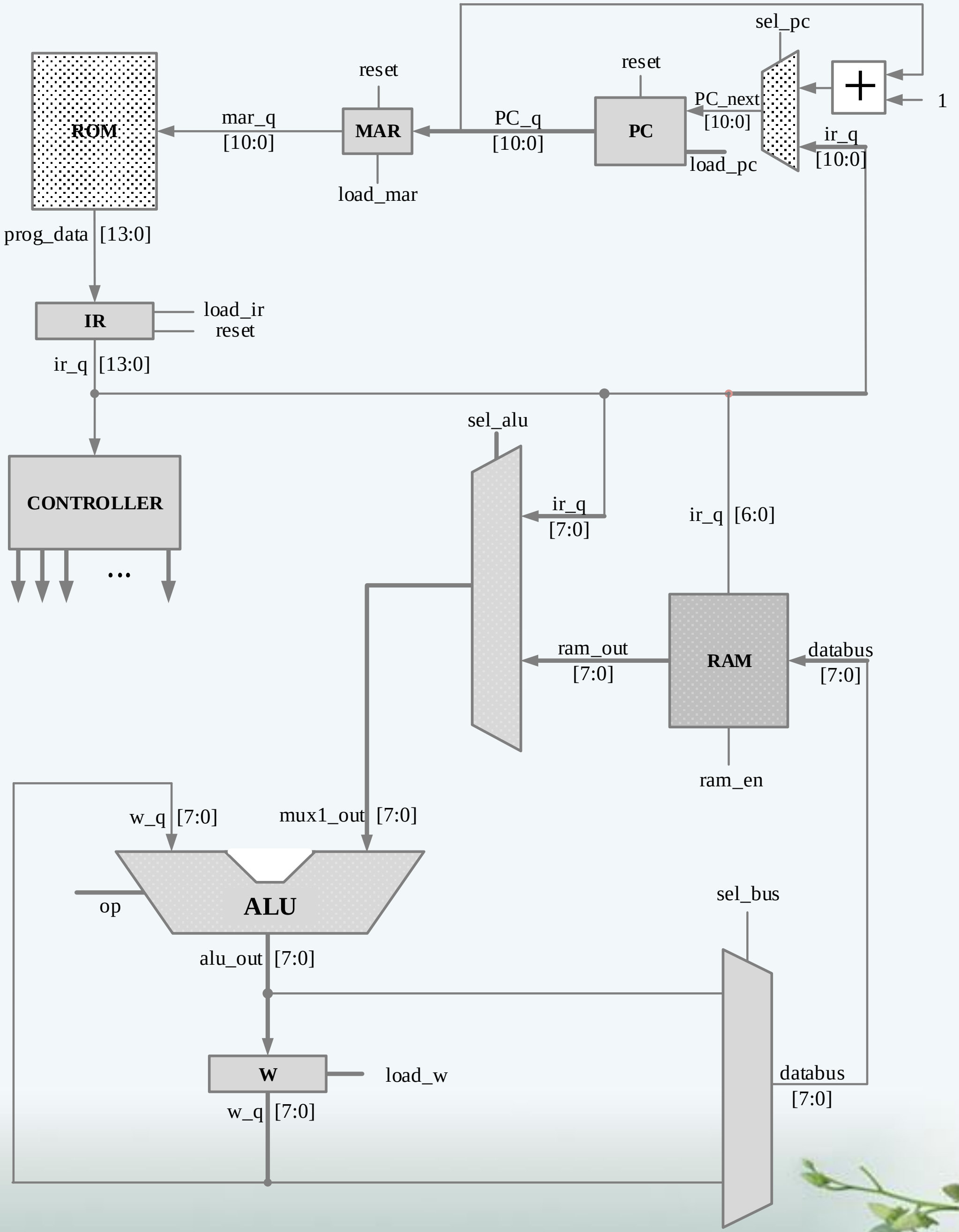
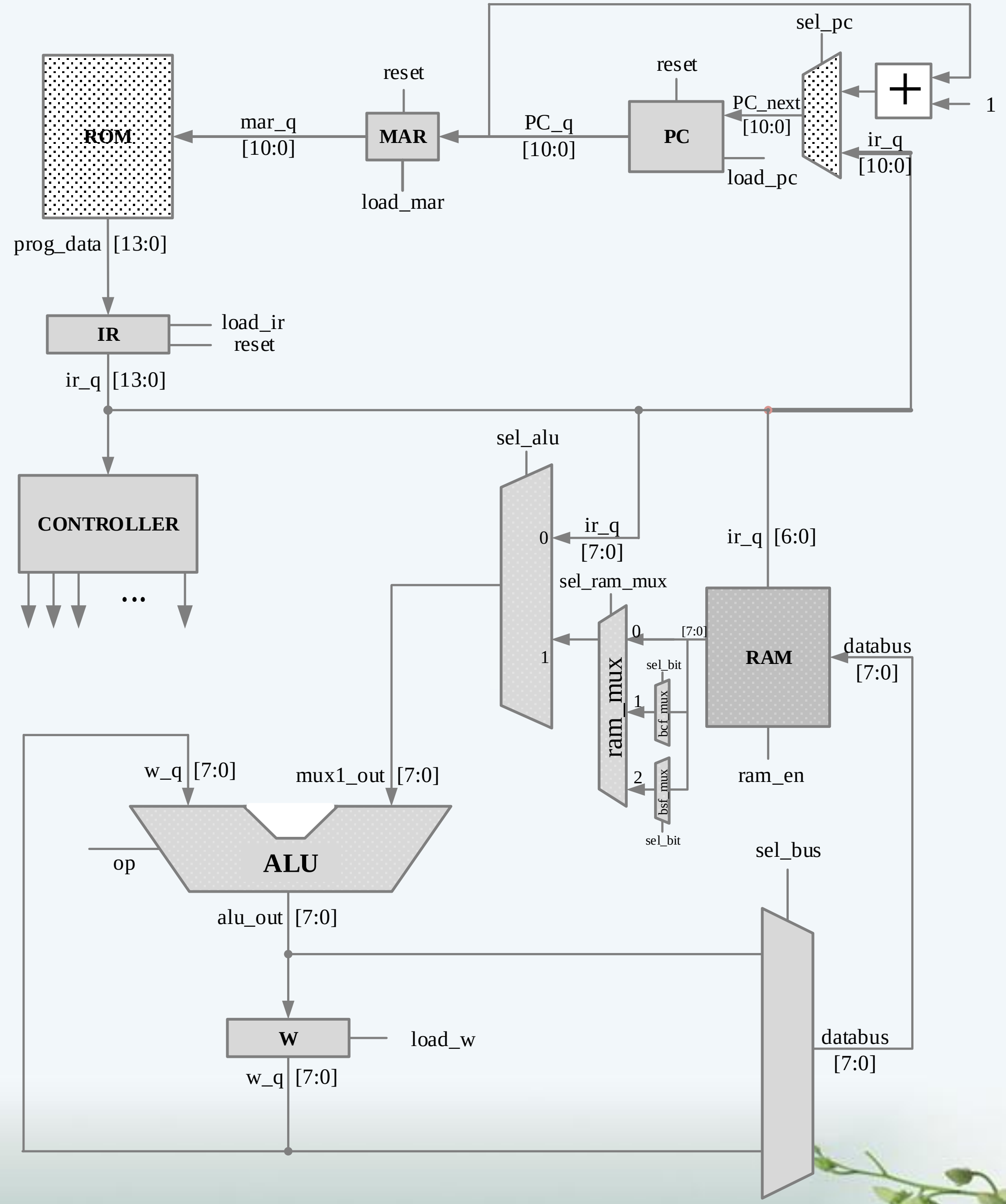
條件跳躍指令

舊架構圖


sel_pc
reset
reset
ROM
mar_q
[10:0]
MAR
PC_q
[10:0]
PC
PC_next
[10:0]
+
1
ir_q
[10:0]
load_pc
load_mar
prog_data
[13:0]
IR
load_ir
reset
ir_q
[13:0]
CONTROLLER
…
sel_alu
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]

新架構圖


ROM
mar_q
[10:0]
reset
MAR
load_mar
PC_q
[10:0]
reset
PC
load_pc
PC_next
[10:0]
sel_pc
+
1
ir_q
[10:0]
prog_data
[13:0]
IR
load_ir
reset
ir_q
[13:0]
CONTROLLER
…
sel_alu
0
1
ir_q
[7:0]
sel_ram_mux
ram_mux
0
[7:0]
1
2
sel_bit
bcf_mux
bsf_mux
sel_bit
ir_q
[6:0]
RAM
databus
[7:0]
ram_en
w_q
[7:0]
mux1_out
[7:0]
op
ALU
alu_out
[7:0]
W
load_w
w_q
[7:0]
sel_bus
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。

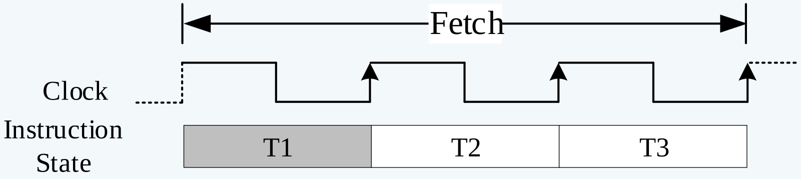
- 兩個時間週期：
  Byte-oriented File Register Operations、Bit-oriented File Register Operations、Bit-oriented Skip Operations。

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

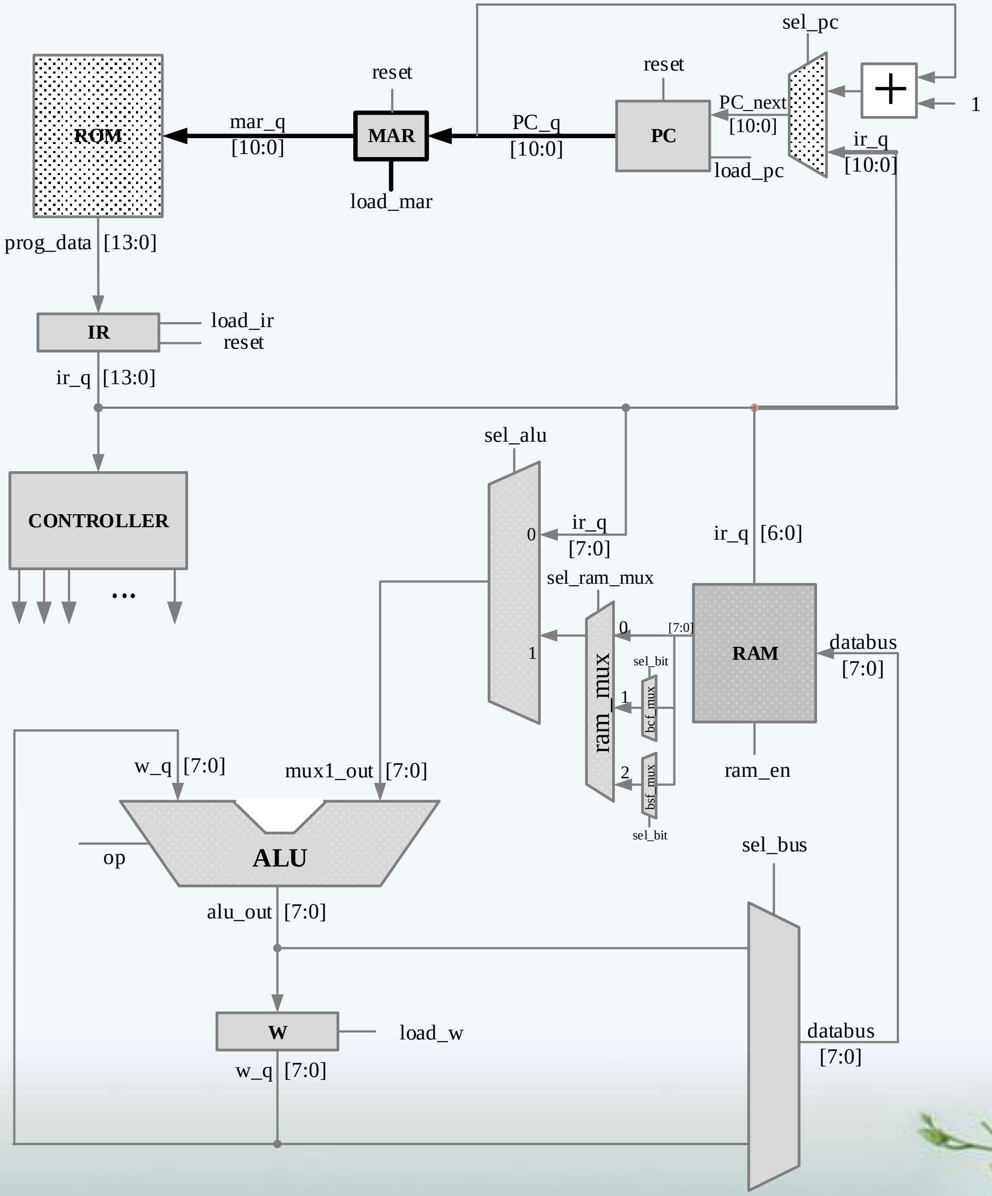
T_1、T_2及T_3擷取階段，控制訊號均相同，如下表所示。

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

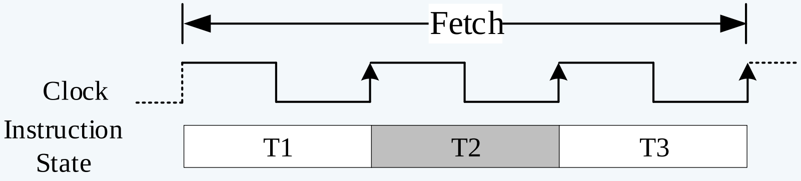
Fetch T1



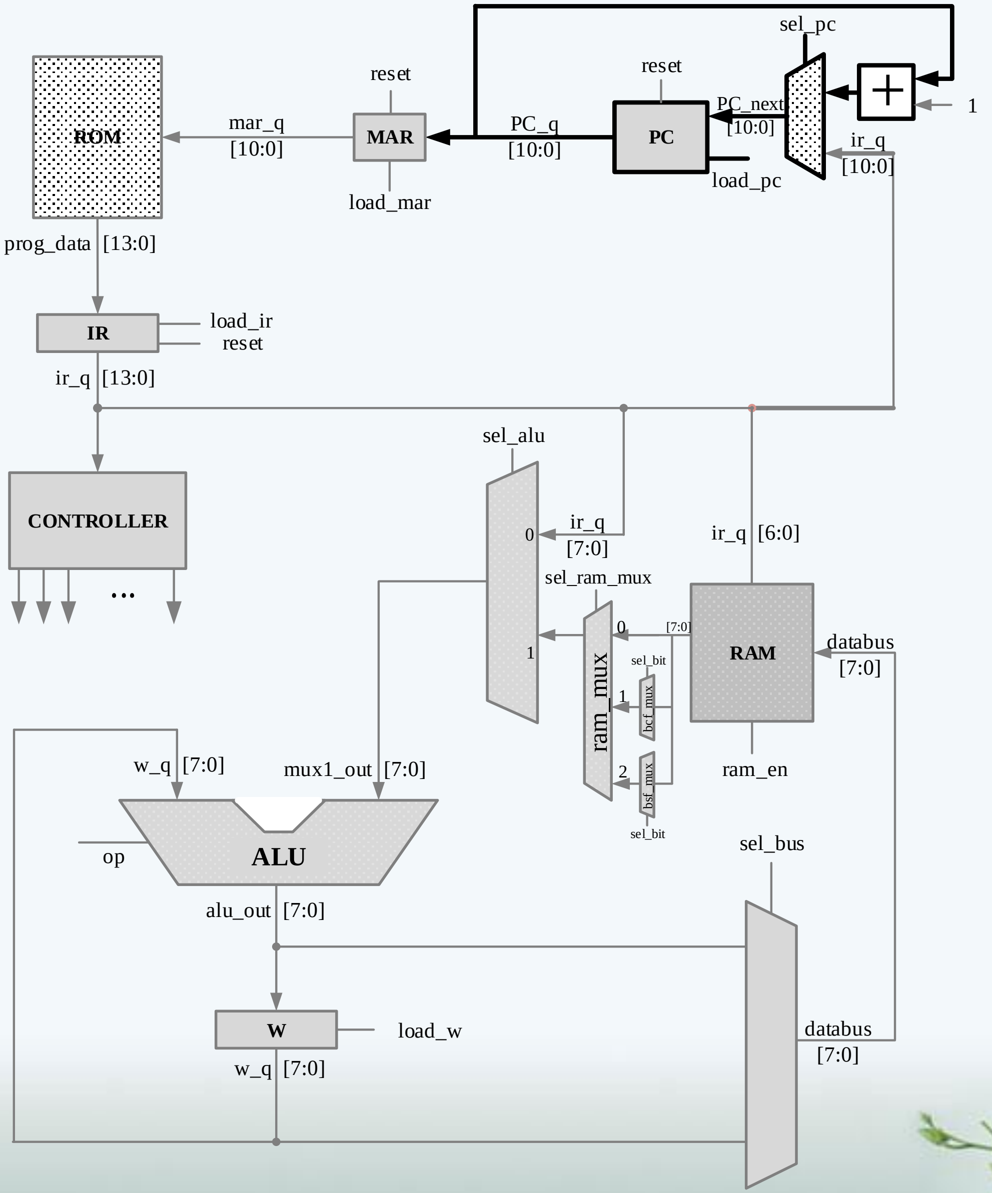
狀態	動作	控制訊號
T_1	MAR←PC	load_mar



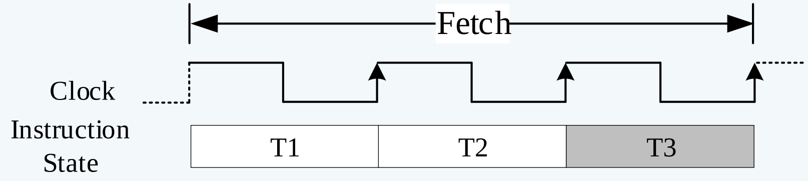
Fetch T2



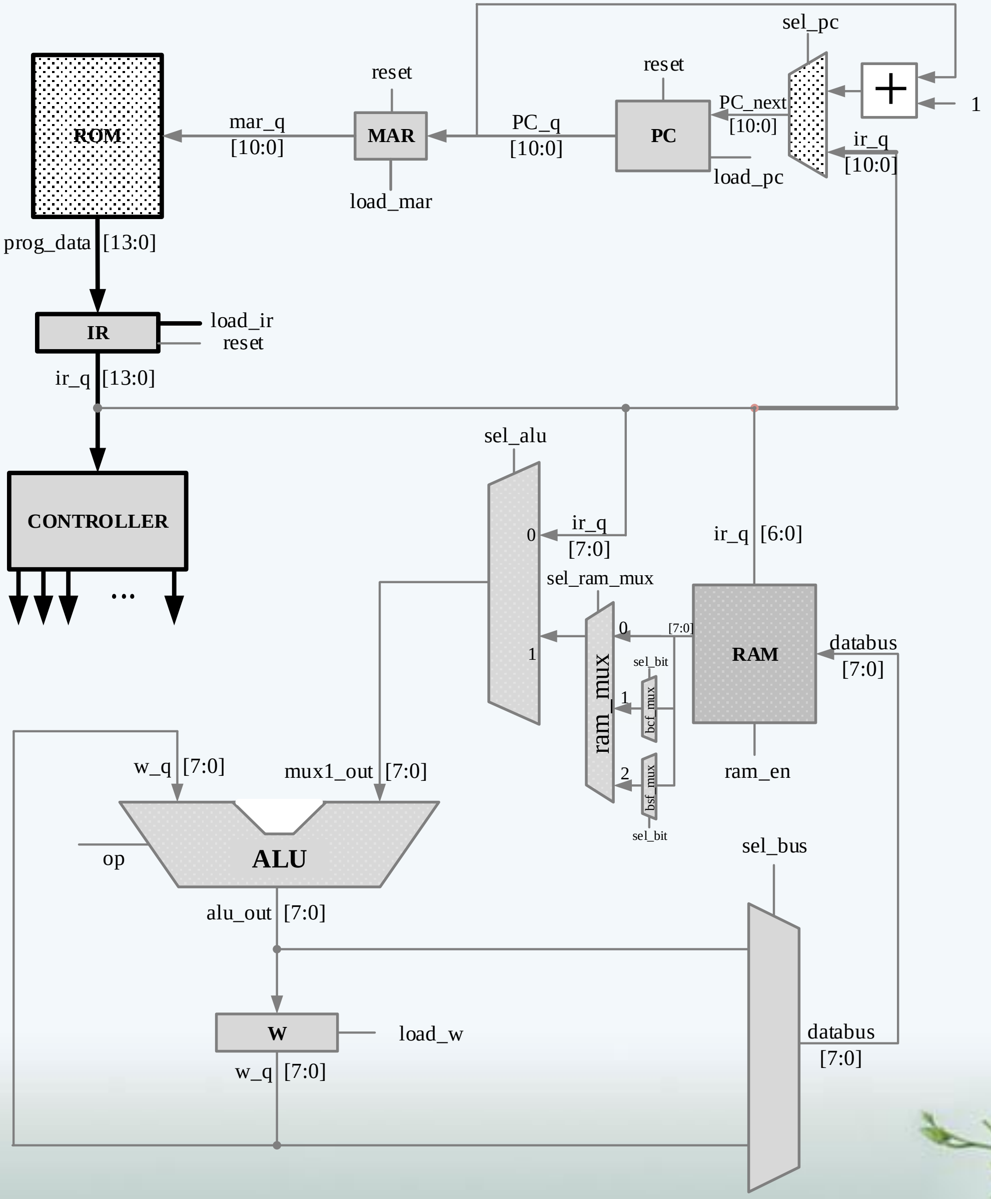
狀態	動作	控制訊號
T_2	PC←PC+1	load_pc



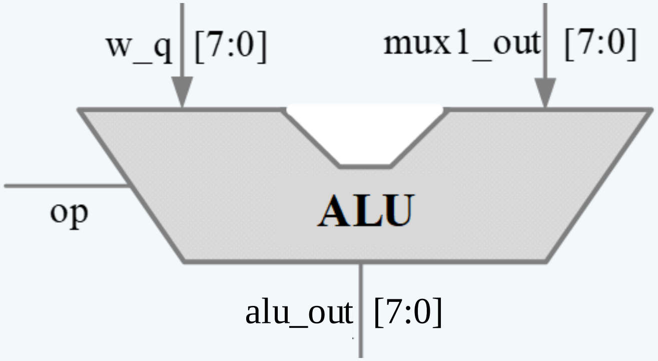
Fetch T3

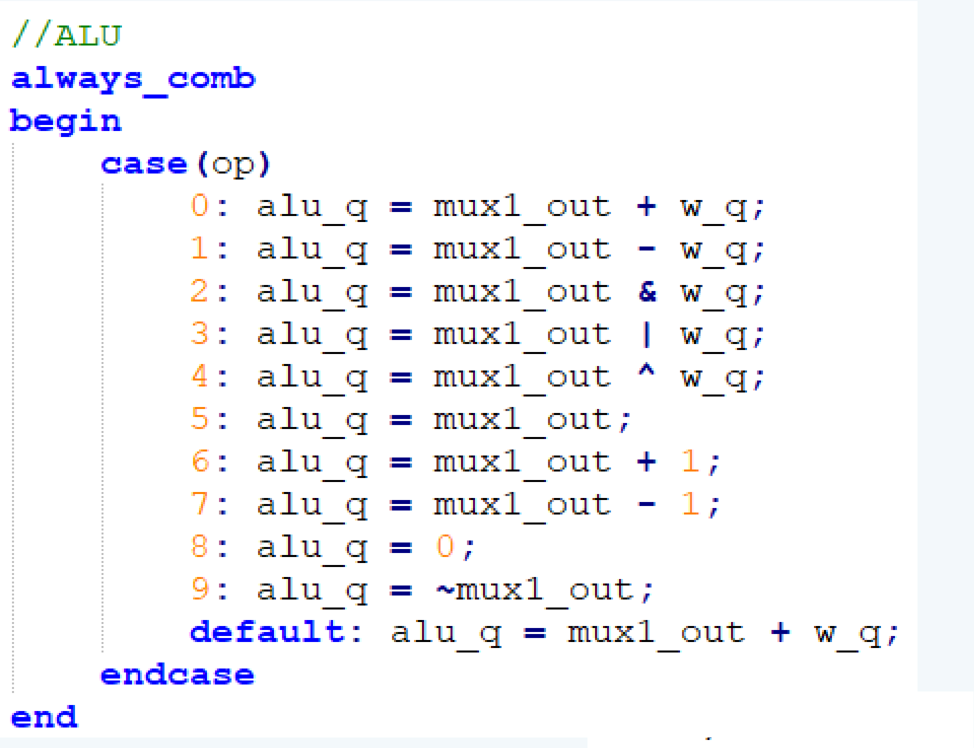


狀態	動作	控制訊號
T_3	IR←ROM[MAR]	load_ir



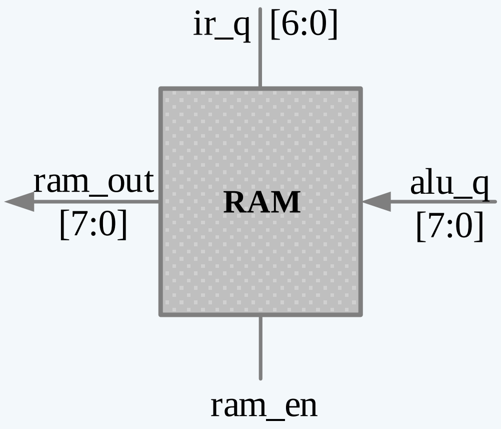
ALU

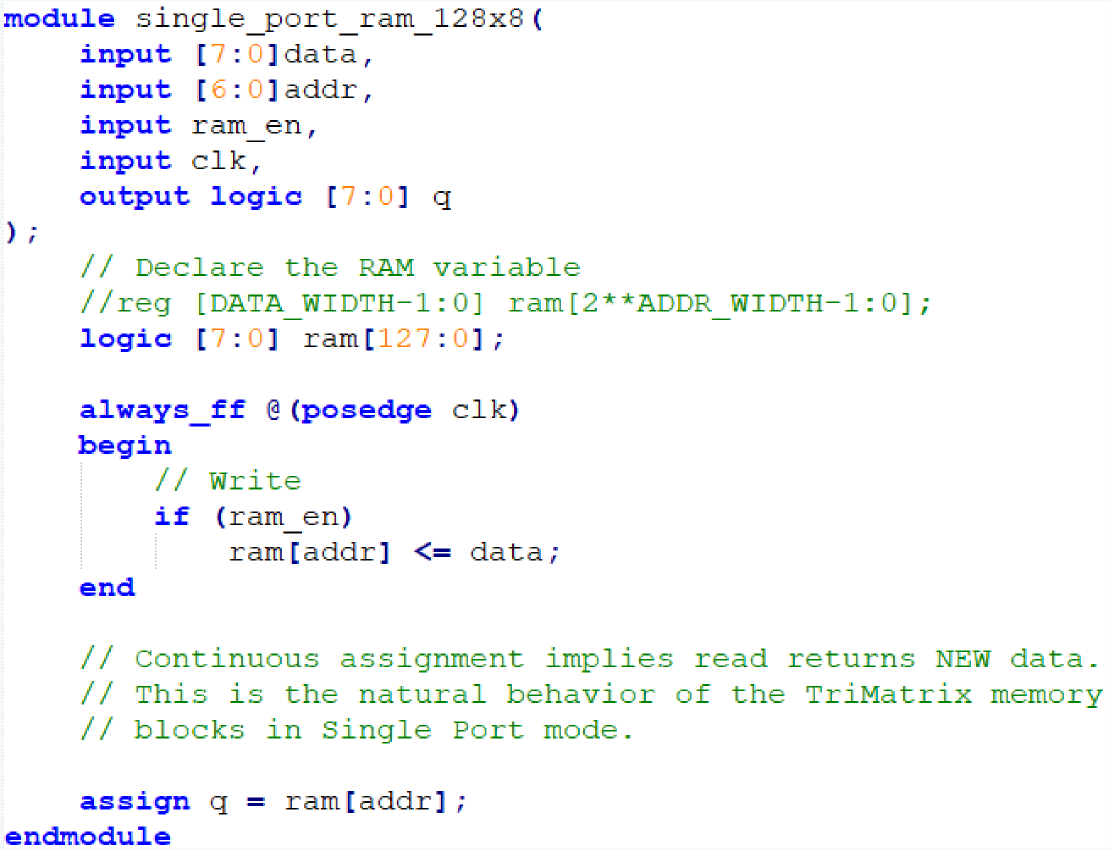




```
//ALU
always_comb
begin
    case(op)
        0: alu_q = mux1_out + w_q;
        1: alu_q = mux1_out - w_q;
        2: alu_q = mux1_out & w_q;
        3: alu_q = mux1_out | w_q;
        4: alu_q = mux1_out ^ w_q;
        5: alu_q = mux1_out;
        6: alu_q = mux1_out + 1;
        7: alu_q = mux1_out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

SRAM





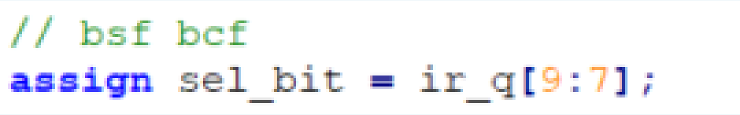
```
module single_port_ram_128x8(
    input [7:0]data,
    input [6:0]addr,
    input ram_en,
    input clk,
    output logic [7:0] q
);
    // Declare the RAM variable
    //reg [DATA_WIDTH-1:0] ram[2**ADDR_WIDTH-1:0];
    logic [7:0] ram[127:0];

    always_ff @(posedge clk)
    begin
        // Write
        if (ram_en)
            ram[addr] <= data;
    end

    // Continuous assignment implies read returns NEW data.
    // This is the natural behavior of the TriMatrix memory
    // blocks in Single Port mode.

    assign q = ram[addr];
endmodule
```

BIT-ORIENTED FILE REGISTER OPERATIONS

PIC16F1826 INSTRUCTION SET

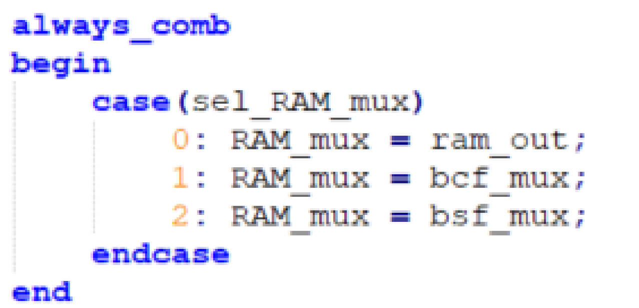
Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes
			MSB			LSB		
BIT-ORIENTED FILE REGISTER OPERATIONS								
BCF f, b	Bit Clear f	1	01	00bb	bfff	ffff		2
BSF f, b	Bit Set f	1	01	01bb	bfff	ffff		2

sel_bit

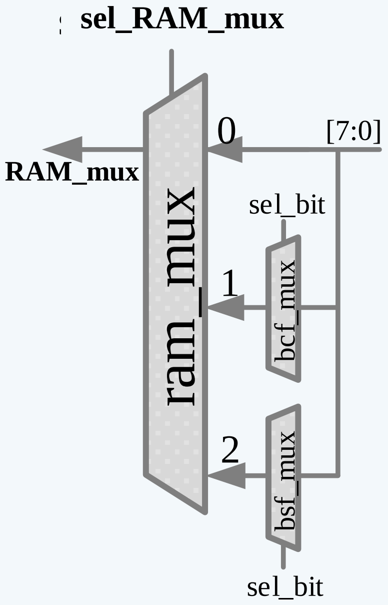


```
// bsf bcf
assign sel_bit = ir_q[9:7];
```

RAM_mux



```
always_comb
begin
    case(sel_RAM_mux)
        0: RAM_mux = ram_out;
        1: RAM_mux = bcf_mux;
        2: RAM_mux = bsf_mux;
    endcase
end
```



Note

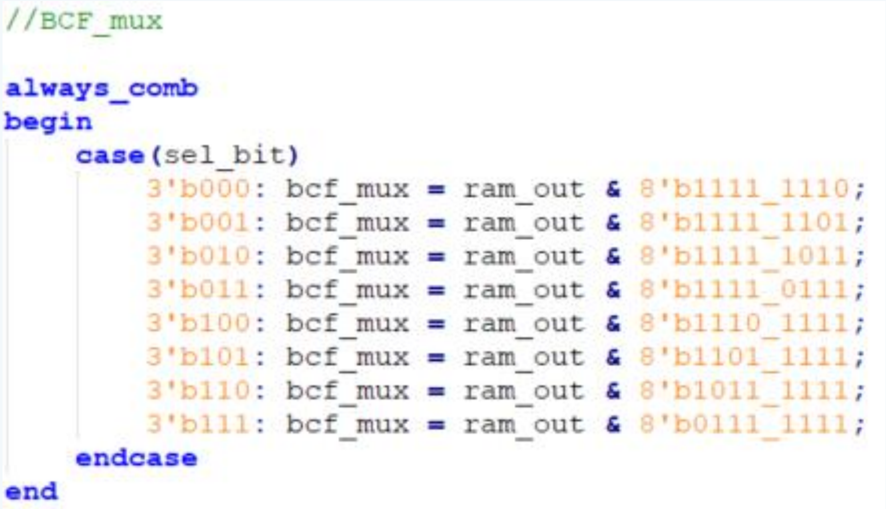
1：If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

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.

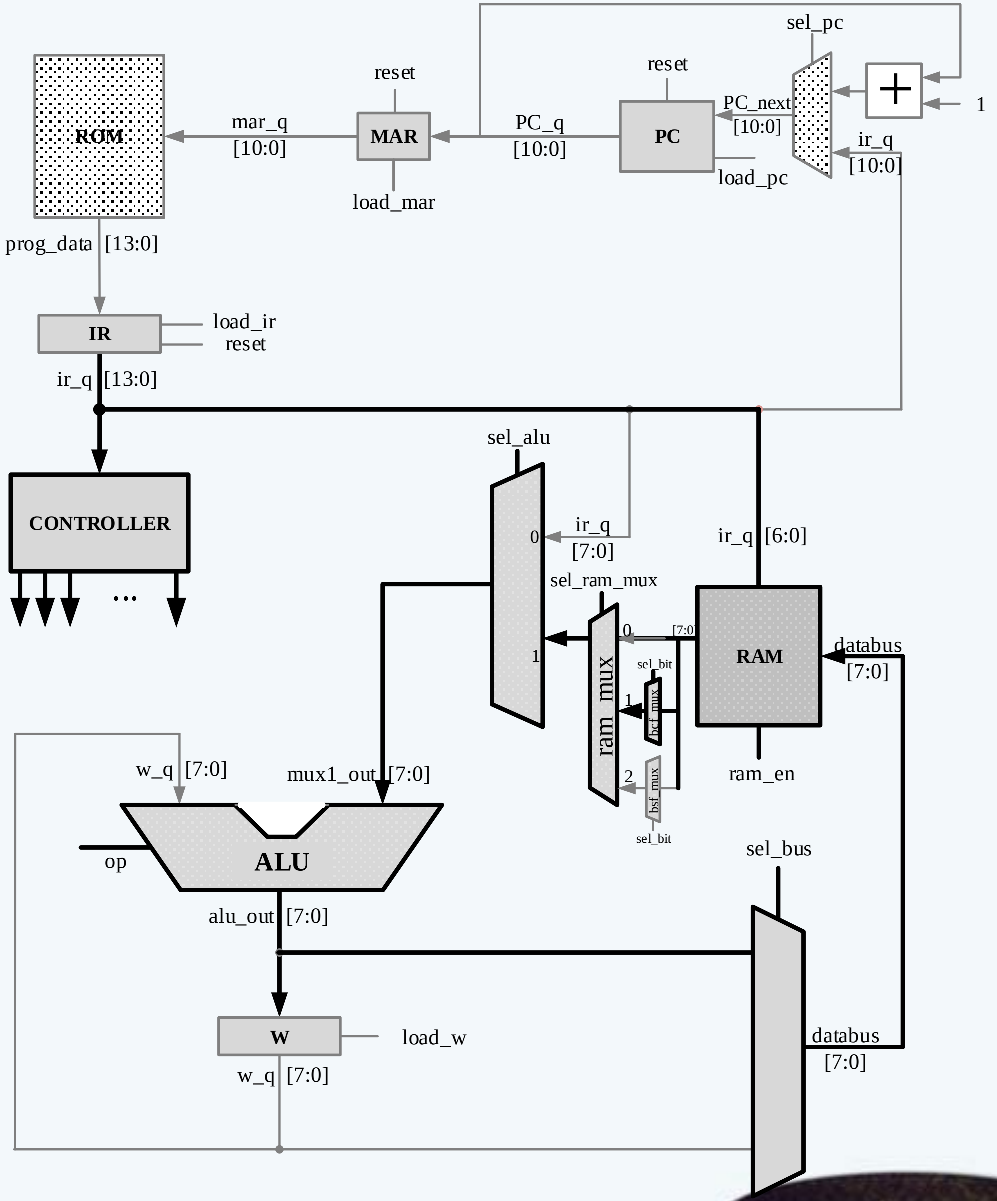
BCF T4

狀態	動作	控制訊號

T_4	addr = ir[6:0]	sel_alu = 1
		sel_RAM_mux = 1
		op[3:0] = 5
		sel_bus = 0
		ram_en = 1



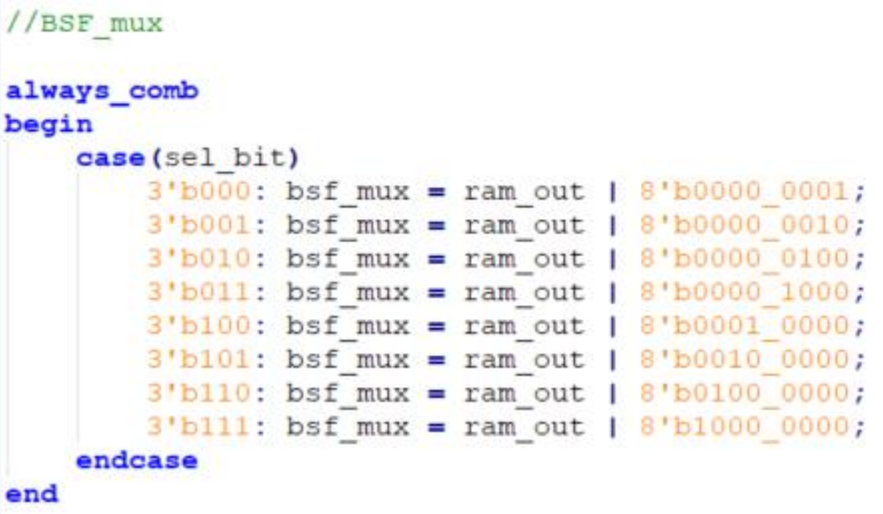
```
//BCF_mux

always_comb
begin
    case(sel_bit)
        3'b000: bcf_mux = ram_out & 8'b1111_1110;
        3'b001: bcf_mux = ram_out & 8'b1111_1101;
        3'b010: bcf_mux = ram_out & 8'b1111_1011;
        3'b011: bcf_mux = ram_out & 8'b1111_0111;
        3'b100: bcf_mux = ram_out & 8'b1110_1111;
        3'b101: bcf_mux = ram_out & 8'b1101_1111;
        3'b110: bcf_mux = ram_out & 8'b1011_1111;
        3'b111: bcf_mux = ram_out & 8'b0111_1111;
    endcase
end
```



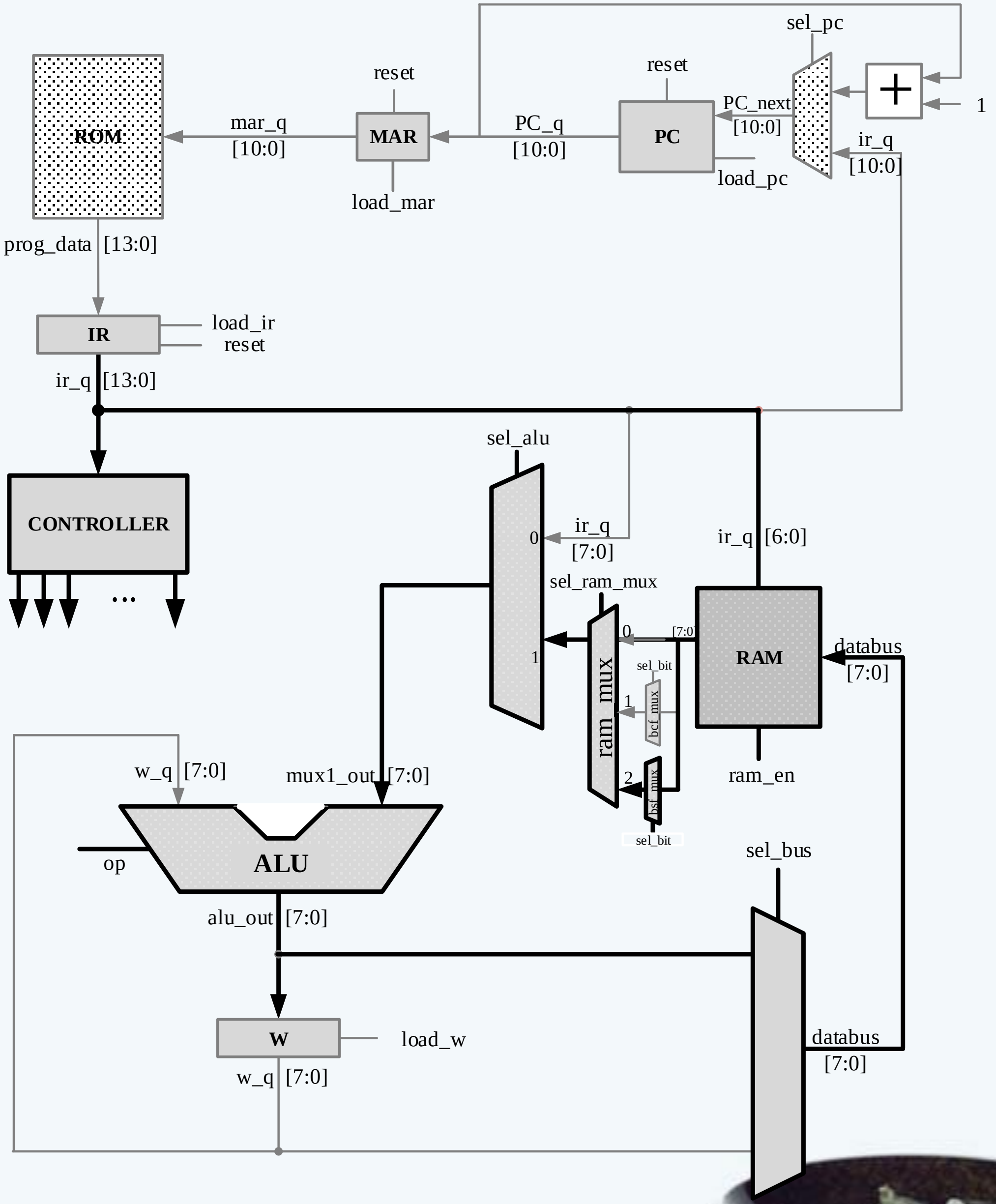
BSF T4

狀態	動作	控制訊號

T_4	addr = ir[6:0]	sel_alu = 1
		sel_RAM_mux = 2
		op[3:0] = 5
		sel_bus = 0
		ram_en = 1



```
//BSF_mux

always_comb
begin
    case(sel_bit)
        3'b000: bsf_mux = ram_out | 8'b0000_0001;
        3'b001: bsf_mux = ram_out | 8'b0000_0010;
        3'b010: bsf_mux = ram_out | 8'b0000_0100;
        3'b011: bsf_mux = ram_out | 8'b0000_1000;
        3'b100: bsf_mux = ram_out | 8'b0001_0000;
        3'b101: bsf_mux = ram_out | 8'b0010_0000;
        3'b110: bsf_mux = ram_out | 8'b0100_0000;
        3'b111: bsf_mux = ram_out | 8'b1000_0000;
    endcase
end
```



PIC16F1826 INSTRUCTION SET

Mnemonic, Operands	**Description**	**Cycles**	**14-Bit Opcode**				**Status Affected**	**Notes**
			MSB			LSB		
BIT-ORIENTED SKIP OPERATIONS								
BTFSC f, b	Bit Test f, Skip if Clear	1(2)	01	10bb	bfff	ffff		1,2
BTFSS f, b	Bit Test f, Skip if Set	1(2)	01	11bb	bfff	ffff		1,2

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.

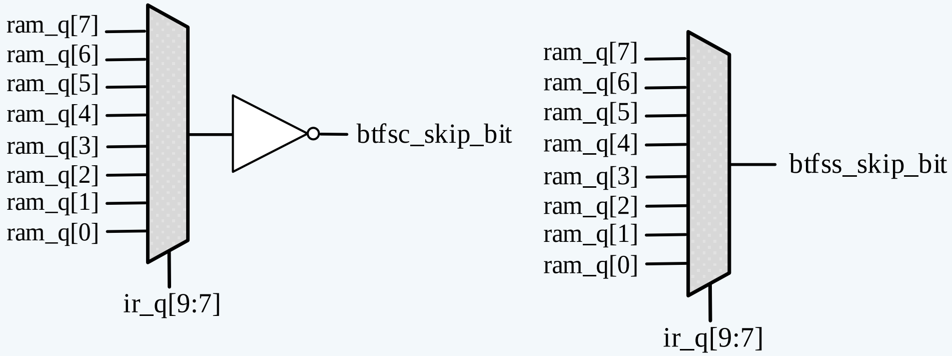
2：If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

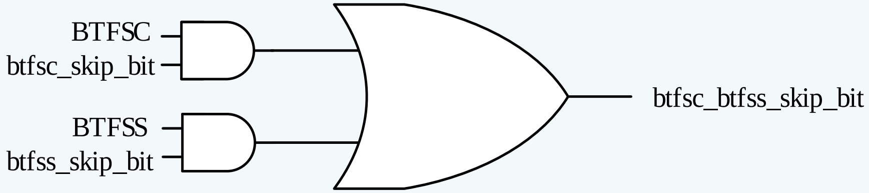
BTFSC & BTFSS

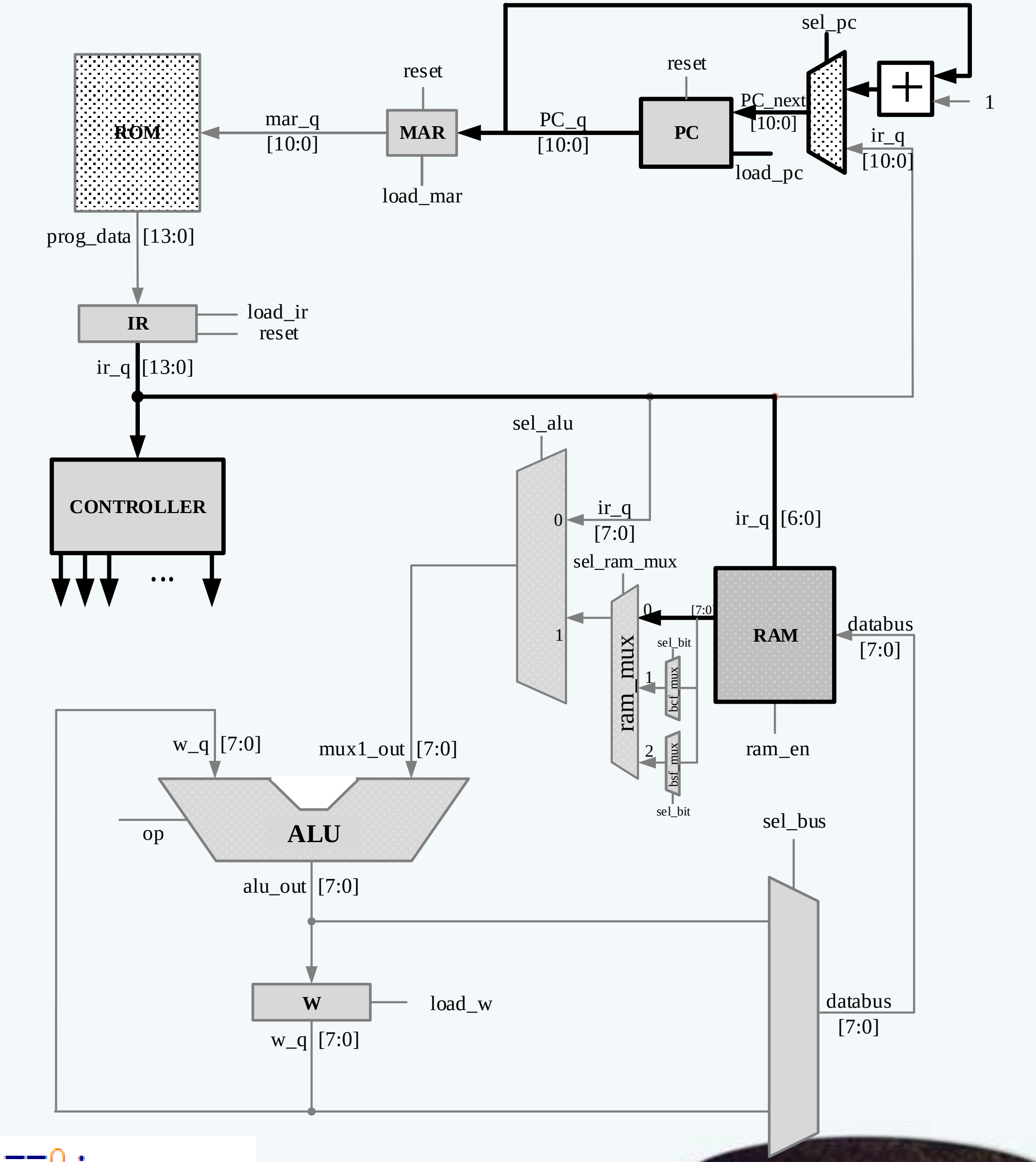
T4

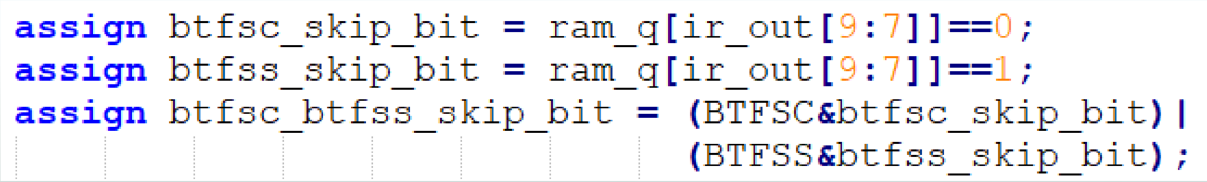
狀態	動作	控制訊號

T_4	addr = ir[6:0]	if **btfsc_btfss_skip_bit** == 1 :
		load_pc = 1
		sel_pc = 0









```
assign btfsc_skip_bit = ram_q[ir_out[9:7]]==0;
assign btfss_skip_bit = ram_q[ir_out[9:7]]==1;
assign btfsc_btfss_skip_bit = (BTFSC&btfsc_skip_bit)|
                              (BTFSS&btfss_skip_bit);
```

BYTE-ORIENTED SKIP OPERATIONS

PIC16F1826 INSTRUCTION SET

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes
			MSB			LSB		
BYTE ORIENTED SKIP OPERATIONS								
DECFSZ f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2
INCFSZ f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2

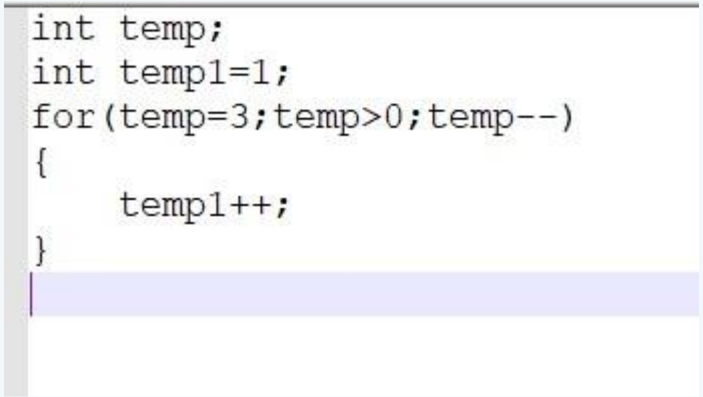
Note

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.

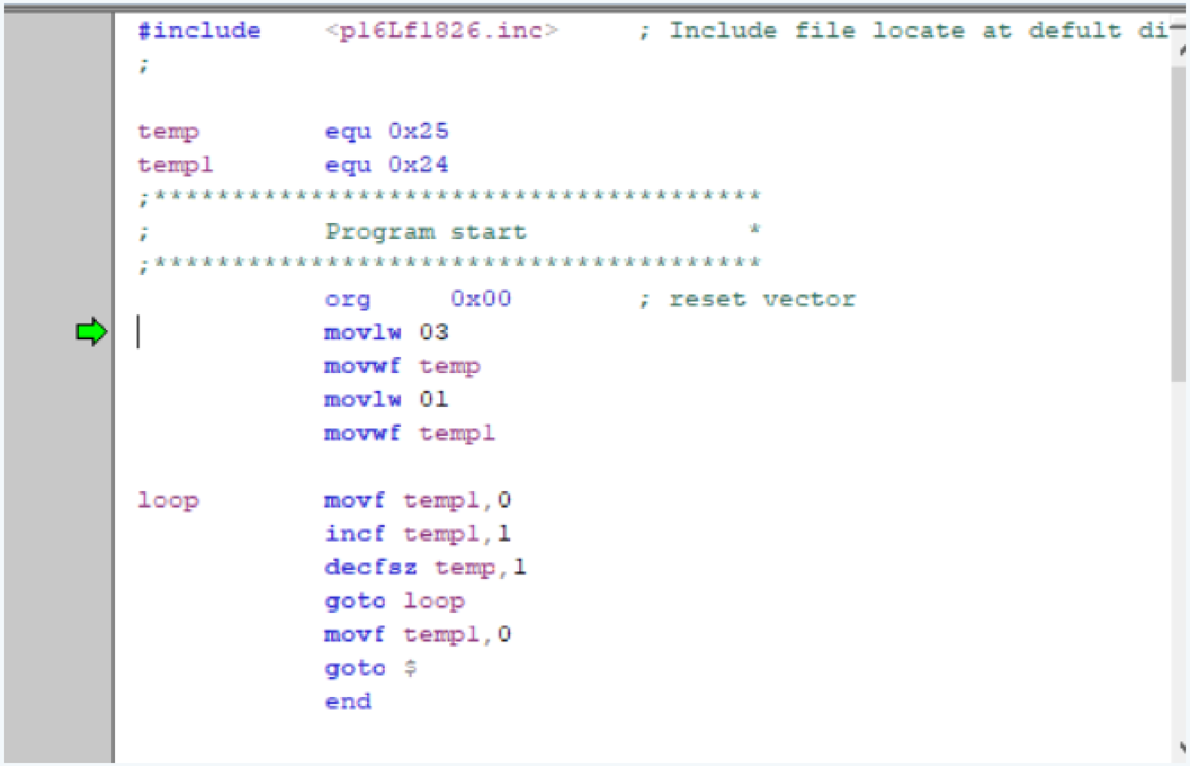
範例1

C語言

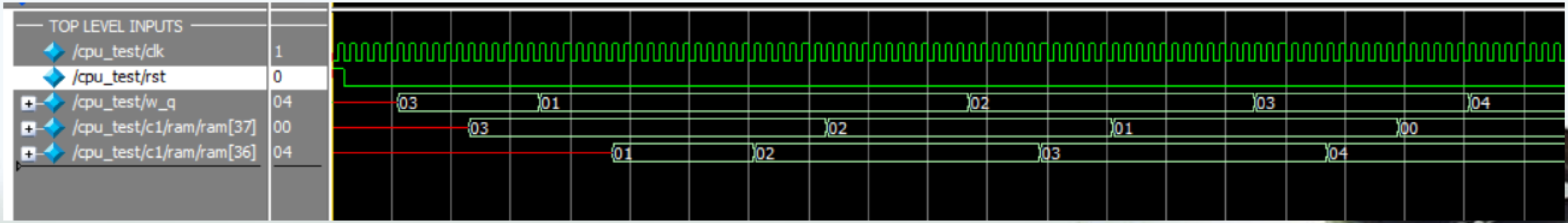


```
int temp;
int temp1=1;
for(temp=3;temp>0;temp--)
{
    temp1++;
}
```

組合語言



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

temp         equ 0x25
temp1        equ 0x24
;***************************************
;            Program start            *
;***************************************
             org     0x00        ; reset vector
             movlw 03
             movwf temp
             movlw 01
             movwf temp1

loop         movf temp1,0
             incf temp1,1
             decfsz temp,1
             goto loop
             movf temp1,0
             goto $
             end
```

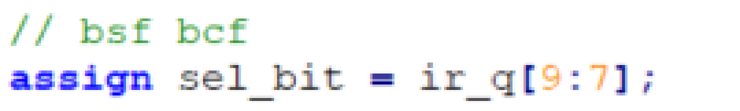


PIC16F1826 INSTRUCTION SET

Mnemonic, Operands	Description	Cycles	14-Bit Opcode MSB			LSB	Status Affected	Notes
BYTE ORIENTED SKIP OPERATIONS								
DECFSZ f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2
INCFSZ f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2

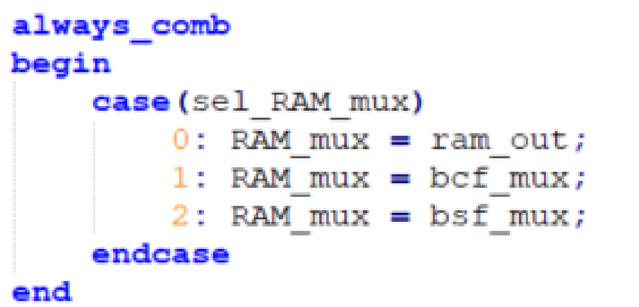
（此訊號為本章後面的指令所使用）

sel_bit

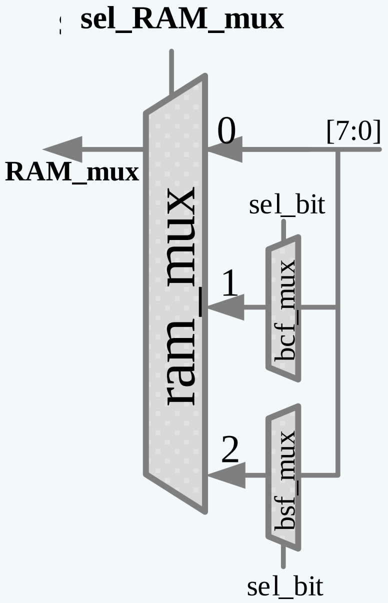


```
// bsf bcf
assign sel_bit = ir_q[9:7];
```

RAM_mux



```
always_comb
begin
    case(sel_RAM_mux)
        0: RAM_mux = ram_out;
        1: RAM_mux = bcf_mux;
        2: RAM_mux = bsf_mux;
    endcase
end
```



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.

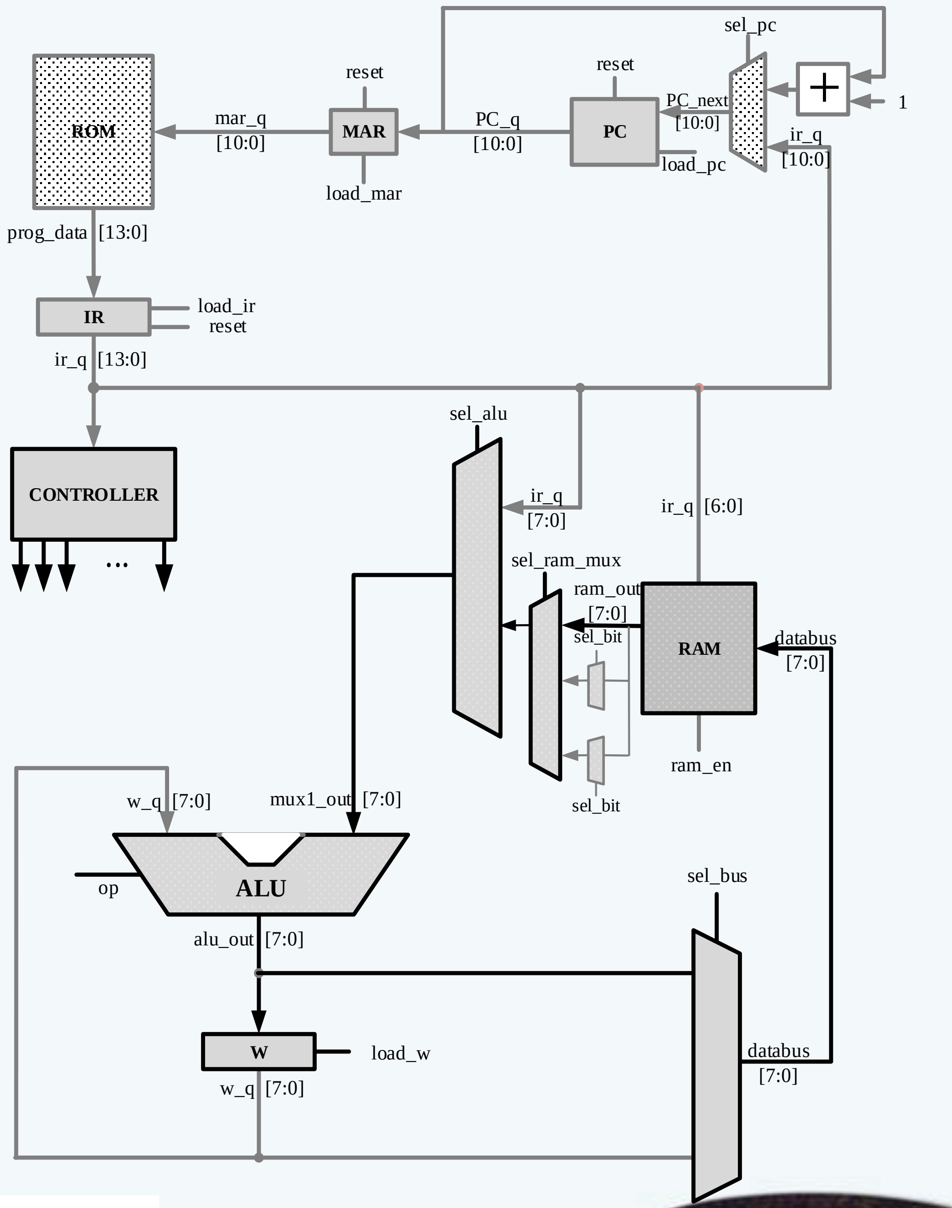
DECFSZ T4

狀態	動作	控制訊號

T_4
d = 0
	addr = ir[6:0]
	W <= RAM[addr] -1
	
	if **alu_zero** == 1 :
	PC_q <= PC_q + 1
		sel_alu = 1
		op[3:0] = 7
		load_w = 1
		if **alu_zero** == 1 :
		load_pc = 1
		sel_pc=0

狀態	動作	控制訊號

T_4
d = 1
	addr = ir[6:0]
	RAM[addr] <= RAM[addr] -1
	
	if **alu_zero** == 1 :
	PC_q <= PC_q + 1
		sel_alu = 1
		op[3:0] = 7
		ram_en = 1
		sel_bus = 0
		if **alu_zero** == 1 :
		load_pc = 1
		sel_pc=0



```
assign aluout_zero = (aluout == 0)? 1'b1: 1'b0;
```

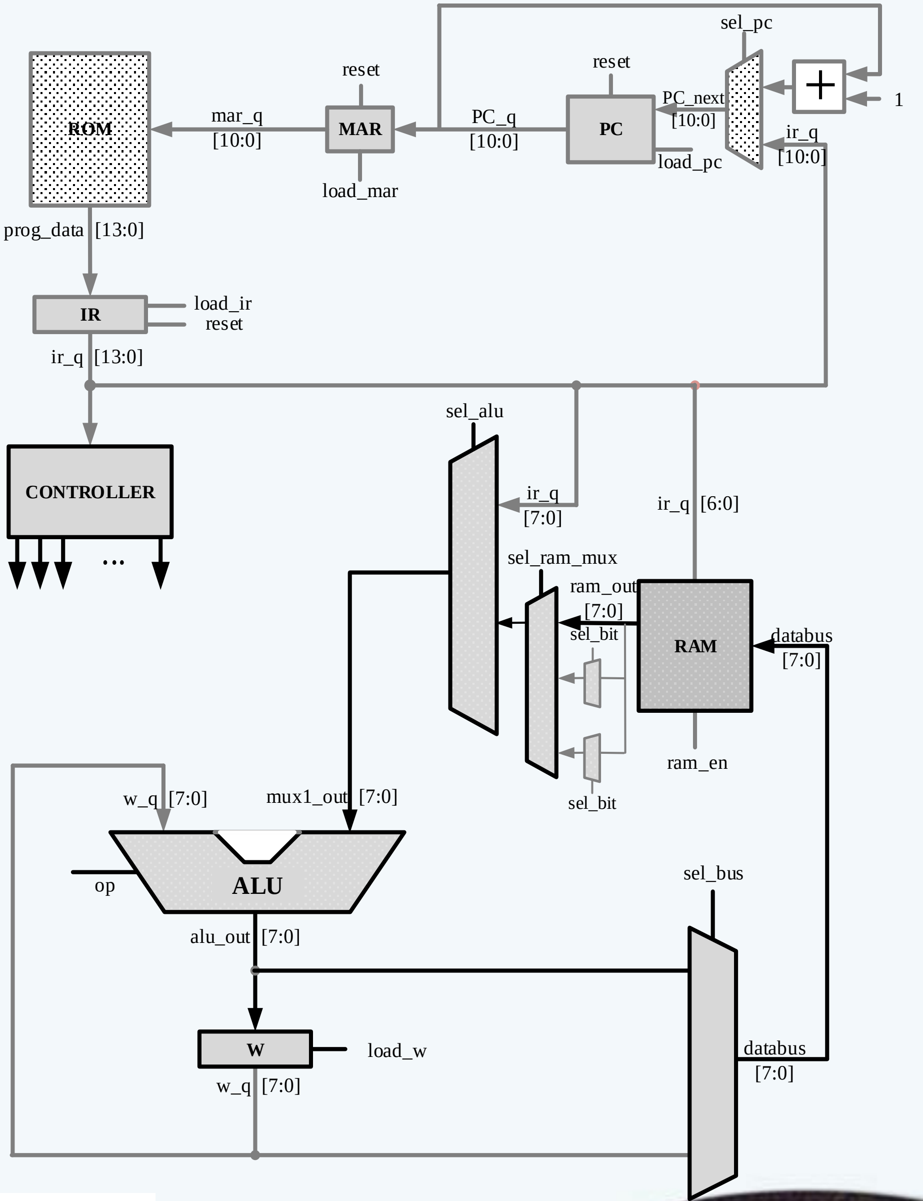
INCFSZ T4

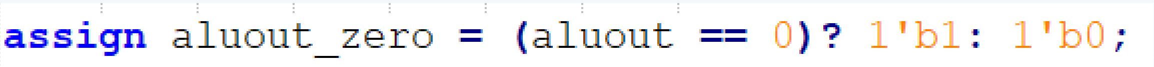
狀態	動作	控制訊號

T_4
d = 0
	addr = ir[6:0]
	W <= RAM[addr]+1
	
	if **alu_zero** == 1 :
	PC_q <= PC_q + 1
		sel_alu = 1
		op[3:0] = 6
		load_w = 1
		if **alu_zero** == 1 :
		load_pc = 1
		sel_pc=0

狀態	動作	控制訊號

T_4
d = 1
	addr = ir[6:0]
	RAM[addr] <= RAM[addr]+1
	
	if **alu_zero** == 1 :
	PC_q <= PC_q + 1
		sel_alu = 1
		op[3:0] = 6
		ram_en = 1
		sel_bus = 0
		if **alu_zero** == 1 :
		load_pc = 1
		sel_pc=0

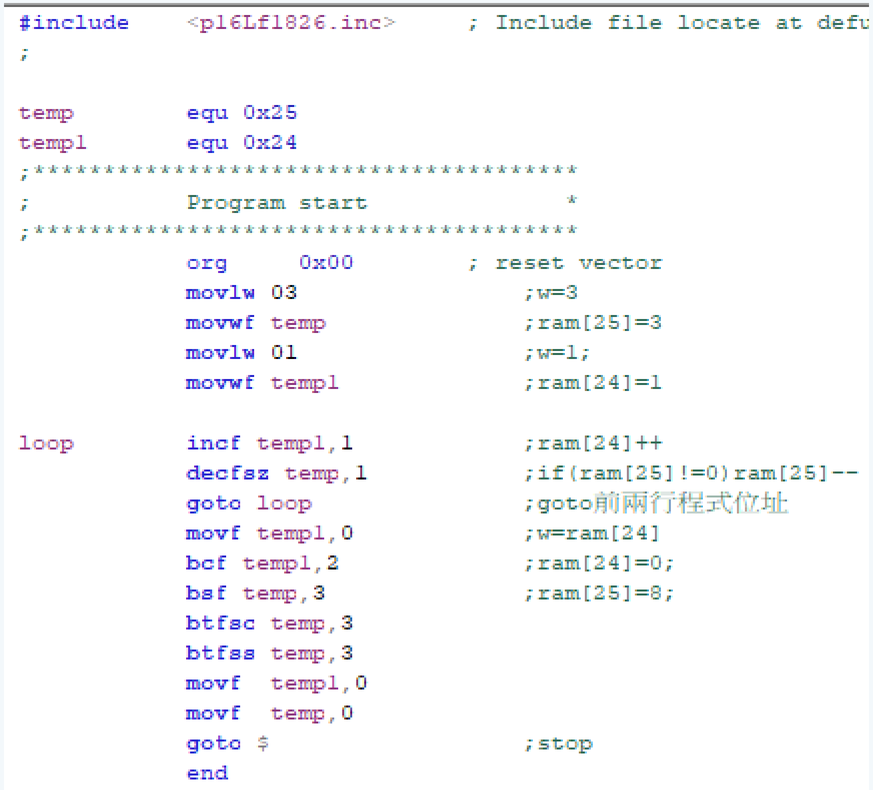




```
assign aluout_zero = (aluout == 0)? 1'b1: 1'b0;
```

回家作業

組合語言

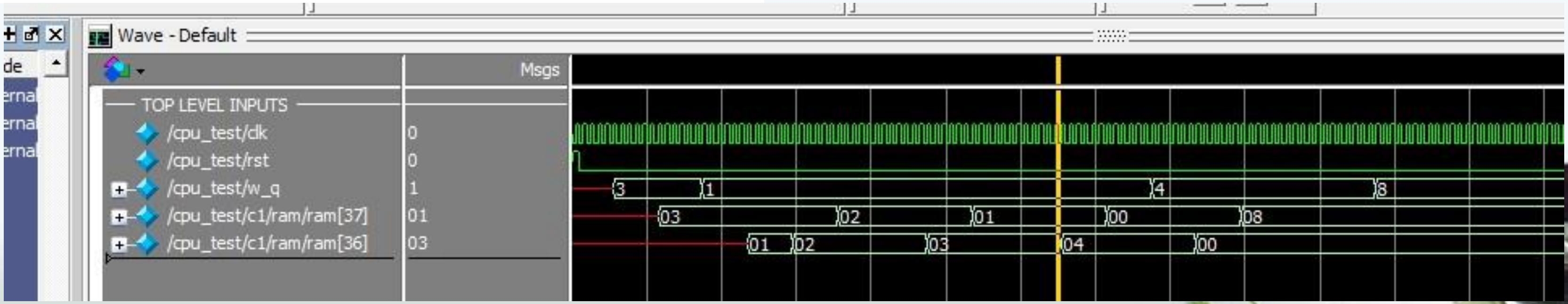


```
#include      <p16Lf1826.inc>       ; Include file locate at defu
;

temp          equ 0x25
temp1         equ 0x24
;****************************************
;             Program start            *
;****************************************
              org     0x00          ; reset vector
              movlw 03                  ;w=3
              movwf temp                ;ram[25]=3
              movlw 01                  ;w=1;
              movwf temp1               ;ram[24]=1

loop          incf temp1,1              ;ram[24]++
              decfsz temp,1             ;if(ram[25]!=0)ram[25]--
              goto loop                 ;goto前兩行程式位址
              movf temp1,0              ;w=ram[24]
              bcf temp1,2               ;ram[24]=0;
              bsf temp,3                ;ram[25]=8;
              btfsc temp,3
              btfss temp,3
              movf  temp1,0
              movf  temp,0
              goto $                    ;stop
              end
```

請搭配
1. MPLAB軟體操作投影片
2. 「HEX轉ROM」投影片
3. 「MPLAB及hex轉Program_Rom.sv教學」影片

練習撰寫組合語言及建立MPLAB專案，
然後搭配「課程_2k_perl」將所建立的MPLAB專案生成出的.hex檔轉換成Program_Rom.sv

Program_Rom.sv將自動生成，不用自己撰寫。



請將w_q、ram[36]、ram[37]以16進制表示