Code (Octal)	Instruction	Description
01	Right Shift•Q	Shift (Q) Right by Y
02	Right Shift•A	Shift (A) Right by $\overline{Y}$
03	Right Shift•AQ	Shift (AQ) Right by Y
*04	COMpare•A_r•Q_r•AQ	Sense (j); A_r—A_r
05	Left Shift•Q	Shift (Q) Left by Y
06	Left Shift•A	Shift (A) Left by $\overline{Y}$
07	Left Shift•AQ	Shift (AQ) Left by Y
10	ENTer•Q	Y→Q
10	CLear•Q	$\overline{Y}$→0, Y→Q
11	ENTer•A	Y→A
11	CLear•A	$\overline{Y}$→0, Y→A
12	ENTer•B^x	Y→B^j
12	CLear•B^x	$\overline{Y}$→0, Y→B^j
12	NO-OPeration	Enter B^x with 0 (do nothing operation)
^13k0	EXternal-COMmand•C^k•W(Y)•MONITOR	(Y)→O; (interrupt at 00500+j)
^13k1	EXcom-COMmand•C^k•W(Y)•MONFORCE	(Y)→O; (interrupt at 00500+j)
^13k2	EXternal-COMmand•C^k•W(Y)	(Y)→O
^13k3	EXternal-COMmand•C^k•W(Y)•FORCE	(Y)→O
14	STOreQ	(Q)→Y
14k0	ComPlement•Q	When Y is Q; then Q^1→Q
15	STOreA	(A)→Y
16	STOre•B^x	(B) →Y
^17k0	JumP•Y•C^k•COMACTIVE	Jump to Y if external function buffer active
^17k1	JumP•L(Y)•C^k•COMACTIVE	Jump to L(Y) if external function buffer active
^17k2	STOre•C^k•W(Y)•FORCE	Force (O)→Y—(abnormal test mode)
^17k3	STOre•C^k•W(Y)	(00520+j)→Y
20	ADD•A	(A)+Y→A
21	SUBtract•A	(A)−Y→A
22	MULtiply	(Q)Y→AQ
*23	DIVide	(AQ) Y→Q; R→A
*23k7	SQuare RooT	$\sqrt{Q}$→Q; remainder→A
24	REPLace•A↔Y	(A)↔(Y)→Y&A
25	REPLace•A−Y	(A)−(Y)→Y&A
*26	ADD•Q	(Q)+Y→Q
*27	SUBtract•Q	(Q)−Y→Q
30	ENTer•Y→Q	Y→Q→A
31	ENTer•Y−Q	Y−Q→A
32	STOre•A+Q	(A)+(Q)→Y&A
33	STOre•A−Q	(A)−(Q)→Y&A
34	REPLace•Y+Q	(Y)+(Q)→Y&A
35	REPLace•Y−Q	(Y)−(Q)→Y&A
36	REPLace•Y+1	(Y)+1→Y&A
37	REPLace•Y−1	(Y)−1→Y&A
40	ENTer•LP	L(Y)(0)→A
41	ADD•LP	L($\overline{Y}$)(0)+(A)→A
42	SUBtract•LP	(A)−L(Y)(0)→A

Code (Octal)	Instruction	Description
43	COMpare•MASK	(A)−L(Y)(0) sense (j); A+L(Y)(0)]; $(A)_i$=$(A)_j$
44	REPLace•LP	L(Y)(0)→Y&A
45	REPLace•A+LP	(A)+L(Y)(0)→Y&A
46	REPLace•A−LP	(A)−L(Y)(0)→Y&A
47	STOReA•LP	L(A)(0)→Y; $(A)_i$=$(A)_j$
50	SELective•SET	Set $(A)_i$ for $Y_{j=1}$
51	SELective•ComPlement	Complement $(A)_i$ for $Y_{j=1}$
51k4	ComPlement•A	When Y is 77777, then A!→A
52	SELective•CLear	Clear $(A)_i$ for $Y_{j=1}$
53	SELective•SUbstitute	$Y_{j=0}$→$(A)_j$ for $(Q)_{j=1}$
54	Replace SElective•SET	Set $(A)_j$ for $(Y)_{j=1}$→Y&A
55	Replace SElective•CP	Complement $(A)_j$ for $(Y)_{j=1}$→Y&A
56	Replace SElective•CL	Clear $(A)_j$ for $(Y)_{j=1}$→Y&A
57	Replace SElective•SU	$(Y)_j$↔$(A)_j$ for $(Q)_{j=1}$→Y
*60	JumP (arithmetic)	Jump to Y if jump j-condition is satisfied
60j0	Remove Interrupt Lockout	RIL on all internal channels and all external channels not locked out by SIL-EX
60j1	Remove Interrupt Lockout JumP•Y	RIL; jump to Y
*61	JumP (manual)	Jump to Y if jump j-condition is satisfied
^62	JumP (if•C^k has•ACTIVE INput buffer)	Jump to Y if C^k input buffer active
^63	JumP (if•C^k has•ACTIVE OUTput buffer)	Jump to Y if C^k output buffer active
*64	Return JumP (arithmetic)	Jump to Y+1 and P+1→Y if j-condition
*65	Return JumP (manual)	is satisfied (see JP and RJP j-designators)
^66	TERMinate•C^k•INPUT	Terminates input buffer on C^j
^66k1b0	Remove Interrupt Lockout•ALL	RIL on all internal channels and all external channels not locked out by SIL-EX
^66k2b0	Remove Interrupt Lockout-EXternal•ALL	RIL for external interrupts on all channels
^66k3b0	Remove Interrupt Lockout-EXternal•C^k	RIL for external interrupts on C^j
^66k1b0	Set Interrupt Lockout•ALL	Sets external and internal lockout on all channels
^66k2b0	Set Interrupt Lockout-EXternal•ALL	Sets external interrupt lockout on all channels
^66k3b0	Set Interrupt Lockout-EXternal•C^k	Sets external interrupt lockout on C^j
^67	TERMinate•C^k•OUTPUT	Terminates output buffer on C^j
^67k1	TERMinate•C^k•COMmand	Terminates external function buffer on C^j
^67k2	TERMinate•ALL	Terminates all buffers
*70	RePeaT	Execute NI Y times
71	BSKip•B^x	$(B)^j$=Y, skip NI and clear $(B)^j$; $(B)^j \neq \overline{Y}$, advance B^j and read NI
72	BJumP•B^x	$(B)^j$=0, read NI; $(B)^j \neq 0$, $(B)^j$−1 and jump to address Y
^73	INput•C^k (without monitor mode)	Buffer in on C^j; buffer control word→00100+j
^74	OUTput•C^k (without monitor mode)	Buffer out on C^j; buffer control word→00120+j
^74k2	EXternal-COMmand-MultiWord•C^k•W(Y)	Buffer out on C^j; (Y)→(00140+j)
^75	Input•C^k (with•MONITOR mode)	Buffer in on C^j with monitor; buffer control word→00100+j; (interrupt address 00040+j)
^76	OUTput•C^k (with MONITOR mode)	Buffer out on C^j with monitor; buffer control address→00120+j; (interrupt address 00060+j)
^76k2	EXternal-COMmand-MultiWord•C^k•W(Y)•MONITOR	Buffer out on C^j; (interrupt at 00500+j)

x { Special j and k designators (* j, $\hat{j}$ or $\hat{k}$)
Y—The operand; Y, (Y) or (A)

UNIVAC CP-642B COMPUTER
REPERTOIRE OF INSTRUCTIONS

NORMAL i-DESIGNATORS

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

NORMAL k-DESIGNATORS

k	READ Code	READ Origin	STORE Code	STORE Dest.	REPLACE Code	REPLACE Origin	REPLACE Dest.
0	'blank'	U_L	Q	Q	'not used'	—	—
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	'not used'	—	—
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	'not used'	—	—

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
U_L—U-register, lower half

SPECIAL ^i-DESIGNATORS

i	COM-A ,-Q ,-AQ f04	DIV f23	ADD-Q, SUB-Q f26 / f27	ENT-LP, RPL-LP f40 / f44	SQRT f23 k7	ADDRESS MODIFICATIONS RPT f70
0	(no skip)	(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	Y LESS : Y≥(Q)	NO Over Flow	A POS	EVEN parity	'not used'	BACK : Y of NE=Y−1
3	Y MORE : Y<(Q)	Over Flow	A NEG	ODD parity	'not used'	ADD B : Y of NE=Y+B^3
4	Y IN :(Q)≥Y and Y>(A)	A ZERO	Q ZERO	A ZERO	NO REM	Rpl. inc. : Y of NE=Y|+B^6| √
5	Y OUT :(Q)<Y or Y (A)	A NOT Zero	Q NOT Zero	A NOT Zero	REM	ADV R : Y of NE=Y+1|+B^6| √
6	Y LESS : Y≥(A)	SKIP	Q POS	A POS	'not used'	BACK R : Y of NE=Y−1|+B^6| √
7	Y MORE : Y<(A)	no skip	Q NEG	A NEG	'not used'	ADD BR : Y of NE=Y+B^6|+B^6| √

√ If NI is RPL class, B^6 increments Y address for the store portion of the replace.
NE—Next execution.

NORMAL b-DESIGNATOR

b=0: Do not modify y
b=1: Add (B1) to y
b=2: Add (B2) to y
b=3: Add (B3) to y
b=4: Add (B4) to y
b=5: Add (B5) to y
b=6: Add (B6) to y
b=7: Add (B7) to y

k-DESIGNATORS

f60, 61, 64, 65

k=0, 4 Jump to Y
k=1, 3, 5 Jump to (Y)L
k=2, 6 Jump to (Y)U
k=7 Jump to the Address in A.

JP & RJP ^j & ^i-DESIGNATORS

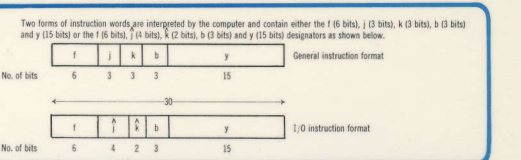
^j	JP f60	RJP f64	JP f61	RJP f65
0	(No Jump)#		(Uncond. Jump)	
1	(Uncond. Jump)#		KEY 1	
2	Q POS		KEY 2	
3	Q NEG		KEY 3	
4	A ZERO		STOP	
5	A NOT Zero		STOP 5	
6	A POS		STOP 6	
7	A NEG		STOP 7	
^i	62 ↑		63 ↑	
0-17	C^i ACTIVE IN		C^i ACTIVE OUT	

#60 Clears interrupt & bootstrap modes.

CP-642B COMPUTER MEMORY ASSIGNMENTS

MEMORY TYPE	MEMORY ADDRESSES	FUNCTION
MAGNETIC CORE	00000	Fault Entrance
	00001-00017	Unassigned
	00020-00037	External Interrupt Entrance
	00040-00057	Input Monitor Interrupt Entrance
	00060-00077	Output Monitor Interrupt Entrance
THIN-FILM	00100-00117	Input Buffer Control Registers
	00120-00137	Output Buffer Control Registers
	00140-00157	External Function Buffer Control Registers
	00160	Real-Time Clock
	00161-00167	B1 through B7 Index Registers
	00170-00177	Unassigned Film Locations
MAGNETIC CORE	00200-00477	Unassigned Core Locations
	00500-00517	External Function Monitor Interrupt Entrance
	00520-00537	External Interrupt Code Storage
UNIFLUXOR	00540-00577	NDRO Bootstrap Program I
	00540-00577	NDRO Bootstrap Program II
MAGNETIC CORE	00600-00617	Intercomputer Time-Out Interrupt Entrance
	00620-77777	Unassigned Core

Two forms of instruction words are interpreted by the computer and contain either the f (6 bits), j (3 bits), k (3 bits), b (3 bits) and y (15 bits) or the f (6 bits), ^j (4 bits), ^k (2 bits), b (3 bits) and y (15 bits) designators as shown below.



SPERRY RAND

UNIVAC
FEDERAL SYSTEMS DIVISION