

Kyber 4: instruction set

0000rrrr	mra + reg	; (reg) > A	0x000[register] load
0001rrrr	mar + reg	; A > (reg)	save to 0x000[register]
0010rrrr	msa + reg	; (ind/reg) > A	0x00[index][register] load
0011cccc	mca + const	; const > A	constant load
0100rrrr	lgr + reg	; (reg) & /A > A	NOT(A) AND 0x000[register] load
0101rrrr	mas + reg	; A > (ind/reg)	save to 0x00[index][register]
0110rrrr	lgs + reg	; (ind/reg) & /A > A	NOT(A) AND 0x00[index][register] load
01110000	brp	; break point	break if enable -> manual control enter
0111cccc	lgc + const	; /zerro(const): const & /A > A	NOT(A) AND constant load if non zero constant
100000ii	ppw + i	; A > ptr[i]	pointer part [i] write
10000100	mtp	; /(ram++) > (port)	inverted RAM table move to port, pointer postincrement
10000101	mpt	; /(port) > (ram++)	inverted port move to RAM table, pointer postincrement
10000110	ica	; A + 1 > A	increment A
10000111	dca	; A - 1 > A	decrement A
10001000	luc	; /(romu++) > A	inverted table constant load from ROM up nibble, pointer postincrement
10001001	mat	; A > (ram++)	table variable save to RAM, pointer postincrement
10001010	fng	; /F > F	flag complement
10001011	flg	; /zerro(/A) > F	\$f in A test
10001100	mta	; (ram++) > A	table variable load from RAM, pointer postincrement
10001101	icp	; ptr + 1 > ptr	pointer increment
10001110	fst	; 1 > F	flag set
10001111	dcp	; ptr - 1 > ptr	pointer decrement
10010000	map	; A > (port)	write to port
10010001	juc	; segA > PC	segment[A] uncondition jump
10010010	pas	; A > port	bits 0..2 port address / index set; bit 3 cache SRAM write enable
10010011	buc	; PC + 0x0001 > PC, A > PCd	after PC increment uncondition branch
10010100	jfh	; F: segA > PC	segment[A] jump if flag high
10010101	jfl	; /F: segA > PC	segment[A] jump if flag low
10010110	bfh	; F: PC - 0x0010 > PC, A > PCd	branch down if flag high
10010111	bfl	; /F: PC + 0x0001 > PC, A > PCd	after PC increment branch if flag low
10011000	ldc	; /(romd) > A	inverted table constant load from ROM down nibble
10011001	mpa	; (port) > A	port load
10011010	lro	; F <- A <- F	left rotate
10011011	rro	; F -> A -> F	right rotate
100111ii	ppr + i	; ptr[i] > A	pointer part [i] read
10100000			
..	a..	... ; PC > ptr, 0x0000 > PC	save PC to address pointer, jump to null address
..			used for advanced instructions emulation
11111111			
reset:		0x0000 > PC, 0x0000 > ptr, 0 > A	ptr clear, a clear, jump to null address