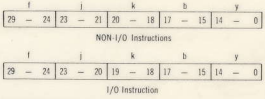
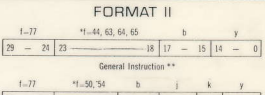
INSTRUCTION WORD FORMATS

FORMAT I



FORMAT II



*l=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
00005-00017	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 Interrupt Entrance Address
00620 and over	Unassigned

UNIVAC 1830A COMPUTER

CODE (Octal) f	INSTRUCTION	DESCRIPTION	Time μsec.
01	Right SHift,Q	Shift (Q) Right by Y	2-6
02	Right SHift,A	Shift (A) Right by Y	2-6
03	Right SHift,AQ	Shift (AQ) Right by Y	2-6
*04	COMpare,A,Q,AQ	Sense (j); $A_i=A_j$	2-4
05	Left SHift,Q	Shift (Q) Left by Y	2-6
06	Left SHift,A	Shift (A) Left by Y	2-6
07	Left SHift,AQ	Shift (AQ) Left by Y	2-6
10	ENTer,Q	Y → Q	2-4
10	CLear,Q	Y=0, Y → Q	2-4
11	ENTer,A	Y → A	2-4
11	CLear,A	Y=0, Y → A	2-4
12	ENTer,B^{k}	Y → B^{k}	2-4
12	NO OPeration	Enter B^{0} with 0 (do nothing operation)	2-4
^13k0	EXternal COMmand,C^{k},W(Y),MONITOR	(Y) → $\widehat{C_j}$ (interrupt at 00140+j)	4
^13k1	External COMmand,C^{k},W(Y),MONFORCE	Force (Y) → $\widehat{C_j}$ (interrupt at 00140+j)	4
^13k2	External COMmand,C^{k},W(Y)	Initiate external function and read NI	4
^13k3	EXternal COMmand,C^{k},W(Y),FORCE	Force (Y) → $\widehat{C_j}$	4
14k≠0	SToRe,Q	(Q) → Y	2-4
14k0	ComPLement,Q	(Q)' → Q	2-4
15k≠4	SToRe,A	(A) → Y	2-4
15k4	ComPLement,A	(A)' → A	2-4
16	SToRe,B^{k}	(B)j → Y	2-4
^17k0	JumP,Y,C^{k},COMACTIVE	Jump to Y if external function buffer active on $\widehat{C_j}$	4-6
^17k1	JumP,L(IY),C^{k},COMACTIVE	Jump to Y_L if external function buffer active on $\widehat{C_j}$	4-6
^17k2	SToRe,C^{k},W(Y) FORCE	$\widehat{C_j}$ → Y (abnormal test mode)	4-6
^17k3	SToRe,C^{k},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_I	32
*23k7	SQuare RooT,Q	$\sqrt{Q}$ → 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	SToRe,A,+,Q	(A)+(Q) → A & Y	2-4
33	SToRe,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)]; $(A)_j=(A)_j$	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

CODE (Octal) j	INSTRUCTION	DESCRIPTION	Time μsec.
47	STorE•LP	L (A) (Q) → Y	2-4
50	SELective•SET	Set An for Yn=1	2-4
51	SELective•ComPlement	Complement An for Yn=1	2-4
51k4	ComPlement•A	When Y is 77777, then (A)' → A	2-4
52	SELective•CLear	Clear An for Yn=1	2-4
53	SELective•SUbstitute	Yn → An for Yn=1	2-4
54	Replace SELective•SET	Set An 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 An for (Y)n=1, (A) → Y	4-6
57	Replace SELective•SUbstitute	(Y)n → An for Qn=1, (A) → Y	4-6
*60	JumP (arithmetic)	Jump to Y if jump j condition satisfied	2-4
60j0	Remove Interrupt Lockout	RIL	2-4
60j1	Remove Interrupt Lockout, JumP	RIL, jump to Y (see k-designators, Address Mode Selection)	2-4
61	JumP (manual)	Jump to Y if j condition is satisfied	2-4
62	JumP (if•C^j has•ACTIVE INput buffer)	Jump to Y if C_j input buffer active	4
63	JumP (if•C^j has•ACTIVE OUTput buffer)	Jump to Y if C_j output buffer active	4
*64	Return JumP (arithmetic)	Jump to Y+1 and P→Y, if j condition is satisfied (see JP and RJP j designators & k-designators, Address Mode Selection) (see JP and RJP j designators)	4-6
*65	Return JumP (manual)		4-6
^66k0	TERMinate•C^k•INPUT	Terminate input buffer on C_j	2
^66k1	Remove Interrupt Lockout•ALL	RIL on all internal and external channels not locked out by SIL-K (166k2, 3)	2
^66k2	Remove Interrupt Lockout•EXternal•ALL	RIL for external interrupts on all channels	2
^66k3	Remove Interrupt Lockout-EXternal-C^k	RIL for external interrupts on C_j	2
^66k1b1	Set Interrupt Lockout•ALL	Set external and internal lockout on all channels	2
^66k2b1	Set Interrupt Lockout-EXternal•ALL	Set external interrupt lockout on all channels	2
^66k3b1	Set Interrupt Lockout-EXternal-C^k	Set external interrupt lockout on C_j	2
^67k0	TERMinate•C^k•OUTPUT	Terminate output buffer on C_j	2
^67k1	TERMinate•C^k•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•B^k	(B)=Y, skip NI and clear B^j; (B)≠Y, (B)+1 → B^j and read NI	4-6
72	B JumP•B^k	(B)=0, read NI; (B)≠0, (B)−1 → B^j and jump to Y	4
^73	INput•C^k (without monitor mode)	Buffer IN on C_j; (Y) → 00100+j	4
^74	OUTput•C^k (without monitor mode)	Buffer OUT on C_j; (Y) → 00120+j	4
^74k2	EXternal-COMmand-MultiWord•C^k•W(Y)	Buffer EX function; (Y) → 00140+j	4
^75	INput•C^k (with MONITOR mode)	Buffer IN on C_j with monitor; (Y) → 00100+j; monitor interrupt address is 00040+j	4
^76	OUTput•C^k (with MONITOR mode)	Buffer OUT on C_j with monitor; (Y) → 00120+j; monitor interrupt address is 00060+j	4
^76k2	EXternal-COMmand MultiWord•C^k•W(Y)•MONITOR	Buffer EX function on C_j; (Y) → 00140+j; monitor interrupt address is 00500+j	4
7711	Enter•A•U address**	Enter A using address in U; (Y) → A $\begin{cases} Y=U_{18-0} \\ Y_{35-19}=0 \end{cases}$	2-4
7715	Store•A•U address**	Store (A) using address in U; (A) → Y	2-4
7744	Test and set flag**	(Y)≠0, read NI; (Y)=0, Set Y to 0 and skip NI	4-6
7750	Enter•absolute page register**	Y_{13} → APR	4
7754	Store absolute page register**	APR → Y	4
7763	Load•B^j and jump**	(P) → B^j; jump to (Y)	4
7764	Load•B^j, jump•set direct mode**	(P) → B^j; set direct mode, jump to (Y)	4
7765	Load•B^j, jump•set page mode**	(P) → B^j; set indirect mode; jump to (Y)	4

Y—The operand designator modified by B^k
* [Special j and k designators
Y—The operand; Y or (Y)
NI—Next Instruction
**Machine language
APR—Absolute page register

JP & RJP j-DESIGNATORS

j	JP f60	RJP f64	JP f61	RJP f65
0	(No Jump)*		Uncond. Jump	
1	Uncond. Jump*		JUMP 1	
2	Q POS		JUMP 2	
3	Q NEG		JUMP 3	
4	A ZERO		STOP	
5	A NOT zero		STOP 5	
6	A POS		STOP 6	
7	A NEG		STOP 7	
j	f62j		f63j	
0-17₈	C^k ACTIVE IN		C^k ACTIVE OUT	

*60 clears interrupt mode

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	ADD•Q, SUB•Q f26 f27	ENT•LP, RPL•LP f40 f44	RPT f70	SQRT f23 k7
0	no skip	no skip	no skip	no skip	(no mod): Y of NE=Y	no skip
1	unconditional skip	SKIP	SKIP	SKIP	ADV : Y of NE=Y+1	SKIP
2	Y LESS : Y<(Q)	NO Over Flow	A POS	EVEN parity	BACK : Y of NE=Y−1	REM
3	Y MORE : Y>(Q)	Over Flow	A NEG	ODD parity	ADD B : Y of NE=Y+B^k	NO REM
4	Y IN : (Q)≥Y and Y>(A)	A ZERO	Q ZERO	A ZERO	Rpl. inc : Y of NE=Y|+B^k|$\checkmark$	—
5	Y OUT : (Q)>Y or Y≤(A)	A NOT zero	Q NOT zero	A NOT zero	ADVR : Y of NE=Y+1|+B^k|$\checkmark$	—
6	Y LESS : Y<(A)	A POS	Q POS	A POS	BACK R : Y of NE=Y−1|+B^k|$\checkmark$	—
7	Y MORE : Y>(A)	A NEG	Q NEG	A NEG	ADD B R : Y of NE=Y+B^k|+B^k|$\checkmark$	—

$\checkmark$ B^k Increment if NI is RPL class; increment Y address for the store portion of the replace.
NE—Next execution.

NORMAL j-DESIGNATORS

j	(Not applicable on * or ^)
J	Skip Code
0	no skip
1	SKIP
2	Q POS
3	Q NEG
4	A ZERO
5	A NOT zero
6	A POS
7	A NEG

ADDRESS MODE SELECTION k-DESIGNATORS

k	f-Code	Operand	Remarks
0	f64 & f65	$Y=U_L$	jump in current mode to Y
1	f64 & f65	$Y=M_L$	jump in current mode to $(Y)_L$
2	f64 & f65	$Y=M_U$	jump in current mode to $(Y)_U$
3	f60 & f61 f64 & f65	Y=M	set mode contained in operand (Y) and jump to (Y) jump in current mode to (Y)
4	f64 f65	$Y=U_L$	Obtain instruction in current mode, set DA mode, store P & jump in DA mode to Y. Obtain instruction in current mode, set IA mode, store P & jump in IA mode to Y
5	f64 f65	$Y=M_L$	Obtain instruction in current mode, set DA mode, store P & jump in DA mode to $(Y)_L$ Obtain instruction in current mode, set IA mode, store P & jump in IA mode to $(Y)_L$
6	f64 f65	$Y=M_U$	Obtain instruction in current mode, set DA mode, store P & jump in DA mode to $(Y)_U$ Obtain instruction in current mode, set IA mode, store P & jump in IA mode to $(Y)_U$
7	f64 & f65	Y=A	jump in current mode to (A)

Note: (1) DA—Direct Addressing, IA—Indirect Addressing
(2) Bit 2_{35}=mode definition bit; 0=DA, 1=IA.

LEGEND
M—Memory word (30 bits)
M_L—Lower half memory word
M_U—Upper half memory word
X—Sign bit extended
Cpl—Complement
A—A-register
Q—Q-register