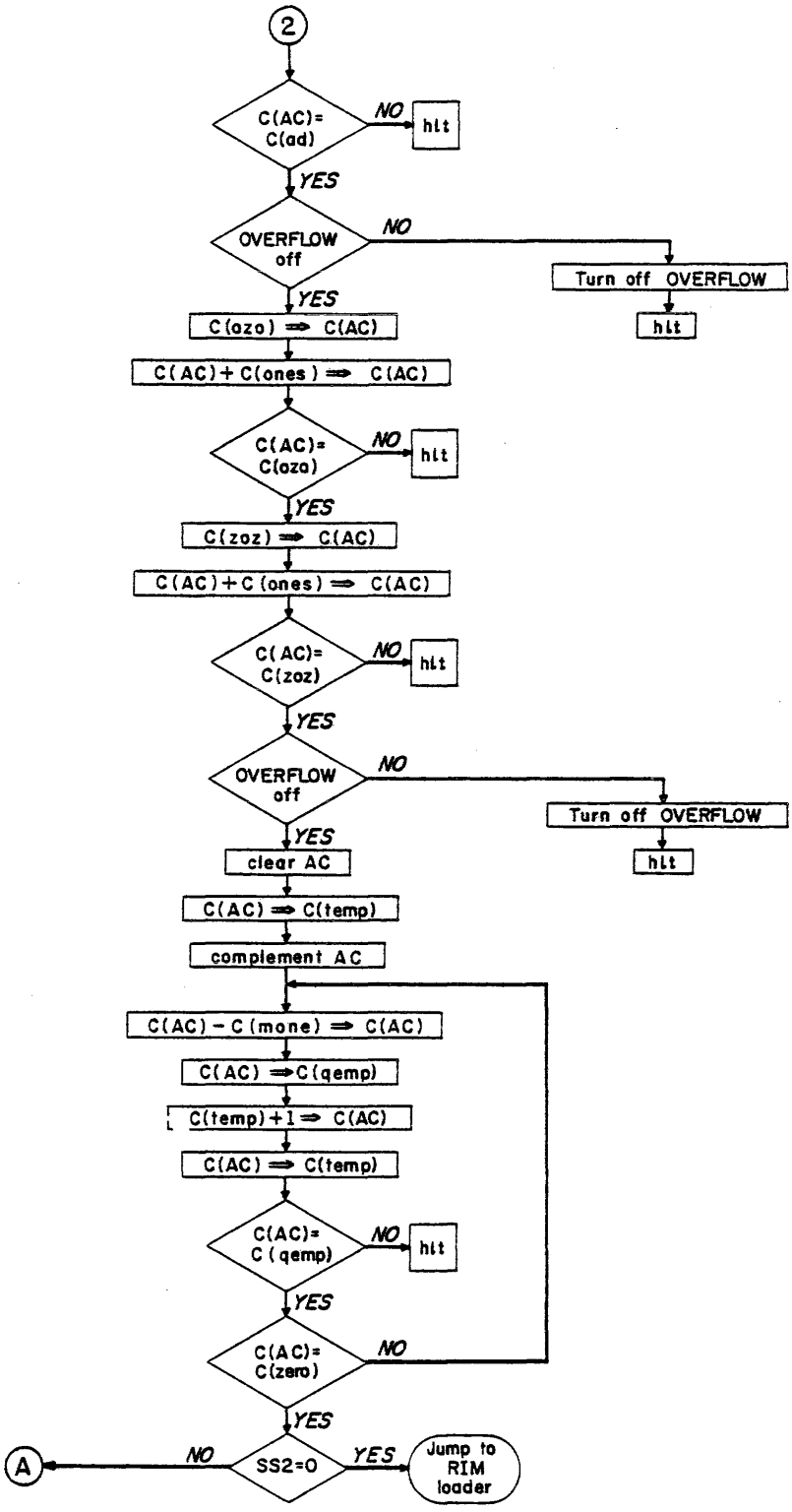


Figure 11 Instruction Test Program 12 (continued)

3-13 PROGRAM 13

Program 13 tests two instructions (law, cli). The AC is loaded (law) so that, for each bit, all combinations occur. The law instruction is also checked with the defer bit equal to 1 (Load Accumulator With -N). To check cli, the IO is cleared twice; the first time the IO contains all 1s, the second time all 0s.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The law test follows the sensing of switch 1. This part of the program occupies locations 0003 through 0020 and has four error halts. The law instruction clears the AC prior to loading it with a number. To check the correct execution of the AC clear, the accumulator is loaded with all 1s, and then loaded (law) with 0s and checked for +0. Then the MB $\xrightarrow[6-17]{1}$ AC is checked by successively loading 7777, -0, -7777. The contents of the AC are checked against a location in core after each load instruction. The program then enters the cli test.

The cli test occupies locations 0021 through 0035 and has two error halts. To check the cli instruction the IO is loaded with 1s, cleared, checked for all 0s, cleared again, and checked for 0s again. The IO is checked for 0s by transferring its contents to the AC and checking the accumulator for +0. The transfer from IO to AC, as well as the loading of the IO, is effected using a memory location as a temporary buffer. After the cli test is executed, switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

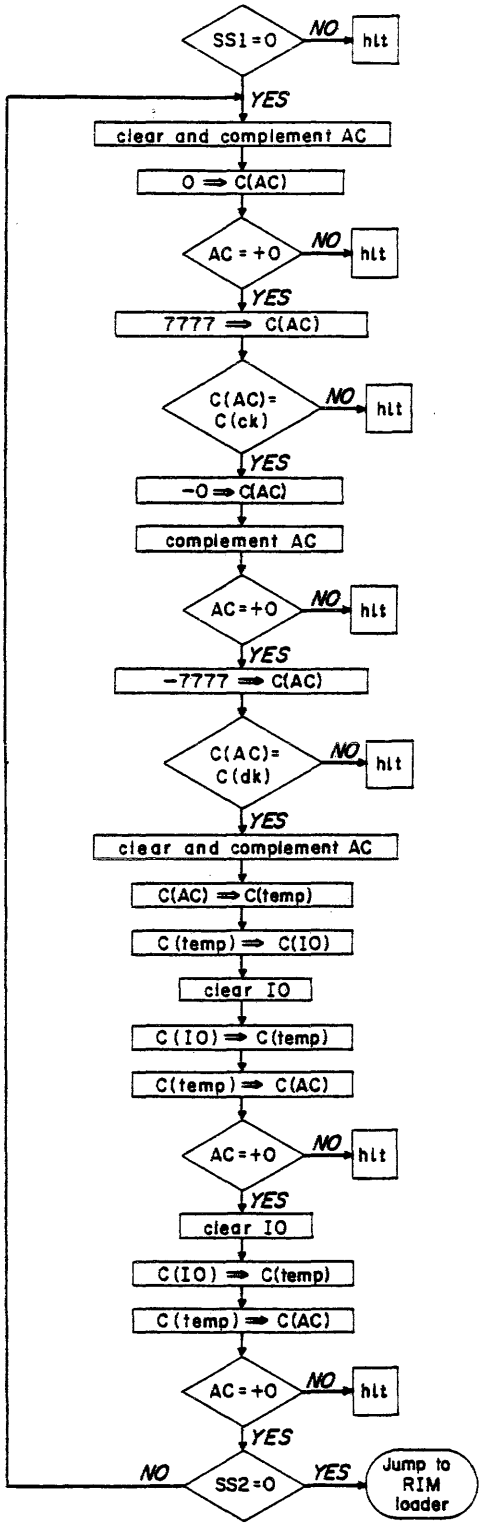


Figure 12 Instruction Test Program 13

3-14 PROGRAM 14

Program 14 tests all instructions pertaining to the program flags (clf, stf, szf), then clears mem before reading in the next program. (The reason for clearing memory is explained in the Program 15 description.) Program 14 clears any flags that may be on, sets each flag in sequence (starting with flag 1 and proceeding through flag 6), clears all flags simultaneously, sets all flags simultaneously, and then clears each flag in sequence (again proceeding 1 - 6). The program transfers to a count loop before each clf or stf instruction. The count loop introduces a time delay so that the program flag settings may be observed at the console. An szf instruction follows each clf or stf to check for the correct execution of the instructions.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer h with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The program has five parts. First, a short sequence initializes the loops which comprise parts two and four, and clears all the flags. Since the second part sets the flags, it is necessary to have the flags cleared in order to test the stf instruction.

Second, a loop sets all flags, one at a time, and checks to ensure that they were set. Third, a sequence clears all flags simultaneously, checks that they were all cleared sets all flags simultaneously, and checks that they were all set. Fourth, a loop clears all flags, one at a time, and checks that each was cleared. Fifth, a count loop is used as a time delay by parts two, three and four. The delay is generated by counting to 2^{15}.

The count loop, part five, occupies locations 0057 through 0065. It is used as a time delay by parts two, three and four. When the program enters the count loop, the AC contains the address for the return. Therefore, this address is deposited in the address portion of a jump instruction. Then the number 700000 is deposited into location 0057. That location is incremented until it becomes positive, and the program executes the jump which was set up for the return.

The first part of the test occupies locations 0003 through 0015 and has one error halt. In preparation for part two, all the program flags are cleared and checked to ensure that they were clea Then the address part of szf, stf and clf instructions in parts two and four is set to operate on

flag 1. After setting the instruction addresses, the program proceeds to part two.

Part two occupies locations 0016 through 0026 and has one halt. It uses the count loop to introduce a time delay, sets flag 1, checks that flag 1 was set, indexes the stf and szf ' so that the next flag will be set and checked, and then jumps back to the start of part two. Part two is executed six times (once for each flag 1 - 6); the program then proceeds to part three.

Part three occupies locations 0027 through 0042 and has one halt. This part introduces a time delay, clears all flags simultaneously, checks that all flags were cleared, introduces another time delay, sets all flags simultaneously, and then sets up part four to execute six times.

Part four occupies locations 0043 through 0053 and has one halt. It is identical to part two, except that the flags are cleared and checked for clear, instead of being set and checked for set. After the program completes part four, switch 2 is sensed.

With SS2 off, the program jumps to location 7766, and clears memory locations 0000 through 7765, then transfers to the RIM loader which reads in Program 15. However, if SS2 is on, the computer loops back to location 0003 and iterates Program 14 until the sense switch is turned off. Note that switch 1 is sensed only after read-in. Thus the program can iterate, regardless of the setting of switch 1.

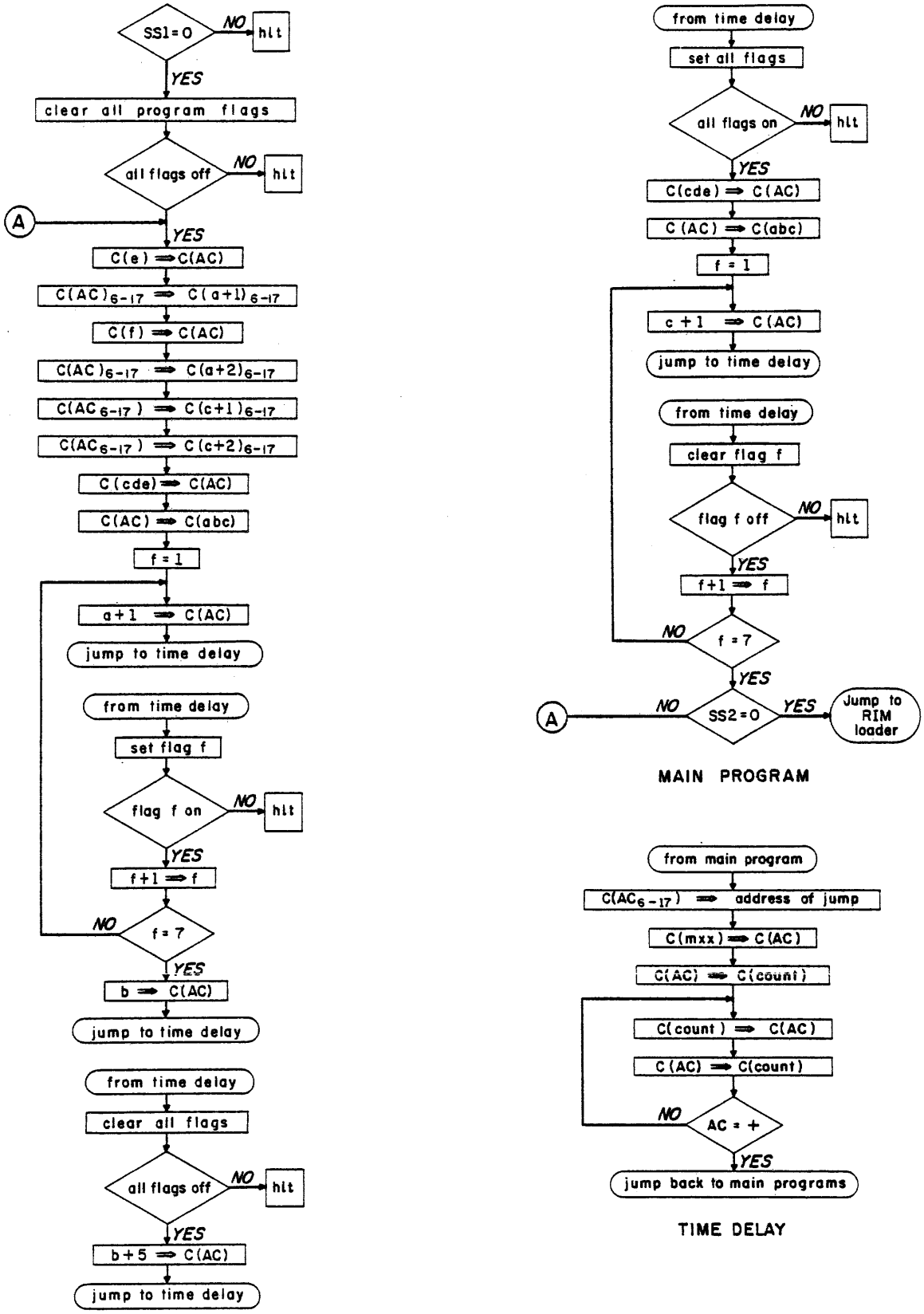


Figure 13 Instruction Test Program 14

3-15 PROGRAM 15

Program 15 tests three instructions (jsp, jda, cal). Memory locations 0000 through 7765 are cleared prior to reading this program into core. Clearing memory increases the probability that a jump to an incorrect address will cause a halt. Consequently, the memory-clear facilities error diagnosis, should such a jump occur.

The program checks all the transfers which make up the jsp, jda and cal instructions except for the transfer of the EXD flip-flop into AC_1 (which is executed concurrently with the PC $\xrightarrow{1}$ AC). The program does not test for the EXD transfer because the extend-mode instructions are not checked within the Instruction Test. The transfers are checked for all possible combinations for each bit (refer to Tables 1-16B, 1-16C, and 1-16D under Program 15 Error Halts).

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The jsp test occupies 0016-0017, 0757-0764, 1100-1105, 1755-1756, 6020-6025 and 6076-6077. This part of the program has three error halts. The jsp instruction comprises four transfers:

$\xrightarrow{0}$ AC, PC $\xrightarrow{1}$ AC, $\xrightarrow{0}$ PC and MB $\xrightarrow{1}$ PC. By locating the test in various parts of memory, all possible combinations of each transfer are checked for each bit.

Three jsp instructions are executed within the jsp test. After each jump, bit 1 of the AC is cleared to mask out the transfer of EXD into AC_1. Then the contents of AC are checked to ensure that the PC $\xrightarrow{1}$ AC was correctly performed.

The jda test occupies locations 32-37, 737-750, 1000-1011, 1731-1736, 5772-5777 and 6040-6052. This part of the program has six error halts. The jda instruction comprises six transfers:

AC $\dashrightarrow$ MB, $\xrightarrow{0}$ AC, PC $\xrightarrow{1}$ AC, $\xrightarrow{0}$ PC, MA $\xrightarrow{1}$ PC and $\xrightarrow{+1}$ PC.

By locating the test in various parts of memory, all possible combinations of these six transfers are checked for each bit.

As in the jsp test, three instructions are executed within the jda test and AC_1 is cleared before the contents of the AC are checked. Moreover, a check is made to ensure that the contents of the AC were saved in the location specified by the address of the jda.

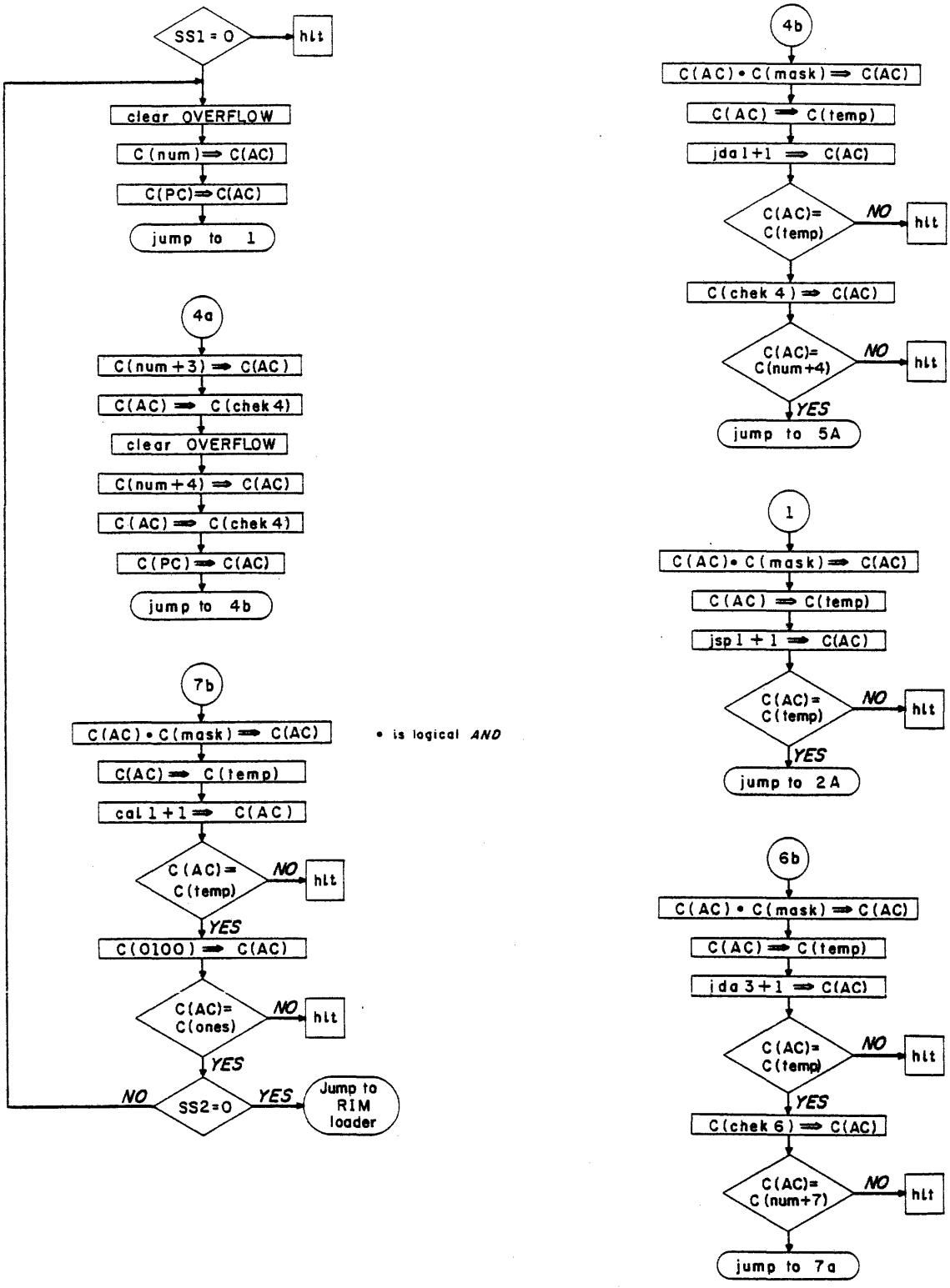


Figure 14 Instruction Test Program 15

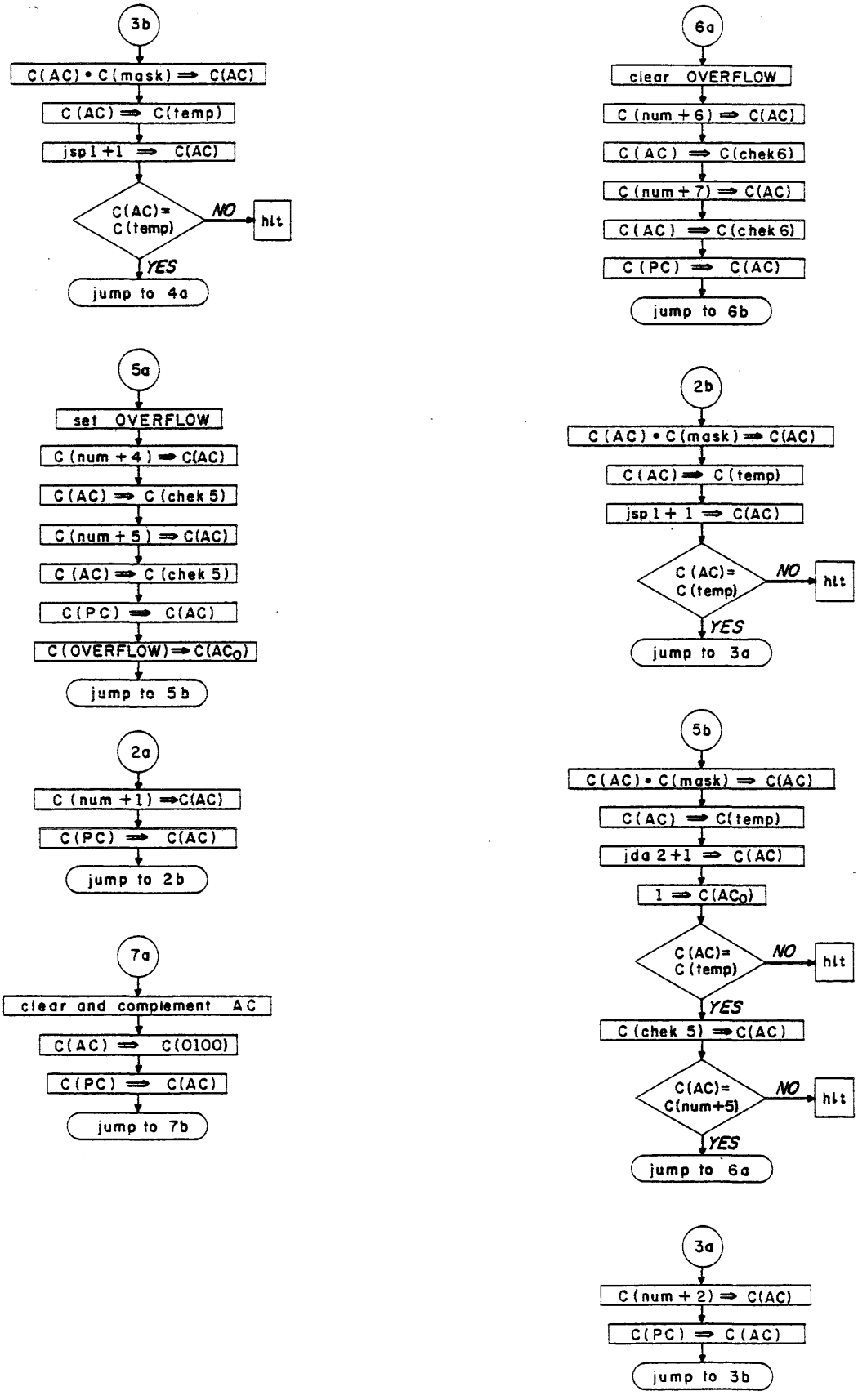


Figure 14 Instruction Test Program 15 (continued)

The transfer of the overflow flip-flop into AC_0 is checked during the second jda. Overflow is transferred into AC_0 during any PC $\xrightarrow{1}$ AC. Therefore, following this second jda, AC_0 is loaded with a 1 after masking out AC_1 but before checking the contents of the AC. Overflow is cleared before executing the third jda.

The cal test occupies locations 0100-0110 and 3000-3001. This part of the program has two error halts. The jda and cal instructions are identical except that jda inhibits the 100 $\xrightarrow{1}$ MA. Therefore, it suffices to check that the cal signal is properly decoded and that the jump is correctly executed to location 0101. After the cal test, switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even SS1 is on.

3-16 PROGRAM 16

Program 16 tests two instructions (nop, xct) and also tests that indirect addressing is properly executed. The indirect address is deferred five times. Note that if the computer is operating in the extend mode, indirect addressing is limited to one level; this causes an error halt.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The first instruction, after the program senses switch 1, is nop. This constitutes the entire test for the nop instruction. If the nop is executed, the program enters the test for xct.

The program tests the execute instruction by using it to jump to the beginning of the indirect-addressing test. If xct fails to execute the jump, the program halts.

The test for indirect addressing occupies locations 0015 through 0020, and has one halt. The AC is cleared, and then loaded with all 1s, using the lac instruction and using five levels of indirect addressing. Then the contents of the AC are checked to ensure that indirect addressing was performed six times. This completes the test for indirect addressing. Switch 2 is then sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

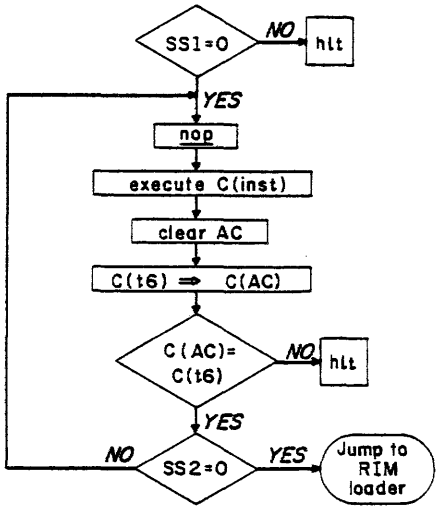


Figure 15 Instruction Test Program 16

3-17 PROGRAM 17

Program 17 tests the rotate instructions (ral, ril, rar, rir, rcl, rcr). All instructions are check for nine-bit rotation and for no rotation. Successively using each of the nine available bits (bits 17-9 of the instruction), ral, ril, rar and rir are checked to ensure that they rotate the registers, one bit at a time. The combined-register rotate commands are checked for stop and go, and also for rapid execution. The rapid execution tests both left and right rotation as well as alternation between left and right.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The rotate test follows the sensing of switch 1. This part of the program occupies locations 0003 through 0255, and has 26 error halts. The test has three parts. It first checks ral, ril, rar and rir, for one-bit rotation, using each of the last nine bits in the instruction. Second, the test checks that all instructions execute nine-bit rotations properly, and checks that no rotation takes plac bits 9-17 of the instruction are all zero. Finally, the test checks rcl and rcr for repeated execution as well as for alternate (right-left) execution.

The first part of the rotate test checks ral, ril, rar and rir to ensure that a one-bit rotate is executec properly. Each instruction is executed nine times, each time using a different bit to specify the rotate; e.g., for ral: 661001, 661002, 661004, 661010, 661020, 661040, 661100, 661200, 66[illegible]0 After every pair of commands (ral-ril; rar-rir), the contents of AC and IO are checked to ensure that the instruction was properly executed.

The second part of the rotate test executes a nine-bit rotate, using each of the six instructions once. Then all six instructions are executed, specifying no rotation (bits 9-17 all zero). The registers are checked after each pair of instructions (ral-ril; rar-rir) and after each of the rcl, rcr instructions.

The third and final part of the rotate test is a three-loop sequence which is iterated, using different test numbers. The purpose of this iteration is to subject the computer to the most adverse condi[illegible]ns of rotation. The first loop executes four rcl (9 bits) instructions in a row, checks the result, and

repeats this sequence 256 (2^8) times. The second loop is like the first, except that the rcr command is used instead of the rcl. The third loop alternately executes rcl and rcr for a total of eight rotate commands, checks the result, and repeats the sequence 256 times. Using different test numbers, the three loops are executed six times. This concludes the test for rotate, and switch 2 is sensed.

With SS2 off, the program transfers to the RIM loader and the next program is read in. However, if SS2 is on, the computer loops back to location 0003 and iterates the same program until SS2 is turned off. Note that switch 1 is sensed only after read-in. Thus, the program iterates even if SS1 is on.

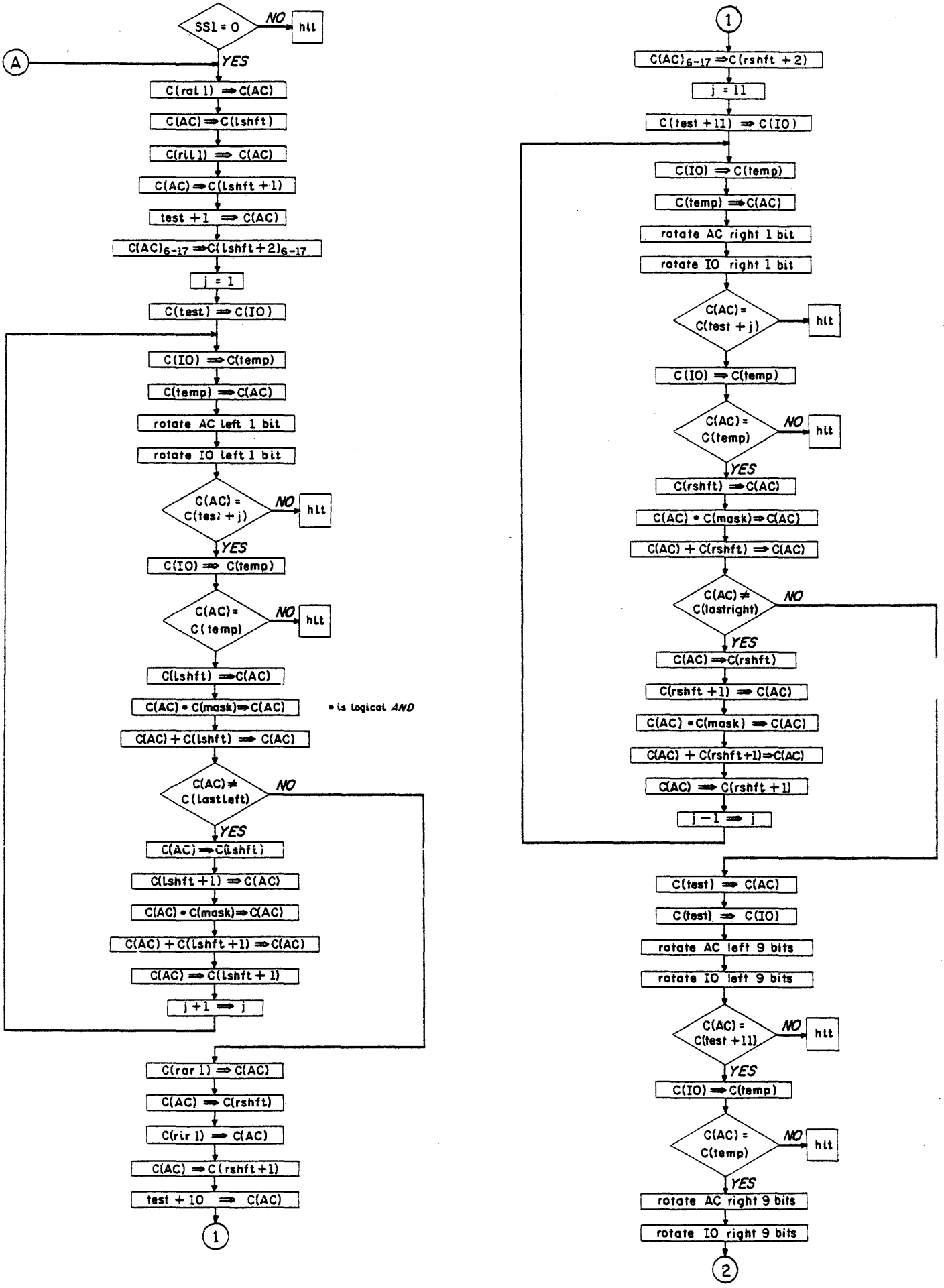


Figure 16 Instruction Test Program 17

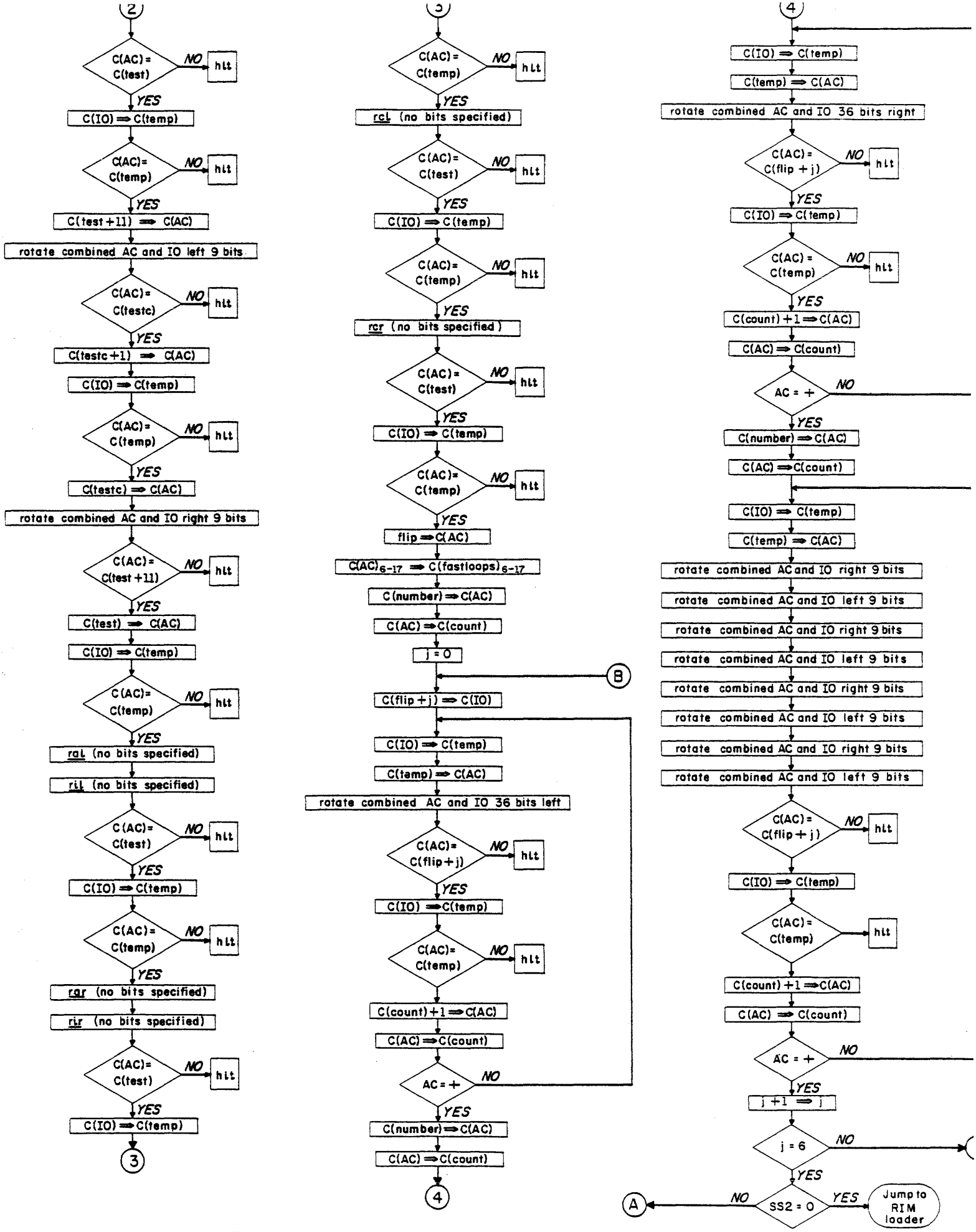


Figure 16 Instruction Test Program 17 (continued)

3-18 PROGRAM 20

Program 20 tests all the shift instructions (sal, sil, sar, sir, scl, scr). Some test numbers are shif to check the following operations: That bit 0 is shifted into bit 1 on the right shifts; that 0s are shifted into bit 17 on left shifts; and that the AC and IO are treated as one register on combined shifts.

The program begins by sensing SS1. With SS1 off, the program is automatically executed after being read into core memory. However, if SS1 is on, upon completing read-in the computer halts with the MA equal to 0002. Program execution is resumed by pushing CONTINUE. The above conditions hold both when the program is read in using the RIM loader, and when the READ IN switch is used. (Location 0000 contains the program number.)

The shift test commences after the sensing of switch 1. The test occupies locations 0003 through 0070, and has ten halts. The test has five parts. The first part checks the left shifts on both AC and IO to ensure that 0s are introduced into bit 17. The second checks the right shifts on both AC and IO to ensure that the sign bit (equal to 0) is shifted into bit 1. The third checks the right shifts of both AC and IO to ensure that the sign bit (equal to 1) is shifted into bit 1. The fourth checks the combined left shift to ensure that bit 0 of the IO is shifted into bit 17 of the AC The fifth and last part of the test checks the combined right shift to ensure that AC bit 17 is shifted into IO_0.

To check the left shift (part one of the test), the AC and IO are both loaded with the number 377777, and each of these two registers is shifted left seventeen times. Then the AC and IO are checked for zero; the AC by means of an sza instruction, the IO by depositing its contents into a memory location and comparing the location with AC.

Part two of the test, which comprises half of the right-shift check, is identical to part one except that the shift is right. The other half of the right-shift check (part three of the test) is identical to part two except that the number 400000 is used instead of 377777.

For the combined shift check (parts four and five of the test), the AC and IO are each loaded with the number 377777; the combination is shifted left once, and the results are checked. The test number is reloaded into both registers; the combination is shifted right one bit; and the results are again checked. This completes the shift test. Switch 2 is then sensed.

If switch 2 is on, the program iterates the shift test until the switch is turned off. If switch 2 is

off, the program executes a short terminal routine, and halts. The terminal routine sets the principal computer registers to the configuration shown in Table 3-3 below. This configuration indicates the successful completion of the PDP-1 Instruction Test.

TABLE 3-3 COMPUTER STATE AT COMPLETION OF INSTRUCTION TEST

Register	Contents
PROGRAM COUNTER	000 000 000 001
MEMORY ADDRESS	000 000 000 000
MEMORY BUFFER	000 000 000 000 010 000
ACCUMULATOR	000 000 000 111 111 111
IN-OUT	111 111 111 000 000 000
PROGRAM FLAGS	1 1 1 1 1 1

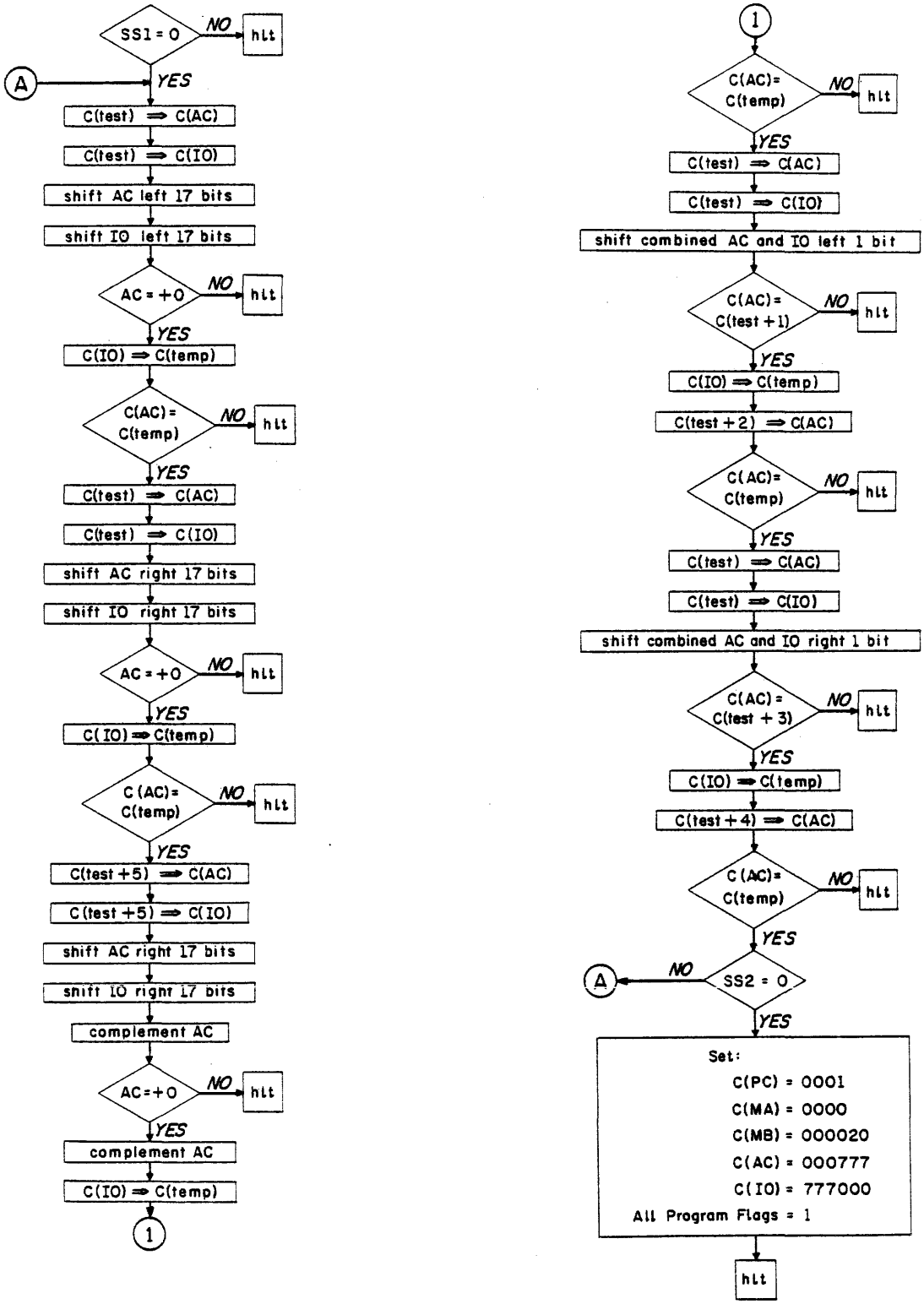


Figure 17 Instruction Test Program 20

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST

PROGRAM 1 (Clears Locations 0000 through 7766) and RIM LOADER

Location	Contents	Mnemonic Code	Remarks
7766	340000	zero dzm 000	START OF Program 1. Clears memory locations 0000 through 7766. (Note: location 7766 is indexed once after clearing.)
7767	447766	idx zero	Increments dzm instruction so that next location is cleared.
7770	467776	isp temp	Leaves routine after clearing location 7766.
7771	607766	jmp zero	Loops back to clear next location. END OF Program 1.
7772	730002	rimr rpb '	START OF the RIM loader. Reads an instruction from the tape.
7773	327776	dio temp	Deposits the instruction in location temp.
7774	652000	spi '	Executes the instruction if it is jmp and leaves the RIM loader.
7775	730002	rpb '	Reads another instruction from tape.
7776	770010	temp 770010	Temporary storage for dio instructions (which stores program read) and for jmp (to leave RIM loader). (Program 1 uses location to count up to 7766.)
7777	607772	jmp rimr	Loops back to start of RIM loader. END OF the RIM loader.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 2 (Tests hlt, jmp, szs)

Location	Contents	Mnemonic Code	Remarks
0000	000002	000002	Program number.
0001	760400	start hlt	Tests for hlt.
0002	603000	jmp 3000	Start of jmp test. Checks
0003	760400	hlt	$MB_{7,8} \xrightarrow{1} PC_{7,8}$.
0004	650010	sense szs ' 10	Tests szs' with all sense switches off.
0005	640010	szs 10	Tests szs with all sense switches off.
0006	760400	hlt	
0007	650020	szs ' 20	
0010	640020	szs 20	
0011	760400	hlt	
0012	650030	szs ' 30	
0013	640030	szs 30	
0014	760400	hlt	
0015	650040	szs ' 40	
0016	640040	szs 40	
0017	760400	hlt	
0020	650050	szs ' 50	
0021	640050	szs 50	
0022	760400	hlt	
0023	650060	szs ' 60	
0024	640060	szs 60	
0025	760400	hlt	
0026	650070	szs ' 70	
0027	640070	szs 70	
0030	760400	hlt	
0031	600002	jmp start + 1	Loops back to start of jmp test.
0032	640010	a szs 10	Tests szs with all sense switches on.
0033	650010	szs ' 10	Tests szs ' with all sense switches on.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 2

(continued)

Location	Contents	Mnemonic Code	Remarks
0034	760400	hlt	
0035	640020	szs 20	
0036	650020	szs ' 20	
0037	760400	hlt	
0040	640030	szs 30	
0041	650030	szs ' 30	
0042	760400	hlt	
0043	640040	szs 40	
0044	650040	szs ' 40	
0045	760400	hlt	
0046	640050	szs 50	
0047	650050	szs ' 50	
0050	760400	hlt	
0051	640060	szs 60	
0052	650060	szs ' 60	
0053	760400	hlt	
0054	640070	szs 70	
0055	650070	szs ' 70	
0056	760400	hlt	
0057	600032	jmp a	Iterates szs test with all switches on.
3000	604777	jmp 4777	Checks $\overset{0}{\rightarrow}$ $PC_{7,8}$ and
3001	760400	hlt	$MB_6 \overset{1}{\rightarrow} PC_6$ and
			$MB_{9-17} \overset{1}{\rightarrow} PC_{9-17}$.
4777	606000	jmp 6000	Checks $\overset{0}{\rightarrow}$ PC_{9-17} and
5000	760400	hlt	$MB_7 \overset{1}{\rightarrow} PC_7$.
6000	600004	jmp sense	Checks $\overset{0}{\rightarrow}$ $PC_{6,7}$ and leaves jmp
6001	760400	hlt	test.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 3

(Tests cla, cma, sma, spa, sza, 654000, 650000, lat partly)

Location	Contents	Mnemonic Code	Remarks
0000	000003	000003	Program number.
0001	640010	Start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	650000	650000	Test the two unconditional skips.
0004	760400	hlt	
0005	654000	654000	
0006	760400	hlt	
0007	640000	640000	Tests no selection on skip.
0010	600012	jmp a	
0011	760400	hlt	
0012	644000	a 644000	
0013	600016	jmp b	
0014	760400	hlt	
0015	762200	lat	Loads AC with all 1's in preparation for cla test. Note: no check for proper execution of lat.
0016	760200	b cla	Jointly tests cla and sza. If halt occurs,
0017	640100	sza	check AC to determine which of the two
0020	760400	hlt	instructions failed.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 3 (continued)

Location	Contents	Mnemonic Code	Remarks
0021	762200	lat	Tests cma, sza, and partially tests lat. If
0022	761000	cma	halt occurs, check AC to determine which
0023	640100	sza	instruction failed.
0024	760400	hlt	
0025	640200	spa	Tests spa for skip.
0026	760400	hlt	
0027	650400	sma '	Tests sma for skip.
0030	760400	hlt	
0031	761000	cma	Tests cma, sza for skip. If halt occurs,
0032	650100	sza '	check contents of AC to determine which
0033	760400	hlt	instruction failed.
0034	650200	spa '	Checks spa ' for skip.
0035	760400	hlt	
0036	640400	sma	Checks sma for skip.
0037	760400	hlt	
0040	640020	szs 20	With SS2 on, this program iterates. With SS2
0041	600003	jmp start +2	off, jumps to the RIM loader and reads in
0042	607772	jmp 7772	the next program.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 4 (Tests xor, sas, sad)

Location	Contents	Mnemonic Code	Remarks
0000	000004	000004	Program number.
0001	640010	start szs 10	If SS1 is on, program halts after reading in.
0002	760400	hlt	
0003	761200	761200	Loads accumulator with all 1's.
0004	060054	xor ones	START OF xor Test. Checks exclusive-OR
0005	640100	sza	of all 1's with all 1's.
0006	760400	hlt	
0007	060055	xor zeros	Checks exclusive-OR of all 0's with all 0's.
0010	640100	sza	
0011	760400	hlt	
0012	060054	xor ones	Checks exclusive-OR of all 0's with all 1's.
0013	761000	cma	
0014	640100	sza	
0015	760400	hlt	
0016	761000	cma	Checks exclusive-OR of all 1's with all 0's.
0017	060055	xor zeros	END OF xor Test.
0020	761000	cma	
0021	640100	sza	
0022	760400	hlt	
0023	520055	sas zeros	START OF sas and sad Tests. Compares 0's
0024	760400	hlt	with 1's and checks for skip.
0025	640100	sza	Checks that sas replaces contents of AC.
0026	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 4 (continued)

Location	Contents	Mnemonic Code	Remarks
0027	500054	sad ones	Compares 1's with 0's and checks for skip.
0030	760400	hlt	
0031	640100	sza	Checks that sad replaces contents of AC.
0032	760400	hlt	
0033	761000	cma	Checks that sas does not skip. AC, all 1's,
0034	520055	sas zeros	compared against all 0's.
0035	650000	650000	
0036	760400	hlt	
0037	761000	cma	Checks that sas replaces contents of AC.
0040	640100	sza	
0041	760400	hlt	
0042	761000	cma	Checks that sad does not skip. AC, all 1's,
0043	500054	sad ones	compared against all 1's.
0044	650000	650000	
0045	760400	hlt	
0046	761000	cma	Checks that sad replaces contents of AC.
0047	640100	sza	END OF sas and sad Test.
0050	760400	hlt	
0051	640020	szs 20	With SS2 on, this program iterates. With
0052	600003	jmp start +2	SS2 off, jumps to RIM loader and reads in
0053	607772	jmp 7772	next program.
0054	777777	ones 777777	Contains all 1's.
0055	000000	zeros 000000	Contains all 0's.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 5 (Tests dip, dap, dac, lac)

Location	Contents	Mnemonic Code	Remarks
0000	000005	000005	Program number.
0001	640010	start szs 10	If SS1 is on, program halts after reading in.
0002	760400	hlt	
0003	760200	cla	START OF dip, dap, and dac Test. Clears the AC.
0004	300063	dip regs	Checks that dip and dap deposit all 0's into all 0's.
0005	260063	dap regs	
0006	520063	sas regs	
0007	760400	hlt	
0010	240064	dac temp	Checks that dac deposits all 0's into all 0's.
0011	520064	sas temp	
0012	760400	hlt	
0013	761000	cma	Loads AC with all 1's.
0014	300063	dip regs	Checks that dip and dap deposit all 1's into all 0's.
0015	260063	dap regs	
0016	520063	sas regs	
0017	760400	hlt	
0020	240064	dac temp	Checks that dac deposits all 1's into all 0's.
0021	520064	sas temp	
0022	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 5 (continued)

Location	Contents	Mnemonic Code	Remarks
0023	300063	dip regs	Checks that dip and dap deposit all 1's into
0024	260063	dap regs	all 1's.
0025	520063	sas regs	
0026	760400	hlt	
0027	240064	dac temp	Checks that dac deposits all 1's into all 1's.
0030	520064	sas temp	
0031	760400	hlt	
0032	761000	cma	Clears AC.
0033	300063	dip regs	Checks that dip and dap deposit all 0's into
0034	260063	dap regs	all 1's.
0035	520063	sas regs	
0036	760400	hlt	
0037	240064	dac temp	Checks that dac deposits all 0's into all 1's.
0040	520064	sas temp	
0041	760400	hlt	END OF dip, dap, dac Test.
0042	200061	lac zeros	START OF lac Test. Checks that all 0's
0043	640100	sza	are loaded into AC, which contains all 0's.
0044	760400	hlt	
0045	200062	lac ones	Checks that all 1's are loaded into AC,
0046	520062	sas ones	which contains all 0's.
0047	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 5 (continued)

Location	Contents	Mnemonic Code	Remarks
0050	200062	lac ones	Checks that all 1's are loaded into AC,
0051	520062	sas ones	which contains all 1's.
0052	760400	hlt	
0053	200061	lac zeros	Checks that all 0's are loaded into AC, which
0054	640100	sza	contains all 1's.
0055	760400	hlt	END OF lac Test.
0056	640020	szs 20	With SS2 on, this program iterates. With SS2
0057	600003	jmp start +2	off, jumps to RIM loader and reads in next
0060	607772	jmp 7772	program.
0061	000000	zeros 000000	Contains all 0's.
0062	777777	ones 777777	Contains all 1's.
0063	000000	regs 000000	Test location for dip and dap.
0064	000000	temp 000000	Test location for dac.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 6 (Tests dzm, idx)

Location	Contents	Mnemonic Code	Remarks
0000	000006	000006	Program number.
0001	640010	start szs 10	If SS1 is on, program halts after reading in.
0002	760400	hlt	
0003	761200	761200	Loads AC with all 1's and deposits all 1's in
0004	240166	dac temp	test location for the dzm.
0005	340166	dzm temp	START OF dzm Test. Zeros the test location, which contains all 1's.
0006	200166	lac temp	Checks that test location was cleared.
0007	640100	sza	
0010	760400	hlt	
0011	340166	dzm temp	Zeros test location, which contains all 0's.
0012	200166	lac temp	Checks that test location was cleared. END
0013	640100	sza	OF dzm Test.
0014	760400	hlt	
0015	761200	761200	START OF idx Test. Loads AC with all 1's
0016	240166	dac temp	and deposits all 1's in test location temp.
0017	440166	idx temp	Indexes test location once to 000001.
0020	640200	spa	Checks that the carry propagated to AC_0.
0021	760400	hlt	
0022	650100	sza '	Checks that end-around carry was executed.
0023	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0024	761000	cma	Checks the contents due to previous idx, and also checks that the attempt to index -1 results in +0.
0025	240166	dac temp	
0026	440166	idx temp	
0027	640100	sza	
0030	760400	hlt	
0031	200167	lac tbl	Checks that the carry chain terminates at bit 17 of the AC. The number 377776 is deposited in the test location and indexed. If the carry chain does not stop correctly, the AC changes sign and the program halts.
0032	240166	dac temp	
0033	440166	idx temp	
0034	640200	spa	
0035	760400	hlt	
0036	200170	lac tbl + 1	Checks that the carry chain terminates at bit 16 of the AC by indexing the number 377775.
0037	240166	dac temp	
0040	440166	idx temp	
0041	640200	spa	
0042	760400	hlt	
0043	200171	lac tbl + 2	Checks that the carry chain terminates at bit 15 of the AC by indexing the number 377773.
0044	240166	dac temp	
0045	440166	idx temp	
0046	640200	spa	
0047	760400	hlt	
0050	200172	lac tbl + 3	Checks that the carry chain terminates at bit 14 of the AC by indexing the number 377767.
0051	240166	dac temp	
0052	440166	idx temp	
0053	640200	spa	
0054	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0055	200173	lac tbl + 4	Checks that the carry chain terminates at bit
0056	240166	dac temp	13 of the AC by indexing the number 377757.
0057	440166	idx temp	
0060	640200	spa	
0061	760400	hlt	
0062	200174	lac tbl + 5	Checks that the carry chain terminates at bit
0063	240166	dac temp	12 of the AC by indexing the number 377737.
0064	440166	idx temp	
0065	640200	spa	
0066	760400	hlt	
0067	200175	lac tbl + 6	Checks that the carry chain terminates at bit
0070	240166	dac temp	11 of the AC by indexing the number 377677.
0071	440166	idx temp	
0072	640200	spa	
0073	760400	hlt	
0074	200176	lac tbl + 7	Checks that the carry chain terminates at bit
0075	240166	dac temp	10 of the AC by indexing the number 377577.
0076	440166	idx temp	
0077	640200	spa	
0100	760400	hlt	
0101	200177	lac tbl + 10	Checks that the carry chain terminates at bit
0102	240166	dac temp	9 of the AC by indexing the number 377377.
0103	440166	idx temp	
0104	640200	spa	
0105	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0106	200200	lac tbl + 11	Checks that the carry chain terminates at bit
0107	240166	dac temp	8 of the AC by indexing the number 376777.
0110	440166	idx temp	
0111	640200	spa	
0112	760400	hlt	
0113	200201	lac tbl + 12	Checks that the carry chain terminates at bit
0114	240166	dac temp	7 of the AC by indexing the number 375777.
0115	440166	idx temp	
0116	640200	spa	
0117	760400	hlt	
0120	200202	lac tbl + 13	Checks that the carry chain terminates at bit
0121	240166	dac temp	6 of the AC by indexing the number 373777.
0122	440166	idx temp	
0123	640200	spa	
0124	760400	hlt	
0125	200203	lac tbl + 14	Checks that the carry chain terminates at bit
0126	240166	dac temp	5 of the AC by indexing the number 367777.
0127	440166	idx temp	
0130	640200	spa	
0131	760400	hlt	
0132	200204	lac tbl + 15	Checks that the carry chain terminates at bit
0133	240166	dac temp	4 of the AC by indexing the number 357777.
0134	440166	idx temp	
0135	640200	spa	
0136	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 6 (continued)

Location	Contents	Mnemonic Code	Remarks
0137	200205	lac tb1 + 16	Checks that the carry chain terminates at bit
0140	240166	dac temp	3 of the AC by indexing the number 337777.
0141	440166	idx temp	
0142	640200	spa	
0143	760400	hlt	
0144	200206	lac tb2	Checks that the carry chain terminates at bit
0145	240166	dac temp	2 of the AC by indexing the number 677777.
0146	440166	idx temp	
0147	640400	sma	
0150	760400	hlt	
0151	200207	lac tb2+1	Checks that the carry chain terminates at bit
0152	240166	dac temp	1 of the AC by indexing the number 577777.
0153	440166	idx temp	
0154	640400	sma	
0155	760400	hlt	
0156	200210	lac tb2 + 2	Checks that the carry chain terminates at bit
0157	240166	dac temp	0 of the AC by indexing the number 377777.
0160	440166	idx temp	
0161	640400	sma	END OF idx Test.
0162	760400	hlt	
0163	640020	szs 20	With SS2 on, this program is iterated. With
0164	600003	jmp start +2	SS2 off, the program jumps to the RIM loader
0165	607772	jmp 7772	and reads in the next program.
0166	000000	temp 000000	Temporary storage.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 6 (continued)

Location	Contents	Mnemonic Code	Remarks
0167	377776	tb1 377776	Numbers to test carry chain termination in
0170	377775	377775	AC bits 3 through 17.
0171	377773	377773	
0172	377767	377767	
0173	377756	377756	
0174	377737	377737	
0175	377677	377677	
0176	377577	377577	
0177	377377	377377	
0200	376777	376777	
0201	375777	375777	
0202	373777	373777	
0203	367777	367777	
0204	357777	357777	
0205	337777	337777	
0206	677777	tb2 677777	Numbers to test carry chain termination in
0207	577777	577777	AC bits 0 through 2.
0210	377777	377777	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 7 (Tests isp)

Location	Contents	Mnemonic Code	Remarks
0000	000007	000007	Program number.
0001	640010	start szs 10	If SS1 is on, computer halts after reading in.
0002	760400	hlt	
0003	200050	lac tn1	Initializes cnt1 and t1 with the number 377777
0004	240044	dac cnt1	Initializes cnt2 and t2 with the number 777776
0005	240046	dac t1	
0006	200051	lac tn2	
0007	240045	dac cnt2	
0010	240047	dac t2	
0011	460047	loop isp t2	START OF isp Test. Indexes t2 to 000000 and through to 377777 and checks that it skips each time.
0012	760400	hlt	
0013	440045	idx cnt2	Checks that t2 was indexed to the correct number.
0014	520047	sas t2	
0015	760400	hlt	
0016	500050	sad tn1	Leaves loop when cnt2 has been indexed to 377777.
0017	600027	jmp last	
0020	460046	isp t1	Indexes t1 to 400000 and through to 777776 and checks each time that it does not skip.
0021	650000	650000	
0022	760400	hlt	
0023	440044	idx cnt1	Checks that t1 was indexed to the correct number.
0024	520046	sas t1	
0025	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 7 (continued)

Location	Contents	Mnemonic Code	Remarks
0026	600011	jmp loop	Jumps to the start of the loop.
0027 0030	460046 760400	last isp t1 hlt	Indexes the number 777776 to 000000. Also checks that a skip occurs.
0031 0032	640100 760400	sza hlt	Checks that the result of the index was 000000.
0033 0034 0035	460047 650000 760400	isp t2 650000 hlt	Indexes the number 377777 to 400000. Also checks that no skip results.
0036 0037 0040	440045 520047 760400	idx cnt2 sas t2 hlt	Checks that the result of the index was 400000. END OF isp Test.
0041 0042 0043	640020 600003 607772	szs 20 jmp start +2 jmp 7772	With SS2 on, the program iterates. With SS2 off, program jumps to RIM loader and reads in next program.
0044	000000	cnt1 000000	Contains number which is compared against t1; to check isp.
0045	000000	cnt2 000000	Contains number which is compared against t2; to check isp.
0046	000000	t1 000000	Test location 1.

Location	Contents	Mnemonic Code	Remarks
0047	000000	t2 000000	Test location 2.
0050	377777	tn1 377777	Test number 1.
0051	777776	tn2 777776	Test number 2.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 10 (Tests and, ior)

Location	Contents	Mnemonic Code	Remarks
0000	000010	000010	Program number.
0001	640010	start szs 10	With SS1 on, the program halts after reading
0002	760400	hlt	in.
0003	200047	lac t2	START OF and Test. ANDs all 1's with all
0004	020047	and t2	1's. Checks that the result is all 1's.
0005	520047	sas t2	
0006	760400	hlt	
0007	200046	lac t1	ANDs all 0's with all 1's. Checks that the
0010	020047	and t2	result is all 0's.
0011	640100	sza	
0012	760400	hlt	
0013	200046	lac t1	ANDs all 0's with all 0's. Checks that the
0014	020046	and t1	result is all 0's.
0015	640100	sza	
0016	760400	hlt	
0017	200047	lac t2	ANDs all 1's with all 0's. Checks that the
0020	020046	and t1	result is all 0's.
0021	640100	sza	END OF and Test.
0022	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 10 (continued)

Location	Contents	Mnemonic Code	Remarks
0023	200046	lac t1	START OF ior Test. Forms the inclusive-OR
0024	040046	ior t1	of all 0's with all 0's. Checks that the result
0025	640100	sza	is all 0's.
0026	760400	hlt	
0027	200046	lac t1	Forms the inclusive-OR of all 0's with all 1's.
0030	040047	ior t2	Checks that the result is all 1's.
0031	520047	sas t2	
0032	760400	hlt	
0033	200047	lac t2	Forms the inclusive-OR of all 1's with all 1's.
0034	040047	ior t2	Checks that the result is all 1's.
0035	520047	sas t2	
0036	760400	hlt	
0037	200047	lac t2	Forms the inclusive-OR of all 1's with all 0's.
0040	040046	ior t1	Checks that the results is all 1's. END OF
0041	520047	sas t2	ior Test.
0042	760400	hlt	
0043	640020	szs 20	With SS2 on, the program iterates. With
0044	600003	jmp start+2	SS2 off, the program jumps to RIM loader
0045	607772	jmp 7772	and reads in next program.
0046	000000	t1 000000	Contains all 0's.
0047	777777	t2 777777	Contains all 1's.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 11

(Tests lio, dio, spi)

Location	Contents	Mnemonic Code		Remarks
0000	000011		000011	Program number.
0001 0002	640010 760400	start	szs 10 hlt	With SS1 on, the program halts after reading in.
0003	340033		dzm t1	Initializes t1 with all 0's.
0004	760200		cla	Clears AC for the test.
0005 0006	220033 320034	a	lio t1 dio t2	START OF lio, dio, and spi Test. Loads IO from t1; deposits IO in t2.
0007 0010	520034 760400		sas t2 hlt	Checks that contents of t2 are same as those of t1. If error halt occurs, check IO to determine which of the two instructions failed.
0011 0012	640200 600017		spa jmp b	Jumps out of loop when number in t1 reaches 400000.
0013 0014	642000 760400		spi hlt	Tests spi to ensure that it skips.
0015	440033		idx t1	Increments by 1 the number in t1.
0016	600005		jmp a	Jumps to start of loop at a.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 11 (continued)

Location	Contents	Mnemonic Code		Remarks
0017 0020	652000 760400	b	spi ' hlt	Checks that spi ' skips on a negative IO.
002?	440033		idx t1	Increments by 1 the number in t1.
0022 0023	220033 320034		lio t1 dio t2	Loads IO from t1; deposits IO in t2.
0024 0025	520034 760400		sas t2 hlt	Checks that contents of t2 are same as those of t1.
0026	640100		sza	Jumps out of loop when number in t1 is 000000.
0027	600017		jmp b	Jumps to start of loop at b. END OF lio, dio, spi Test.
0030 0031 0032	640020 600003 607772		szs 20 jmp start+2 jmp 7772	With SS2 on, program iterates. With SS2 off, program jumps to RIM loader and reads in next program.
0033	000000	t1	000000	Test location 1.
0034	000000	t2	000000	Test location 2.

PROGRAM LISTING

MAINDEC 1- INSTRUCTION TEST PROGRAM 12

(Tests add, szo, sub)

Location	Contents	Mnemonic Code	Remarks
0000	000012	000012	Program number.
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	641000	szo	START OF szo Test. Turns off OVERFLOW.
0004	651000	szo '	Checks that szo ' does not skip.
0005	641000	szo	Checks that szo does not skip.
0006	760400	hlt	
0007	651000	szo '	Checks that szo ' does not skip.
0010	650000	650000	
0011	760400	hlt	
0012	760200	cla	START OF add Test. Checks partial-add
0013	400116	add zero	of all 0's to all 0's.
0014	640100	sza	
0015	760400	hlt	
0016	400117	add pmax	Checks partial-add of 377777 to 000000.
0017	520117	sas pmax	
0020	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 12 (continued)

Location	Contents	Mnemonic Code	Remarks
0021	400116	add zero	Checks partial-add of 000000 to
0022	520117	sas pmax	377777.
0023	760400	hlt	
0024	760200	cla	Checks partial-add of 400000 to
0025	400120	add nmax	000000.
0026	520120	sas nmax	
0027	760400	hlt	
0030	400116	add zero	Checks partial-add of 000000 to
0031	520120	sas nmax	400000.
0032	760400	hlt	
0033	641000	szo	Checks that OVERFLOW is off.
0034	760400	hlt	
0035	400117	add pmax	Checks that 400000 add to
0036	640100	sza	377777 results in all 0's.
0037	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0040	200121	lac zoz	Sets OVERFLOW and checks the
0041	400121	add zoz	full-register carry for the odd
0042	520122	sas ozo	bits in the AC (AC_1, AC_3, AC_5,
0043	760400	hlt	etc.)
0044	651000	szo '	Tests that OVERFLOW was set,
0045	760400	hlt	and clears it.
0046	200123	lac ab	Checks full-register carry for
0047	400123	add ab	even bits of the AC (AC_0, AC_2,
0050	520124	sas ac	etc.)
0051	760400	hlt	
0052	200115	lac ones	Checks clear-on-AC-minus-0.
0053	400116	add zero	
0054	640100	sza	
0055	760400	hlt	
0056	641000	szo	Checks that OVERFLOW is off.
0057	760400	hlt	
0060	761000	cma	Checks that ripple-carry propagates
0061	400125	add ad	entire length of AC by adding 000001
0062	520125	sas ad	to 777777.
0063	760400	hlt	

MAINDEC 1 - INSTRUCTION TEST PROGRAM 12 (continued)

Location	Contents	Mnemonic Code		Remarks
0064	641000		szo	Checks that OVERFLOW is off.
0065	760400		hlt	
0066	200122		lac ozo	Checks that carry chain initiates
0067	400115		add ones	in all even bits of AC (AC_0, AC_2,
0070	520122		sas ozo	etc.)
0071	760400		hlt	
0072	200121		lac zoz	Checks that carry chain initiates
0073	400115		add ones	in all odd bits of AC.
0074	520121		sas zoz	
0075	760400		hlt	
0076	641000		szo	Checks that OVERFLOW is off.
0077	760400		hlt	END OF szo and add Test.
0100	760200		cla	Initializes in preparation for the
0101	240127		dac temp	sub test.
0102	761000		cma	
0103	420126	loop	sub mone	START OF sub Test. Adds 1 to
0104	240130		dac qemp	AC (subtracts -1) and deposits
0105	440127		idx temp	AC in location qemp. Checks by
0106	520130		sas qemp	incrementing location temp, by 1, and
0107	760400		hlt	comparing contents against qemp.

Location	Contents	Mnemonic Code	Remarks
0110	520116	sas zero	Jumps out of loop when contents of temp reach 0 (temp is initially 0 and is incremented by 1 until it becomes 0 again).
0111	600103	jmp loop	Jumps to start of loop. END OF sub Test.
0112 0113 0114	640020 600003 607772	szs 20 jmp start+2 jmp 7772	With SS2 on, program iterates. With SS2 off, program jumps to RIM loader and reads in next program.
0115	777777	ones 777777	Location containing all 1's.
0116	000000	zero 000000	Location containing all 0's.
0117	377777	pmax 377777	Maximum positive number.
0120	400000	nmax 400000	Maximum negative number.
0121	252525	zoz 252525	Alternate 0's and 1's.
0122	525252	ozo 525252	Alternate 1's and 0's.

Location	Contents	Mnemonic Code	Remarks
0123	125252	ab 125252	Number used in full register carry test.
0124	252524	ac 252524	Result of full register carry test.
0125	000001	ad 000001	Plus one.
0126	777776	mone 777776	Minus one.
0127	000000	temp 000000	Test location used as temporary storage.
0130	000000	qemp 000000	Test location used as temporary storage.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 13

(Tests law, cli)

Location	Contents	Mnemonic Code	Remarks
0000	000013	000013	Program number.
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	761200	761200	START OF law Test. Loads AC with all 1's.
0004	700000	law 0	Checks loading of AC with +0.
0005	640100	sza	
0006	760400	hlt	
0007	707777	law 7777	Checks loading of AC with +7777.
0010	520042	sas ck	
0011	760400	hlt	
0012	710000	law ' 0	Checks loading of AC with -0000.
0013	761000	cma	
0014	640100	sza	
0015	760400	hlt	
0016	717777	law ' 7777	Checks loading of AC with -7777.
0017	520043	sas dk	END OF law Test.
0020	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0021 0022	761200 240041	761200 dac temp	START OF cli Test. Loads AC with all 1's and deposits 1's in location temp.
0023	220041	lio temp	Loads IO with all 1's.
0024 0025 0026 0027 0030	764000 320041 200041 640100 760400	cli dio temp lac temp sza hlt	Checks that IO is cleared when it contains all 1's.
0031 0032 0033 0034 0035	764000 320041 200041 640100 760400	cli dio temp lac temp sza hlt	Checks that IO is cleared when it contains all 0's. END OF cli Test.
0036 0037 0040	640020 600003 607772	szs 20 jmp start+2 jmp 7772	With SS2 on, program iterates. With SS2 off, program jumps to RIM loader and reads in next program.
0041	000000	temp 000000	Location used for transfer between IO and AC.
0042	007777	ck 007777	Check number for law 7777.

Location	Contents	Mnemonic Code	Remarks
0043	770000	dk 770000	Check number for law -7777.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 14

(Tests clf, stf, szf)

Location	Contents	Mnemonic Code	Remarks
0000	000014	000014	Program number.
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	760007	clf 7	Clears any flags that may be on, and checks
0004	640007	szf 7	that all flags are off.
0005	760400	hlt	
0006	200066	lac e	Initializes all instructions for flag 1.
0007	260020	dap a + 1	
0010	200067	lac f	
0011	260021	dap a + 2	
0012	260045	dap c + 1	
0013	260046	dap c + 2	
0014	200072	lac cde	Sets up count for loop which sets flags.
0015	240071	dac abc	
0016	700020	xa law a + 1	Introduces a time delay.
0017	600060	a jmp count + 1	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 14 (continued)

Location	Contents	Mnemonic Code	Remarks
0020	760011	stf 1	Sets flag (beginning with 1) and checks that it
0021	650001	szf ' 1	was set. If a halt occurs, flag must be checked
0022	760400	hlt	to determine which instruction caused the halt.
0023	440020	idx a + 1	Increments so that the next flag
0024	440021	idx a + 2	is checked (2, 3, . . ., 6).
0025	460071	isp abc	Jumps out of loop when all six flags have been set and checked.
0026	600016	jmp xa	Jumps to start of loop.
0027	700031	law b	Introduces a time delay.
0030	600060	jmp count+1	
0031	760007	b clf 7	Clears all flags and checks that all are cle .
0032	640007	szf 7	If halt occurs, flags must be checked to deter-
0033	760400	hlt	mine which instruction caused the halt.
0034	700036	law b + 5	Introduces a time delay.
0035	600060	jmp count+1	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 14 (continued)

Location	Contents	Mnemonic Code	Remarks
0036 0037 0040	760017 650007 760400	stf 7 szf ' 7 hlt	Sets all flags and checks that all flags were set. If halt occurs, all flags must be checked to determine which instruction caused the halt.
0041 0042	200072 240071	lac cde dac abc	Sets up count for loop which clears flags.
0043 0044	700045 600060	xc law c + 1 c jmp count+1	Introduces a time delay.
0045 0046 0047	760001 640001 760400	clf 1 szf 1 hlt	Clears flags (beginning with 1) and checks that each flag was cleared. If halt occurs, flag must be checked to determine which instruction caused the halt.
0050 0051	440045 440046	idx c + 1 idx c + 2	Indexes instructions so that next flag is checked.
0052	460071	isp abc	Leaves loop when all six flags have been cleared.
0053	600043	jmp xc	Jumps to start of loop.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 14 (continued)

Location	Contents	Mnemonic Code		Remarks
0054	640020		szs 20	With SS2 on, program iterates.
0055	600006		jmp start+5	With SS2 off, program jumps to
0056	607766		jmp 7766	the sequence which clears memory.
0057	000000	count	000000	Sequence introduces time delay
0060	260065		dap count+6	so that action of program flags is
0061	200070		lac mxx	visible from console.
0062	240057		dac count	
0063	460057		isp count	
0064	600063		jmp count+4	
0065	600000		jmp . . .	
0066	000011	e	000011	Numbers used to set the flag
0067	000001	f	000001	instruction to operate on flag 1.
0070	700000	mxx	700000	Number used to count up the count loop.
0071	000000	abc	000000	Location loaded with number in cde to count up to 6 in flag loops.
0072	777771	cde	777771	Number loaded into abc for counting up flag loops.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 14 (continued)

Location	Contents	Mnemonic Code		Remarks
7765	770013		770013	Sequence clears memory locations
7766	340000	zero	dzm 0000	0000 through 7764 in preparation
7767	447766		idx zero	for next program. After location
7770	467765		isp 7765	7765 is cleared, program jumps
7771	607766		jmp zero	to RIM loader and reads in next program.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15

(Tests jsp, ida, cal)

Location	Contents	Mnemonic Code		Remarks
0000	000015		000015	Program number.
0001 0002	640010 760400	start	szs 10 hlt	With SS1 on, program halts after reading in.
0003 0004	641000 651000		szo szo '	Clears OVERFLOW.
0005	600016		jmp test1	Goes to first jsp test.
0016	200120	test1	lac num	Initializes AC to 777757.
0017	620757	jsp1	jsp chek1	Checks bit 13 on $\xrightarrow{0}$ PC. Checks bits 9-12 and 14-17 on MB $\xrightarrow{1}$ PC.
0032 0033	200123 240737	test4	lac num+3 dac chek4	Deposits 000040 into location equal to jda address.
0034 0035	641000 651000		szo szo '	Clears OVERFLOW.
0036	200124		lac num+4	Initializes AC to 771737.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15 (continued)

Location	Contents	Mnemonic Code	Remarks
0037	170737	jda1 jda chek4	Checks bit 12 on $\xrightarrow{0}$ PC. Checks bits 9-11 and 13-17 on MA $\xrightarrow{1}$ PC. Checks $\xrightarrow{+1}$ PC.
0100	000000	000000	Stores number that AC contained when cal was executed.
0101	020117	and mask	Masks out transfer of EXD flip-flop to AC_1.
0102	240116	dac temp	Saves AC.
0103	703002	law call+1	Contents of PC when cal was executed.
0104 0105	520116 760400	sas temp hlt	Checks that PC was saved.
0106 0107 0110	200100 520114 760400	lac 0100 sas ones hlt	Checks that AC was saved in location 100.
0111 0112 0113	640020 600003 607772	szs 20 jmp start+2 jmp 7772	With SS2 on, program iterates. With SS2 off, program jumps to RIM loader and reads in next program.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15 (continued)

Location	Contents	Mnemonic Code		Remarks
0114	777777	ones	777777	Locations that hold test numbers or constants; or that are used for temporary storage.
0115	400000	ovflo	400000	
0116	000000	temp	000000	
0117	577777	mask	577777	
0120	777757	num	777757	
0121	776020		776020	
0122	771677		771677	
0123	000040		000040	
0124	771737		771737	
0125	006040		006040	
0126	006000		006000	
0127	771777		771777	
0737	000000	chek4	000000	Stores number that AC contained when the first jda was executed.
0740	020117		and mask	Masks out transfer of EXD flip-flop to AC
0741	240116		dac temp	Saves AC.
0742	700040		law jda 1+1	Contents of PC when the first jda was executed
0743	520116		sas temp	Checks bits 0-5, 8-11 and 13-17 on $\xrightarrow{0}$ AC.
0744	760400		hlt	Checks bit 12 on PC $\xrightarrow{1}$ AC.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15 (continued)

Location	Contents	Mnemonic Code	Remarks
0745 0746 0747	200737 520124 760400	lac chek4 sas num+4 hlt	Checks AC $\dashv\!\!\rightarrow$ MB. Correct contents of AC equal 771737.
0750	601731	jmp test5	Goes to second jda test.
0757	020117	chek1 and mask	Masks out transfer of EXD flip-flop to AC_1.
0760	240116	dac temp	Saves AC.
0761	700020	law jsp 1+1	Contents of PC when first jsp was executed.
0762 0763	520116 760400	sas temp hlt	Checks bits 0-12 and 14-17 on $\overset{10}{\longrightarrow}$ AC. Checks bit 13 on PC $\overset{1}{\longrightarrow}$ AC.
0764	601755	jmp test2	Goes to second jsp test.
1000	000000	chek6 000000	Stores number that AC contained when the third jda was executed.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15 (continued)

Location	Contents	Mnemonic Code	Remarks
1001	020117	and mask	Masks out transfer of EXD flip-flop to AC_1.
1002	240116	dac temp	Saves AC.
1003	706000	law jda3+1	Contents of PC when third jda was executed.
1004	520116	sas temp	Checks bits 0-5 and 8-17 on $\xrightarrow{0}$ AC.
1005	760400	hlt	Checks bits 6 and 7 on PC $\xrightarrow{1}$ AC.
1006	201000	lac chek6	Checks AC $\dashv\!\!\rightarrow$ MB. Correct contents of
1007	520127	sas num+7	AC equal 771777.
1010	760400	hlt	
1011	603000	jmp test7	Goes to cal test.
1100	020117	chek3 and mask	Masks out transfer of EXD flip-flop to AC_1.
1101	240116	dac temp	Saves AC.
1102	706100	law jsp3+1	Contents of PC when third jsp was executed.
1103	520116	sas temp	Checks bits 0-5, 8-10 and 12-17 on $\xrightarrow{0}$ AC.
1104	760400	hlt	Checks bits 6, 7, and 11 on PC $\xrightarrow{1}$ AC.

Location	Contents	Mnemonic Code		Remarks
1105	600032		jmp test4	Goes to first jda test.
1731 1732	201731 401731	test5	lac test5 add test5	Turns on OVERFLOW.
1733 1734	200124 246040		lac num+4 dac chek5	Deposits 771737 into location equal to address of second jda.
1735	200125		lac num+5	Initializes AC to 006040.
1736	176040	jda2	jda chek5	Checks bits 8-11 and 13-17 on $\xrightarrow{0}$ PC. Check bits 6, 7, and 12 on MA $\xrightarrow{1}$ PC. Checks $\xrightarrow{+1}$ PC.
1755	200121	test2	lac num+1	Initializes AC to 776020.
1756	626020	jsp2	jsp chek2	Checks bits 8-12 and 14-17 on $\xrightarrow{0}$ PC. Checks bits 6, 7, and 13 on MB $\xrightarrow{1}$ PC.
3000	761200	test7	761200	Initializes AC to all 1's.
3001	160000	call	cal	Checks that cal signal is properly decoded.
5772 5773	641000 651000	test6	szo szo '	Clears OVERFLOW.

Location	Contents	Mnemonic Code		Remarks
5774 5775	200126 241000		lac num+6 dac chek6	Deposits 006000 into location equal to address of third jda.
5776	200127		lac num+7	Initializes AC to 771777.
5777	171000	jda3	jda chek6	Checks bits 6 and 7 on $\xrightarrow{0}$ PC. Checks bit 6 on MA $\xrightarrow{1}$ PC. Checks $\xrightarrow{+1}$ PC.
6020	020117	chek2	and mask	Masks out transfer of EXD flip-flop to AC_1.
6021	240116		dac temp	Saves AC.
6022	701757		law jsp2+1	Contents of PC when the second jsp was executed.
6023 6024	520116 760400		sas temp hlt	Checks bits 0-7 and 13 on $\xrightarrow{0}$ AC. Checks bits 8-12 and 14-17 on PC $\xrightarrow{1}$ AC.
6025	606076		jmp test3	Goes to third jsp test.
6040	000000	chek5	000000	Stores number contained in AC when the second jda was executed.
6041	020117		and mask	Masks out transfer of EXD flip-flop to AC_1.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 15 - (continued)

Location	Contents	Mnemonic Code	Remarks
6042	240116	dac temp	Saves AC.
6043	701737	law jda2 +1	Contents of PC when second jda was executed.
6044	040115	ior ovflo	To check transfer of OVERFLOW to AC_0.
6045 6046	520116 760400	sas temp hlt	Checks bits 6, 7, and 12 on $\xrightarrow{0}$ AC. Checks bits 8-11 and 13-17 on PC $\xrightarrow{1}$ AC. Checks transfer of OVERFLOW to AC_0.
6047 6050 6051	206040 520125 760400	lac chek5 sas num+5 hlt	Checks AC $\not\rightarrow$ MB. Correct contents of AC equal 006040.
6052	605772	jmp test6	Goes to third jda test.
6076	200122	test3 lac num+2	Initializes AC to 771677.
6077	521100	jsp3 jsp chek3	Checks bits 6, 7, and 11 on $\xrightarrow{0}$ PC. Checks bits 8 and 11 on MB $\xrightarrow{1}$ PC.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 16

(Tests nop, xct, and deferred addressing)

Location	Contents	Mnemonic Code	Remarks
0000	000016	000016	Program number.
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	760000	nop	If computer halts here, program failed to execute nop.
0004	100014	xct inst	Checks xct by jumping to next.
0005	760400	hlt	Program failed to execute the xct instruction.
0006	350007	t1 dzm ' t2	Used to give address t6 to instruction located in next.
0007	350010	t2 dzm ' t3	
0010	350011	t3 dzm ' t4	
0011	350012	t4 dzm ' t5	
0012	340013	t5 dzm t6	
0013	777777	t6 777777	
0014	600015	inst jmp next	Instruction which is executed by xct.
0015	760200	next cla	Clears AC and checks that deferable addressing was correctly executed.
0016	210006	lac ' t1	
0017	520013	sas t6	
0020	760400	hlt	
0021	640020	szs 20	With SS2 on, program iterates. With SS2 off, program jumps to RIM loader and reads in next program.
0022	600003	jmp start + 2	
0023	607772	jmp 7772	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17

(Tests ral, ril, rar, rir, rcl, rcr)

Location	Contents	Mnemonic Code	Remarks
0000	000017	000017	Program number.
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	200261	lac ral 1	START OF ral s1 and ril s1 Test. Sets up
0004	240014	dac 1shft	instructions for rotate left one bit.
0005	200262	lac ril 1	
0006	240015	dac 1shft+1	
0007	700303	law test+1	
0010	260016	dap 1shft+2	
0011	220302	lio test	
0012	320265	loopleft dio temp	Sets contents of AC equal to contents of IO.
0013	200265	lac temp	
0014	661000	1shft ral . . .	Rotates AC left one bit. This operation is repeated nine times; each time a different bit is used to specify the rotation.
0015	662000	ril . . .	Rotates IO left one bit. This operation is repeated nine times; each time a different bit is used to specify the rotation.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0016	520000	sas . . .	Checks that AC was rotated.
0017	760400	hlt	
0020	320265	dio temp	Checks that IO was rotated.
0021	520265	sas temp	
0022	760400	hlt	
0023	200014	lac lshft	Moves the 1, which specifies the single rota-
0024	020276	and mask	tion, left one bit in the ral instruction.
0025	400014	add lshft	
0026	500277	sad lastleft	Jumps out of rotate left loop if the registers
0027	600037	jmp setup r	have been rotated nine bits left.
0030	240014	dac lshft	Sets up the next ral instruction.
0031	200015	lac lshft+1	Moves the 1, which specifies the single rota-
0032	020276	and mask	tion, left one bit in the ril instruction.
0033	400015	add lshft+1	
0034	240015	dac lshft+1	
0035	440016	idx lshft+2	Sets address of the sas instruction that checks left rotation of AC.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0036	600012	jmp loopleft	Jumps to start of rotate left loop. END OF ral sl and ril sl Test.
0037	200263	setupr lac rarl	START OF rar sl and rir sl Test. Sets up loop for right rotation.
0040	240050	dac rshft	
0041	200264	lac rirl	
0042	240051	dac rshft+1	
0043	700312	law test+10	
0044	260052	dap rshft+2	
0045	220313	lio test+11	
0046	320265	loopright dio temp	Sets contents of AC equal to contents of IO.
0047	200265	lac temp	
0050	671000	rshft rar . . .	Rotates AC right one bit. This operation is repeated nine times; each time a different bit is used to specify the rotation.
0051	672000	rir . . .	Rotates IO right one bit. This operation is repeated nine times; each time a different bit is used to specify the rotation.
0052	520000	sas . . .	Checks that AC was rotated.
0053	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0054	320265	dio temp	Checks that IO was rotated.
0055	520265	sas temp	
0056	760400	hlt	
0057	200050	lac rshft	Moves the 1, which specifies the single rotation, left one bit position.
0060	020276	and mask	
0061	400050	add rshft	
0062	500300	sad lastright	Jumps out of loop if registers have been shifted right nine times.
0063	600075	jmp next	
0064	240050	dac rshft	Finishes setting up the rar instruction.
0065	200051	lac rshft+1	Moves the 1, which specifies the single rotation, left one bit in the rir instruction.
0066	020276	and mask	
0067	400051	add rshft+1	
0070	240051	dac rshft+1	
0071	200052	lac rshft+2	Sets address of the sas instruction that checks right rotation of AC.
0072	420270	sub one	
0073	240052	dac rshft+2	
0074	600046	jmp loopright	Jumps to start of rotate right loop. END OF rar sl and rir sl Test.

Location	Contents	Mnemonic Code		Remarks
0075	200302	next	lac test	START OF 9 bit rotation Test. Loads
0076	220302		lio test	AC and IO with test number.
0077	661777		ral s9	Rotates both registers lett
0100	662777		ril s9	nine bits.
0101	520313		sas test+11	Checks that AC was correctly
0102	760400		hlt	rotated.
0103	320265		dio temp	Checks that IO was correctly
0104	520265		sas temp	rotated.
0105	760400		hlt	
0106	671777		rar s9	Rotates both registers right
0107	672777		rir s9	nine bits.
0110	520302		sas test	Checks that AC was correctly
0111	760400		hlt	rotated.
0112	320265		dio temp	Checks that IO was correctly
0113	520265		sas temp	rotated.
0114	760400		hlt	
0115	200313		lac test+11	Sets up the AC for combined rotation test.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0116	663777	rcl s9	Rotates combined registers left nine bits.
0117	520315	sas testc	Checks that AC has correct contents.
0120	760400	hlt	
0121	200316	lac testc +1	Checks that IO has correct contents.
0122	320265	dio temp	
0123	520265	sas temp	
0124	760400	hlt	
0125	200315	lac testc	Sets up AC for combined right rotation.
0126	673777	rcr s9	Rotates combined registers right nine bits.
0127	520313	sas test+11	Checks that AC has correct contents.
0130	760400	hlt	
0131	200302	lac test	Checks that IO has correct contents. END
0132	320265	dio temp	OF 9 bit rotation Test.
0133	520265	sas temp	
0134	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0135	661000	ral	START OF 0 bit rotation Test. Executes
0136	662000	ril	ral and ril with no bits specified.
0137	520302	sas test	Checks that AC did not rotate.
0140	760400	hlt	
0141	320265	dio temp	Checks that IO did not rotate.
0142	520265	sas temp	
0143	760400	hlt	
0144	671000	rar	Executes rar and rir with no bits specified.
0145	672000	rir	
0146	520302	sas test	Checks that AC did not rotate.
0147	760400	hlt	
0150	320265	dio temp	Checks that IO did not rotate.
0151	520265	sas temp	
0152	760400	hlt	
0153	663000	rcl	Executes rcl with no bits specified.
0154	520302	sas test	Checks that contents of AC are correct.
0155	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0156	320265	dio temp	Checks that contents of IO are correct.
0157	520265	sas temp	
0160	760400	hlt	
0161	673000	rcr	Executes rcr with no bits specified.
0162	520302	sas test	Checks that contents of AC are correct.
0163	760400	hlt	
0164	320265	dio temp	Checks that contents of IO are correct. END
0165	520265	sas temp	OF 0 bit rotation Test.
0166	760400	hlt	
0167	700266	law flip	Sets up loops for fast rotation.
0170	260173	dap fastloops	
0171	200274	lac number	START OF rcl s9 high speed Test. Sets up
0172	240275	dac count	location which counts the number of loops.
0173	220000	fastloops lio ...	Loads IO with test number.
0174	320265	fastleft dio temp	Sets contents of AC equal to IO.
0175	200265	lac temp	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0176	663777	rcl s9	Rotates combined registers 36 bits left.
0177	663777	rcl s9	
0200	663777	rcl s9	
0201	663777	rcl s9	
0202	530173	sas ' fastloops	Checks that contents of AC are correct.
0203	760400	hlt	
0204	320265	dio temp	Checks that contents of IO are correct.
0205	520265	sas temp	
0206	760400	hlt	
0207	460275	isp count	Leaves loop when it has been executed 2^8
0210	600174	jmp fastleft	times. END OF rcl s9 high speed Test.
0211	200274	lac number	START OF rcr s9 high speed Test. Sets up
0212	240275	dac count	location which counts the number of loops.
0213	320265	fastright dio temp	Sets contents of AC equal to IO.
0214	200265	lac temp	
0215	673777	rcr s9	Rotates combined registers right 36 bits.
0216	673777	rcr s9	
0217	673777	rcr s9	
0220	673777	rcr s9	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0221	530173	sas ' fastloops	Checks that contents of AC are correct.
0222	760400	hlt	
0223	320265	dio temp	Checks that contents of IO are correct.
0224	520265	sas temp	
0225	760400	hlt	
0226	460275	isp count	Leaves loop when it has been executed 2^8
0227	600213	jmp fastright	times. END OF rcr s9 high speed Test.
0230	200274	lac number	START OF rcl s9 - rcr s9 high speed Test.
0231	240275	dac count	Sets up location which counts the number of loops.
0232	320265	reverse dio temp	Sets contents of AC equal to contents of IO.
0233	200265	lac temp	
0234	673777	rcr s9	Alternates combined nine-bit left and right
0235	663777	rcl s9	rotations.
0236	673777	rcr s9	
0237	663777	rcl s9	
0240	673777	rcr s9	
0241	663777	rcl s9	
0242	673777	rcr s9	
0243	663777	rcl s9	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 17 (continued)

Location	Contents	Mnemonic Code	Remarks
0244	530173	sas ' fastloops	Checks that contents of AC are correct.
0245	760400	hlt	
0246	320265	dio temp	Checks that contents of IO are correct.
0247	520265	sas temp	
0250	760400	hlt	
0251	460275	isp count	Leaves loop when it has been executed 2^8
0252	600232	jmp reverse	times. END OF rcl s9 - rcr s9 Test.
0253	440173	idx fastloops	Sets up the address of lio instruction so that the next test number is retrieved.
0254	520301	sas finish	Skips when all test numbers have been checked
0255	600173	jmp fastloops	in the loops.
0256	640020	szs 20	With SS2 on, program iterates. With SS2
0257	600003	jmp start+2	off, program jumps to RIM loader and reads
0260	607772	jmp 7772	in the next program.

Location	Contents	Mnemonic Code	Remarks
0261	661001	rall 661001	
0262	662001	rill 662001	
0263	671001	rarl 671001	
0264	672001	rirl 672001	
0265	000000	temp 000000	
0266	000000	flip 000000	Test numbers used in high-speed loops for
0267	777777	777777	combined rotation.
0270	000001	one 000001	
0271	777776	777776	
0272	525252	525252	
0273	525254	525254	
0274	777000	number 777000	
0275	000000	count 000000	
0276	000777	mask 000777	
0277	662000	lastleft 662000	
0300	672000	lastright 672000	
0301	220274	finish lio flip+6	

Location	Contents	Mnemonic Code		Remarks
0302	777070	test	777070	Test numbers used in one-bit rotation.
0303	776161		776161	
0304	774343		774343	
0305	770707		770707	
0306	761617		761617	
0307	743437		743437	
0310	707077		707077	
0311	616177		616177	
0312	434377		434377	
0313	070777		070777	
0314	777777		777777	
0315	777777	testc	777777	Test numbers used in nine-bit rotation.
0316	070070		070070	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 20

(Tests sal, sil, sar, sir, scl, scr)

Location	Contents	Mnemonic Code	Remarks
0000	000020	000020	Program number
0001	640010	start szs 10	With SS1 on, program halts after reading in.
0002	760400	hlt	
0003	200077	lac test	START OF sal and sil Test. Loads AC and IO
0004	220077	lio test	with test number 377777.
0005	665777	sal s9	Shifts AC left 17 bits.
0006	665377	sal s8	
0007	666777	sil s9	Shifts IO left 17 bits.
0010	666377	sil s8	
0011	640100	sza	Checks that 0's were shifted into AC through
0012	760400	hlt	bit 17 and that the sign bit was not changed.
0013	320106	dio temp	Checks that 0's were shifted into IO through
0014	520106	sas temp	bit 17 and that the sign bit was not changed.
0015	760400	hlt	END OF sal and sil Test.
0016	200077	lac test	START OF sar and sir Test. Loads AC and IO
0017	220077	lio test	with 377777.

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 20 (continued)

Location	Contents	Mnemonic Code	Remarks
0020	675777	sar s9	Shifts AC right 17 places.
0021	675377	sar s8	
0022	676777	sir s9	Shifts IO right 17 places
0023	676377	sir s8	
0024	640100	sza	Checks that sign bit shifts into bit 1 of the
0025	760400	hlt	AC.
0026	320106	dio temp	Checks that sign bit shifts into bit 1 of the
0027	520106	sas temp	IO.
0030	760400	hlt	
0031	200104	lac test+5	Loads AC and IO with 400000.
0032	220104	lio test+5	
0033	675777	sar s9	Shifts AC right 17 bits
0034	675377	sar s8	
0035	676777	sir s9	Shifts IO right 17 bits .
0036	676377	sir s8	
0037	761000	cma	Checks that the sign bit was shifted 17 times
0040	640100	sza	into bit 1 of AC.
0041	760400	hlt	

Location	Contents	Mnemonic Code	Remarks
0042	761000	cma	Checks that the sign bit was shifted 17 times
0043	320106	dio temp	into bit 1 of IO. END OF sar and sir Test.
0044	520106	sas temp	
0045	760400	hlt	
0046	200077	lac test	START OF scl Test. Loads AC and IO with
0047	220077	lio test	the number 377777.
0050	667001	scl s1	Shifts the combined registers left one bit.
0051	520100	sas test+1	Checks that IO_0 was shifted into AC_{17}.
0052	760400	hlt	
0053	320106	dio temp	Checks that IO_1 was shifted into IO_0. END
0054	200101	lac test+2	OF scl Test.
0055	520106	sas temp	
0056	760400	hlt	
0057	200077	lac test	START OF scr Test. Loads AC and IO with
0060	220077	lio test	the number 377777.
0061	677001	scr s1	Shifts combined registers right one bit.
0062	520102	sas test+3	Checks AC for correct contents.
0063	760400	hlt	

PROGRAM LISTING

MAINDEC 1 - INSTRUCTION TEST PROGRAM 20 (continued)

Location	Contents	Mnemonic Code		Remarks
0064	320106		dio temp	Checks that AC_{17} was shifted
0065	200103		lac test+4	into IO_0 and that IO_0 was shifted
0066	520106		sas temp	into IO_1. END OF scr Test.
0067	760400		hlt	
0070	640020		szs 20	With SS2 on, program iterates. With SS2 off,
0071	600003		jmp start+2	program enters sequence which signifies completion of instruction test.
0072	200105		lac last	Loads AC with the number 000777.
0073	761000		cma	
0074	220105		lio last	Loads IO with the number 777000.
0075	760017		stf 7	Sets all program flags.
0076	600000		jmp 0000	Loads MB with program number; sets MA equal to 0; sets PC equal to 1, and halts. END OF the Instruction Test.
0077	377777	test	377777	
0100	377776		377776	
0101	777776		777776	
0102	177777		177777	
0103	577777		577777	
0104	400000		400000	
0105	777000	last	777000	
0106	000000	temp	000000	