

Try to Design and Implementation
of a PIC16 compatible microcontroller

Skill up course

PIC16 Microcontroller

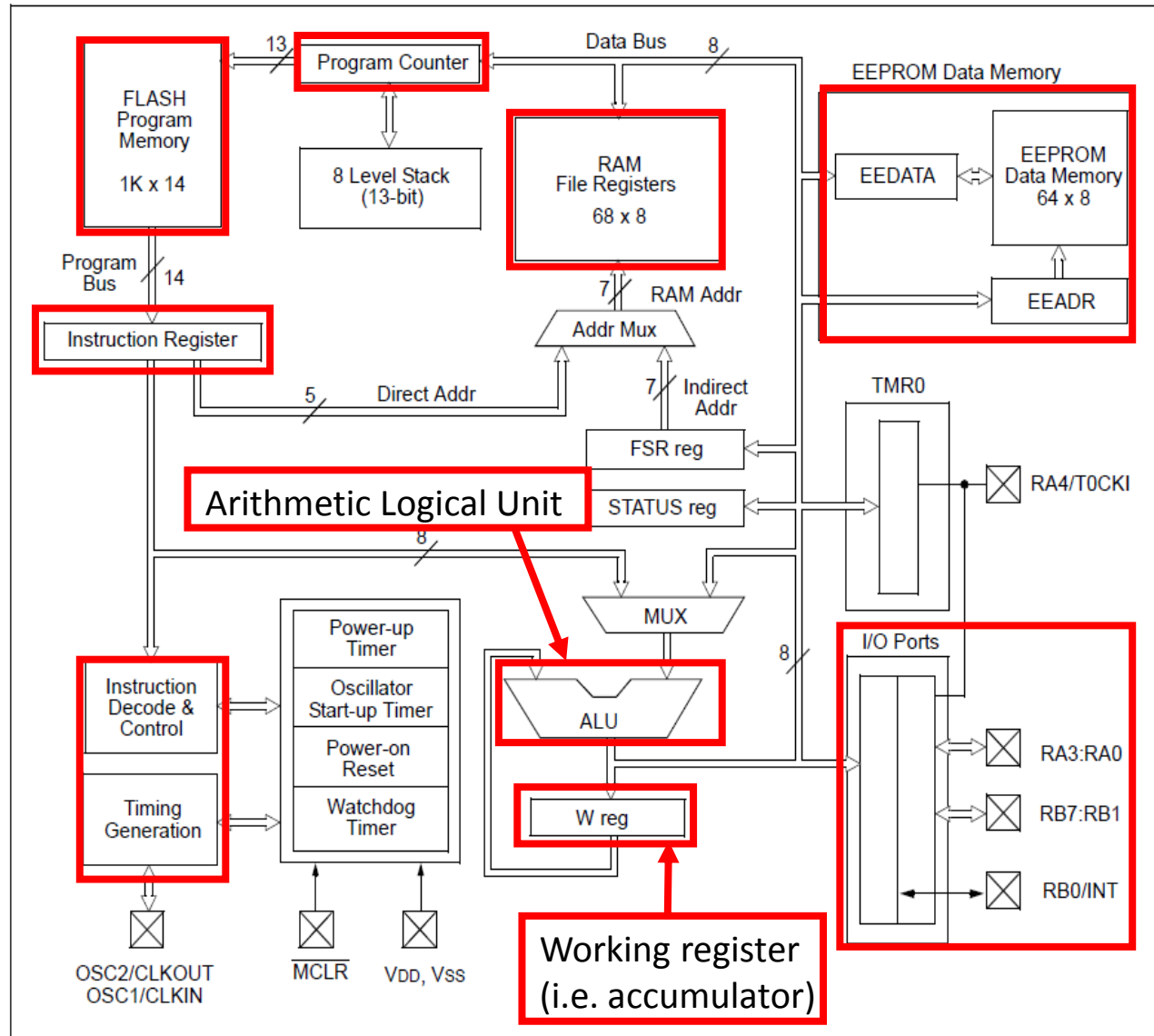
PIC16F84A by Microchip

Widely used for
electronics workshop.



Microcontroller:
It is one chip computer
includes of microprocessor,
memory and some
peripherals.
You can make several
controllers.

FIGURE 1-1: PIC16F84A BLOCK DIAGRAM



Sample program : Store a value into data memory

A constant value is stored into data memory combined with "movlw" and "movwf" instructions

movlw D'10' Store a constant 10₍₁₀₎ in instruction field to W register
movwf H'20' Store a value of W register into address 20₍₁₆₎ of data memory

mnemonic operand

Assembly Language:

It is a language for ease of understanding machine language.

Processor only recognizes binary value.

Program is also represented by binary value.

(Machine Language)

movlw D'10' 11_0000_0000_1010
movwf H'20' 00_0000_1010_0000

Machine language is generated by Assembler automatically.

Constants are represented by binary.

10₍₁₀₎ → 1010₍₂₎, 20₍₁₆₎ → 10000₍₂₎

Instruction register

· · 10(10)

movlw

W

10(10)

Address

20(16)

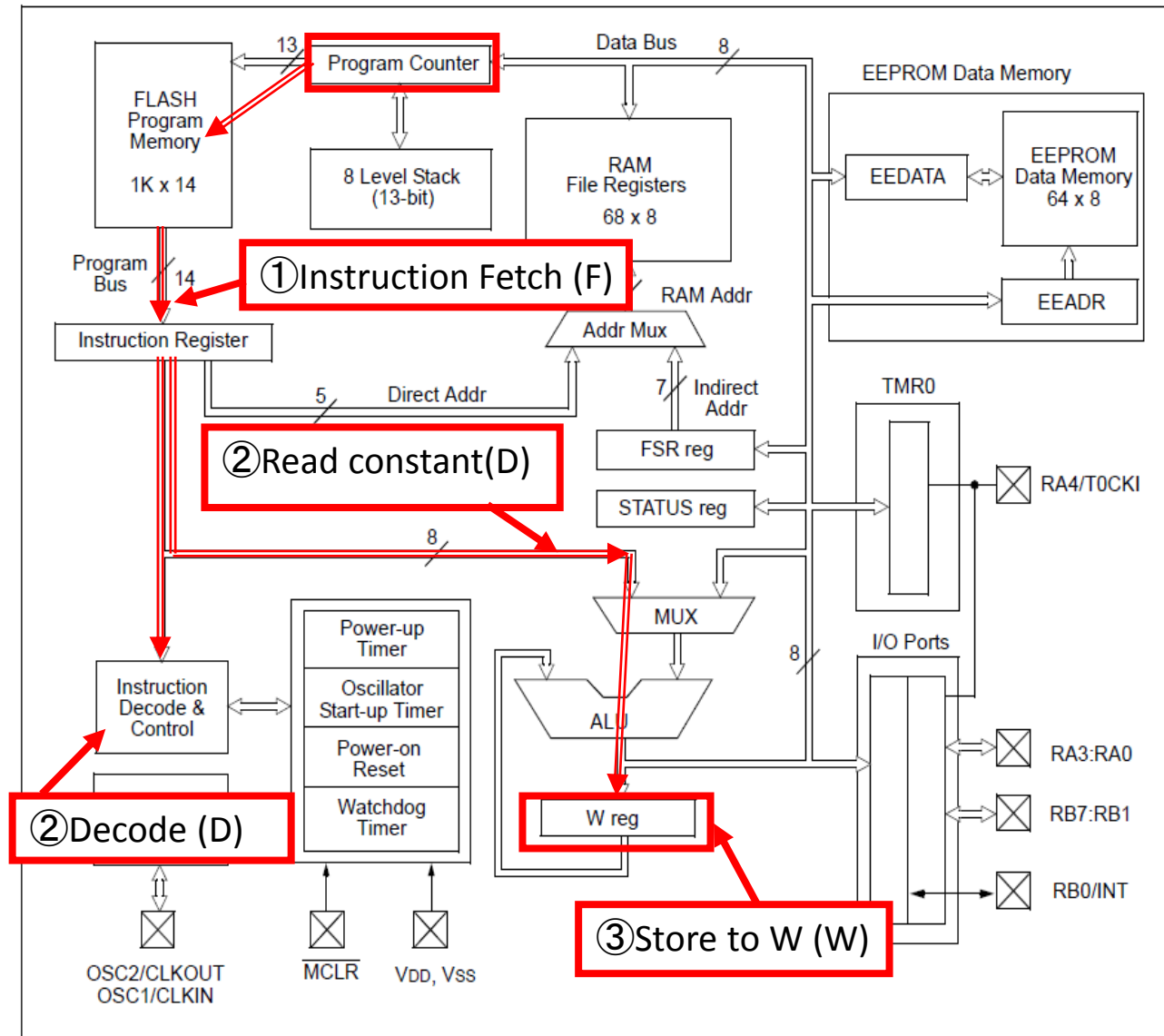
10(10)

movwf

Data memory

Behavior of "movlw" instruction

FIGURE 1-1: PIC16F84A BLOCK DIAGRAM



Instruction behavior that
a constant in instruction
is stored to W register.

movlw constant

① Instruction pointed
out by program
counter is stored to
instruction register.

② Instruction is
decoded. And constant
in instruction is ready
to be read.

③ Store the constant to
W register.

Behavior of "movwf" instruction

Instruction behavior that
a value in W register is
stored into data memory.

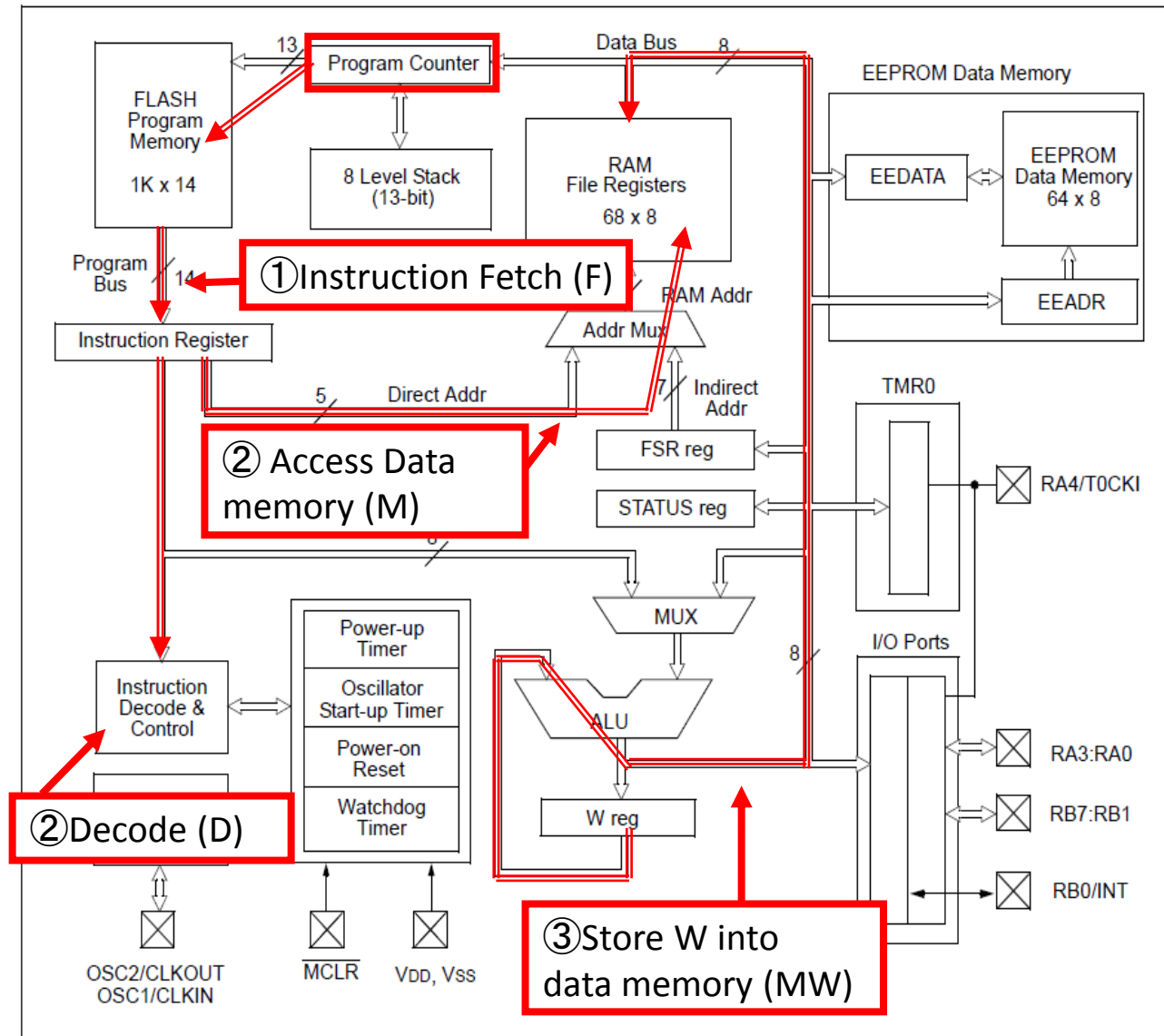
movwf Address

① Instruction pointed
out by program counter
is stored to instruction
register.

② Instruction is
decoded. And address
for accessing data
memory is ready.

③ W register value is
stored into data memory
through the
ALU

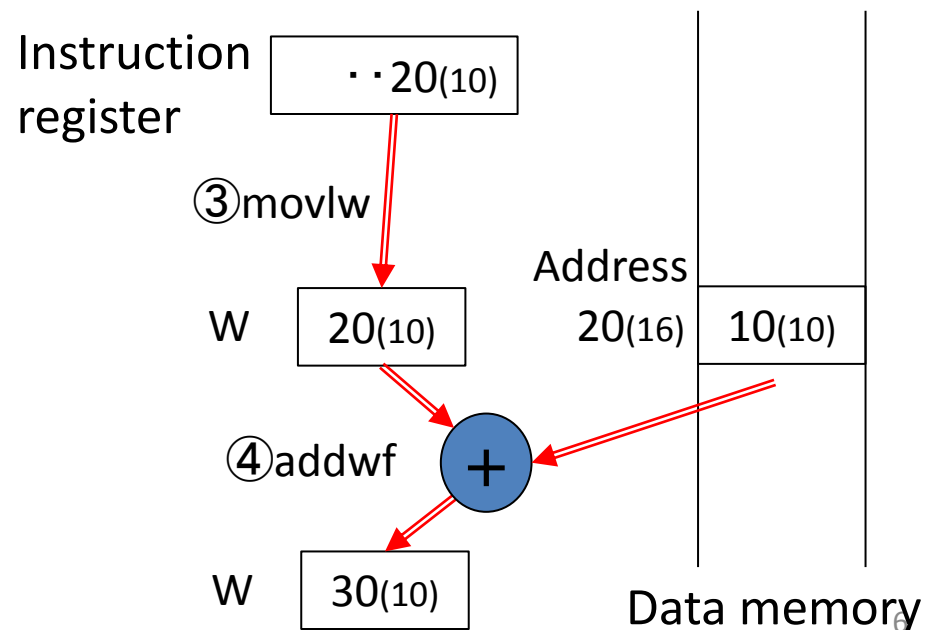
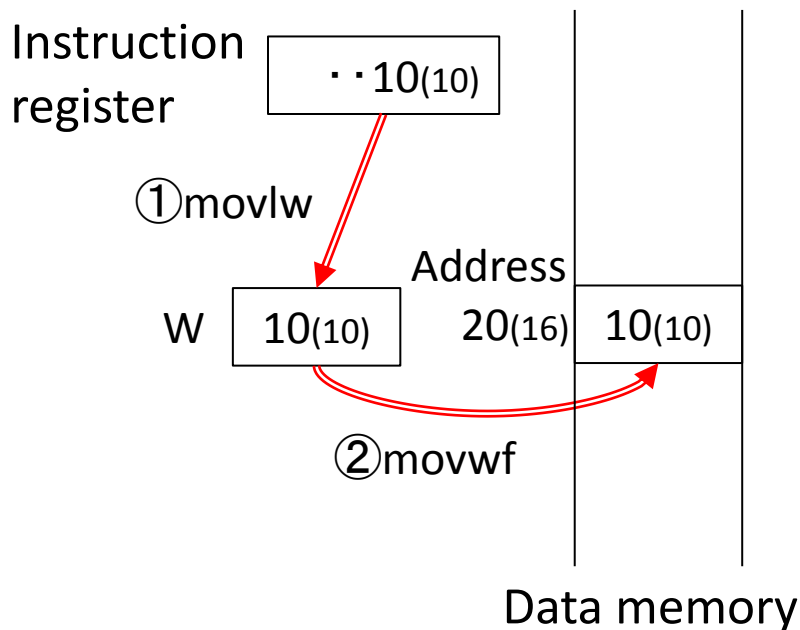
FIGURE 1-1: PIC16F84A BLOCK DIAGRAM



Sample Program : Addition

Add a value in data memory and W register by "addwf" instruction (Example: $10+20=30$)

- ① movlw D'10' Store a constant 10(10) in instruction field to W register
- ② movwf H'20' Store a value of W register into address 20(16) of data memory
- ③ movlw D'20' Store a constant 20(10) in instruction field to W register
- ④ addwf H'20',w Add W register and a value of data memory addressed by 20(20), and the result 20(10) is stored to W register



加算命令の動作例 (addwf Address,w)

Instruction behavior that a value in data memory and W register value are added and the result is stored to W register.

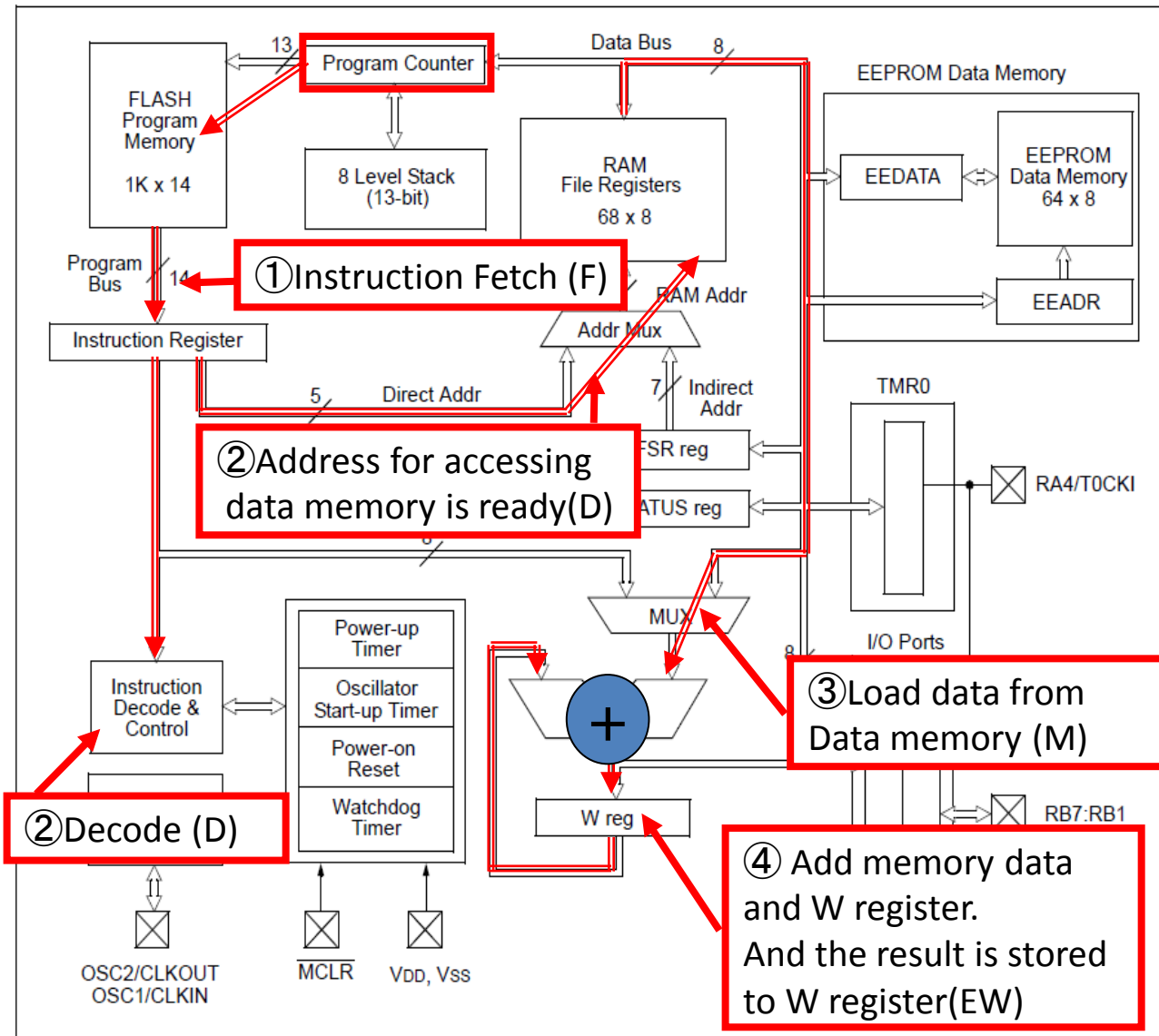
addwf Address, w

① Instruction pointed out by program counter is stored to instruction register.

② Instruction is decoded. And address for accessing data memory is ready.

③④ Add memory data and W register. And the result is stored to W register.

FIGURE 1-1: PIC16F84A BLOCK DIAGRAM



Instruction Set of PIC16F84A (1)

Mnemonic, Operands	Description	Cycles	14-Bit Opcode				Status Affected	Notes	
			MSb		LSb				
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	1fff	ffff	Z	2
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1 (2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1 (2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		
NOP	-	No Operation	1	00	0000	0xx0	0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2

x: Don't care

C: Carry flag, DC: Digit Carry flag

Z: Zero flag

 : Explained instructions in previous

f: Data memory Address
d: Data memory when d=0(f)
W register when d=1(W)

Instruction Set of PIC16F84A (2)

BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}, \overline{PD}$	
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	-	Return from interrupt	2	00	0000	0000	1001		
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	-	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}, \overline{PD}$	
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

f: Address of Data memory
b: Bit index for bit operation
k: Constant value (literal)

x: Don't care

C: Carry flag, DC: Digit Carry flag

Z: Zero flag

TO, PD: Status for internal states of processor

 : Explained instruction in previous

Summation program from 1 to 10

$$s = \sum_{i=1}^{10} i = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55$$

```
INCLUDE "p16f84a.inc"
```

```
LIST      P=PIC16F84A
```

```
ORG       0
```

```
clrf     H' 20'      ; s ← 0
```

```
movlw    D' 10'      ; w ← 10
```

```
movwf    H' 21'      ; i ← 10
```

LOOP

```
movf     H' 21', w    ; w ← i
```

```
addwf    H' 20', f    ; s ← i + s
```

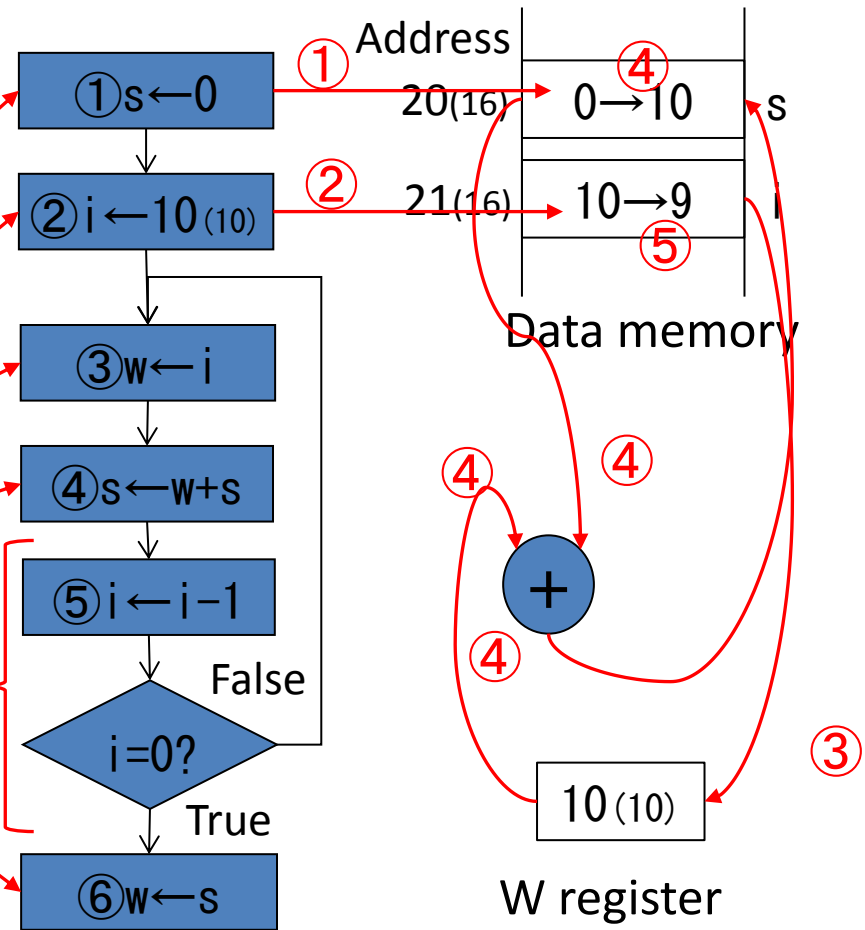
```
decfsz   H' 21', f    ; i ← i - 1
```

```
goto     LOOP
```

```
movf     H' 20', w    ; w ← s
```

```
clrw
```

```
end
```



Flow chart

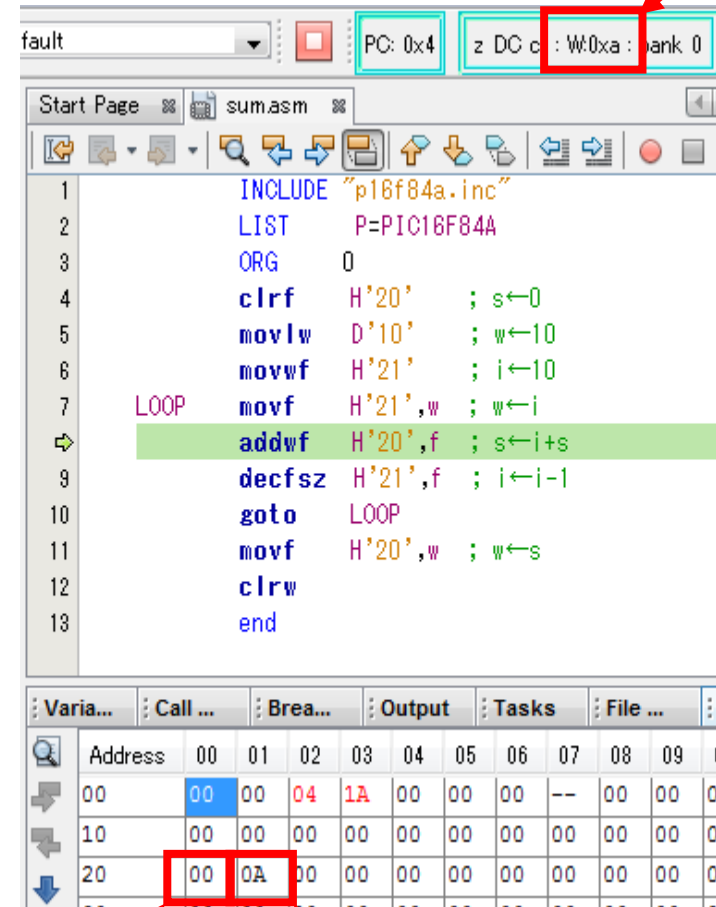
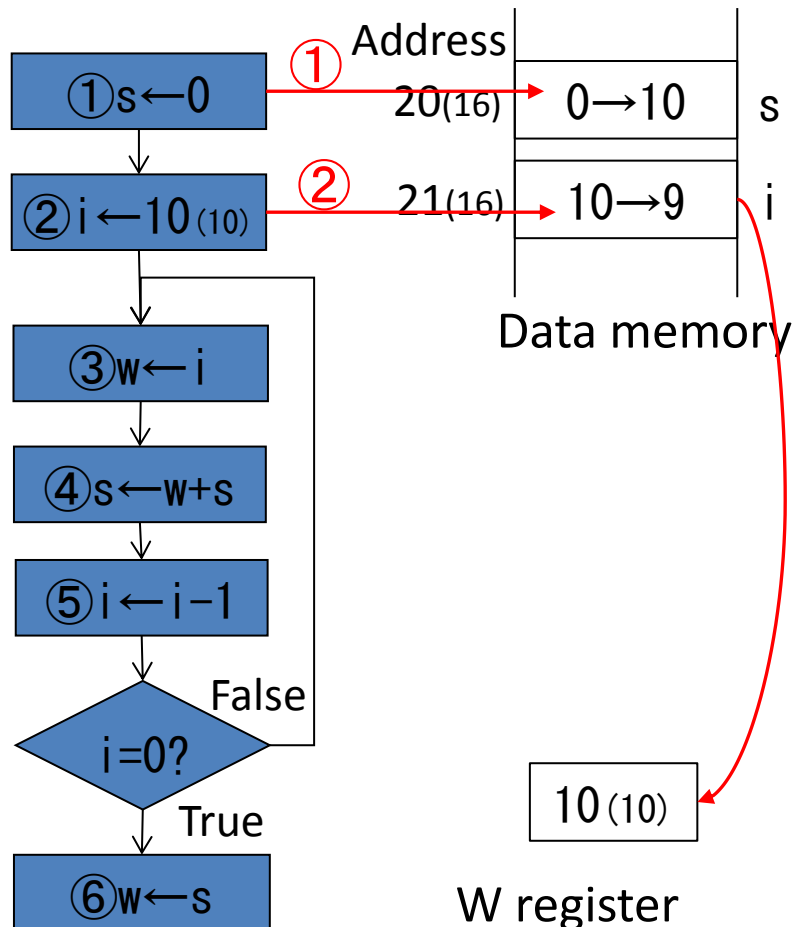
W register

When $i-1$ equals 0, next instruction is canceled and after the next instruction will be executed.

Simulation Result (1)

- Simulator: MPLAB X IDE by Microchip
 - The first iteration after executed ①②③ instructions

W=10₍₁₀₎



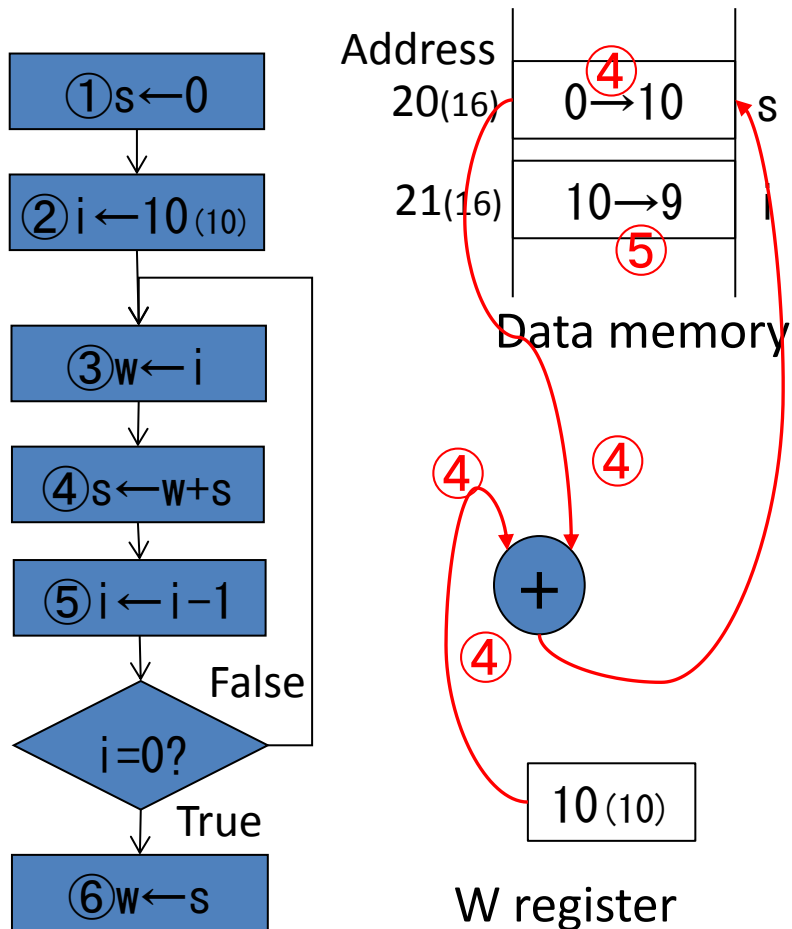
s=0₍₁₀₎

i=10₍₁₀₎

Simulation Result (2)

- Simulator: MPLAB X IDE by Microchip
 - The first iteration after executed ④⑤ instructions

W=10(10)



Assembly Code:

```

1  INCLUDE "p18f84a.inc"
2  LIST P=PIC18F84A
3  ORG 0
4  clrf H'20' ; s ← 0
5  movlw D'10' ; w ← 10
6  movwf H'21' ; i ← 10
7  LOOP movf H'21',w ; w ← i
8      addwf H'20',f ; s ← i + s
9      decfsz H'21',f ; i ← i - 1
10     goto LOOP
11     movf H'20',w ; w ← s
12     clrw
13     end
    
```

Register Values:

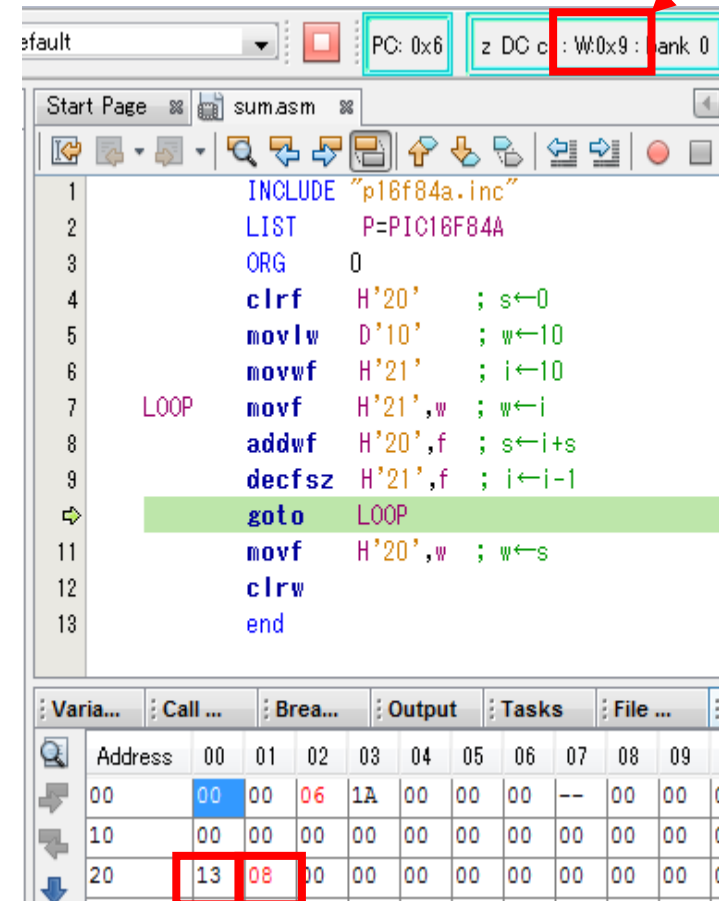
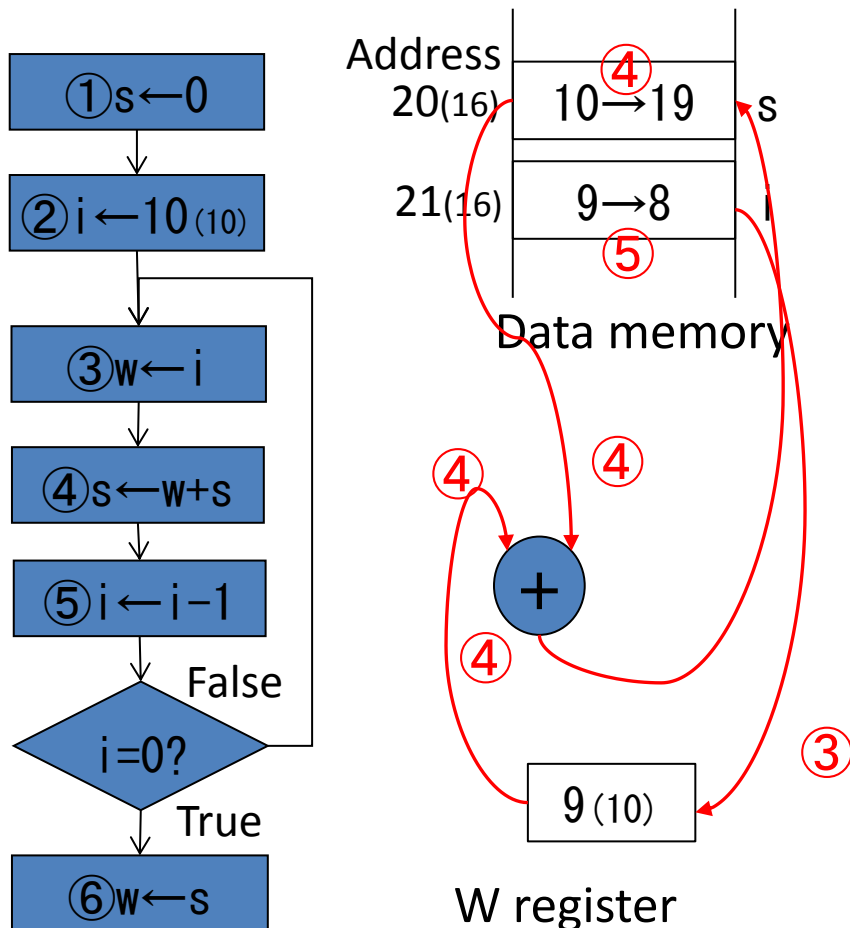
Address	00	01	02	03	04	05	06	07	08	09	0A	0B
00	00	00	06	18	00	00	00	--	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00
20	0A	09	00	00	00	00	00	00	00	00	00	00

s=10(10)

i=9(10)

Simulation Result (3)

- Simulator: MPLAB X IDE by Microchip
 - The second iteration after executed ③④⑤ instructions

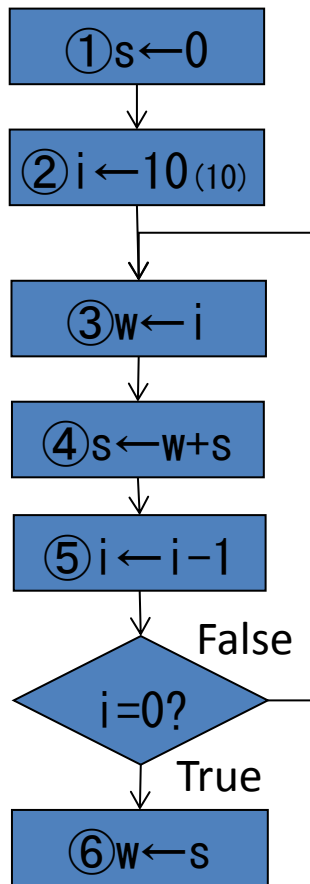


$s=19_{(10)}$

$i=8_{(10)}$

Simulation Result (4)

- Simulator: MPLAB X IDE by Microchip
 - The tenth iteration after executed ④⑤⑥ instructions



Flow chart

Address		
20(16)	54 → 55	s
21(16)	1 → 0	i

Data memory

W register

55(10)

W=55(10)

The screenshot shows the MPLAB X IDE interface. The top status bar displays 'PC: 0x8', 'z dc c', and 'W: 0x37 : bank 0'. The assembly code window shows the following code:

```

1  INCLUDE "p16f84a.inc"
2  LIST P=PIC16F84A
3  ORG 0
4  clrf H'20' ; s ← 0
5  movlw D'10' ; w ← 10
6  movwf H'21' ; i ← 10
7  LOOP movf H'21',w ; w ← i
8      addwf H'20',f ; s ← i+s
9      decfsz H'21',f ; i ← i-1
10     goto LOOP
11     movf H'20',w ; w ← s
12     clrw
13     end
    
```

The register window at the bottom shows the following values:

Address	00	01	02	03	04	05	06	07	08	09	0A
00	00	00	08	18	00	00	00	--	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00
20	37	00	00	00	00	00	00	00	00	00	00

s=55(10)

i=0(10)

Differences from the original

Original PIC16

- 4 cycles in one stage
- Selectable external clock or internal clock
- Sleep mode (Low power consumption mode)
- Flash memory

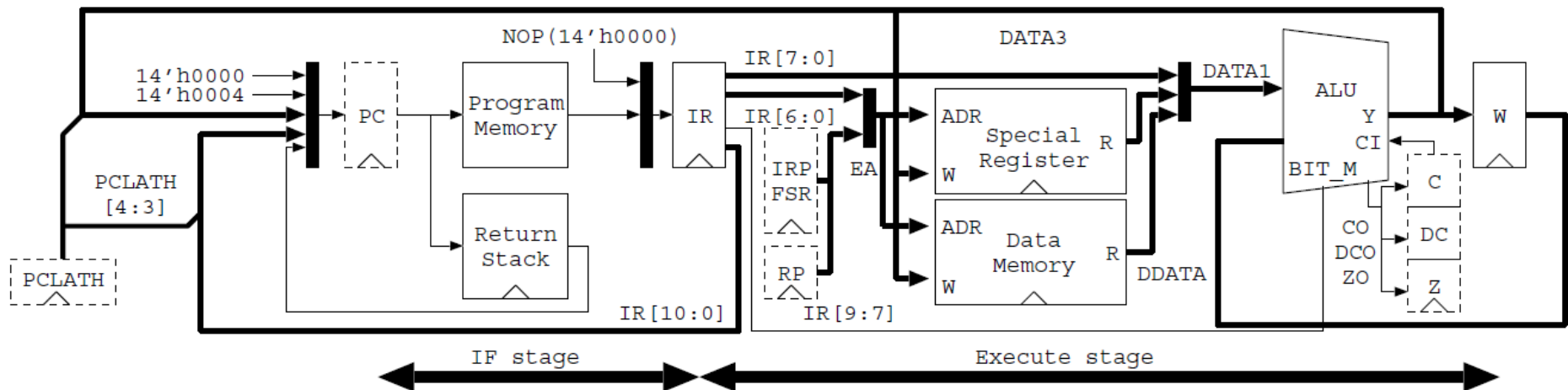
Our specification

- 1 cycle in one stage
- Only external clock
- Different behavior by 1 cycle in one stage:
interrupt, I/O write
- Pseudo Sleep mode (It is repeating NOP instruction)
- No flush memory

Block Diagram

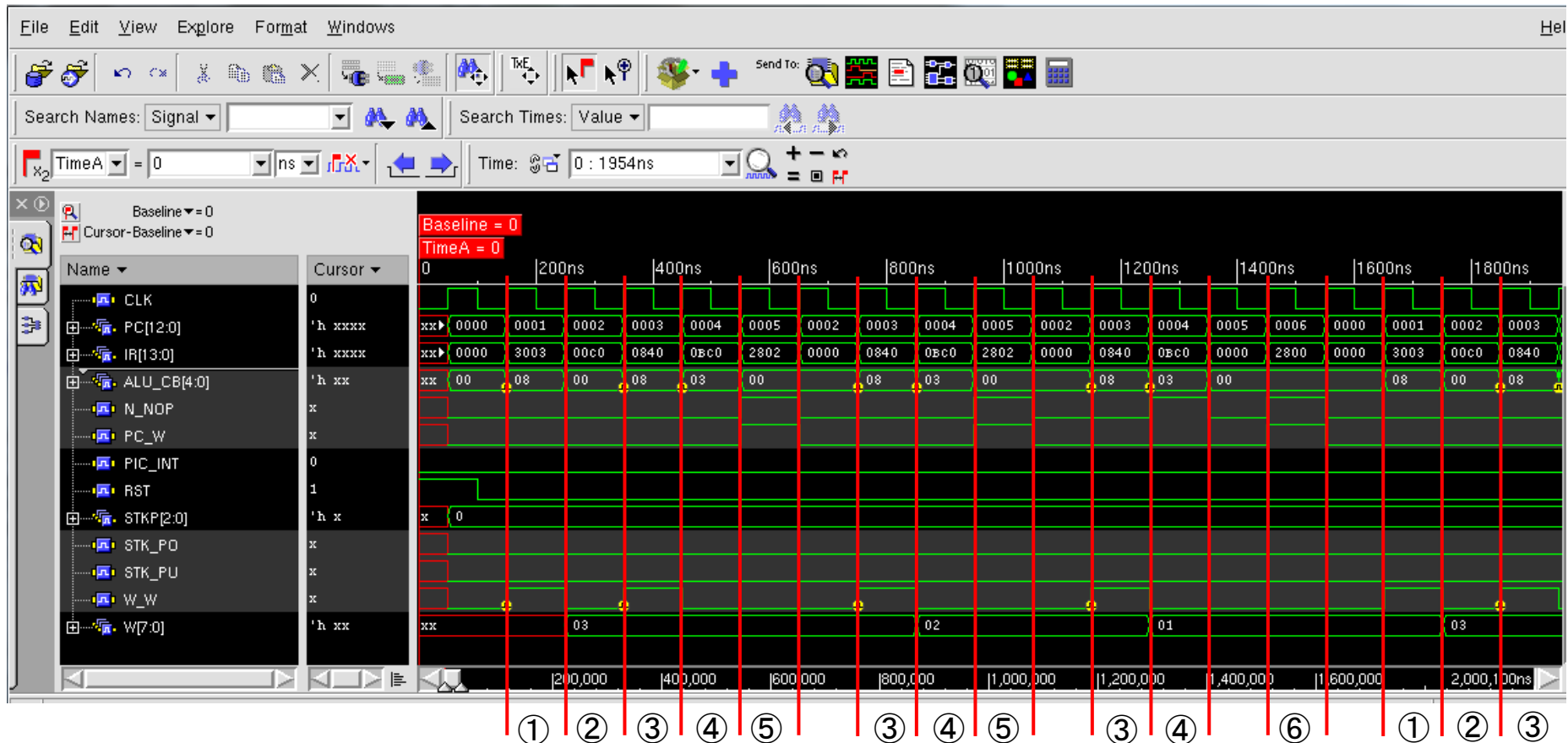
- 2-stage Pipeline
 - Instruction Fetch (IF) stage
 - Execution (E) stage

} Each stage is executed in parallel
- Critical Path
 - IR→Data Memory→ALU→Data Memory
 - Operating frequency cannot be faster because of Read/Write accessing of Data Memory in one cycle.

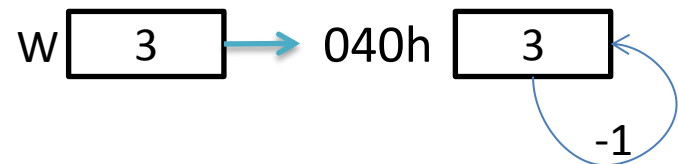


Behavior of 2 Stage Pipeline Processor

Behavior of PIC16 compatible processor designed in our Lab. (Note that original PIC 16 has 4 cycles in a one stage)

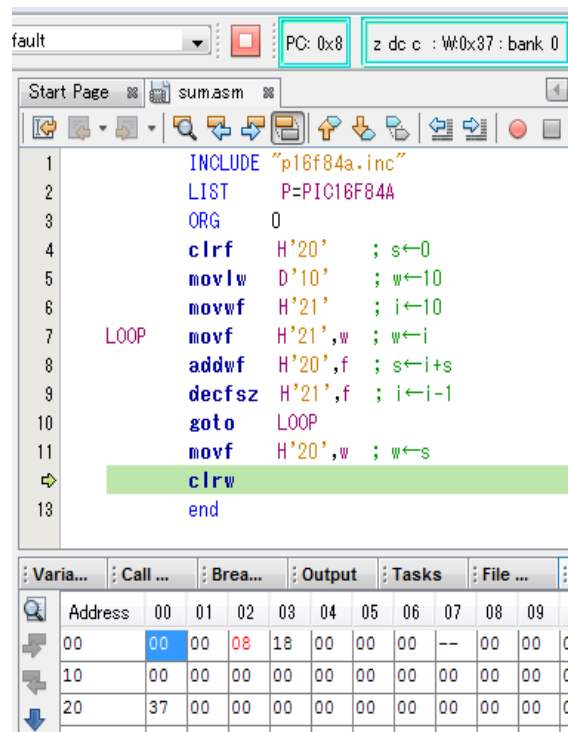


0000	0000	00001	ORG 0
0000	3003	00002	L2: ① MOVLW 3
0001	00C0	00003	② MOVWF 040h
0002	0840	00004	L1: ③ MOVF 040h, 0
0003	0BC0	00005	④ DECFSZ 040h, 1
0004	2802	00006	⑤ GOTC L1
0005	2800	00007	⑥ GOTC L2
		00008	end



Programmer for download a program

Developed program is downloadable via programmer, such as PICkit3 and others.



The screenshot shows a PIC assembler software interface. The main window displays assembly code for a PIC16F84A. The code includes a loop that increments a counter and checks for a zero flag. The memory dump at the bottom shows the program memory addresses and their contents.

```
1 INCLUDE "p16f84a.inc"
2 LIST P=PIC16F84A
3 ORG 0
4 clrf H'20' ; s←0
5 movlw D'10' ; w←10
6 movwf H'21' ; i←10
7 LOOP movf H'21',w ; w←i
8 addwf H'20',f ; s←i+s
9 decfsz H'21',f ; i←i-1
10 goto LOOP
11 movf H'20',w ; w←s
12 clrw
13 end
```

Address	00	01	02	03	04	05	06	07	08	09	0A
00	00	00	0E	18	00	00	00	--	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00
20	37	00	00	00	00	00	00	00	00	00	00

