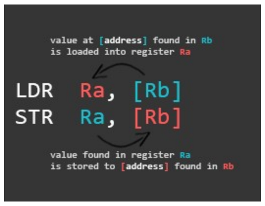
Hex.	Binary		Dave OpCode	Address	Notes	Registers
00	0000	0000	hlt		Halts the CPU	AX/AH/AL
01	0000	0001	nop		No operation	BX/BH/BL
02	0000	0010	fss			CX/CH/CL
03	0000	0011	fsc			DX/DH/DL
04	0000	0100	fis			RT
05	0000	0101	fic			SP
06	0000	0110	fcs			PC
07	0000	0111	fcc			
08	0000	1000	fac			
09	0000	1001	ret			
0a	0000	1010	reti			
0b	0000	1011				
0c	0000	1100				
0d	0000	1101				
0e	0000	1110	dly			
0f	0000	1111				
10	0001	0000	bcs	PC+N		
11	0001	0001	bcc	PC+N		
12	0001	0010	bss	PC+N		
13	0001	0011	bsc	PC+N		
14	0001	0100	bzs	PC+N		
15	0001	0101	bzc	PC+N		
16	0001	0110	glt	PC+N		
17	0001	0111	bge	PC+N		
18	0001	1000	bgt	PC+N		
19	0001	1001	ble	PC+N		
1e	0001	1110		PC+N		
1f	0001	1111		PC+N		
20	0010	0000	inc	_l	Increment lower byte of explicit register	
21	0010	0001	dec	_l	Decrement lower byte of explicit register	
22	0010	0010	clr	_l	Clear lower byte of explicit register	
23	0010	0011	not	_l	Invert lower byte of explicit register	
24	0010	0100	lsl	_l	Shift lower byte of explicit register left	
25	0010	0101	lsr	_l	Shift lower byte of explicit register right	
26	0010	0110	rrc	_l	Rotate lower byte of explicit register right?	
27	0010	0111	rlc	_l	Rotate lower byte of explicit register left?	
28	0010	1000	inc.al!		Increment lower byte of implicit AL register	
29	0010	1001	dec.al!		Decrement lower byte of implicit AL register	
2a	0010	1010	clr.al!		Clear AL lower byte of implicit register	
2b	0010	1011	not.al!		Invert AL lower byte of implicit register	
2c	0010	1100	lsl.al!		Shift AL lower byte of implicit register left	
2d	0010	1101	lsr.al!		Shift AL lower byte of implicit register right	
2e	0010	1110	rrc.al!		Rotate AL lower byte of implicit register right	
2f	0010	1111	rlc.al!		Rotate AL lower byte of implicit register left	
30	0011	0000	inc	_x	Increment full word of explicit register	
31	0011	0001	dec	_x	Decrement full word of explicit register	
32	0011	0010	clr	_x	Clear full word of explicit register	
33	0011	0011	not	_x	Invert full word of explicit register	
34	0011	0100	lsl	_x	Shift full word of explicit register left	
35	0011	0101	lsr	_x	Shift full word of explicit register right	
36	0011	0110	rrc	_x	Rotate full word of explicit register right	
37	0011	0111	rlc	_x	Rotate full word of explicit register left	
38	0011	1000	inc.ax!		Increment full word of implicit AX register	
39	0011	1001	dec.ax!		Decrement full word of implicit AX register	
3a	0011	1010	clr.ax!		Clear full word of implicit AX register	
3b	0011	1011	not.ax!		Invert full word of implicit AX register	
3c	0011	1100	lsl.ax!		Shift full word of implicit AX register left	
3d	0011	1101	lsr.ax!		Shift full word of implicit AX register right	
3e	0011	1110	rrc.ax!		Rotate full word of implicit AX register right	
3f	0011	1111	rlc.ax!		Rotate full word of implicit AX register left	
40	0100	0000	add	_l, _l	Add lower bytes of two explicit registers	
41	0100	0001	sub	_l, _l	Subtract lower bytes of two explicit registers	
42	0100	0010	and	_l, _l	AND lower bytes of two explicit registers	
43	0100	0011	or	_l, _l	OR lower bytes of two explicit registers	
44	0100	0100	xor	_l, _l	XOR lower bytes of two explicit registers	
45	0100	0101	mov	_l, _l	Move lower byte of one explicit register into other explicit register	
46	0100	0110		_l, _l		
47	0100	0111		_l, _l		
48	0100	1000	add.al!	!	Add lower bytes of implicit AL and BL	
49	0100	1001	sub.al!	!	Subtract lower bytes of implicit AL and BL	
4a	0100	1010	and.al!	!	AND lower bytes of implicit AL and BL	
4b	0100	1011	or.al!	!	OR lower bytes of implicit AL and BL	


value at [address] found in Rb is loaded into register Ra
LDR Ra, [Rb]
STR Ra, [Rb]
value found in register Ra is stored to [address] found in Rb

4c	0100	1100	xor.al!	!	XOR lower bytes of implicit AL and BL
4d	0100	1101	mov.al!	!	Move lower byte of implicit AL into implcit BL (backwards?)
4e	0100	1110		!	
4f	0100	1111		!	
50	0101	0000	add	_x, _x	Add full word of two explicit registers
51	0101	0001	sub	_x, _x	Subtract full word of two explicit registers
52	0101	0010	and	_x, _x	AND full word of two explicit registers
53	0101	0011	or	_x, _x	OR full word of two explicit registers
54	0101	0100	xor	_x, _x	XOR full word of two explicit registers
55	0101	0101	mov	_x, _x	Move full word of one explicit register into other explicit register
56	0101	0110		_x, _x	
57	0101	0111		_x, _x	
58	0101	1000	add.ax!	!	Add full word of implicit AX and BX
59	0101	1001	sub.ax!	!	Subtract full word of implicit AX and BX
5a	0101	1010	and.ax!	!	AND full word of implcit AX and BX
5b	0101	1011	or.ax!	!	OR full word of implicit AX and BX
5c	0101	1100	xor.ax!	!	XOR full word of implicit AX and BX
5d	0101	1101	mov.ax!	!	Move full word of implicit AX into implicit BX (backwards?)
5e	0101	1110		!	
5f	0101	1111		!	
60	0110	0000	ld.cx	#D	Load immediate address into full word CX
61	0110	0001	ld.cx	A	Load direct address into full word CX
62	0110	0010	ld.cx	[A]	Load indirect address into full word CX
63	0110	0011	ld.cx	PC+N	Load direct Program Counter offset by N address into full word CX
64	0110	0100	ld.cx	[PC+N]	Load indirect Program Counter offset by N address into full word CX
65	0110	0101	ld.cx	_[R]_	Load indexed mode register into full word CX
66	0110	0110	ld.cx	?R?	
67	0110	0111	ld.cx	?R?	
68	0110	1000	st.cx	#D	Store full word of CX into immediate address
69	0110	1001	st.cx	A	Store full word of CX into direct address
6a	0110	1010	st.cx	[A]	Store full word of CX into indirect address
6b	0110	1011	st.cx	PC+N	Store full word of CX into direct Program Counter offset by N address
6c	0110	1100	st.cx	[PC+N]	Store full word of CX into indirect Program Counter offset by N address
6d	0110	1101	st.cx	_[R]_	Store full word of CX into indexed mode register
6e	0110	1110	st.cx	?R?	
6f	0110	1111	st.cx	?R?	
70	0111	0000	jump	#D	Jump to immediate address
71	0111	0001	jump	A	Jump to direct address
72	0111	0010	jump	[A]	Jump to indirect address
73	0111	0011	jump	PC+N	Jump to direct Program Counter offset by N address
74	0111	0100	jump	[PC+N]	Jump to indirect Program Counter offset by N address
75	0111	0101	jump	_[R]_	Jump to indexed mode register
76	0111	0110	jump	?R?	
77	0111	0111	jump	?R?	
78	0111	1000	call	#D	Call immediate address
79	0111	1001	call	A	Call direct address
7a	0111	1010	call	[A]	Call indirect address
7b	0111	1011	call	PC+N	Call direct Program Counter offset by N address
7c	0111	1100	call	[PC+N]	Call indirect Program Counter offset by N address
7d	0111	1101	call	_[R]_	Call indexed mode register
7e	0111	1110	call	?R?	
7f	0111	1111	call	?R?	
80	1000	0000	ld.al	#D	Load immediate address into lower byte of AL register
81	1000	0001	ld.al	A	Load direct address into lower byte of AL register
82	1000	0010	ld.al	[A]	Load indirect address into lower byte of AL register
83	1000	0011	ld.al	PC+N	Load direct Program Counter offset by N address into lower byte of AL register
84	1000	0100	ld.al	[PC+N]	Load indirect Program Counter offset by N address into lower byte of AL register
85	1000	0101	ld.al	_[R]_	Load indexed register into lower byte of AL register
86	1000	0110	ld.al	?R?	
87	1000	0111	ld.al	?R?	
88	1000	1000	ld.al	AL	Load lower byte of explicit AL? register into lower byte of AL register
89	1000	1001	ld.al	BL	Load lower byte of explicit BL? register into lower byte of AL register
8a	1000	1010	ld.al	CL	Load lower byte of explicit CL? register into lower byte of AL register
8b	1000	1011	ld.al	DL	Load lower byte of explicit DL? register into lower byte of AL register
8c	1000	1100	ld.al	RT	Load lower byte of explicit RT? register into lower byte of AL register
8d	1000	1101	ld.al	SP	Load lower byte of explicit SP? register into lower byte of AL register
8e	1000	1110	ld.al	PC	Load lower byte of explicit PC? register into lower byte of AL register
8f	1000	1111	ld.al	??	
90	1001	0000	ld.ax	#D	Load immediate address into full word of AL register
91	1001	0001	ld.ax	A	Load direct address into full word of AL register
92	1001	0010	ld.ax	[A]	Load indirect address into full word of AL register
93	1001	0011	ld.ax	PC+N	Load direct Program Counter offset by N address into full word of AL register
94	1001	0100	ld.ax	[PC+N]	Load indirect Program Counter offset by N address into full word of AL register

95	1001	0101	ld.ax	_[R]_	Load indexed register into full word of AL register
96	1001	0110	ld.ax	?R?	
97	1001	0111	ld.ax	?R?	
98	1001	1000	ld.ax	AL	Load full word of explicit AX? register into full word of AX register
99	1001	1001	ld.ax	BL	Load full word of explicit BX? register into full word of AX register
9a	1001	1010	ld.ax	CL	Load full word of explicit CX? register into full word of AX register
9b	1001	1011	ld.ax	DL	Load full word of explicit DX? register into full word of AX register
9c	1001	1100	ld.ax	RT	Load full word of explicit RT? register into full word of AX register
9d	1001	1101	ld.ax	SP	Load full word of explicit SP? register into full word of AX register
9e	1001	1110	ld.ax	PC	Load full word of explicit PC? register into full word of AX register
9f	1001	1111	ld.ax	??	
a0	1010	0000	st.al	#D	Store lower byte of AL register into immediate address
a1	1010	0001	st.al	A	Store lower byte of AL register into direct address
a2	1010	0010	st.al	[A]	Store lower byte of AL register into indirect address
a3	1010	0011	st.al	PC+N	Store lower byte of AL register into direct Program Counter offset by N address
a4	1010	0100	st.al	[PC+N]	Store lower byte of AL register into indirect Program Counter offset by N address
a5	1010	0101	st.al	_[R]_	Store lower byte of AL register into indexed register
a6	1010	0110	st.al	?R?	
a7	1010	0111	st.al	?R?	
a8	1010	1000	st.al	AL	Store lower byte of AL? register into lower byte of AL? register
a9	1010	1001	st.al	BL	Store lower byte of AL? register into lower byte of BL? register
aa	1010	1010	st.al	CL	Store lower byte of AL? register into lower byte of CL? register
ab	1010	1011	st.al	DL	Store lower byte of AL? register into lower byte of DL? register
ac	1010	1100	st.al	RT	Store lower byte of AL? register into lower byte of RT? register
ad	1010	1101	st.al	SP	Store lower byte of AL? register into lower byte of SP? register
ae	1010	1110	st.al	PC	Store lower byte of AL? register into lower byte of PC? register
af	1010	1111	st.al	??	
b0	1011	0000	st.ax	#D	Store full word of AX register into immediate address
b1	1011	0001	st.ax	A	Store full word of AX register into direct address
b2	1011	0010	st.ax	[A]	Store full word of AX register into indirect address
b3	1011	0011	st.ax	PC+N	Store full word of AX register into direct Program Counter offset by N address
b4	1011	0100	st.ax	[PC+N]	Store full word of AX register into indirect Program Counter offset by N address
b5	1011	0101	st.ax	_[R]_	Store full word of AX register into indexed register
b6	1011	0110	st.ax	?R?	
b7	1011	0111	st.ax	?R?	
b8	1011	1000	st.ax	AL	Store full word of AX? register into full word of AX? register
b9	1011	1001	st.ax	BL	Store full word of AX? register into full word of BX? register
ba	1011	1010	st.ax	CL	Store full word of AX? register into full word of CX? register
bb	1011	1011	st.ax	DL	Store full word of AX? register into full word of DX? register
bc	1011	1100	st.ax	RT	Store full word of AX? register into full word of RT? register
bd	1011	1101	st.ax	SP	Store full word of AX? register into full word of SP? register
be	1011	1110	st.ax	PC	Store full word of AX? register into full word of PC? register
bf	1011	1111	st.ax	??	
c0	1100	0000	lb.bl	#D	Load immediate address into lower byte of BL register
c1	1100	0001	lb.bl	A	Load direct address into lower byte of BL register
c2	1100	0010	lb.bl	[A]	Load indirect address into lower byte of BL register
c3	1100	0011	lb.bl	PC+N	Load direct Program Counter offset by N address into lower byte of BL register
c4	1100	0100	lb.bl	[PC+N]	Load indirect Program Counter offset by N address into lower byte of BL register
c5	1100	0101	lb.bl	_[R]_	Load indexed register into lower byte of BL register
c6	1100	0110	lb.bl	?R?	
c7	1100	0111	lb.bl	?R?	
c8	1100	1000	lb.bl	AL	Load lower byte of explicit AL? register into lower byte of BL register
c9	1100	1001	lb.bl	BL	Load lower byte of explicit BL? register into lower byte of BL register
ca	1100	1010	lb.bl	CL	Load lower byte of explicit CL? register into lower byte of BL register
cb	1100	1011	lb.bl	DL	Load lower byte of explicit DL? register into lower byte of BL register
cc	1100	1100	lb.bl	RT	Load lower byte of explicit RT? register into lower byte of BL register
cd	1100	1101	lb.bl	SP	Load lower byte of explicit SP? register into lower byte of BL register
ce	1100	1110	lb.bl	PC	Load lower byte of explicit PC? register into lower byte of BL register
cf	1100	1111	lb.bl	??	
d0	1101	0000	ld.bx	#D	Load immediate address into full word of BL register
d1	1101	0001	ld.bx	A	Load direct address into full word of BL register
d2	1101	0010	ld.bx	[A]	Load indirect address into full word of BL register
d3	1101	0011	ld.bx	PC+N	Load direct Program Counter offset by N address into full word of BL register
d4	1101	0100	ld.bx	[PC+N]	Load indirect Program Counter offset by N address into full word of BL register
d5	1101	0101	ld.bx	_[R]_	Load indexed register into full word of BL register
d6	1101	0110	ld.bx	?R?	
d7	1101	0111	ld.bx	?R?	
d8	1101	1000	ld.bx	AL	Load full word of explicit AX? register into full word of BX register
d9	1101	1001	ld.bx	BL	Load full word of explicit BX? register into full word of BX register
da	1101	1010	ld.bx	CL	Load full word of explicit CX? register into full word of BX register
db	1101	1011	ld.bx	DL	Load full word of explicit DX? register into full word of BX register
dc	1101	1100	ld.bx	RT	Load full word of explicit RT? register into full word of BX register
dd	1101	1101	ld.bx	SP	Load full word of explicit SP? register into full word of BX register

de	1101	1110	ld.bx	PC	Load full word of explicit PC? register into full word of BX register
df	1101	1111	ld.bx	??	
e0	1110	0000	st.bl	#D	Store lower byte of BL register into immediate address
e1	1110	0001	st.bl	A	Store lower byte of BL register into direct address
e2	1110	0010	st.bl	[A]	Store lower byte of BL register into indirect address
e3	1110	0011	st.bl	PC+N	Store lower byte of BL register into direct Program Counter offset by N address
e4	1110	0100	st.bl	[PC+N]	Store lower byte of BL register into indirect Program Counter offset by N address
e5	1110	0101	st.bl	_[R]_	Store lower byte of BL register into indexed register
e6	1110	0110	st.bl	?R?	
e7	1110	0111	st.bl	?R?	
e8	1110	1000	st.bl	AL	Store lower byte of BL? register into lower byte of AL? register
e9	1110	1001	st.bl	BL	Store lower byte of BL? register into lower byte of BL? register
ea	1110	1010	st.bl	CL	Store lower byte of BL? register into lower byte of CL? register
eb	1110	1011	st.bl	DL	Store lower byte of BL? register into lower byte of DL? register
ec	1110	1100	st.bl	RT	Store lower byte of BL? register into lower byte of RT? register
ed	1110	1101	st.bl	SP	Store lower byte of BL? register into lower byte of SP? register
ee	1110	1110	st.bl	PC	Store lower byte of BL? register into lower byte of PC? register
ef	1110	1111	st.bl	??	
f0	1111	0000	st.bx	#D	Store full word of BX register into immediate address
f1	1111	0001	st.bx	A	Store full word of BX register into direct address
f2	1111	0010	st.bx	[A]	Store full word of BX register into indirect address
f3	1111	0011	st.bx	PC+N	Store full word of BX register into direct Program Counter offset by N address
f4	1111	0100	st.bx	[PC+N]	Store full word of BX register into indirect Program Counter offset by N address
f5	1111	0101	st.bx	_[R]_	Store full word of BX register into indexed register
f6	1111	0110	st.bx	?R?	
f7	1111	0111	st.bx	?R?	
f8	1111	1000	st.bx	AL	Store full word of BX? register into full word of AX? register
f9	1111	1001	st.bx	BL	Store full word of BX? register into full word of BX? register
fa	1111	1010	st.bx	CL	Store full word of BX? register into full word of CX? register
fb	1111	1011	st.bx	DL	Store full word of BX? register into full word of DX? register
fc	1111	1100	st.bx	RT	Store full word of BX? register into full word of RT? register
fd	1111	1101	st.bx	SP	Store full word of BX? register into full word of SP? register
fe	1111	1110	st.bx	PC	Store full word of BX? register into full word of PC? register
ff	1111	1111	st.bx	??	