Exercício

- Projete um processador dedicado que gera em sua saída os n primeiros elementos da sequência de Fibonacci.

- Sequência de Fibonacci: 1, 1, 2, 3, 5, 8, 13, 21, ...

$$a_n = \begin{cases} 1, n = 0 \\ 1, n = 1 \\ a_{n-1} + a_{n-2}, \text{alhures} \end{cases}$$

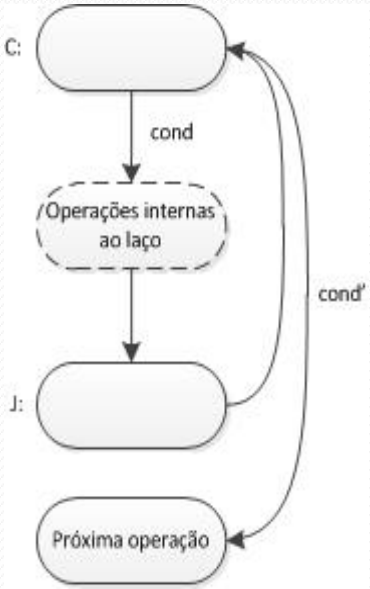
```
int x1, x2, temp, count, n;
while(1){
  while(!go_i);
  x1 = 1;
  x2 = 1;
  n = n_i;
  count = 0;
  while(count < n){
    if(count != 0 && count !=1){
      temp = x1;
      x1 = x2;
      x2 = x1 + temp;
    }
    fib_o = x2;
    count ++;
  }
}
```

Projeto de Processadores Dedicados

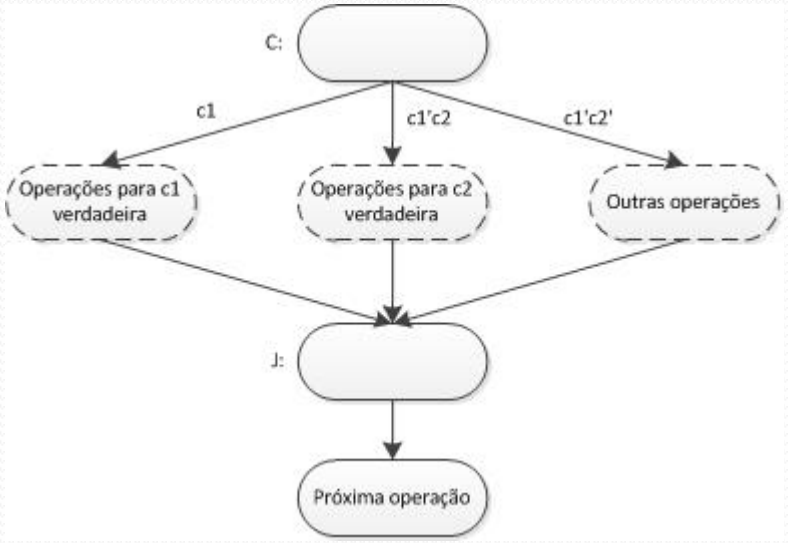
Atribuição de valor a uma variável


```
int x1, x2, temp, count, n;
while(1){
  while(!go_i);
  x1 = 1;
  x2 = 1;
  n = n_i;
  count = 0;
  while(count < n){
    if(count != 0 && count !=1){
      temp = x1;
      x1 = x2;
      x2 = x1 + temp;
    }
    fib_o = x2;
    count ++;
  }
}
```

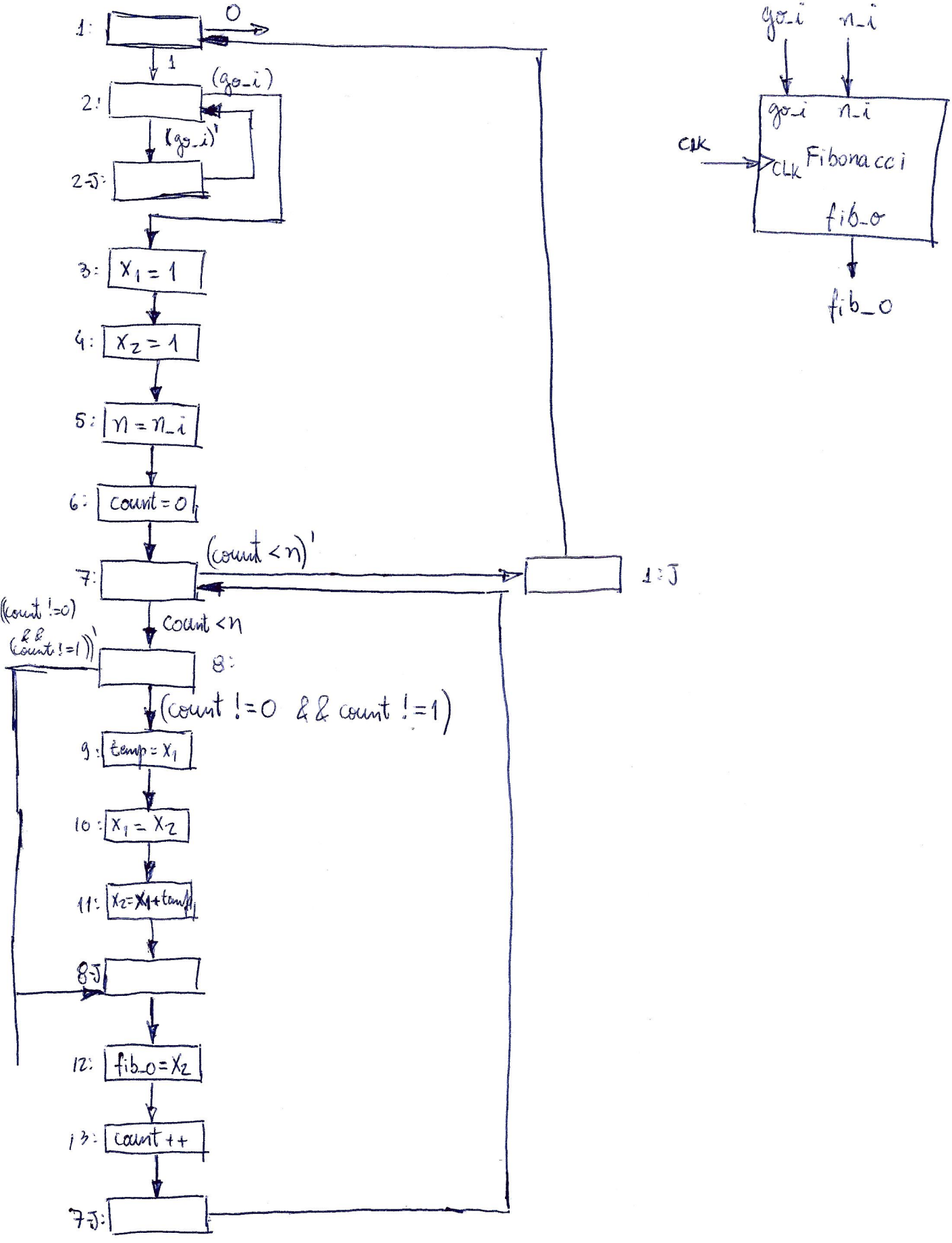
Laço de repetição



Desvio condicional



FSMD

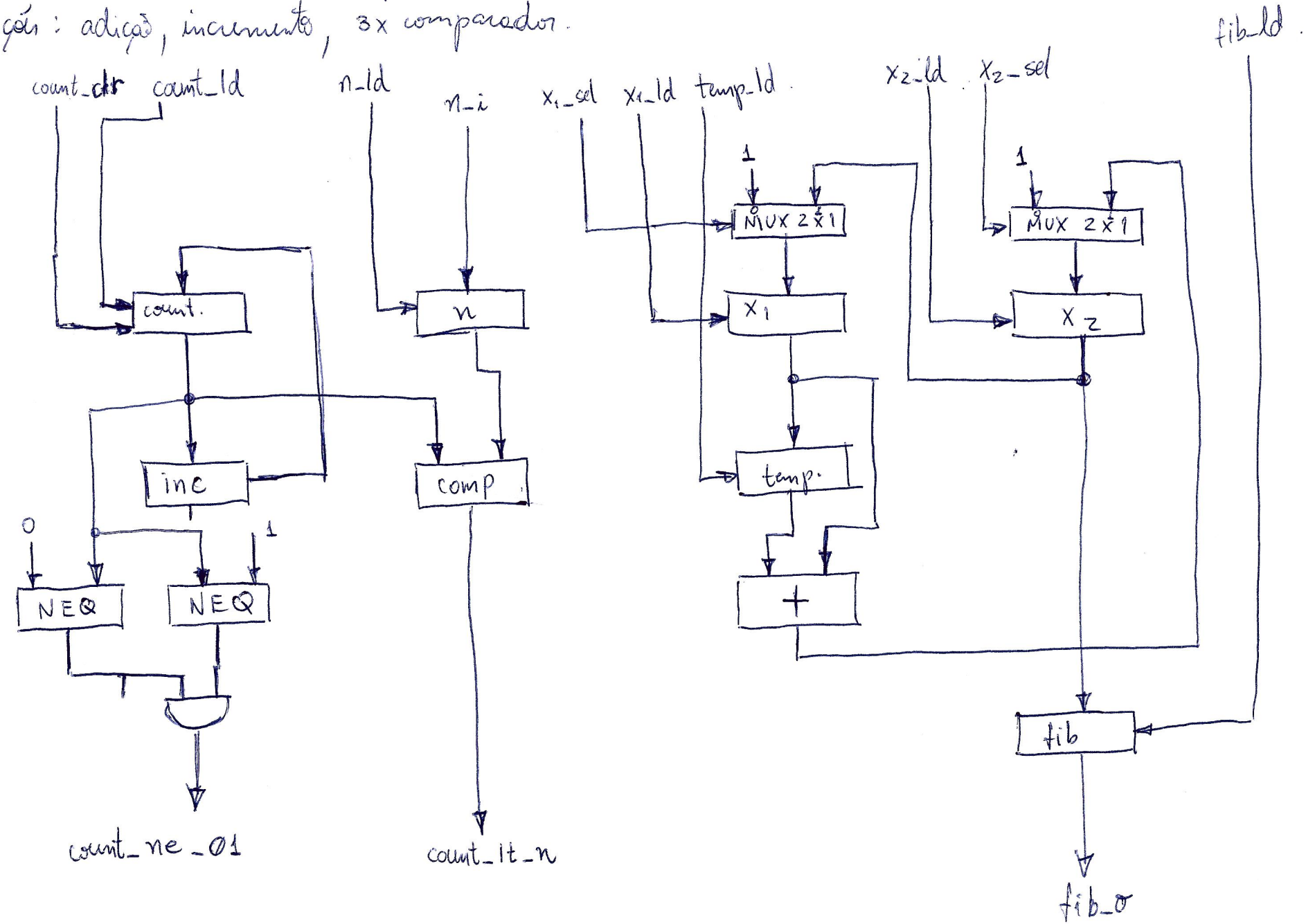

1:
0
1
2:
(go_i)
(go_i)'
2-J:
3:
x1 = 1
4:
x2 = 1
5:
n = n_i
6:
count = 0
7:
(count < n)'
1:J
count < n
((count != 0 && count != 1))'
8:
(count != 0 && count != 1)
9:
temp = x1
10:
x1 = x2
11:
x2 = x1 + temp
8-J
12:
fib_o = x2
13:
count ++
7-J:

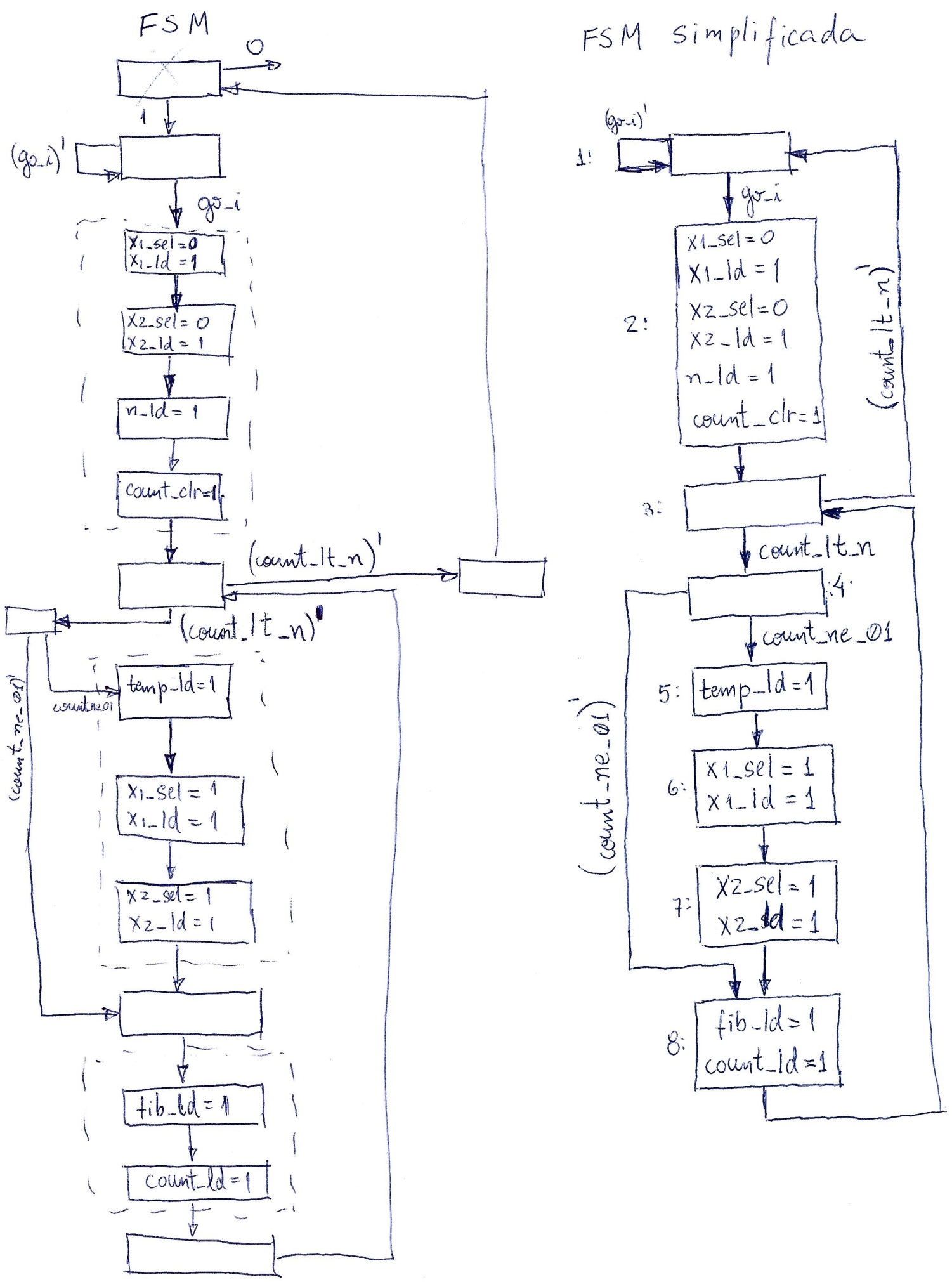
go_i
n_i
go_i
n_i
clk
clk
Fibonacci
fib_o
fib_o

Datapath:

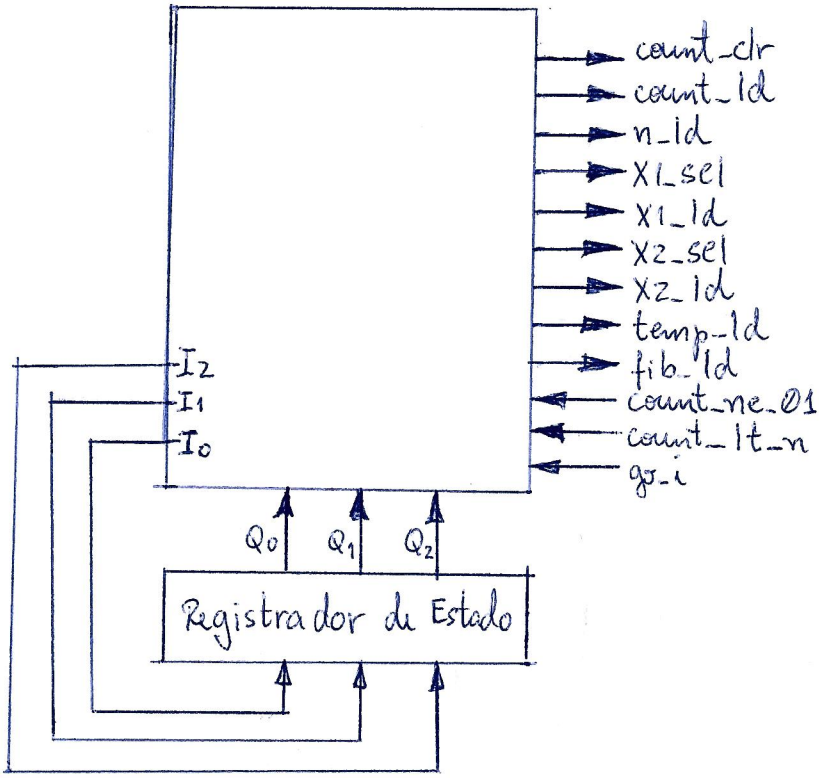
variáveis: x_1, x_2, n, count, temp, fib.

operações: adição, incremento, 3x comparador.




FSM
0
1
(go_i)'
go_i
x1_sel=0
x1_ld=1
x2_sel=0
x2_ld=1
n_ld=1
count_clr=1
(count_lt_n)'
(count_lt_n)'
(count_ne_01)'
count_ne_01
temp_ld=1
x1_sel=1
x1_ld=1
x2_sel=1
x2_ld=1
fib_ld=1
count_ld=1
FSM simplificada
(go_i)'
1:
go_i
2:
x1_sel=0
x1_ld=1
x2_sel=0
x2_ld=1
n_ld=1
count_clr=1
(count_lt_n)'
3:
count_lt_n
4:
count_ne_01
5:
temp_ld=1
6:
x1_sel=1
x1_ld=1
7:
x2_sel=1
x2_ld=1
(count_ne_01)'
8:
fib_ld=1
count_ld=1

Controlador



Projeto de Processadores Dedicados

count_ne_01	count_lt_n	go_i	Q2	Q1	Q0	I2	I1	I0	count_clr	count_ld	n_ld	x1_sel	x1_ld	x2_sel	x2_ld	temp_ld	fib_ld
*	*	0	0	0	0	0	0	0	0	0	0	X	0	X	0	0	0
*	*	1	0	0	0	0	0	1	0	0	0	X	0	X	0	0	0
*	*	*	0	0	1	0	1	0	1	0	1	0	1	0	1	0	0
*	0	*	0	1	0	0	0	0	0	0	0	X	0	X	0	0	0
*	1	*	0	1	0	0	1	1	0	0	0	X	0	X	0	0	0
0	*	*	0	1	1	1	1	1	0	0	0	X	0	X	0	0	0
1	*	*	0	1	1	1	0	0	0	0	0	X	0	X	0	0	0
*	*	*	1	0	0	1	0	1	0	0	0	X	0	X	0	1	0
*	*	*	1	0	1	1	1	0	0	0	0	1	1	X	0	0	0
*	*	*	1	1	0	1	1	1	0	0	0	X	0	1	1	0	0
*	*	*	1	1	1	0	1	0	0	1	0	X	0	X	0	0	1