Instruction Set H08A
User's Manual

H08 A Instruction Index

Instruction		Description	Cycles	Status Affected	Ref Page
BYTE-ORIENTED FILE REGISTER OPERATIONS					
ADDC	f,d,a	Add W, F and C, and place the result to W or F.	1	C,DC,N,OV,Z	**6**
ADDF	f,d,a	Add W and F, and place the result to W or F.	1	C,DC,N,OV,Z	**7**
ADDL	k	Add constant k and W, and place the result to W.	1	C,DC,N,OV,Z	**9**
ANDF	f,d,a	AND W and F, and place the result to W or F	1	N,Z	**10**
ANDL	k	AND constant k and W, and place the result to W.	1	N,Z	**11**
ARLC	f,d,a	Rotate left F value and C and place the result to W or F.	1	C,N,OV,Z	**12**
ARRC	f,d,a	Rotate right F value, MSB remains unchanged, moves LSB to C.	1	C,N,Z	**13**
CLRF	f,a	Clear F contents to 0.	1	None	**20**
COMF	f,d,a	Complement F value and place the result to W or F.	1	N,Z	**21**
CPSE	f,a	If F=W, skip the next instruction.	1(2)(3)	None	**22**
CPSG	f,a	If F>W, skip the next instruction.	1(2)(3)	None	**23**
CPSL	f,a	If F<W, skip the next instruction.	1(2)(3)	None	**24**
DCF	f,d,a	Subtract 1 of F value and place the result to W or F.	1	C,DC,N,OV,Z	**26**
DCSUZ	f,d,a	If subtracts 1 of F value, the value≠0, skip the next instruction and place the result to W or F.	1(2)(3)	None	**28**
DCSZ	f,d,a	If subtracts 1 of F value, the value is 0, skip the next instruction and place the result to W or F.	1(2)(3)	None	**29**
INF	f,d,a	Add 1 to F value, and place the result to W or F.	1	C,DC,N,OV,Z	**31**
INSUZ	f,d,a	Add 1 to F value, if the value≠0, skip the next instruction and place the result to W or F.	1(2)(3)	None	**32**
INSZ	f,d,a	Add 1 to F value, if the value =0, skip the next instruction and place the result to W or F.	1(2)(3)	None	**33**
IORF	f,d,a	Inclusive OR W and F, and place the result to W or F.	1	N,Z	**34**
IORL	k	OR constant k and w, and place the result to W.	1	N,Z	**35**
LBSR	k	Move constant k to register BSRCN.	1	None	**45**
LDPR	k,f	Move constant k (9-bit) to register FSR (f = 0 ~ 1).	2	None	**46**
MULF	f,a	Multiply W and F.	2	None	**47**
MULL	k	Do multiplication of constant k and W.	2	None	**48**
MVF	f,d,a	Move W value to F(d=1) or move F value to W(d=0).	1	None	**49**
MVFF	fs,fd	Move Fs data to Fd.	2	None	**50**
MVL	k	Move constant k to W.	1	None	**51**
RETL	k	Place top-of-stack value to PC, and configure W value as k. Main program will be executed from current PC value.	2	None	**58**
RLF	f,d,a	Rotate left F value and place the result to W or F.	1	N,Z	**60**
RLFC	f,d,a	Rotate left F value and C and place the result to W or F.	1	C,N,Z	**61**

H08 A Instruction Index (continued)

Instruction		Description	Cycles	Status Affected	Ref Page
BYTE-ORIENTED FILE REGISTER OPERATIONS					
RRF	f,d,a	Rotate right F value and place the result to W or F.	1	N,Z	**62**
RRFC	f,d,a	Rotate right F value and C and place the result to W or F.	1	C,N,Z	**63**
SETF	f,a	Configure F value as 0xFF.	1	None	**64**
SUBC	f,d,a	Subtract W of F value and reverse C and place the result to W or F.	1	C,DC,N,OV,Z	**66**
SUBF	f,d,a	Subtract W of F value and place the result to W or F.	1	C,DC,N,OV,Z	**67**
SUBL	k	Subtract constant k and W and place the result to W.	1	C,DC,N,OV,Z	**68**
SWPF	f,d,a	Switch the high and low 4 bit of F value and place the result to W or F.	1	None	**69**
TFSZ	f,a	Test if F value equals to 0. If=0, skip the next instruction.	1(2)(3)	None	**70**
XORF	f,d,a	Exclusive OR W and F and place the result to W or F.	1	N,Z	**72**
XORL	k	XOR constant k and W and place the result to W.	1	N,Z	**73**

Remark				
	f	register	b	Register b bit
	n	Memory address	k	8 bit constant
	d	Data stored place; d = 0 means it is saved in accumulator W; d = 1 means it is saved in register f.		
	a	Memory address where data is stored, a=0 means saved in current memory address； a=1 means it is saved in the appointed memory address of register BSRCN.		

H08 A Instruction Index (continued)

Instruction		Description	Cycles	Status Affected	Ref Page
CONTROL OPERATIONS					
CALL	n,s	Save the PC value of next instruction to the top-of-stack and jump to n. ◦	2	None	**19**
CWDT		Clear watch dog timer as 0.	1	TO	**25**
IDLE		Get access to idle mode.	1	IdleB	**30**
JC	n	If C = 1, jump to address n.	1(2)	None	**36**
JMP	n	Unconditionally jump to address n.	2	None	**37**
JN	n	If N = 1, jump to address n.	1(2)	None	**38**
JNC	n	If C = 0, jump to address n.	1(2)	None	**39**
JNN	n	If N = 0, jump to address n.	1(2)	None	**40**
JNO	n	If OV = 0, jump to address n.	1(2)	None	**41**
JNZ	n	If Z = 0, jump to address n.	1(2)	None	**42**
JO	n	If OV = 1, jump to address n.	1(2)	None	**43**
JZ	n	If Z = 1, jump to address n.	1(2)	None	**44**
NOP		Blank instruction.	1	None	**53**
POP		Subtract 1 of stack pointer register, read out the pointed stack value to register, TOS.	1	None	**54**
RCALL	n	Save the PC value of instruction to top-of-stack and jump to address: n,-1024≦n≦1023	2	None	**55**
RET	s	Return from vice program and read the top-of-stack value to PC and the main program is executed from the current PC value.	2	None	**56**
RETI	s	Return form interrupt and read the top-of-stack value to PC, and the main program will be executed form current PC value.	2	GIE	**57**
RJ	n	Unconditionally jump to n,-1024≦n≦1023	2	None	**59**
SLP	f,a	Go to sleep mode.		PD	**65**

Remark				
	f	register	b	Register b bit
	n	Memory address	k	8 bit constant
	d	Data stored place; d = 0 means it is saved in accumulator W; d = 1 means it is saved in register f.		
	a	Memory address where data is stored, a=0 means saved in current memory address ; a=1 means it is saved in the appointed memory address of register BSRCN.		

H08 A Instruction Index (continued)

Instruction		Description	Cycles	Status Affected	Ref Page
BIT-ORIENTED FILE REGISTER OPERATIONS					
BCF	f,b,a	Configure a specific bit of F as 0.	1	None	**14**
BSF	f,b,a	Configure a specific bit of F as 1.	1	None	**15**
BTGF	f,b,a	NOT a specific bit of F.	1	None	**16**
BTSS	f,b,a	Test if a specific bit of F equals to 1. If=1, skip the next instruction.	1(2)(3)	None	**17**
BTSZ	f,b,a	Test if a specific bit of F equals to 0. If=0, skip the next instruction.	1(2)(3)	None	**18**
PROGRAM MEMORY OPERATIONS					
MVLP	k	Move constant k (0≦k≦16384) to TABLE pointer (TBLPTRH/TBLPTRL).	2	None	**52**
TBLR	*	Make the contents of TBLPTR as address pointer, read program memory contents to register, TBLDH/TBLDL.	2	None	**70**
TBLR	*+	Make the contents of TBLPTR as address pointer, read program memory contents to register, TBLDH/TBLDL. And +1 to address pointer.	2	None	**70**

Remark

f	register	b	Register b bit
n	Memory address	k	8 bit constant
d	Data stored place; d = 0 means it is saved in accumulator W; d = 1 means it is saved in register f.		
a	Memory address where data is stored, a=0 means saved in current memory address；a=1 means it is saved in the appointed memory address of register BSRCN.		

ADDC ADD w and Carry bit to f

Syntax: ADDC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0)

Operation: (W) + (f) + (Status<C>) → destination

Status Affected: C, DC, N, OV, Z

Description: Add accumulator W value, register f value and carry flag C together and place the result to d appointed register；

If d = 0, the operation result will be placed into accumulator W ；

If d = 1, the operation result will be placed into register f ；

If a = 0, the operation result will be placed into current RAM address ；

If a=1, the operation result will be placed to register BSRCN appointed RAM address.

Words: 1

Cycles: 1

Example 1: ADDC REG, 0, 0

Before Instruction:	After Instruction:
W=001H	W=020H
REG(080H)=01FH	REG(080H)=01FH
C=DC=N=OV=Z=0	DC=1, C= N=OV=Z=0

Remark: d=0, the execution result will be placed into accumulator W.

Example 2: ADDC REG, 1, 1 (if BSRCN=001H)

Before Instruction:	After Instruction:
W=001H	W=001H
REG(170H)=00EH	REG(170H)=010H
C=1, DC=N=OV=Z=0	DC=1, C= N=OV=Z=0

Remark: d=1, the execution result will be placed into register f.
a=1, the result will be placed back into register BSRCN appointed RAM address.

ADDF ADD w to F

Syntax: ADDF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0)

Operation: (W) + (f) → destination

Status Affected: C, DC, N, OV, Z

Description: Add accumulator W value and register f value together, then place the result to d appointed register ;

If d = 0, the operation result will be placed into accumulator W ;

If d = 1, the operation result will be placed into register f ;

If a = 0, the operation result will be placed into RAM address ;

If a=1, the operation result will be placed into appointed RAM address of register BSRCN.

Words: 1

Cycles: 1

Example 1: ADDF REG, 0, 0

Before Instruction:

W=001H

REG(080H)=01FH

C=DC=N=OV=Z=0

After Instruction:

W=020H

REG(080H)=01FH

DC=1, C= N=OV=Z=0

Remark: d=0, the execution result will be placed into accumulator W.
a=0 is default value. If program a=0, then this argument can not be added into this program.

Example 2: ADDF REG, 1

Before Instruction:

W=001H

REG(080H)=01FH

C=DC=N=OV=Z=0

After Instruction:

W=001H

REG(080H)=020H

DC=1, C= N=OV=Z=0

Remark: d=1, the execution result will be placed into register f.
The default value is a=0. If program a=0, then this argument can not be added into this program.

Example 3: ADDF REG, 1, 1 (if BSRCN=001H)

Before Instruction:

W=001H

REG(070H)=00EH

REG1(170H)=01FH

C=DC=N=OV=Z=0

After Instruction:

W=001H

REG(070H)=00EH

REG1(170H)=020H

DC=1, C= N=OV=Z=0

Remark: d=1, the result will be placed into register f.

a=1, the result will be placed into register BSRCN appointed RAM address.

Though REG RAM address is 070H, BSRCN=001H will be operated with registe170H value and

the result will be placed to address, 170H.

ADDL ADD Literal to w

Syntax: ADDL k

Operands: $0 \leqq k \leqq 255$

Operation: (W) + K → W

Status Affected: C, DC, N, OV, Z

Description: Add accumulator W value and k value, and place the result to accumulator W.

Words: 1

Cycles: 1

Example 1: ADDL 00FH

Before Instruction:

W=001H

C=DC=N=OV=Z=0

After Instruction:

W=010H

DC=1, C= N=OV=Z=0

Remark: Add low 4 bit together will generates carry bit. Half carry bit flag, DC=1.

Example 2: ADDL 00FH

Before Instruction:

W=071H

C=DC=N=OV=Z=0

After Instruction:

W=080H

DC=N=OV=1, C=Z=0

Remark: Add low 4 bit together will generates carry bit. Half carry bit flag, DC=1.
If the result>127, it is deemed as negative, negative flag N=1.
After execution, bit 7=1, overflow flag OV=1.

Example 3: ADDL 00FH

Before Instruction:

W=081H

C=DC=N=OV=Z=0

After Instruction:

W=090H

DC=N=1, C=OV=Z=0

Remark: Add low 4 bit together will generates carry bit. Half carry bit flag, DC=1.
If the result>127, it is deemed as negative, negative flag N=1.
Before execution, bit 7=1. After execution, bit 7 is 1, overflow flag remains unchanged, OV=0.

Example 4: ADDL 00FH

Before Instruction:

W=0F1H

C=DC=N=OV=Z=0

After Instruction:

W=000H

C=DC= Z=1, N=OV= 0

Remark: The execution result > 0FFH, carry flag, C=1. Add low 4 bit together will generates carry bit.
Half carry bit flag, DC=1.
If the result is 000H, the zero flag, Z=1.

ANDF AND w with F

Syntax: ANDF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (W) AND (f) → destination

Status Affected: N, Z

Description: Logic AND accumulator W value and register f value, and place the result to d appointed register.

If d = 0, place the result to accumulator W ;

If d = 1, place the result to register f ;

If a = 0, place the result to RAM address ;

If a=1, place the result to appointed RAM address of register BSRCN.

Words: 1

Cycles: 1

Example 1: ANDF REG, 0

Before Instruction:	After Instruction:
W=055H	W=000H
REG(080H)=0AAH	REG(080H)=0AAH
C=DC=N=OV=Z=0	Z=1, C=DC=N=OV=0

Remark: If the result is 000H, the zero flag, Z=1. If the default value is a=0, when a=0, it is alright to exclude this argument into the program.

Example 2: ANDF REG, 1, 1 (if BSRCN=001H)

Before Instruction:	After Instruction:
W=080H	W=080H
REG(070H)=0FFH	REG(070H)=080H
C=DC=N=OV=Z=0	N=1, C=DC= OV=Z=0

Remark: If the result>127, it is deemed as negative. Negative flag N=1.

ANDL AND Literal with w

Syntax: ANDL k

Operands: $0 \leqq k \leqq 255$

Operation: (W) AND k → W

Status Affected: N, Z

Description: Logic AND accumulator W value and k value, and place the result to accumulator W.

Words: 1

Cycles: 1

Example 1: ANDL 0A0H

Before Instruction:

W=055H

C=DC=N=OV=Z=0

After Instruction:

W=000H
Z=1, C=DC=N=OV=0

Remark: The result is 000H, zero flag is Z=1.

Example 2: ANDL 0FF0H

Before Instruction:

W=080H

C=DC=N=OV=Z=0

After Instruction:

W=080H
N=1, C=DC= OV=Z=0

Remark: If the result>127, it is deemed as negative. Negative flag N=1.

ARLC Another Rotate Left f through Carry

Syntax: ARLC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<n>) → destination <n+1 >, (f<7>) → Status< C >,

Status< C > → destination < 0 >

Status Affected: C, N, OV, Z

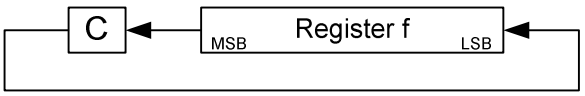
Description: Rotate left register f value and carry flag C (this instruction is the same as RLFC function, only differs in OV flag)

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to current RAM address ;

If a = 1, the result is placed to appointed RAM address of BSRCN register.



Words: 1

Cycles: 1

Example 1: ARLC REG, 1, 0

Before Instruction:	After Instruction:
WREG(02CH)=00FH	WREG(02CH)=00FH
REG(080H)=0AAH	REG(080H)=054H
C=N=OV=Z=0	C=OV=1, N=Z=0

Remark: BIT7 result moves from 1→0, so overflow flag, OV=1.

Example 2: ARLC REG, 0, 1

Before Instruction:	After Instruction:
WREG(02CH)=0FH	WREG(02CH)=0D4H
REG(170H)=0EAH	REG(170H)=0EAH
C=N=OV=Z=0	C=N=1, OV=Z=0

Example 3: ARLC REG, 1, 1

Before Instruction:	After Instruction:
WREG(02CH)=00FH	WREG(02CH)=00FH
REG(170H)=080H	REG(170H)=000H
C=N=OV=Z=0	C=OV=Z=1, N=0

ARRC Another Rotate Right f through Carry

Syntax: ARRC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<n>) → destination <n-1 >, (f<7>) → destination < 7 >,

(f<0>) → Status< C >

Status Affected: C, N, Z

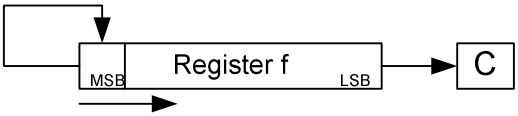
Description: Rotate right register f value, rotate right BIT0 value to carry flag C, BIT7 stays in BIT7 address.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to current RAM address ;

If a = 1, the result is placed to appointed RAM address of BSRCN register.



Words: 1

Cycles: 1

Example 1: ARRC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(080H)=0D5H

N=1, C= Z=0

Example 2: ARRC REG, 0, 1

Before Instruction:

WREG(02CH)=00FH

REG(17FH)=055H

C=N=Z=0

After Instruction:

WREG(02CH)=02AH

REG(17FH)=055H

C=1, N=Z=0

Example 3: ARRC REG, 1, 1

Before Instruction:

WREG(02CH)=00FH

REG(17FH)=001H

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(17FH)=000H

C=Z=1, N=0

BCF Bit Clear F

Syntax: BCF f, b, a

Operands: 0 ≦ f ≦ 255 ; 0 ≦ b ≦ 7 ; a € (0, 1)

Operation: 0 → f < b >

Status Affected: None

Description: Clear register f configuration as 0.

Words: 1

Cycles: 1

Example 1: BCF REG,2

Before Instruction:

REG(080H)=1111 1111B

After Instruction:

REG(080H)=1111 1011B

Remark: Clear BIT2 of register REG as 0, other bits remain unchanged.

BSF Bit Set F

Syntax: BCF f, b, a

Operands: 0 ≦ f ≦ 255 ; 0 ≦ b ≦ 7 ; a € (0, 1)

Operation: 1 → f < b >

Status Affected: None

Description: Configure register f value as 1.

Words: 1

Cycles: 1

Example 1: BSF REG,2

Before Instruction:

REG(080H)=00000000B

After Instruction:

REG(080H)=00000100B

Remark: Configure BIT2 of REG as 1, other bits remain unchanged.

BTGF Bit ToGgle F

Syntax: BTGF f, b, a

Operands: 0 ≦ f ≦ 255 ; 0 ≦ b ≦ 7 ; a € (0, 1)

Operation: $\overline{(f < b >)} \rightarrow f < b >$

Status Affected: None

Description: Complement a specific bit of the register

Words: 1

Cycles: 1

Example 1: BTGF REG, 7, 0

Before Instruction:

REG(080H)=0111 1111B

After Instruction:

REG(080H)=1111 1111B

Remark: Complement BIT7 of register, REG.

BTSS Bit Test and Skip if Set

Syntax: BTSS f, b, a

Operands: 0 ≦ f ≦ 255 ; 0 ≦ b ≦ 7 ; a € (0)

Operation: skip if (f < b >)=1

Status Affected: None

Description: Compare whether a specific bit of register is 1. Skip the next instruction if the bit is 1. If not, execute the next instruction.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
BTSS    REG, 7, 0
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(080H)=0FFH

After Instruction:

WREG(02CH)=000H

REG(080H)=0FFH

Remark: BIT7of register, REG is 1, so skip the next instruction.

Example 2:

```
BTSS    REG, 7, 0
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(080H)=07FH

After Instruction:

WREG(02CH)=001H

REG(080H)=07FH

Remark: BIT7of register, REG is 0, so the program can execute the next instruction.

BTSZ Bit Test and Skip if Zero

Syntax: BTSZ f, b, a

Operands: 0 ≦ f ≦ 255 ; 0 ≦ b ≦ 7 ; a € (0, 1)

Operation: skip if (f < b >)=0

Status Affected: None

Description: Compare whether a specific bit of the register is 0. Skip the next instruction if the bit is 0. If not, execute the next instruction.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
BTSZ    REG, 7, 0
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(080H)=0FFH

After Instruction:

WREG(02CH)=001H

REG(080H)=0FFH

Remark: BIT7 of register, REG is not 0, so the program can execute the next instruction.

Example 2:

```
BTSZ    REG, 7, 0
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(080H)=07FH

After Instruction:

WREG(02CH)=000H

REG(080H)=07FH

Remark: BIT7 of register, REG is 0, so skip the next instruction.

CALL subroutine CALL

Syntax: CALL n

Operands: 0 ≦ n ≦ 16384(03FFFH); s € (0, 1);

Operation: (PC) + 1 → TOS, n → PC,
If s=1,
(WREG) → WREGSDW,
(STATUS) → STASDW
(BSRCN) → BSRC

Status Affected: STKPTR<STKFL>, STKPTR<STKOV>, Pstatus<SKERR>

Description: Call vice program, the maximum call range is 2Kbytes memory range.
If s=1, register WREG, STATUS and BSRCN value will be placed into corresponding shadow register.
If the stack is the last layer of the specific product after calling vice program, flag bit, STKFL will be configured as 1.

Under SBMSET1<7>=0 condition, STKOV flag will be configured as 1 when stack full and CALL instruction is executed, SKERR will be configured as 1. PC operates normally.

Under SBMSET1<7>=1 condition, STKOV flag will be configured as 1 when stack full and CALL instruction is executed, SKERR will be configured as 1. IC will be reset and PC will return to 000H.

When STKFL or STKOV occurs, either one is cleared; the other one will be cleared.

Words: 2

Cycles: 2

Example 1:

```
LABEL:  CALL    NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

PC= address (NEXT)
TOS= address (LABEL + 2)
WREGSDW= WREG
BSRSDW= BSRCN
STASDW=STATUS

Remark: When s=1, register WREG, STATUS and BSRCN value will be placed into corresponding shadow register.

CLRF CLeaR F

Syntax: CLRF f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: 000H → f

Status Affected: None

Description: Clear register f value to 0.

Words: 1

Cycles: 1

Example 1: CLRF REG,0

Before Instruction:

REG(080H)=055H

After Instruction:

REG(080H)=000H

Remark: Clear register, REG value as 0.

COMF COMplement F

Syntax: COMF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: $\overline{(f)}$ → destination

Status Affected: N, Z

Description: The value of register f is complemented and the result is stored in d appointed register.
If d = 0, the result is stored in accumulator W ;
If d = 1, the result is stored in register f ;
If a = 0, the result is stored in RAM address ;
If a = 1, the result is stored in the appointed RAM address of BSRCN register.

Words: 1

Cycles: 1

Example 1: COMF REG, 0, 0

Before Instruction:	After Instruction:
WREG(02CH)=055H	WREG(02CH)=000H
REG(080H)=0FFH	REG(080H)=0FFH
N=Z=0	Z=1, N=0

Example 2: COMF REG, 1, 1 (if BSRCN=001H)

Before Instruction:	After Instruction:
WREG(02CH)=055H	WREG(02CH)=055H
REG(070H)=055H	REG(070H)=0AAH
N=Z=0	N=1, Z=0

CPSE ComPare f with w, Skip if f Equal w

Syntax: CPSE f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: skip if (f) = (W)

Status Affected: None

Description: Compare register value and accumulator W value. If the value is equivalent, skip the next value. If the value is > or <, continue the next instruction.

Words: 1

Cycles: 1(2)

Example 1:

```
CPSE    REG, 0
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(080H)=000H

After Instruction:

WREG(02CH)=000H

REG(080H)=000H

Remark: The value of register f and accumulator W is the same, skip the next instruction.

Example 2:

```
CPSE    REG, 1    (if BSRCN=001H)
MVL     001H
NOP
```

Before Instruction:

WREG(02CH)=000H

REG(070H)=07FH

After Instruction:

WREG(02CH)=001H

REG(070H)=07FH

Remark: The value of register f and accumulator W is different, continue the next instruction.

CPSG ComPare f with w, Skip if f Greater than w

Syntax: CPSG f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: skip if (f) > (W)

Status Affected: None

Description: Compare the register value and accumulator W value.
If register value is greater than accumulator W value, skip the next instruction..
If register value is smaller than accumulator W value, continue the next instruction.

Words: 1

Cycles: 1(2)

Example 1:

```
CPSG    REG, 0
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(080H)=006H

After Instruction:

WREG(02CH)=005H

REG(080H)=006H

Remark: The value of register f is greater than accumulator W content, skip the next instruction.

Example 2:

```
CPSG    REG, 1    (if BSRCN=001H)
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(070H)=005H

After Instruction:

WREG(02CH)=00FH

REG(070H)=005H

Remark: The value of register f and accumulator W content is the same, continue the next instruction.

CPSL ComPare f with w, Skip if f Less than w

Syntax: CPSL f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: skip if (f) < (W)

Status Affected: None

Description: Compare the register value and accumulator W value.
If register value is smaller than accumulator W value, skip the next instruction.
If register value is greater or equivalent to accumulator value, continue the next instruction.

Words: 1

Cycles: 1(2)

Example 1:

```
CPSL    REG, 0
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(080H)=004H

After Instruction:

WREG(02CH)=005H
REG(080H)=004H

Remark: Register f value is smaller than accumulator W value, skip the next instruction.

Example 2:

```
CPSL    REG, 1    (if BSRCN=001H)
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(070H)=005H

After Instruction:

WREG(02CH)=00FH
REG(070H)=005H

Remark: Register f value is equivalent to accumulator value, continue the next instruction.

CWDT Clear WatchDog Timer

Syntax: CWDT

Operands: None

Operation: 000H → Watch dog counter

Status Affected: TO

Description: Clear zero the content of watch dog timer.

Words: 1

Cycles: 1

Example 1: CWDT

Before Instruction:

WDT counter = ???

After Instruction:

WDT counter = 000H

Pstatus<TO> = 0

DCF DeCrement F

Syntax: DCF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) – 1 → destination

Status Affected: C, DC, N, OV, Z

Description: Subtract 1 of register f value, and place the result back to d appointed register.
If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address ; .
If a = 1, the result is stored in the appointed RAM address of BSRCN register.

Words: 1

Cycles: 1

Example 1: DCF REG, 0, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=0FFH

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=0FEH

REG(080H)=0FFH

C=DC=N=1, OV=Z=0

Remark: C, DC has not been borrowed, so C=DC=1 ; the result >127, so N=1.

Example 2: DCF REG, 1, 1 (if BSRCN=001H)

Before Instruction:

WREG(02CH)=055H

REG(070H)=000H

C=DC=N=OV=Z=

After Instruction:

WREG(02CH)=055H

REG(070H)=0FFH

N=1, C=DC=OV=Z=0

Remark: C, DC has been borrowed, so C=DC=0 ; the result still >127, so N=1.

Example 3: DCF REG, 1, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=080H

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=055H

REG(080H)=07FH

C= OV=1, DC= N= Z=0

Remark: Only DC has been borrowed, so DC=0, C=1 ; BIT7 has changed from 1to 0 after execution, so OV=1.

Example 4: DCF REG, 0, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=001H

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=000H

REG(080H)=001H

C= DC= Z=1, N=OV=0

Remark: Remark: C, DC has not been borrowed, so C=DC=1；the result is 0, so Z=1.

DCSUZ DeCrement f, Skip if Un-Zero

Syntax: DCSUZ f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) – 1 → destination, skip if destination≠0

Status Affected: None

Description: Compare the decremented register value with 0. If register value is not equal to 0, skip the next instruction. If the value is 0, continue the next instruction and place the result to d appointed register.

If d = 0, the result is placed in accumulator W ;

If d = 1, the result is placed in register f ;

If a = 0, the result is placed in RAM address ;

If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
DCSUZ   REG, 1, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=00FH

REG(080H)=001H

After Instruction:

WREG(02CH)=00AH

REG(080H)=000H

Remark: If the result is 0, continue to execute the next program, and the result will be placed in register REG.

Example 2:

```
DCSUZ   REG, 0, 1  (if BSRCN=001H)
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=055H

REG(070H)=000H

After Instruction:

WREG(02CH)=0FFH

REG(070H)=000H

Remark: If the result is not 0, skips the next instruction and places the result to accumulator W.

DCSZ DeCrement f, Skip if Zero

Syntax: DCSZ f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) – 1 → destination, skip if destination=0

Status Affected: None

Description: Compare the decremented register value with 0. If the register value equals to 0, skip the next instruction. If not, continue the next instruction and place the result back to d appointed register.

If d = 0, place the result to accumulator W ;

If d = 1, place the result to register f ;

If a = 0, place the result to RAM address ;

If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
DCSZ    REG, 0, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=00FH

REG(080H)=001H

After Instruction:

WREG(02CH)=000H

REG(080H)=001H

Remark: The result is 0, so skip the next instruction.

Example 2:

```
DCSZ    REG, 1, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=055H

REG(070H)=000H

After Instruction:

WREG(02CH)=00AH

REG(070H)=0FFH

Remark: The result is not 0, so continue the next program and place the result to register REG.

IDLE IDLE mode

Syntax: IDLE

Operands: None

Operation: CPU Halt

Status Affected: Pstatus<IdleB>

Description: CPU accesses to idle mode, program blank instruction executes action.
It is recommended to add NOP instruction after IDLE instruction.

Words: 1

Cycles: 1

Example 1: IDLE
NOP

Before Instruction:

Pstatus<IdleB>=0

After Instruction:

Pstatus<IdleB>=1

IDLE
NOP……program break here

INF INcrement F

Syntax: INF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) ＋ 1 → destination

Status Affected: C, DC, N, OV, Z

Description: Add 1 to the content of register f and place the result to d appointed register.
If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address (000H~0FFH) ;
If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1

Example 1: INF REG, 0, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=0FFH

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=000H

REG(080H)=0FFH
C=DC= Z=1, N=OV=0

Remark: C, DC is carried, so C=1. After execution, the result equals to 0, so Z=1.

Example 2: INF REG, 1, 1 (if BSRCN=001H)

Before Instruction:

WREG(02CH)=055H

REG(070H)=00FH

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=055H

REG(070H)=010H
DC=1, C=N=OV=Z=0

Remark: DC is carried, so DC=1.

Example 3: INF REG, 1, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=07FH

C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=055H

REG(080H)=080H
DC= N=OV=1, C=Z=0

Remark: DC is carried, so DC=1. BIT7 changed from 0 to 1 after execution, so OV=1. If the result > 127, N=1.

INSUZ INcrement f, Skip if Un-Zero

Syntax: INSUZ f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0)

Operation: (f) ＋ 1 → destination, skip if destination≠0

Status Affected: None

Description: Compare the register incremented value with 0. If the register value is not 0, then skip the next instruction. If the value equals to 0, then continue executing next instruction and store the result to d appointed register.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
INSUZ   REG, 1, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0FFH

After Instruction:

WREG(02CH)=00AH

REG(080H)=000H

Remark: The result is 0, so continue the next program. The result will be placed back to register REG.

Example 2:

```
INSUZ   REG, 0, 1  (if BSRCN=001H)
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=055H

REG(070H)=000H

After Instruction:

WREG(02CH)=001H

REG(070H)=000H

Remark: The result is not 0, so skip the next instruction. The result will be stored in accumulator W.

INSZ INcrement f, Skip if Zero

Syntax: INSZ f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) ＋ 1 → destination, skip if destination=0

Status Affected: None

Description: Compare the incremented register value with 0. If the value is 0, skip the next instruction. If not equals to 0, continue the next instruction and store the result to d appointed register.
If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address ; .
If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1(2)(3)

Example 1:

```
INSZ    REG, 0, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0FFH

After Instruction:

WREG(02CH)=000H
REG(080H)=0FFH

Remark: The result is 0, skip the next instruction.

Example 2:

```
INSZ    REG, 1, 0
MVL     00AH
NOP
```

Before Instruction:

WREG(02CH)=055H

REG(070H)=000H

After Instruction:

WREG(02CH)=00AH
REG(070H)=001H

Remark: The result is not 0, so continue the next instruction and place the result to register REG.

IORF Inclusive OR w with F

Syntax: IORF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (W) OR (f) → destination

Status Affected: N, Z

Description: Inclusive OR accumulator W value and register F value, and place the result to d appointed register.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result is placed in the appointed address of register BSRCN.

Words: 1

Cycles: 1

Example 1: IORF REG, 0, 0

Before Instruction:

WREG(02CH)=055H

REG(080H)=0AAH

N=Z=0

After Instruction:

WREG(02CH)=0FFH

REG(080H)=0AAH

N=1, Z=0

Remark: The result >127, so N=1.

Example 2: IORF REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(070H)=0F0H

N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(070H)=0FFH

N=1, Z=0

Remark: The result >127, so N=1.

IORL Inclusive OR Literal with w

Syntax: IORL k

Operands: $0 \leqq k \leqq 255$

Operation: (W) OR k → W

Status Affected: Z

Description: Inclusive OR accumulator W value and k and then place the result back to accumulator W.

Words: 1

Cycles: 1

Example 1: IORL 055H

Before Instruction:

WREG(02CH)=0AAH

N=Z=0

After Instruction:

WREG(02CH)=0FFH

N=1, Z=0

Remark: The result>127, so N=1.

Example 2: IORL 000H

Before Instruction:

WREG(02CH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

N=1, Z=0

Remark: The result is 0, so Z=1.

JC Jump if Carry

Syntax: JC n

Operands: -128 ≦ n ≦ 127

Operation: If Status <carry bit> is 1, jump to n

Status Affected: None

Description: When the carry flag C=1 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JC    NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If C=0, PC= address (LABEL + 1)

If C=1, PC= address (NEXT)

JMP unconditional JuMP

Syntax: JMP n

Operands: $0 \leqq n \leqq 16384(03FFFH)$

Operation: n → PC

Status Affected: None

Description: Unconditionally jump to appointed address n.

Words: 2

Cycles: 2

Example 1:

```
LABEL:  JMP  NEXT
          .
          .
          .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

PC= address (NEXT)

JN Jump if Negative

Syntax: JN n

Operands: -128 ≦ n ≦ 127

Operation: If Status <negative bit> is 1, jump to n

Status Affected: None

Description: When the negative flag N=1 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JN   NEXT
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If N=0, PC= address (LABEL + 1)

If N=1, PC= address (NEXT)

JNC Jump if Not Carry

Syntax: JNC n

Operands: -128 ≦ n ≦ 127

Operation: If Status <carry bit> is 0, jump to n

Status Affected: None

Description: When carry flag C=0 of status register, jump to appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JNC  NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If C=0, PC= address (NEXT)

If C=1, PC= address (LABEL + 1)

JNN Jump if Not Negative

Syntax: JNN n

Operands: -128 ≦ n ≦ 127

Operation: If Status <negative bit> is 0, jump to n

Status Affected: None

Description: When the negative flag N=0 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JNN  NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If N=0, PC= address (NEXT)
If N=1, PC= address (LABEL + 1)

JNO Jump if Not Overflow

Syntax: JNO n

Operands: -128 ≦ n ≦ 127

Operation: If Status <overflow bit> is 0, jump to n

Status Affected: None

Description: When overflow flag OV=0 of status register, jump to appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JNO NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If OV=0, PC= address (NEXT)

If OV=1, PC= address (LABEL + 1)

JNZ Jump if Not Zero

Syntax: JNZ n

Operands: -128 ≦ n ≦ 127

Operation: If Status <zero bit> is 0, jump to n

Status Affected: None

Description: When the zero flag Z=0 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JNZ  NEXT
          .
          .
          .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If Z=0, PC= address (NEXT)

If Z=1, PC= address (LABEL + 1)

JO Jump if Overflow

Syntax: JO n

Operands: -128 ≦ n ≦ 127

Operation: If Status <overflow bit> is 1, jump to n

Status Affected: None

Description: When overflow flag OV=1 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JO    NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If OV=0, PC= address (LABEL + 1)

If OV=1, PC= address (NEXT)

JZ Jump if Zero

Syntax: JZ n

Operands: -128 ≦ n ≦ 127

Operation: If Status <zero bit> is 1, jump to n

Status Affected: None

Description: When zero flag Z=1 of status register, jump to the appointed address n.

Words: 1

Cycles: 1(2)

Example 1:

```
LABEL:  JZ   NEXT
          .
          .
          .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

If Z=0, PC= address (LABEL + 1)

If Z=1, PC= address (NEXT)

LBSR Load literal into Bank Select Register

Syntax: LBSR k

Operands: $0 \leqq k \leqq 7$

Operation: k → BSRCN

Status Affected: None

Description: Move constant k to Bank Select Register (BSRCN) to configure data origin address.

Words: 1

Cycles: 1

Example 1:

```
LBSR    001H        …… total instruction cycles = 1
```

Before Instruction: BSRCN=000H

After Instruction: BSRCN=001H

Remark: Configure BSRCN=001H.

Example 2:

```
MVL     001H
MVF     BSRCN, 1, 0     …… total instruction cycles = 2
```

Before Instruction: BSRCN=000H

After Instruction: BSRCN=001H

Remark: This sample program action is the same with Example 1.

LDPR LoaD Point into fsR

Syntax: LDPR k, f

Operands: $0 \leqq k \leqq 1279(04FFH)$; $0 \leqq f \leqq 1$

Operation: k → FSR (FSRxH, FSRxL)

Status Affected: None

Description: Data addressing method not only include direct addressing and instant addressing, but also indirect addressing. The purpose of this instruction is to simplify indirect addressing configuration way.

Indirect addressing utilizes register, FSR (File Select Register). Register FSR stores data address and data value is place in register INDF. There are two FSR registers of indirect addressing that currently can be used, namely FSR0 and FSR1 register. The definition memory address length can reach 11 bits. It can be further divided into 2 high bit registers, FSR0 is separated into FSR0H and FSR0L, and FSR1 can be portioned out FSR1H and FSR1. L. Memory data is saved in between INDF0 and INDF1 correspondingly.

Words: 2

Cycles: 2

Example 1:

```
LDPR    017FH, 0
MVL     0AAH
MVF     INDF0, 1, 0          …… total instruction cycles = 4
```

Before Instruction:	After Instruction:
FSR0H=000H	FSR0H=001H
FSR0L=080H	FSR0L=07FH
INDF0=0FFH	INDF0=0AAH
Address (017FH)=055H	Address (017FH)=0AAH

Remark: f=0 is default value, FSR0. If f argument is 0 of the program, it can be saved.

Sample program describes the way to use indirect addressing to write data value, 0AAH of data address (017FH).

Example 2:

```
MVL     07FH
MVF     FSR0L, 1, 0
MVL     001H
MVF     FSR0H, 1, 0
MVL     0AAH
MVF     INDF0, 1, 0          …… total instruction cycles = 6
```

Remark: This program action is the same as described in Example 1.

MULF MULtiply w with F

Syntax: MULF f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: (W) × (f) → PRODH (high byte), PRODL (low byte)

Status Affected: None

Description: Multiply accumulator W value and register f value and then place the result to PRODH, PRODL registers.

a = 0 or a =1 configuration is determined by RAM address of register f.

If a = 0 means register f exists in 080H to 0FFH appointed RAM address (BSRCN=000H).

If a = 1means register f exists in 100H to 17FH appointed RAM address (BSRCN=001H).

Words: 1

Cycles: 2

Example 1: MULF REG, 1

Before Instruction:

WREG(02CH)=00FH

REG(017FH)=0FFH

PRODH=??

PRODL=??

After Instruction:

WREG(02CH)=00FH

REG(017FH)=0FFH

PRODH=00EH

PRODL=0F1H

Remark: Use direct address to execute multiplication.

Example 2: MULF INDF0, 0

Before Instruction:

WREG(02CH)=00FH

FSR0H=001H, FSR0L=07FH

Address (017FH)=0FFH

PRODH=??

PRODL=??

After Instruction:

WREG(02CH)=00FH

FSR0H=001H, FSR0L=07FH

Address (017FH)=0FFH

PRODH=00EH

Remark: Use indirect address to execute multiplication.

MULL MULtiply Literal with w

Syntax: MULL k

Operands: $0 \leqq k \leqq 255$

Operation: (W) × k → PRODH (high byte), PRODL (low byte)

Status Affected: None

Description: Multiply constant k and accumulator W value and place the result to register, PRODH and PRODL.

Words: 1

Cycles: 2

Example 1: MULL 0FFH

Before Instruction:	After Instruction:
WREG(02CH)=00FH	WREG(02CH)=00FH
PRODH=??	PRODH=00EH
PRODL=??	PRODL=0F1H

MVF MoVe F to w or MoVe w to F

Syntax: MVF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) → W, or (W) → f

Status Affected: None

Description: Move register f value to accumulator W; or move accumulator W value to register f.
If d = 0, it means moving register f value to accumulator W ;
If d = 1m, it means moving accumulator W value to register f ;
a = 0 or a =1 configuration must be determined by register f address of RAM :
If a = 0, it means register f exists in the appointed RAM address of 000H~0FFH (BSRCN=000H)
If a = 1, it means register f exists in the appointed RAM address of 100H~17FH (BSRCN=001H).

Words: 1

Cycles: 1

Example 1: MVF REG, 0, 0

Before Instruction:

WREG(02CH)=055H
REG(080H)=0AAH
REG1(170H)=0FFH

After Instruction:

WREG(02CH)=0AAH
REG(080H)=0AAH
REG1(170H)=0FFH

Remark: d=0, it means moving register REG value to accumulator W.

Example 2: MVF REG1, 1, 0

Before Instruction:

WREG(02CH)=055H
REG1(170H)=0FFH
REG(080H)=0AAH

After Instruction:

WREG(02CH)=055H
REG1(170H)=055H
REG(080H)=0AAH

Remark: d=1, it means moving accumulator W value to register f.

MVFF MoVe F to F

Syntax: MVFF fs, fd

Operands: 0 ≦ fs ≦ 1279(04FFH) ; 0 ≦ fd ≦ 1279(04FFH)

Operation: (fs) → fd

Status Affected: None

Description: Move register fs value to register fd.

Words: 2

Cycles: 2

Example 1: MVFF REG, REG1

Before Instruction:	After Instruction:
REG=055H	REG=055H
REG1=0AAH	REG1=055H

MVL MoVe Literal to w

Syntax: MVL k

Operands: $0 \leqq k \leqq 255$

Operation: $k \rightarrow W$

Status Affected: None

Description: Move constant k to accumulator W.

Words: 1

Cycles: 1

Example 1: MVL 0FFH

Before Instruction:

WREG(02CH)=000H

After Instruction:

WREG(02CH)=0FFH

MVLP MoVe Literal to Pointer

Syntax: MVLP k

Operands: 0 ≦ k ≦ 16384(03FFFh)

Operation: k → TBLPTR (TBLPTRH, TBLPTRL)

Status Affected: None

Description: MVLP is the instruction to configure program memory pointer, mainly used in look-up-table instruction and TBLR collocation.

Words: 2

Cycles: 2

Example 1: MVLP 001FF0H

Before Instruction:	After Instruction:
TBLPTRH=000H	TBLPTRH=01FH
TBLPTRL=000H	TBLPTRL=0F0H

Remark: Configure program memory pointer, load constant k to reisterTBLPTR.

NOP No OPeration

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Description: No operation is executed, only delay for 1 instruction time.

Words: 1

Cycles: 1

Example 1: NOP

Remark: Blank instruction, only executing the delay time of 1 instruction cycle.

POP POP return stack

Syntax: POP

Operands: None

Operation: (TOS) → Bit bucket, then ((TOS) at STKPTR-1) → TOS

Status Affected: None

Description: Discard the stack pointer pointed stack value of and subtract 1 of stack pointer register and place the value to register, TOS.

Register TOS can be divided into TOSH and TOSL register.

Words: 1

Cycles: 1

Example 1:

```
LABEL:  POP
RJ      LABEL1

LABEL1: NOP
```

Before Instruction:

STKPTR=003H

TOS= 001666H

TOS (STKPTR=002H) = 001234H

TOS (STKPTR=001H) = 000567H

PC=LABEL

After Instruction:

STKPTR=002H

TOS= 001234H

PC=LABEL

Remark: If STKPTR=00H, no influence will be aroused by executing POP instruction. TOS is 0, STKPTR is 0.

RCALL Relative subroutine CALL

Syntax: RCALL n

Operands: -1024 ≦ n ≦ 1023

Operation: (PC) + 1 → TOS, n → PC,

Status Affected: STKPTR<STKFL>, STKPTR<STKOV>, Pstatus<SKERR>.

Description: Call vice program, maximum call range is ±1K bytes of memory range.
If the layer is the top-of-stack after calling vice program, STKFL will be configured as 1.
Under the configuration of SBMSET1<7>=0, if RCALL instruction is executed after stack overflow, STKOV flag will be configured as 1.
SKERR will be configured as 1 as well. PC operates normally.
Under the configuration of SBMSET1<7>=1, if RCALL instruction is executed after stack overflow, STKOV flag will be configured as 1.
SKERR will be configured as 1. IC will be reset and then PC will return back to 000H.
When STKFL or STKOV occurs, either one flag is erased; the other will be erased too.

Words: 1

Cycles: 2

Example 1:

```
LABEL:  RCALL   NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

TOS=??

After Instruction:

PC= address (NEXT)

TOS= address (LABEL + 2)

RET RETurn from subroutine

Syntax: RET

Operands: s € (0, 1)

Operation: (TOS) → PC,
If s=1,
(WREGSDW) → WREG,
(STASDW) → STATUS,
(BSRSDW) → BSRCN

Status Affected: STKPTR<STKUN>, Pstatus<SKERR>

Description: Leave vice program and store stack pointer register value to PC.
If s=1, shadow register value will be placed into corresponding register (WREG, STATUS, BSRCN).
When vice program is not called and STKPTR=000H, executes instruction RET may result in IC reset and STKUN flag may be configured as 1.
SKERR flag will be configured as 1.

Words: 1

Cycles: 2

Example 1: RET

Before Instruction:	After Instruction:
None	PC=TOS

RETI RETurn from Interrupt

Syntax: RETI

Operands: None

Operation: (TOS) → PC, 1 → GIE
If s=1,
(WREGSDW) → WREG,
(STASDW) → STATUS,
(BSRSDW) → BSRCN

Status Affected: GIE, STKPTR<STKUN>, Pstatus<SKERR>

Description: Return form interrupt and store the stack pointer register value to PC. Interrupt enable pin is configured again as 1.
If s=1, shadow register value will be placed into corresponding register (WREG, STATUS, BSRCN)
When vice program is not called and STKPTR=000H, execute RETI instruction will lead to IC reset. STKUN flag will be configured as 1 so as SKERR flag.

Words: 1

Cycles: 2

Example 1: RETI 1

Before Instruction:
None

After Instruction:
PC=TOS
WREG = WREGSDW
BSRCN = BSRSDW
STATUS = STASDW
GIE=1

1

RETL RETurn Literal to w

Syntax: RETL k

Operands: $0 \leqq k \leqq 255$

Operation: k → W, (TOS) → PC

Status Affected: STKPTR<STKUN>, Pstatus<SKERR>

Description: Return from vice program to main program. While returning the instruction, constant k value will be loaded to accumulator W.

This instruction is often implemented in look-up-table function.

When vice program is not called and STKPTR=000H, executing RETL instruction may reset the IC. STKUN flag will be configured as 1, so as SKERR flag.

Words: 1

Cycles: 2

Example 1:

```
LABEL:  MVL     001H
        CALL    TABLE
        .
        .
        .
TABLE:  ADDF    PCLATL, 1, 0
        RETL    055H
        RETL    0AAH
```

Before Instruction:

WREG(02CH)=001H

After Instruction:

WREG(02CH)=0AAH

Remark: While returning to main program, constant k will be loaded to accumulator W.
This example presents how to write Offset value of PCLATL to determine the value of TABLE.

RJ unconditional Relative Jump

Syntax: RJ n

Operands: -1024 ≦ n ≦ 1023

Operation: n → PC

Status Affected: None

Description: Unconditionally jump to appointed address n.

Words: 1

Cycles: 2

Example 1:

```
LABEL:  RJ   NEXT
        .
        .
        .
NEXT:   NOP
```

Before Instruction:

PC = address (LABEL)

After Instruction:

PC= address (NEXT)

RLF Rotate Left F (no carry)

Syntax: RLF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0)

Operation: (f<n>) → destination <n+1 >,

(f<7>) → destination < 0 >

Status Affected: Z

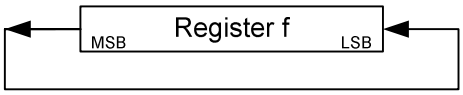
Description: Rotate left f value.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result will be placed to appointed RAM address of register BSRCN.



Words: 1

Cycles: 1

Example 1: RLF REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(080H)=055H

N=Z=0

Example 2: RLF REG, 0, 0

Before Instruction:

WREG(02CH)=00FH

REG(07FH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

REG(07FH)=000H

Z=1, N=0

Example 3: RLF REG, 0, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=055H

N=Z=0

After Instruction:

WREG(02CH)=0AAH

REG(080H)=055H

N=1, Z=0

RLFC Rotate Left F through Carry

Syntax: RLFC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<n>) → destination <n+1 >,

(f<7>) → Status< C >,

Status< C > → destination < 0 >

Status Affected: C, N, Z

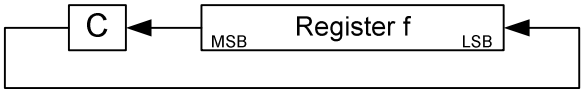
Description: Rotate left register f value and carry flag C.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result will be placed to appointed RAM address of register BSRCN.



Words: 1

Cycles: 1

Example 1: RLFC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(080H)=054H

C=1, N=Z=0

Example 2: RLFC REG, 0, 0

Before Instruction:

WREG(02CH)=0FH

REG(070H)=0EAH

C=N=Z=0

After Instruction:

WREG(02CH)=0D4H

REG(070H)=0EAH

C=N=1, Z=0

Example 3: RLFC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(070H)=080H

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(070H)=000H

C=Z=1, N=0

RRF Rotate Righ F (no carry)

Syntax: RRF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<n>) → destination <n - 1 >,

(f<0>) → destination < 7 >

Status Affected: Z

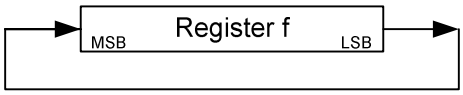
Description: Rotate right the register f value.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result will be placed to appointed RAM address of register BSRCN.



Words: 1

Cycles: 1

Example 1: RRF REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(080H)=055H

N=Z=0

Example 2: RRF REG, 0, 0

Before Instruction:

WREG(02CH)=00FH

REG(07FH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

REG(07FH)=000H

Z=1, N=0

Example 3: RRF REG, 0, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=055H

N=Z=0

After Instruction:

WREG(02CH)=0AAH

REG(080H)=055H

N=1, Z=0

RRFC Rotate Right F through Carry

Syntax: RRFC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<n>) → destination <n-1 >,

(f<0>) → Status< C >,

Status< C > → destination < 7 >

Status Affected: C, N, Z

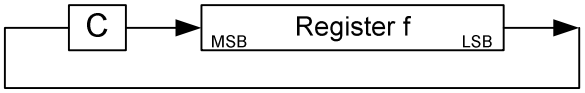
Description: Rotate right the register f value and carry flag C.

If d = 0, the result is placed to accumulator W ;

If d = 1, the result is placed to register f ;

If a = 0, the result is placed to RAM address ;

If a = 1, the result will be placed to appointed RAM address of register BSRCN.



Words: 1

Cycles: 1

Example 1: RRFC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(080H)=055H

C=N=Z=0

Example 2: RRFC REG, 0, 0

Before Instruction:

WREG(02CH)=00FH

REG(07FH)=055H

C=1, N=Z=0

After Instruction:

WREG(02CH)=0AAH

REG(07FH)=055H

C=N=1, Z=0

Example 3: RRFC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH

REG(07FH)=001H

C=N=Z=0

After Instruction:

WREG(02CH)=00FH

REG(07FH)=000H

C=Z=1, N=0

SETF SET F

Syntax: SETF f, a

Operands: 0 ≦ f ≦ 255 ; a € (0, 1)

Operation: 0FFH → f

Status Affected: None

Description: Configure all the contents of register f to 1.

a = 0 or a =1 is determined by RAM address of register f :

If a = 0, it means that register f exists in the appointed RAM address of 000H to 0FFH (BSRCN=000H).

If a = 1, it means that register f exists in the appointed RAM address of 100H to 17FH (BSRCN=001H).

Words: 1

Cycles: 1

Example 1: SETF REG, 0

Before Instruction:

WREG(02CH)=00FH

REG(080H)=0AAH

After Instruction:

WREG(02CH)=00FH

REG(080H)=0FFH

SLP enter SLeeP mode

Syntax: SLP

Operands: None

Operation: 1 → PD

Status Affected: PD

Description: CPU accesses into sleep mode, oscillator stop operating.

Words: 1

Cycles: 1

Example 1:

```
SLP
NOP
```

Before Instruction:

PD=0

After Instruction:

PD=1

SUBC SUBtract w from f with Carry

Syntax: SUBC f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) – (W) – $\overline{(C)}$ → destination

Status Affected: C, DC, N, OV, Z

Description: Subtract accumulator W and carry flag C's reversed value of register f and place the result to d.

If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address ;
If a = 1, the result will be placed to appointed RAM address of register BSRCN.

Words: 1

Cycles: 1

Example 1: SUBC REG, 0, 0

Before Instruction:

WREG=001H

REG(080H)=001H

C=1, DC=N=OV=Z=0

After Instruction:

WREG=000H

REG(080H)=001H

C= DC=Z=1, N=OV= 0

Remark: C has not been borrowed, so C=DC=1. The result is 0, so Z=1.

Example 2: SUBF REG, 1, 0

Before Instruction:

WREG=000H

REG(07FH)=080H

C=DC=N=OV=Z=0

After Instruction:

WREG=000H

REG(07FH)=07FH

C=OV=1, DC= N=Z=0

Remark: C has not been borrowed, so C=1; DC has been borrowed, so DC=0;

OV=1 matches the judgment criterion: (Negative) – (Positive) = Positive or (Positive) – (Negative) = Negative ;
This example matches: (Negative) – (Positive) = Positive, so OV=1.

SUBF SUBtract w from F

Syntax: SUBF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f) — (W) → destination

Status Affected: C, DC, N, OV, Z

Description: Subtract accumulator W value of register f and place the result to d.
If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address ;
If a = 1, the result will be placed to appointed RAM address of register BSRCN.

Words: 1

Cycles: 1

Example 1: SUBF REG, 0, 0

Before Instruction:

WREG=001H

REG(080H)=001H

C=DC=N=OV=Z=0

After Instruction:

WREG=000H

REG(080H)=001H

C= DC=Z=1, N=OV= 0

Remark: C has not been borrowed, so C=DC=1. The result is 0, so Z=1.

Example 2: SUBF REG, 1, 0

Before Instruction:

WREG=001H

REG(07FH)=080H

C=DC=N=OV=Z=0

After Instruction:

WREG=001H

REG(07FH)=07FH

C=OV=1, DC= N=Z=0

Remark: C has not been borrowed, so C=1; DC has been borrowed, so DC=0;
OV=1 matches the judgment criterion: (Negative) — (Positive) = Positive or (Positive) — (Negative) = Negative ;
This example matches: (Negative) — (Positive) = Positive, so OV=1.

SUBL SUBtract w from Literal

Syntax: SUBL k

Operands: $0 \leqq k \leqq 255$

Operation: K − (W) → W

Status Affected: C, Z

Description: Subtract constant k and accumulator W value and place the result back to accumulator W.

Words: 1

Cycles: 1

Example 1: SUBL 001H

Before Instruction:

WREG=001H

C=DC=N=OV=Z=0

After Instruction:

WREG=000H

C= DC=Z=1, N=OV= 0

Remark: C, DC has not been borrowed, so C=DC=1. The result is 0, so Z=1.

Example 2: SUBL 080H

Before Instruction:

WREG=001H

C=DC=N=OV=Z=0

After Instruction:

WREG=07FH

C=OV=1, DC= N=Z=0

Remark: C has not been borrowed, so C=1; DC has been borrowed, so DC=0;

OV=1 matches the judgment criterion: (Negative) − (Positive) = Positive or (Positive) − (Negative) = Negative；

This example matches: (Negative) − (Positive) = Positive, so OV=1.

Example 3: SUBL 07FH

Before Instruction:

WREG=0FFH

C=DC=N=OV=Z=0

After Instruction:

WREG=080H

DC=N=OV=1, C= Z=0

Remark: DC has not been borrowed, so DC=1；C has been borrowed, so C=0; the result > 127, so N=1.

This example matches (Positive) − (Negative) = Negative, so OV=1.

Example 4: SUBL 000H

Before Instruction:

WREG=001H

C=DC=N=OV=Z=0

After Instruction:

WREG=0FFH

N= 1, C=DC=OV=Z=0

Remark: C, DC has been borrowed, so C=DC=0；the result >127, so N=1.

SWPF SWaP F

Syntax: SWPF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0, 1)

Operation: (f<3:0>) → destination<7:4>

(f<7:4>) → destination<3:0>

Status Affected: None

Description: Switch high and low 4 bit value of register f.
If d = 0, the result is placed to accumulator W ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address (000H~0FFH) ;
If a = 1, the result will be placed to appointed RAM address of register BSRCN.

Words: 1

Cycles: 1

Example 1: SWPF REG, 1, 0

Before Instruction:

WREG=001H

REG(080H)=05AH

After Instruction:

WREG=001H

REG(080H)=0A5H

TBLR TaBLe Read

Syntax: TBLR (* , *+)

Operands: *, or *+

Operation: # If (TBLR *)

(Program Memory (TBLPTRH, TBLPTRL)) → TBLDH, TBLDL,

TBLPTR (TBLPTRH, TBLPTRL) do not change.

If (TBLR *+)

(Program Memory (TBLPTRH, TBLPTRL)) → TBLDH, TBLDL,

(TBLPTR) +1 ->TBLPTR.

Status Affected: None

Description: TBLR is an instruction that reads program memory contents; it is mainly implemented in look-up-table instruction. It is utilized in two ways:

- TBLR *

  Register TBLPTRH and TBLPTRL value is address pointer, read corresponding program memory contents to register TBLD (TBLDH, TBLDL).

- TBLR *+

  Register TBLPTRH and TBLPTRL value is address pointer, read corresponding program memory contents to register TBLD (TBLDH, TBLDL) and then add 1 to address pointer.

Words: 1

Cycles: 2

Example 1: TBLR

Before Instruction:

TBLDH, TBLDL= 0123H

At TBLPTR=0017FFH

Address (0017FFH) = data (5678H)

After Instruction:

TBLDH, TBLDL= 5678H

TBLPTR=0017FFH

Remark: Take 2 bytes data of TBLPTR address and place it to TBLD (TBLDH, TBLDL), the contents of BLPTR pointer remain unchanged.

Example 2: TBLR *

Before Instruction:

TBLDH, TBLDL= 0123H

At TBLPTR=0017FFH

Address(0017FFH) = data (5678H)

After Instruction:

TBLDH, TBLDL= 5678H

TBLPTR=001800H

Remark: Take 2 bytes data of TBLPTR address and place it to TBLD (TBLDH, TBLDL), add 1 to TBLPTRpointer.

TFSZ Test F, Skip if Zero

Syntax: TFSZ f, a

Operands: 0 ≦ f ≦ 255 ; a € (0)

Operation: skip if f = 0

Status Affected: None

Description: If register f value is 0, skip the next instruction. If the value is unequal to 0, the next instruction is executed.

a = 0 or a =1 is determined by RAM address of register f :

If a = 0, it means that register f exists in appointed RAM address of 000H to 0FFH (BSRCN=000H).

If a = 1, it means that register f exists in appointed RAM address of 100H to 17FH (BSRCN=001H).

Words: 1

Cycles: 1(2)(3)

Example 1:

```
TFSZ    REG, 0
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(080H)=000H

After Instruction:

WREG(02CH)=005H

REG(080H)=000

Remark: Register f value is 0, skip the next instruction.

Example 2:

```
TFSZ    REG, 1    (if BSRCN=001H)
MVL     00FH
NOP
```

Before Instruction:

WREG(02CH)=005H

REG(070H)=001H

After Instruction:

WREG(02CH)=00FH

REG(070H)=001H

Remark: Register f value is not 0, continue the next instruction.

XORF eXclusive OR w with F

Syntax: XORF f, d, a

Operands: 0 ≦ f ≦ 255 ; d € (0, 1) ; a € (0)

Operation: (W) XOR (f) → destination

Status Affected: Z

Description: Exclusive OR the constant k and accumulator w and place the result back to d.
If d = 0, the result is placed to accumulator w ;
If d = 1, the result is placed to register f ;
If a = 0, the result is placed to RAM address ;
If a = 1, the result will be placed to appointed RAM address of register BSRCN. .

Words: 1

Cycles: 1

Example 1: XORF REG, 0, 0

Before Instruction:

WREG(02CH)=0AAH

REG(080H)=055H

N=Z=0

After Instruction:

WREG(02CH)=0FFH

REG(080H)=055H
N=1, Z=0

Remark: The result >127, so N=1.

XOR: if both values equal, the result is 0 ; if both values are unequal, the result is 1.

Example 2: XORF REG, 1, 0

Before Instruction:

WREG(02CH)=0FFH

REG(070H)=0FFH

N=Z=0

After Instruction:

WREG(02CH)=0FFH

REG(070H)=000H
Z=1, N=0

Remark: The result is 0, so Z=1.

XOR: if both values equal, the result is 0 ; if both values are unequal, the result is 1.

Example 3: XORF REG, 0, 0

Before Instruction:

WREG(02CH)=000H

REG(080H)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

REG(080H)=000H
Z=1, N=0

Remark: The result is 0, so Z=1.

XOR: if both values equal, the result is 0 ; if both values are unequal, the result is 1.

XORL eXclusive OR Literal with w

Syntax: XORL k

Operands: 0 ≦ f ≦ 255

Operation: (W) XOR k → W

Status Affected: Z

Description: Exclusive OR the constant k and accumulator w and place the result back to accumulator W.

Words: 1

Cycles: 1

Example 1: XORL 055H

Before Instruction:

WREG(02CH)=0AAH

N=Z=0

After Instruction:

WREG(02CH)=0FFH

N=1, Z=0

Remark: The result >127, so N=1.

XOR: if both values equal, the result is 0；if both values are unequal, the result is 1.

Example 2: XORL 0FFH

Before Instruction:

WREG(02CH)=0FFH

N=Z=0

After Instruction:

WREG(02CH)=000H

Z=1, N=0

Remark: The result is 0, so Z=1.

XOR: if both values equal, the result is 0；if both values are unequal, the result is 1.

Example 3: XORL 000H

Before Instruction:

WREG(02CH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

Z=1, N=0

Remark: The result is 0, so Z=1.

XOR: if both values equal, the result is 0；if both values are unequal, the result is 1.

REVISION HISTORY

Major differences are stated thereinafter:

Version	Page	Revision Summary
V01	ALL	First Edition
V02	ALL	Layout revision
V03	-	Delete DAW instruction

.