UNIVAC 1219 EMULATOR

with

UNIVAC 1532 I/O CONSOLE

and

1219 ASSEMBLER

by

Duane B. Craps

3 March 2013

TABLE OF CONTENTS

Works pretty much like a real machine. Click on registers , type in octal values into registers and click start.. Well, we were really punching in binary, but we were thhinking in octal

To manually store a program:

1. click on Function repeate button to light red

2. click on RUN button to change ledgend to "Op Step"

3. Enter first address to be stored into P register

4. put octal 44 (Store AL) in the F (Function) register

5. Put word to be stored in AL (Accumulator Lower) register

6. Click start , word will be stored , repeat steps 5& 6 until all words are in

To manually view memory contents:

1. click on Function repeate button to light red

2. click on RUN button to change ledgend to "Op Step"

3. Enter first address to be read into P register

4. put a 12 (Enter AL) in the F (Function) registerd

5. Click start , word will be viewed in AL, repeat this step fot each word

To load or store a memory Images

NOTE: Memory Image"0" is loaded at emulator start

To load a saved image:

Clicking on Load with 1 in "P" register will load a memory image (32K)as referenced by "AL" register bits (0-2)

To Save a memory image:

Clicking on Load with 2 in "P" register will save memory image (32K) as referenced by "AL" register bits (0-2)

At this time there is not much software. The "memory image 0" sent with the 1219 emulator contains the following:

1. A program residing at octal 1000 that prints out HELLO WORLD over and over until stooped by selecting stop key 0
2. A program residing at octal 2000 that inputs a characters from the keyboard and displays the octal code of the key pressed in register AL
3. Paper tape Bootstrap at octal 500 which unconditionally loads $147_{(8)}$ at address 540 and jumps to 540 this should be a loader program able to read address from tape and load an object program and verify the checksum .
4. Paper tape loader at address 540
5. A keyboard inspect and change program at octal 70000with the following commands:
    - I (inspect) type 6 digit address or shorter terminated by a space
    - N (next) inspects next address
    - L (last) goes back one address
    - C (change) inputs 6 digit octal number to be stored at last address inspected
    - T (terminal) inputs 6 digit terminal address
    - M (mask) inputs 6 digit mask
    - D (data) inputs 6 digit data word
    - D (data) inputs 6 digit data word
    - CTRL+D (dump) Dumps contents of memory from inital to terminal
    - CTRL+S (search) Searches form initial to terminal for data subject to mask (must match where bits are set in mask)

1532 INPUT-OUTPUT CONSOLE NOTES

1. Console defaults to channel 0; No need to select Channel unless program uses something else. Almost all 1532's were on channel 0, as it had the lowest I/O priority.

2. Punch feature not implemented at this time

3. Put file name in File window before starting the bootstrap or loader. Failing to do so will cause message “Mount File and select Reader”; Clear message put file name in file window and select reader. The assembler “.76” programs load with the loader at octal address 540.

4. I added a check box to inhibit carriage returns. Modern computer equipment use a linefeed character to both linefeed and character return.. Legacy programs, using both, would skip lines

1219 ASSEMBLER NOTES

The assembler program started many years as an assembler I wrote on a Radio Shack Color Computer for the AN/UYK-20 computer; which was the standard US Navy minicomputer at the time. Not having Disc I/O all all source code / object listing had to fit in memory. Source was entered as data statements in the program. Later when I bought my first P.C. (a Packard bell 80286 machine with 40 Mb HD) I converted it to GW basic. About the time all the 1219's went away ,I modified it to assemble 1219 instructions. Now it is running under Q basic 64. Maybe some day I will port it to Visual basic. The code is UGLY. To save memory, I used all one or two character variable names, a gazillion statements per line and no comments.

To run assembler start program and enter the name of the source program , in upper case, containing your program's source code in TRIM neumonics. Enter only name portion as assembler adds the .txt . The assembler writes the output files in the same directory as it is run from. The output *name .PRG* is the assembly listing. The name.76 is a bioctal tape image that can be loaded into the computer by putting it's name in the file window of the 1532 I/O console and starting the computer at $540_{(8)}$. The assembler was assembled with QBASIC and must be run in a DOS emulator on late model computers.(I use DosBox)

CODING CONVENTIONS:

- Labels must start in column 1 and be preceded by a >

>LABLE1 ENTAL LOK+2

- All numbers are assumed to be decimal values except:

&100	octal 100 (decimal 64)
&HABC	hexadecimal ABC (decimal 2748)

ASSEMBLER DIRECTIVES AND PSEUDO OPERATIONS:

- ORG Sets beginning address at which to assemble the following instructions. multiple ORG directives are permitted.
- END Terminates assembly.
- SADD Sets Program starting address. If present assemler will produce self starting tape.
- RES Skips a number of addresses.
- DATA Enters data into the current address.
- BCW Same as DATA but lets you use BCW to make your listing easier to read.
- AS (ASCII string) Packs the message that follow into memory two ASCII characters per address.“[“= RETURN ; “]” = LINEFEED . Bit 9 is set in last character
- AW Puts the text that follows into memory one ASCII character per memory address.“[“ = RETURN ; “]” = LINEFEED. Bit 9 is set in last character.
- EVEN Causes the next instruction (usually data) to be assembled an even address. The 1219 computer requires double length operands to be at even addresses as it forces bit 0 set to retreive the most siginifant half.
- EQU Assigns the value of its operand to its label.
- **;** The rest of line is taken as a remark.
- LOK Holds the value of it's address. Do not put space before offset .LOK+4 correct LOK +4 wrong

REPERTOIRE OF INSTRUCTIONS

ULTRA SYMBOL	CODE	TRIM SYMBOL	DESCRIPTION	TIME μS	ULTRA SYMBOL	CODE	TRIM SYMBOL	DESCRIPTION	TIME μS
CL	02	CMAL	Compare Y	4	JUN JLS*	66	JPAUNG	Jump Au Negative, Y	2
	03	CMALB	Compare Y+B	4	JLN	67	JPALNG	Jump AL Negative, Y	2
MSL	04	SLSU	Selective Substitute	4	LLK	70	ENTALK	Enter AL, Y	2
	05	SLSUB	Selective Substitute Y+B	4	ALK	71	ADDALK	Add U, 12 bits	2
CLM	06	CMSK	Masked Compare Y	4	SIR	72	STRICR	Store ICR, Y	4
	07	CMSKB	Masked Compare Y+B	4	JBNZ	73	BJP	Decrement B, Jump, Y	2
LU	10	ENTAU	Enter AU, Y	4	SAD	74	STRADR	Store Address, Y	4
	11	ENTAUB	Enter AU, Y+B	4	SSR	75	STRSR	Store SR, Deactivate SR, Y	4
LL	12	ENTAL	Enter AL, Y	4	SLJ	76	RJP	Return Jump, Y	4
	13	ENTALB	Enter AL, Y+B	4	SIC	5001	SIN	Set Input Active	2
AL	14	ADDAL	Add Y, 18 bit	4	SOC	5002	SOUT	Set Output Active	2
	15	ADDALB	Add Y+B, 18 bit	4	SFC	5003	SEXF	Set External Function Active	2
ANL	16	SUBAL	Subtract Y, 18 bit	4	LIC	5011	IN	Initiate Input Buff, k	6
	17	SUBALB	Subtract Y+B, 18 bit	4	LOC	5012	OUT	Initiate Output Buff, k	6
AA	20	ADDA	Add Y, 36 bit	6	LFC	5013	EXF	External Function	6
	21	ADDAB	Add Y+B, 36 bit	6	ACI	5014	RTC	Enable Real-Time Clock	2
ANA	22	SUBA	Subtract Y, 36 bit	6	STIC	5015	INSTP	Terminate Input, k	2
	23	SUBAB	Subtract Y+B, 36 bit	6	STOC	5016	OUTSTP	Terminate Output, k	2
M	24	MULAL	Multiply Y	14	STEF	5017	EXFSTP	Terminate External Function, k	2
	25	MULALB	Multiply Y+B	14	SRD	5020	SRSM	Set Resume ff (Intercomp)	2
D	26	DIVA	Divide, Y	14	TIC	5021	SKPIIN	Skip Input Inact, k	2
	27	DIVAB	Divide, Y+B	14	TOC	5022	SKPOIN	Skip Output Inact, k	2
SLJI	30	IRJP	Indirect RJP, Y	6	TFC	5023	SKPFIN	Skip on Ext. Fnct. Inact.	2
	31	IRJPB	Indirect RJP, Y+B	6	WFI	5024	WTFI	Wait for Interrupt	2
LB	32	ENTB	Enters B, Y	4	OV	5026	OUTOV	Force Output One Word, k	2
	33	ENTBB	Enter B, Y+B	4	EFOV	5027	EXFOV	Force Ext Function One Word, k	2
J	34	JP	Jump, Y	2	AAI	5030	RIL	Remove Interrupt Lockout	2
	35	JPB	Jump, Y+B	2	AFI	5032	EXL	Remove Ext Interrupt Lockout	2
LBK	36	ENTBK	Enter, B, U	2	PAI	5034	SIL	Set Interrupt Lockout	2
	37	ENTBKB	Modify B, U	2	PFI	5036	SXL	Set Ext Interrupt Lockout	2
CY	40	CL	Store Zero, Y	4	SRU	5041	RSHAU	Right Shift AU, k	4-10
	41	CLB	Store Zero, Y+B	4	SRL	5042	RSHAL	Right Shift AL, k	4-10
SB	42	STRB	Store, B, Y	4	SRA	5043	RSHA	Right Shift A, k	4-20
	43	STRBB	Store B, Y+B	4	SCA	5044	SF	Scale A Left, k, SF	4-20
SL	44	STRAL	Store AL, Y	4	SLU	5045	LSHAU	Left Shift AU, k	4-10
	45	STRALB	Store AL, Y+B	4	SLL	5046	LSHAL	Left Shift AL, k	4-10
SU	46	STRAU	Store AU, Y	4	SLA	5047	LSHA	Left Shift A, k	4-20
	47	STRAUB	Store AU, Y+B	4	TK	5050	SKP	Skip Console Key, k	2
OR	51	SLSET	Selective Set (IOR), Y	4	TNB	5051	SKPNBO	Skip No Borrow	2
AND	52	SLCL	Selective Clear (AND), Y	4	TOF	5052	SKPOV	Skip Overflow	2
XOR	53	SLCP	Selective Complement (XOR), Y	4	TNO	5053	SKPNOV	Skip No Overflow	2
EJI	54	IJPEI	Indirect Jump (RIL), Y	4	TOP	5054	SKPODD	Skip L(AU, AL) Odd Parity	2
JI	55	IJP	Indirect Jump, Y	4	TEP	5055	SKPEVN	Skip L(AU, AL) Even Parity	2
TB	56	BSK	Increment B, Skip, Y	4	SK	5056	STOP	Stop Console Key, k	2
TZ	57	ISK	Decrement Index, Skip, Y	6	TRD	5057	SKPNR	Skip No Resume (Intercomp)	2
JE* JUZ	60	JPAUZ	Jump AU Zero, Y	2	RND	5060	RND	Round AU	2
JLZ	61	JPALZ	Jump AL Zero, Y	2	CPL	5061	CPAL	Complement AL	2
JNE* JUNZ	62	JPAUNZ	Jump AU Not Zero, Y	2	CPU	5062	CPAU	Complement AU	2
JLNZ	63	JPALNZ	Jump AL Not Zero, Y	2	CPA	5063	CPA	Complement A	2
JNLS* JUP	64	JPAUP	Jump AU Positive, Y	2	LIR	5072	ENTICR	Enter ICR, k	2
JLP	65	JPALP	Jump AL Positive, Y	2	LSR	5073	ENTSR	Enter, SR, k	2

*If compare designator is set.

ASSIGNED MEMORY ADDRESS

Address	Assignment
00000	Fault Interrupt Entrance Register
00001-00010	8 Index Registers
00011	Intercomputer Time-Out Interrupt Register
00012	Real-Time Clock Interrupt Register
00013	Clock Overflow Interrupt Register
00014	Real-Time Clock Monitor Word Register
00015	Real-Time Clock Incrementing Register
00016	Synchronizing Interrupt Register
00017	Scale Factor Shift Count
00020-00037	Continuous Data Mode or External Function Buffer Control Registers (Channels 0-7)
00040-00057	Output Buffer Control Registers (Channels 0-7)
00060-00077	Input Buffer Control Registers (Channels 0-7)
00100-00117	External Interrupt Registers (Channels 0-7)
00120-00137	Unassigned
00140-00157	Output Monitor Interrupt Registers (Channels 0-7)
00160-00177	Input Monitor Interrupt Registers (Channels 0-7)
FOR COMPUTERS EQUIPPED WITH 16 I/O CHANNELS:	
00200-00217	Unassigned
00220-00237	Continuous Data Mode or External Function Buffer Control Registers (Channels 8-15)
00240-00257	Output Buffer Control Registers (Channels 8-15)
00260-00277	Input Buffer Control Registers (Channels 8-15)
00300-00317	External Interrupt Registers (Channels 8-15)
00320-00337	Unassigned
00340-00357	Output Monitor Registers (Channels 8-15)
00360-00377	Input Monitor Registers (Channels 8-15)
00400-00477	Unassigned
00500-00537	Initial Input Routine (Bootstrap)
00600-00677	Unassigned
00540-177777	Unassigned

SPERRY UNIVAC

UNIVAC 1218 COMPUTER

Repertoire of Instructions

	f m	SYMBOL	INSTRUCTION	DESCRIPTION	EXEC TIME SKIP	EXEC TIME NO SKIP
LOGICAL	#02*	CMAL	COMPARE SET DESIGNATOR	$(AL):(Y)$; $(AL)_f = (AL)_i$	8	
	#06*	CMSK	COMPARE WITH MASK SET DES.	$L(AL)(AU):L(Y)(AU)$; $(A)_f=(A)_i$	8	
	#04*	SLSU	SELECTIVE SUBSTITUTE	$L(AU)'(AL)+L(AU)(Y)\rightarrow AL$; $(AU)_f=(AU)_i$	8	
	51	SLSET	SELECTIVE SET(INCLUSIVE OR)	$(AL) \vee (Y)\rightarrow AL$; SET $(AL)_n$ FOR $(Y)_n = 1$	8	
	52	SLCL	SELECTIVE CLEAR (LOGICAL PROD)	$L(AL)(Y)\rightarrow AL$; CLEAR $(AL)_n$ FOR $(Y)_n = 0$	8	
	53	SLCP	SELECTIVE COMPLEMENT(EXCL.OR)	$(AL) \oplus (Y)\rightarrow AL$; COMPLEMENT $(AL)_n$ FOR $(Y)_n=1$	8	
	50 61	CPAL	COMPLEMENT AL	$(AL)'\rightarrow AL$ IGNORE k	5.33	
	50 62	CPAU	COMPLEMENT AU	$(AU)'\rightarrow AU$ IGNORE k	5.33	
	50 63	CPA	COMPLEMENT A	$(A)'\rightarrow A$ IGNORE k	5.33	
SHIFT	50 41	RSHAU	RIGHT SHIFT AU	SHIFT (REG.) RIGHT k BIT POSITIONS END OFF & FILL THE UPPER k BITS WITH INITIAL SIGN	$4+2k/3$ μsec (k≠0); 5.33 (k=0)	
	50 42	RSHAL	RIGHT SHIFT AL			
	50 43	RSHA	RIGHT SHIFT A			
	50 44	SF	SCALE FACTOR SHIFT	LEFT CIRCULAR SHIFT A UNTIL $A_{35} \neq A_{34}$ OR k — SHIFTCOUNT = 0 THEN k — SHIFTCOUNT → 00017	$9.33+2k/3$ (k≠0); 8 (k=0)	
	50 45	LSHAU	LEFT SHIFT AU	SHIFT(REG.)LEFT k BIT POSITIONS CIRCULARLY	4 μsec (k=0); $+2k/3$ μsec (k≠0)	5.33
	50 46	LSHAL	LEFT SHIFT AL			
	50 47	LSHA	LEFT SHIFT A			
SKIP ARITHMETIC	50 51	SKPNBO	SKIP ON NO BORROW	SKIP NI IF BORROW DESIGNATOR NOT SET	6	4.67
	50 52	SKPOV	SKIP ON OVERFLOW	SKIP NI IF OVERFLOW DESIGNATOR SET	6	4.67
	50 53	SKPNOV	SKIP ON NO OVERFLOW	SKIP NI IF OVERFLOW DESIGNATOR NOT SET	6	4.67
	50 54	SKPODD	SKIP ON ODD PARITY	SKIP NI IF SUM OF ONES IN L(AU)(AL)IS ODD	6	4.67
	50 55	SKPEVN	SKIP ON EVEN PARITY	SKIP NI IF SUM OF ONES IN L(AU)(AL)IS EVEN	6	4.67
SKIP I/O	50 21	SKPIIN	SKIP ON INPUT INACTIVE	SKIP NI IF CHAN. k INPUT IS INACTIVE	6	4.67
	50 22	SKPOIN	SKIP ON OUTPUT INACTIVE	SKIP NI IF CHAN. k OUTPUT IS INACTIVE	6	4.67
	50 23	SKPFIN	SKIP ON EXT. FUNCT. INACTIVE	SKIP NI IF CHAN. k EXT. FUNCT. IS INACTIVE	6	4.67
	50 50	SKP	SKIP ON KEY SETTING k	SKIP NI IF k = CONSOLE KEY SETTING	6	4.67
	50 57	SKPNR	SKIP ON NO RESUME	SKIP NI IF RESUME DESIGNATOR NOT SET	6	4.67
INPUT / OUTPUT	50 11	IN	INPUT TRANSFER	$(P+1)\rightarrow 60+2k$; $(P+2)\rightarrow 61+2k$; SET IN. ACTIVE	20	
	50 12	OUT	OUTPUT TRANSFER	$(P+1)\rightarrow 40+2k$; $(P+2)\rightarrow 41+2k$; SET OUT. ACTIVE	20	
	50 13	EXF	EXT. FUNCT. TRANSFER	$(P+1)\rightarrow 20+2k$; $(P+2)\rightarrow 21+2k$; SET EXT. F. ACTIVE	20	
	50 15	INSTP	TERMINATE INPUT	CLEAR INPUT ACTIVE CHAN. k	4	
	50 16	OUTSTP	TERMINATE OUTPUT	CLEAR OUTPUT ACTIVE CHAN. k	4	
	50 17	EXFSTP	TERMINATE EXT. FUNCT.	CLEAR FUNCTION ACTIVE CHAN. k	4	
	50 20	SRSM	SET RESUME	SET THE RESUME DESIGNATOR (INTERCOMPUTER)	4	
	50 26	OUTOV	OUTPUT OVERRIDE	FORCE ONE WORD OUT CHAN.k WITH OUTPUT ACK.	4.67	
	50 27	EXFOV	FUNCTION OVERRIDE	FORCE ONE WORD OUT CHAN.k WITH EXT.FUNCT.	4.67	
	50 30	RIL	REMOVE INTERRUPT LOCKOUT	ENABLE ALL INTERRUPTS	4	
	50 32	RXL	REMOVE EXT. INT. LOCKOUT	ENABLE EXTERNAL INTERRUPTS	4	
	50 34	SIL	SET INTERRUPT LOCKOUT	DISABLE ALL INTERRUPTS	4	
	50 36	SXL	SET EXT. INT. LOCKOUT	DISABLE EXTERNAL INTERRUPTS	4	
MISC.	50 24	WTFI	WAIT FOR INTERRUPT	STOP;THEN INTERRUPT ENTRANCE REG. FOR NI	4	
	50 56	STOP	STOP ON KEY SETTING	STOP IF k = KEY SETTING	4.67	
FAULT	00, 01, 77 FAULT = 4 usec 50 00, 50 01, 50 77 FAULT = 4 usec					

ASSIGNED CORE MEMORY LOCATIONS (OCTAL)

000	FAULT ENTRANCE REGISTER	060-077	INPUT BUFFER CONTROL
001-010	INDEX REGISTERS	100-117	EXTERNAL INTERRUPT ENTRANCE
016	SYNCHRONIZATION INTERRUPT ENTRANCE	120-137	EXT. FUNCT. TERMINATION INTERRUPT
017	SCALE FACTOR SHIFT COUNT	140-157	OUTPUT TERMINATION INTERRUPT
020-037	EXTERNAL FUNCTION BUFFER CONTROL	160-177	INPUT TERMINATION INTERRUPT
040-057	OUTPUT BUFFER CONTROL	200-237	BOOTSTRAP PROGRAM

UNIVAC 1218 COMPUTER

Repertoire of Instructions

INSTRUCTION WORD FORMAT I | 17-f-12 | 11——u——0 | FORMAT II | 17-f-12 | 11-m-6 | 5—k—0 |

	f m	SYMBOL	INSTRUCTION	DESCRIPTION	EXEC TIME
ENTER	#10*	ENTAU	ENTER AU WITH (Y)	(Y) → AU	8
	#12*	ENTAL	ENTER AL WITH (Y)	(Y) → AL	8
	#32*	ENTB	ENTER B REGISTER WITH (Y)	(Y) → B REGISTER	12
	36*	ENTBK	ENTER B REGISTER WITH CONSTANT	Y → B REGISTER (Note 1)	8
	70	ENTALK	ENTER AL WITH CONSTANT	Y → AL (Note 1)	4.67
	50 72	ENTICR	ENTER INDEX CONTROL REGISTER	k_{2-0} → ICR	4
	50 73	ENTSR	ENTER SR	k_{3-0} → SR	4
STORE	#40*	CL	STORE ZERO (CLEAR Y)	0 → Y	8
	#42*	STRB	STORE B IN Y	B → Y	12
	#44*	STRAL	STORE (AL) IN Y	(AL) → Y	8
	#46*	STRAU	STORE (AU) IN Y	(AU) → Y	8
	72	STRICR	STORE (ICR) IN Y_L	(ICR) → Y_{3-0}; $(Y)_{17-6}$ UNCHANGED	8
	74	STRADR	STORE ADDRESS IN Y_L	$(AL)_{11-0}$ → Y_{11-0}; $(Y)_{17-12}$ UNCHANGED	8
	75	STRSR	STORE (SR) IN Y_L	(SR) → Y_{3-0}; $(Y)_{17-6}$ UNCHGD. 0 → SR_3	8
ARITHMETIC	#14*	ADDAL	ADD (Y) TO AL	(AL) + (Y) → AL	8
	#16*	SUBAL	SUBTRACT (Y) FROM AL	(AL) - (Y) → AL	8
	#20*	ADDA	ADD (Y+1, Y) TO A	(A)+(Y+1,Y) → A	12
	#22*	SUBA	SUBTRACT (Y+1,Y) FROM A	(A)-(Y+1,Y) → A	12
	#24*	MULAL	MULTIPLY AL BY (Y)	(AL).(Y) → A	26-48.67
	#26*	DIVA	DIVIDE A BY (Y)	(A)÷(Y) → AL; Remainder → AU	48
	71	ADDALK	ADD CONSTANT TO AL	(AL)+Y → AL, Note 1	4.67
	50 60	RND	ROUND AU	{+(AU)+$\overline{AL}_{17}$ → AL); OR / -(AU)-$\overline{AL}_{17}$ → AL); } IGNORE k, $AU_f = AU_i$	5.33 / 5.33
MODIFYING	37	ENTBKB	MODIFY B WITH CONSTANT	B+Y → B REGISTER	12
	56	BSK	B SKIP	{IF B = (Y), SKIP NI / IF B ≠ (Y), B+1 → B REGISTER & DO NI}	16
	57	ISK	INDEX SKIP	{IF (Y)=0, SKIP NI / IF (Y) ≠ 0, (Y)-1 → Y & DO NI}	12
	73	BJP	B JUMP	{IF B≠0, B-1 → B REGISTER AND Y → P / IF B=0, DO NI}	12
JUMP UNCOND.	30*	IRJP	INDIRECT RETURN JUMP	(P)+1 → (Y); (Y)+1 → P (Note 2)	12
	34*	JP	UNCONDITIONAL JUMP	Y → P	4
	54	IJPEI	IND JUMP & ENABLE INT.	(Y) → P & ENABLE INTERRUPTS	8
	55	IJP	INDIRECT JUMP	(Y) → P	8
	76	RJP	DIRECT RETURN JUMP	(P)+1 → Y; Y+1 → P (Note 2)	8

JUMP TO Y (Y → P) IF COMPARISON DESIGNATOR IS:

	NOT SET AND	SET AND		EXEC TIME
60	JPAUZ (AU) = 0			4
61	JPALZ (AL) = 0	JPEQ M = (AL)		4
62	JPAUNZ (AU) ≠ 0		M=(Y),(AL)=(AL)	4
63	JPALNZ (AL) ≠ 0	JPNOT M ≠ (AL)	for f = 02,03	4
64	JPAUP (AU) POSITIVE		M=L(Y)(AU),(AL)=	4
65	JPALP (AL) POSITIVE	JPMLEQ M ≤ (AL)	L(AU)(AL)	4
66	JPAUNG (AU) NEGATIVE		for f = 06,07.	4
67	JPALNG (AL) NEGATIVE	JPMGR M > (AL)		4

LEGEND

Y = y ⊕ y+B

y = $u_p \oplus u_{SR} \oplus u$

B = CONTENTS OF INDEX REG.

NI = NEXT INST.

\# SR SENSITIVE

* B MODIFICATION of "y" REQUESTED: ADD SUFFIX "B" TO SYMBOL and "1" to f-CODE; ADD 4 usec to EXECUTION TIME

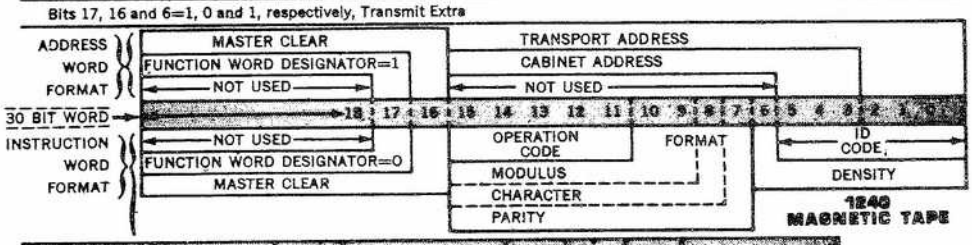
NOTES: 1. For f = 36,37,70,71 y=u extended with sign to 18 bits.
2. RETURN JUMPS, executed from Interrupt Entrance Registers, Store (P)

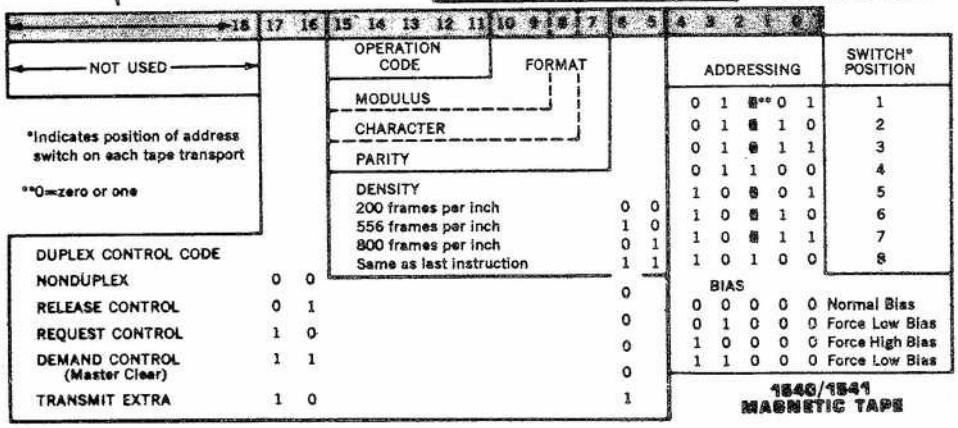
Code (Octal) f	m	TRIM SYMBOL	INSTRUCTION	DESCRIPTION	TIME μsec.
50	11	IN	Input transfer	((P) · 1)→60 · 2k;((P) · 2)→61 · 2k; set input active	20
50	12	OUT	Output transfer	((P) · 1)→40 · 2k;((P) · 2)→41 · 2k; set output active	20
50	13	EXF	Ext. funct. transfer	((P) · 1)→20 · 2k;((P) · 2)→21 · 2k; set ext. func. active	20
50	15	INSTP	Terminate input	Clear input active, channel k	4
50	16	OUTSTP	Terminate output	Clear output active, channel k	4
50	17	EXFSTP	Terminate ext. funct.	Clear function active, channel k	4
50	20	SRSM	Set resume	Set the resume designator (intercomputer)	4
50	21	SKPIIN	Skip on input inactive	Skip NI if chan. k input is inactive	
50	22	SKPOIN	Skip on output inactive	Skip NI if chan. k output is inactive	
50	23	SKPEIN	Skip on ext. funct. inactive	Skip NI if chan. k ext. funct. is inactive	4.67 6
50	24	WTFI	Wait for interrupt	Stop; then interrupt entrance reg. for NI	4
50	26	OUTOV	Output override	Force one word out channel k with output ack.	4.67
50	27	EXFOV	External function override	Force one word out channel k with ext. funct.	4.67
50	30	RIL	Remove interrupt lockout	Enable all interrupts	4
50	32	RXL	Remove ext. int. lockout	Enable external interrupts	4
50	34	SIL	Set interrupt lockout	Disable all interrupts	4
50	36	SXL	Set ext. int. lockout	Disable external interrupts	4
50	41	RSHAU	Right shift (AU)	Shift (AU) right by k; sign fill	5.33
50	42	RSHAL	Right shift (AL)	Shift (AL) right by k; sign fill	plus
50	43	RSHA	Right shift (A)	Shift (A) right by k; sign fill	2k 3
50	44	SF	Scale factor shift	Shift (A) left, with sign fill, until $\cdot(A)_{35} \neq (A)_{34}$ or |k-shiftcount|=0; |k-shiftcount|→00017	9.33 plus 2k 3
50	45	LSHAU	Left shift (AU)	Cyclic shift (AU) left by k	5.33
50	46	LSHAL	Left shift (AL)	Cyclic shift (AL) left by k	plus
50	47	LSHA	Left shift (A)	Cyclic shift (A) left by k	2k 3
50	50	SKP	Skip on key setting	Skip NI if k=console key setting	4.67 6
50	51	SKPNBO	Skip on no borrow	Skip NI if borrow designator not set	
50	52	SKPOV	Skip on overflow	Skip NI if overflow designator set	
50	53	SKPNOV	Skip on no overflow	Skip NI if overflow designator not set	4.67 6
50	54	SKPODD	Skip on odd parity	Skip NI if sum of ones in L(AU)(AL) is odd	
50	55	SKPEVN	Skip on even parity	Skip NI if sum of ones in L(AU)(AL) is even	
50	56	STOP	Stop on key setting	Stop if k Console key setting	4.67
50	57	SKPNR	Skip on no resume	Skip NI if resume designator is not set	4.67 6
50	60	RND	Round AU	If (A)≥0, (AU) · $(AL)_{17}$→AL, (AU), (AU), If (A)<0, (AU)-Compl $(AL)_{17}$→AL, (AU), (AU),	5.33
50	61	CPAL	Complement (AL)	(AL)'→AL	5.33
50	62	CPAU	Complement (AU)	(AU)'→AU	5.33
50	63	CPA	Complement (A)	(A)'→A	5.33
50	72	ENTICR	Enter Index Control Reg.	k_{2-0}→ICR	4
50	73	ENTSR	Enter Special Register	k_{3-0}→SR	4

OPERATION CODES

CODE	OPERATION 1540/1541	OPERATION 1240
00000	Read (Read Forward)	Read
00001	Read; Selective (Selective Read-Forward)	Read; Selective
00010	Read; Modified Stop	Read; Ignore Error Halt
00011	Space File	Space File
00100	Search Type I	Search Type I
00101	Search Type II	Search Type II
00110	Search File Type I	Search File Type I
00111	Search File Type II	Search File Type II
01000	Write	Write
01001	Write XIRG	Write XIRG
01010	Write Ignore Error Halt	Write Ignore Error Halt
01011	Write XIRG, Ignore Error Halt	Write XIRG, Ignore Error Halt
01100	Write Modified Stop	Write Tape Mark
01101	Write Edit	Write Tape Mark, XIRG
01110	Write Tape Mark	
01111	Write Tape Mark, XIRG	
10000	Backread (Read Backward)	Backspace
10001	Backread Selective (Selective Read-Backward)	
10010	Backread Modified Stop	Backspace Read
10011	Backspace File	Backspace File
10100	Backsearch Type I	Backsearch Type I
10101	Backsearch Type II	Backsearch Type II
10110	Backsearch File Type I	Backsearch File Type I
10111	Backsearch File Type II	Backsearch File Type II
11000	Rewind	Rewind
11001	Rewind, Clear Write Enable	Rewind, Clear Write Enable
11010	Rewind	Rewind
11011	Rewind, Clear Write Enable	Rewind, Clear Write Enable
11100	Rewind-Read	Rewind-Read
11101	Rewind-Read, Clear Write Enable	Rewind-Read, Clear Write Enable
11110	Rewind-Read	Rewind-Read
11111	Request Transport-Status	

Bits 17, 16 and 6=1, 0 and 1, respectively, Transmit Extra





UNIVAC 1218 COMPUTER
REPERTOIRE OF INSTRUCTIONS

Code (Octal) f	m	TRIM SYMBOL	INSTRUCTION		DESCRIPTION	TIME μsec.
57		ISK	Index skip		If (Y)=0, skip NI If (Y)≠0, (Y)−1→Y, read NI	12
			Jump to Y, i.e., Y→P, if comparison designator is (1) or (2) as follows:			
			(1) not set, and	SYMBOL	(2) set (See Note 3), and	
60		JPAUZ	(AU)=0			
61		JPALZ	(AL)=0	JPEQ	M = (AL) or L(AU)(AL)	M = (Y) of last compare instruction
62		JPAUNZ	(AU)≠0			
63		JPALNZ	(AL)≠0	JPNOT	M ≠ (AL) or L(AU)(AL)	M : (AL) if last compare instruction was an 02 or an 03. 4
64		JPAUP	(AU) positive			
65		JPALP	(AL) positive	JPMLEQ	M < (AL) or L(AU)(AL)	M : L(AU)(AL) if last compare instruction was an 06 or an 07.
66		JPAUNG	(AU) negative			
67		JPALNG	(AL) negative	JPMGR	M > (AL) or L(AU)(AL)	
70		ENTALK	Enter AL with constant		Y→AL (See Note 1)	4.67
71		ADDALK	Add constant to (AL)		(AL)+Y→AL, (See Note 1)	4.67
72		STRICR	Store (ICR) in Y_L		(ICR)→Y_{2-0}; Y_{17-6} unchanged; if (ICR)≠0, 000→Y_{5-3}; if (ICR)=0, 001→Y_{5-3}	8
73		BJP	B jump		If (B)≠0, (B)−1→B, Y→P If (B)=0, read NI	12
74		STRADR	Store address in Y_L		$(AL)_{11-0}$→Y_{11-0}; $(Y)_{17-12}$ unchanged	8
75		STRSR	Store (SR) in Y_L		(SR)→Y_{3-0}; 0→Y_{5-4}; $(Y)_{17-6}$ unchanged; 0→SR_3	8
76		RJP	Return jump		(P)+1→Y; Y+1→P (See Note 2)	8

NOTES: 1. For f=36, 37, 70, or 71, y=u extended with sign to 18 bits.

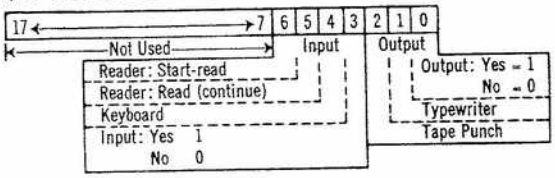
2. Store (P) rather than (P)+1 if return jump is executed from Interrupt Entrance Registers.

3. Comparison designator is set by a "compare" instruction, is unaffected by a "conditional jump" instruction, and is cleared by any other instruction.

SPECIAL ADDRESSES

000	Fault entrance register	060-077	Input buffer control
001-0010	Index registers	100-117	External interrupt entrance
016	Synchronization interrupt entrance	120-137	Ext. funct. termination interrupt
017	Scale factor shift count	140-157	Output termination interrupt
020-037	External function buffer control	160-177	Input termination interrupt
040-057	Output buffer control	200-237	Bootstrap program

I/O CONSOLE FUNCTION WORD FORMAT



I/O CODES

OCTAL CODE	1232 I/O CONSOLE FIELD DATA	XS-3 CHARACTER	FLEXOWRITER UPPER CASE	FLEXOWRITER LOWER CASE	TELETYPE LETTERS	TELETYPE FIGURES	BCD TAPE CHARACTER	BCD CHARACTER
00	Master space	Space	Tape Feed				Invalid	0
01	Shift up	⌋	T	t	T	5	1	1
02	Shift down	·-	Color Shift		Carriage Ret.		2	2
03	Line feed	0	0	o	0	9	3	3
04	Car. Return	1	Space		Space		4	4
05	Space	2	H	h	H		5	5
06	A	3	N	n	N	,	6	6
07	B	4	M	m	M	.	7	7
10	C	5			Line feed		8	8
11	D	6	L	l	L	)	9	9
12	E	7	R	r	R	4	0	Invalid
13	F	8	G	g	G	&	=	=
14	G	9	I	i	I	8	'	'
15	H	\	P	p	P	0	Invalid	Invalid
16	I	;	C	c	C	:	Invalid	Invalid
17	J	⌊	V	v	V	;	Invalid	Invalid
20	K	&	E	e	E	3	Blank	+
21	L	:	Z	z	Z	"	/	A
22	M	. (PERIOD)	D	d	D	$	S	B
23	N	P	B	b	B	P	T	C
24	O	A	S	s	S	Δ	U	D
25	P	B	Y	y	Y	6	V	E
26	Q	C	F	f	F	!	W	F
27	R	D	X	x	X	/	X	G
30	S	E	A	a	A	-	Y	H
31	T	F	W	w	W	2	Z	I
32	U	G	J	j	J	'	R.M.	+0
33	V	H	9	9	Figures		, (COMMA)	.
34	W	I	U	u	U	7	(	)
35	X	#	Q	q	Q	1	Invalid	Invalid
36	Y	<	K	k	K	(	Invalid	Invalid
37	Z	=	0	0	Letters		Invalid	Invalid
40	)	'					-	-
41	-	*					J	J
42	+	$	)	. (PERIOD)			K	K
43	<	!	Stop				L	L
44	=	J	*	=			M	M
45	>	K	Carriage Ret				N	N
46	—	L	(	, (COMMA)			O	O
47	$	M	Shift up				P	P
50	*	N	—	!			Q	Q
51	(	O	Tab				R	R
52	"	P	1	1			-0	-0
53	:	Q					$	$
54	P	R	/	+			Invalid	Invalid
55	!	(					Invalid	Invalid
56	, (COMMA)	@	-	- (MINUS)			Invalid	Invalid
57	Ⓢ	Δ	Shift down				Invalid	Invalid
60	0	≠	8	8			+	Blank
61	1	%	Back space				A	/
62	2	, (COMMA)	5	5			B	S
63	3	+					C	T
64	4	/	4	4			D	U
65	5	S					E	V
66	6	T	6	6			F	W
67	7	U					G	X
70	8	V	3	3			H	Y
71	9	W					I	Z
72	'	X	7	7			+0	R.M.
73	;	Y					.	, (COMMA)
74	/	Z	2	2			)	(
75	.	⌘					Invalid	Invalid
76	□	>					Invalid	Invalid
77	↑	)	Code delete				Invalid	Invalid

American Standard Code For Information Interchange (ASCII)																
DEC	HEX	OCT	CHAR	FUNCTION	DEC	HEX	OCT	CHAR	DEC	HEX	OCT	CHAR	DEC	HEX	OCT	CHAR
0	0	0	NUL	null	32	20	040	Space	64	40	100	@	96	60	140	`
1	1	1	SOH	start of heading	33	21	041	!	65	41	101	A	97	61	141	a
2	2	2	STX	start of text	34	22	042	"	66	42	102	B	98	62	142	b
3	3	3	ETX	end of text	35	23	043	#	67	43	103	C	99	63	143	c
4	4	4	EOT	end of transmission	36	24	044	$	68	44	104	D	100	64	144	d
5	5	5	ENQ	enquiry	37	25	045	%	69	45	105	E	101	65	145	e
6	6	6	ACK	acknowledge	38	26	046	&	70	46	106	F	102	66	146	f
7	7	7	BEL	bell	39	27	047	'	71	47	107	G	103	67	147	g
8	8	10	BS	backspace	40	28	050	(	72	48	110	H	104	68	150	h
9	9	11	TAB	horizontal tab	41	29	051	)	73	49	111	I	105	69	151	i
10	A	12	LF	line feed	42	2A	052	*	74	4A	112	J	106	6A	152	j
11	B	13	VT	vertical tab	43	2B	053	+	75	4B	113	K	107	6B	153	k
12	C	14	FF	form feed	44	2C	054	,	76	4C	114	L	108	6C	154	l
13	D	15	CR	carriage return	45	2D	055	-	77	4D	115	M	109	6D	155	m
14	E	16	SO	shift out	46	2E	056	.	78	4E	116	N	110	6E	156	n
15	F	17	SI	shift in	47	2F	057	/	79	4F	117	O	111	6F	157	o
16	10	20	DLE	data link escape	48	30	060	0	80	50	120	P	112	70	160	p
17	11	21	DC1	direct control 1	49	31	061	1	81	51	121	Q	113	71	161	q
18	12	22	DC2	direct control 2	50	32	062	2	82	52	122	R	114	72	162	r
19	13	23	DC3	direct control 3	51	33	063	3	83	53	123	S	115	73	163	s
20	14	24	DC4	direct control 4	52	34	064	4	84	54	124	T	116	74	164	t
21	15	25	NAK	negative acknowledge	53	35	065	5	85	55	125	U	117	75	165	u
22	16	26	SYN	synchronous idle	54	36	066	6	86	56	126	V	118	76	166	v
23	17	27	ETB	end of trans block	55	37	067	7	87	57	127	W	119	77	167	w
24	18	30	CAN	cancel	56	38	070	8	88	58	130	X	120	78	170	x
25	19	31	EM	end of medium	57	39	071	9	89	59	131	Y	121	79	171	y
26	1A	32	SUB	substitute	58	3A	072	:	90	5A	132	Z	122	7A	172	z
27	1B	33	ESC	escape	59	3B	073	;	91	5B	133	[	123	7B	173	{
28	1C	34	FS	file separator	60	3C	074	<	92	5C	134	\	124	7C	174	|
29	1D	35	GS	group separator	61	3D	075	=	93	5D	135	]	125	7D	175	}
30	1E	36	RS	record separator	62	3E	076	>	94	5E	136	^	126	7E	176	~
31	1F	37	US	unit separator	63	3F	077	?	95	5F	137	_	127	7F	177	DEL

```
;SAMPLE BOOTSTRAP FOR UNIVAC 1219 COMPUTER
;COMPATABLE WITH TARTAR LOADER
;PROGRAM UNCONDITIONALY LOADS 147 OCTAL WORDS (LOADER)
;ON BIOCTAL PAPER TAPE STARTING AT OCTAL ADDRESS 436
;THEN JUMPS TO OCTAL 540 (FIRST 2 WORDS ARE HEADER)
;TAPE FORMAT: 76=> BIOCTAL TAPE, IIIII =>INITIAL, TTTTT=> TERMINAL
;76 (BIOCTAL TAPE ID CODE)
;II (MSD INITIAL ADDRESS)
;II
;IT
;TT
;TT (LSD OF FINAL ADDRESS)
;PP (FIRST 6 BITS OF FIRST INSTRUCTION)
;PP
;PP (LAST 6 BITS OF FIRST INSTRUCTION)
;** REMEMBER BOOTSTRAP IGNORES HEADER DATA**
;
;
ORG &O500 ; ADDRESS TO START ASSEMBLY
>TARBOOT SIL 0
ENTSR &O10
ENTICR 1
EXF 00  ; TURN ON READER
BCW READ
BCW READ
ENTBK 0 ;SET FRAME COUNT
>NXTWD STRB 3
ENTALK 2
STRAL 4
ENTALK 0
>INPUT IN 0 ;GET FRAME
BCW 2
BCW 2
SKPIIN 0 ; WAIT FOR DATA
JP LOK-1
LSHAL 6  ; SHIFT UP 6 BITS
SLSET 2
JPALNZ LOK+3
ENTAU 3
JPAUZ INPUT
ISK 4  ;THREE FRAMES?
JP INPUT ; NO GET MORE
STRALB &O536 ;YES STORE WORD
BSK WDCT ;DONE YET?
JP NXTWD ;NO GET NEXT
JP &O540  ;yES JP TO LOADER
>WDCT DATA &O147 ;WORDS TO LOAD
>READ DATA &O151 ; CODE TO TURN ON READER
END
```

```
TARTAR.PRG
;SAMPLE BOOTSTRAP FOR UNIVAC 1219 cOMPUTER
;COMPATABLE WITH TARTAR LOADER
;PROGRAM UNCONDITIONALY LOADS 147 OCTAL WORDS (LOADER)
;ON BIOCTAL PAPER TAPE STARTING AT OCTAL ADDRESS 436
;THEN JUMPS TO OCTAL 540 (FIRST 2 WORDS ARE HEADER)
;TAPE FORMAT: 76=> BIOCTAL TAPE, IIIII =>INITIAL, TTTTT=> TERMINAL
;76 (BIOCTAL TAPE ID CODE)
;II (MSD INITIAL ADDRESS)
;II
;IT
;TT
;TT (LSD OF FINAL ADDRESS)
;PP (FIRST 6 BITS OF FIRST INSTRUCTION)
;PP
;PP (LAST 6 BITS OF FIRST INSTRUCTION)
;** REMEMBER BOOTSTRAP IGNORES HEADER DATA**
;
;
                        ORG &O500 ; ADDRESS TO START ASSEMBLY
TARBOOT   000500 503400         SIL 0
          000501 507310         ENTSR &O10
          000502 507201         ENTICR 1
          000503 501300         EXF 00  ; TURN ON READER
          000504 000534         BCW READ
          000505 000534         BCW READ
          000506 360000         ENTBK 0 ;SET FRAME COUNT
NXTWD     000507 420003         STRB 3
          000510 700002         ENTALK 2
          000511 440004         STRAL 4
          000512 700000         ENTALK 0
INPUT     000513 501100         IN 0 ;GET FRAME
          000514 000002         BCW 2
          000515 000002         BCW 2
          000516 502100         SKPIIN 0 ; WAIT FOR DATA
          000517 340516         JP LOK-1
          000520 504606         LSHAL 6  ; SHIFT UP 6 BITS
          000521 510002         SLSET 2
          000522 630525         JPALNZ LOK+3
          000523 100003         ENTAU 3
          000524 600513         JPAUZ INPUT
          000525 570004         ISK 4  ;THREE FRAMES?
          000526 340513         JP INPUT ; NO GET MORE
          000527 450536         STRALB &O536 ;YES STORE WORD
          000530 560533         BSK WDCT ;DONE YET?
          000531 340507         JP NXTWD ;NO GET NEXT
          000532 340540         JP &O540  ;yES JP TO LOADER
WDCT      000533 000147         DATA &O147 ;WORDS TO LOAD
READ      000534 000151         DATA &O151 ; CODE TO TURN ON READER

   LABEL TABLE

TARBOOT       000500
INPUT         000513
NXTWD         000507
READ          000534
WDCT          000533
```