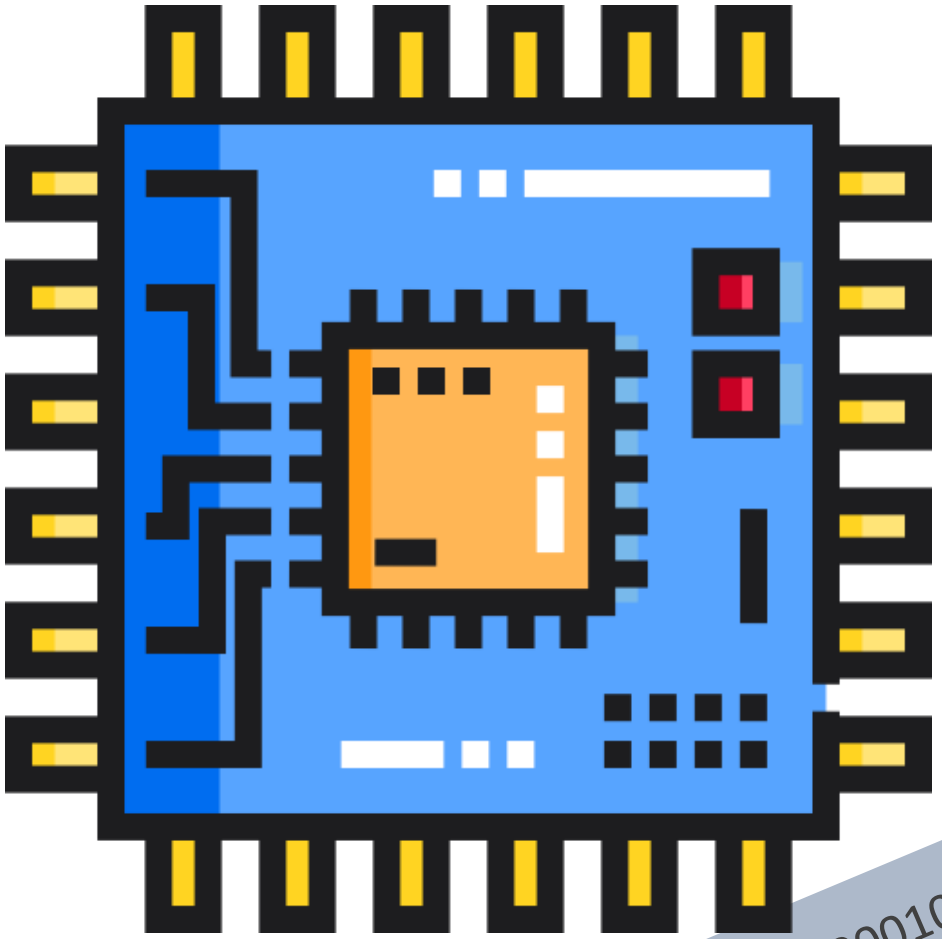
圖片來源：https://www.flaticon.com/

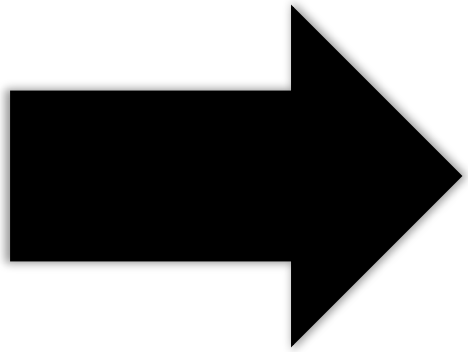


伴學松

Decode Module

```
int main (void)
{
    printf("Hello world!\n");
    return 0;
}
```

我們看得懂

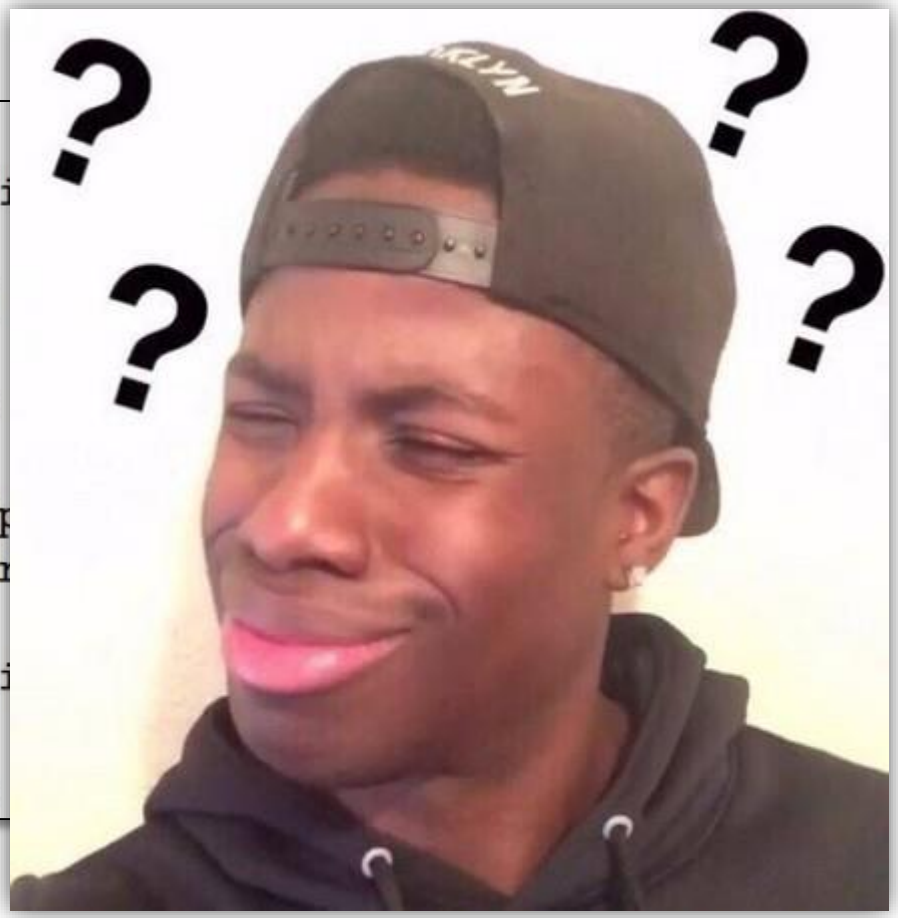


```
00000000 <main>:
 0: ff010113  addi  sp,sp,-16
 4: 00112623  sw    ra,12(sp)
 8: 00000537  lui   a0,0x0
 c: 00050513  mv    a0,a0
10: 000005b7  lui   a1,0x0
14: 00058593  mv    a1,a1
18: 00000097  auipc ra,0x0
1c: 000080e7  jalr  ra
20: 00c12083  lw    ra,12(sp)
24: 01010113  addi  sp,sp,16
28: 00000513  li    a0,0
2c: 00008067  ret
```

機器看得懂

圖片來源：https://www.ettoday.net

```
00000000 <main>:
 0: ff010113  addi
 4: 00112623  sw
 8: 00000537  lui
 c: 00050513  mv
10: 000005b7  lui
14: 00058593  mv
18: 00000097  auip
1c: 000080e7  jalr
20: 00c12083  lw
24: 01010113  addi
28: 00000513  li
2c: 00008067  ret
```



```
int x = 6;
int y = 2;
int z = x << y;
int j = x << 2;
```

高階語言

編譯器
Compiler

```
li a0 6
li a1 2
sll a2 a0 a1
slli a3 a0 2
```

組合語言

組譯器
Assembler

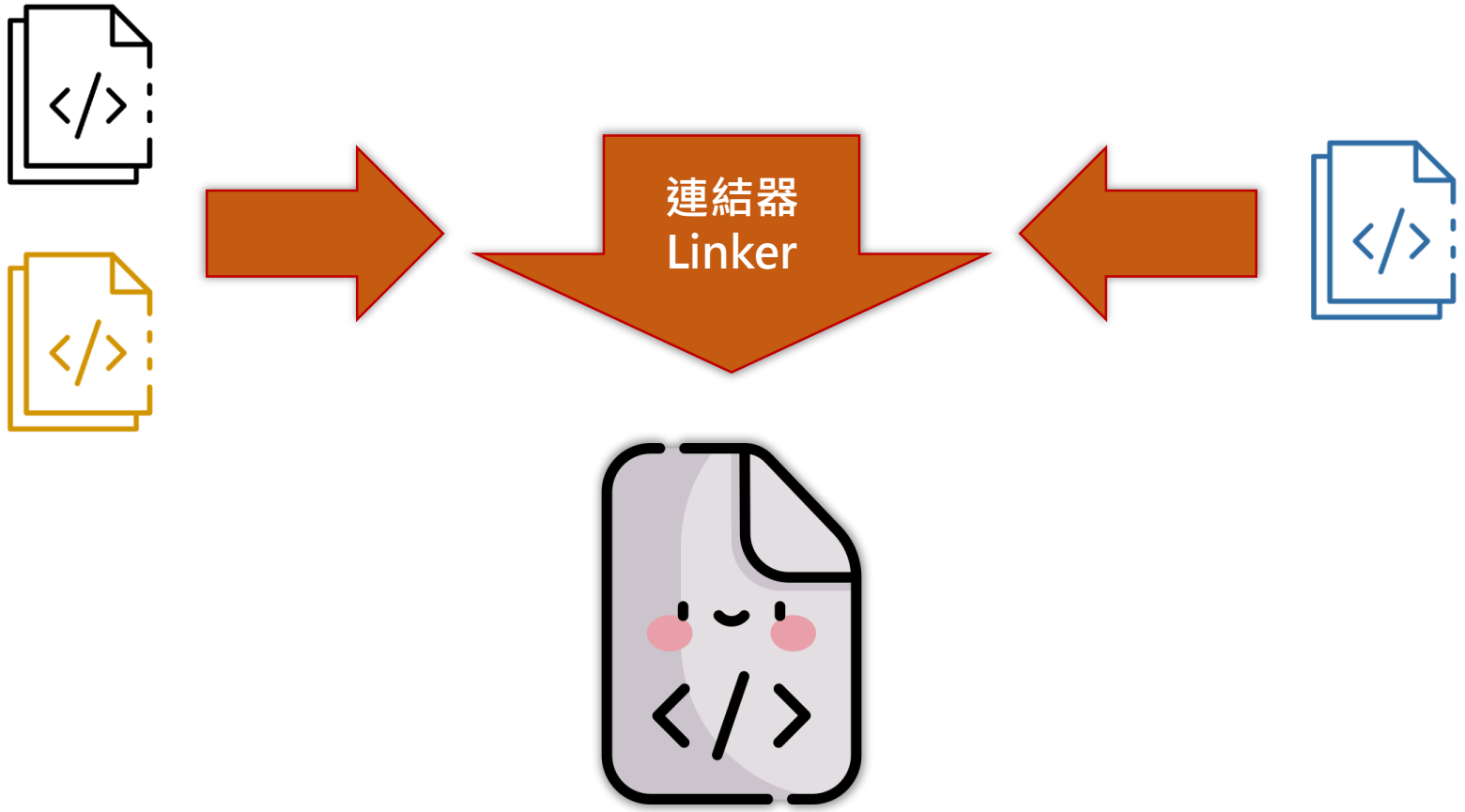
```
0: 00600513
4: 00200593
8: 00b51633
c: 00251693
```

機器語言

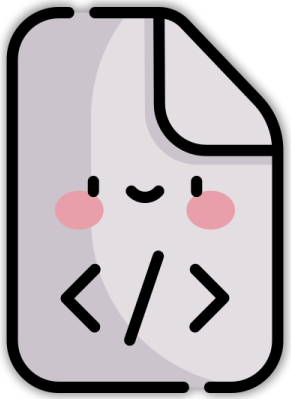
```
00000000 <main>:
 0: ff010113  addi  sp,sp,-16
 4: 00112623  sw    ra,12(sp)
 8: 00000537  lui   a0,0x0
 c: 00050513  mv    a0,a0
10: 000005b7  lui   a1,0x0
14: 00058593  mv    a1,a1
18: 00000097  auipc ra,0x0
1c: 000080e7  jalr  ra
20: 00c12083  lw    ra,12(sp)
24: 01010113  addi  sp,sp,16
28: 00000513  li    a0,0
2c: 00008067  ret
```

圖片來源：https://www.flaticon.com/

```
0:  00600513
4:  00200593
8:  00b51633
c:  00251693
```

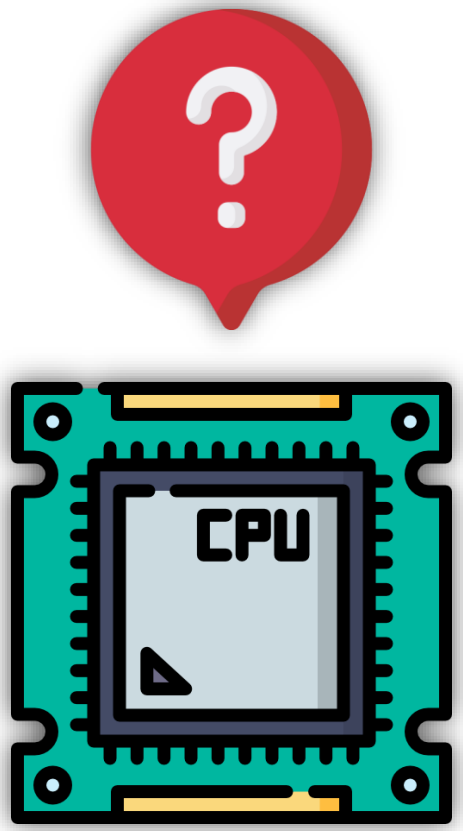


圖片來源：https://www.flaticon.com/



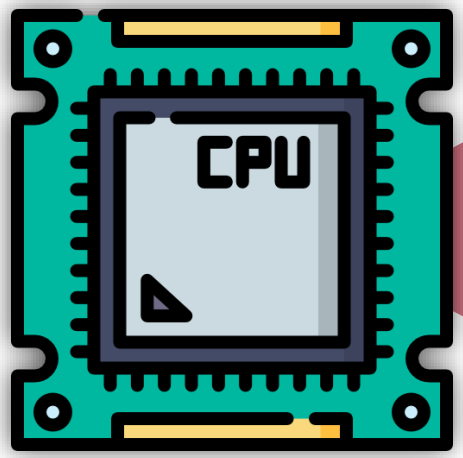
可執行檔

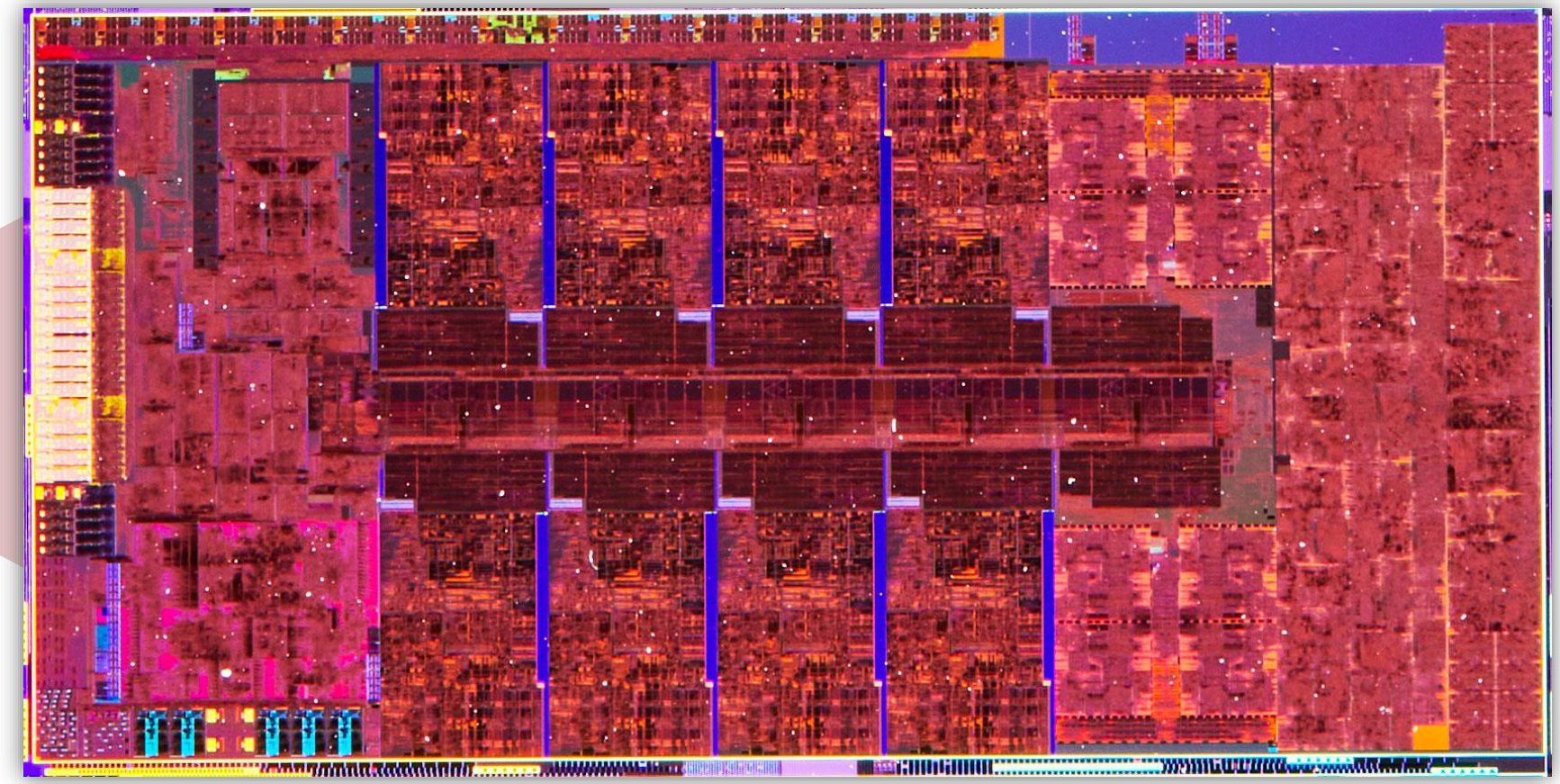
載入器
Loader



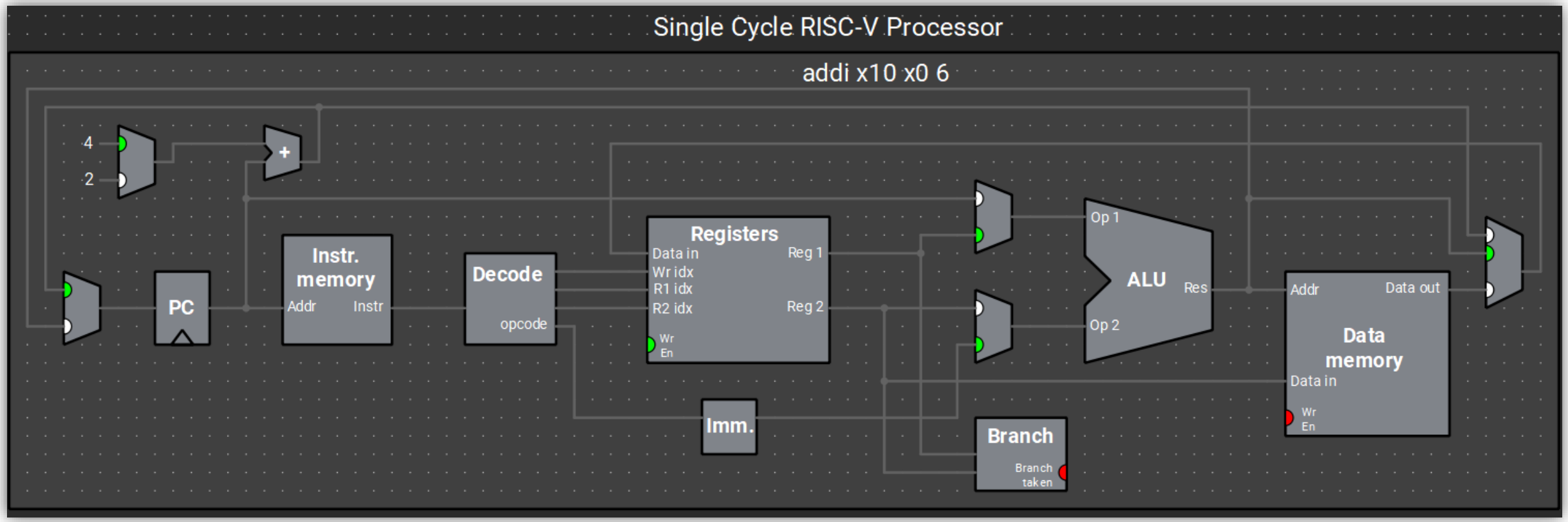
如何知道
需要運作的步驟？

圖片來源：https://www.techpowerup.com



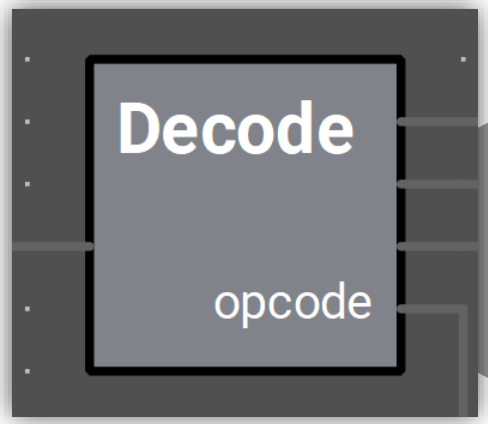


圖片來源：Ripes - visual computer architecture simulator



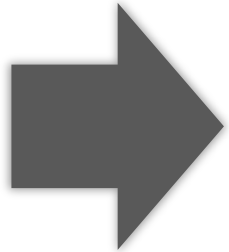
圖片來源：Ripes - visual computer architecture simulator

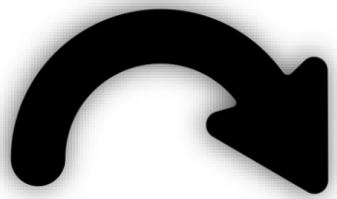
圖片來源：https://www.flaticon.com/



解碼器

```
00000000 <main>:
 0: ff010113   addi   sp,sp,-16
 4: 00112623   sw     ra,12(sp)
 8: 00000537   lui    a0,0x0
 c: 00050513   mv     a0,a0
10: 000005b7   lui    a1,0x0
14: 00058593   mv     a1,a1
18: 00000097   auipc  ra,0x0
1c: 000080e7   jalr   ra
20: 00c12083   lw     ra,12(sp)
24: 01010113   addi   sp,sp,16
28: 00000513   li     a0,0
2c: 00008067   ret
```





跳躍

計算

決策

R Type							
指令	funct7	rs2	rs1	funct3	rd	OP	ALU_SIGN
add	0000000	rs2	rs1	000	rd	0110011	rd = rs1 + rs2
sub	0100000	rs2	rs1	000	rd	0110011	rd = rs1 - rs2
sll	0000000	rs2	rs1	001	rd	0110011	rd = rs1 << rs2[4:0]
slt	0000000	rs2	rs1	010	rd	0110011	rd = rs1 <s rs2
sltu	0000000	rs2	rs1	011	rd	0110011	rd = rs1 <u rs2
xor	0000000	rs2	rs1	100	rd	0110011	rd = rs1 ^ rs2
srl	0000000	rs2	rs1	101	rd	0110011	rd = rs1 >> rs2[4:0]
sra	0100000	rs2	rs1	101	rd	0110011	rd = rs1 >>s rs2[4:0]
or	0000000	rs2	rs1	110	rd	0110011	rd = rs1 | rs2
and	0000000	rs2	rs1	111	rd	0110011	rd = rs1 & rs2

7 bits	5 bits	5 bits	3 bits	5 bits	7 bits
32~25	24~20	19~15	14~12	11~7	6~0

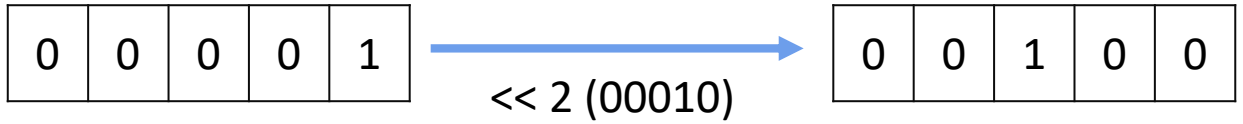
```
add    rd = rs1 + rs2
sub    rd = rs1 - rs2

and    rd = rs1 & rs2
or     rd = rs1 |  rs2
xor    rd = rs1 ^ rs2
```

輸入		輸出
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

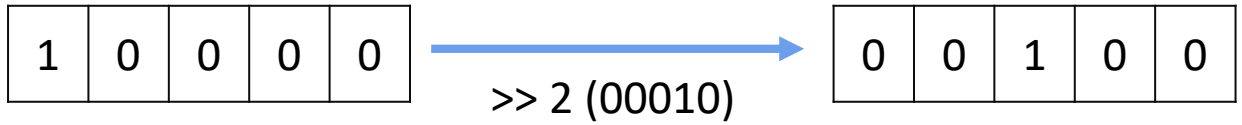
輸入		輸出
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

輸入		輸出
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

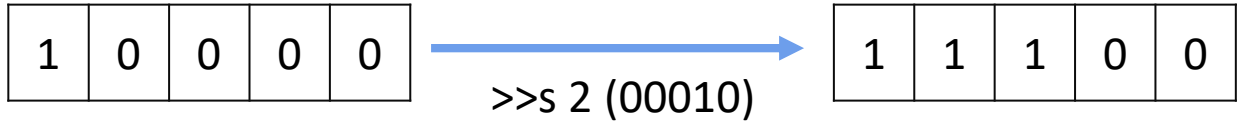
邏輯左移: sll rd = rs1 << rs2[4:0]



邏輯右移: srl rd = rs1 >> rs2[4:0]



算數右移: sra rd = rs1 >>s rs2[4:0]



slt　　if (rs1<rs2)　rd=1

　　else　　rd=0

sltu　if (|rs1| < |rs2|)　rd=1　(比較時視為無號數)

　　else　　rd=0

0x00940533

↓ 16進制轉2進制

0000000	01001	01000	000	01010	0110011
funct7	rs1	rs2	funct3	rd	opcode
32~25	24~20	19~15	14~12	11~7	6~0

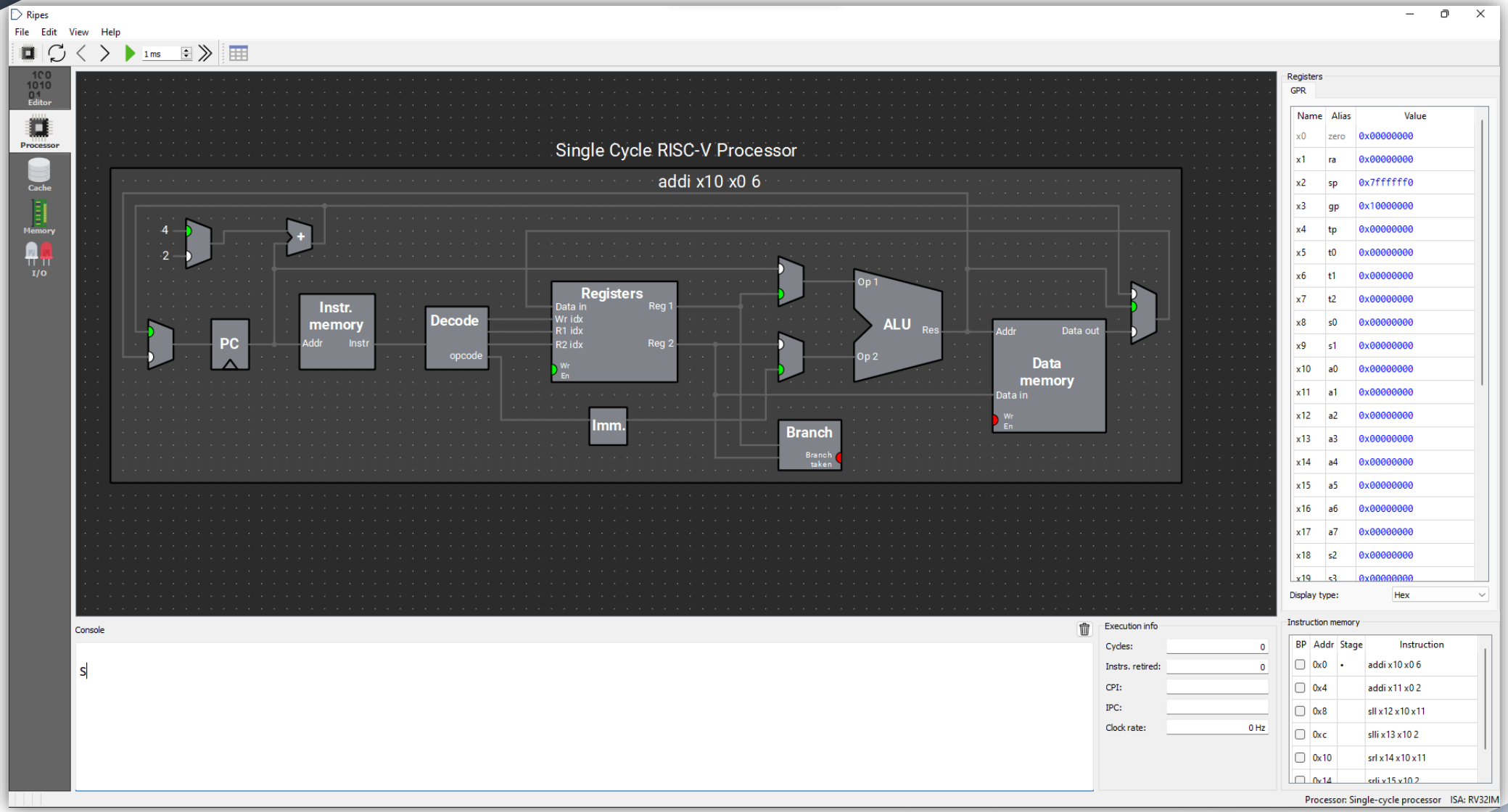
↓ 判斷opcode(指令類型)

0110011 => R-type

↓ 判斷funct3、funct7(細分指令)

funct3: 000 funct7: 0000000 => add

圖片來源：Ripes - visual computer architecture simulator



Any Question?