

```
i = 0;
a = 0;
do {
    a = a + i;
    i = i + 1;
} while (i < 10);
print( a );
end
```

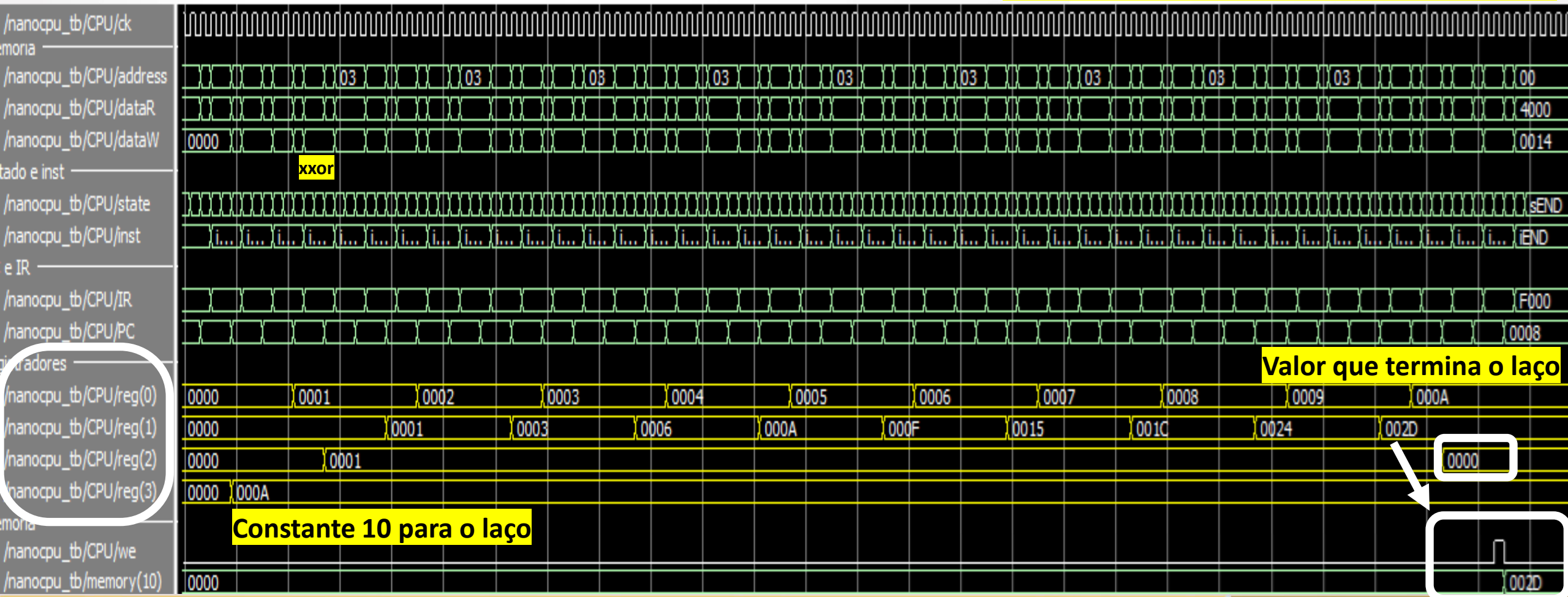
```
signal memory: memory_array :=
(
    0 => X"4000",
    1 => X"4111",
    2 => X"0093",
    3 => X"6110",
    4 => X"8000",
    5 => X"7203",
    6 => X"3032",
    7 => X"10A1",
    8 => X"F000",
    9 => X"000A",
    others => (others => '0')
);
```

Endereço	Instrução - <i>assembly</i>	Comentário	Binário (hexa)
0	<b>iXOR</b> R0, R0, R0	$R0 \leftarrow 0$ (i)	4 0 0 0
1	<b>iXOR</b> R1, R1, R1	$R1 \leftarrow 0$ (a)	4 1 1 1
2	<b>iREAD</b> R3, 9	Lê o valor 10 da posição 9: $R3 \leftarrow M[9]$	0 09 3
3	<b>iADD</b> R1, R1, R0	$a = a + i$	6 1 1 0
4	<b>iINC</b> R0, R0	$R0 \leftarrow R0 + 1$	8 0 0 0
5	<b>iLESS</b> R2, R0, R3	$R2 \leftarrow 1$ se $i < 10$ else 0	7 2 0 3
6	<b>iBRANCH</b> 3, R2	se $R2=1$ salta para 3	3 0 3 2
7	<b>iWRITE</b> R1, 10	Escreve a na posição 10	1 0A 1
8	<b>END</b>	Termina a execução	F000
9	10	Número de vezes que executa o laço	000A
10 (A)		Receberá o valor de a	

## Atividade 6: Executar o programa exemplo (b)

- Observar o **reg(0)** – variável **i** que conta de 0 a 9; termina o laço quando chega a 10
- **reg(1)** – variável **a** que recebe 0, 1, 3, 6, 10, 15, 21(x15), 28 (x1C), 36 (x24), 45 (x2D)
- No final grava na posição 12 da memória o valor 0x2D

**Tempo de simulação : 300 ns**



-- N = 14 já inicializado na memória

```
fib1 = 0;
fib2 = 1;
do {
    MEM[] = fib1;
    next = fib1 + fib2;
    fib1 = fib2;
    fib2 = next;
    N--;
} while(N > 0);
```

Dicas:

- Reservar **R0** como uma constante 0 (zero)
- Dois registradores devem ser dedicados a fib1 (**R1**) e fib2 (**R2**)
- Fazer xor com 0 para atribuição do tipo fib1 = fib2

Exemplo:

X"4000", -- R0 <= 0 (constante)

X"4120", -- R1 <= R2 xor R0 (fib1 <= fib2)

- **R3** – usar para tratar *next*, N (lê N da memória, decrementa N, grava N na memória), e usa para o desvio condicional (R3 = 1 se N < 0) – é o registrador “temporário”

```
signal memory: memory_array := -- Fibonnaci
(
    ...
    20 => x"000E", -- 14 primeiros elementos da série
    21 => x"0000", -- Recebe os valores da série
    others => (others => '0')
);
```

# Atividade 7: Calcular a sequência de Fibonacci (b)

-- N = 14 já inicializado na memória

fib1 = 0;

fib2 = 1;

```
do {
    MEM[] = fib1;
    next = fib1 + fib2;
    fib1 = fib2;
    fib2 = next;
    N--;
} while(N > 0);
```

- **R0**: constante 0
- **R1**: fib1
- **R2**: fib2
- **R3**: temporário

