CODE (Octal) f	INSTRUCTION	DESCRIPTION	Time μsec.
54	**Replace SElective•SET**	Set A_n for $(Y)_n$=1, (A) → Y	4-6
55	**Replace SElective•ComPlement**	Complement A_n for $(Y)_n$=1, (A) → Y	4-6
56	**Replace SElective•CLear**	Clear A_n for $(Y)_n$=1, (A) → Y	4-6
57	**Replace SElective•SUbstitute**	(Y)n → A_n for Q_n=1, (A) → Y	4-6
‡*60	**JumP** (arithmetic)	Jump to Y if jump j condition satisfied	2-4
60j0	**Remove Interrupt Lockout**	Release master I/O interrupt lockout	2
‡60j1	**Remove Interrupt Lockout, JumP**	RIL same as above, jump to Y	2-4
‡61	**JumP** (manual)	Jump to Y if j condition is satisfied	2-4
^62	**JumP•Y•C_j•ACTIVE INput buffer**	Jump to Y if C_j input buffer active	4
^63	**JumP•Y•C_j•ACTIVE OUTput buffer**	Jump to Y if C_j output buffer active	4
‡*64	**Return JumP** (arithmetic)	If j condition is satisfied, P → Y_L, Addressing mode → Y_U, jump to Y+1	4-6
‡*65	**Return JumP** (manual)		4-6
^66k0	**TERMinate-C_j-INPUT**	Terminate input buffer on C_j	2
^66k1b0	**Remove Interrupt Lockout-ALL**	Release master I/O interrupt lockout	2
^66k1b≠0	**Set Interrupt Lockout-ALL**	Set master I/O interrupt lockout	2
^66k2b0	**Remove Interrupt Lockout-EXternal-ALL**	Release all external channel interrupt lockouts	2
^66k2b≠0	**Set Interrupt Lockout-EXternal-ALL**	Set all external channel interrupt lockouts	2
^66k3b0	**Remove Interrupt Lockout-EXtergal-C_j**	Release external channel interrupt lockout on C_j	2
^66k3b≠0	**Set Interrupt Lockout-EXternal-C_j** Note: Master Clear sets all external channel interrupt	Set external channel interrupt lockout on C_j lockouts and releases master I/O interrupt lockout.	2
^67k0	**TERMinate-C_j-OUTPUT**	Terminate output buffer on C_j	2
^67k1	**TERMinate-C_j-COMmand**	Terminate external function buffer on C_j	2
^67k2	**TERMinate-ALL**	Terminate all buffers	2
*70	**RePeaT**	Execute NI Y times	2-4
71	**B SKip•Bj**	(Bj)≠Y, (Bj)+1 → Bj and read NI; (Bj)=Y, skip NI and clear Bj	4-6
72	**B JumP•Bj**	(Bj)≠0, (Bj)−1 → Bj and jump to Y; (Bj)=0, read NI	4
^73	**INput-C_j** (without monitor mode)	Initiate input buffer on C_j, (Y) → 00100+j	6 min.
^74	**OUTput-C_j** (without monitor mode)	Initiate output buffer on C_j; (Y) → 00120+j	6 min.
^74k2	**EXternal-COMmand-MultiWord-C_j-W(Y)**	Initiate EF buffer on C_j; (Y) → 00140+j	6 min.
^75	**INput-C_j** (with **MONITOR** mode)	Initiate input buffer on C_j with monitor; (Y) → 00100+j, monitor interrupt address is 00040+j	6 min.
^76	**OUTput-C_j** (with **MONITOR** mode)	Initiate output buffer on C_j with monitor; (Y) → 00120+j, monitor interrupt address is 00060+j	6 min.
^76k2	**EXternal-COMmand-MultiWord-C_j-W(Y)-MONITOR**	Initiate EF buffer on C_j with monitor; monitor interrupt address is 00500+j; (Y) → 00140+j	6 min.
7710	**Enter Q With Y**	Enter Q using address in U; (Y) → Q	2-4
7711	**Enter A With Y**	Enter A using address in U; (Y) → A	2-4
7712	**Enter B With Y**	Enter B7 using address in U; (Y) → B7	2-4
^7713k2	**Enter Control memory With Y**	Load I/O extension registers C_j with $(Y+(Bb))_{5-0}$ $Y_{5,4}$=Input$_{16,15}$; $Y_{3,2}$=Output$_{16,15}$; $Y_{1,0}$=EF$_{16,15}$	6 min.
7714	**Store Q With Y**	Store Q using address in U; (Q) → Y	2-4
7715	**Store A With Y**	Store A using address in U; (A) → Y	2-4
7716	**Store B With Y**	Store B7 using address in U; (B7) → Y	2-4
^7717k0	**Store Control memory With Y-C_j•EF**	Store EF BCW C_j at address 00005	6 min.
^7717k1	**Store Control memory With Y-C_j-EXTENS**	Store I/O extension registers C_j at 00005	6 min.
^7717k2	**Store Control memory With Y-C_j-INPUT**	Store Input BCW C_j at 00005	6 min.
^7717k3	**Store Control memory With Y-C_j-OUTPUT**	Store Output BCW C_j at 00005	6 min.
7744	**Test and Set Flag**	(Y) ≠ 0, read NI; (Y)=0, skip NI; always set Y=1s	4-6
^7750	**Enter Absolute Page Register**	Y_{5-0} → APR$_j$	4
^7754	**Store Absolute Page Register**	APR$_j$ → Y	2-4
7763	**Load B and JumP**	(P) → B7, jump to (Y)	4
7764	**Direct Load B and JumP**	(P) → B7, set direct addressing mode, jump to (Y)	4
7765	**Page Load B and JumP**	(P) → B7, set page addressing mode, jump to (Y)	4

Y—The operand designator modified by Bb
k, j Special j and k designators
Y—The operand; Y or (Y)
‡See Address Mode Selection k designators

NI=Next Instruction
BCW=Buffer Control Word.
APR=Absolute page register
min—minimum

UNIVAC 1830A COMPUTER MODIFIED

CODE (Octal) f	INSTRUCTION	DESCRIPTION	Time μsec.
01	**Right SHift•Q**	Shift (Q) Right by Y	2-4-6
02	**Right SHift•A**	Shift (A) Right by Y	2-4-6
03	**Right SHift•AQ**	Shift (AQ) Right by Y	2-4-6
*04	**COMpare•A,Q•AQ**	Sense (j); A_j=At	4
05	**Left SHift•Q**	Shift (Q) Left by Y	2-4-6
06	**Left SHift•A**	Shift (A) Left by Y	2-4-6
07	**Left SHift•AQ**	Shift (AQ) Left by Y	2-4-6
10	**ENTer•Q**	Y → Q	2-4
10k0	**CLear•Q**	0 → Q	2
11	**ENTer•A**	Y → A	2-4
10k0	**CLear•A**	0 → A	2
12	**ENTer•Bj**	Y → Bj	2-4
12k0	**CLear•Bj**	0 → Bj	2
12j0	**NO-OPeration**	Enter Bb with 0 (do nothing operation)	2-4
^13k0	**EXternal-COMmand-C_j-W(Y)-MONITOR**	(Y) → Cj, Y → 140+j monitor interrupt address is 500+j	6 min.
^13k1	**EXternal-COMmand-C_j-W (Y)-MONFORCE**	Force (Y) → Cj, Y → 140+j address is 500+j	6 min.
^13k2	**EXternal-COMmand-C_j-W (Y)**	Initiate external function Y → 140+j	6 min.
^13k3	**EXternal-COMmand-C_j-W (Y)-FORCE**	Force (Y) → Cj, Y → 140+j	6 min.
14k≠0	**STo•Re•Q**	(Q) → Y	2-4
14k0	**ComPlement•Q**	(Q) → Q	2
15k≠4	**STo•Re•A**	(A) → A	2-4
15k4	**ComPlement•A**	(A) → A	2
16	**STo•Re•B**	(Bj) → Y	2-4
^17k0	**JumP•Y•C_j•COMACTIVE**	Jump to Y if external function buffer active on C_j	4
^17k1	**JumP•L(Y)•C_j•COMACTIVE**	Jump to (Y)L if external function buffer active on C_j	4
^17k2	**STo•Re•C_j•W (Y)•FORCE**	Input data word C_j → Y, force IDA	6 min.
^17k3	**STo•Re•C_j•W (Y)**	(00520+j) → Y	4-6
20	**ADD•A**	(A)+Y → A	2-4
21	**SUBtract•A**	(A)−Y → A	2-4
*22	**MULtiply**	(Q) Y → AQ	18
*23	**DIVide**	(AQ) Y → Q; R → A	32
*23k7	**SQuare RooT**	√Q → Q; residue → A	32
24	**RePLace•A+Y**	(A)+(Y) → Y & A	4-6
25	**RePLace•A−Y**	(A)−(Y) → Y & A	4-6
*26	**ADD•Q**	(Q)+Y → Q	2-4
*27	**SUBtract•Q**	(Q)−Y → Q	2-4
30	**ENTer•Y+Q**	Y+(Q) → A	2-4
31	**ENTer•Y−Q**	Y−(Q) → A	2-4
32	**STo•Re•A+Q**	(A)+(Q) → A & Y	2-4
33	**STo•Re•A−Q**	(A)−(Q) → A & Y	2-4
34	**RePLace•Y+Q**	(Y)+(Q) → Y & A	4-6
35	**RePLace•Y−Q**	(Y)−(Q) → Y & A	4-6
36	**RePLace•Y+1**	(Y)+1 → Y & A	4-6
37	**RePLace•Y−1**	(Y)−1 → Y & A	4-6
*40	**ENTer•LP**	L(Y)(Q) → A	2-4
41	**ADD•LP**	(A)+L(Y)(Q) → A	2-4
42	**SUBtract•LP**	(A)−L(Y)(Q) → A	2-4
43	**COMpare•MASK**	(A)−L(Y)(Q) sense (j), A_j+L(Y)(Q)]; $(A)_j$=(A)t	2-4
*44	**RePLace•LP**	L((Y)(Q)) → Y & A	4-6
45	**RePLace•A+LP**	(A)+L((Y)(Q)) → Y & A	4-6
46	**RePLace•A−LP**	(A)−L((Y)(Q)) → Y & A	4-6
47	**STo•Re•LP**	L((A) (Q)) → Y	2-4
50	**SELective•SET**	Set A_n for Y_n=1	2-4
51	**SELective•ComPlement**	Complement A_n for Y_n=1	2-4
52	**SELective•CLear**	Clear A_n for Y_n=1	2-4
53	**SELective•SUbstitute**	Y_n → A_n for Q_n=1	2-4

Y—The operand designator modified by Bb
k, j Special j and k designators

Y—The operand; Y or (Y)
min—minimum
LP or L( )—Logical Product

JP & RJP J-DESIGNATORS

j	JP f60	RJP f64	JP f61	RJP f65
0	(No Jump)*		Uncond. Jump	
1	Uncond. Jump*		KEY1	
2	QPOS		KEY2	
3	QNEG		KEY3	
4	AZERO		STOP	
5	ANOT zero		STOP5	
6	APOS		STOP6	
7	ANEG		STOP7	
↑	f62		f63	
0-17$_8$	C$\hat{j}$•ACTIVEIN		C$\hat{j}$•ACTIVEOUT	

*f60 Bootstrap or Spec L/O set, clear L/O;
Bootstrap or Spec L/O clear, clear I/O L/O and set C$\hat{j}$ EIE line after external interrupt.

NORMAL K-DESIGNATORS

k	READ Code	READ Origin	STORE Code	STORE Dest.	REPLACE Code	REPLACE Origin	REPLACE Dest.
0	'blank'	U_L	Q	Q	—	—	—
1	L	M_L	L	M_L	L	M_L	M_L
2	U	M_U	U	M_U	U	M_U	M_U
3	W	M	W	M	W	M	M
4	X	XU_L	A	A	—	—	—
5	LX	XM_L	CPL	Cpl M_L	LX	XM_L	M_L
6	UX	XM_U	CPU	Cpl M_U	UX	XM_U	M_U
7	A	A	CPW	Cpl M	—	—	—

*J-DESIGNATORS

j	COM•A, •Q, •AQ f04	DIV f23	SQRT f23 k7	ADD•Q, SUB•Q f26	ENT•LP, RPL•LP f27 f40 f44	RPT f70
0	no skip	no skip	no skip	no skip	(no mod)	: Y of NE=Y
1	unconditional skip	SKIP	SKIP	SKIP	SKIP	ADV : Y of NE=Y+1
2	YLESS : $Y_{\leq}(Q)$	NOOver Flow	—	APOS	EVEN parity	BACK : Y of NE=Y−1
3	YMORE : $Y_{>}(Q)$	Over Flow	—	ANEG	ODD parity	ADDB : Y of NE=Y+B^b
4	YIN : $(Q)_{\geq}Y$ and $Y_{>}(A)$	AZERO	NOREM	QZERO	AZERO	R : Y of NE=Y|+B^6|$\sqrt{}$
5	YOUT : $(Q)_{<}Y$ or $Y_{\geq}(A)$	ANOT zero	REM	QNOT zero	ANOT zero	ADVR : Y of NE=Y+1|+B^6|$\sqrt{}$
6	YLESS : $Y_{\leq}(A)$	skip	—	QPOS	APOS	BACKR : Y of NE=Y−1|+B^6|$\sqrt{}$
7	YMORE : $Y_{>}(A)$	no skip	—	QNEG	ANEG	ADDBR : Y of NE=Y+B^b|+B^6|$\sqrt{}$

$\sqrt{}$modifies Y address for the store portion by (B^6) if repeated instruction is replace class.
NE—Next execution.

NORMAL J-DESIGNATORS

j	(Not applicable on * or ^) Skip Code
0	no skip
1	SKIP
2	QPOS
3	QNEG
4	AZERO
5	ANOT zero
6	APOS
7	ANEG

ADDRESS MODE SELECTION ‡K-DESIGNATORS

k	f-Code	Operand	Remarks
0	f64 & f65	Y=U_L	CA → Y_U, jump in CA
1	f64 & f65	Y=M_L	CA → Y_U, jump in CA
2	f64 & f65	Y=M_U	CA → Y_U, jump in CA
3	f60 & f61 f64 & f65	Y=M	Set addressing mode contained in $(Y)_{15}$ and jump to $(Y)_L$ See note 3
			CA → Y_U, jump in CA
4	f64 f65	Y=U_L	Obtain instruction & operand in CA, set DA, P → Y_L, previous addressing mode → Y_U, jump in DA.
			Obtain instruction & operand in CA, set PA, P → Y_L, previous addressing mode → Y_U, jump in PA.
5	f64 f65	Y=M_L	Obtain instruction & operand in CA, set DA, P → Y_L, previous addressing mode → Y_U, jump in DA.
			Obtain instruction & operand in CA, set PA, P → Y_L, previous addressing mode → Y_U, jump in PA.
6	f64 f65	Y=M_U	Obtain instruction & operand in CA, set DA, P → Y_L, previous addressing mode → Y_U, jump in DA.
			Obtain instruction & operand in CA, set PA, P → Y_L, previous addressing mode → Y_U, jump in PA.
7	f64 & f65	Y=A	CA → Y_U, jump in CA

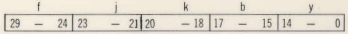
Note: (1) CA=Current Addressing mode (DA or PA)
(2) DA=Direct Addressing mode; PA=Page Addressing mode
(3) Bit 2_{15}=mode definition bit; 0=DA, 1=PA

LEGEND

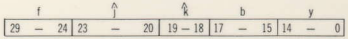
M—Memory word (30 bits)
M_L—Lower half memory word
M_U—Upper half memory word
X—Sign bit extended
Cpl—Complement
C$\hat{j}$—Channel $\hat{j}$
A—A-register
Q—Q-register

INSTRUCTION WORD FORMATS

FORMAT I

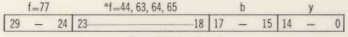


NON-I/O Instruction

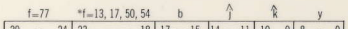


I/O Instruction

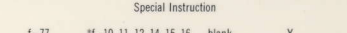
FORMAT II



General Instruction**



Special Instruction



Direct Addressing Instruction**

*f=Format II Function Code **Forced k=3

MEMORY ADDRESS ASSIGNMENT

OCTAL ADDRESS RANGE	USE
00000	Program Fault Interrupt Entrance Address
00001	Count-down Clock Interrupt Entrance Address
00002	Memory Protect Interrupt Entrance Address
00003	Input Power Failure Interrupt Entrance Address
00004	Power On Entrance Address
00005	Read Control Memory Storage Address
00006-00017	Unassigned
00020-00037	External Interrupt Entrance Addresses
00040-00057	Input Monitor Interrupt Entrance Addresses
00060-00077	Output Monitor Interrupt Entrance Addresses
00100-00117	*Input Buffer Control Words
00120-00137	*Output Buffer Control Words
00140-00157	*External Function Buffer Control Words
00160	Real Time Clock
00161	Count-down Clock
00162-00177	Unassigned
00200-00217	ESI Input Buffer Termination Words
00220-00237	ESI Output Buffer Termination Words
00240-00257	ESI External Function Buffer Termination Words
00260-00477	Unassigned
00500-00517	External Function Buffer Monitor Interrupt Entrance Addresses
00520-00537	Interrupt Word Storage Addresses
00540-00577	Unassigned
00600-00617	Intercomputer Time-Out Entrance Address
00620 and over	Unassigned
40000-40777	NDRO Memory when enabled

*Located in I/O Control Memory when installed