Sonne CPU Instruction and Opcode Reference (rev. Schneelein)

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
KICKS		
NOP	00	Do nothing. Increment Low by 1.
ISI	01	Shift SIR left by one position, replace least significant digit by bit on MISO line.
OSO	02	Set bit on MOSI line to most significant digit of SOR, shift SOR left by one position.
SCL	03	Set SPI clock line low. Increment Low by 1.
SCH	04	Set SPI clock line high. Increment Low by 1.
RDY	05	Set POR to tristate. Increment Low by 1.
LLF	06	Copy value at memory address 256*L+0xC4 into R, set O to 0, increment L by one. Incre
ELF	07	Decrement L by one, store R into memory at address 256*L+0xC4. Increment Low by 1.
REA	08	Copy Clip to A, set xMx. Increment Low by 1.
REB	09	Copy Clip to B, set xMx. Increment Low by 1.
_	0A	
_	0B	
_	0C	
_	0D	
_	0E	
_	0F	
GET-PUT		
G0A	10	Copy A into Clip, copy value at memory address 0x80 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
G1A	11	Copy A into Clip, copy value at memory address 0x81 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
G2A	12	Copy A into Clip, copy value at memory address 0x82 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
G3A	13	Copy A into Clip, copy value at memory address 0x83 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
L0A	14	Copy A into Clip, copy value at memory address 0xC0 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
L1A	15	Copy A into Clip, copy value at memory address 0xC1 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
L2A	16	Copy A into Clip, copy value at memory address 0xC2 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
L3A	17	Copy A into Clip, copy value at memory address 0xC3 into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
AG0	18	Copy A into memory at address 0x80. Increment Low by 1.
AG1	19	Copy A into memory at address 0x81. Increment Low by 1.
AG2	1A	Copy A into memory at address 0x82. Increment Low by 1.
AG3	1B	Copy A into memory at address 0x83. Increment Low by 1.
AL0	1C	Copy A into memory at address 0xC0. Increment Low by 1.
AL1	1D	Copy A into memory at address 0xC1. Increment Low by 1.
AL2	1E	Copy A into memory at address 0xC2. Increment Low by 1.
AL3	1F	Copy A into memory at address 0xC3. Increment Low by 1.
G0B	20	Copy B into Clip, copy value at memory address 0x80 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
G1B	21	Copy B into Clip, copy value at memory address 0x81 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
G2B	22	Copy B into Clip, copy value at memory address 0x82 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
G3B	23	Copy B into Clip, copy value at memory address 0x83 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
L0B	24	Copy B into Clip, copy value at memory address 0xC0 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
L1B	25	Copy B into Clip, copy value at memory address 0xC1 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
L2B	26	Copy B into Clip, copy value at memory address 0xC2 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
L3B	27	Copy B into Clip, copy value at memory address 0xC3 into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
BG0	28	Copy B into memory at address 0x80. Increment Low by 1.
BG1	29	Copy B into memory at address 0x81. Increment Low by 1.
BG2	2A	Copy B into memory at address 0x82. Increment Low by 1.
BG3	2B	Copy B into memory at address 0x83. Increment Low by 1.
BL0	2C	Copy B into memory at address 0xC0. Increment Low by 1.
BL1	2D	Copy B into memory at address 0xC1. Increment Low by 1.
BL2	2E	Copy B into memory at address 0xC2. Increment Low by 1.
BL3	2F	Copy B into memory at address 0xC3. Increment Low by 1.
G0R	30	Copy value at memory address 0x80 into R, set O to 0. Increment Low by 1.
G1R	31	Copy value at memory address 0x81 into R, set O to 0. Increment Low by 1.
G2R	32	Copy value at memory address 0x82 into R, set O to 0. Increment Low by 1.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
G3R	33	Copy value at memory address 0x83 into R, set O to 0. Increment Low by 1.
L0R	34	Copy value at memory address 0xC0 into R, set O to 0. Increment Low by 1.
L1R	35	Copy value at memory address 0xC1 into R, set O to 0. Increment Low by 1.
L2R	36	Copy value at memory address 0xC2 into R, set O to 0. Increment Low by 1.
L3R	37	Copy value at memory address 0xC3 into R, set O to 0. Increment Low by 1.
RG0	38	Copy R + O into memory at address 0x80. Increment Low by 1.
RG1	39	Copy R + O into memory at address 0x81. Increment Low by 1.
RG2	3A	Copy R + O into memory at address 0x82. Increment Low by 1.
RG3	3B	Copy R + O into memory at address 0x83. Increment Low by 1.
RL0	3C	Copy R + O into memory at address 0xC0. Increment Low by 1.
RL1	3D	Copy R + O into memory at address 0xC1. Increment Low by 1.
RL2	3E	Copy R + O into memory at address 0xC2. Increment Low by 1.
RL3	3F	Copy R + O into memory at address 0xC3. Increment Low by 1.
TRAPS		
*0	40	Copy High into H, copy Low into R, set O to zero, copy 0x00 into H, set Low to zero.
*1	41	Copy High into H, copy Low into R, set O to zero, copy 0x01 into H, set Low to zero.
*2	42	Copy High into H, copy Low into R, set O to zero, copy 0x02 into H, set Low to zero.
*3	43	Copy High into H, copy Low into R, set O to zero, copy 0x03 into H, set Low to zero.
*4	44	Copy High into H, copy Low into R, set O to zero, copy 0x04 into H, set Low to zero.
*5	45	Copy High into H, copy Low into R, set O to zero, copy 0x05 into H, set Low to zero.
*6	46	Copy High into H, copy Low into R, set O to zero, copy 0x06 into H, set Low to zero.
*7	47	Copy High into H, copy Low into R, set O to zero, copy 0x07 into H, set Low to zero.
*8	48	Copy High into H, copy Low into R, set O to zero, copy 0x08 into H, set Low to zero.
*9	49	Copy High into H, copy Low into R, set O to zero, copy 0x09 into H, set Low to zero.
*10	4A	Copy High into H, copy Low into R, set O to zero, copy 0x0A into H, set Low to zero.
*11	4B	Copy High into H, copy Low into R, set O to zero, copy 0x0B into H, set Low to zero.
*12	4C	Copy High into H, copy Low into R, set O to zero, copy 0x0C into H, set Low to zero.
*13	4D	Copy High into H, copy Low into R, set O to zero, copy 0x0D into H, set Low to zero.
*14	4E	Copy High into H, copy Low into R, set O to zero, copy 0x0E into H, set Low to zero.
*15	4F	Copy High into H, copy Low into R, set O to zero, copy 0x0F into H, set Low to zero.
*16	50	Copy High into H, copy Low into R, set O to zero, copy 0x10 into H, set Low to zero.
*17	51	Copy High into H, copy Low into R, set O to zero, copy 0x11 into H, set Low to zero.
*18	52	Copy High into H, copy Low into R, set O to zero, copy 0x12 into H, set Low to zero.
*19	53	Copy High into H, copy Low into R, set O to zero, copy 0x13 into H, set Low to zero.
*20	54	Copy High into H, copy Low into R, set O to zero, copy 0x14 into H, set Low to zero.
*21	55	Copy High into H, copy Low into R, set O to zero, copy 0x15 into H, set Low to zero.
*22	56	Copy High into H, copy Low into R, set O to zero, copy 0x16 into H, set Low to zero.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
*23	57	Copy High into H, copy Low into R, set O to zero, copy 0x17 into H, set Low to zero.
*24	58	Copy High into H, copy Low into R, set O to zero, copy 0x18 into H, set Low to zero.
*25	59	Copy High into H, copy Low into R, set O to zero, copy 0x19 into H, set Low to zero.
*26	5A	Copy High into H, copy Low into R, set O to zero, copy 0x1A into H, set Low to zero.
*27	5B	Copy High into H, copy Low into R, set O to zero, copy 0x1B into H, set Low to zero.
*28	5C	Copy High into H, copy Low into R, set O to zero, copy 0x1C into H, set Low to zero.
*29	5D	Copy High into H, copy Low into R, set O to zero, copy 0x1D into H, set Low to zero.
*30	5E	Copy High into H, copy Low into R, set O to zero, copy 0x1E into H, set Low to zero.
*31	5F	Copy High into H, copy Low into R, set O to zero, copy 0x1F into H, set Low to zero.
OFFSET		
R0P	60	Set O to 0. Increment Low by 1.
R1P	61	Set O to 1. Increment Low by 1.
R2P	62	Set O to 2. Increment Low by 1.
R3P	63	Set O to 3. Increment Low by 1.
R4P	64	Set O to 4. Increment Low by 1.
R5P	65	Set O to 5. Increment Low by 1.
R6P	66	Set O to 6. Increment Low by 1.
R7P	67	Set O to 7. Increment Low by 1.
R8M	68	Set O to -8. Increment Low by 1.
R7M	69	Set O to -7. Increment Low by 1.
R6M	6A	Set O to -6. Increment Low by 1.
R5M	6B	Set O to -5. Increment Low by 1.
R4M	6C	Set O to -4. Increment Low by 1.
R3M	6D	Set O to -3. Increment Low by 1.
R2M	6E	Set O to -2. Increment Low by 1.
R1M	6F	Set O to -1. Increment Low by 1.
ALU		
IDA	70	Store A into R, set O to 0. Increment Low by 1.
IDB	71	Store B into R, set O to 0. Increment Low by 1.
OCA	72	Store one's complement of A into R, set O to 0. Increment Low by 1.
OCB	73	Store one's complement of B into R, set O to 0. Increment Low by 1.
SLA	74	Store result of shift-left A into R, set O to 0. Increment Low by 1.
SLB	75	Store result of shift-left B into R, set O to 0. Increment Low by 1.
SRA	76	Store result of logical shift-right of A into R, set O to 0. Increment Low by 1.
SRB	77	Store result of logical shift-right of B into R, set O to 0. Increment Low by 1.
AND	78	Store result of A logical-AND B into R, set O to 0. Increment Low by 1.
IOR	79	Store result of A logical-OR B into R, set O to 0. Increment Low by 1.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
EOR	7A	Store result of A logical-XOR B into R, set O to 0. Increment Low by 1.
ADD	7B	Store 8-bit result of A plus B into R, set O to 0. Increment Low by 1.
CAR	7C	Store carry bit (0 or 1) of A plus B into R, set O to 0. Increment Low by 1.
ALB	7D	Store 255 into R if A less than B, else store 0, set O to 0. Increment Low by 1.
AEB	7E	Store 255 into R if A equals B, else store 0, set O to 0. Increment Low by 1.
AGB	7F	Store 255 into R if A greater than B, else store 0, set O to 0. Increment Low by 1.
COPY		
NI	80	Copy next opcode into I, increment Low by 2.
RET	81	Copy H into High, copy R into Low, set O to zero.
NR	82	Copy next opcode into R, set O to 0, increment Low by 2.
NH	83	Copy next opcode into H, increment Low by 2.
NG	84	Copy next opcode into G, increment Low by 2.
NL	85	Copy next opcode into L, increment Low by 2.
NS	86	Copy next opcode into SOR, increment Low by 2.
NP	87	Copy next opcode into POR. Set POR to low impedance. Increment Low by 2.
NE	88	Copy next opcode into E, increment Low by 2.
NJ	89	Copy next opcode into Low.
ND	8A	Decrement I, then if I is not zero, copy next opcode into Low, else increment Low by 1.
NT	8B	If R plus O is not zero, then copy next opcode into Low, else increment Low by 1.
NF	8C	If R plus O is zero, then copy next opcode into Low, else increment Low by 1.
NC	8D	Copy High into H, copy Low into R, copy next opcode into H, set Low to zero, set O to
NB	8E	Copy B into Clip, copy next opcode into A, increment Low by 2. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
NA	8F	Copy A into Clip, copy next opcode into A, increment Low by 2. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
MI	90	Copy value at memory address xMx into I. Increment Low by 1.
LID	91	Increment High by 1, store High into H, set Low to zero.
MR	92	Copy value at memory address xMx into R, set O to 0. Increment Low by 1.
MH	93	Copy value at memory address xMx into H. Increment Low by 1.
MG	94	Copy value at memory address xMx into G. Increment Low by 1.
ML	95	Copy value at memory address xMx into L. Increment Low by 1.
MS	96	Copy value at memory address xMx into SOR. Increment Low by 1.
MP	97	Copy value at memory address xMx into POR. Set POR to low impedance. Increment Low by 1.
ME	98	Copy value at memory address xMx into E. Increment Low by 1.
MJ	99	Copy value at memory address xMx into Low.
MD	9A	Decrement I, then if I is not zero, copy value at memory address xMx into Low, else increment Low by 1.
MT	9B	If R plus O is not zero, then copy value at memory address xMx into Low, else increme

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
MF	9C	If R plus O is zero, then copy value at memory address xMx into Low, else increment L
MC	9D	Copy High into H, copy Low into R, copy value at memory address xMx into H, set Low t
MB	9E	Copy B into Clip, copy value at memory address xMx into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
MA	9F	Copy A into Clip, copy value at memory address xMx into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
RI	A0	Copy R + O to I. Increment Low by 1.
RM	A1	Copy R + O to memory at address xMx. Increment Low by 1.
RR	A2	Do nothing. Increment Low by 1.
RH	A3	Copy R + O to H. Increment Low by 1.
RG	A4	Copy R + O to G. Increment Low by 1.
RL	A5	Copy R + O to L. Increment Low by 1.
RS	A6	Copy R + O to SOR. Increment Low by 1.
RP	A7	Copy R + O to POR. Set POR to low impedance. Increment Low by 1.
RE	A8	Copy R + O to E. Increment Low by 1.
RJ	A9	Copy R + O to Low.
RD	AA	Decrement I, then if I is not zero, copy R into Low, else increment Low by 1.
RT	AB	If R plus O is not zero, then copy R into Low, else increment Low by 1.
RF	AC	If R plus O is zero, then copy R into Low, else increment Low by 1.
RC	AD	Copy High into H, copy R + O into High, copy Low into R, set Low to zero, set O to zero.
RB	AE	Copy B into Clip, copy R + O into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
RA	AF	Copy A into Clip, copy R + O into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
HI	B0	Copy H into I. Increment Low by 1.
HM	B1	Copy H to memory at address xMx. Increment Low by 1.
HR	B2	Copy H into R, set O to 0. Increment Low by 1.
HH	B3	Do nothing. Increment Low by 1.
HG	B4	Copy H into G. Increment Low by 1.
HL	B5	Copy H into L. Increment Low by 1.
HS	B6	Copy H into SOR. Increment Low by 1.
HP	B7	Copy H into POR. Set POR to low impedance. Increment Low by 1.
HE	B8	Copy H into E. Increment Low by 1.
HJ	B9	Copy H into Low.
HD	BA	Decrement I, then if I is not zero, copy H into Low, else increment Low by 1.
HT	BB	If R plus O is not zero, then copy H into Low, else increment Low by 1.
HF	BC	If R plus O is zero, then copy H into Low, else increment Low by 1.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
HC	BD	Copy H into High, copy Low into R, set Low to zero, set O to zero.
HB	BE	Copy B into Clip, copy H into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
HA	BF	Copy A into Clip, copy H into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
GI	C0	Copy G into I. Increment Low by 1.
GM	C1	Copy G into memory at address xMx. Increment Low by 1.
GR	C2	Copy G into R, set O to 0. Increment Low by 1.
GH	C3	Copy G into H. Increment Low by 1.
GG	C4	Do nothing. Increment Low by 1.
GL	C5	Copy G into L. Increment Low by 1.
GS	C6	Copy G into SOR. Increment Low by 1.
GP	C7	Copy G into POR. Set POR to low impedance. Increment Low by 1.
GE	C8	Copy G into E. Increment Low by 1.
GJ	C9	Copy G into Low.
GD	CA	Decrement I, then if I is not zero, copy G into Low, else increment Low by 1.
GT	CB	If R plus O is not zero, then copy G into Low, else increment Low by 1.
GF	CC	If R plus O is zero, then copy G into Low, else increment Low by 1.
GC	CD	Copy High into H, copy G into High, copy Low into R, set Low to zero, set O to zero.
GB	CE	Copy B into Clip, copy H into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
GA	CF	Copy A into Clip, copy H into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
LI	D0	Copy L into I. Increment Low by 1.
LM	D1	Copy L into memory at address xMx. Increment Low by 1.
LR	D2	Copy L into R, set O to 0. Increment Low by 1.
LH	D3	Copy L into H. Increment Low by 1.
LG	D4	Copy L into G. Increment Low by 1.
LL	D5	Do nothing. Increment Low by 1.
LS	D6	Copy L into SOR. Increment Low by 1.
LP	D7	Copy L into POR. Set POR to low impedance. Increment Low by 1.
LE	D8	Copy L into E. Increment Low by 1.
LJ	D9	Copy L into Low.
LD	DA	Decrement I, then if I is not zero, copy L into Low, else increment Low by 1.
LT	DB	If R plus O is not zero, then copy L into Low, else increment Low by 1.
LF	DC	If R plus O is zero, then copy G into Low, else increment Low by 1.
LC	DD	Copy High into H, copy L into High, copy Low into R, set Low to zero, set O to zero.
LB	DE	Copy B into Clip, copy L into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.

Mnemonic	Opcode (hexadecimal)	Operation (Low = low order byte of program counter, High = high order byte of program counter, xMx is an implied 16-bit memory address. Pseudo register N stands for the next opcode in the instruction stream, essentially a code literal.)
LA	DF	Copy A into Clip, copy L into B. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
SI	E0	Copy SIR into I. Increment Low by 1.
SM	E1	Copy SIR to memory at address xMx. Increment Low by 1.
SR	E2	Copy SIR to R. Increment Low by 1.
SH	E3	Copy SIR to H. Increment Low by 1.
SG	E4	Copy SIR to G. Increment Low by 1.
SL	E5	Copy SIR to L. Increment Low by 1.
SS	E6	Copy SIR to SOR. Increment Low by 1.
SP	E7	Copy SIR to POR. Set POR to low impedance. Increment Low by 1.
SE	E8	Copy SIR to E. Increment Low by 1.
SJ	E9	Copy SIR to Low.
SD	EA	Decrement I, then if I is not zero, copy SIR into Low, else increment Low by 1.
ST	EB	If R plus O is not zero, then copy SIR into Low, else increment Low by 1.
SF	EC	If R plus O is zero, then copy SIR into Low, else increment Low by 1.
SC	ED	Copy High into H, copy SIR into High, copy Low into R, set Low to zero, set O to zero
SB	EE	Copy B into Clip, copy SIR into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
SA	EF	Copy A into Clip, copy SIR into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.
PI	F0	Copy PIR into I. Increment Low by 1.
PM	F1	Copy PIR to memory at address xMx. Increment Low by 1.
PR	F2	Copy PIR to R, set O to 0. Increment Low by 1.
PH	F3	Copy PIR to H. Increment Low by 1.
PG	F4	Copy PIR to G. Increment Low by 1.
PL	F5	Copy PIR to L. Increment Low by 1.
PS	F6	Copy PIR to SOR. Increment Low by 1.
PP	F7	Do nothing. Increment Low by 1.
PE	F8	Copy PIR to E. Increment Low by 1.
PJ	F9	Copy PIR to D. Increment Low by 1.
PD	FA	Decrement I, then if I is not zero, copy PIR into Low, else increment Low by 1.
PT	FB	If R plus O is not zero, then copy PIR into Low, else increment Low by 1.
PF	FC	If R plus O is zero, then copy PIR into Low, else increment Low by 1.
PC	FD	Copy High into H, copy PIR into High, copy Low into R, set Low to zero, set O to zero
PB	FE	Copy B into Clip, copy PIR into B. Increment Low by 1. If bit 7 of B is zero then set mXm to 256*H+B, else if bit 6 of B is zero then set xMx to 256*G+B, else set xMx to 256*L+B.
PA	FF	Copy A into Clip, copy PIR into A. Increment Low by 1. If bit 7 of A is zero then set mXm to 256*H+A, else if bit 6 of A is zero then set xMx to 256*G+A, else set xMx to 256*L+A.