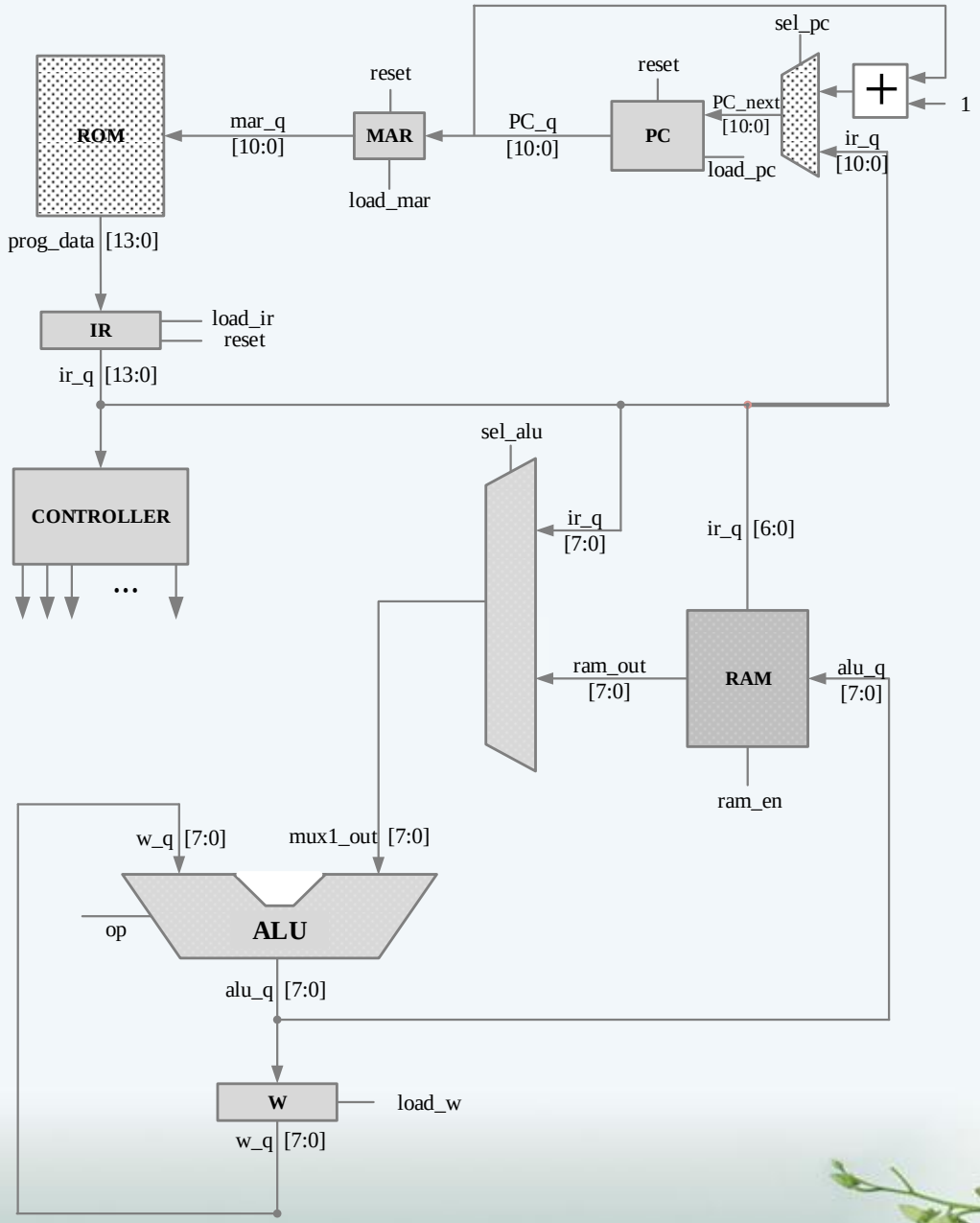
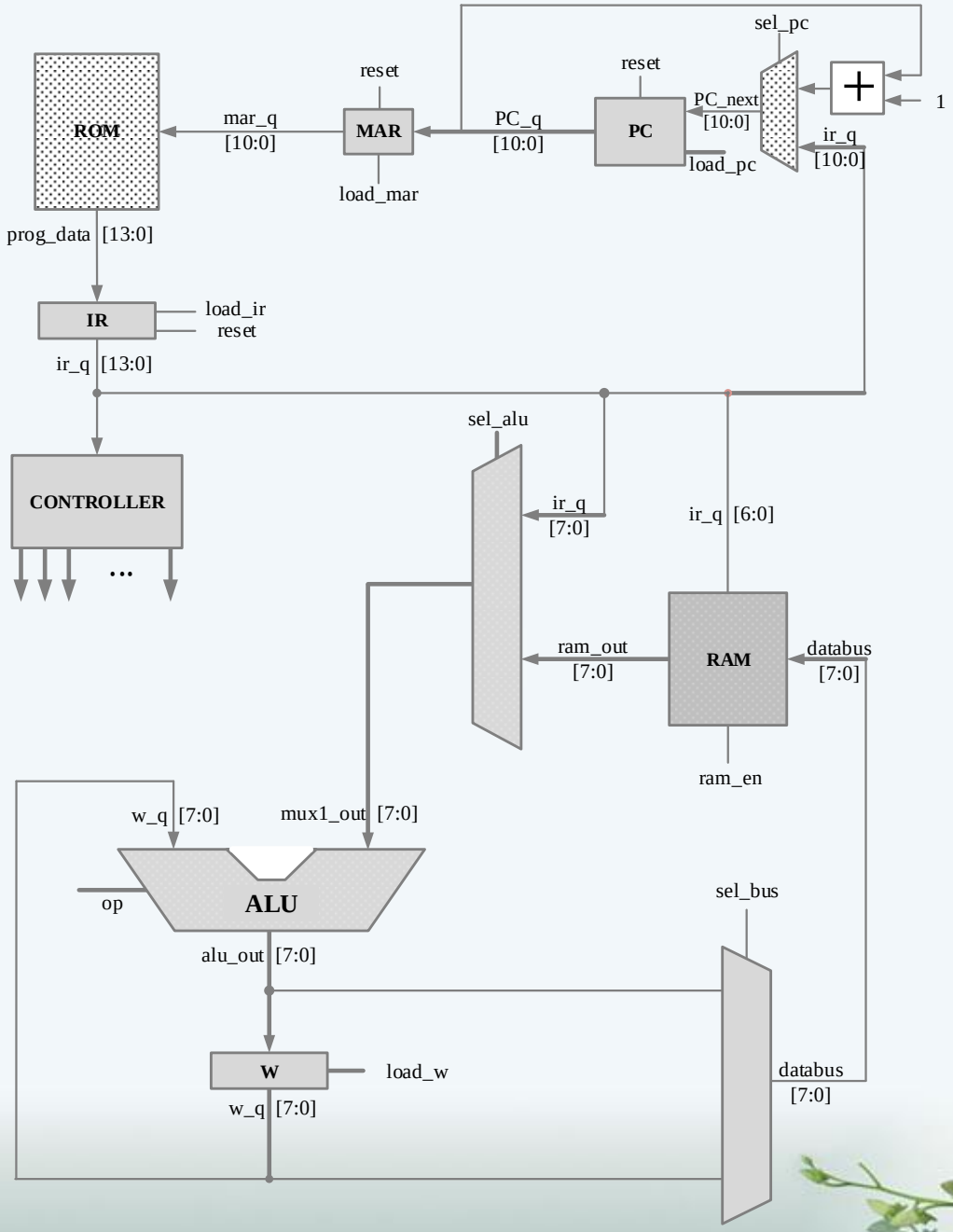
暫存器定址

舊架構圖


sel_pc
reset
reset
ROM
mar_q
[10:0]
MAR
PC_q
[10:0]
PC
PC_next
[10:0]
1
load_pc
ir_q
[10:0]
load_mar
prog_data [13:0]
IR
load_ir
reset
ir_q [13:0]
sel_alu
CONTROLLER
ir_q
[7:0]
ir_q [6:0]
...
ram_out
[7:0]
RAM
alu_q
[7:0]
ram_en
w_q [7:0]
mux1_out [7:0]
op
ALU
alu_q [7:0]
W
load_w
w_q [7:0]

新架構圖


sel_pc
reset
reset
ROM
mar_q
[10:0]
MAR
PC_q
[10:0]
PC
PC_next
[10:0]
1
load_mar
load_pc
ir_q
[10:0]
prog_data [13:0]
IR
load_ir
reset
ir_q [13:0]
sel_alu
CONTROLLER
...
ir_q
[7:0]
ir_q [6:0]
ram_out
[7:0]
RAM
databus
[7:0]
ram_en
w_q [7:0]
mux1_out [7:0]
op
ALU
sel_bus
alu_out [7:0]
W
load_w
w_q [7:0]
databus
[7:0]

指令資料流

49個指令分成八個類別，
從八個類別中各挑出部分指令做控制訊號及資料流向範例。

各指令執行所需時間不盡相同，大致上可由類別區分：

- 一個時間週期：
  Literal Operations、Inherent Operations。

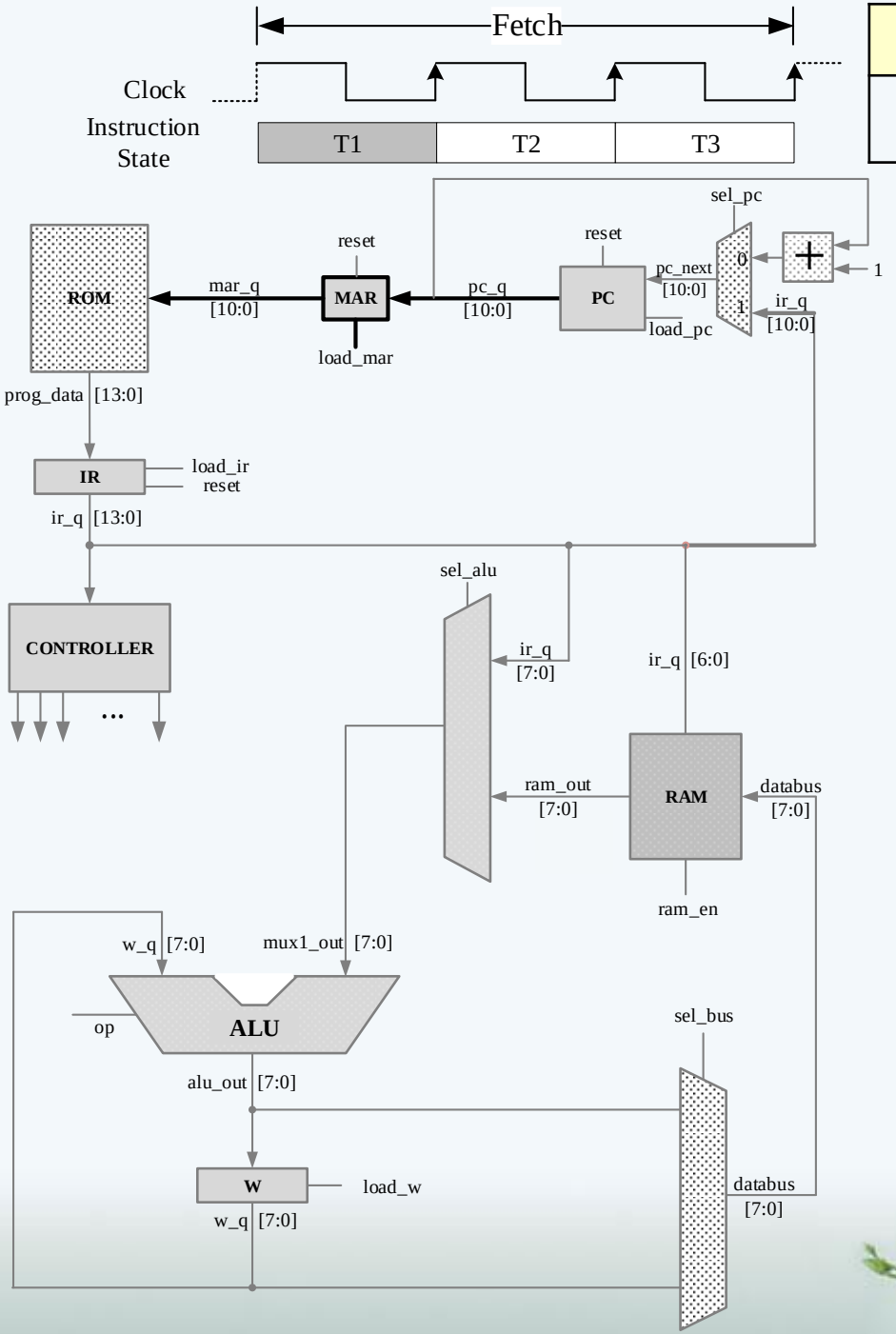
- 兩個時間週期：
  Byte-oriented File Register Operations、Bit-oriented File Register Operations、
  Bit-oriented Skip Operations。

- 三個時間週期：
  Byte-oriented Skip Operations、Control Operations、C-Compiler Optimized。

T_1、T_2及T_3擷取階段，控制訊號均相同，如下表所示。

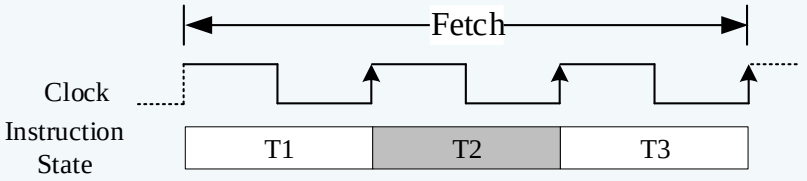
狀態	動作	控制訊號
T_1	MAR←PC	load_mar
T_2	PC←PC+1	sel_pc; load_pc
T_3	IR←ROM[MAR]	load_ir

Fetch T1

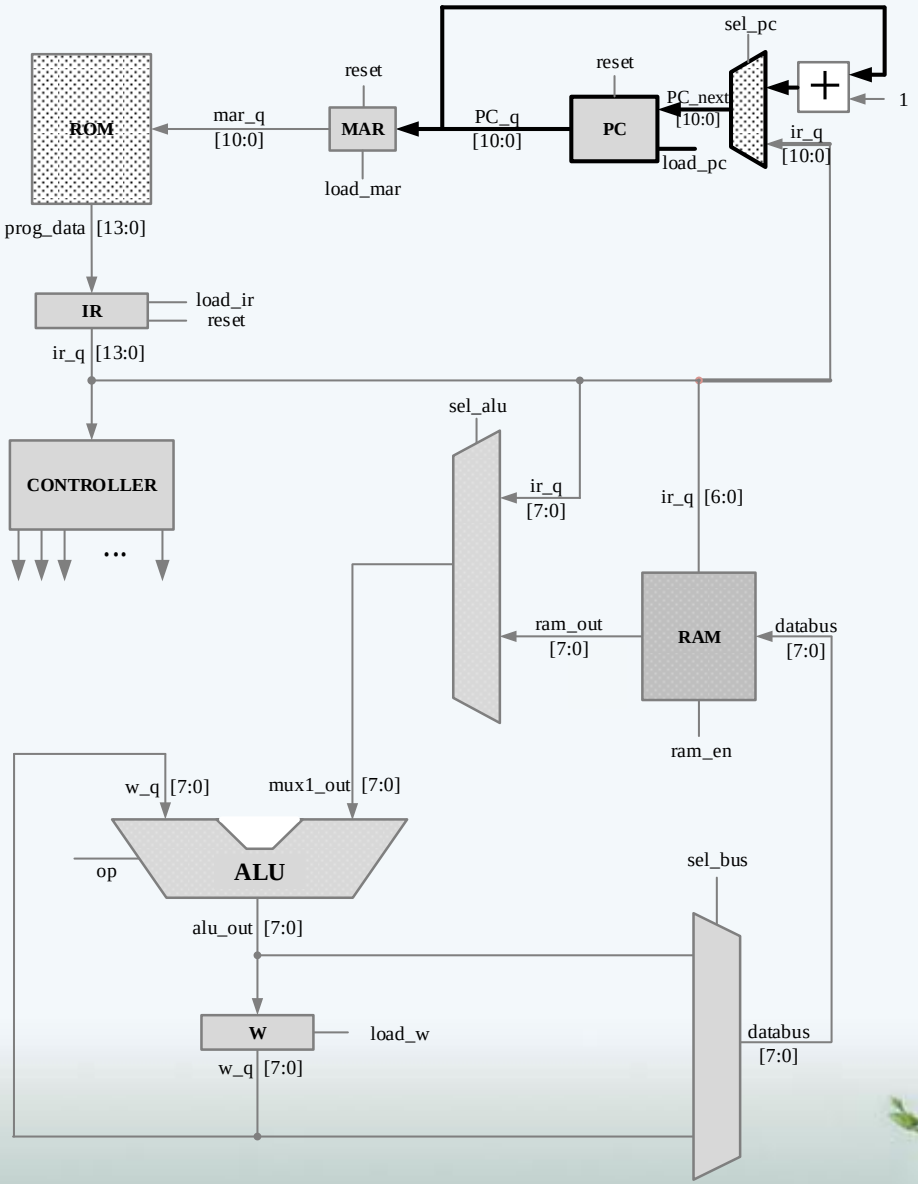


狀態	動作	控制訊號
T_1	MAR←PC	load_mar

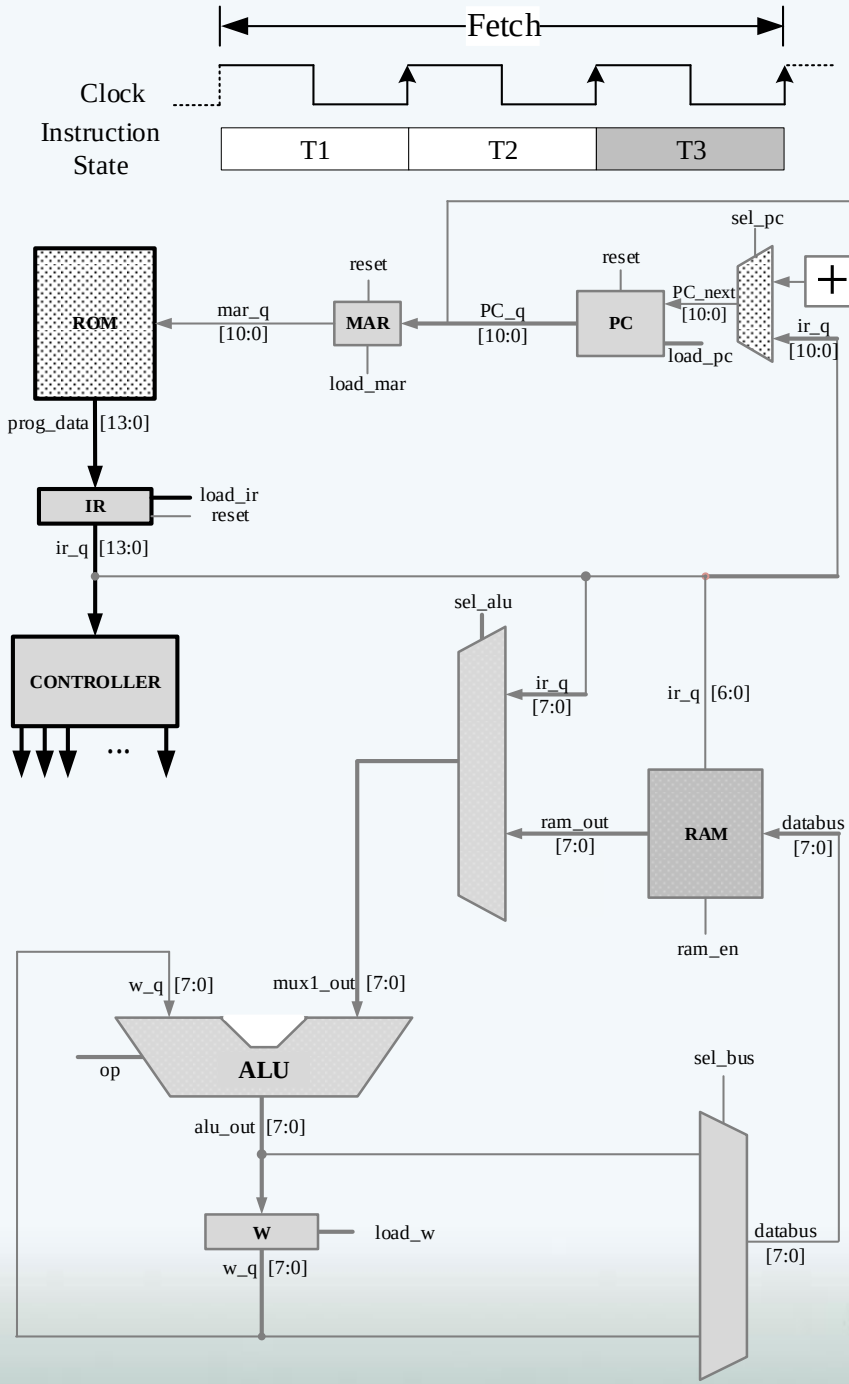
Fetch T2



狀態	動作	控制訊號
T_2	PC←PC+1	load_pc

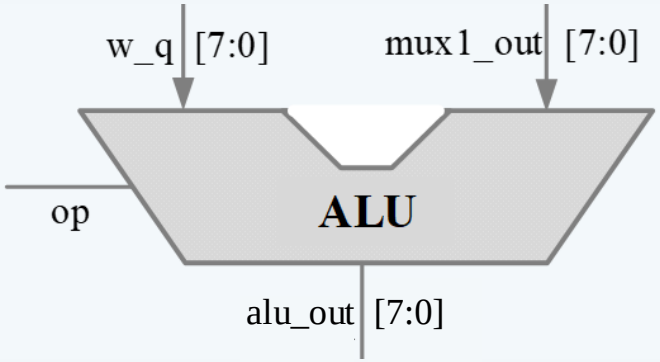


Fetch T3



狀態	動作	控制訊號
T_3	IR←ROM[MAR]	load_ir

ALU




```
//ALU
always_comb
begin
    case(op)
        0: alu_q = mux1_out + w_q;
        1: alu_q = mux1_out - w_q;
        2: alu_q = mux1_out & w_q;
        3: alu_q = mux1_out | w_q;
        4: alu_q = mux1_out ^ w_q;
        5: alu_q = mux1_out;
        6: alu_q = mux1_out + 1;
        7: alu_q = mux1_out - 1;
        8: alu_q = 0;
        9: alu_q = ~mux1_out;
        default: alu_q = mux1_out + w_q;
    endcase
end
```

BYTE-ORIENTED FILE REGISTER OPERATIONS

PIC16F1826 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSB LSB				Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS (2)									
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	2
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C, DC, Z	2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	2

Note

1：If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

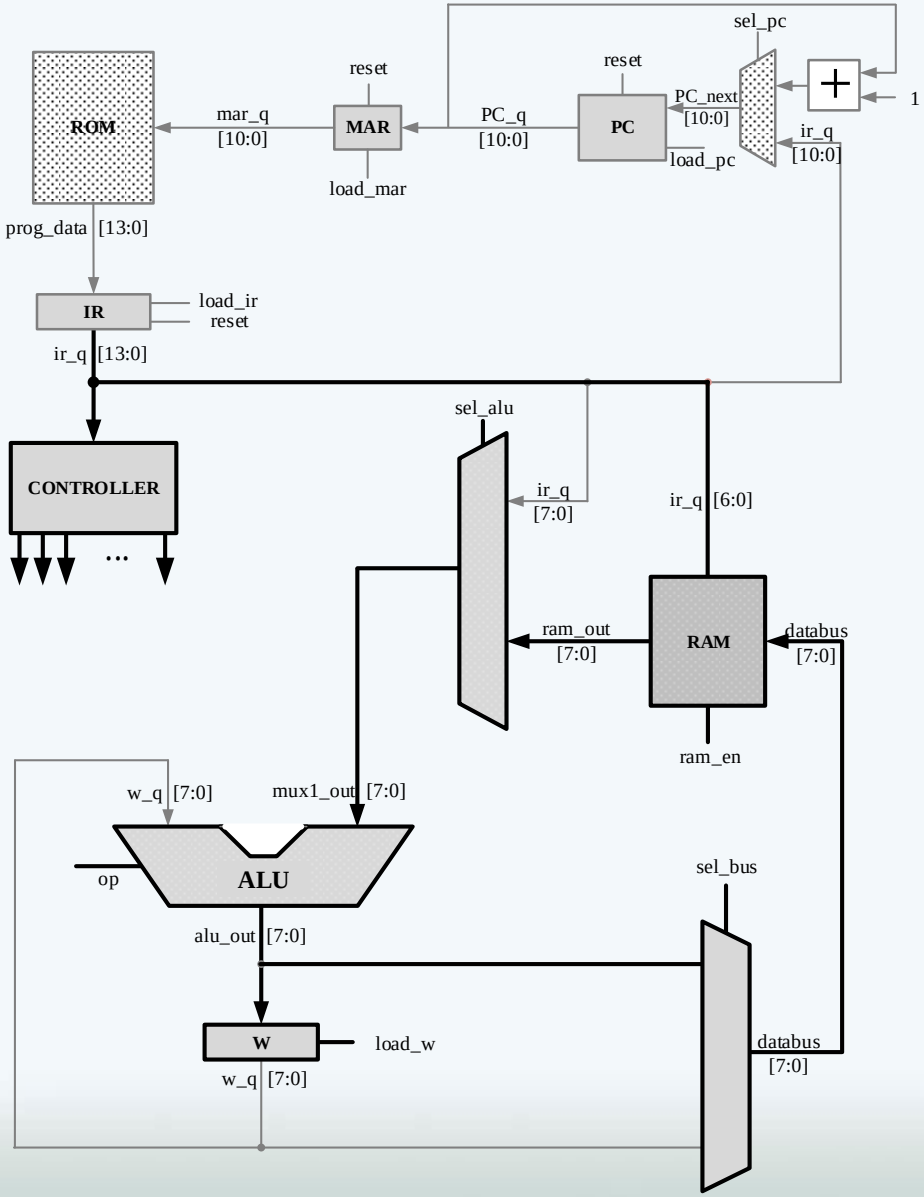
2：If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

TABLE 29-1: OPCODE FIELD DESCRIPTIONS

Field	Description
f	Register file address (0x00 to 0x7F)
W	Working register (accumulator)
b	Bit address within an 8-bit file register
k	Literal field, constant data or label
x	Don't care location (= 0 or 1). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
d	Destination select; d = 0: store result in W, d = 1: store result in file register f. Default is d = 1.
n	FSR or INDF number. (0-1)
mm	Pre-post increment-decrement mode selection

INCF：F遞增1

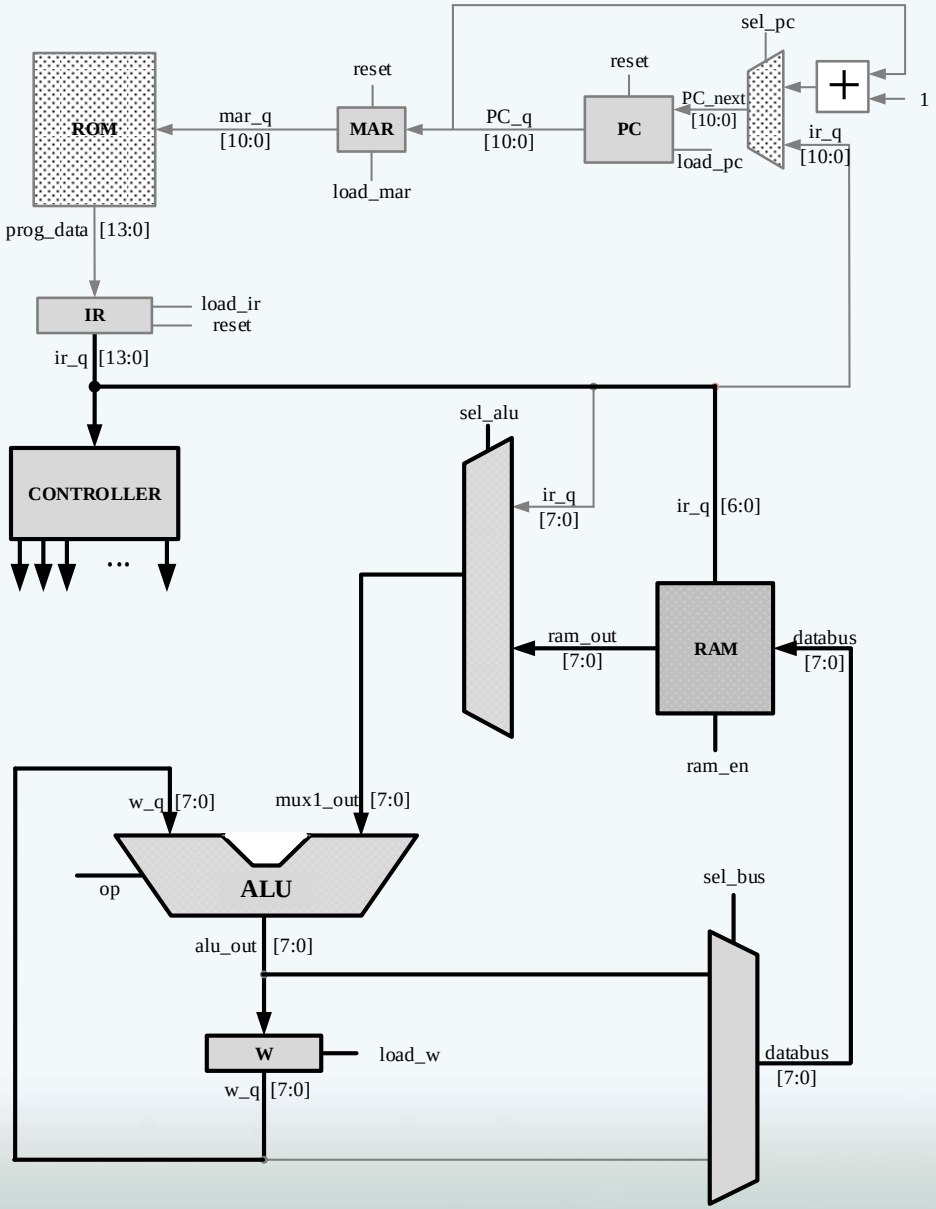
00 1010 dfff ffff



狀態	動作	控制訊號
T_4	**d = 0**：W← F + 1 **d = 1**：F ← F + 1	op=6 sel_alu = 1 **d = 0:** load_w=1 **d = 1:** ram_en=1 sel_bus=0
T_5	無動作	無
T_6	無動作	無

IORWF：W和F做邏輯或運算

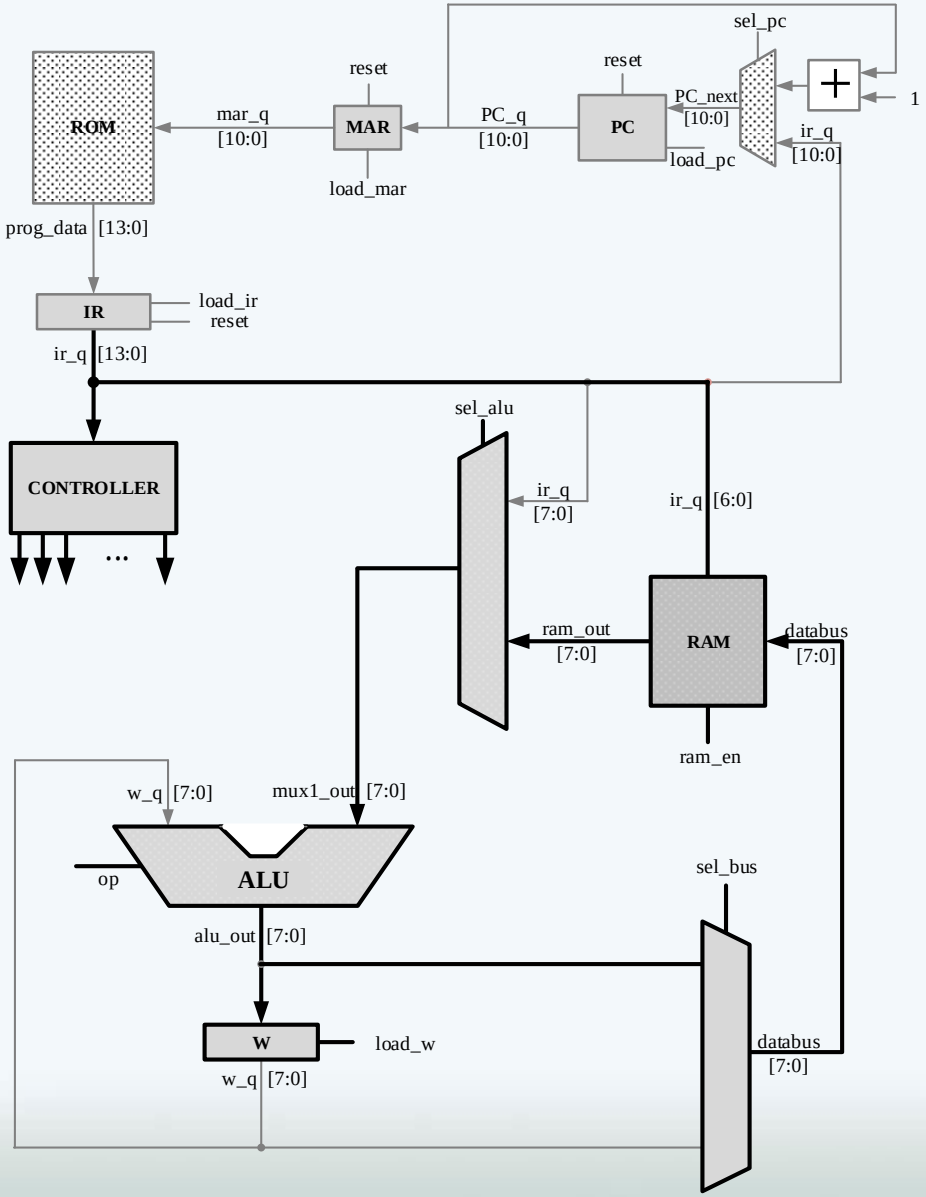
00 0100 dfff ffff



狀態	動作	控制訊號
T_4	**d = 0：** W← W | F **d = 1：** F ← W | F	op=3 sel_alu = 1 **d = 0：** load_w=1 **d = 1：** ram_en=1 sel_bus=0
T_5	無動作	無
T_6	無動作	無

MOVF：傳送F

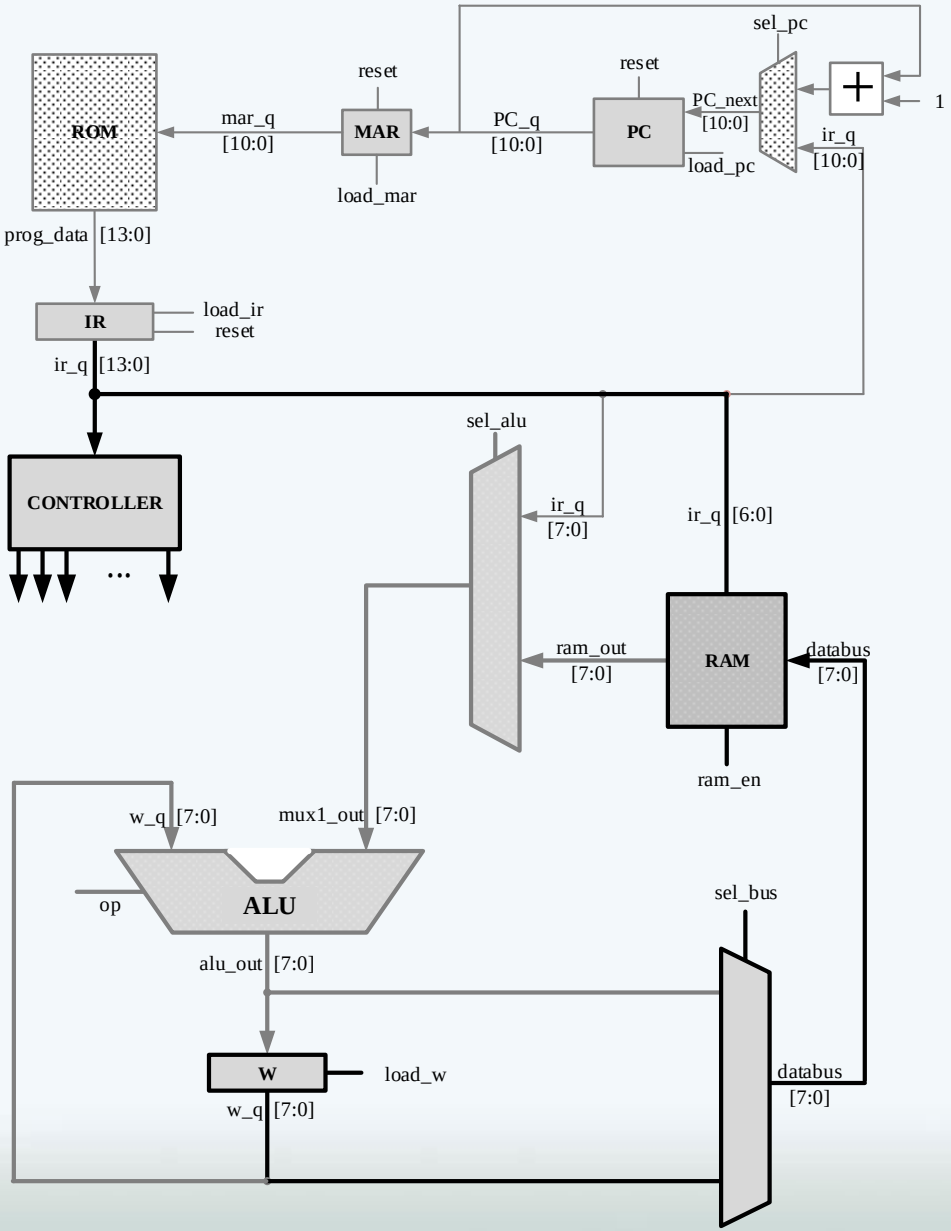
00 1000 dfff ffff



狀態	動作	控制訊號
T_4	**d = 0**：W← F **d = 1**：F ← F	op=5 sel_alu = 1 **d = 0**： load_w=1 **d = 1**： ram_en=1 sel_bus=0
T_5	無動作	無
T_6	無動作	無

MOVWF：將W的內容傳到F

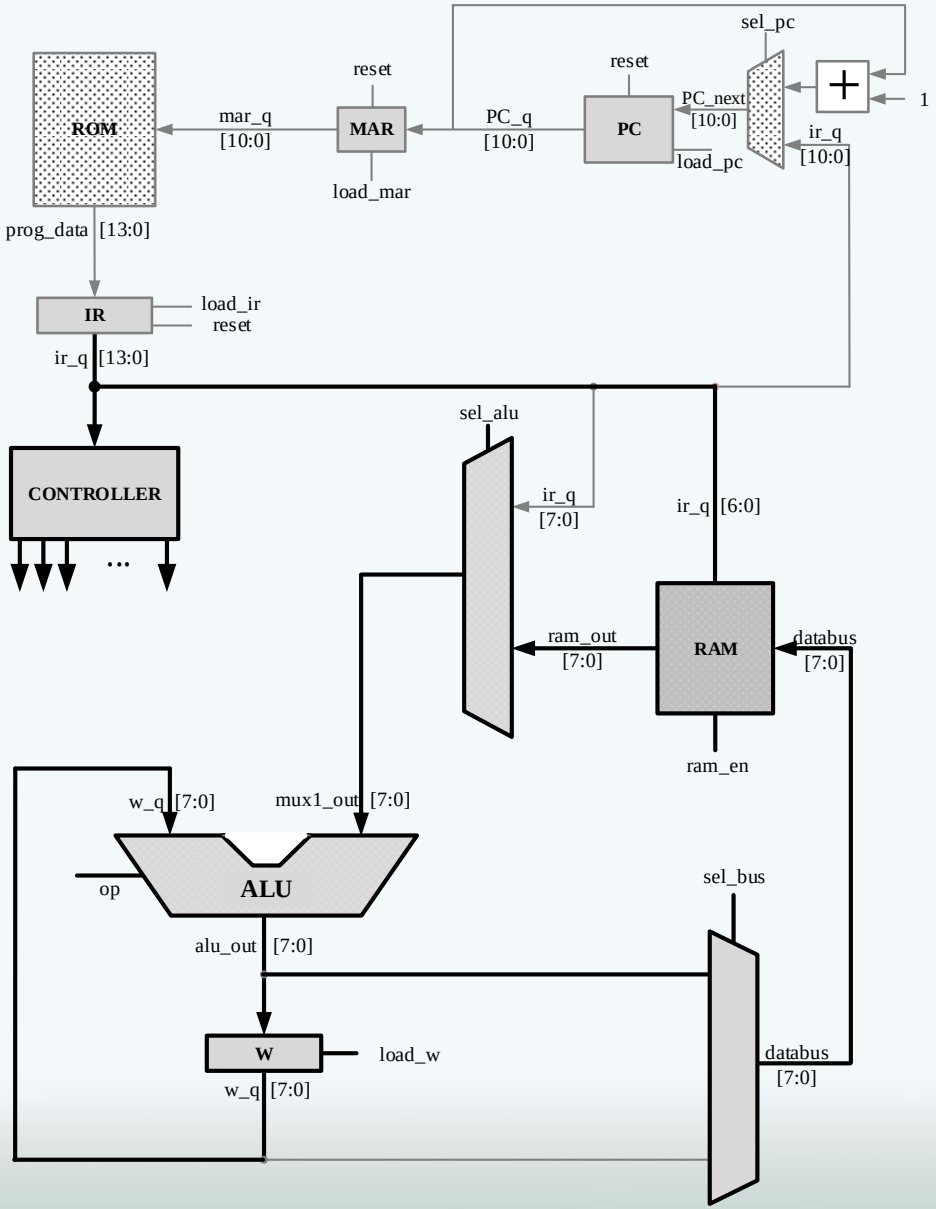
00 0000 1fff ffff



狀態	動作	控制訊號
T_4	F ← W	ram_en=1 sel_bus=1
T_5	無動作	無
T_6	無動作	無

SUBWF：F減去W

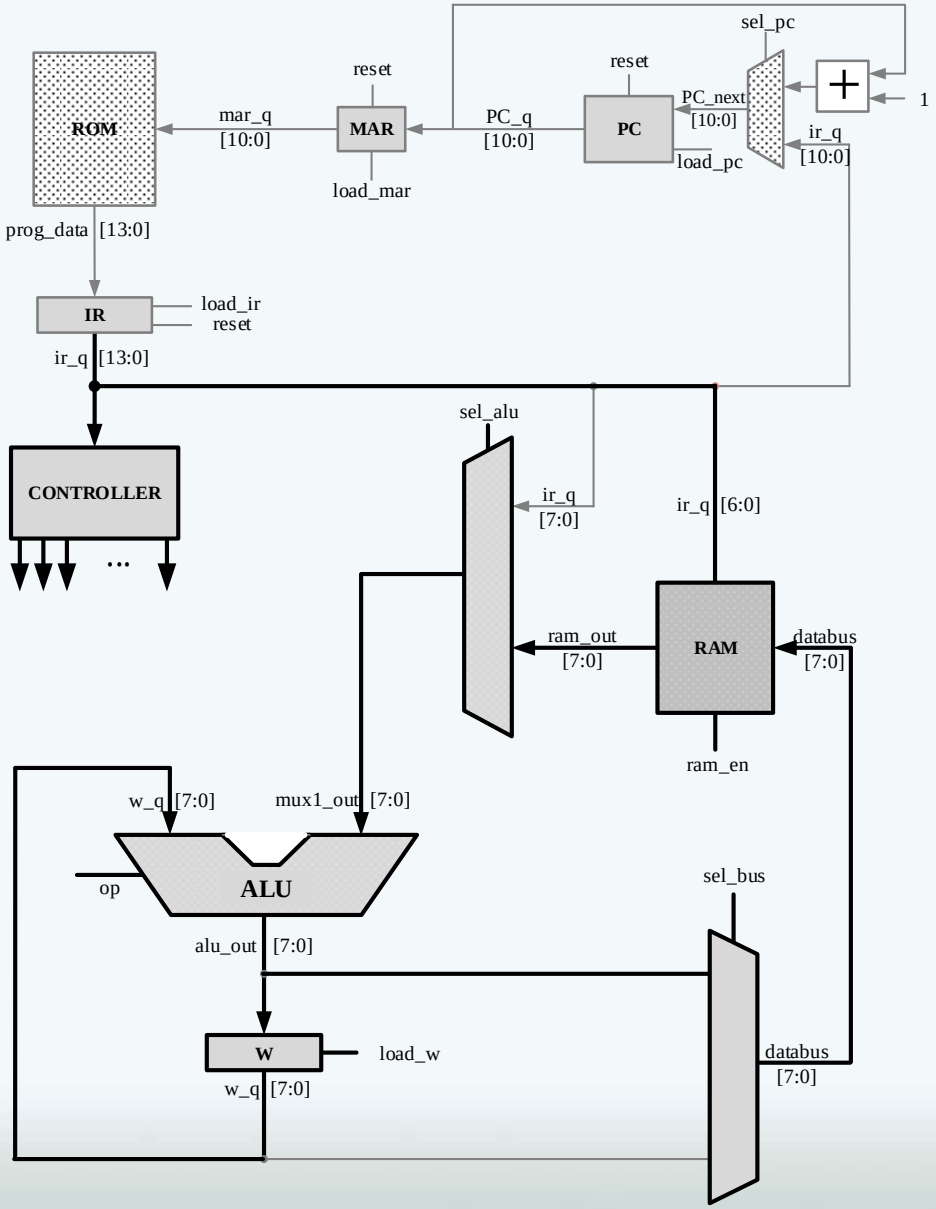
00 0010 dfff ffff



狀態	動作	控制訊號
T_4	**d = 0**：W← F - W **d = 1**：F ← F - W	op=1 sel_alu = 1 **d = 0**： load_w=1 **d = 1**： ram_en=1 sel_bus=0
T_5	無動作	無
T_6	無動作	無

XORWF：W和F做XOR運算

00 0110 dfff ffff



狀態	動作	控制訊號
T_4	**d = 0：** W← W ^ F **d = 1：** F ← W ^ F	op=4 sel_alu = 1 **d = 0：** load_w=1 **d = 1：** ram_en=1 sel_bus=0
T_5	無動作	無
T_6	無動作	無

回家作業

```
#include    <p16Lf1826.inc>     ; Include file locate at defult di
;

temp        equ 0x25
temp1       equ 0x24
;****************************************
;           Program start               *
;****************************************
            org     0x00        ; reset vector

            clrf  temp          ; //ram[25]<=0
            clrw                ; //w<=0
            movlw 7             ; //w<=7
            addwf temp,1        ; //ram[25]<=ram[25]+w  ram[25]=7
            movlw 5             ; //w<=5
            incf temp,1         ; //ram[25]=8
            iorwf temp,1        ; //ram[25]=D
            movwf temp1         ; //ram[24]=5
            subwf temp,0        ; //w<=8
            movf temp,0         ; //w<=D
            xorwf temp1,1       ; //ram[24]=8
            end
```

```
module Program_Rom(
    output logic [13:0] Rom_data_out,
    input [10:0] Rom_addr_in
);

    logic [13:0] data;
    always_comb
        begin
            case (Rom_addr_in)
                10'h0 : data = 14'h01A5;    //CLRF          ram[25] = 0
                10'h1 : data = 14'h0103;    //CLRW          W = 0
                10'h2 : data = 14'h3007;    //MOVLW 7       W = 7
                10'h3 : data = 14'h07A5;    //ADDWF 0x25,1  ram[25] = 7
                10'h4 : data = 14'h3005;    //MOVLW 5       W = 5
                10'h5 : data = 14'h0AA5;    // INCF 0x25,1  ram[25] = 8
                10'h6 : data = 14'h04A5;    //IORWF 0x25,1  ram[25] = D
                10'h7 : data = 14'h00A4;    //MOVWF 0x24    ram[24] = 5
                10'h8 : data = 14'h0225;    //SUBWF 0x25,0  W = 8
                10'h9 : data = 14'h0825;    // MOVF 0x25,0  W = D
                10'ha : data = 14'h06A4;    //XORWF 0x24,1  ram[24] = 8
                10'hb : data = 14'h3400;    //MPLAB清除暫存器的指令，不用管
                10'hc : data = 14'h3400;    //MPLAB清除暫存器的指令，不用管
                default: data = 14'h0;
            endcase
        end

    assign Rom_data_out = data;

endmodule
```

請將w_q、ram[36]、ram[37]以16進制表示