INSTRUCTION SET IN DIRECT MODE

Revision 4
Nov-02-2022

指令集

以直接模式

版本4
2022年11月02日

CODING

OPC	MNEMONIC	FLAGS	CODE															
			OPCODE				OPERAND X OPCODE				OPERAND Y							
1	ADD RX,RY	V Z C	0	0	0	1	X	X	X	X	Y	Y	Y	Y				
2	ADC RX,RY	V Z C	0	0	1	0	X	X	X	X	Y	Y	Y	Y				
3	SUB RX,RY	V Z C	0	0	1	1	X	X	X	X	Y	Y	Y	Y				
4	SBB RX,RY	V Z C	0	1	0	0	X	X	X	X	Y	Y	Y	Y				
5	OR RX,RY	Z	0	1	0	1	X	X	X	X	Y	Y	Y	Y				
6	AND RX,RY	Z	0	1	1	0	X	X	X	X	Y	Y	Y	Y				
7	XOR RX,RY	Z	0	1	1	1	X	X	X	X	Y	Y	Y	Y				
8	MOV RX,RY		1	0	0	0	X	X	X	X	Y	Y	Y	Y				
02	INC RY	Z C	0	0	0	0	0	0	1	0	Y	Y	Y	Y				
03	DEC RY	Z C	0	0	0	0	0	0	1	1	Y	Y	Y	Y				
0D	RRC RY	Z C	0	0	0	0	1	1	0	1	Y	Y	Y	Y				

编码

OPC	MNEMONIC FLAGS		CODE		
			操作码	操作数X 操作码	操作数Y
1	ADD ADC SUB SBB 其它	V Z C	0 0 0 1	X X X Y Y Y Y Y	
2	RX，RY RX，RY RX，RY RX，RY RX，RY	V Z C	0 0 1 0	X X X Y Y Y Y Y	
3		V Z C	0 0 1 1	X X X Y Y Y Y Y	
4		V Z C	0 1 0 0	X X X Y Y Y Y Y	
5	或y	Z	0 1 0 1	X X X Y Y Y Y Y	
6		Z	0 1 1 0	X X X Y Y Y Y Y	
7 8	XOR MOV 文件 RX，RY	Z	0 1 1 1	X X X Y Y Y Y Y	
			1 0 0 0	X X X Y Y Y Y Y	
02	INC DEC RX RY	Z C	0 0 0 0 0 0 1 0	依依	
03		Z C	0 0 0 0 0 0 1 1	依依	
0D	RRC RY	Z C	0 0 0 0 1 1 0 1	依依	

ADD X,Y

Add register Y to register X

Syntax: {label} ADD X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X + Y

Description: Add with Carry contents of the register Y to the contents of the register X and place the result in the register X.

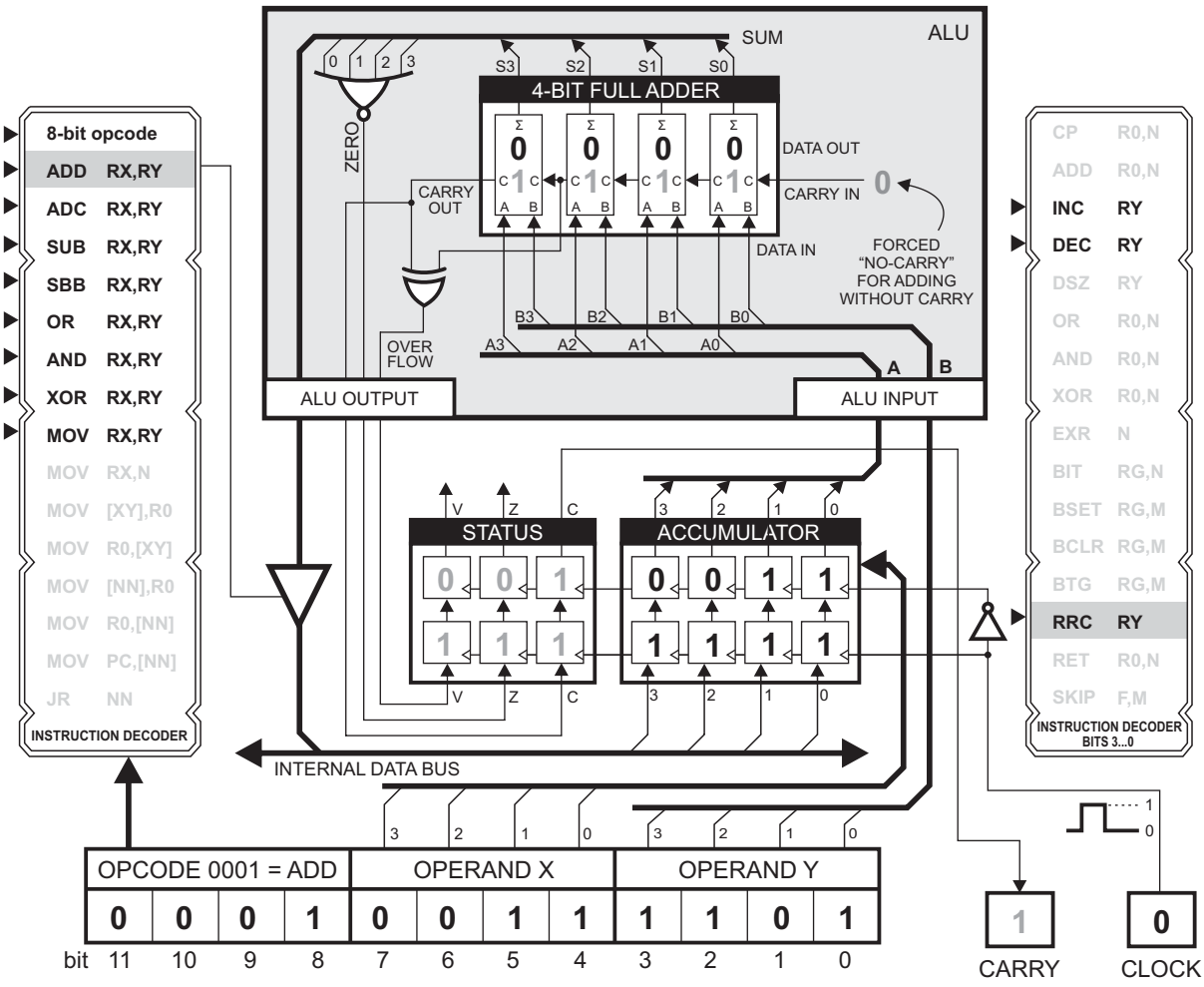
Flags affected: If there is the overflow (if X + Y > 15), set C.
Otherwise, reset C.
If result = 0000 after operation, set Z. Otherwise, reset Z.
If there is the underflow for signed representation, set V.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	1	X	X	X	X	Y	Y	Y	Y

The "0001" bits are the ADD X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: ADD 3, 13



添加X，Y

将寄存器Y添加到寄存器X

语法：

{label} ADD X，Y

操作数：

X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：

X ← X + Y

产品描述：

用进位将寄存器Y的内容与寄存器X的内容相加，并将结果放入寄存器X。

受影响的旗帜：

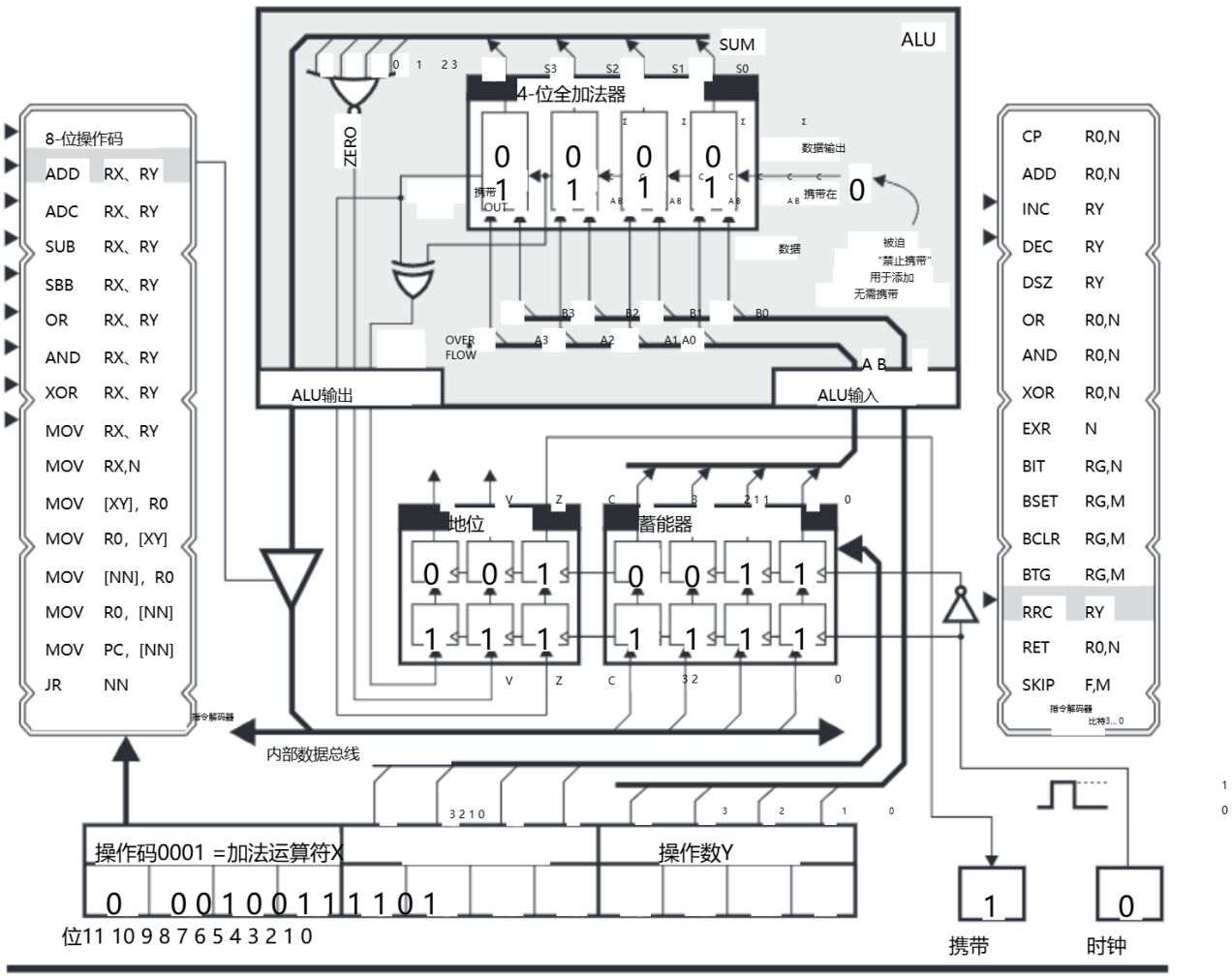
如果存在溢出（如果X + Y > 15），则设置C。
否则，重置C。
如果操作后结果= 0000，则设置Z。否则，重置Z。
如果有符号表示存在下溢，则设置V。

编码方式：

“0001”位是ADD X，Y操作码。“0001”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：

增加3、13



ADC X,Y

Add with Carry register Y to register X

Syntax: {label} ADC X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X + Y + Carry

Description: Add with Carry contents of the register Y to the contents of the register X and place the result in the register X.

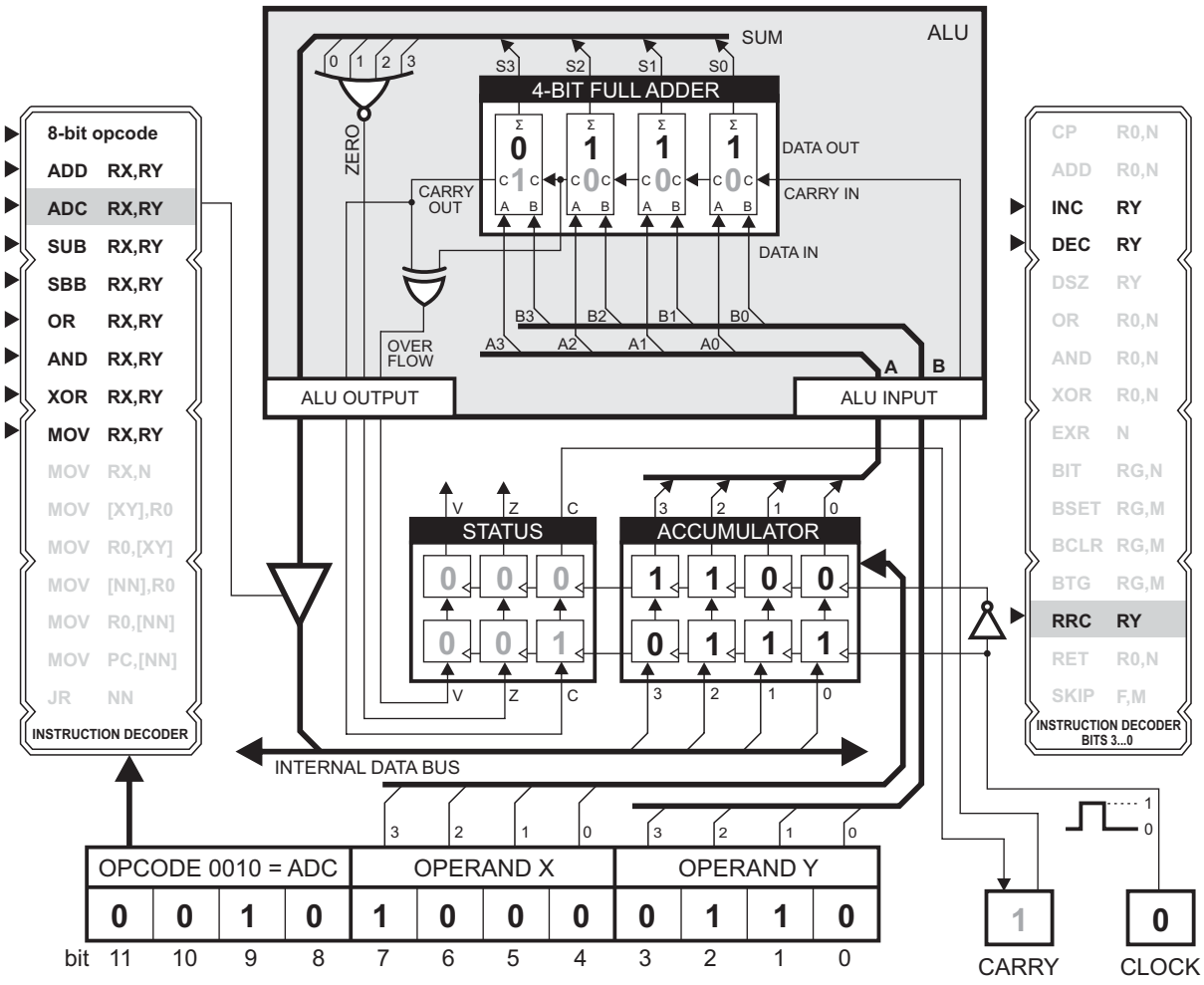
Flags affected: If there is the overflow (if X + Y + Carry > 15), set C.
Otherwise, reset C.
If result = 0000 after operation, set Z. Otherwise, reset Z.
If there is the underflow for signed representation, set V.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	1	0	X	X	X	X	Y	Y	Y	Y

The "0010" bits are the ADD X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: ADC 8, 6 (Carry set)



ADC X，Y　进位寄存器Y与寄存器X相加

语法：　{label} ADC X，Y

操作数：　X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：　X ← X + Y +进位

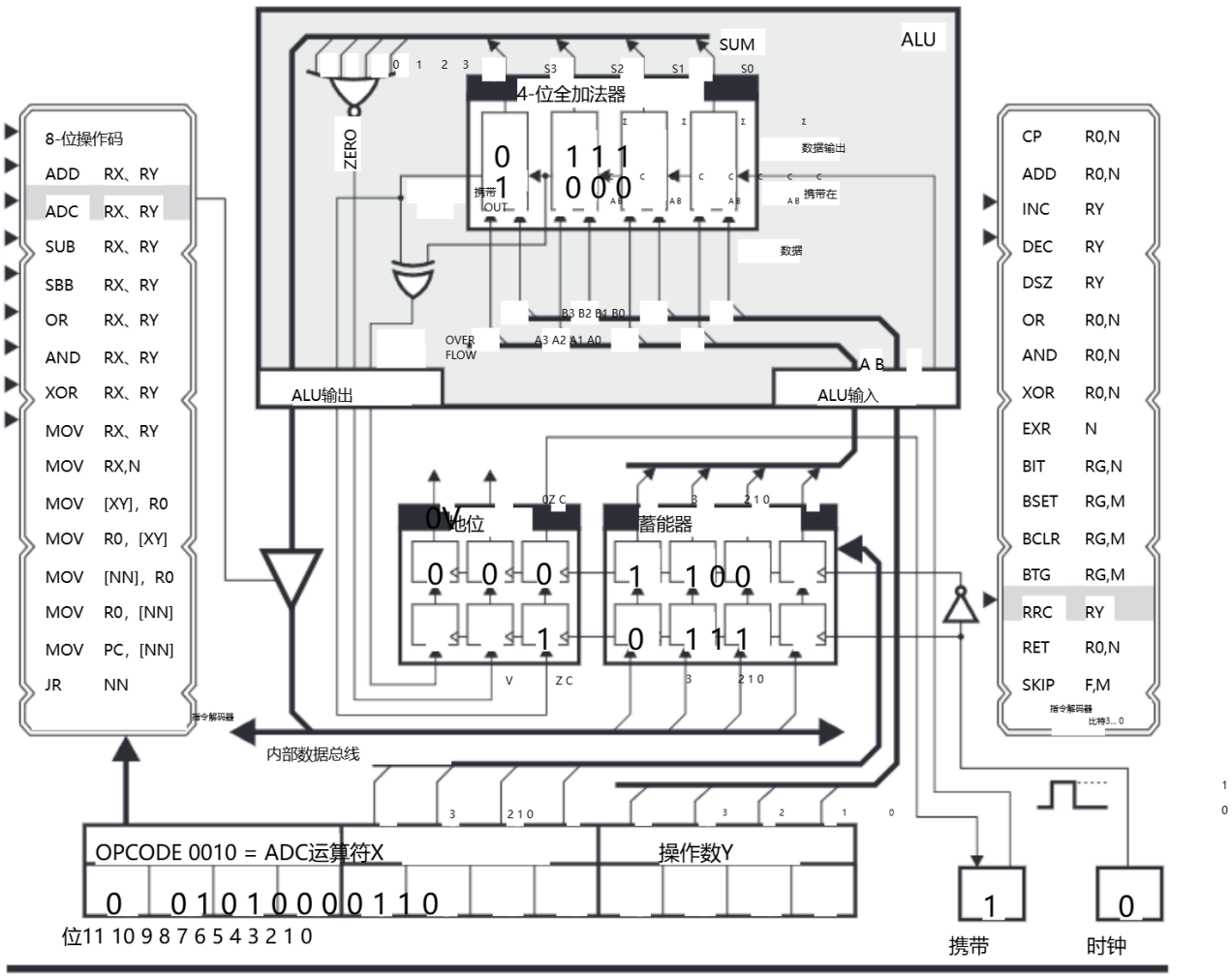
产品描述：　用进位将寄存器Y的内容与寄存器X的内容相加，并将结果放入寄存器X。

受影响的旗帜：　如果有溢出（如果X + Y +进位> 15），设置C。
否则，重置C。
如果操作后结果= 0000，则设置Z。否则，重置Z。
如果有符号表示存在下溢，则设置V。

编码方式：

“0010”位是ADD X，Y操作码。“0”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：　ADC 8、6（进位设置）



SUB X,Y

Subtract register Y from register X

Syntax: {label} SUB X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X - Y

Description: Subtract the contents of the register Y from the contents of the register X and place the result in the register X.

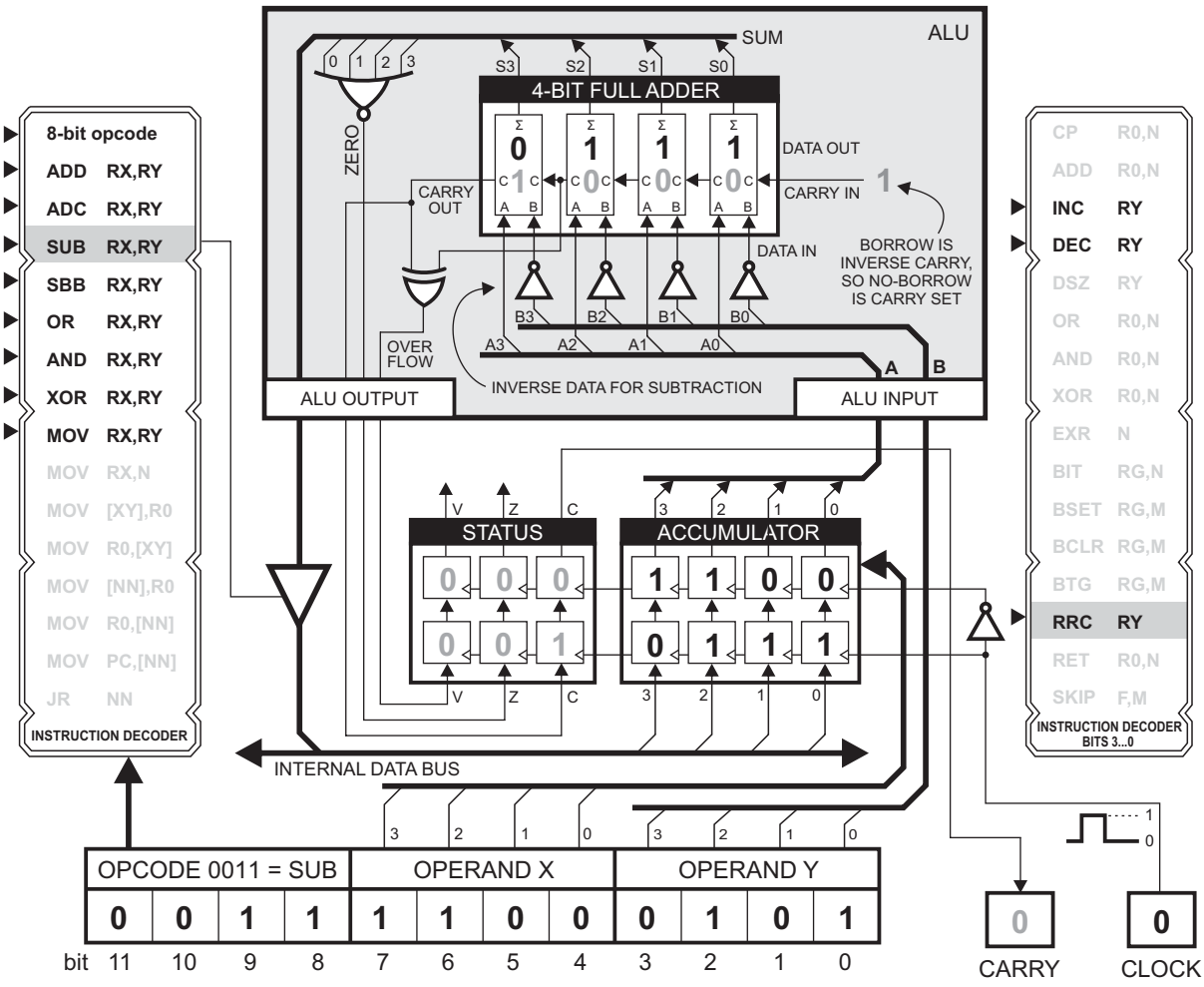
Flags affected: If there is the underflow (if Y < X), reset C.
Otherwise, set C. (note: Borrow is inverse C)
If result = 0000 after operation, set Z. Otherwise, reset Z.
If there is the underflow for signed representation, set V.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	1	1	X	X	X	X	Y	Y	Y	Y

The "0011" bits are the SUB X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: SUB 12, 5



X，Y _

从寄存器X中减去寄存器Y

语法：

{label} X，Y

操作数：

X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：

X ← X - Y

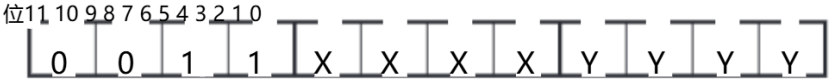
产品描述：

从寄存器X的内容中减去寄存器Y的内容，并将结果放入寄存器X中。

受影响的旗帜：

如果存在下溢（如果Y < X），则重置C。
否则，设置C。(note：Borrow是逆C）如果运算后result = 0000，则设置Z。否则，重置Z。
如果有符号表示存在下溢，则设置V。

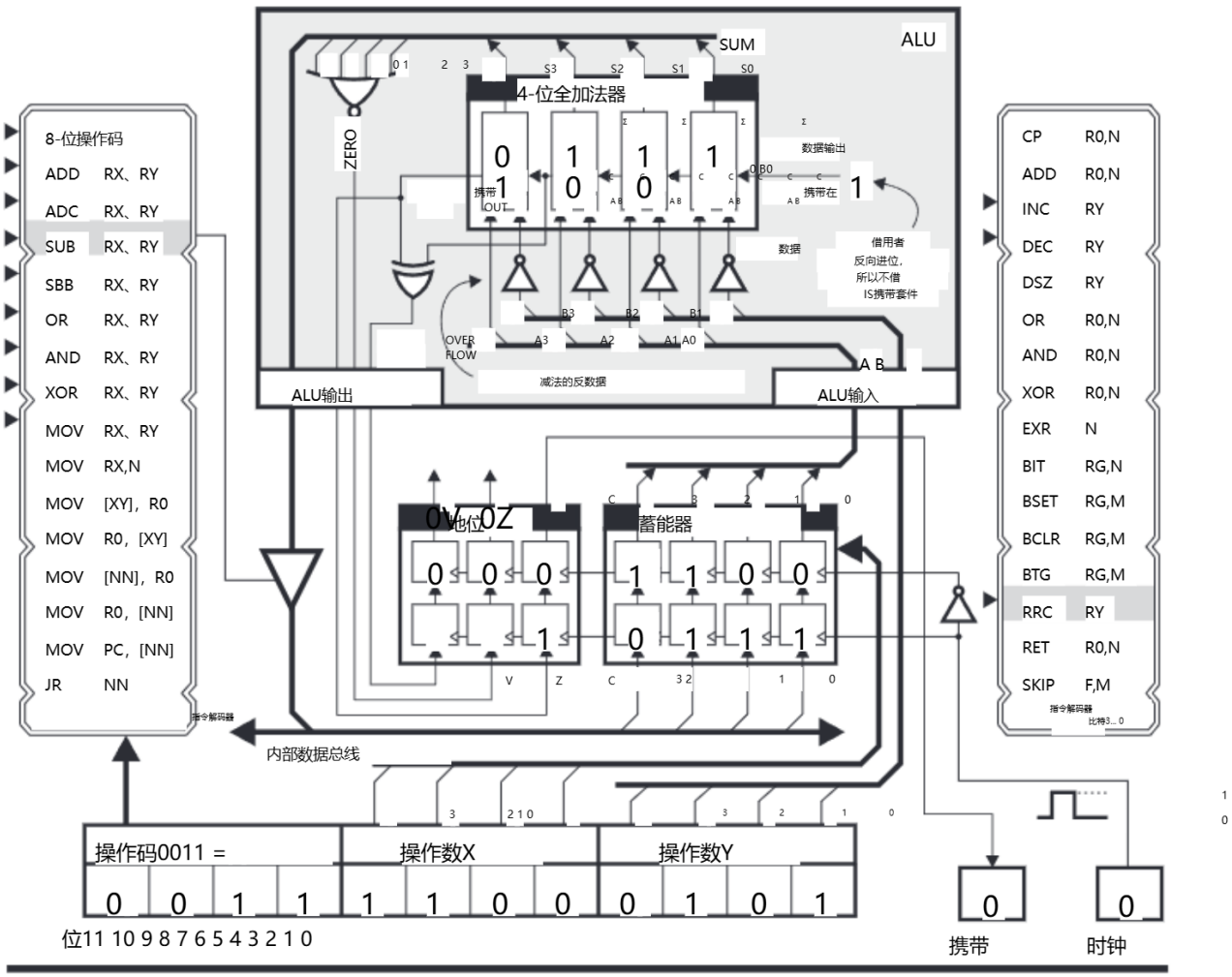
编码方式：



“0011”位是X，Y操作码。“X”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：

2012年，5



SBB X,Y

Subtract register Y from register X with borrow

Syntax: {label} SBB X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: $X \leftarrow X - Y - \overline{Carry}$

Description: Subtract the contents of the register Y and inverse Carry flag from the contents of the register X and place the result in the register X.

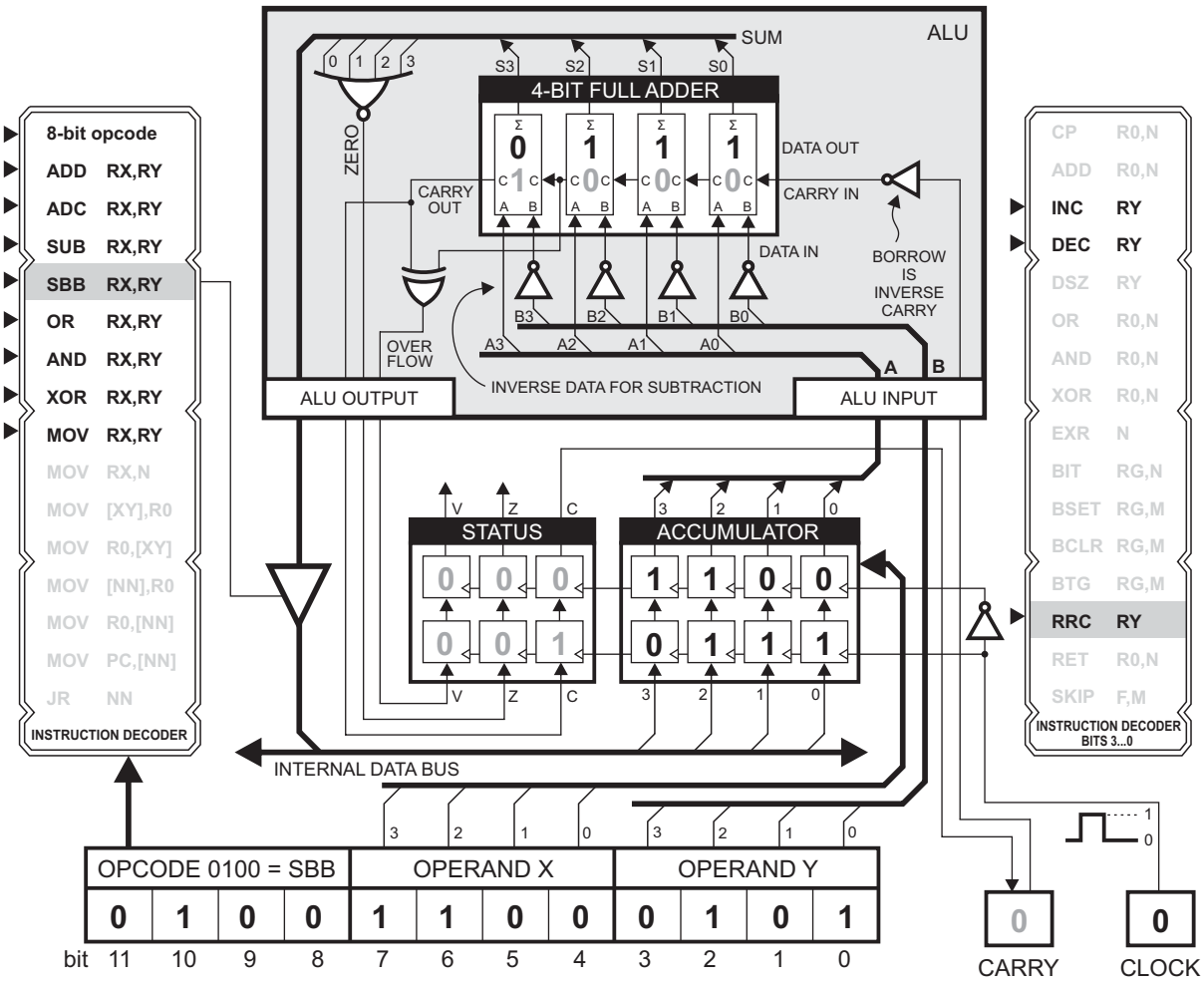
Flags affected: If there is the underflow (if Y < X), reset C.
Otherwise, set C. (note: Borrow is inverse C)
If result = 0000 after operation, set Z. Otherwise, reset Z.
If there is the underflow for signed representation, set V.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	1	0	0	X	X	X	X	Y	Y	Y	Y

The "0100" bits are the SbB X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: SBB 12, 5



SBB X，Y

用借位从寄存器X中减去寄存器Y

语法：

{label} SBB X，Y

操作数：

X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：

X ← X - Y -进位

产品描述：

从寄存器X的内容中减去寄存器Y和反向进位标志的内容，并将结果放入寄存器X。

受影响的旗帜：

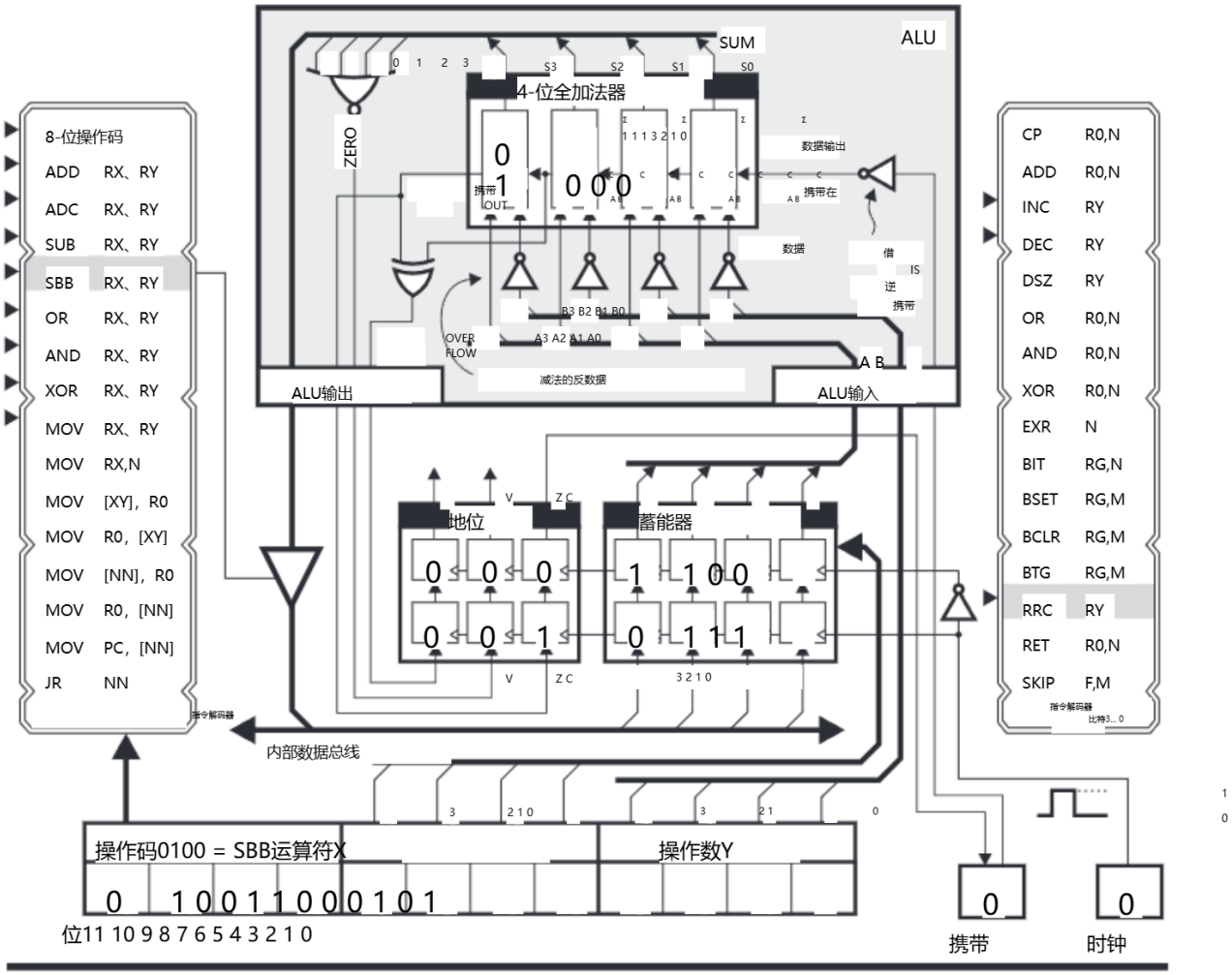
如果存在下溢（如果Y < X），则重置C。
否则，设置C。(note：Borrow是逆C）如果运算后result = 0000，则设置Z。否则，重置Z。
如果有符号表示存在下溢，则设置V。

编码方式：

“0100”位是SbB X，Y操作码。“0”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：

SBB 12、5



OR X,Y

Inclusive OR registers X and Y

Syntax: {label} OR X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X .OR. Y

Description: Compute the logical inclusive OR operation of register X and register Y and place the result into the register X.

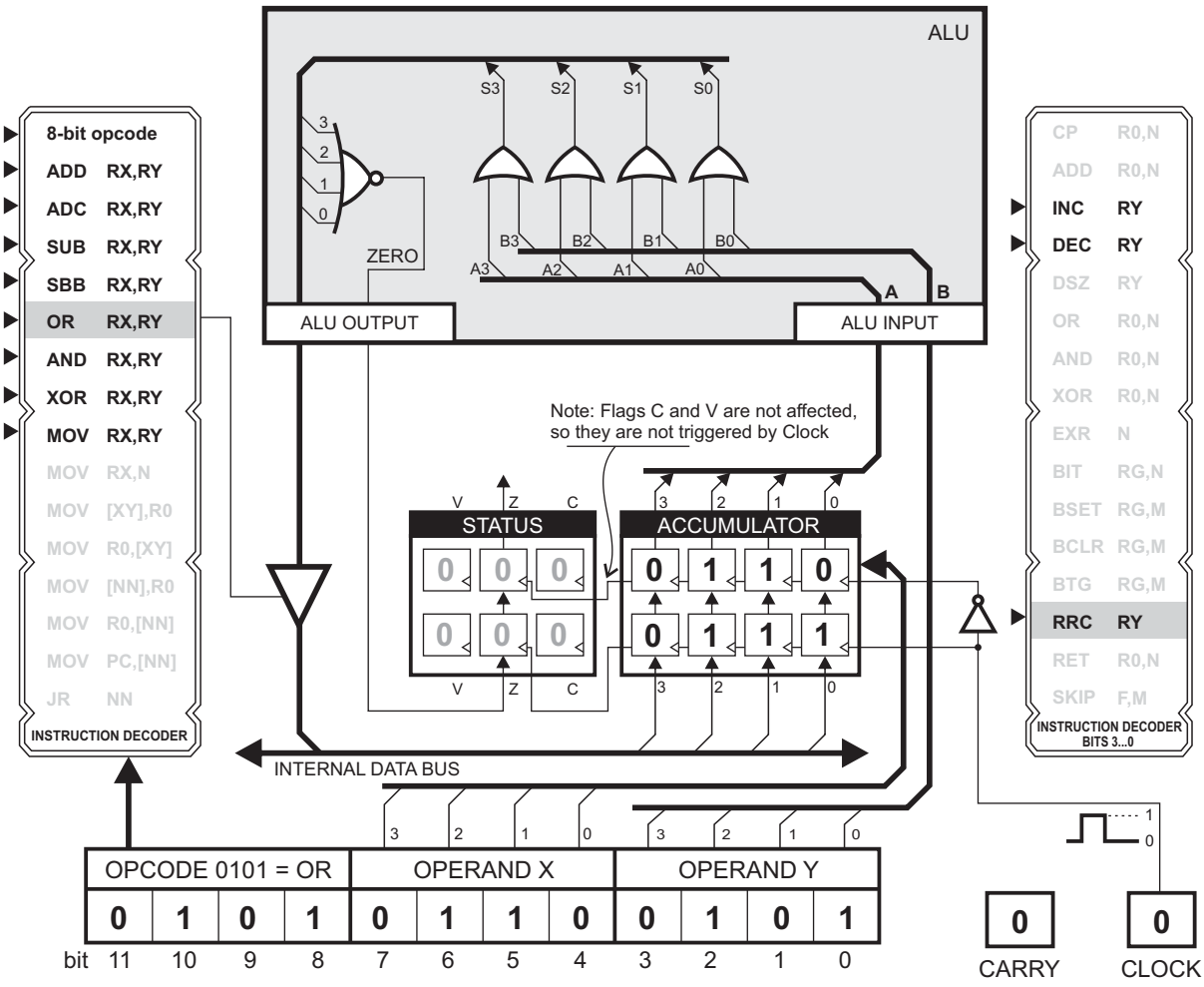
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	1	0	1	X	X	X	X	Y	Y	Y	Y

The "0101" bits are the OR X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: OR 6, 5



或X，Y

包含OR寄存器X和Y

语法：

{label} OR X，Y

操作数：

X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：

X ← X. OR。Y

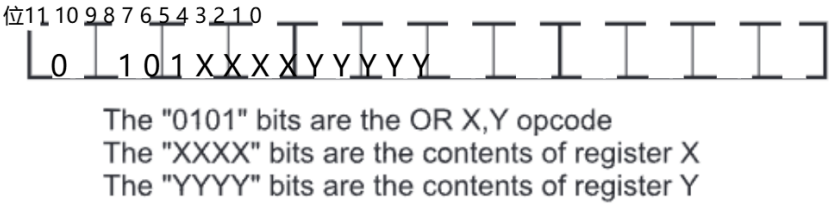
产品描述：

计算寄存器X和寄存器Y的逻辑“或”运算，并将结果放入寄存器X。

受影响的旗帜：

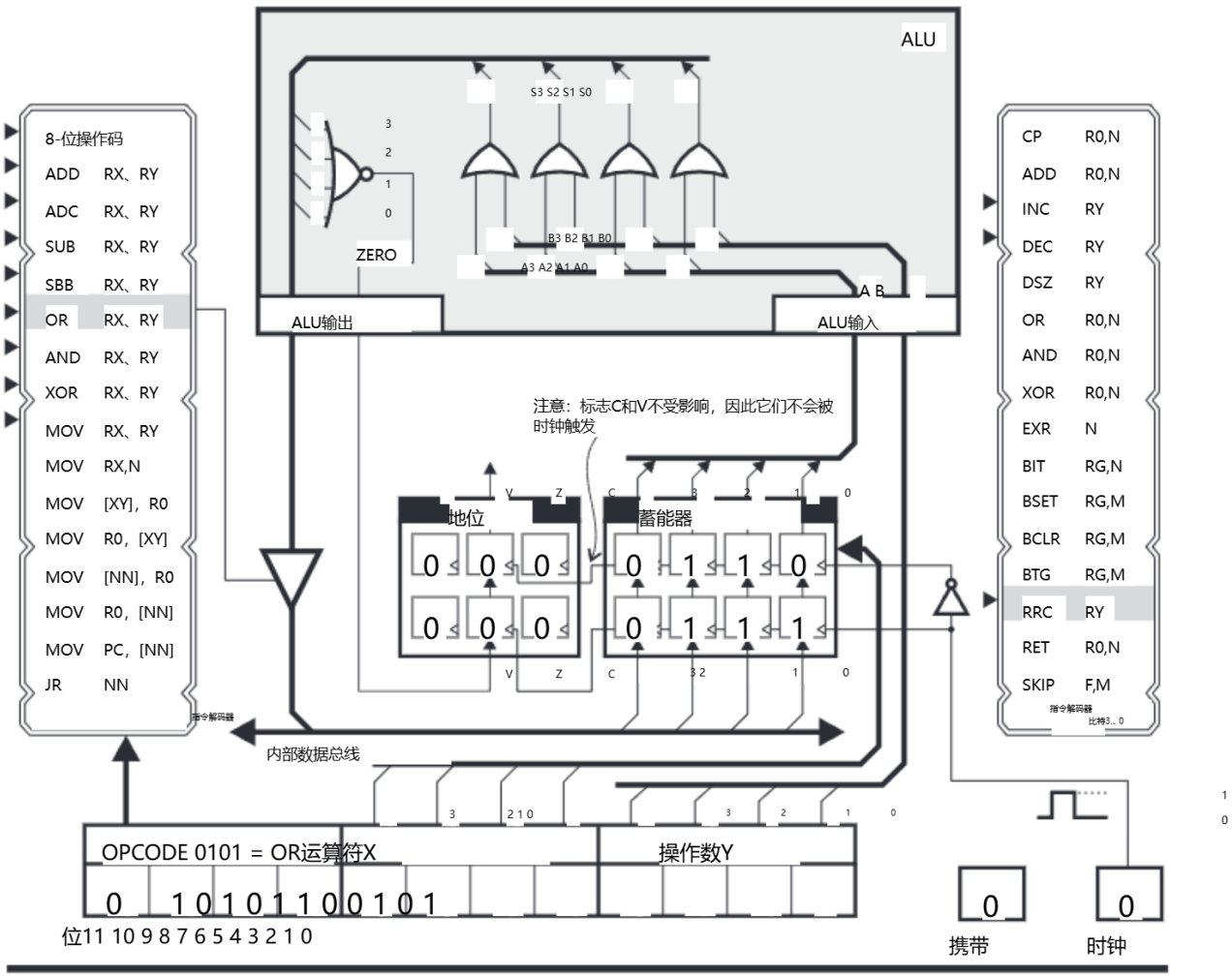
国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

编码方式：



范例：

OR 6、5



AND X,Y

Logical AND registers X and Y

Syntax: {label} AND X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X .AND. Y

Description: Compute the logical inclusive OR operation of register X and register Y and place the result into the register X.

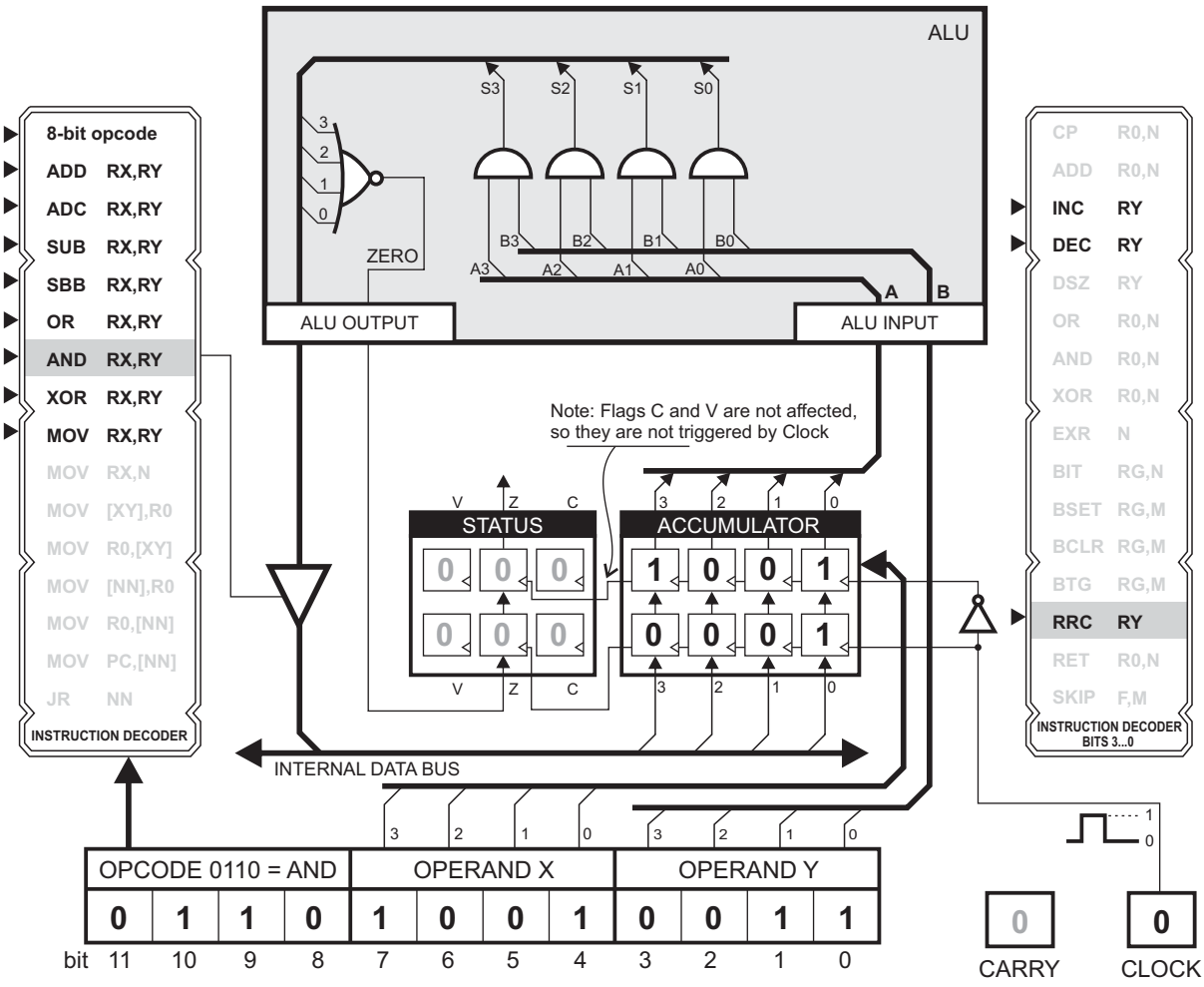
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	1	1	0	X	X	X	X	Y	Y	Y	Y

The "0110" bits are the AND X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: AND 9, 3



并且乂

逻辑与寄存器X和Y

语法：{label} AND X，Y

操作数：X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：X ← X. AND. Y

产品描述：计算寄存器X和寄存器Y的逻辑“或”运算，并将结果放入寄存器X。

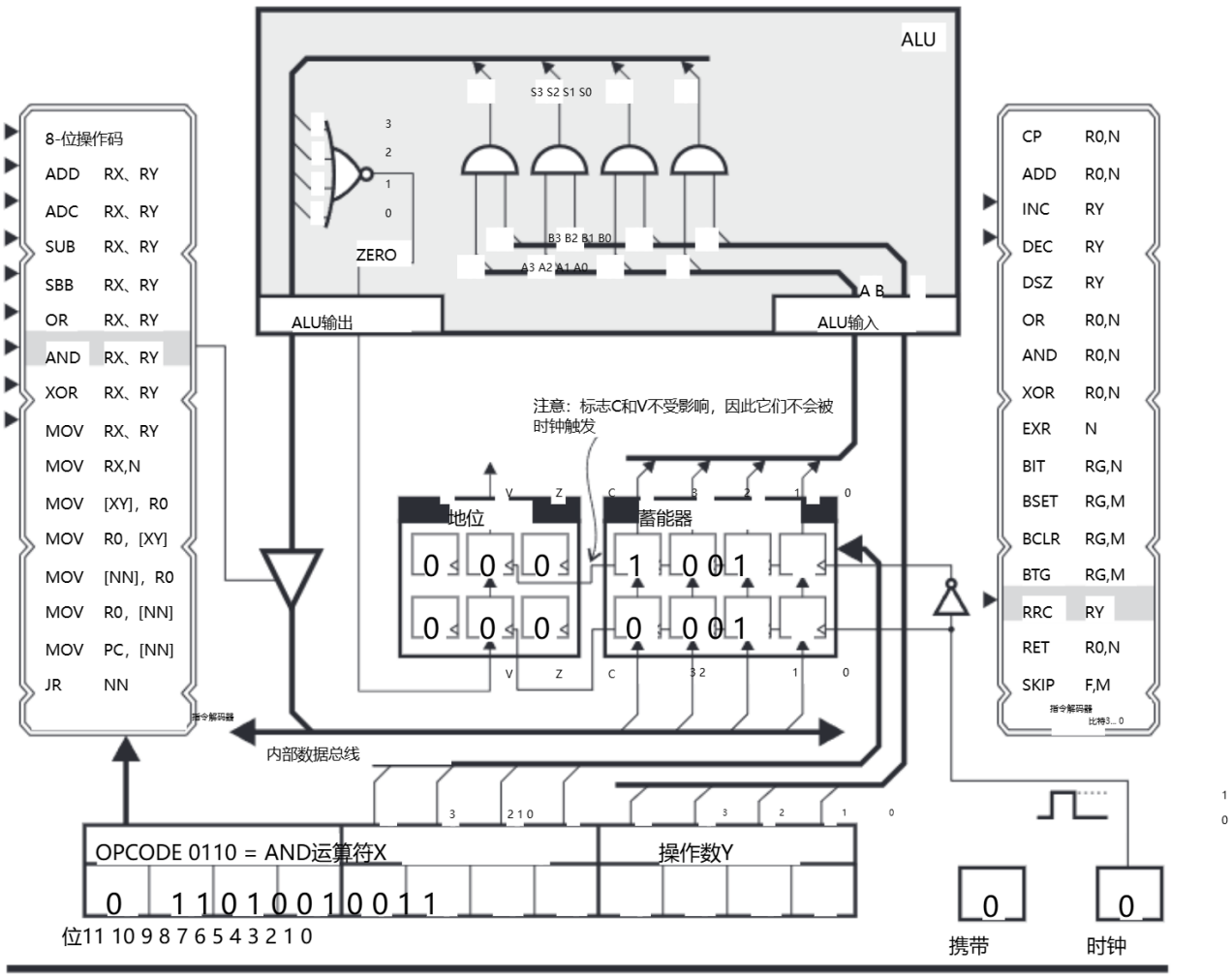
受影响的旗帜：国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

编码方式：

位11 10 9 8 7 6 5 4 3 2 1 0
0 1 1 0 X X X X Y Y Y Y

“0110”位是AND X，Y操作码。“0110”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：和9、3



XOR X,Y

Exclusive OR registers X and Y

Syntax: {label} XOR X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← X .OR. Y

Description: Compute the logical exclusive OR operation of register X and register Y and place the result into the register X.

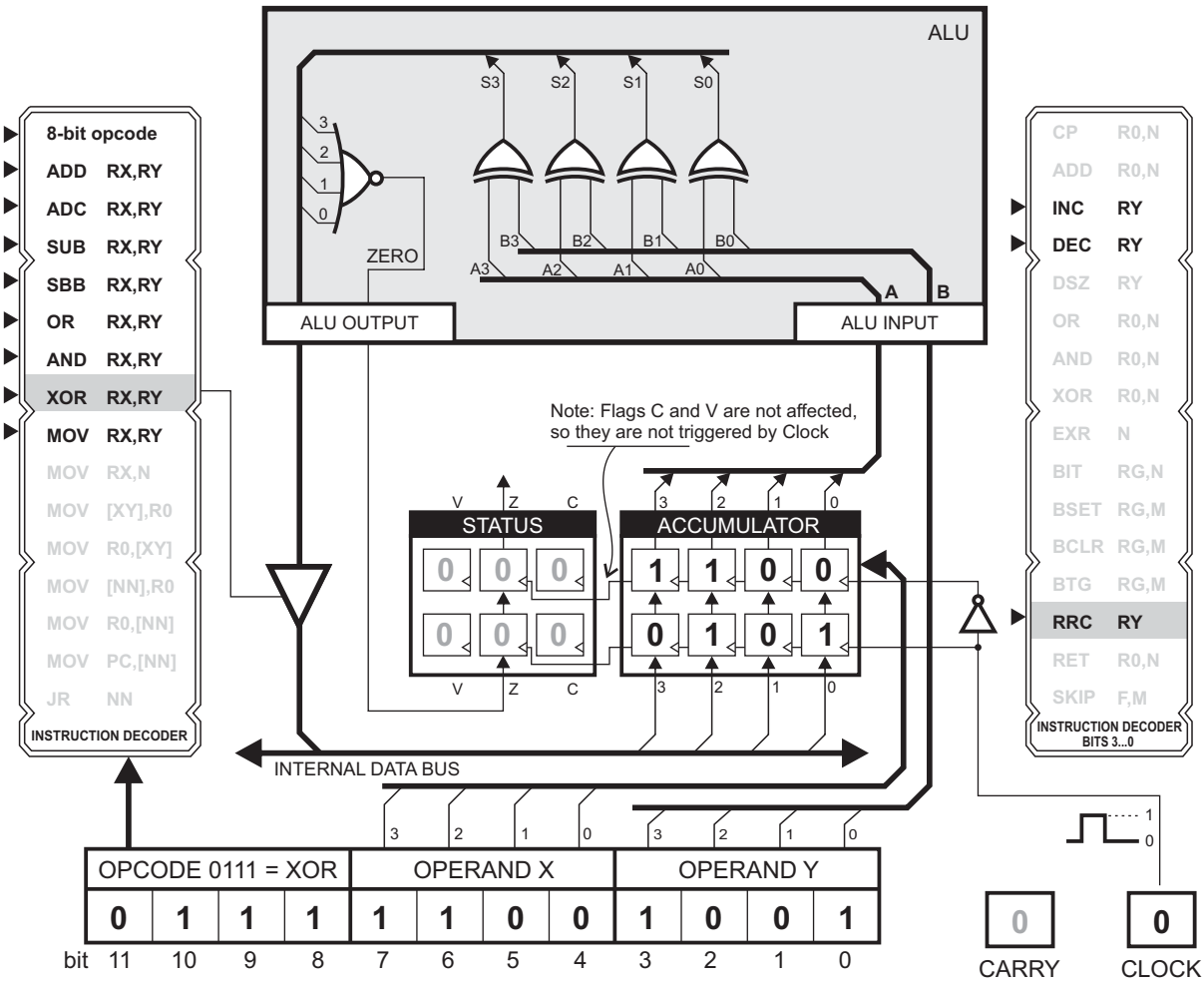
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	1	1	1	X	X	X	X	Y	Y	Y	Y

The "0111" bits are the XOR X,Y opcode
The "XXXX" bits are the contents of register X
The "YYYY" bits are the contents of register Y

Example: XOR 12, 9



XOR X，Y

异或寄存器X和Y

语法：

{label} XOR X，Y

操作数：

X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：

X ← X. OR。Y

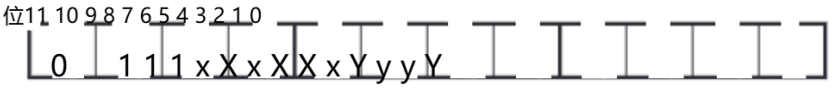
产品描述：

计算寄存器X和寄存器Y的逻辑异或运算，并将结果放入寄存器X。

受影响的旗帜：

国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

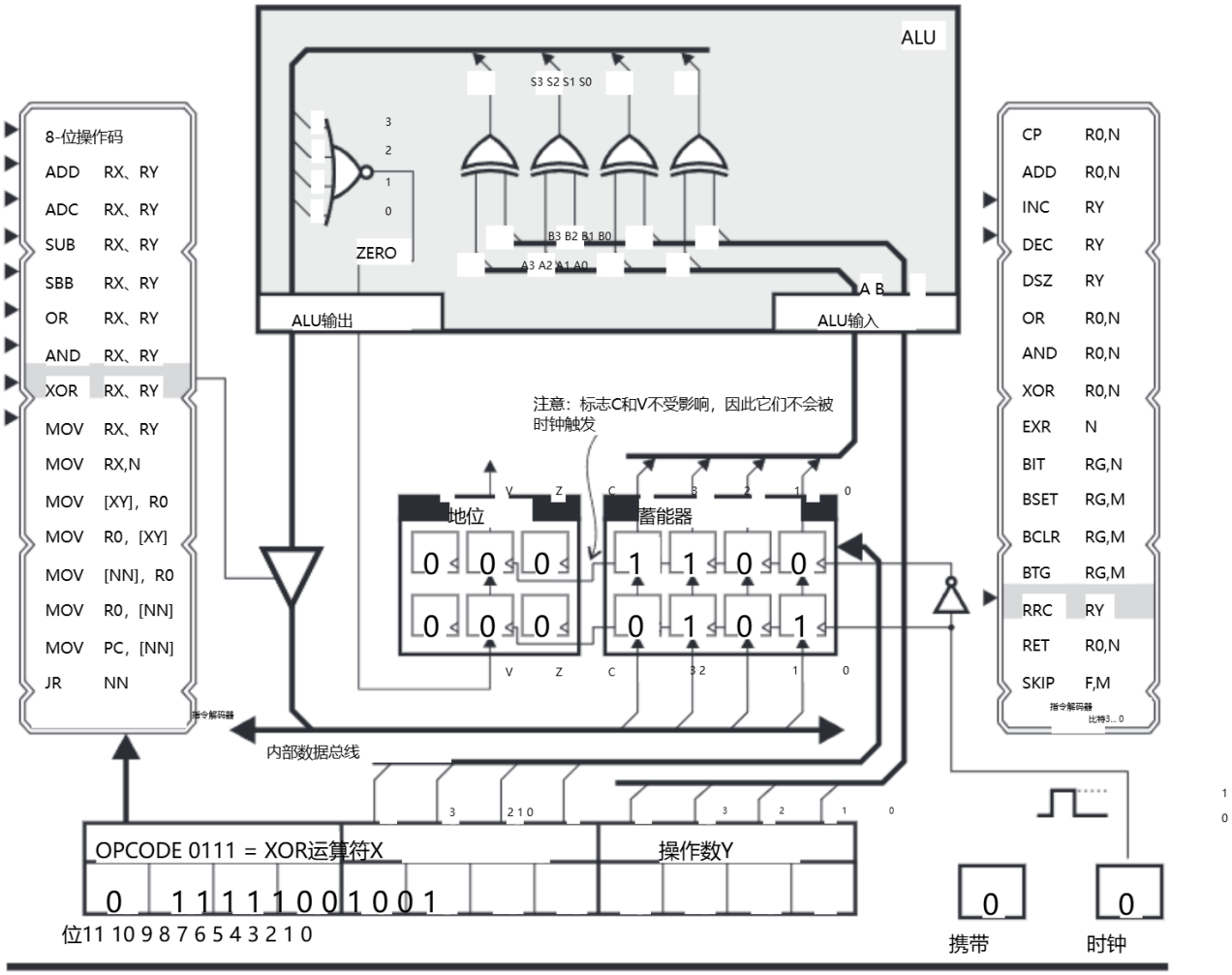
编码方式：



“0111”位是XOR X，Y操作码。“0111”位是寄存器X的内容。“YYYY”位是寄存器Y的内容。

范例：

XOR 12、9



MOV X,Y

Move contents of register Y to register X

Syntax: {label} MOV X, Y

Operands: X ∈ #0...#15
Y ∈ #0...#15

Operation: X ← Y

Description: Move the contents of the register Y to register X. Value of the register Y is unchanged.

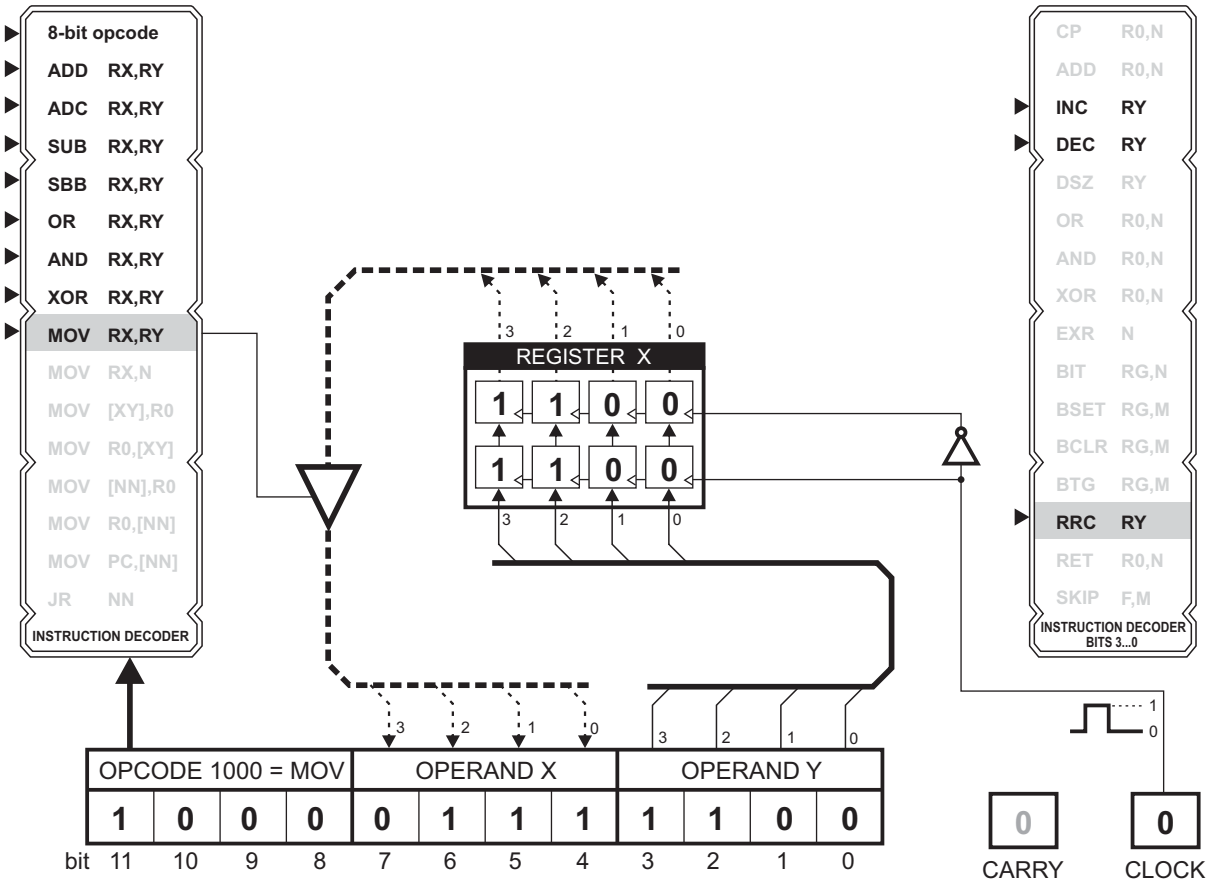
Flags affected: None.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	1	0	0	0	X	X	X	X	Y	Y	Y	Y

The "1000" bits are the MOV X,Y opcode
The "XXXX" bits are the old contents of register X
The "YYYY" bits are the contents of register Y

Example: MOV 7, 12



MOV X，Y　将寄存器Y的内容移动到寄存器X

语法：　{label} MOV X，Y

操作数：　X ∈ #0...# 15
Y ∈ #0...# 15

操作方式：　X ← Y

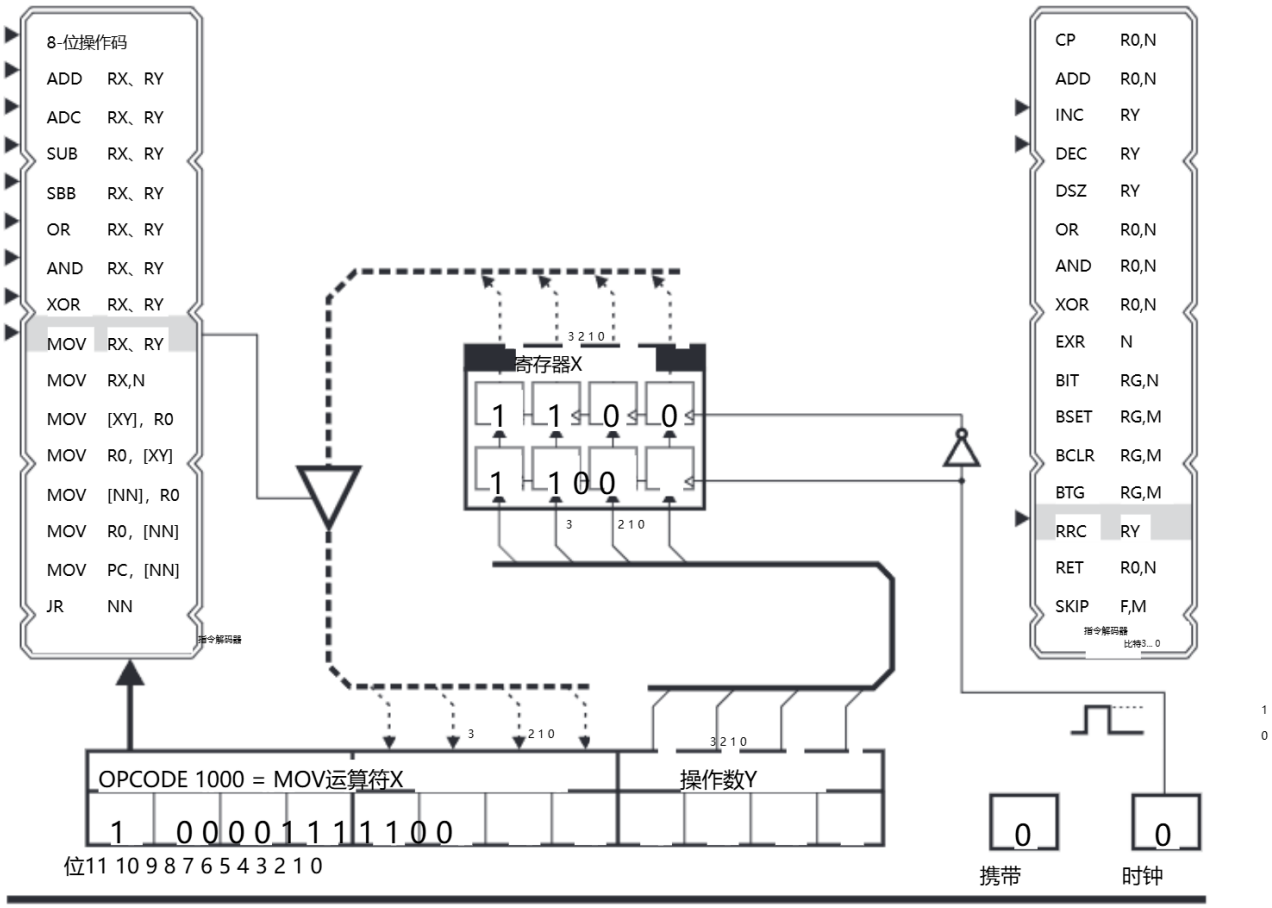
产品描述：　将寄存器Y的内容移至寄存器X。寄存器Y的值不变。

受影响的旗帜：　一个也没有。

编码方式：

“1000”位是MOV X，Y操作码。“1000”位是寄存器X的旧内容。“YYYY”位是寄存器Y的内容。

范例：　MOV 7，12



INC Y

Increment the value of register Y

Syntax: {label} INC Y

Operands: Y ∈ #0...#15

Operation: Y ← Y + 1

Description: Add 1 to the contents of the 4-bit register Y and place the result back into the register Y.

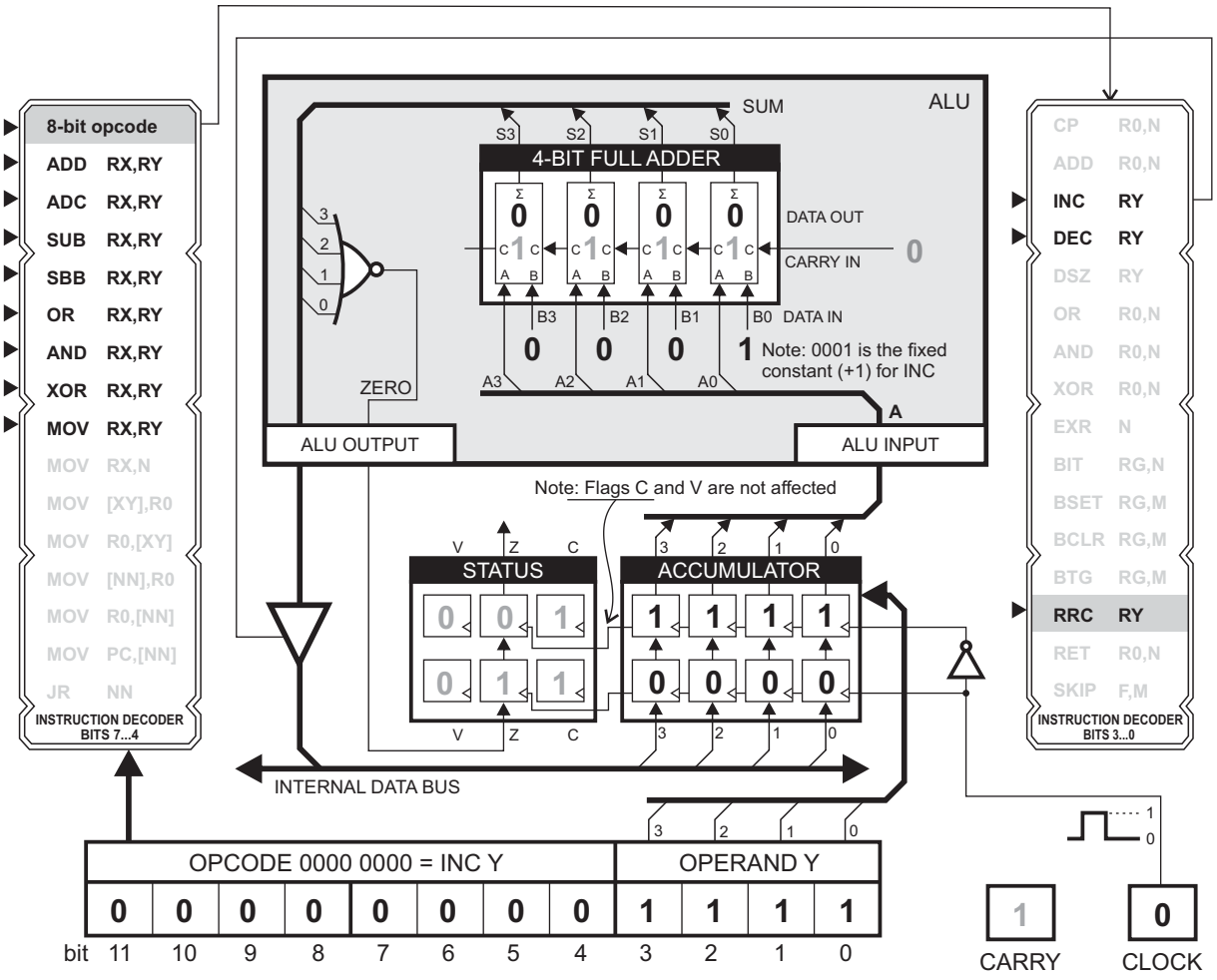
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0	Y	Y	Y	Y

The "0000 0000" bits are the INC Y opcode
The "YYYY" bits are the contents of register Y

Example: INC 15



INC Y

递增寄存器Y的值

语法: {label} INC Y

操作数: Y ∈ #0...# 15

操作方式: Y ← Y + 1

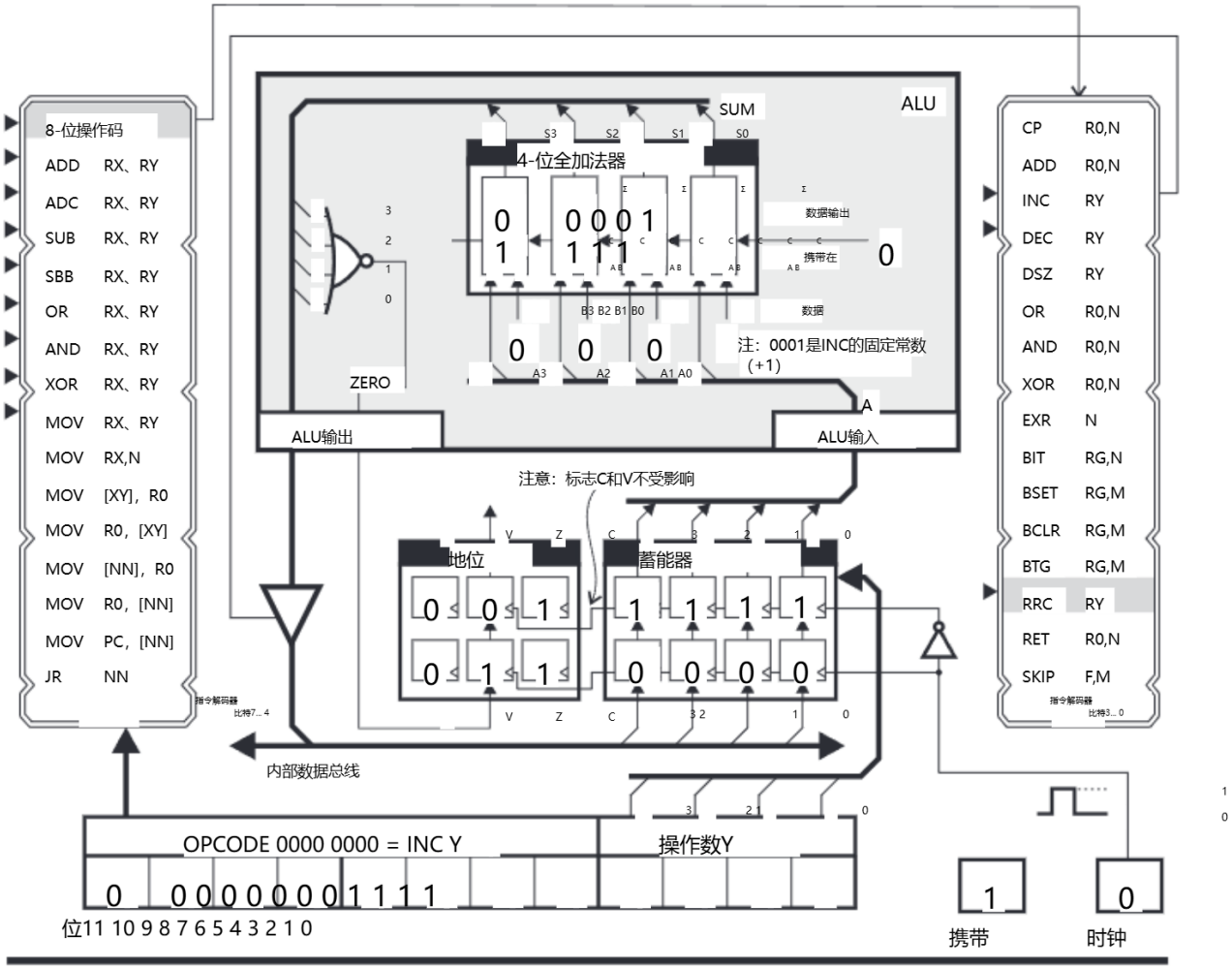
产品描述: 将4位寄存器Y的内容加1，并将结果放回寄存器Y。

受影响的旗帜: 国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

编码方式:

“0000 0000”位是INC Y操作码。“YYYY”位是寄存器Y的内容

范例: INC 15



DEC Y

Decrement the value of register Y

Syntax: {label} DEC Y

Operands: Y ∈ #0...#15

Operation: Y ← Y - 1

Description: Add -1 to the contents of the 4-bit register Y and place the result back into the register Y.

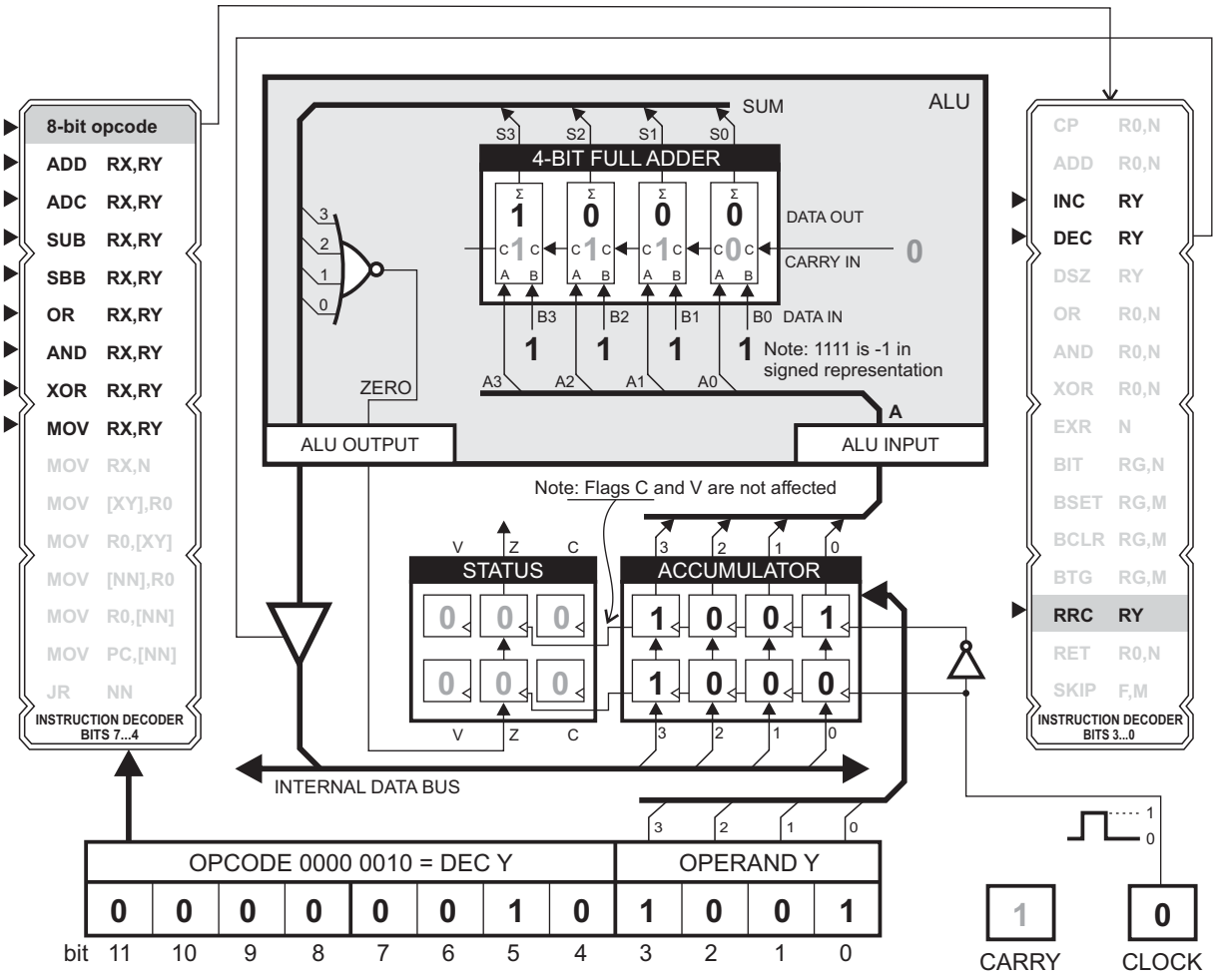
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	1	Y	Y	Y	Y

The "0000 0001" bits are the DEC Y opcode
The "YYYY" bits are the contents of register Y

Example: DEC 9



DEC Y

递减寄存器Y的值

语法： {label} DEC Y

操作数： Y ∈ #0...# 15

操作方式： Y ← Y - 1

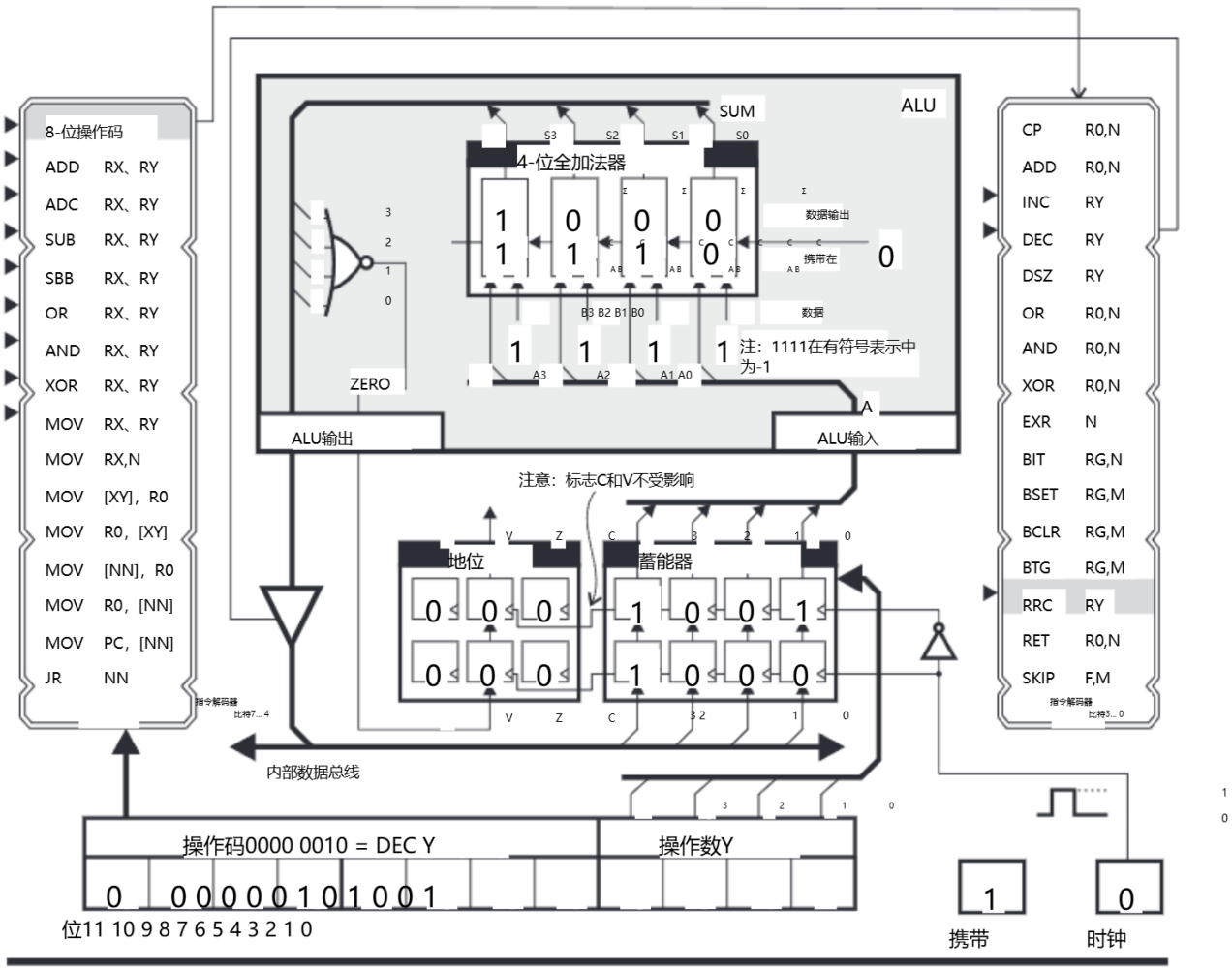
产品描述： 将-1加到4位寄存器Y的内容中，并将结果放回寄存器Y。

受影响的旗帜： 国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

编码方式：

“0000 0001”位是DEC Y操作码。“YYYY”位是寄存器Y的内容

范例： 12月9日



RRC Y

Rotate right through Carry the value of register Y

Syntax: {label} RRC Y

Operands: Y ∈ #0...#15

Operation: C ← Y0, Y3 ← C, Y2 ← Y3, Y1 ← Y2, Y0 ← Y1

Description: Rotate the contents of the register Y one bit to the right through Carry and place the result back in the register Y. The Carry flag is shifted into the Bit 7 of register Y, and Carry is overwritten with the Bit 0 of register Y.

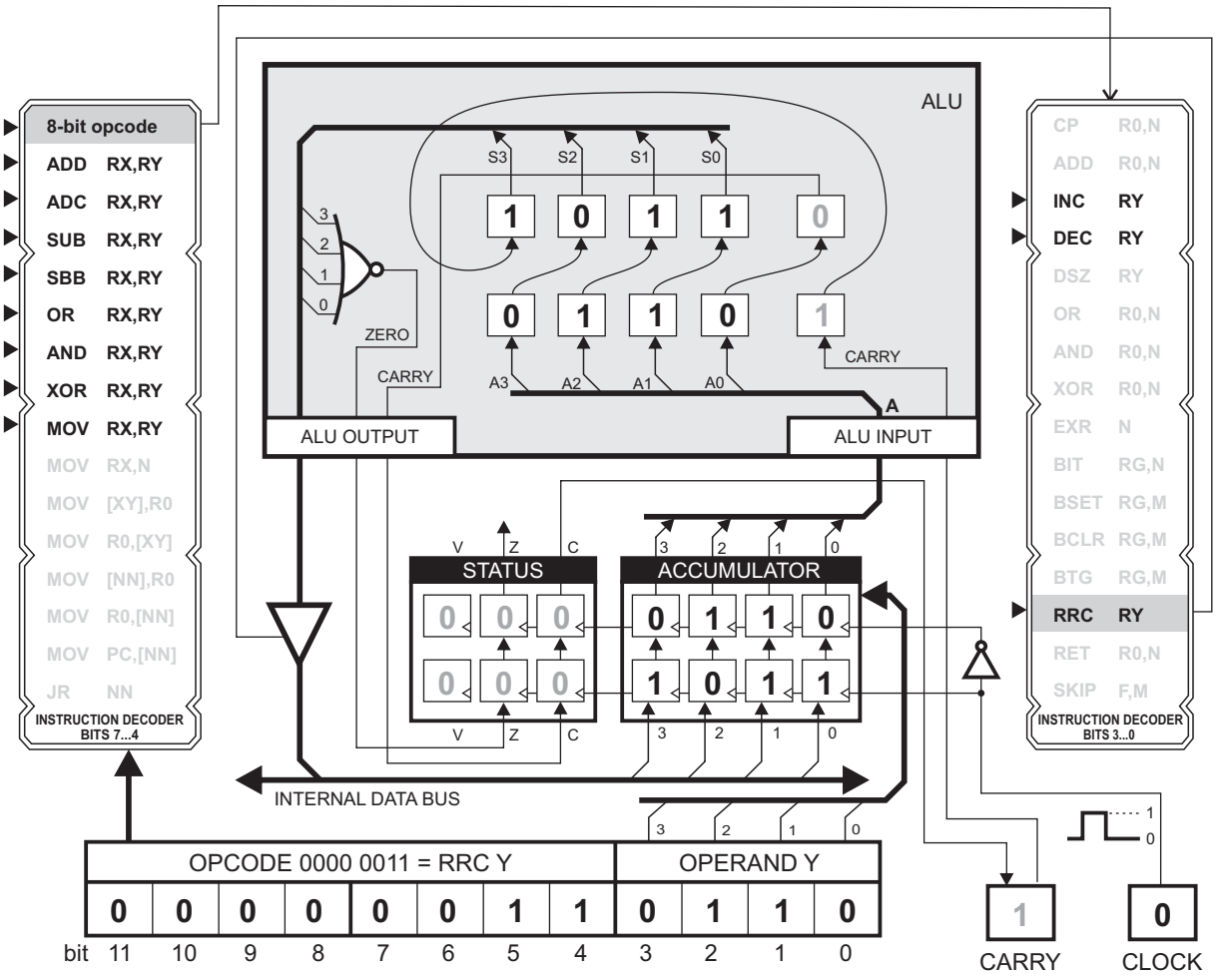
Flags affected: Flag C is not affected.
If result = 0000 after operation, set Z. Otherwise, reset Z.

Encoding:

bit	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	1	0	Y	Y	Y	Y

The "0000 0010" bits are the RRC Y opcode
The "YYYY" bits are the contents of register Y

Example: RRC 6



RRC Y

向右旋转通过进位寄存器Y的值

语法: {label} RRC Y

操作数: Y ∈ #0...# 15

操作方式: C ← Y0，Y3 ← C，Y2 ← Y3，Y1 ← Y2，Y0 ← Y1

产品描述: 通过进位将寄存器Y的内容向右旋转一位，并将结果放回寄存器Y。进位标志移位到寄存器Y的位7，进位被寄存器Y的位0覆盖。

受影响的旗帜: 国旗C不受影响。
如果操作后结果= 0000，则设置Z。否则，重置Z。

编码方式:

“0000 0010”位是RRC Y操作码。“YYYY”位是寄存器Y的内容。

范例: RRC 6