

op	<b>mov rd,rs</b>	0 0 0 1 0 0 0 0	0x10	rd <- rs	registers	a	111
2nd byte		x d d d x s s s	n/a			b	000
3rd byte		next op-code				c	001
						d	010
op	<b>mov r,m</b>	0 0 1 0 0 0 1 0	0x22	r <- [HL]		e	011
2nd byte		x d d d x x x x	n/a			h	100
3rd byte		next op-code				l	101
op	<b>mov m,r</b>	0 0 1 1 0 0 0 0	0x30	[HL] <- r			
2nd byte		x x x x x s s s	n/a				
3rd byte		next op-code					
op	<b>mvi r,data</b>	0 1 0 0 0 0 1 1	0x43	r <- 2nd byte			
2nd byte		x d d d x x x x	n/a				
3rd byte		8 bit data					
4th byte		next op-code					
op	<b>lda addr</b>	0 1 0 1 0 1 1 1	0x57	a <- [address]			
2nd byte		low byte address					
3rd byte		high byte address					
4th byte		next op-code					
op	<b>sta addr</b>	0 1 1 0 0 1 0 1	0x65	[address] <- a			
2nd byte		low byte address					
3rd byte		high byte address					
4th byte		next op-code					
op	<b>add r</b>	0 1 1 1 0 0 0 0	0x70	a <- a + r			
2nd byte		0 0 0 0 0 s s s					
3rd byte		next op-code					
op	<b>adc r</b>	0 1 1 1 0 0 0 0	0x70	a <- a + r + carry			
2nd byte		0 0 0 1 0 s s s					
3rd byte		next op-code					
op	<b>sub r</b>	0 1 1 1 1 0 0 0	0x78	a <- a - r			
2nd byte		0 0 0 0 1 s s s					
3rd byte		next op-code					
op	<b>sbb r</b>	0 1 1 1 1 0 0 0	0x78	a <- a - r - borrow			
2nd byte		0 0 0 1 0 s s s					
3rd byte		next op-code					
op	<b>and r</b>	0 1 1 1 0 0 0 0	0x70	a <- a & r			
2nd byte		0 0 1 0 0 s s s					
3rd byte		next op-code					
op	<b>or r</b>	0 1 1 1 0 0 0 0	0x70	a <- a   r			
2nd byte		0 1 0 0 0 s s s					
3rd byte		next op-code					
op	<b>xor r</b>	0 1 1 1 0 0 0 0	0x70	a <- a xor r			
2nd byte		0 1 1 0 0 s s s					
3rd byte		next op-code					
op	<b>cma</b>	0 1 1 1 0 0 0 0	0x70	a <- ~a			
2nd byte		1 0 0 0 0 x x x	0x80				
3rd byte		next op-code					

op 2nd byte 3rd byte	<b>rla</b>	0 1 1 1 0 0 0 0 1 0 1 0 0 x x x next op-code	0x70 0xA0	rotate left accumulator (circular)
op 2nd byte 3rd byte	<b>rra</b>	0 1 1 1 0 0 0 0 1 1 0 0 0 x x x next op-code	0x70 0xC0	rotate right accumulator (circular)
op 2nd byte	<b>stc</b>	1 0 0 0 0 0 1 0 next op-code	0x82	set carry
op 2nd byte	<b>rtc</b>	1 0 0 1 0 0 1 0 next op-code	0x92	reset carry
op 2nd byte 3rd byte 4th byte 5th byte	<b>jmp</b>	1 0 1 0 0 0 1 1 x x 0 0 0 0 1 1 low byte address high byte address next op-code	0xA3 0x03	jump uncon. to address
op 2nd byte 3rd byte 4th byte 5th byte	<b>jz</b>	1 0 1 0 0 0 1 1 x x 0 0 0 0 0 1 low byte address high byte address next op-code	0xA3 0x01	jump to address if Z=1
op 2nd byte 3rd byte 4th byte 5th byte	<b>jnz</b>	1 0 1 0 0 0 1 1 x x 0 0 0 0 1 0 low byte address high byte address next op-code	0xA3 0x02	jump to address if Z=0
op 2nd byte 3rd byte 4th byte 5th byte	<b>jc</b>	1 0 1 0 0 0 1 1 x x 0 0 0 1 0 0 low byte address high byte address next op-code	0xA3 0x04	jump to address if C=1
op 2nd byte 3rd byte 4th byte 5th byte	<b>jnc</b>	1 0 1 0 0 0 1 1 x x 0 0 1 0 0 0 low byte address high byte address next op-code	0xA3 0x08	jump to address if C=0
op 2nd byte 3rd byte 4th byte 5th byte	<b>jm</b>	1 0 1 0 0 0 1 1 x x 0 1 0 0 0 0 low byte address high byte address next op-code	0xA3 0x10	jump to address if S=1
op 2nd byte 3rd byte 4th byte 5th byte	<b>jp</b>	1 0 1 0 0 0 1 1 x x 1 0 0 0 0 0 low byte address high byte address next op-code	0xA3 0x20	jump to address if S=0
op 2nd byte	<b>hlt</b>	1 0 1 1 0 0 1 0 next op-code	0xB2	halt machine
op 2nd byte	<b>nop</b>	0 0 0 0 0 0 0 0 next op-code	0x00	no operate