Micro Instructions

//FETCH ANY INSTRUCTION
T0 PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN
T1 Z_OUT,PC_IN
T2 MDR_OUT,IR_IN
T3 BRANCH
//
Clocks #4
Memory access #1
//FETCH (SRC/DST)
T0 (SRC/DST)_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #1
Memory access #0
//FETCH (SRC/DST)+
T0 (SRC/DST)_OUT,MAR_IN,READ,Y_IN,INC,Z_IN
T1 Z_OUT,(SRC/DST)_IN
T2 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #3
Memory access #1
//FETCH -(SRC/DST)
T0 (SRC/DST)_OUT,Y_IN,DEC,Z_IN
T1 Z_OUT,(SRC/DST)_IN,MAR_IN,READ
T2 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #3
Memory access #1
//FETCH @(SRC/DST)
T0 (SRC/DST)_OUT,MAR_IN,READ
T1 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #2
Memory access #1
//FETCH @(SRC/DST)+
T0 (SRC/DST)_OUT,MAR_IN,READ,Y_IN,INC,Z_IN
T1 Z_OUT,(SRC/DST)_IN
T2 MDR_OUT,MAR_IN,READ
T3 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #4
Memory access #2

//FETCH @-(SRC/DST)
T0 (SRC/DST)_OUT,Y_IN,DEC,Z_IN
T1 Z_OUT,(SRC/DST)_IN,MAR_IN,READ
T2 MDR_OUT,MAR_IN,READ
T3 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #4
Memory access #2
//FETCH X(SRC/DST)
T0 PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN
T1 Z_OUT,PC_IN
T2 (SRC/DST)_OUT,Y_IN
T3 MDR_OUT,ADD,Z_IN
T4 Z_OUT,MAR_IN,READ
T5 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #6
Memory access #2
//FETCH @X(SRC/DST)
T0 PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN
T1 Z_OUT,PC_IN
T2 (SRC/DST)_OUT,Y_IN
T3 MDR_OUT,ADD,Z_IN
T4 Z_OUT,MAR_IN,READ
T5 MDR_OUT,MAR_IN,READ
T6 MDR_OUT,(SRCTMP/DSTTMP)_IN
//
Clocks #7
Memory access #3
//EXECUTE (MOV)
T0 SRCTMP_OUT,DSTTMP_IN
//
Clocks #1
Memory access #0
//EXECUTE (AND,OR,XNOR,ADD,ADC,SUB,SBC,INC,DEC,CLR,INV,ROR,RRC,ASR,LSL,.....)
T0 DSTTMP_OUT,Y_IN
T1 SRCTMP_OUT,(AND,OR,XOR,INC,DEC,CLR,INV,ROR,RRC,ASR,LSL,.....),Z_IN
T2 Z_OUT,DSTTMP_IN
//
Clocks #3
Memory access #0

//STORE DST
T0 DSTTMP_OUT,DST_IN
//
Clocks #1
Memory access #0

//STORE DST+,-DST,@DST,@DST+,@-DST,X(DST),@X(DST)
T0 DSTTMP_OUT,MDR_IN,WRITE
//
Clocks #1
Memory access #1
//END
Clocks #1
Memory access #0

1-OP

//FETCH ANY INSTRUCTION
//FETCH DST(ADDRESSING MODES ABOVE)
//EXECUTE (INC,DEC,CLR,INV,LSR,ROR,RRC,ASR,LSL...)
//STORE LIKE ABOVE
//END

BRANCH

//FETCH ANY INSTRUCTION
//
T0 IR_OFFSET,Y_IN IF(!condition) then END
T1 PC_OUT,ADD,Z_IN
T2 Z_OUT,PC_IN
Clocks #3 if(condition) #1 if(!condition)
Memory access #0
//END

NOP

//FETCH ANY INSTRUCTION

HLT

//FETCH ANY INSTRUCTION
//loop

F1 (3 bit)

000	No transfer
001	P_OUT
010	MDR_OUT
011	Z_OUT
100	SRC_OUT
101	DST_OUT
110	SRCTMP_OUT
111	DSTTMP_OUT

F2 (3 bit)

000	No transfer
001	PC_IN
010	IR_IN
011	Z_IN
100	SRC_IN
101	DST_IN

F3 (2 bit)

00	No transfer
01	MAR_IN
10	MDR_IN

F4 (2 bit)

00	No transfer
01	Y_IN
10	SCRTMP_IN
11	DSTTMP_IN

F5 (5 bit)

00000	No action
00001	ADD
00010	SUB
00011	No action
00100	SBC
00101	AND
00110	OR
00111	XNOR
01000	ADC
10000	INC
10001	DEC
10010	CLR
10011	INV
10100	LSR
10101	ROR
10110	RRC
10111	ASR
11000	LSL
11001	ROL
11010	RLC

F6 (2 bit)

00	No action
01	READ
10	WRITE
11	IR_OFFSET

F7 (5 bit)

00000	NO OPERATION
00001	OR_OPERAND
00010	OR_2_OPERAND
00011	OR_BRANCH
00100	OR_DSTMOD
00101	OR_EXMOV
00110	OR_SRCIND
00111	OR_DSTIND
01000	OR_STOR
01001	OR_EXOP
01010	OR_EX2OP
01011	OR_EX1OP
01100	OR_CMP
01101	OR_BEQ
01110	OR_BNE
01111	OR_BLO
10000	OR_BLS
10001	OR_BHI
10010	OR_BHS
10011	OR_HLT

F8 NEXT ADDRESS (7 bit)

IR Structure

2 operand

OP CODE	MODE	SRC	MODE	DST

15……..…12 11……..…9 8……..…6 5……...……3 2……..0

1 operand

OP CODE	MODE	DST

Branch

OP CODE	OFFSET

Mode Opcode

Addressing modes

000	Register
001	Auto increment
010	Auto decrement
011	Index
100	Register indirect
101	Auto increment indirect
110	Auto decrement indirect
111	Index indirect

Instruction Opcode

Op code

ADC	**0**000
ADD	**0**001
SUB	**0**010
MOV	**0**011

SBC	**0**100
AND	**0**101
OR	**0**110
XNOR	**0**111
CMP	**101**0
INC	**110**0000000
DEC	**110**0000001
CLR	**110**0000010
INV	**110**0000011
LSR	**110**0000100
ROR	**110**0000101
RRC	**110**0000110
ASR	**110**0000111
LSL	**110**0001000
ROL	**110**0001001
RLC	**110**0001010
BR	**11110**000
BEQ	**11110**001
BNE	**11110**010
BLO	**11110**011
BLS	**11110**100
BHI	**11110**101
BHS	**11110**110
HLT	**1001**
NOP	**100**0

ROM

0000000	PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN	0000001
0000001	Z_OUT,PC_IN	0000010
0000010	MDR_OUT,IR_IN,**OR_OPERAND**	0000**100 bit or** (IR15 & IR14), (!IR15 + **!**IR13)
0000011	HLT	0000011
0000100	NOP,**OR_HLT**	0000**000 bit or** (IR12), (IR12)
0000101	2 operand,**OR_2_OPERAND**	0001**000 bit or** (IR11&**!**IR10&**!**IR9), (IR10), (IR9)
0000110	1 operand,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
0000111	branch,**OR_BRANCH**	01001**000 bit or** (IR10), (IR9), (IR8)
00**01**000	SRC_OUT,SRCTMP_IN,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
00**01**001	SRC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN	0010111
00**01**010	SRC_OUT,Y_IN,DEC,Z_IN	0011011
00**01**011	PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN	0001110
00**01**100	SRC_OUT,MAR_IN,READ	0001101
0001101	MDR_OUT,SRCTMP_IN,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
0001110	Z_OUT,PC_IN	0001111
0001111	SRC_OUT,Y_IN	0011010
00**10**000	DST_OUT,DSTTMP_IN,**OR_EXMOV**	G
00**10**001	DST_OUT,MAR_IN,READ,Y_IN,INC,Z_IN	0100011
00**10**010	DST_OUT,Y_IN,DEC,Z_IN	0100111
00**10**011	PC_OUT,MAR_IN,READ,Y_IN,INC,Z_IN	0010110

00**10**100	DTS_OUT,MAR_IN,READ	0010101
0010101	MDR_OUT,DSTTMP_IN,**OR_EXMOV**	G
0010110	Z_OUT,PC_IN	0011110

0010111	Z_OUT,SRC_IN,**OR_SRCIND**	0011100 **bit or** (! IR11)
0011000	MDR_OUT,MAR_IN,READ	0011001
0011001	MDR_OUT,SRCTMP_IN,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
0011010	MDR_OUT,ADD,Z_IN	0011111
0011011	Z_OUT,SRC_IN,MAR_IN,READ,**OR_SRCIND**	001110**0 bit or** (! IR11)
0011100	MDR_OUT,MAR_IN,READ	0011101
0011101	MDR_OUT,SRCTMP_IN,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
0011110	DST_OUT,Y_IN	0100010
0011111	Z_OUT,MAR_IN,READ,**OR_SRCIND**	010000**0 bit or** (! IR11)
0100000	MDR_OUT,MAR_IN,READ	0100001
0100001	MDR_OUT,SRCTMP_IN,**OR_DSTMOD**	0010**000 bit or** (IR5&**!**IR4&**!**IR3), (IR4), (IR3)
0100010	MDR_OUT,ADD,Z_IN	0100110
0100011	Z_OUT,DST_IN,**OR_DSTIND**	010010**0 bit or** (! IR5)
0100100	MDR_OUT,MAR_IN,READ	0100101
0100101	MDR_OUT,DSTTMP_IN,**OR_EXMOV**	G
0100110	Z_OUT,MAR_IN,READ,**OR_DSTIND**	010101**0 bit or** (! IR5)
0100111	Z_OUT,DST_IN,MAR_IN,READ,**OR_DSTIND**	010100**0 bit or** (! IR5)
0101000	MDR_OUT,MAR_IN,READ	0101001
0101001	MDR_OUT,DSTTMP_IN,**OR_EXMOV**	G

0101010	MDR_OUT,MAR_IN,READ	0101011
0101011	MDR_OUT,DSTTMP_IN,**OR_EXMOV**	G

0101100	END	000000
0101101	Z_OUT,DSTTMP_IN,**OR_STOR**	010111**0 bit or** (IR5||IR4||IR3)
0101110	DSTTMP_OUT,DST_IN	0101100
0101111	DSTTMP_OUT,MDR_IN,WRITE	0101100
0110000		
0110001		
0110010		
0110011	DSTTMP_OUT,Y_IN,**OR_EXOP**	011010**0 bit or** (IR15 & IR14)
0110100	2 OPERAND,**OR_EX2OP**	1000**000 bit or** (IR14), (IR13), (IR12)
0110101	1 OPERAND,**OR_EX1OP**	101**0000 bit or** (IR9), (IR8), (IR7), (IR6)
0110110		
0110111		
0111000		
0111001		
0111010		
0111011		
0111100		
0111101		
0111110		
0111111		

1000000	SRCTMP_OUT,ADC,Z_IN	0101101
1000001	ADD	0101101
1000010	SRCTMP_OUT,SUB,Z_IN,**OR_CMP**	010110**0 bit or** (!IR15)
1000011	SRCTMP_OUT,DSTTMP_IN,**OR_STOR**	010111**0 bit or** (IR5||IR4||IR3)
1000100	SBC	0101101
1000101	AND	0101101
1000110	OR	0101101
1000111	XNOR	0101101
1001000	IR_OFFSET,Y_IN //BR1	1011100
1001001	IR_OFFSET,Y_IN,**OR_BEQ** //BEQ	**000**1100 **bit or** (z), (!z), (z)
1001010	IR_OFFSET,Y_IN,**OR_BNE** //BNE	**000**1100 **bit or** (!z), (z), (!z)
1001011	IR_OFFSET,Y_IN,**OR_BLO** //BLO	**000**1100 **bit or** (!c), (c), (!c)
1001100	IR_OFFSET,Y_IN,**OR_BLS** //BLS	**000**1100 **bit or** (z||!c), (!z&c), (z||!c)
1001101	IR_OFFSET,Y_IN,**OR_BHI** //BHI	**000**1100 **bit or** (c), (!c), (c)
1001110	IR_OFFSET,Y_IN,**OR_BHS** //BHS	**000**1100 **bit or** (z||c), (!z&!c), (z||c)
1001111		
1010000	INC	0101101
1010001	DEC	0101101
1010010	CLR	0101101
1010011	INV	0101101
1010100	LSR	0101101
1010101	ROR	0101101
1010110	RRC	0101101
1010111	ASR	0101101

1011000	LSL	0101101
1011001	ROL	0101101
1011010	RLC	0101101
1011011		
1011100	PC_OUT,ADD,Z_IN	1011101
1011101	Z_OUT,PC_IN	0101100

G=> **000**0011 bit or (! IR15 & ! IR14 & IR13 & IR12), !(! IR15 & ! IR14 & IR13 & IR12), !(! IR15 & ! IR14 & IR13 & IR12)

Analysis

ADD,ADC,SUB,SBC,AND,OR,XNOR

FETCH INSTRUCTION Clocks #4, Memory access #1
EXECUTE Clocks #3, Memory access #0
STORE DST Clocks #1 ,(Memory access #0 OR Memory access #1)
ENDClocks #1, Memory access #0

FETCH (SRC/DST)Clocks #1 ,Memory access #0
FETCH (SRC/DST)+,-(SRC/DST) Clocks #3, Memory access #1
FETCH @(SRC/DST)Clocks #2, Memory access #1
FETCH @(SRC/DST)+,@-(SRC/DST)Clocks #4, Memory access #2
FETCH X(SRC/DST)Clocks #6, Memory access #2
FETCH @X(SRC/DST)Clocks #7, Memory access #3

SRC,DST Clocks #9+1+1=11,Memory access #1
SRC,(DST+,-DST) Clocks #9+1+3=13,Memory access #1+1+1=3
SRC,@DST Clocks #9+1+2=12,Memory access #1+1+1=3
SRC,(@DST+,@-DST) Clocks #9+1+4=14,Memory access #1+1+2=4
SRC,XDST Clocks #9+1+6=16,Memory access #1+1+2=4
SRC,@XDST Clocks #9+1+7=17,Memory access #1+1+3=5

(SRC+,-SRC),DST Clocks #9+3+1=13,Memory access #1+1=2
(SRC+,-SRC),(DST+,-DST) Clocks #9+3+3=15,Memory access #1+1+1+1=4
(SRC+,-SRC),@DST Clocks #9+3+2=14,Memory access #1+1+1+1=4
(SRC+,-SRC),(@DST+,@-DST) Clocks #9+3+4=16,Memory access #1+1+1+2=5
(SRC+,-SRC),XDST Clocks #9+3+6=18,Memory access #1+1+1+2=5
(SRC+,-SRC),@XDST Clocks #9+3+7=19,Memory access #1+1+1+3=6

@SRC,DST Clocks #9+2+1=12,Memory access #1+1=2
@SRC,(DST+,-DST) Clocks #9+2+3=14,Memory access #1+1+1+1=4
@SRC,@DST Clocks #9+2+2=13,Memory access #1+1+1+1=4
@SRC,(@DST+,@-DST) Clocks #9+2+4=15,Memory access #1+1+1+2=5
@SRC,XDST Clocks #9+2+6=17,Memory access #1+1+1+2=5
@SRC,@XDST Clocks #9+2+7=18,Memory access #1+1+1+3=6

(@SRC+@,-SRC),DST Clocks #9+4+1=14,Memory access #1+2=3
(@SRC+@,-SRC),(DST+,-DST) Clocks #9+4+3=16,Memory access #1+2+1+1=5
(@SRC+@,-SRC),@DST Clocks #9+4+2=15,Memory access #1+2+1+1=5
(@SRC+@,-SRC),(@DST+,@-DST) Clocks #9+4+4=17,Memory access #1+2+1+2=6
(@SRC+@,-SRC),XDST Clocks #9+4+6=19,Memory access #1+2+1+2=6
(@SRC+@,-SRC),@XDST Clocks #9+4+7=20,Memory access #1+2+1+3=7

XSRC,DST Clocks #9+6+1=16,Memory access #1+2=3
XSRC,(DST+,-DST) Clocks #9+6+3=18,Memory access #1+2+1+1=5
XSRC,@DST Clocks #9+6+2=17,Memory access #1+2+1+1=5
XSRC,(@DST+,@-DST) Clocks #9+6+4=19,Memory access #1+2+1+2=6
XSRC,XDST Clocks #9+6+6=21,Memory access #1+2+1+2=6
XSRC,@XDST Clocks #9+6+7=22,Memory access #1+2+1+3=7

@XSRC,DST Clocks #9+7+1=17,Memory access #1+3=4
@XSRC,(DST+,-DST) Clocks #9+7+3=19,Memory access #1+3+1+1=6
@XSRC,@DST Clocks #9+7+2=18,Memory access #1+3+1+1=6
@XSRC,(@DST+,@-DST) Clocks #9+7+4=20,Memory access #1+3+1+2=7
@XSRC,XDST Clocks #9+7+6=22,Memory access #1+3+1+2=7
@XSRC,@XDST Clocks #9+7+7=23,Memory access #1+3+1+3=8

CMP
FETCH INSTRUCTION Clocks #4, Memory access #1
EXECUTE Clocks #3, Memory access #0
ENDClocks #1, Memory access #0

FETCH (SRC/DST)Clocks #1 ,Memory access #0
FETCH (SRC/DST)+,-(SRC/DST) Clocks #3, Memory access #1
FETCH @(SRC/DST)Clocks #2, Memory access #1
FETCH @(SRC/DST)+,@-(SRC/DST)Clocks #4, Memory access #2
FETCH X(SRC/DST)Clocks #6, Memory access #2
FETCH @X(SRC/DST)Clocks #7, Memory access #3

SRC,DST Clocks #8+1+1=10,Memory access #1
SRC,(DST+,-DST) Clocks #8+1+3=12,Memory access #1+1=2
SRC,@DST Clocks #8+1+2=11,Memory access #1+1=2
SRC,(@DST+,@-DST) Clocks #8+1+4=13,Memory access #1+2=3
SRC,XDST Clocks #8+1+6=15,Memory access #1+2=3
SRC,@XDST Clocks #8+1+7=16,Memory access #1+3=4

(SRC+,-SRC),DST Clocks #8+3+1=12,Memory access #1+1=2
(SRC+,-SRC),(DST+,-DST) Clocks #8+3+3=14,Memory access #1+1+1=3
(SRC+,-SRC),@DST Clocks #8+3+2=13,Memory access #1+1+1=3
(SRC+,-SRC),(@DST+,@-DST) Clocks #8+3+4=15,Memory access #1+1+2=4
(SRC+,-SRC),XDST Clocks #8+3+6=17,Memory access #1+1+2=4
(SRC+,-SRC),@XDST Clocks #8+3+7=18,Memory access #1+1+3=5

@SRC,DST Clocks #8+2+1=11,Memory access #1+1=2
@SRC,(DST+,-DST) Clocks #8+2+3=13,Memory access #1+1+1=3
@SRC,@DST Clocks #8+2+2=12,Memory access #1+1+1=3
@SRC,(@DST+,@-DST) Clocks #8+2+4=14,Memory access #1+1+2=4
@SRC,XDST Clocks #8+2+6=16,Memory access #1+1+2=4
@SRC,@XDST Clocks #8+2+7=17,Memory access #1+1+3=5

(@SRC+@,-SRC),DST Clocks #8+4+1=13,Memory access #1+2=3
(@SRC+@,-SRC),(DST+,-DST) Clocks #8+4+3=15,Memory access #1+2+1=4
(@SRC+@,-SRC),@DST Clocks #8+4+2=14,Memory access #1+2+1=4
(@SRC+@,-SRC),(@DST+,@-DST) Clocks #8+4+4=16,Memory access #1+2+2=5
(@SRC+@,-SRC),XDST Clocks #8+4+6=18,Memory access #1+2+2=5
(@SRC+@,-SRC),@XDST Clocks #8+4+7=19,Memory access #1+2+3=6

XSRC,DST Clocks #8+6+1=15,Memory access #1+2=3
XSRC,(DST+,-DST) Clocks #8+6+3=17,Memory access #1+2+1=4
XSRC,@DST Clocks #8+6+2=16,Memory access #1+2+1=4
XSRC,(@DST+,@-DST) Clocks #8+6+4=18,Memory access #1+2+2=5
XSRC,XDST Clocks #8+6+6=20,Memory access #1+2+2=5
XSRC,@XDST Clocks #8+6+7=21,Memory access #1+2+3=6

@XSRC,DST Clocks #8+7+1=16,Memory access #1+3=4
@XSRC,(DST+,-DST) Clocks #8+7+3=18,Memory access #1+3+1=5
@XSRC,@DST Clocks #8+7+2=17,Memory access #1+3+1=5
@XSRC,(@DST+,@-DST) Clocks #8+7+4=19,Memory access #1+3+2=6
@XSRC,XDST Clocks #8+7+6=21,Memory access #1+3+2=6
@XSRC,@XDST Clocks #8+7+7=22,Memory access #1+3+3=7

MOV

FETCH INSTRUCTION Clocks #4, Memory access #1
EXECUTE Clocks #1, Memory access #0
STORE DST Clocks #1 ,(Memory access #0 OR Memory access #1)
ENDClocks #1, Memory access #0

FETCH (SRC/DST)Clocks #1 ,Memory access #0
FETCH (SRC/DST)+,-(SRC/DST) Clocks #3, Memory access #1
FETCH @(SRC/DST)Clocks #2, Memory access #1
FETCH @(SRC/DST)+,@-(SRC/DST)Clocks #4, Memory access #2
FETCH X(SRC/DST)Clocks #6, Memory access #2
FETCH @X(SRC/DST)Clocks #7, Memory access #3

SRC,DST Clocks #7+1+1=9,Memory access #1
SRC,(DST+,-DST) Clocks #7+1+3=11,Memory access #1+1+1=3
SRC,@DST Clocks #7+1+2=10,Memory access #1+1+1=3
SRC,(@DST+,@-DST) Clocks #7+1+4=12,Memory access #1+1+2=4
SRC,XDST Clocks #7+1+6=14,Memory access #1+1+2=4
SRC,@XDST Clocks #7+1+7=15,Memory access #1+1+3=5

(SRC+,-SRC),DST Clocks #7+3+1=11,Memory access #1+1=2
(SRC+,-SRC),(DST+,-DST) Clocks #7+3+3=13,Memory access #1+1+1+1=4
(SRC+,-SRC),@DST Clocks #7+3+2=12,Memory access #1+1+1+1=4
(SRC+,-SRC),(@DST+,@-DST) Clocks #7+3+4=14,Memory access #1+1+1+2=5
(SRC+,-SRC),XDST Clocks #7+3+6=16,Memory access #1+1+1+2=5
(SRC+,-SRC),@XDST Clocks #7+3+7=17,Memory access #1+1+1+3=6

@SRC,DST Clocks #7+2+1=10,Memory access #1+1=2
@SRC,(DST+,-DST) Clocks #7+2+3=12,Memory access #1+1+1+1=4
@SRC,@DST Clocks #7+2+2=11,Memory access #1+1+1+1=4
@SRC,(@DST+,@-DST) Clocks #7+2+4=13,Memory access #1+1+1+2=5
@SRC,XDST Clocks #7+2+6=15,Memory access #1+1+1+2=5
@SRC,@XDST Clocks #7+2+7=16,Memory access #1+1+1+3=6

(@SRC+@,-SRC),DST Clocks #7+4+1=12,Memory access #1+2=3
(@SRC+@,-SRC),(DST+,-DST) Clocks #7+4+3=14,Memory access #1+2+1+1=5
(@SRC+@,-SRC),@DST Clocks #7+4+2=13,Memory access #1+2+1+1=5
(@SRC+@,-SRC),(@DST+,@-DST) Clocks #7+4+4=15,Memory access #1+2+1+2=6
(@SRC+@,-SRC),XDST Clocks #7+4+6=17,Memory access #1+2+1+2=6
(@SRC+@,-SRC),@XDST Clocks #7+4+7=18,Memory access #1+2+1+3=7

XSRC,DST Clocks #7+6+1=14,Memory access #1+2=3
XSRC,(DST+,-DST) Clocks #7+6+3=16,Memory access #1+2+1+1=5
XSRC,@DST Clocks #7+6+2=15,Memory access #1+2+1+1=5
XSRC,(@DST+,@-DST) Clocks #7+6+4=17,Memory access #1+2+1+2=6
XSRC,XDST Clocks #7+6+6=19,Memory access #1+2+1+2=6
XSRC,@XDST Clocks #7+6+7=20,Memory access #1+2+1+3=7

@XSRC,DST Clocks #7+7+1=15,Memory access #1+3=4
@XSRC,(DST+,-DST) Clocks #7+7+3=17,Memory access #1+3+1+1=6
@XSRC,@DST Clocks #7+7+2=16,Memory access #1+3+1+1=6
@XSRC,(@DST+,@-DST) Clocks #7+7+4=18,Memory access #1+3+1+2=7
@XSRC,XDST Clocks #7+7+6=20,Memory access #1+3+1+2=7
@XSRC,@XDST Clocks #7+7+7=21,Memory access #1+3+1+3=8
INC,DEC,CLR,INV,LSR,ROR,RRC,ASR,LSL,ROL,RLC

FETCH INSTRUCTION Clocks #4, Memory access #1
EXECUTE Clocks #3, Memory access #0
STORE DST Clocks #1 ,(Memory access #0 OR Memory access #1)
ENDClocks #1, Memory access #0

FETCH (SRC/DST)Clocks #1 ,Memory access #0
FETCH (SRC/DST)+,-(SRC/DST) Clocks #3, Memory access #1
FETCH @(SRC/DST)Clocks #2, Memory access #1
FETCH @(SRC/DST)+,@-(SRC/DST)Clocks #4, Memory access #2
FETCH X(SRC/DST)Clocks #6, Memory access #2
FETCH @X(SRC/DST)Clocks #7, Memory access #3

DST Clocks #9+1=10,Memory access #1
(DST+,-DST) Clocks #9+3=12,Memory access #1+1+1=3
@DST Clocks #9+2=11,Memory access #1+1+1=3
(@DST+,@-DST) Clocks #9+4=13,Memory access #1+1+2=4
XDST Clocks #9+6=15,Memory access #1+1+2=4
@XDST Clocks #9+7=16,Memory access #1+1+3=5

BR,BEQ,BNE,BLO,BLS,BHI,BHS
FETCH INSTRUCTION Clocks #4, Memory access #1
EXECUTE Clocks #3, Memory access #0
ENDClocks #1, Memory access #0

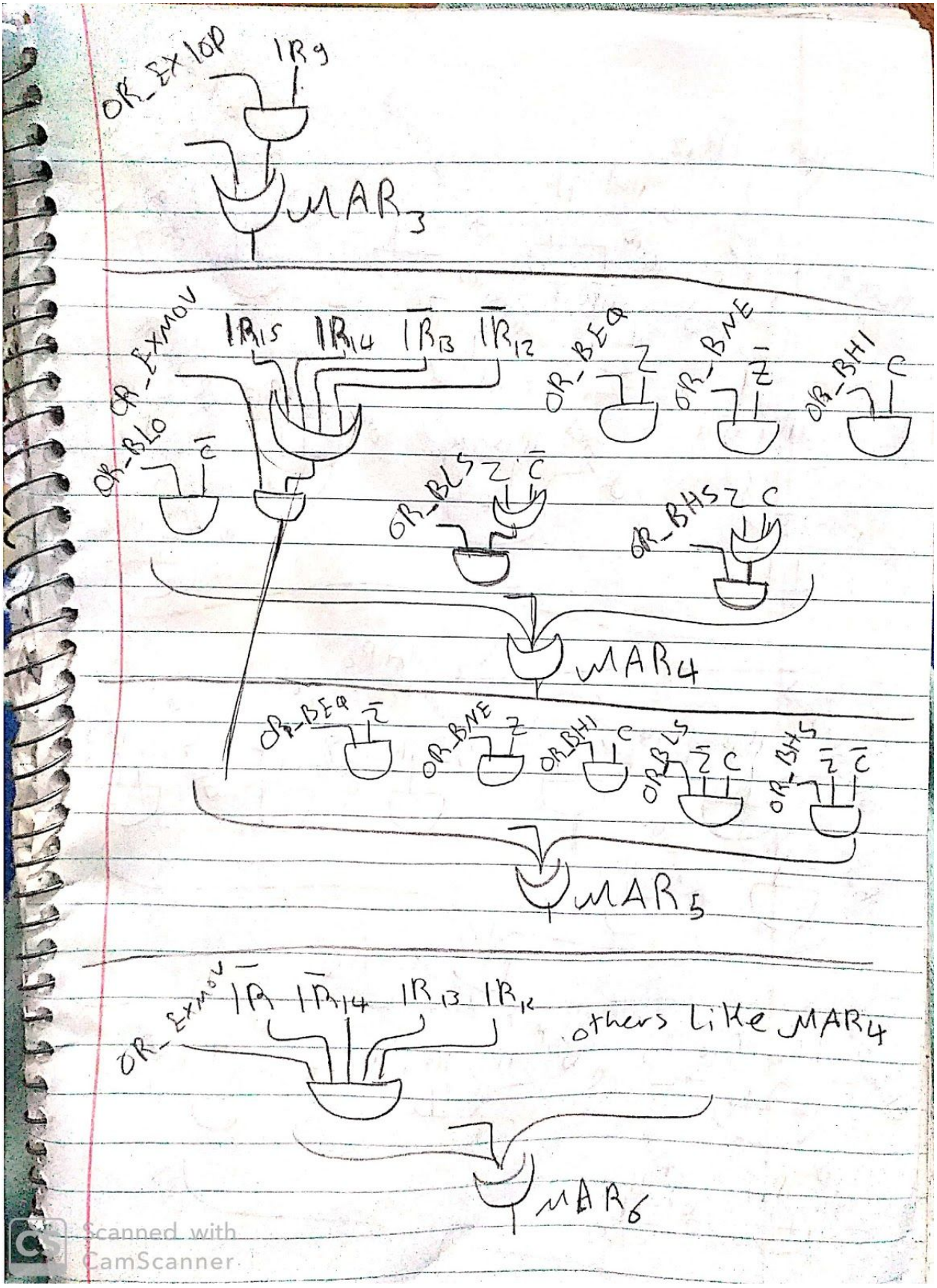
NOP,HLT
FETCH INSTRUCTION Clocks #4, Memory access #1

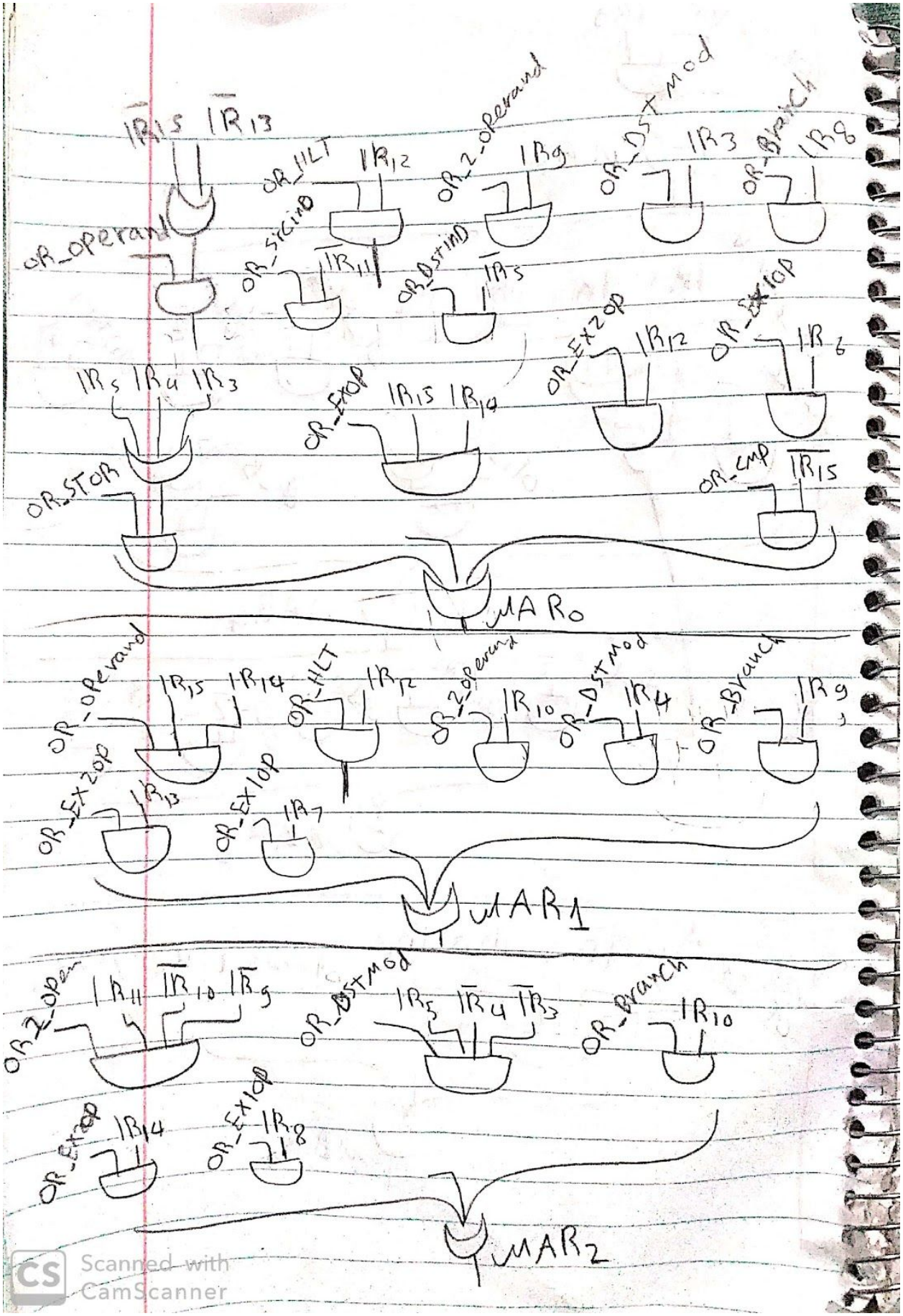
Cycles per Instruction (CPI)

Number of instructions =
448(ADD,ADC..)+64(CMP)+64(MOV)+88(INC,DEC..)+7(BR,BEQ..)+2(NOP,HLT)
= 673 instruction

Sum(clock cycles of all instructions) =
7392(ADD,ADC..)+992(CMP)+928(MOV)+1122(INC,DEC..)+56(BR,BEQ..)+8(NOP,HLT)
= 10498 clock

Average number of clock cycles per instruction= 10498/673
= 15.59881129 clock per instruction




$\overline{IR_{15}}$ $\overline{IR_{13}}$
OR_operand
OR_HLT
IR_{12}
OR_SIGIND
$\overline{IR_{11}}$
OR_2_operand
IR_9
OR_DSTIND
$\overline{IR_5}$
OR_DSTMod
IR_3
OR_Branch
IR_8
IR_5 IR_4 IR_3
OR_STOR
OR_EXOP
IR_{15} IR_{14}
OR_EX2OP
IR_{12}
OR_EX1OP
IR_6
OR_CMP
$\overline{IR_{15}}$
MAR0
OR_Operand
IR_{15} IR_{14}
OR_EX2OP
IR_{13}
OR_HLT
IR_{12}
OR_2Operand
IR_{10}
OR_DSTMod
IR_4
OR_Branch
IR_9
OR_EX1OP
IR_7
MAR1
OR_2Oper
IR_{11} $\overline{IR_{10}}$ $\overline{IR_9}$
OR_DSTMod
IR_5 $\overline{IR_4}$ $\overline{IR_3}$
OR_Branch
IR_{10}
OR_EXOP
IR_{14}
OR_EX1OP
IR_8
MAR2