46 - 47 - 47 - 47 - 47 - 47 - 47 - 47 -	HI7ROM - H17 ROM CODE LISTING.	186.	HEATH HBASH VI.4 01/20/78 PAGE 1 15.54109 11-MAY-78



		54				
:		55.		ORG	30000A	
030.000	303 014 037	57		d.W.	BOOT	BOOT CODE
		59	*	MEMORY	DIAGNOSTIC.	
			*			
030.003 0	041 300 377 071	6 2		LXI	H,-64	
:		4 7		хснв		
	166			HLT	H 40100A	(HL) = SIAKI FAUSE FOR ADJUSTMENT
		89	*		START	
		2,5	*	(DE) =	END	
		72.	*	ZERO TI	ZERO TEST AREA	
030.014 0	042 076 040	:	K	SHLD	40100A-2	
:		•		X		
:	302 017 030	78		JNE	MEM1	
		80 81	**	START THAT RE	SULT TO THE EXPE	T TESTING HEMORY. INCREMENT EACH BYTE IN TURN, AND COMPARE RESULT TO THE EXPECTED VALUE
	000			MUI	В,О	(B) = EXPECTED VALUE
030.032 0	052 076 040 004	- ;	MEM2	INR	40100A-2 B	
:	564	8 8 8 8 7	MEM3	INR	Σ	
	176	88 8		20X	E. Ca	(A) = VALUE
030.041	312 046 030	8		, E	MEM4	IS OK
			*	HAVE EI	ERROR. (HL) = ADDR	ADDRESS OF BYTE IN ERROR
030.044	166 000	4 0		H T		
		96	MEN4	XXI	I	
030.047	315 216 030 302 036 030	8 6 6 6		CALL	♦CDEHL MEM3	NOT AT END OF PASS
	032	100		JMP	MEN2	AT END OF PASS



100X ## \$COMP - COMPARE TWO CHARACTER STRINGS. 100X ## \$COMP - COMPARE TWO CHARACTER STRINGS. 100X # \$COMP - COMPARE TWO CHARACTER STRINGS. 100X # \$COMP COMPARE TWO CHARACTER STRINGS. 110X # \$COMP COMPARE TWO CHARACTER STRINGS. 110X # \$COMP CHARACTER STRINGS. 110X # \$COMP STRINGS	105X ** \$COMP - COMPARE TWO CHARACTER STRINGS. 106X * \$COMP - COMPARES TWO BYTE STRINGS. 106X * \$COMP - COMPARES TWO BYTE STRINGS. 106X * \$COMP - COMPARES TWO BYTE STRINGS. 106X * \$COMP - COMPARE COUNT. 110X * \$COMP - COMPARE COUNT. 111X * \$COMP - COMPARE COUNT. 11X * \$	105X ## \$COMP - COMPARE TWO CHARACTER STRINGS. 106X # \$COMP - COMPARE TWO CHARACTER STRINGS. 109X # \$COMP COMPARE TWO CHARACTER STRINGS. 109X # \$COMP COMPARE TWO STRINGS. 109X # \$COMPARE TWO BY STRING # \$COMPARE TWO CHAPACTER STRINGS. 110X # \$COMPARE TWO CHAPACTER STRINGS. 111X # \$COMPARE TWO CHAPACTER TWO CHA							
1000	1000 1000	1006.X 1006.X 1006.X 1006.X 1006.X 1009.X 1009.X 1009.X 1009.X 1009.X 1009.X 1009.X 1000.X 1				105X **		COMPARE TWO CHARACTER	
1073	100X * \$ \$COMP COMPARES TWO BYTE STRINGS. 100X * ENTRY (C) = COMPAGE COUNT 110X * (R) = FWA OF STRING \$2 111X * (R) = FWA	1097 * \$CONF COMPARES TWO BYTE STRINGS. 1097 * ENTRY (C) = COMPARE CONT. 1107 * ENTRY (C) = COMPARE CONT. 1107 * ENTRY (C) = COMPARE CONT. 1112 * EXII (RE) = FWA DE STRING #1. 1112 * EXII (RE) = FWA DE STRING #2. 1112 * EXII (RE) = FWA DE STRING #2. 1113 * EXII (RE) = FWA DE STRING #2. 1114 * EXII (RE) = FWA DE STRING #2. 1115 * EXII (RE) = FWA DE STRING #2. 111				106X *			
109X # ENTRY (C) = COMPARE COUNT	100X 100X 111X 11X	1099				107X * 108X *	\$ COMF C	OMPARES TWO BYTE STRINGS.	
110	110	110				109X *	ENTRY	= COMFARE COUNT	
111	1113	113				110X *		= FWA OF STRING	
1133 ** (C) = LENGTH REMAINING 1143 ** (C) = LENGTH REMAINING 1153 ** (HL) = ADDRESS OF MISHATCH IN STRING * 1153 ** (HL) = ADDRESS OF MISHATCH IN STRING * 1153 ** (HL) = ADDRESS OF MISHATCH IN STRING * 1153 ** (G) = (D) + (OC) 1153 ** (G) = (D) + (OC) 1153 ** (G) = (D) + (OC) 1253 ** (G) = (HL) + (OC) 1254 ** (G) = (HL) + (OC) 1255 ** (G) = (HL) + (OC) 1255 ** (G) = (HL) + (OC) 1255 ** (HL) = (H	1133 ** (C) = LENGTH REMAINING 1143 ** (C) = ADDRESS OF HISMATCH IN STRING\$ 1153 ** (HL) = ADDRESS OF HISMATCH IN STRING\$ 1153 ** (HL) = ADDRESS OF HISMATCH IN STRING\$ 1153 ** (C) = 0 1154 ** (C) = 0 1155 *	113X # (C) = LENGTH REPAINING 114X * (CD) = ADDRESS OF HISHATCH IN STRING # 115X * (HL) = ADDRESS OF HISHATCH IN STRING # 115X * (HL) = ADDRESS OF HISHATCH IN STRING # 115X * (C) = C F HOVE MATCH 115X * (HL) = (HL) + (OC) 125X					TIVE		
114X	114X	114X						CO = - FNGTH EPROTNING	
115X * (HL) = ADDRESS OF HISMATCH IN STRING * (C) = 0 117X * (C) = 0 C C C C C C C C C	1155	115X * (H.L.) = ADDRESS OF HISMATCH IN STRING * (1.5 * FT.) HAVE MATCH IN STRING * (RESS OF MISMATCH IN	
116X * (C SET, HAVE MA 117X * (C) = 0 117X * (C) = 0 117X * (C) = 0 117X * (E) = (DE) + (119X * (HL) + (120X * USES A,F,C,D,E,H,L 121X 122X 032 123X *COMP LDAX D 276 124X RNE 023 125X RNE 023 125X RNE 043 125X RNE 044 127X RET 130X RET 130X * *DADA - FERFORM (H,L) = BEFORE V 135X * ENTRY (H,L) = BEFORE V 135X * ENTRY (H,L) = (H,L) + (135X * ENTRY (H,L) = (H,L) + (135X * USES F,H,L 137X * USES F,H,L 141X * DAD D 321 144X MOV E,A 026 000 144X MOV E,A 031 144X RET 032 144X RET 044X MOV E,A 032 144X RET 044X MOV E,A 033 144X RET 044X RET 044X MOV E,A 034 144X RET 044X MOV E,A 035 144X RET 045 144X RET 046 144X RET 047 144X	116X * (C) = 0 117X * (C) = 0 119X * (RE) = (DE) + (C) = 0 119X * (RE) = (DE) + (C) = 0 119X * (RE) = (DE) + (C) = 0 120X * USES	116X * (C) = 0 117X * (C) = 0 118X * (BE) + (C) = 0 121X 122X 122X 123X * CMP						ADDRESS OF MISMATCH IN STRING #	
117X * (C) = 0 118X * (DE) = (DE) + (C) 119X * USES	117X ** (C) = 0 118X ** (RE) = (DE) + (C) 119X ** (RE) = (DE) + (C) 120X ** USES A,F,C,D,E,H,L 121X *COMP LDAX D 276 124X CMP M 320 125X *COMP LDAX D 320 125X CMP M 320 125X CMP M 321 126X RNE D 322 126X RNE D 323 126X RNE D 324 126X RNE D 325 126X RNE D 326 126X RNE D 327 126X RET CMP (H,L) = BEFORE V 328 135X ** ENTRY (H,L) = BEFORE V 335X ** USES F,H,L 335X ** USES F,H,L 335X ** USES F,H,L 340X RET DAD D 351 144X BDAD D 351 145X BDAD D 351 145X RET RET	117X ** (C) = 0 119X ** (DE) = (DE) + (C) 119X ** (DE) = (DE) + (C) 119X ** (DE) = (DE) + (C) 120X ** (DE) = (DE) + (C) 121X 121X 121X 122X 122				116X *		C. SET, HAVE MATCH	
118X * (DE) + (C) = (DE) +	118X * (DE) + (C) = (DE) +	119X * (JES + (JES) = (DE) + (C) 119X * USES				117X *		0 = (2)	
1199 * * (HL) + (L) + (L	1199 * * (HL) + (L) + (L	119X * (HL) = (HL) + (121X				118X *		= (bE) +	
121X 121X 122X 122X 122X 122X 122X 122X	121X 121X 122X 123X \$CDMF LDAX D 276 124X RNE 276 125X RNE 276 125X RNE 276 125X RNE 277 278 278 279 270 270 270 270 270 270 270 270	121X 122X 123X \$COMP LDAX D 203 123X \$COMP H 300 123X RN D 043 126X INX D 043 127X INX D 043 127X INX H 015 129X JNC C 302 060 030 129X JNC C 130X RET COMP 134X * *DADA - FERFORM (H,L) = BFFORE VAL 135X * * EXIT (H,L) = BFFORE VAL 135X * EXIT (H,L) = H,L) + 135X * GOOD B D 137X * EXIT (H,L) = H,L) + 135X * GOOD B D 137X * BDD D 147X BDD D 321 145X BDD D 321 145X RET					0101	+	
122X 122X 122X 123X 123X 123X 123X 123X 123X 123X 125X	032 122X	122X		:		121X	0.35.3	HAT FULL WEEFINGE	
032 123X \$COMP LDAX D 276 124X CMP M 276 124X CMP M 300 125X INX H 043 127X INX H 015 128X DCR C 301 129X DCR C 302 060 030 129X DCR C 311 133X ** \$DADA - FERFORM (H,L) = 134X ** ENTRY (H,L) = BEFORE VAL 135X ** ENTRY (H,L) = (H,L) + 135X ** ENTRY (H,L) = 135X **	032 123X \$COMP LDAX D 276 124X CMP M 276 124X CMP M 330 126X INX H 043 127X INX H 015 128X DCR C 310 050 030 129X BET 311 130X RET 312 133X ** * \$DADA - FERFORM (H,L) = 1,34X * EXIT (H,L) = (H,L) + 1,34X * EXIT (H,L) + (H,L) + 1,35X * EXIT (H,L) + (H,L) + 1,35X * EXIT (H,L) + (H,L) + 1,35X * USES F,H,L 325 142X * DAD D 325 142X * DAD D 327 143X DAD D 328 142X * DAD D 328 142X * DAD D 329 144X DAD D 321 145X DAD D 331 147X RET	032 123X #COMP LDAX D 276 124X RNE 270 126X RNE 043 126X RNE 015 128X DCR C 311 130X RET 311 133X ## *DADA - FERFORM (H*L) = 134X # *DADA - FERFORM (H*L) = 135X # ENTRY (H*L) = BEFORE VAL 135X # ENTRY (H*L) = (H*L) + 135X # USES F*H*L 137 140X #USE F*H*L 140X #USE F*H*L 137 144X #USE F*H*L 137 144X #USE 031 145X #USE 144X #U							
276 124X CMP M 125X RNE D 125X RNE D 125X RNE D 125X RNE D 127X INX H D 127X INX H D 127X INX H D 127X INX H D 129X JNZ \$COMP \$ \$COMP \$ \$129X & RET \$ \$COMP \$ \$130X ** * * * * * * * * * * * * * * * * *	276 124X CMP M 300 125X RNE 300 125X INX D 043 1274X INX H 015 128X INX H 015 128X INX H 016 C 302 060 030 129X INZ \$COMP 311 133X ** * \$DADA - FERFORM (H,L) = 135X ** * * \$DADA - FERFORE VAL 135X ** * * \$CC SET IF OVERF 136X ** * CC SET IF OVERF 136X ** * CC SET IF OVERF 137	276 124X CMP M 300 125X RNE 300 125X INX D 043 127X INX H 015 128X DCR C 302 060 030 129X DCR C 311 133X ** *DADA - PERFORM (H,L) = 134X * * ENTRY (H,L) = BEFORE VAL 135X * EXIT (H,L) = HFFORE VAL 135X * EXIT (H,L) = (H,L) + 135X * EXIT (H,L) = 1	030.050				LDAX		
300 125X RNE D 125X 10X D 129X DCR C 130X ** *DADA - FERFORM (H,L) = BEFORE V 135X ** ENTRY (H,L) = BEFORE V 135X ** EXIT (H,L) = (H,L) + 135X ** DADA PUSH D 140X 141X DAD D 145X D	300 125X RNE D 125X 10X D 125X 10X D 125X 10X D 120X 10X D 120X DCR C 120X DCR C 120X DCR C 120X DCR C 120X RET SET C 130X RET SET C (H,L) = BEFORE V 134X R EXIT (H,L) = BEFORE V 135X R EXIT (H,L) = (H,L) + 136X R EXIT (H,L) = (H,L) +	300 125X RNE D 125X 1NX D 127X 1NX H 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	30.061				CMP		
023 126X INX D 043 127X INX H 045 127X INX H 015 128X INX C 302 060 030 129X RET 311 130X RET 133X ** \$DADA - FERFORM (H,L) = 134X * (A) = BEFORE V 135X * ENTRY (H,L) + 135X * EXIT (H,L) + 135X * EXIT (H,L) + 135X * EXIT (H,L) = (H,L) + 135X * USES F,H,L 140X * MOV E,A 141X MOV E,A 031 143X MOV E,A 031 145X MOV E,A 311 145X RET	023 126X INX D 043 127X INX H 045 129X DIR C 302 060 030 129X DIR C 311 130X RET SECONF 134X * * * * * * * * * * * * * * * * * * *	023 126X INX D 043 127X INX H 045 129X DIRX H 302 060 030 129X RET 311 130X RET 3133X ** \$DADA - FERFORM (H,L) = 134X * ENTRY (H,L) = BEFORE VAL 135X ** ENTRY (H,L) = BEFORE VAL 135X ** ENTRY (H,L) = HFFORE VAL 135X ** ENTRY (H,L) = HFFORE VAL 135X ** ENTRY (H,L) = HFFORE VAL 135X ** ENTRY (H,L) = H,L) + 135X ** ENTRY (H,L) = 135X	30.062	•		125X	E SE		
015 128X INX H 015 128X DINZ \$COMP 311 130X RET \$ECOMP 311 130X RET \$ECOMP 133X * \$DADA - FERFORM (H,L) = REFORE V 135X * ENTRY (H,L) = REFORE V 135X * EXIT (H,L) = REFORE V 135X * EXIT (H,L) = (H,L) + 135X * EXIT (H,L) = (H,L) + 137X * EXIT (H,L) = (H,L) + 137X * EXIT (H,L) = (H,L) + 137X * DADA PUSH D D D D D D D D D D D D D D D D D D D	015 128X INX H 015 128X DIX \$COMP 311 130X RET \$COMP 311 130X RET \$ECOMP 133X ** \$DADA - FERFORM (H,L) = BEFORE V 135X ** ENTRY (H,L) = BEFORE V 135X ** EXIT (H,L) = HFORE V 135X ** EXIT (H,L) = HFORE V 135X ** EXIT (H,L) = HFORE V 135X ** EXIT (H,L) = (H,L) + 1 135X ** USES F,H,L 141X ** MOV E,A 026 000 144X MOV E,A 031 145X DAD D 321 146X PQP D	043 127X INX H 302 060 030 129X JNZ \$COMP 311 133X ** \$DADA - FERFORM (H,L) = 134X * \$DADA - FERFORM (H,L) = 135X * \$ENTRY (H,L) = BEFORE VAL 135X * \$ENTRY (H,L) = H+L) + 135X * \$USES F,H,L 137	30.063			126X	XXI	-	
302 060 030 129X JNZ \$COMP 311 130X RET 133X ** *DADA - FERFORM (H,L) = 134X * 135X * ENTRY (H,L) = BEFORE V 135X * EXIT (H,L) = REFORE V 135X * EXIT (H,L) = (H,L) + 135X * USES F,H,L 137X * DADA PUSH D 137	302 060 030 129X JNZ \$COMP 311 130X RET ET 133X ** *DADA - FERFORM (H,L) = 134X * (A) = BEFORE VA 135X * EXIT (H,L) = BEFORE VA 135X * EXIT (H,L) = HFORE VA 135X * EXIT (H,L) = HFORE VA 135X * EXIT (H,L) = HFORE VA 135X * EXIT (H,L) = HPORE 139X * USES F,H,L 141X * DADA PUSH D 141X * MUI D,O 026 000 144X MUI D,O 031 145X DADA D 321 146X PQP D 3311 147X RET	302 060 030 129X JNZ \$COMP 311 130X RET 133X ** *DADA - FERFORM (H,L) = 135X * 135X * ENIT (H,L) = BEFORE VH 135X * EXIT (H,L) = REFORE VH 135X * EXIT (H,L) = HFORE VH 135X * EXIT (H,L) = HFORE VH 135X * USES F,H,L 139X * USES F,H,L 140X 137 141X MOV E,A 026 000 144X MVI D,0 031 145X DAD D 145X PDP D 321 145X RET	30.064			12/X 128Y	XXI		
311 130X RET 133X ** * *DADA - FERFORM (H,L) = 134X *	311 130X RET 133X ** * *DADA - FERFORM (H,L) = 135X * * (A) - E BEFORE VALUED	311 130X RET 133X ** * \$DADA - FERFORM (H,L) = 135X * * (A) = BEFORE VAL 135X * (A) = BEFORE VAL 135X * (A,L) = REFORE VAL 135X * (A,L) = (H,L) + (A,L) = (H,L) = (H,L) + (A,L) = (H,L) = (H,	30.066	:	060 030	129X	JNZ		
133X ** *DADA - FERFORM (H,L) = (H,L) + 134X * 135X * ENTRY (H,L) = BEFORE VALUE 135X * EXIT (H,L) = HEFORE VALUE 135X * EXIT (H,L) = (H,L) + (0,A) 135X * EXIT (H,L) = (H,L) + (0,A) 135X * USES F,H,L 140X 141X 141X 141X 141X 141X 141X 141X	133X ** *DADA - FERFORM (H,L) = (H,L) + 134X * 135X * ENTRY (H,L) = BEFORE VALUE 135X * EXIT (H,L) = BEFORE VALUE 135X * EXIT (H,L) = (H,L) + (0,A) 136X * C. SEI IF QVERFLOM 139X * USES F,H,L 140X 141X 141X 142X *DADA PUSH D 137 143X MOV E,A 026 000 144X MOV E,A 026 000 144X MOV E,A 031 145X DAD D 3321 146X RET EXIT	133X ** *DADA - FERFORM (H,L) = (H,L) + 134X * 135X * ENTRY (H,L) = BEFORE VALUE 135X * EXIT (H,L) = BEFORE VALUE 135X * EXIT (H,L) + (O,A) 138X * USES F,H,L 139X * USES F,H,L 140X 137	30.071	:		130X	RET		
133X ** * *DADA - FERFORM (H,L) = (H,L) + 134X *	133X ** * * \$DADA - FERFORM (H,L) = (H,L) + (134X * 135X * ENTRY (H,L) = BEFORE VALUE 135X * EXIT (H,L) = BEFORE VALUE 135X * EXIT (H,L) = (H,L) + (0,A) 138X * (2, SET IF OVERFLOW 138X * (2, SET IF OVERFLOW 138X * (4,L) = (H,L) + (0,A) + (0,A) 138X * (141X HOV E,A) 141X HOV E,A	133X ** * *DADA - FERFORM (H,L) = (H,L) + 134X * 135X * ENTRY (H,L) = BEFORE VALUE 136X * (A) = BEFORE VALUE 136X * EXIT (H,L) + (O.A) 137X * EXIT (H,L) + (O.A) 137X * EXIT (H,L) + (O.A) 137X * DADA PUSH D 137X * DADA D 137X * DADA D 141X							
134X *	134X *	135X * ENTRY (H,L) = BEFORE VALUE 135X * ENTRY (H,L) = BEFORE VALUE 135X * EXIT (H,L) + (0,A) 135X * LX (H,L) + (0,A) 135X * USES F,H,L 141X USES F,H,L 141X HOV E,A 026 000 144X HOV E,A 031 145X BAD D 15X BAD				* X22	<u> </u>	+ (1.1) #	
135X * ENTRY 136X * 136X * 137X * EXIT 137 * 140X 140X 140X 140X 141X * DADA 137 141X * DADA 137 143X * HOU 137 143X * DADA 137 144X * HOU 031 145X * DADA 311 145X * CET	135X # ENTRY 136X # 136X # EXIT 137 # EXIT 137 140X 140X 140X 140X 141X 141X 137 141X HDU 026 000 143X HDU 031 145X HDU 031 145X HDU 031 146X HDU 321 146X HDU 321 146X HDU 321 146X HDU 331 146X HDU 331 147X HDU 331 147X HDU 331 147X HDU 331 147X HDU	135X # ENTRY 136X # 136X # 137X # EXIT 138X # 139X # USES 140X 140X 140X 141X * DADA 141X * DADA 143X MOV 143X MOV 143X MOV 143X MOV 144X MOV 145X DAD 321 145X PRET 331 145X RET				X45		. :	
136X * 136X * 136X * 137X * EXIT 137 * USES 149X * USES 141X * USES 141X * USES 141X * USE 141X * USE 141X * USE 151X * USE 141X * USE 151X * USE 144X * USE 151X * U	136X * 136X * 136X * 137X * EXIT 137 * USES 149X * USES 141X * USES 141X * USE 144X * US	136X * 136X * 137X * EXIT 137 * USES 143X * USES 140X 141X * USES 141X * USE				35X	ENTRY	(H)L) = BEFORE VALUE	
139X * 139X * 139X * 139X * 139X * 140X 140X 141X 137 141X 141X 137 141X 141X 141X 145X 146X 147X 147X 147X 147X 147X 147X 147X 147	139X # USES 139X # USES 140X 140X 141X 141X 137 142X #DADA 137 143X #DAD 145X #DAD 145X #DAD 145X #DAD 145X #DAD 146X #DAD 146X #DAD 146X #DAD 146X #DAD 147X #DAD 147X #DAD	139X # USES 139X # USES 140X 140X 140X 140X 140X 150 140X 150 140X 160 160 160 160 160 170 170 170 170 170 170 170 17				36X	1177	(A) = BELUKE VALUE	
139X * USES F,H,L 140X 140X 140X 151 141X * DADA PUSH D 137 143X H DU E,A 137 145X H DU D,O 031 145X PRF DAD 321 146X RET	139X * USES F,H,L 140X 140X 121 141X * DADA PUSH D 137 143X HUU E,A 137 144X HUI D,O 031 145X DAD D 321 146X RET	139X * USES F;H;L 140X 140X 140X 141X 141X 142X *DADA PUSH D 137 143X HDV E;A 026 000 144X HUI D;0 031 145X PQP D 321 145X PQP D 321 147X RET				* > > > 1		(112) - (112) - (011)	
325 140X 141X \$DADA PUSH D 137 143X MUV E,A 036 000 145X MVD D,0 031 145X PDP D 321 146X RET	325 140X 141X \$DADA PUSH D 137 143X MOV E,A 031 144X MOV E,A 031 145X DAD D 321 146X PRF D 311 147X RET	140X 141X 141X 141X 141X 141X 140V 144X 140V 141X				1 70X *	SESI	F. L. 1	
325 141X \$DADA PUSH D 137 143X MOV E,A 026 000 144X MOV D,O 031 145X DAD D 321 146X PQP D 311 147X RET	325 141X \$DADA PUSH D 137 143X MDV E,A 026 000 144X MUI D,O 031 145X MUI D,O 321 146X PQP D 321 147X RET	325 141X \$DADA PUSH D 137 143X MDV E'A 026 000 144X MUI D'O 031 145X DAD D 321 146X PRP D 331 147X RET				140X			
325 142X \$DADA PUSH D 137 143X MDV E,A 026 000 145X MVI D,O 031 145X PDA D 321 146X PDA D 311 147X RET	325 142X \$DADA PUSH D 137 143X MDV E,A 026 000 144X MVI D,O 031 145X DAD D 321 146X PDP D 311 147X RET	325 142X \$DADA PUSH D 137 143X HDV E'A 026 000 144X HVI D'O 031 145X DAD D 321 146X PRP D 3311 147X RET				41×			
137 143X MOV E'A 026 000 144X MVI D'O 031 145X DAD D 321 145X PRF D 311 147X RET	137 143X MDV E,A 026 000 144X MVI D,O 031 145X DAD D 321 146X PBP D 311 147X RET	137 143X MDV E,A 026 000 144X MVI D,Q 031 145X DAD D 321 146X PDP D 3311 147X RET	30.072			42X \$DAD		4	
026 000 144X MUI D.O. 031 145X DAD D 321 146X PDP D 311 147X RET	026 000 144X MUI D.O. 031 145X DAD D 321 146X PDF D 311 147X RET	026 000 144X MUI D70 031 145X DAD D 321 145X RET 311 147X RET	30.073	:		43X	MOV	E,A	
031 145X DAD D 321 146X PDP D 311 147X RET	031 145X DAD D 321 146X PDP D 311 147X RET	031 145X DAD D 321 146X POP D 311 147X RET	30.074		8	144X	¥01	D,0	
321 146X PDP D 311 147X RET	321 146X PDP D 311 147X RET	321 146X PDP D 311 147X RET	30.076			145X	DAD	a	
311 147X RET	311 147X RET	311 147X RET	30.077	:		146X	POP		
			30,100			147X	RET		



				:		
		150X ** 151X *	*DADA.	: 1	1.0	
		52X	ENTRY	NONE		
			EXIT	(HC) = (HC) + (OA)	۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰۰	
			USES	AiriHil		
	14 C	,				
3 5	700	15/X \$UAUA.	HILL HILL HILL HILL HILL HILL HILL HILL			
y M	320	138X	ב ב ב ב ב ב ב ב ב ב ב ב ב ב ב ב ב ב ב	L,A		
30.104	044	160X	N N			
5	311	161X	RET			
			*DU66	- UNSIGNED 16 / 16	14 ntotne.	
		166X *	(HL) =	(BC)/(DE)		
			ENTRY	(BC). (DE) PRESET		
			EXIT	= RES		
				(DE) = REMAINDER		
		171X *	USES	ALL		
30	172	174X \$DU66	¥0¢	A, D	TWOS COMPLEMENT (DE)	
	127	X6/1	E C	* 6		
30.1	173	177X	202	A, E		
77	057	178X	C.W.	· .		
30.1	023	180X	> > 2	T.		
30.1	172	181X	2	. A		
30.1		182X	ORA			
7:-	312 205 030	183X	JZ	DU665	IF DIVIDE BY O	
30.1	/67	185X 185X		∢		
		186X *	SHIFT	(DE) LEFT UNTIL:		
		187X *	2 / 50	3		
		189× *	2) OUFBEI OU	/ BL		
		190X		kat sa		
030.123	4	191X DU661	H00	Hrb		
030.124	io -	192X	200	١,٤		
030.126	t Ci	194X	, CON	101662	TS TOO I ARRE	
030.131	74	195X	INE	¥	COUNT SHIFT	
030.132	142	196X	NO.	Q.		
030.134	כווי כ	198X	DAD	7.7E		
030.135	53	199X	ХСНС		$(DE) = (DE) \times 2$	
030,136	M	200X	<u>ا</u>	DU661	IF NOT OVERFLOW	
		202X *	(DE) 0	OVERFLOWED, PUT IT	BACK.	



030.141 353 203X XCHG 030.142 075 205X EABDY TC 200X READY TC 200X READY TC 200X H 030.143 140 209X DU662 HDV 030.145 001 000 000 211X HDV 030.150 365 223 213X DU663 FUSH 030.157 157 213X DU663 FUSH 030.157 157 213X DU663 FUSH 030.157 157 213X DU664 HDV 030.160 174 223 213X DU664 HDV 030.161 177 221X DU664 HDV 030.162 117 222X HDV 030.164 027 222X HDV 030.165 117 222X HDV 030.167 127 223X HDV 030.167 127 223X HDV 030.171 067 222X HDV 030.172 127 223X HDV 030.174 127 223X HDV 030.175 127 223X HDV 030.177 067 223X HDV 030.177 127 233X HDV 030.177 127 234X HDV 030.201 351 233X HDV 030.201 351 243X HDV 030.201 353 353 243X HDV 030.201 354 353 243X HDV 030.201 354 243X HDV	A REMOVE EXTRA COUNT TO START SUBTRACTING. (A) = LOOP COUNT H,B L,C B,O B,O CBC) = RESULT FSW D DU664 IF SUBTRACT OK A,L A,L	
42 075 205X * READY 206X DUG62 MOU 44 151 210X DUG62 MOU 44 151 210X DUG62 MOU 209X DUG62 MOU 201X DUG63 PUSH 210X DUG63 PUSH 210X DUG63 PUSH 210X DUG63 DUG64 DUG9 DUG63 DUG64 DUG9 DUG63 DUG64 DUG9 DUG63 DUG64 DUG9 DUG64 DU	SUBTRACTING. (A) (H,L) = (BC) = F SAVE (A) IF SUBTE ADD BACK	
43 140 209X # READY 200X 210X # READY 200X 200X 200X 200X 200X 200X 200X 200	SUBTRACTING. (A) (H,L) = (BC) = F SAVE (A) IF SUBTE ADD BACK	
43 140 200X DU662 MOV 44 151 210X MOV 45 001 000 000 211X MOV 45 001 000 000 211X MOV 55 352 352 163 030 214X MOV 56 223 215X MOV 57 157 215X MOV 58 171 217 221X MOV 58 171 221X MOV 58 171 221X MOV 58 171 222X MOV 58 172 223X MOV 58 173 225X MOV 58 173 2	(BC) = R SAVE (A) IF SUBTE ADD BACK	
45 001 000 000 211X LXI 50 345 1031 213X Du663 Pu9H 51 332 143 030 215X Du663 Pu9H 52 332 143 030 215X Du663 Pu9H 53 125 127 H0V 60 174 218X H0V 61 174 218X H0V 62 1232 220X H0V 64 027 222X H0V 65 170 222X H0V 66 170 222X H0V 67 172 223X H0V 68 170 223X H0V 69 174 223X H0V 69 174 223X H0V 60 174 223X H0V 60 177 067 225X H0V 61 127 225X H0V 62 172 225X H0V 63 172 225X H0V 64 027 225X H0V 65 170 225X H0V 67 225X		
50 365 212X pu663 FUSH PuSH 51 031 214X DAD 52 175 215X MOV 56 223 214X JC 56 223 215X MOV 160 174 215X MOV 161 232 219X MOV 162 147 229X MOV 163 171 221X MOV 164 027 222X MOV 165 117 222X MOV 165 117 222X MOV 165 117 224X MOV 166 170 224X MOV 170 224X MOV 170 224X MOV 170 224X MOV 171 222X RAL 172 223X MOV 173 037 234X MOV 174 037		
52 332 163 030 214X JC 55 175 215X HOV 16 223 215X HOV 160 174 219X HOV 161 232 219X HOV 164 232 219X HOV 164 232 219X HOV 164 232 222X HOV 164 027 222X HOV 164 027 222X HOV 165 170 224X HOV 166 170 224X HOV 167 027 222X RAL 170 107 224X HOV 170 107 224X HOV 170 224X HOV 171 067 233X RAR 173 235X HOV 174 127 235X HOV 175 037 235X		
175 175		
157 157 217X HOV HOV HOV HOV HOV 154 218X HOV HO	A	
16. 17.4 SBB 16. 147 S220X HOV 16. 147 222X HOV 16. 027 222X HOV 16. 027 223X HOV 17. 027 224X HOV 17. 027 225X HOV 17. 027 225X HOV 17. 107 224X HOV 17. 107 225X HOV 17. 107 235X HOV		
62 147 220X MOV 163 171 221X DU664 MOV 164 027 222X RAL 165 117 222X MOV 166 170 222X MOV 167 027 222X MOV 167 027 222X RAL 170 027 222X RAL 170 027 222X RAL 171 067 222X RAL 172 222X RAL 173 037 233X MOV 176 037 234X MOV 176 037 235X MOV 241X MOV 241	Q	
163 171 221X DU664 MOV 164 027 222X RAL 165 177 222X MOV 167 027 225X MOV 167 027 225X MOV 170 107 226X MOV 171 067 226X MOV 172 172 229X RAR 173 037 233X MOV 174 127 233X MOV 175 172 233X MOV 176 037 235X RAR 177 137 234X MOV 176 037 235X MOV 177 137 234X MOV 176 037 235X MOV 177 137 234X MOV 178 037 235X MOV 179 127 234X MOV 170 341 235X MOV 170 351 243X MOV 170 351 243X MOV 170 351 243X MOV 170 351 243X MOV	н, А	
165 177 223X HOV 166 170 224X HOV 167 027 225X RAL 170 107 226X HOV 171 067 229X STC 172 172 230X STC 173 037 233X HOV 174 127 233X HOV 175 173 234X HOV 176 037 234X HOV 176 037 234X HOV 177 137 234X HOV 176 037 235X HOV 177 137 234X HOV 177 137 234X HOV 176 037 235X HOV 177 137 137 137 137 137 137 137 137 137	A,C	
166 170 224X MOV 167 027 225X RAL 170 107 228X MUV 228X RIGHT 228X RIGHT 228X RIGHT 229X STC 172 172 233X MOV 174 127 233X MOV 175 173 234X MOV 176 037 234X MOV 176 037 234X MOV 177 137 234X MOV 177 137 235X MOV 200 361 235X MOV 201 361 235X MOV 202 362 150 030 239X DE 204 140 241X MOV 205 353 240X DU665 XCHG 206 140 241X MOV 207 151 243X RET	C,A	
167 027 225X RAL 170 107 226X MQU 228X RIGHT 228X RIGHT 229X STC 229X STC 172 172 233X HOU 174 127 233X HOU 175 173 233X HOU 176 037 233X HOU 176 037 235X HOU 177 137 235X HOU 200 361 235X HOU 201 361 235X HOU 202 362 150 030 239X DGR 205 353 240X DU665 XCHG 206 140 241X HOU 207 151 243X RET	A, B	
170 10.7 2228X	<	
228X * RIGHT 229X 529X 5229X 5229X 5229X 5229X 530X 530X 531X 530X 531X 60V 730X 740X 750X 750X 750X 750X 750X 750X 750X 75		
71 967 2297 72 172 231X 73 037 232X 74 127 233X 75 173 234X 76 037 234X 77 137 234X 70 037 235X 80 341 237X 80 341 237X 80 341 237X 80 341 237X 80 341 237X 80 341 237X 80 341 80 80 237X 80 341 80 80 80 80 80 80 80 80 80 80 80 80 80	SHFT (DE)	
72 172 231X 73 037 232X 74 127 233X 75 173 233X 76 037 235X 77 137 235X 77 137 235X 800 361 235X 801 975 235X 802 361 235X 803 351 235X 804 140 241X 807 151 243X		
73 037 232X 74 127 234X 76 037 235X 77 137 236X 200 341 237X 201 075 235X 202 362 150 030 239X 202 353 240X pu665 206 140 241X 207 151 243X	A, D	
75 173 76 037 235X 77 137 235X 200 341 237X 201 075 237X 202 342 150 030 239X 205 353 240X pu665 206 140 241X 207 151 243X	Π.Δ	
76 037 235X 77 137 236X 200 361 237X 201 075 237X 205 353 239X 205 353 240X DU665 206 140 241X 207 151 243X	A,E	
200 351 233X 201 351 238X 202 362 150 030 239X 205 353 240X DU665 206 140 241X 207 151 243X 210 311 243X	4	
201 075 202 362 150 030 239X 205 353 240X pu665 206 140 241X 207 151 242X 210 311 243X	PSM	
202 362 150 030 239X 205 353 240X pu665 206 140 241X 207 151 242X 210 311 243X		
206 140 241X 207 151 242X 210 311 243X	53 IF NO! = (1961)	
207, 151 210 311 243X	" (量)	
	.L.A.C.	
246X ** \$HLIHL	- LOAD HL INDIRECT THROUGH HL.	
	((H))	
**	202	
	NONE	
*	Arthol	
253X 211 124 254X \$HITHI	X	
212 043 255X	H	

82	TTT A	THKIT.
	HEA	THAT

** *CDEHL	- COMPARE (DE) TO (HL)
* *CDEHL	COMPARES (DE) TO (HL) FOR EQUALITY:
* ENTRY NONE	V 117 - V 207 31 130
* EXII 2	(DE) = (HL)
268X 269X	
173 270X \$CDEHL 255 271X	
300 272X RNZ 172 273X MDU	IF DIFFERENT
030.222 254 274X XRA H 030.223 311 275X RET	
	(HL).
	TWO'S COMPLEMENT
282X # ENTRY NONE	
* USES	
286X	
174 Z8/X \$CHL 057 288X	
147. 289X.	
030.227 175 290X MDV A,L 030.230 057 291X CMA	
157 292X MOV	
311 294X	



- INDEXED LOAD. LOADS DE WITH THE TWO BYTES AT (HL)+DISPLACY ACTS AS AN INDEXED FULL WORD LOAD. ((RET)) = DISPLACHENT ((HL) = TABLE ADDRESS TO (REF+2) A,F,D.E B,M (HL) = RET, ((SP)) = TBL E,M H ((SP)) = RET, ((HL) = TBL (HL) = TBL (HL	
2595	
300X * THIS ACTS AS AN INDEXED FULL WORD LOAD. 302X * (NE) = ((HL) + DSFLACKHNT) 303X * (NE) = ((HL) + DSFLACKHNT) 304X * (NE) = ((HL) = FABLE ADDRESS 304X * (NE) = ((HL) = FABLE ADDRESS) 304X * (NE) = ((HL) = ((HL) = TBL ADDRESS) 304X * (NE) = ((HL) = ((HL) = TBL ADDRESS) 304X * (NE) = ((HL)	LACMENT
303 x x (be) = ((HL) + bSPLACEHENT) 304 x x (ReT) = DISPLACENTY (FULL WORD) 305 x x EXTT (H) = DISPLACENTY (FULL WORD) 305 x x EXTT (H) = TABLE ADDRESS 306 x x (HL) = TABLE ADDRESS 306 x x (HL) = TABLE ADDRESS 310 x (HL) = TABLE ADDRESS 310 x (HL) = TABLE ADDRESS 311 x (HL) = TABLE ADDRESS 312 x (HL) = TABLE ADDRESS 313 x (HL) = TABLE ADDRESS 314 x (HL) = TABLE ADDRESS 315 x (HL) = TABLE ADDRESS 316 x (HL) = TABLE ADDRESS 317 x (HL) = TABLE ADDRESS 318 x (HL) = TABLE ADDRESS 318 x (HL) = TABLE ADDRESS 318 x (HL) = TABLE ADDRESS 319 x (HL) = TABLE ADDRESS 310 x (HL) = TABLE ADDRESS 310 x (HL) = TABLE ADDRESS 311 x (HL) = TABLE 322 x (HL) = TABLE 323 x (HL) = TABLE 324 x (HL) = TABLE 325 x (HL) = TABLE 326 x (HL) = TABLE 327 x	
303X ** (DE) = ((HL) + DSPLACEMENT (FULL WORD) 305X ** (HL) = TABLE ADDRESS 306X ** (SE) = DSPLACEMENT (FULL WORD) 310X ** (SE) = TABLE ADDRESS 310X ** (HL) = TABLE ADDRESS 310X ** (HL) = TABLE ADDRESS 311X **IMDL XTHL (HL) = TABLE ADDRESS 311X **IMDL HOVE HOVE DATA (HL) = TABLE ADDRESS 311X **IMDL HOVE IS TO A LOWER ADDRESS, THE BYTES ARE 323X ** FIRST TO LAST; (DE) = VALUE, (HL) = TABLE ADDRESS 311X **IT THE MOVE IS TO A LOWER ADDRESS, THE BYTES ARE 323X ** THY IS DONE SO THAT AN OVERLAPED MOVE WILL NOT 325X ** (HL) = TABLE ADDRESS OF NEXT FROM BYTE 344X ** (HL) = TOD TABLE ADDRESS OF NEXT FROM BYTE 344X ** (HL) = TOD TABLE ADDRESS OF NEXT FROM BYTE 344X ** (HL) = ADDRESS OF NEXT FROM BYTE 344X ** (HL) = ADDRESS OF NEXT FROM BYTE 344X ** (HL) = ADDRESS OF NEXT FROM BYTE 344X ** (HL) = ADDRESS OF NEXT FROM BYTE 344X ** (HL) = ADDRESS OF NEXT FROM BYTE	
305X ** EVITY ((RET) = DISPLACKENT (FULL WORD) 306X ** LEXIT TO (RET+2) 306X ** LEXIT TO (RET+2) 306X ** LOSES AFF.D.E 310X 310X 310X 310X 310X 311X *1MDL XTHL 10A 31X ** LINX H 10A 31X ** CHOR H 10	
306X * EXT	
307 X EXIT TO (RET+2) 308 EXIT TO (RET+2) 308 USES AFFDE 308 USES AFFDE 308 USES AFFDE 311	
343 311X *INDL XTHL E.M (HL) = RET, ((SP)) = TBL 136 311X *INDL XTHL E.M (HL) = RET, ((SP)) = TBL 136 312X MOV E.M (OE) = DISPLACEMENT 315X MOV B.M (OE) = TBL ADDRESS (HL) = TABL ADDRE	
310 × 310 × 310 × 310 × 310 × 310 × 310 × 310 × 313 × 310 × 313 × 313 × 313 × 313 × 313 × 313 × 313 × 313 × 313 × 314 × 314 × 314 × 315 × 315 × 316 × 316 × 316 × 317 × 318 ×	
343 3311X \$1NUL XTHL 1346 312X	o du cade x
126 315X INX H (DE) = DISPLACEMENT 126 314X MOV D.H (DE) = DISPLACEMENT 126 315X INX H (CSP); = RET, (HL) = TBL 315X XTHL (DE) = TBL ADDRESS, (HL) = TBL 322X MOV L, H (CHL) = (CHL); (HL) = TBL 322X MOV L, H (CHL) = (CHL); (HL) = TBL 322X MOV L, H (CD) = (CHL); (HL) = TBL 322X MOV L, H (CD) = (CDN) = VALUE, (HL) = TBL 322X MOV L, H (CD) = (CDN) = VALUE, (HL) = TBL 322X MOV L, H (DE) = (CDN) = (CDN) = (CC) = (CDN) = (CC) = (CDN) = (CC) =	BL HUDRESS
126 314X HOV D.H (DE) = DISPLACEMENT 315X INX H (SP) = KET; (HL) = TBL ADDRESS; (HL) 317X XTHL HOV A.H (HL) = TRL ADDRESS; (HL) 323X HOV H.H (HL) = TARGET ADDRESS; (HL) 321X HOV H.H (HL) = (HL) (HL) TARL ADDRESS; (HL) ADDRESS; (HL) ADDRESS; (HL) ADDRESS; (HL) = TARL ADDRESS; (HL) ADDRESS; (HL) = TARL ADDRESS; (HL) = A	
315X 315X 315X 315X 316X 317X 317X 317X 318X 318X 328X 328X 321X 322X 322X 322X 322X 322X 322X 322	
241 343 317X XTHL ((SP)) = RET, (HL) = TBL ADDRESS, (HL) 242 353 318X XCHG	
242 353 318X XCHG	TRU ADDRESS
243 031 319X	
244 176 320X MOV A'M 5 4 5 221	
245 043 321X INX H 246 146 322X MOU L;A 322X MOU L;A 323X MOU L;A 324X XMO L;A 325X XMO L;A 325X XMO - MOVE DATA 329X X 330X X 4 MOVE MOVES A BLOCK 331X X 1F THE MOVE IS TO A 333X X 1F THE MOVE IS TO A 334X X 1F THE MOVED A 344X X (HL) = A ADDR 344X X (HL) = A ADDR 344X X (HL) = A ADDR 344X X (CCLEAK 345X	
250 146 522X HOV L'A 253	
250 353 XCHO LTH 2250 353 XCHO LTH 2250 353 324X KET	
251 311 325X RET 328X ** ** ** ** ** ** ** ** ** ** ** ** **	TABLE ADDRESS
** ** ** ** ** ** ** ** ** ** ** ** **	
# #MOVE MOVES A BLOCK FIRST TO LAST. FIRST TO LAST. FIRST TO LAST. FIRST TO LAST. FIRST. F	
## #MOVE MOVES A BLOCK IF THE MOVE IS TO A FIRST TO LAST. THIS IS DONE SO THA	
FIRST TO LAST:	ADDRESS.
## IF THE MOVE IS TO A LAST TO FIRST. ## THIS IS DONE SO THA ## ENTRY (BC) = COUN ## EXIT MOVED ## EXIT MOVED ## (ML) = ADDR ## (ML) = ADDR ## (ML) = ADDR	AKE MUVEU FKUM
## IF THE MOVE IS TO A LAST TO FIRST.	
# LAST TO FIRST. # THIS IS DONE SO THA # ENTRY (BC) = COUN # EXIT (DE) = FROM # EXIT (DE) = ADDR # (HL) = ADDR # (HL) = ADDR	ARE MOVED FROM

****	OT /51651 5/
* ENTRY (BC) = COUNT (MUST BE < 3276 * (BE) = FROM * (HL) = 10 * EXIT HOUSE * (DE) = ADDRESS OF NEXT FROM * (HL) = ADDRESS OF NEXT *TO*	• • • • • • • • • • • • • • • • • • • •
* (DE) = FROM * (HL) = TO * EXIT MOVED * (DE) = ADDRESS OF NEXT FROM * (HL) = ADDRESS OF NEXT *TO*	
* (HL) = TO * EXIT MOVED * (DE) = ADDRESS OF NEXT FROM * (H.) = ADDRESS OF NEXT *TO* * 'C' CLEAR	
# EXIT MOVED # (DE) = ADDRESS OF NEXT FROM # (HL) = ADDRESS OF NEXT *TO* * 'C' CLEAR	
* (DE) = ADDRESS OF NEXT FROM * (HL) = ADDRESS OF NEXT *TO* * 'C' CLEAR	
* CC CLEAR	
× ·	
346X X	
348X	
030,252 349X \$MOVE EQU *	



030,252	170 261	350X 351X		MOV ORA	A,B		
030.254	310	352X		RZ		NONE TO MOVE	
030.256	223	354X		SUB		CUMPARE XFRUMX 10 XTQX	
030,257	174	355X		700	AtH		
030.260	232 332 311 030	356X 357X		SBB	D MOV2	IS MOVE DOWN (TO LOWER ADDRESSES)	
		358X	.	IS MOVE		SSES)	
030.264	013	360X 361X		DCX	~		
030.265	011	362X		DAD	, eq. 1		
030.267	353	364X		XCHG	c	SHVE *-U* LIMI	
030.270	011 345	365X		DAD	aa 3	(HL) # KFROM* LWA	
				L .	c	*FRUM*	
030.272	176 022		M0V1	MOV	E.	MOVE BYTE	
030.274	033	370X		DCX	0		
030,275	053	371X		DCX	T 6	INCREMENT *FROM* ADDRESS	
030.277	170	373X		¥6	A,B		
030,300	Ç	374X		ANA	Œ	: 1	
030.304	321	376X		704	MUV1 D	MORE TO GO	
030.305	341	377X		ő	· I		
030.306	023	378X		XXI	A 3		
030.310	311	380X		RET	· · · · · · · · · · · · · · · · · · ·	DONE	
			*	TS MOUF	HOUN (TO LOUER ANDRESSES)	o Antibereeres	
:				2		A HUDRESSES!	
030.311	032		H0V2	LDAX	×	MOVE BYTE	
030.313	043	386X		INX	T.	INCREMENT AFROMA	
030.315	013	388X		אַעַל	÷ α	DECEMBER ALUX	
030,316	170	389X		MDV	A, B	:	
030.320	302 311 030	390X		ORA IN 7	K C	בייים דכיי פד	
030,323		392X		RET	7	DONE	
		395X	*	\$MU10	HILTIPLY UNSIG	- MULTIPLY UNSIGNED 14 RIT DUANTITY BY 10.	
		396X	*	Ċ		AND PROPERTY MATERIAL PROPERTY AND THE PROPERTY OF THE PROPERT	
		398X 398X		= (H)	(DE)#10		
		399X		ENTRY		LIER	
		400X	* 1	EXIT	\mathbf{D}	OK	
		402X			C' SET IF ERRO	ERROR	



404X 404X 405X \$MU10 XCHG 406X \$MU10 DAD H 410X RC 411X RC 41X RC				*	USES	D,E,H,L,F	
12.5 12.5 406X 540110 54011 5402 52.5			404X				
12.5 0.51 40.0	324				ХСНВ		(HL) = MULTIPLIER
132 124 4095 100 10 10 10 10 10 10	325				DAD	I	(HL) = X*2
1330 631 331 051 332 330 412X RC 412X RC 413X RC 414X RC 414X RC 415X RC 41	327		409X		MOV	H.O.	•
333 355 412X RC H (HL) = X#10 334 330 414X RC H (HL) = X#10 335 330 414X RC H (HL) = X#10 335 331 415X RC H (HL) = X#10 335 331 415X RC H (HL) = X#10 335 331 415X RC H (HL) = X#10 420X * * * * * * * * * * * * * * * * * * *	330		410X		200	, L	AXX # / [1]
333 651	332		412X		7. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.	c	**
1334 330 414X FC	333		413X		DAD	I	8*X = (TI)
419X ** \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$	335		4 14 4 4 1 5 X X X X X X X X X X X X X X X X X X		RC DAD RET	a	H :
419X ** \$MUGG - UNSIGNED 16X16 MULTIPLY, 420X * 420							
421X * ENTRY (BC) = MULTIPLICAND 422X * (DE) = MULTIPLIER 422X * (DE) = MULTIPLIER 422X * (SE) IF NOT DVERFLOW 425X * USES ALL 425X * USES ALL 426X 426X * HOSE ALL 426X 426X * HOSE ALL 426X 426X * HOSE ALL			419X	· * 1	*MU66	UNSIGNED 16X16	S MULTIPLY.
422X * (DE) = MULTIPLIER 423X * (SET IF NOT DVERFLOW 423X * USES ALL 424X * USES ALL 425X * US			420X	* *			DAND
425X * CAN TO THE SET IF NOT DVERFLOW 425X * USES ALL 425X * USES ALL 425X * USES ALL 425X * USES ALL 426X * MU66 XRA A 427X * MU66 XRA A 447X * MU66 XRA A			422X	*3			IER
426X # USES ALL 426X 426X 426X 426X 427X 427X 427X 427X 427X 428X 427X 428X 428X 428X 428X 428X 428X 428X 428			4 2 5 X		EVII	, U	
37 257 426X 40 420X 420X 40 365 429X \$MU66 XRA 41 041 000 000 430X LXI H*O (HL) = KESULT 44 170 433X MU641 MU641 (HL) = KESULT 44 170 433X MU641 MU642 IF BIT CLEAR 45 107 433X MU0 A*C IF BIT CLEAR 45 171 435X MU0 A*C IF BIT CLEAR 45 172 435X MU642 IF BIT CLEAR 55 031 435X MU642 IF BIT CLEAR 56 322 3443X MU642 IF MUT OVERFLOW 56 343 443X MU642 IF MUT OVERFLOW 56 344 MU642 IF MUT OVERFLOW 56 345 444X MU643 IS ZERQ1 & MU 57 344 A45X A45X A45X 56 312			425X	*	USES	ALL	
37 257 428 4906 584			426X				
40 365 429X PUSH PSW SAVE DVERFLOW 41 041 000 000 430X LXI H+0 (HL) = RESULT 43 1 1 1	337		428X	₩.	XRA	∢	
44 170 430X LXI H+0 (HL) = RESULT 431X MU641 MUV A+B 432X MU641 MUV A+B 432X MU641 MUV A+B 432X MU641 MUV A+C 433X MU641 MUV A+C 434X MUV A+C 434X MUV A+C 434X MUV A+C 434X MUV A+C 52 32 364 030 435X MU642 MU662 IF NOT OVERFLC 52 32 364 030 440X MU662 MU662 IF NOT OVERFLC 56 32 364 030 440X MU662 MU662 IF NOT OVERFLC 56 32 364 030 440X MU662 MU663 IS ZERQ; AM DC 56 312 005 031 445X MU662 MU663 IS ZERQ; AM DC 56 32 447X MU662 MU663 IS ZERQ; AM DC 56 32 447X MU662 MU663 IS ZERQ; AM DC 56 32 447X MU662 MU663 IS ZERQ; AM DC 56 32 447X MU662 MU663 IS ZERQ; AM DC 56 33 447X MU662 MU663 IS ZERQ; AM DC 56 33 447X MU662 MU663 IS ZERQ; AM DC 56 33 4447X MU662 MU663 MU663 IF NOT OVERFLOW 500 074 453X MU663 MU6	340 365		429X		PUSH	PSW	OVERFLOW
45 037 432X MU661 MOV A'B 45 037 433X MU661 MOV A'B 46 107 433X MOV A'C 43 43 MOV A'C 43 52 364 030 438X MOV C'A 55 031 439X DAD D 55 031 442X DAD D 56 322 364 030 440X DAD D 56 322 364 030 440X DAD D 56 32 364 030 A40X DAD D 57 074 442X DAD D 58 32 364 030 A40X DAD D 68 170 A40X DAD D 68 170 A40X DAD D 69 170 A40X DAD D 69 170 A40X DAD D 60 170 D 60	341 041	8	430X		LXI	И,0	= RESULT
445 037 433X RAR HOU B.A HO B.	44		432X	MU6	¥0¢	A, E	
435X MOV A:C 50 037 435X RAR 551 117 435X HOV C.A 552 322 344 030 439X JNC MU662 IF BIT CLEAR 555 031 439X JND MU662 IF NOT OVERFLOW 561 361 361 441X POP PSW 562 074 441X PUSH PSW 562 074 443X PUSH PSW 563 365 443X PUSH PSW 563 365 444X MU662 RW 563 365 261 A45X AM 564 312 005 031 445X AM 565 261 445X XCHG RW R REE IF MULTIPLIER 371 353 4445X XCHG C C R REE IF MULTIPLIER 372 34	545		433X		K A C C C C	4.	
117			435X		¥0¢	A, C	
117	220		436X		RAR		
155 031 439X DAD D 156 322 364 030 440X JNC MU662 IF NOT DVERFLOW 151 351 441X PSW 152 364 030 440X JNC MU662 IF NOT DVERFLOW 152 365 074 443X PUSH PSW 153 365 4443X MU662 MOV A,B 152 261 445X MU662 MOV A,B 153 365 312 005 031 446X JZ MU663 IS ZERG; AM RONE 151 353 44030 449X XCHG HU663 IF NOT DVERFLOW 157 353 44030 450X JNC MU661 IF NOT DVERFLOW 158 352 344 030 450X JNC MU661 IF NOT DVERFLOW 158 352 344 030 453X PUSH PSW 158 353 353 453X PUSH PSW	351 117 352 322	364 03	437X 438X			C,A MU662	BIT
12 12 12 12 12 13 14 14 15 15 15 15 15 15	555 031		439X		DAD	a	
10. 10. 10. 10. 10. 10. 10. 10. 10. 10.	556 322	364 0	440X		280	MU662	IF NOT OVERFLOW
363 365 443X PUSH PSW 364 170 444X MU662 MDV A:B 365 261 445X MU662 MDV C SEE IF MULTIPLIER 366 312 005 031 446X JZ MU663 IS ZERG; AM RONE 372 051 446X DAP H (P.E.) = (PE)*2 373 353 449X XCHG H (P.E.) = (PE)*2 373 353 449X XCHG H (P.E.) = (PE)*2 374 322 344 030 450X JC MU661 IF NOT DVERFLOW 377 361 FNG DVERFLOW 377 361 FNG DVERFLOW 377 361 FNG DVERFLOW	192		442X		ı K		
364 170 444X MU662 MUV A.B SEE IF MULTIPLIER 365 261 445X ORA C SEE IF MULTIPLIER 365 312 005 031 446X JZ MU663 IS ZERG! AM RONE 371 353 4475X XCHG H (D.E.) = (DE)*2 372 051 449X XCHG H (D.E.) = (DE)*2 373 353 449X XCHG H (D.E.) = (DE)*2 374 353 450X H (D.E.) = (D.E.)*2 374 452X INR A FSW FLOW FSW	363		443X	:	PUSH	PSW	
365 261 SEE IF MULTIPLIER 365 312 005 031 446X JZ MU663 IS ZERG; AM RONE 371 353 4474X XCHG H (D.E.) = (DE)*2 372 051 4494 XCHG H (D.E.) = (DE)*2 373 353 4494 XCHG H (D.E.) = (DE)*2 374 322 344 030 450X JNC MU661 IF NOT DYERFLOW 377 361 451X POP PSW 500 074 452X INR A 501 365 453X PUSH PSW	364		444 X	MUS.	¥0	A, B	
371 353 447X XCHG 372 051 448X DAD H 373 353 449X XCHG 374 353 44030 450X JNC 377 341 451X POP PSW 500 074 452X INR A 501 345 453X PUSH PSW	365 261	5005 03	4 4 U X X X X X X X X X X X X X X X X X		UKA 17	E MI1663	
372 051 448X DAD H 373 353 449X XCHG 374 352 344 030 450X JNC MU661 377 361 451X POP PSW 000 074 452X INR A 001 365 453X PUSH PSW	371 353		447X		XCHG		
373 353 449X XCHG MU661 374 322 344 030 450X JNC POP PSW 377 361 451X POP PSW 000 074 452X INR A 001 365 453X PUSH PSW	372		448X		DAD	I	(p,E) = (pE)*2
377 361 451X POP PSW 000 074 452X INR A 001 365 453X PUSH PSW	373 353	344 03	449X		XCHG	MU661	IF NOT OVERFLOW
000 074 452X INR A 001 365 453X PUSH PSW	377 361		451X		POP	PSW	
\$0.1 C00.1 C00.1 T00.1 T	0.00		452X		INK	A DQU	EI AR NUFREI NU
002 303 344 030 454X JMP MU661	200	34	454X		a E	MU661	PROCESS NEXT BIT
455X	: : : :		455X				



031.005	361 311	456X 457X	X HU663	POP RET	18 S.4.	(A,F) = OVERFLOW STATUS	
		460X 461X	* * *	- 980₩\$	- MULTIPLY 8X16 UNSIGNED.	NSIGNED.	
		462X 463X		#MU86 MUL BIT VALUE	TIPLIES	A 16 BIT VALUE BY A 8	
		464X 465X	* * × ×	ENTRY	: "	~	
		466X 467X		EXIT	(DE) = MULTIPLICAND	CAND	
		468X	•		T IF	NOT OVERFLOW	
		470X 470X	•	USES	HILIHIL		
031.007	041 000 000	:	X \$MU86	LXI	0-4Н	(HL) = RESULT ACCUMULATOR	
031.012	305	473	× >	FUSH	an a	04 17 10 100110 1 Vav	
031.014	267	470	X MUB60	ORA	.	(B) = UVENTLUW FLAG CLEAR CARRY	
031.015	037	476	781W	ů ď			
031.016	322 026 031	-		S S	MU862	IF NOT TO ADD	
031.022	026	031 480X	* ×	JNC	л MU862	NOT OVERFLOW	
031.025	267	481	CYBIN X	INR	26 Q		
031.027	044	031 483)		JZ	MU863	IF DONE	
031.032	353 051	484 485	××	XCHG DAD	I		
031.034		:	×	ХСНВ			
031.035	322 015	031 487X	×	U N	MU861	LOOP IF NOT OVERFLOW	
• •	014	031 489	(×	Ě	MU860		
031.044		490	X X MU863	ORA	æ	SET *Z* FLAG IF NOT OVERFLOW	
031.045	301 311	492X 493X		POP RET	m,		
		496X	** *	*RSTALL	- RESTORE ALL REGISTERS.	EGISTERS.	
		497X	* + × >	- 14 + 00 4			
		490		RETURNS	TO THE PREVIOUS CALLER.	CALLER.	
		200	* ×				
		501X	* * × ×	ENTRY	(SP) = PSW (SP+2) = BC		
		203X			H		
		504X	* * × >		(SP+6) = HL		
		4000			1		



	H17ROM - H17 ROM CODE LISTING COMMON DECKS	TING				#FSTALL HEATH HBASM V1.4 01/20/78 #FSTALL 15:54:19 11-MAY-78	PAGE 11
052 341 510X *FSTALL POP FSW 050 509 509 509 509 509 509 509 509 509		507X	*	USES	ALL		
055 341 510X *KSTALL POP PSH B C C C C C C C C C C C C C C C C C C		208X					
100 001 001 000 001	.047	509X		gUd	Mod		
055 331 512X POP D 055 311 513X POP H 055 311 513X POP H 056 311 514X RET 057 343 516X ** \$SAVALL SAVE ALL THE REGISTERS ON STACE 058 325 522X ENTRY NOWE 058 325 522X (SP+4) = RC 058 522X (SP+4) =	050	511X		90	E 24		
052 341 514X POP H 053 311 514X ** *SAVALL - SAVE ALL REGISTERS ON STACT 516X ** *SAVALL - SAVE ALL REGISTERS ON STACT 510X ** *SAVALL SAVE ALL THE REGISTERS ON STACT 510X ** *SAVALL SAVE ALL THE REGISTERS ON STACT 510X ** *SAVALL SAVE SAVE SAVE SAVE SAVE SAVE SAVE SAVE	051	512X		POP	Ω		
514X * \$\$AVALL - \$AVE ALL REGISTERS ON STACT 517X * \$\$AVALL - \$AVE ALL THE REGISTERS ON TI 517X * \$\$AVALL SAVES ALL THE REGISTERS ON TI 517X * \$\$AVALL SAVES ALL THE REGISTERS ON TI 517X * \$\$AVALL SAVE	052	513X 514X		POP RET	I	•	
518X * \$SAVALL SAVES ALL THE REGISTERS ON TI 518X * \$SAVALL SAVES ALL THE REGISTERS ON TI 520X * EXIT (\$P\$) = PSU 521X * EXIT (\$P\$) = PSU 522X * USES HILL DE 522X * USES HILL DE 522X * USES HILL DE 523X * SAVALL XTHL DE 525X * USAGE B 525X * USAGE B 535X * TUMP - TABLE JUMP. 535X * USAGE B 535X * TUMP - TABLE JUMP. 535X * DUSH PSU 605 351 532X * TUMP - TABLE JUMP. 535X * TUMP - TABLE JUMP. 535X * TUMP - TABLE JUMP. 535X * TUMP - TABLE JUMP. 541X * TUMBER = N-1. 541X * DUSH PSU 541X * DUSH PSU 541X * TUMBER = N-1. 542X * TUMBER = N-1. 543X * USES AIT TO PROCESSOR 544X * USES AIT TO PROCESSOR 644X * USES A		>7 F W					
5187 * *SAVALL SAVES ALL THE REGISTERS ON TI 519X * ENTRY 520X * ENTRY 520X * ENTRY 521X * EXIT 697 = PSW 522X * (SP+4) = BE 522X * BW 655 325 525X * BW 665 325 525X * BW 666 351 525X * BW 666 351 525X * BW 667 351 525X * BW 668 351 668 351 668 351 668 351 668 351 668 351 669		110A		* SHVHLL	- SHVE HLL	5	
519X * 519X * 519X * 519X * 519X * 520X * 520X * 520X * 6844) = FSW 521X * 6844) = BC 523X * 6844) = BC 523X * 6844) = BC 523X * 68441 = BC 524X * 6844) = BC 524X * 68441 = BC 524X * 68441		518X		*SAVALL	SAVES ALL THE	ON THE	
521X * EXIT (SP) = PSW 522X * (SF+4) = BC 523X * (SF+4) = DE 523X * (SF+4) = DE 524X * (SF+4) = DE 525X * USES H;L 525X 525X * SAVALL XTHL 525X 525X * PUSH 525X * PUSH 525X * FUSH 525X *		519X		ENTRY	NON E		
523X * (SP+4) = BC 523X * (SP+4) = DE 524X * (SP+4) = DE 525X * USES H;L 055 325 526X * SAVALL XTHL 055 325 529X * SAVALL XTHL 065 325 532X * PUSH 060 351 532X * FUNH 532X * USAGE 533X * GALL 533X * USAGE 533X * CALL 533X * CALL 534X * USAGE 534X * UNH 535X * UNH 544X * UNH 544X * UNH 545X * ENTRY (A) = INDEX* 545X * ENTRY (A) = INDEX* 546X * ENTRY (A) = INDEX* 555X * TUHP 555X * TUHP 60 = INDEX*		521X		EXIT			
SEST		522X	٠.		. 11:		
525X ** USES H,L 525X ** USES H,L 526X 526X 526X 526X 526X 626X 626X 626X		523X			11		
526X 527		525X		USES	H :		
054 343 522X		526X	- 1				
055 325 529X PUSH B 056 305 530X PUSH B 060 351 PUSH PSW RETURN TO CA 060 351 532X PUSH PSW RETURN TO CA 060 351 532X PCHL PSW RETURN TO CA 536X W USAGE 538X W USAGE 548X W USAGE 548X W USAGE 548X W USAGE 558X W W W W W W W W W W W W W W W W W W W	054	527X	8				
100 100	055	529X			9		
535X ** FCHL FSW RETURN TO CAL 535X ** \$TJMP - TABLE JUMP. 540X ** \$100 - 100	056	530X		PUSH	A		
535X ** \$TJMP - TABLE JUMP. 536X * USAGE 538X * USAGE 538X * USAGE 538X * CALL	090	532X		PUSH	PSW	RETURN TO CALLER	
535X ** \$1JMP - TABLE JUMP. 536X ** USAGE 538X ** USAGE 538X ** CALL \$1JMP (A) = INDEX 539X ** CALL \$1JMP (A) = INDEX = 0 542X **				:			
535X ** \$TJMP - TABLE JUMP. 536X ** USAGE 536X ** USAGE 538X ** USAGE 538X ** CALL \$TJMP (A) = INDEX 540X ** DW ADDR1 INDEX = 0 541X ** 542X ** DW ADDRN INDEX = N-1 545X ** ENTR (A) = INDEX = N-1 545X ** ENTR (A) = INDEX = N-1 545X ** ENTR (A) = INDEX = N-1 545X ** USES A F = INDEX = N-1 545X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX = N-1 55X ** USES A F = INDEX							
539X * USAGE 539X * USAGE 539X * CALL 539X * CALL 540X * DW 541X *		535X		*TJMP -	TABLE		
538X * CALL *TJMF (A) = INDEX = 0 540X * DW ADDR1 INDEX = 0 541X *		537X		USAGE			
539X * CALL *TJMP (A) = INDEX = 0 540X * DW ADDR1 INDEX = 0 541X *		538X					
541X *		539X		CALL	#TUMP		
542X *		541X			•		
543X *		542X		. •	•		
545X * 546X * ENTRY (A) = INDEX 546X * ENTRY (A) = INDEX 547X * EXIT TO PROCESSOR 549X * USES A,F 550X 551X 551X 551X 552X \$TJMP RLC 553X 553X 553X 553X 553X 553X 553X 553		0.45X		. 2	A TITES		
546X * ENTRY (A) = INDEX 547X * EXIT TO PROCESSOR 549X * USES A,F 549X * USES A,F 550X 551X 551X 551X 552X *TJMP RLC 553X 553X 555X *TJMP EQU * 662 343 555X *THL GALE TABLE 643 343 555X		545X					
547X * EXIT TO PROCESSOR 548X * GA = INDEX*2 549X * USES A*F 550X * USES A*F 551X * TUMP R.L.C 552X * TUMP R.L.C 553X * TUMP R.L.C 662 343 555X * TUMP EQU * (HL) = TABLE CAN TANEX*2		246X		ENTRY	(A) = INDEX		
550X * USES A,F + 1445646 * 550X * 550X * USES A,F + 1445646 * 550X 551X 551X		547X		EXIT	10 PROCESSOR		
.061 007 552X \$TJMP RLC (A) = INDEX#2 553X 553X \$TJMP, EQU * .062 343 555X XTHL CALL TABLE .063 343 555X DIEU FOLL CALL TABLE		549X		USES	AFF. T. HITTENSE		
551X 061 007 552X \$TJMP RLC (A) = INDEX#2 553X 553X *TJMP, EQU * (HL) = TABLE 062 343 555X XTHL CALLE TABLE 063 344 555X THL CALLE TABLE		550X					
062 553X \$1JMP, EQU * (HL) = TABLE 602 343 555X XTHL 6010 601 6010 6011 745 6747	.061 00	551X	-	<u>بر</u> ت		(A) = TNDFX#0	
2062 343 525X XTHL * (HL) = TABLE (AL) = TABLE (AZ 345 555X THL CALLE TABLE (AZ 345 554X 554X 554X 554X 554X 554X 554X		553X					
.062 343 555X XIML (HL) = TABLE (AL) = TABLE	062	7400 X 7 1 1 1	₩.	EOU	*		
	063 365	2000 3000X		Y I HL	mS _d	(HL) = TABLE AUDRESS SAUF INTEX#2	
.064 315 101 030 557X CALL \$DADA.	064 315 101 0	557X		CALL	\$DADA.		



144 540	072 146 560X HOV H,H 072 157 561X HOV L,A 072 351 562X POP PSW 073 351 564X XTHL 075 311 564X XTHL 075 311 564X XTHL 564X XTHL 566X X X X X X X X X X X X X X X X X X X	INDEX#2 S ON STACK P PROCESSOR I TABLE, TO SELECT A BYTE FROM THE TIS BYTE ARE ADDED TO THE THE PROCESSOR ADDRESS. = 0 FOR LAB1 = 1 FOR LAB2 = 1 FOR LAB1 = 1 FOR LABN = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
70.73 34.1 56.2X YOPP FSW 70.75 31.1 56.4X KET 70.75 31.2 56.4X KET 70.7	262X POP FSW 263X XTHL 264X # \$TBRA - BRANCH 568X * \$ 568X * \$TBRA USES THE 569X * JUMP TABLE. TH 569X * JUMP TABLE. TH 569X * JUMP TABLE. TH 571X * DB LABI-* 572X * CALL \$TBRA 572X * CALL \$TBRA 572X * CALL \$TBRA 573X * CALL \$TB	INDEX#2 INDEX#2 SIGN STACK P PROCESSOR TO SELECT A BYTE FROM THE IS BYTE ARE ADDED TO THE THE PROCESSOR ADDRESS. = 0 FOR LABI = 1 FOR LABI = 1 FOR LABI = 1 FOR LABI = ADDRESS = TABLE ADDRESS = PROCESSOR ADDRESS
567X ** * *TBRA - BRANCH 568X * * *TBRA - BRANCH 568X * * *TBRA USES THE 570X * JUNP TABLE. TH 571X * ADDRESS OF THE 572X * CALL *TBRA- 572X * TALL *TBRA- 572X * TAL	567X ** \$TBRA - BRANCH 568X * \$568X * \$568X * \$TBRA - BRANCH 568X * \$568X * \$TBRA - BRANCH 571X * DUMP FABLE: TH 571X * DUMP FABLE: THE 572X * DB LABL-* 572X * DB D 5	P PROCESSOR 1 TABLE. 10 SELECT A BYTE FROM THE 11S BYTE ARE ADDED TO THE 11S BYTE ARE ADDRESS. 1 FOR LAB1 1 FOR LAB2 1 FOR LAB2 1 ADDRESS OF ELEMENT 1 PROCESSOR ADDRESS
5627X ** \$TBRA - BRANCH 5687	567X ** \$TBRA - BRANCH 568X * \$ TBRA USES THE 570X * JUNF TABLE. TH 571X * ADDRESS OF THE 572X * CALL 572X * CALL 572X * DB CABL-* 574X * DB C	TABLE. TO SELECT A BYTE FROM THE IIS BYTE ARE ADDED TO THE THE PROCESSOR ADDRESS. = 0 FOR LABI = 1 FOR LABI = 1 FOR LABN = TABLE ADDRESS = ADDRESS = PROCESSOR ADDRESS
5568X * \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$ \$	568X	TO SELECT A BYTE FROM THE IIS BYTE ARE ADDED TO THE THE PROCESSOR ADDRESS. = 0 FOR LAB1 = 1 FOR LAB2 = 1 FOR LAB2 = A-1 FOR LAB3 = A-1 FOR LAB3 = A-1 FOR LAB3 = A-1 FOR LAB3 = PROCESSOR ADDRESS
2007X * * * IBRA USES THE 572X * * DUNE TABLE THE 572X * DUNE TABLE TABLE THE 572X * DUNE TABLE TABLE TABLE THE 572X * DUNE TABLE TABLE THE 572X * DUNE TABLE TABLE THE 572X * DUNE TABLE TA	2077 * * * IBRA USES THE 571X * * FIBRA USES THE 571X * ADDIRESS OF THE 571X * ADDIRESS OF THE 571X * ADDIRESS OF THE 572X * DB LAB1-* 575X * DB LAB1-* 575X * DB LAB2-* 575X * DB LAB2-* 577X * DB LAB2-* 584X * TBRA EQU * COT 325 584X * TBRA EQU * COT 325 587X * DAD D D D D D D D D D D D D D D D D D	10 SELECT A BYTE FROM THE 11S BYTE ARE ADDED TO THE 11F PROCESSOR ADDRESS. = 0 FOR LAB1 = 1 FOR LAB2 = 1 FOR LAB2 = 1 FOR LAB3 = 1 FOR LA
572X * CALL *TBRA 572X * CALL *TBRA 572X * CALL *TBRA 572X * DB LAB1-# 572X * CALL *TD COM 580X * ENTRY (A) = 572X * DB LAB1-# 572X * DB D D D D D D D D D D D D D D D D D	572X * CALL *TBRA B LAB1-# 572X * CALL *TBRA 572X * DB LAB1-# 572X * DB LAB1-# 572X * DB LAB1-# 572X * DB LAB1-# 572X * DB LABN-# 572X * CALL *TBRA B LABN-# 572X * CALL *TBRA B LABN-# 572X * CALL *TBRA B LABN-# 572X * * THL B LABLE - TABLE - TABLE - TABLE	= 0 FOR LABI = 1 F
STATE STAT	573. * CALL *TBRA 574. * DB LAB1-* INDEX 575. * DB LABN-* INDEX 577. * DB LABN-* INDEX 577. * ENTRY (A) = INDEX 577. * ENTRY (A) = INDEX 577. * ENTRY (A) = INDEX 580. * ENTRY (A) = INDEX 580. * ENTRY (A) = INDEX 580. * ENTRY (A) = INDEX 581. * ENTRY (B) = INDEX 581. * ENTRY (B) = INDEX 582. * ENTRY (B) = INDEX 583. * ENTRY (B	= O FOR LAB1 = 1 FOR LABN = TABLE ADDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
2752 * 118	5757 *	= 0 FOR LABI = 1 FOR LABI = 1 FOR LABI = 1 FOR LABI = 1 FOR LABI = ADDRESS = PROCESSOR ADDRESS
576X * DB LABN-* INDEX 577X * ENTRY (A) = INDEX 577X * ENTRY (A) = INDEX 579X * EXIT TO COMPUTED ADDRESS 581X * USES F,H,L 607 325 584X \$TBRA EQU * 607 325 584X \$TBRA EQU * 607 325 585X MOU E,A 607 325 587X MOU E,A 607 321 594X RTH 608 321 594X RTH 608 321 594X RET 608 * ENTRY (A) = PATTERN 608 * COS * ENTRY (A) = PATTERN	576. * DB LABN-* INDEX 577. * ENTRY (A) = INDEX 579. * ENTRY (A) = INDEX 579. * ENT (RE) = TABLE FUA 580. * ENT TO COMPUTED ADDRESS 581. * USES F,H,L 582. 583. * USES F,H,L 607. 325 584. * TBRA EQU * (HL) 607. 325 584. * TBRA EQU * (HL) 607. 325 586. * MOU E,A 600. 588. * MOU E,A 600. 588. * MOU E,A 600. 699. * MOU E,A 600. 699. * * * TBLS - TABLE SEARCH 600. 321 592. * TTAL 600. 600. 600. 600. 600. 600. 600. 600.	= N-1 FOR LABN TABLE ADDRESS ADDRESS OF ELEMENT PROCESSOR ADDRESS
578X * ENTRY (A) = INDEX 578X * 578X * (RET) = TABLE FWA 580X * EXIT TO COMPUTED ADDRESS 581X * USES F.H.L. TO COMPUTED ADDRESS 582X 583X 584X 8TBRA EQU * (HL) 593X XTHL E.A (HL) 100 137 586X MOV E.A (HL) 100 137 589X MOV E.A (HL) 100 130 589X MOV E.A (HL) 100 131 599X * RET 600X * TABLE SEARCH 599X X TABLE FORMAT 600X * TABLE FORMAT 600X * TABLE FORMAT 600X * TABLE FORMAT 600X * ENTRY (A) = PAITERN	579X * ENTRY (A) = INDEX 579X * 579X * (RET) = TABLE FUA 579X * 529X * 581X * USES F,H,L COMPUTED ADDRESS 581X * 582X 583X 583X 583X 583X 583X 583X 585X 583X 585X 583X 585X 583X 583	= TARLE APDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
579X * (RET) = TABLE FUA 580X * USES FIFT 600 332 580X * USES FIFT 600 332 580X * USES FIFT 600 332 580X * USES FIFT 600 4	579X * (RET) = TABLE FWA 580X * EXIT TO COMPUTED ADDRESS 581X * USES F,H,L TO COMPUTED ADDRESS 581X * USES F,H,L TO COMPUTED ADDRESS 581X 581X * THL D	= TABLE ADDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
580X * EXIT TO COMPUTED ADDRESS 581X * USES FINIC ADDRESS 583X 584X *TBRA EQU * * 5076 343 585X THL * (HL) 5075 325 586X MOV E,A 100 137 589X MOV E,A 101 026 000 589X MOV E,A 103 031 599X MOV E,A 105 031 599X MOV E,A 106 321 599X MOV E,A 110 311 594X * *TBLS - TABLE SEARCH 599X * TABLE FORMAT 600X * DB KEYL,VALL, 600X * DB KEYL, 600X *	580X * EXIT TO COMPUTED ADDRESS 581X * USES F,H,L 582X 583X 583X 584X \$TBRA EQU * 525 343 585X 584X MOV E,A 100 137 586X 101 026 000 589X 104 136 599X 105 031 599X 106 321 592X 107 343 593X 110 311 594X	= TABLE ADDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
582X 583X 584X \$TBRA EQU ** 076 343 585X THL ** 077 325 586X MOV E,A MOV E	582X 583X 583X 584X \$TBRA EQU ** 076 343 585X THL 077 325 586X MOV E,A 100 137 589X MOV E,A 101 026 000 588X MOV E,A 103 031 589X MOV E,A 104 021 590X MOV E,A 106 321 592X MOV E,A 110 311 593X XTHL 110 311 594X ** \$TBLS - TABLE SEARCH	= TARLE ADDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
076 584X \$TBRA EQU # 076 343 585X XTHL 077 325 586X XTHL 100 137 587X MOU E,A 101 026 000 588X MOU E,A 104 136 590X MOU E,A 105 031 589X MOU E,A 106 321 589X MOU E,A 107 343 589X MOU E,A 110 311 589X XTHL 599X X XTHL 598X X X XTHL 608X X X X XTHL 608X X X X X X X X X X X X X X X X X X X	076 584X \$TBRA EQU # 076 343 584X \$TBRA EQU # 077 325 586X	= TARLE ADDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
076 343 585X XTHL D (HL) 077 325 586X PUSH D (HL) 100 137 589X PUSH D D (HL) 1103 031 589X PUSH D D (HL) 1104 032 589X PUSH D D (HL) 1105 031 589X PUSH D D (HL) 1105 031 589X PUSH D D (HL) 1107 343 589X PUSH D D (HL) 1107 344 589X PUSH D D (HL) 1107 345 589X PUSH D D (HL) 1107 347 PUSH D D (HL) 1107 348 PUSH D D (HL) 1108 PUSH D D (HL) 1109 PUSH D D (HL) 110	076 343 585X XTHL D (HL) 077 325 586X PUSH D (HL) 100 135 588X MUI D;0 103 031 589X MUI D;0 104 136 590X MOV E;H 105 031 590X MOV E;H 106 331 590X MOV E;H 110 311 594X RET 597X ** \$TRLS - TABLE SEARCH	= TARLE APDRESS = ADDRESS OF ELEMENT = PROCESSOR ADDRESS
7077 325 586X FUSH D 7070 325 587X MUV E'A 7101 026 000 588X MUI D'D 7103 031 589X MUI D'D 7103 031 589X MUI D'D 7104 136 589X MUI 7105 031 589X MUI 7106 031 589X MUI 7107 343 580X MUI 7107 343 580X MUI 7107 34	027 325 586X FUSH D 100 135 587X M0V E,A 101 026 000 588X M0I D,O 103 031 589X M0I D,O 104 136 590X M0V E,H 105 031 591X M0V E,H 106 331 591X MOP D 107 343 593X XTHL 110 311 594X RET 597X ** \$TRLS - TABLE SEARCH	= ADDRESS OF ELEMENT = PROCESSOR ADDRESS
101 026 000 588X MUI D:0 103 031 589X DAB D 104 136 590X DAB D 105 031 592X DAB D 106 321 592X XTHL 107 343 593X XTHL 110 311 594X RET 110 311 596X RET 110	101 026 000 588X MUI D:0 103 031 589X DAD D 104 134 590X DAD D 106 321 592X DAD D 107 343 592X XTHL 110 311 593X XTHL 110 311 593X XTHL 594X RET 597X ## \$TBLS - TABLE SEARCH	= ADDRESS OF ELEMENT = PROCESSOR ADDRESS
104 136 590X H0V E,H 105 031 591X DAD D 106 332 593X POP D 107 343 593X XTHL 110 311 594X RET 598X * *TBLS - TABLE SEARCH 598X * 598X * TABLE FORMAT 600X * DB KEY1,VAL1, 602X * DB KEYN,VALN 605X * ENTRY (A) = PATTERN	104 136 590X MOV E, H 105 031 591X DAD D 106 321 592X POP D 107 343 593X XTHL 110 311 594X RET 597X ** \$TBLS - TABLE SEARCH	= PROCESSOR ADDRESS
105 031 591X DAD D (HL) = PROCESSOR 106 321 592X POP D 592X TTHL 5943X RTHL 5943X RET SPAX RET 5943X RET 599X # #TBLS - TABLE SEARCH 599X # TABLE FORMAT 600X # DB KEY1, VAL1, 602X # 602X # DB KEY1, VAL1, 602X # 605X # DB KEYN, VALN 600X # 606X # ENTRY (A) = PAITERN	105 031 591X DAD D (HL) 106 321 592X POP D 107 343 593X XTHL 110 311 594X RET 597X ** \$TBLS - TABLE SEARCH 500X **	= PROCESSOR ADDRESS
110 311 592X POP 110 311 593X XTHL 110 311 594X RET 597X ** \$TBLS - 598X * 598X * 598X * 598X * 600X * 600X * 602X * 604X * 605X * 606X * 606X * 606X * 606X * 606X *	592X POP D 592X XTHL 5110 311 594X RET 597X ## \$TBLS - TABLE	
594X RET 597X ** *TBLS - 598X * 598X * 500X * 601X * DB 601X * 603X * 605X *	594X RET 597X ** *TBLS - TABLE	
**** * *TBLS - TABLE FI DB ** * * * * * * * * * * * * * * * * *	## \$1BFS - IOBEE	
* TABLE FI * DB * DB * DB * ENTRY	44	
* TABLE FI * DB * DB * DB * DB * DB		
** DB ** DB ** DB ** DB ** DB	* TABLE	
* * * * DB DB DB DB DB	* *	
* DB * DB * DB * ENTRY	**	
* AB * ENTRY	* 08	
* .* ENTRY	* DB	
	* ENTRY	



			X809	*		(H,L) = TABLE F	C31	
			X609		EX1	(A) = PATTEKN IF FOUND	IF FOUND	
			610X	* *	250	'Z' SET IF FOUN	9	
			612X					
:	305		614X	\$TBLS	PUSH	10 m		
	176		616X	\$TBL1	X 00	E.	(A) = CHARACTER	
~~ ,	;	33 031	618X		JZ	#TBL2	IF MATC	
031.120	043		620X		XXI	c II	OKTO DACT	
7:7	∵⊷	13 031	622X		JNZ	\$TBL1	IF NOT END OF TABLE	
٦,	:		623X		DCX	20.3		
77	053 264		624X 625X		OKA OKA	r I	,Σ,	
		000	626X		MUI	A,0	SET (A) = 0 FOR OLD USERS	
			627X 628X	*	DONE			
- 3			629X					
13	301		630X	\$TBL2	P0P	24 :		
031,134	311		631X 632X		RET	I.		
			636X 637X		\$TYPTX	STYPTX IS CALLED TO TYPE	PE A BLOCK OF TEXT ON THE SYSTEM CONSOLE.	
			638X		Y	13 CHEEE 10 11		
			639X		IMBEDDED A BYTE WI	ZERO BYTES TH THE 2000	INDICATE A CARRIAGE RETURN LINE FEED. 3 BIT SET IS THE LAST BYTE IN THE MESSAGE.	
			641X 642X	**	ENTRY	(RET) = TEXT		
			643X		:	10 (KE) TENGIL	42	
			6.45X		:	L .		
•	,		646X	* *	1117			
031.137	315 1	44 031	648X	•:	CALL	*TYPTX.	TYPE II	
	31		650X		RET			
7	176		652X	\$TYPTX.	MOV	E. A		
4	346	122	653X		ANI	1770		
7.	377	002	654X		108 108 108 108 108 108 108 108 108 108	SYSCALL, SCOUT	-	
! =	:		656X		INX	I		
031,153	312.1	144.031	657X 658X		RET	*IYFIX*	MORE . TO . GD	



		*UDD - (UNFACK DECIMAL DIGITS	IGITS.
	* XE99	UDD CON	UDD CONVERTS A 16 BIT V	VALUE INTO A SPECIFIED NUMBER OF
		DECIMAL	•	THE RESULT. IS ZERO FILLED.
		ENTRY	(B,C) = ADDRESS	WALUE
			(A) = DIGIT COUNT (H,L) = MEMORY ADDRESS	NT ADDRESS
	* X699	EXIT	(HL) = (HL) + ((A)
	671X 672X			
5. 070. 030	673X \$UDD	ERU	***************************************	
31.162 345	675X	PUSH	T	SAVE FINAL (H,L) VALUE
31.163 365	677X UDD1	FUSH	PSW	
31 - 164 345	678X	PUSH	I	
31.165 021 012 000 31.170 315 106 030	×629 ×089	LXI CALL	D,10 ♦DU66	(H,L) = VALUE/10
31.173 345	681X	FUSH	T e	75.07 - PERMANDER
31.175 341	683X	904	4 7	(B)C) # KERINDEK
31.176 076 060	684X	HOI	A, '0'	
31,200 203 31,201 053	685X 686X	ADD	шт	ADD REMAINDER
31.202 167	XZ89	¥06	¥,	STORE DIGIT
31 - 203 361 31 - 204 025	X889	POP	PSW	
31.205 302 163 031	X069	JNZ	UDD1	IF MORE TO 60
	691X 692X	P0P RFT	=	RESTORE H RETIEN
	895X **	*ZERO -	*ZERO - ZERO MEMORY	
	•	*ZERO Z	ZEROS A BLOCK OF	MEMORY.
		FNTRY	SSECTION = CHI	
			(B) = COUNT	
	701X *	EXIT	(A) = 0	
	703X			
212	704X 705X 47FP0	VQX (•	
213	ZRO		4	
031.214 043		X	T.	
216	X60Z	JNZ	ZR01	IF MORE
221 311	710X	RET		
	711			