



Instruction Set H08A User's Manual

H08A 指令快速索引

Instruction	Description	Cycles	Status Affected	Ref Page
BYTE-ORIENTED FILE REGISTER OPERATIONS				
ADDC	f,d,a 將 W 與 F 和 C 做相加，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	5
ADDF	f,d,a 將 W 與 F 做相加，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	6
ADDL	k 將常數 k 與 W 做相加，並將結果放至 W。	1	C,DC,N,OV,Z	7
ANDF	f,d,a 將 W 與 F 做 AND 運算，並將結果放至 W 或 F。	1	N,Z	8
ANDL	k 將常數 k 與 W 做 AND 運算，並將結果放至 W。	1	N,Z	9
ARLC	f,d,a 將 F 內的值與 C 一起做左移動作，並將結果放至 W 或 F。	1	C,N,OV,Z	10
ARRC	f,d,a 將 F 內的值做右移動作,MSB 保留不變,LSB 移至 C	1	C,N,Z	11
CLRF	f,a 將 F 內的值都清為 0。	1	None	18
COMF	f,d,a 將 F 內的值取補數，並將結果放至 W 或 F。	1	N,Z	19
CPSE	f,a 若 F 與 W 的值相等，則跳過下一個指令。	1(2)(3)	None	20
CPSG	f,a 若 F 大於 W，則跳過下一個指令。	1(2)(3)	None	21
CPSL	f,a 若 F 小於 W，則跳過下一個指令。	1(2)(3)	None	22
DCF	f,d,a 將 F 內的值減 1，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	24
DCSUZ	f,d,a 將 F 內的值減 1，若不為 0 則跳過下一個指令，並將結果放至 W 或 F。	1(2)(3)	None	25
DCSZ	f,d,a 將 F 內的值減 1，若為 0 則跳過下一個指令，並將結果放至 W 或 F。	1(2)(3)	None	26
INF	f,d,a 將 F 內的值加 1，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	28
INSUZ	f,d,a 將 F 內的值加 1，若不為 0 則跳過下一個指令，並將結果放至 W 或 F。	1(2)(3)	None	29
INSZ	f,d,a 將 F 內的值加 1，若為 0 則跳過下一個指令，並將結果放至 W 或 F。	1(2)(3)	None	30
IORF	f,d,a 將 W 與 F 做 OR 運算，並將結果放至 W 或 F。	1	N,Z	31
IORL	k 將常數 k 與 W 做 OR 運算，並將結果放至 W。	1	N,Z	32
LBSR	k 將常數 k 搬到 BSRCN 暫存器去。	1	None	42
LDPR	k,f 將常數 k (9-bit) 搬到第 f 個 FSR 暫存器去 (f = 0 ~ 1)。	2	None	43
MULF	f,a 將 W 與 F 做相乘。	2	None	44
MULL	k 將常數 k 與 W 做乘法運算。	2	None	45
MVF	f,d,a 將 W 內的值搬到 F 中(d=1)或 F 內的值搬到 W(d=0)。	1	None	46
MVFF	fs,fd 將 Fs 內的資料搬到 Fd 中。	2	None	47
MVL	k 將常數 k 搬到 W 去。	1	None	48
RETL	k 將堆疊最上層的值取出來放至 PC 中，並將 W 的值設為 k，而主程式由目前 PC 值開始執行	2	None	55
RLF	f,d,a 將 F 內的值做左移動作，並將結果放至 W 或 F。	1	N,Z	57
RLFC	f,d,a 將 F 內的值與 C 一起做左移動作，並將結果放至 W 或 F。	1	C,N,Z	58
RRF	f,d,a 將 F 內的值做右移動作，並將結果放至 W 或 F。	1	N,Z	59
RRFC	f,d,a 將 F 內的值與 C 一起做右移動作，並將結果放至 W 或 F。	1	C,N,Z	60
SETF	f,a 將 F 內的值設為 0xFF。	1	None	61
SUBC	f,d,a 將 F 內的值減掉 W 及反向 C，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	63
SUBF	f,d,a 將 F 內的值減掉 W，並將結果放至 W 或 F。	1	C,DC,N,OV,Z	64
SUBL	k 將常數 k 與 W 做減法，並將結果放至 W。	1	C,DC,N,OV,Z	65
SWPF	f,d,a 將 F 內的值高 4 位元與低 4 位元對調，並將結果放至 W 或 F。	1	None	66
TFSZ	f,a 測試 F 內的值是否等於 0，若為 0 則跳過下一個指令。	1(2)(3)	None	68
XORF	f,d,a 將 W 與 F 做 XOR 運算，並將結果放至 W 或 F。	1	N,Z	69
XORL	k 將常數 k 與 W 做 XOR 運算，並將結果放至 W。	1	N,Z	70

Remark	f	暫存器	b	暫存器第 b 個位元
	n	記憶體位置	k	8 位元常數
	d	資料存放地方; d = 0 表示存放在 W 累加器; d = 1 表示存放在 f 暫存器。		
	a	資料存放在哪個記憶體位置,a=0 存放在目前記憶體位置; a=1 存放在 BSRCN 暫存器內所指定記憶體位置		

ADDC ADD w and Carry bit to f

Syntax: ADDC f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(W) + (f) + (\text{Status}\langle C \rangle) \rightarrow \text{destination}$

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

Description: W 累加器中的值與 f 暫存器中的值與進位旗標 C 三者相加，並將運算結果放回 d 所指定的暫存器中；
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

Words: 1

Cycles: 1

Example 1: ADDC 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，則執行結果放回 W 累加器中。

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

Before Instruction:

W=001H
 REG(170H)=00EH
 C=1, DC=N=OV=Z=0

After Instruction:

W=001H
 REG(170H)=010H
 DC=1, C= N=OV=Z=0

Remark: d=1，則執行結果放回 f 暫存器中。
 a=1，則執行結果放回 BSRCN 暫存器所指定的 RAM 位址中。

ADDF ADD w to F

Syntax: ADDF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(W) + (f) \rightarrow \text{destination}$

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

Description: W 累加器中的值與 f 暫存器中的值相加，並將運算結果放回 d 所指定的暫存器中。
 若 $d = 0$ ，則運算後的結果放到 W 累加器中；
 若 $d = 1$ ，則運算後的結果放到 f 暫存器中；
 若 $a = 0$ ，則運算後的結果放到目前 RAM 的位址中；
 若 $a = 1$ ，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: $d=0$ ，則執行結果放回 W 累加器中。
 $a=0$ 為預設值，若程式中 $a=0$ 時，則程式中可以不加入此參數。

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: $d=1$ ，則執行結果放回 f 暫存器中。
 $a=0$ 為預設值，若程式中 $a=0$ 時，則程式中可以不加入此參數。

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: $d=1$ ，則執行結果放回 f 暫存器中。
 $a=1$ ，則執行結果放回 BSRCN 暫存器所指定的 RAM 位址中。
 雖然 REG RAM 位址為 070H，但因為 BSRCN=001H，所以會和 170H 暫存器中的數值作運算後，
 將執行果放回位址 170H 處。

ADDL ADD Literal to w

Syntax: ADDL k

Operands: $0 \leq k \leq 255$

Operation: $(W) + K \rightarrow W$

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

Description: W 累加器中的值與 k 值相加，並將運算結果放回 W 累加器中。

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: 低四位元相加後，有進位情形，半進位旗標 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: 低四位元相加後，有進位情形，半進位旗標 DC=1。
執行結果大於 127，則視為負數，負號旗標 N=1。
執行後，位元 7=1，溢位旗標 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: 低四位元相加後，有進位情形，半進位旗標 DC=1。
執行結果大於 127，則視為負數，負號旗標 N=1。
執行前位元 7=1，執行後位元 7 仍為 1 的狀態，則溢位旗標不變 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: 執行結果大於 0FFH，進位旗標 C=1。
低四位元相加後，有進位情形，半進位旗標 DC=1。
執行結果為 000H，則零位旗標 Z=1。

ANDF AND w with F

Syntax: ANDF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (W) AND (f) → destination

Status Affected: N, Z

Description: W 累加器中的值與 f 暫存器中的值做邏輯 AND 運算，並將運算結果放回 d 所指定的暫存器中。
 若 $d = 0$ ，則運算後的結果放到 W 累加器中；
 若 $d = 1$ ，則運算後的結果放到 f 暫存器中存放；
 若 $a = 0$ ，則運算後的結果放到目前 RAM 位址中；
 若 $a = 1$ ，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

Words: 1

Cycles: 1

Example 1: ANDF REG, 0

Before Instruction:

W=055H
 REG(080H)=0AAH
 C=DC=N=OV=Z=0

After Instruction:

W=000H
 REG(080H)=0AAH
 Z=1, C=DC=N=OV=0

Remark: 執行結果為 000H，零位旗標 Z=1。
 a=0 為預設值，若程式中 a=0 時，則程式中可以不加入此參數。

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

Before Instruction:

W=080H
 REG(170H)=0FFH
 C=DC=N=OV=Z=0

After Instruction:

W=080H
 REG(170H)=080H
 N=1, C=DC=OV=Z=0

Remark: 執行結果大於 127，視為負數，負號旗標 N=1。

ANDL AND Literal with w

Syntax: ANDL k

Operands: $0 \leq k \leq 255$

Operation: (W) AND k \rightarrow W

Status Affected: N, Z

Description: W 累加器中的值與 k 值做邏輯的 AND 運算，並將運算結果放回 W 累加器中。

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: 執行結果為 000H，零位旗標 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: 執行結果大於 127，視為負數，負號旗標 N=1。

ARLC Another Rotate Left f through Carry

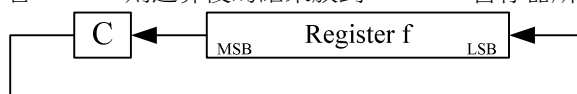
Syntax: ARLC f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f \ll n) \rightarrow \text{destination} \ll n+1 \gg$, $(f \ll 7) \rightarrow \text{Status} \langle C \rangle$,
 $\text{Status} \langle C \rangle \rightarrow \text{destination} \langle 0 \rangle$

Status Affected: C, N, OV, Z

Description: 將 f 暫存內的值與進位旗標 C 一起向左旋轉。(此指令相同於 RLFC 功能，只差異在 OV 旗標)
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



Words: 1

Cycles: 1

Example 1: ARLC REG, 1, 0

Before Instruction:

WREG(02CH)=00FH
 REG(080H)=0AAH
 C=N=OV=Z=0

After Instruction:

WREG(02CH)=00FH
 REG(080H)=054H
 C=OV=1, N=Z=0

Remark: 執行結果 BIT7 由 1→0，所以溢位旗標 OV=1。

Example 2: ARLC REG, 0, 1

Before Instruction:

WREG(02CH)=0FH
 REG(170H)=0EAH
 C=N=OV=Z=0

After Instruction:

WREG(02CH)=0D4H
 REG(170H)=0EAH
 C=N=1, OV=Z=0

Example 3: ARLC REG, 1, 1

Before Instruction:

WREG(02CH)=00FH
 REG(170H)=080H
 C=N=OV=Z=0

After Instruction:

WREG(02CH)=00FH
 REG(170H)=000H
 C=OV=Z=1, N=0

ARRC Another Rotate Right f through Carry

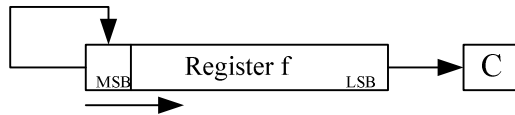
Syntax: ARRC f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f<n>) → destination <n-1 >, (f<7>) → destination < 7 >,
(f<0>) → Status< C >

Status Affected: C, N, Z

Description: 將 f 暫存內的值向右旋轉，BIT0 向右旋轉到進位旗標 C，BIT7 保留在 BIT7 位置。
若 d = 0，則運算後的結果放到 W 累加器中；
若 d = 1，則運算後的結果放到 f 暫存器中；
若 a = 0，則運算後的結果放到目前 RAM 位址中；
若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



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: $0 \leq f \leq 255$; $0 \leq b \leq 7$; $a \in (0, 1)$

Operation: $0 \rightarrow f < b >$

Status Affected: None

Description: 將 f 暫存器中的設定的位元清除為 0。

Words: 1

Cycles: 1

Example 1: BCF REG,2

Before Instruction:

REG(080H)=1111 1111B

After Instruction:

REG(080H)=1111 1011B

Remark: 將 REG 暫存器中 BIT2 清除為 0，其他位元值不變。

BSF Bit Set F

Syntax: BCF f, b, a

Operands: $0 \leq f \leq 255$; $0 \leq b \leq 7$; $a \in (0, 1)$

Operation: $1 \rightarrow f \langle b \rangle$

Status Affected: None

Description: 將 f 暫存器中的設定的位元設定為 1。

Words: 1

Cycles: 1

Example 1: BSF REG,2

Before Instruction:

REG(080H)=00000000B

After Instruction:

REG(080H)=00000100B

Remark: 將 REG 暫存器中 BIT2 設定為 1，其他位元值不變。

BTGF Bit ToGgle F

Syntax: BTGF f, b, a

Operands: $0 \leq f \leq 255$; $0 \leq b \leq 7$; $a \in (0, 1)$

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

Status Affected: None

Description: 將暫存器中的某一個位元取補數。

Words: 1

Cycles: 1

Example 1: BTGF REG, 7, 0

Before Instruction:

REG(080H)=0111 1111B

After Instruction:

REG(080H)=1111 1111B

Remark: 針對 REG 暫存器中 BIT7 取補數。

BTSS Bit Test and Skip if Set

Syntax: BTSS f, b, a
Operands: $0 \leq f \leq 255$; $0 \leq b \leq 7$; $a \in (0, 1)$
Operation: skip if $(f < b) = 1$
Status Affected: None

Description: 比較暫存器中的某一個位元是否為 1。若為 1 則跳過下一個指令，若不為 1 則往下執行。

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: REG 暫存器中 BIT7 為 1，所以跳過下一個指令。

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: REG 暫存器中 BIT7 為 0，所以程式直接往下執行。

BTSZ Bit Test and Skip if Zero

Syntax: BTSZ f, b, a
Operands: $0 \leq f \leq 255$; $0 \leq b \leq 7$; $a \in (0, 1)$
Operation: skip if (f < b >)=0

Status Affected: None

Description: 比較暫存器中的某一個位元是否為 0。若為 0 則跳過下一個指令，若不為 0 則往下執行。

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: REG 暫存器中 BIT7 不為 0，所以程式直接往下執行。

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: REG 暫存器中 BIT7 為 0，所以跳過下一個指令。

CALL subroutine CALL

Syntax: CALL n, s

Operands: $0 \leq n \leq 16384(03FFFH)$; $s \in (0, 1)$

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

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

Description: 呼叫副程式，呼叫的範圍最大到 16Kbytes 的記憶體範圍。
 If s=1, WREG, STATUS, BSRCN 暫存器內的值會被放進相對應的遮蔽暫存器(shadow register)中。
 若呼叫副程式之後堆疊層為該產品最後一層堆疊，則 STKFL 旗標會被設定為 1。
 在 SBMSET1<7>=0 的條件下，堆疊層滿之後再進行 CALL 指令則 STKOV 旗標會被設定為 1，SKERR 也會被設定為 1。PC 正常執行。
 在 SBMSET1<7>=1 的條件下，堆疊層滿之後再進行 CALL 指令則 STKOV 旗標會被設定為 1，SKERR 也會被設定為 1。晶片重置，PC 回到 000H。
 STKFL 或 STKOV 發生時，只要其中一個旗標被清除時，兩者同時都會被清除。

Words: 2

Cycles: 2

Example 1: LABEL: CALL NEXT, 1
 :
 :
 NEXT: NOP

Before Instruction:

PC = address (LABEL)

After Instruction:

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

Remark: s=1, 則 WREG, STATUS, BSRCN 暫存器內的值會被放進相對應的遮蔽暫存器(shadow register)中

CLRF CLear F

Syntax: CLRF f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: $000H \rightarrow f$

Status Affected: None

Description: 將暫存器 f 的值全部清除為 0。

Words: 1

Cycles: 1

Example 1: CLRF REG, 0

Before Instruction:

REG(080H)=055H

After Instruction:

REG(080H)=000H

Remark: REG 暫存器中的數值被清為 0。

COMF COMplement F

Syntax: COMF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $\overline{(f)} \rightarrow \text{destination}$

Status Affected: N, Z

Description: 將暫存器中的值取補數，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSR CN 暫存器所指定的 RAM 位址中。

Words: 1

Cycles: 1

Example 1: COMF REG, 0, 0

Before Instruction:

WREG(02CH)=055H
 REG(080H)=0FFH
 N=Z=0

After Instruction:

WREG(02CH)=000H
 REG(080H)=0FFH
 Z=1, N=0

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=055H
 N=Z=0

After Instruction:

WREG(02CH)=055H
 REG(170H)=0AAH
 N=1, Z=0

CPSE ComPare f with w, Skip if f Equal w

Syntax: CPSE f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: skip if (f) = (W)

Status Affected: None

Description: 將暫存器的值與 W 累加器的值作比較，
 若是兩個值相等則跳過下一個指令，若大於或是小於則往下執行。

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: f 暫存器和 W 累加器數值相等，所以跳過下一個指令。

Example 2:
 CPSE REG, 1 (if BSRCN=001H)
 MVL 001H
 NOP

Before Instruction:

WREG(02CH)=000H
 REG(170H)=07FH

After Instruction:

WREG(02CH)=001H
 REG(170H)=07FH

Remark: f 暫存器和 W 累加器數值不相等，所以程式繼續往下執行。

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

Syntax: CPSG f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: skip if $(f) > (W)$

Status Affected: None

Description: 將暫存器的值與 W 累加器的值作比較，
 若是暫存器的值大於 W 累加器的值則跳過下一個指令，若小於或是等於則往下執行。

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: f 暫存器的數值大於 W 累加器數值，所以跳過下一個指令。

Example 2:
 CPSG REG, 1 (if BSRCN=001H)
 MVL 00FH
 NOP

Before Instruction:

WREG(02CH)=005H
 REG(170H)=005H

After Instruction:

WREG(02CH)=00FH
 REG(170H)=005H

Remark: f 暫存器數值等於 W 累加器數值，所以程式繼續往下執行。

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

Syntax: CPSL f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: skip if $(f) < (W)$

Status Affected: None

Description: 將暫存器的值與 W 累加器的值作比較，
 若是暫存器的值小於 W 累加器的值則跳過下一個指令，若大於或是等於則往下執行。

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: f 暫存器的數值小於 W 累加器數值，所以跳過下一個指令。

Example 2:
 CPSL REG, 1 (if BSRCN=001H)
 MVL 00FH
 NOP

Before Instruction:

WREG(02CH)=005H
 REG(170H)=005H

After Instruction:

WREG(02CH)=00FH
 REG(170H)=005H

Remark: f 暫存器數值等於 W 累加器數值，所以程式繼續往下執行。

CWDT Clear WatchDog Timer

Syntax: CWDT

Operands: None

Operation: 000H → Watch dog counter

Status Affected: Pstatus<TO>

Description: 將看門狗計時器的值全部清除為 0。

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: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) - 1 \rightarrow \text{destination}$

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

Description: 將暫存器 f 的值減 1，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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 未被借位，所以 C=DC=1；執行後結果大於 127，所以 N=1。

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=000H
 C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=055H
 REG(170H)=0FFH
 N=1, C=DC=OV=Z=0

Remark: C, DC 皆被借位，所以 C=DC=0；執行後結果仍大於 127，所以 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: 只有 DC 被借位，所以 DC=0, C=1；執行後 BIT7 被變動，由 1 變 0，所以 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: C, DC 未被借位，所以 C=DC=1；執行後結果為 0，所以 Z=1。

DCSUZ DeCrement f, Skip if Un-Zero

Syntax: DCSUZ f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) - 1 \rightarrow \text{destination}$, skip if destination $\neq 0$

Status Affected: None

Description: 將暫存器的值減 1 後與 0 作比較，若是暫存器的值不等於 0 則跳過下一個指令，若等於 0 則往下執行，並將運算結果放回 d 所指定的暫存器中。
 若 $d = 0$ ，則運算後的結果放到 W 累加器中；
 若 $d = 1$ ，則運算後的結果放到 f 暫存器中；
 若 $a = 0$ ，則運算後的結果放到目前 RAM 位址中；
 若 $a = 1$ ，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果為 0，所以繼續往下執行程式段，執行結果被放回 REG 暫存器。

Example 2:

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=000H

After Instruction:

WREG(02CH)=0FFH
 REG(170H)=000H

Remark: 執行結果不為 0，所以跳過下一個指令，執行結果被放回 W 累加器。

DCSZ DeCrement f, Skip if Zero

Syntax: DCSZ f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) - 1 \rightarrow \text{destination}$, skip if destination=0

Status Affected: None

Description: 將暫存器的值減 1 後與 0 作比較，若是暫存器的值等於 0 則跳過下一個指令，若不等於 0 則往下執行，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果為 0，所以跳過下一個指令。

Example 2:

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=000H

After Instruction:

WREG(02CH)=00AH
 REG(170H)=0FFH

Remark: 執行結果不為 0，所以繼續往下執行程式段，執行結果被放回 REG 暫存器。

IDLE IDLE mode

Syntax: IDLE
Operands: None
Operation: CPU Halt

Status Affected: Pstatus<IDLEB>

Description: CPU 進入暫停模式，程式空指令執行動作。
 IDLE 指令後，建議加上 NOP 指令。

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: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) + 1 \rightarrow \text{destination}$

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

Description: 將暫存器 f 的值加 1，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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 進位所以 C=DC=1；執行後結果等於 0，所以 Z=1。

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=00FH
 C=DC=N=OV=Z=0

After Instruction:

WREG(02CH)=055H
 REG(170H)=010H
 DC=1, C=N=OV=Z=0

Remark: DC 進位所以 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 進位所以 DC=1；執行後 BIT7 被變動，由 0 變 1，所以 OV=1；執行結果大於 127，所以 N=1。

INSUZ INcrement f, Skip if Un-Zero

Syntax: INSUZ f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) + 1 \rightarrow \text{destination}$, skip if destination $\neq 0$

Status Affected: None

Description: 將暫存器的值加 1 後與 0 作比較，若是暫存器的值不等於 0 則跳過下一個指令，若等於 0 則往下執行，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果為 0，所以繼續往下執行程式段，執行結果被放回 REG 暫存器。

Example 2:

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

Before Instruction:

WREG(02CH)=055H
 REG(170H)=000H

After Instruction:

WREG(02CH)=001H
 REG(170H)=000H

Remark: 執行結果不為 0，所以跳過下一個指令，執行結果被放回 W 累加器。

INSZ INcrement f, Skip if Zero

Syntax: INSZ f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) + 1 \rightarrow \text{destination}$, skip if destination=0

Status Affected: None

Description: 將暫存器的值加 1 後與 0 作比較，若是暫存器的值等於 0 則跳過下一個指令，若不等於 0 則往下執行，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果為 0，所以跳過下一個指令。

Example 2: INSZ REG, 1, 1 (if BSRCN=001H)
 MVL 00AH
 NOP

Before Instruction:

WREG(02CH)=055H
 REG(170H)=000H

After Instruction:

WREG(02CH)=00AH
 REG(170H)=001H

Remark: 執行結果不為 0，所以繼續往下執行程式段，執行結果被放回 REG 暫存器。

IORF Inclusive OR w with F

Syntax: IORF f, d, a
Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$
Operation: (W) OR (f) \rightarrow destination
Status Affected: N, Z

Description: W 累加器中的值與 f 暫存器中的值作邏輯的 OR 運算，並將運算結果放回 d 所指定的暫存器中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果大於 127，所以 N=1。

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

Before Instruction:

WREG(02CH)=00FH
 REG(170H)=0F0H
 N=Z=0

After Instruction:

WREG(02CH)=00FH
 REG(170H)=0FFH
 N=1, Z=0

Remark: 執行結果大於 127，所以 N=1。

IORL Inclusive OR Literal with w

Syntax: IORL k

Operands: $0 \leq k \leq 255$

Operation: (W) OR k \rightarrow W

Status Affected: N, Z

Description: W 累加器中的值與 k 值作邏輯的 OR 運算，並將運算結果放回 W 累加器中。

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: 執行結果大於 127，所以 N=1。

Example 2: IORL 000H

Before Instruction:

WREG(02CH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

Z=1, N=0

Remark: 執行結果等於 0，所以 Z=1。

JMP unconditional JuMP

Syntax: JMP n

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

Operation: $n \rightarrow PC$

Status Affected: None

Description: 無條件跳躍至指定的位址 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: 當狀態暫存器中的負號旗標 N 等於 1 時，就跳到指定的位址 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: 當狀態暫存器中的進位旗標 C 等於 0 時，就跳到指定的位址 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: $-128 \leq n \leq 127$

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

Status Affected: None

Description: 當狀態暫存器中的負號旗標 N 等於 0 時，就跳到指定的位址 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: 當狀態暫存器中的溢位旗標 OV 等於 0 時，就跳到指定的位址 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: 當狀態暫存器中的零位旗標 Z 等於 0 時，就跳到指定的位址 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)

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: 當狀態暫存器中的零位旗標 Z 等於 1 時，就跳到指定的位址 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)

MULF MULtiply w with F

Syntax: MULF f, a

Operands: $0 \leq f \leq 255$; $a \in (0, 1)$

Operation: $(W) \times (f) \rightarrow \text{PRODH (high byte), PRODL (low byte)}$

Status Affected: None

Description: 將 W 累加器中的值與暫存器 f 的值相乘，並將結果放到 PRODH, PRODL 這二個暫存器中。
 $a = 0$ or $a = 1$ 的設定須取決 f 暫存器位於 RAM 位址：
 若 $a = 0$ ，則表示 f 暫存器存在於 080H 到 0FFH 所指定的 RAM 位址中(BSRCN=000H)。
 若 $a = 1$ ，則表示 f 暫存器存在於 100H 到 17FH 所指定的 RAM 位址中(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: 使用直接定址執行乘法運算。

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
 PRODL=0F1H

Remark: 使用間接定址執行乘法運算。

MULL MULTiPLY Literal with w

Syntax: MULL k

Operands: $0 \leq k \leq 255$

Operation: $(W) \times k \rightarrow \text{PRODH (high byte), PRODL (low byte)}$

Status Affected: None

Description: 將常數 k 與 W 累加器中的值相乘，並將結果放到 PRODH, PRODL 這二個暫存器。

Words: 1

Cycles: 2

Example 1: MULL 0FFH

Before Instruction:

WREG(02CH)=00FH

PRODH=??

PRODL=??

After Instruction:

WREG(02CH)=00FH

PRODH=00EH

PRODL=0F1H

MVF MoVe F to w or MoVe w to F

Syntax: MVF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f) → W, or (W) → f

Status Affected: None

Description: 將 f 暫存器的值搬到 W 累加器中；或是將 W 累加器的值搬到 f 暫存器中。
 若 d = 0，則表示將 f 暫存器的值搬到 W 累加器中；
 若 d = 1，則表示將 W 累加器的值搬到 f 暫存器中；
 a = 0 或 a = 1 的設定須取決 f 暫存器位於 RAM 位址：
 若 a = 0，則表示 f 暫存器存在於 080H 到 0FFH 所指定的 RAM 位址中(BSRCN=000H)。
 若 a = 1，則表示 f 暫存器存在於 100H 到 17FH 所指定的 RAM 位址中(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，表示將 REG 暫存器的值搬到 W 累加器中。

Example 2: MVF REG1, 1, 1 (if BSRCN=001H)

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，表示將 W 累加器的值搬到 f 暫存器中。

MVFF MoVe F to F

Syntax: MVFF fs, fd

Operands: $0 \leq fs \leq 1279(04FFH)$; $0 \leq fd \leq 1279(04FFH)$

Operation: (fs) \rightarrow fd

Status Affected: None

Description: 將暫存器 fs 的數值般到暫存器 fd 去。

Words: 2

Cycles: 2

Example 1: MVFF REG, REG1

Before Instruction:

REG=055H

REG1=0AAH

After Instruction:

REG=055H

REG1=055H

Remark: 將暫存器 fs 的數值般到暫存器 fd 去，資料移轉過程不經由 W 累加器。

MVL MoVe Literal to w

Syntax: MVL k

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow W$

Status Affected: None

Description: 將常數 k 搬到 W 累加器中。

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: $0 \leq k \leq 16384(03FFFh)$

Operation: $k \rightarrow TBLPTR$ (TBLPTRH, TBLPTRL)

Status Affected: None

Description: MVLP 是一個設定程式記憶體指標的指令，多用在查表指令搭配 TBLR 使用。

Words: 2

Cycles: 2

Example 1: MVLP 001FF0H

Before Instruction:

TBLPTRH=000H

TBLPTRL=000H

After Instruction:

TBLPTRH=01FH

TBLPTRL=0F0H

Remark: 進行程式記憶體指標設定，將常數 k 載入 TBLPTR 暫存器。

NOP No OPeration

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Description: 不做任何運算，只延遲一個指令時間。

Words: 1

Cycles: 1

Example 1: NOP

Remark: 空指令，只做一個指令週期的延遲時間。

POP POP return stack

Syntax: POP

Operands: None

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

Status Affected: None

Description: 將堆疊指標(Stack Pointer)所指向堆疊層中的堆疊值丟棄，並將堆疊指標暫存器減 1 後，取出該堆疊指標所指向堆疊層中堆疊的值，將其放在 TOS 這個暫存器中。而 TOS 暫存器被分為 TOSH, TOSL。

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=LABEL1

Remark: 如果 STKPTR=00H，則執行 POP 指令並不會造成任何影響，TOS 仍為 0，STKPTR 亦為 0。

RET RETurn from subroutine

Syntax: RET s

Operands: s ∈ (0, 1)

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

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

Description: 離開副程式，並將推疊指標暫存器的數值存到 PC 中。
 If s=1，遮蔽暫存器(shadow register)中的值，會被放回相對應暫存器中(WREG, STATUS, BSRCN)。
 當未呼叫副程式且 STKPTR=000H 時，執行 RET 指令時會造成晶片重置，STKUN 旗標會被設定為 1，SKERR 旗標會被設定為 1。

Words: 1

Cycles: 2

Example 1: RET 1

Before Instruction:

None

After Instruction:

PC=TOS

WREG = WREGSDW

BSRCN = BSRSDW

STATUS = STASDW

Remark: s=1，所以遮蔽暫存器(shadow register)中的值，會被放回相對應暫存器中(WREG, STATUS, BSRCN)。

RETI RETurn from Interrupt

Syntax: RETI s

Operands: s ∈ (0, 1)

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

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

Description: 離開中斷程式副程式，並將推疊指標暫存器的值存到 PC 中，中斷致能接腳再度被設定為 1。
 If s=1, 遮蔽暫存器(shadow register)中的值，會被放回相對應暫存器中(WREG, STATUS, BSRCN)。
 當未呼叫副程式且 STKPTR=000H 時，執行 RETI 指令時會造成晶片重置，STKUN 旗標會被設定為 1，SKERR 旗標會被設定為 1。

Words: 1

Cycles: 2

Example 1: RETI 1

Before Instruction:

None

After Instruction:

PC=TOS
 WREG = WREGSDW
 BSRCN = BSRSDW
 STATUS = STASDW
 GIE=1

RETL RETurn Literal to w

Syntax: RETL k

Operands: $0 \leq k \leq 255$

Operation: $k \rightarrow W, (TOS) \rightarrow PC$

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

Description: 從副程式返回主程式的指令，但本指令在返回的時候，還會順便將常數 k 載入到 W 累加器中。此指令多半在查表法時會用到。
當未呼叫副程式且 STKPTR=000H 時，執行 RETL 指令時會造成晶片重置，STKUN 旗標會被設定為 1，SKERR 旗標會被設定為 1。

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: 返回主程式時，順便將常數 k 載入到 W 累加器中。
此範例為藉由寫入 PCLATL 的 Offset 數值來決定取得 TABLE 表中第幾筆數值。

RJ unconditional Relative Jump

Syntax: RJ n

Operands: $-1024 \leq n \leq 1023$

Operation: $n \rightarrow PC$

Status Affected: None

Description: 無條件跳躍至指定的位址 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)

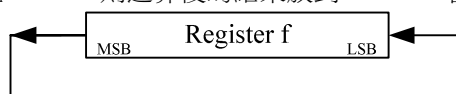
Syntax: RLF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f<n>) → destination <n+1 > ,
(f<7>) → destination < 0 >

Status Affected: N, Z

Description: 將 f 暫存內的值向左旋轉。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



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, 1

Before Instruction:

WREG(02CH)=00FH
 REG(17FH)=000H
 N=Z=0

After Instruction:

WREG(02CH)=000H
 REG(17FH)=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

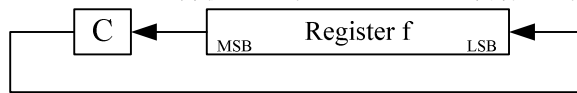
Syntax: RLFC f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f<n>) → destination <n+1 >,
 (f<7>) → Status< C >,
 Status< C > → destination < 0 >

Status Affected: C, N, Z

Description: 將 f 暫存內的值與進位旗標 C 一起向左旋轉。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



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, 1

Before Instruction:

WREG(02CH)=0FH
 REG(170H)=0EAH
 C=N=Z=0

After Instruction:

WREG(02CH)=0D4H
 REG(170H)=0EAH
 C=N=1, Z=0

Example 3: RLFC REG, 1, 1

Before Instruction:

WREG(02CH)=00FH
 REG(170H)=080H
 C=N=Z=0

After Instruction:

WREG(02CH)=00FH
 REG(170H)=000H
 C=Z=1, N=0

RRF Rotate Right F (no carry)

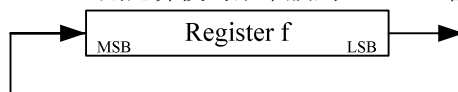
Syntax: RRF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f<n>) → destination <n - 1 >,
(f<0>) → destination < 7 >

Status Affected: N, Z

Description: 將 f 暫存內的值向右旋轉。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



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, 1

Before Instruction:

WREG(02CH)=00FH
 REG(17FH)=000H
 N=Z=0

After Instruction:

WREG(02CH)=000H
 REG(17FH)=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

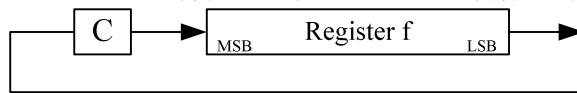
Syntax: RRFC f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (f<n>) → destination <n-1 > ,
 (f<0>) → Status< C > ,
 Status< C > → destination < 7 >

Status Affected: C, N, Z

Description: 將 f 暫存內的值與進位旗標 C 一起向右旋轉。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。



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, 1

Before Instruction:

WREG(02CH)=00FH
 REG(17FH)=055H
 C=1, N=Z=0

After Instruction:

WREG(02CH)=0AAH
 REG(17FH)=055H
 C=N=1, Z=0

Example 3: RRFC 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

SETF SET F

Syntax: SETF f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: $0FFH \rightarrow f$
Status Affected: None

Description: 將 f 暫存內的值全部設定為 1。
 a = 0 or a = 1 的設定須取決 f 暫存器位於 RAM 位址：
 若 a = 0，則表示 f 暫存器存在於 080H 到 0FFH 所指定的 RAM 位址中(BSRCN=000H)。
 若 a = 1，則表示 f 暫存器存在於 100H 到 17FH 所指定的 RAM 位址中(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: Pstatus<PD>

Description: CPU 進入睡眠狀態，震盪器停止動作。

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: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) - (W) - \overline{(C)} \rightarrow \text{destination}$

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

Description: f 暫存器的值減去 W 累加器與進位旗標 C 的反向值，並將結果放置 d。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSR CN 暫存器所指定的 RAM 位址中。

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, DC 皆未借位，所以 C=DC=1，執行結果為 0，所以 Z=1。

Example 2: SUBF REG, 1, 1

Before Instruction:

WREG=000H
 REG(17FH)=080H
 C=DC=N=OV=Z=0

After Instruction:

WREG=000H
 REG(17FH)=07FH
 C=OV=1, DC= N=Z=0

Remark: C 未借位，所以 C=1；DC 被借位，所以 DC=0；
 OV=1 條件判斷標準：(負數) - (正數) = 正數 or (正數) - (負數) = 負數；
 此範例符合(負數) - (正數) = 正數，所以 OV=1。

SUBF SUBtract w from F

Syntax: SUBF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: $(f) - (W) \rightarrow \text{destination}$

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

Description: 將 f 暫存器的值減掉 W 累加器的值，並將結果放置 d。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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, DC 皆未借位，所以 C=DC=1，執行結果為 0，所以 Z=1。

Example 2: SUBF REG, 1, 1

Before Instruction:

WREG=001H
 REG(17FH)=080H
 C=DC=N=OV=Z=0

After Instruction:

WREG=001H
 REG(17FH)=07FH
 C=OV=1, DC= N=Z=0

Remark: C 未借位，所以 C=1；DC 被借位，所以 DC=0；
 OV=1 條件判斷標準：(負數) - (正數) = 正數 or (正數) - (負數) = 負數；
 此範例符合(負數) - (正數) = 正數，所以 OV=1。

SUBL SUBtract w from Literal

Syntax: SUBL k

Operands: $0 \leq k \leq 255$

Operation: $K - (W) \rightarrow W$

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

Description: 將常數 k 與 W 累加器的值相減並將結果放回 W 累加器中。

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 皆未借位，所以 C=DC=1，執行結果為 0，所以 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 未借位，所以 C=1；DC 被借位，所以 DC=0；
OV=1 條件判斷標準：(負數) - (正數) = 正數 or (正數) - (負數) = 負數；
此範例符合(負數) - (正數) = 正數，所以 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 未借位，所以 DC=1；C 被借位，所以 C=0；執行結果大於 127，所以 N=1
此範例符合(正數) - (負數) = 負數，所以 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 均被借位，所以 C=DC=0；執行結果大於 127，所以 N=1。

SWPF SWaP F

Syntax: SWPF f, d, a
Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$
Operation: (f<3:0>) → destination<7:4>
 (f<7:4>) → destination<3:0>

Status Affected: None

Description: 將 f 暫存器內的高 4 位元值與低 4 位元值做交換。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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 是一個讀取程式記憶體內容的指令，多用在查表指令使用，目前它提供以下 2 種用法：

- TBLR *
 以 TBLPTRH, TBLPTRL 暫存器數值為位址指標，讀取對應的程式記憶體內容至 TBLD (TBLDH, TBLDL)暫存器中。
- TBLR *+
 以 TBLPTRH, TBLPTRL 暫存器數值為位址指標，讀取對應的程式記憶體內容至 TBLD (TBLDH, TBLDL)暫存器中，然後將位址指標自動加 1。

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: 取出 TBLPTR 位址處的 2 bytes 資料放回 TBLD (TBLDH, TBLDL)，TBLPTR 指標內容不變。

Example 2: TBLR *+

Before Instruction:

TBLDH, TBLDL= 0123H
 At TBLPTR=0017FFH
 Address(0017FFH) = data (5678H)

After Instruction:

TBLDH, TBLDL= 5678H
 TBLPTR=001800H

Remark: 取出 TBLPTR 位址處的 2 bytes 資料放回 TBLD (TBLDH, TBLDL)，TBLPTR 指標內容+1。

TFSZ Test F, Skip if Zero

Syntax: TFSZ f, a
Operands: $0 \leq f \leq 255$; $a \in (0, 1)$
Operation: skip if $f = 0$
Status Affected: None

Description: 假如 f 暫存器內的值等於 0 則跳過下一個指令；若 f 暫存器內的值不等於 0，則執行下一個指令。
 a = 0 or a = 1 的設定須取決 f 暫存器位於 RAM 位址：
 若 a = 0，則表示 f 暫存器存在於 080H 到 0FFH 所指定的 RAM 位址中(BSRCN=000H)。
 若 a = 1，則表示 f 暫存器存在於 100H 到 17FH 所指定的 RAM 位址中(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)=000H

Remark: f 暫存器的數值等於 0，所以跳過下一個指令。

Example 2: TFSZ REG, 1 (if BSRCN=001H)
 MVL 00FH
 NOP

Before Instruction:

WREG(02CH)=005H
 REG(170H)=001H

After Instruction:

WREG(02CH)=00FH
 REG(170H)=001H

Remark: f 暫存器數值不等於 0，所以程式繼續往下執行。

XORF eXclusive OR w with F

Syntax: XORF f, d, a

Operands: $0 \leq f \leq 255$; $d \in (0, 1)$; $a \in (0, 1)$

Operation: (W) XOR (f) \rightarrow destination

Status Affected: N, Z

Description: 將常數 k 與 W 累加器的值做邏輯互斥或(XOR)運算，並將結果放回 d 中。
 若 d = 0，則運算後的結果放到 W 累加器中；
 若 d = 1，則運算後的結果放到 f 暫存器中；
 若 a = 0，則運算後的結果放到目前 RAM 位址中；
 若 a = 1，則運算後的結果放到 BSRCN 暫存器所指定的 RAM 位址中。

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: 執行結果大於 127，所以 N=1。 XOR: 兩者同則結果為 0；兩者不同則結果為 1。

Example 2: XORF REG, 1, 1

Before Instruction:

WREG(02CH)=0FFH
 REG(170H)=0FFH
 N=Z=0

After Instruction:

WREG(02CH)=0FFH
 REG(170H)=000H
 Z=1, N=0

Remark: 執行結果等於 0，所以 Z=1。 XOR: 兩者同則結果為 0；兩者不同則結果為 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: 執行結果等於 0，所以 Z=1。 XOR: 兩者同則結果為 0；兩者不同則結果為 1。

XORL eXclusive OR Literal with w

Syntax: XORL k

Operands: $0 \leq f \leq 255$

Operation: (W) XOR k \rightarrow W

Status Affected: N, Z

Description: 將常數 k 與 W 累加器的值做邏輯互斥或(XOR)的運算，並將結果放回 W 累加器中。

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: 執行結果大於 127，所以 N=1。
XOR: 兩者同則結果為 0；兩者不同則結果為 1。

Example 2: XORL 0FFH

Before Instruction:

WREG(02CH)=0FFH

N=Z=0

After Instruction:

WREG(02CH)=000H

Z=1, N=0

Remark: 執行結果等於 0，所以 Z=1。
XOR: 兩者同則結果為 0；兩者不同則結果為 1。

Example 3: XORL 000H

Before Instruction:

WREG(02CH)=000H

N=Z=0

After Instruction:

WREG(02CH)=000H

Z=1, N=0

Remark: 執行結果等於 0，所以 Z=1。
XOR: 兩者同則結果為 0；兩者不同則結果為 1。

修訂紀錄

以下描述本文件差異較大的地方，而標點符號與字形的改變不在此描述範圍。

版本	頁次	變更摘要
V01	ALL	初版發行
V02	ALL	修正格式
V03	-	刪除 DAW 指令