Graduação em Ciência da Computação/
Engenharia de Software

Disciplina: Arquitetura de Computadores

Professor: José Wilson da Costa
jwcostaprof@gmail.com

Linguagem de Descrição de Hardware : VHDL e Verilog

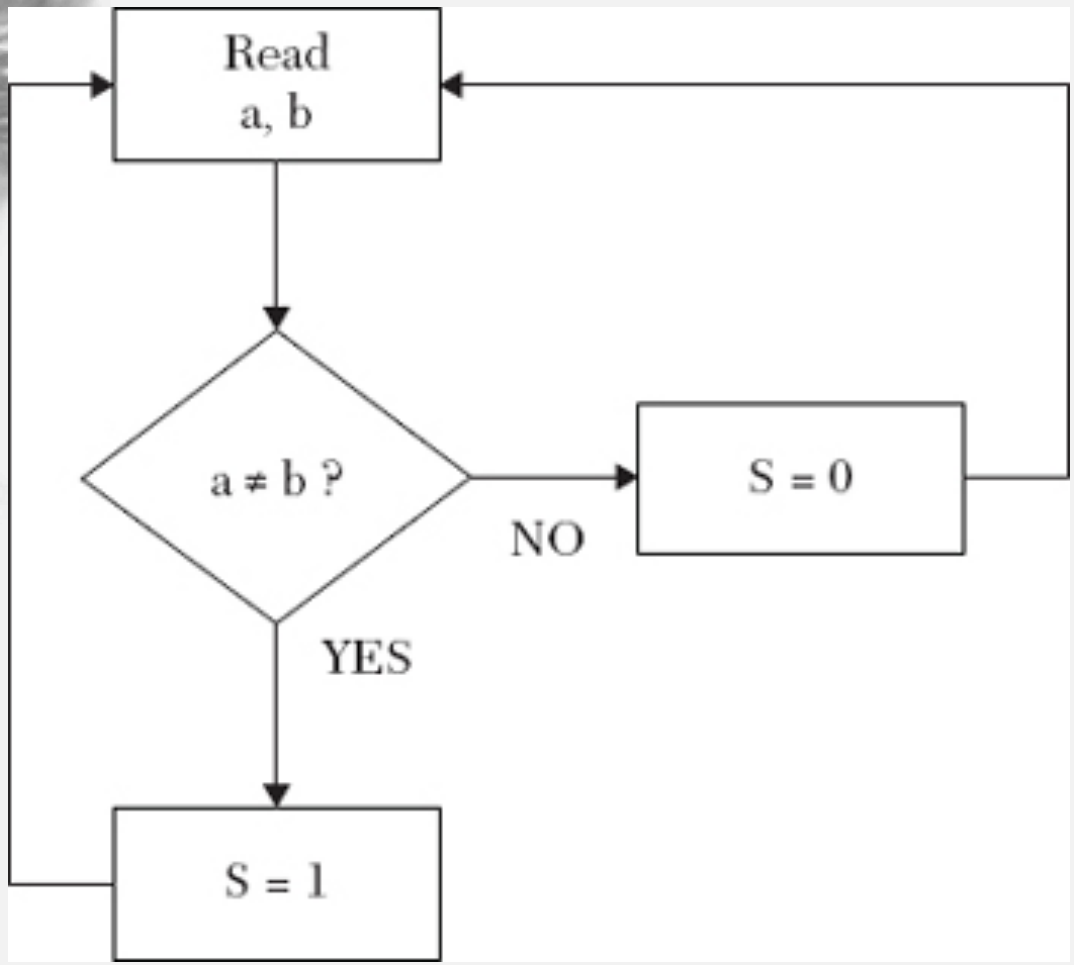
```
module somador(a,b,S,C);
    input a,b;
    output S, C;
    // linhas em branco são permitidas

    assign S = a ^ b; // instrução 1
    assign C= a & b; // instrução 2
Endmodule
```

Linguagem de Descrição de Hardware : Tipos de Descrição

Fluxo de dados : descrevem sinais de saídas em função de sinais de entradas. Normalmente descritos por funções booleanas.

Descrição de comportamentos: descrevem o comportamento das saídas em relação às entradas. Em verilog o module contém a palavra prédefinida always ou initial. Normalmente utilizado quando se é difícil obter a expressão booleana.

Linguagem de Descrição de Hardware : Tipos de Descrição



Linguagem de Descrição de Hardware : Tipos de Descrição

```
module somador(a,b,S,C);
    input a,b;
    output S, C;
    reg S,C;
    always @ (a,b)
    begin
    if (a != b)
    S = 1'b1;
    else
    S = 1'b0;
    end
endmodule
```

Linguagem de Descrição de Hardware : Tipos de Descrição

Descrição estrutural : modela o sistema como componentes ou portas

```
module somador(a,b,S,C);
   input a, b;
   output S,C;
   and a1(C,a,b);
   //porta  AND
   xor x1(S,a,b);
   //porta EXCLUSIVE-OR
endmodule
```

Linguagem de Descrição de Hardware : Tipos de Descrição

Descrição de nível-chave : Descrição de mais baixo nível. O sistema é descrito usando chaves e transistores.

```
module inverte(y,a);
    input a;
    output y;
    supply1 vdd;
    supply0 gnd;
    pmos p1(y, vdd, a);
    nmos n1(y, gnd, a);
        / *usa dos componentes pmos e nmos */
    endmodule
```

Linguagem de Descrição de Hardware : Tipos de Descrição

Descrição de tipo-misturado: Descrição de mais de um tipo. Sistemas de tamanho moderado ou grande.

Portas

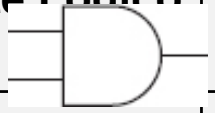
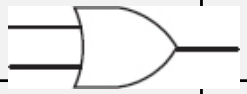
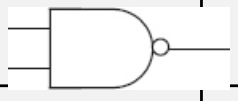
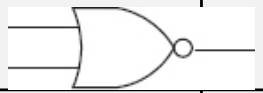
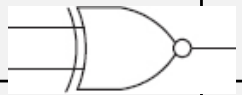
Input: Porta de entrada somente. Em qualquer expressão de atribuição, a porta aparecerá somente no lado direito (isto é a porta é lida).

output: Em Verilog a porta de saída pode aparecer em qualquer lado da expressão.

inout: Esta porta pode ser usa tanto como uma porta de entrada como de saída. A porta inout representa um barramento bidirecional.

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador lógico bit a bit

Operador	Equivalente Lógico	Tipo Operando	Tipo Resultado
&		Bit	Bit
|		Bit	Bit
~ (&)		Bit	Bit
~ (|)		Bit	Bit
^		Bit	Bit
~^		Bit	Bit
~		Bit	Bit

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador lógico booleano

Operador	Operação	Número de Operandos
&&	AND	Dois
||	OR	Dois

Resultado: 0 (false) ou 1 (true)
Exemplo:

Se x= 1011 e y= 0001 então z = x && y resultado 1

Se x= 1010 e y= 0101 então z = x && y resultado 0

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador lógico de redução

Operado	Operação	Número de Operandos
&	Redução AND	Um
|	Redução OR	Um
~&	Redução NAND	Um
~ |	Redução NOR	Um
^	Redução XOR	Um
~^	Redução XNOR	Um
!	NEGATION	Um

Exemplo: Se x= 1011 então z = &x

Resultado z = (1&0&1&1) = 0

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador Relacional

Operador	Descrição	Tipo de Resultado
==	Igualdade	0, 1, x
!=	Diferença	0, 1, x
===	Igualdade inclusive	0, 1
!==	Diferença inclusive	0, 1
<	Menor que	0, 1, x
<=	Menor ou igual	0, 1, x
>	Maior que	0, 1, x
>=	Maior ou igual	0, 1, x
?	Condicional	0, 1, x

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador Aritmético

Operador	Descrição	Tipo A ou B	Tipo Y T
+	Adição A + B	A numérico B numérico	Numérico
-	Subtração A - B	A numérico B numérico	Numérico
*	Multiplicação A * B	A numérico B numérico	Numérico
/	Divisão A / B	A numérico B numérico	Numérico
%	Módulo A % B	A numérico, não real B numérico, não real	Numérico, não real
**	Expoente A ** B	A numérico B numérico	Numérico
{,}	Concatenação {A , B}	A numérico ou array B numérico ou array	Mesmo que A
{N{A}}	Repetição	A numérico ou array	Mesmo que A

Operadores: Lógico, Relacional, Aritmético e de deslocamento

Operador de Deslocamento

Operação	Descrição	Operando A ➔	Operando A
`A << 1`	Deslocamento lógico de uma posição para esquerda	1110	1100
`A << 2`	Deslocamento lógico de duas posições para esquerda	1110	1000
`A >> 1`	Deslocamento lógico de uma posição para direita	1110	0111
`A >> 2`	Deslocamento lógico de duas posições para direita	1110	0011
A.>>> 2	Deslocamento aritmético de duas posições para direita	1110	1111
A.<<< 2	Deslocamento aritmético de duas posições para esquerda	1110	1000

Tipos de dados

Tipo redes

Valor	Definição
0	Lógico 0 (false)
1	Lógico 1 (true)
X	Não conhecido
Z	Alta impedância

Exemplos:

```
wire soma;
wire S1 = 1'b0;
```

Tipos de dados

Tipo registradores

Valor	Definição
0	Lógico 0 (false)
1	Lógico 1 (true)
X	Não conhecido
Z	Alta impedância

Exemplos:

```
reg soma;
```

Tipos de dados

Vetores

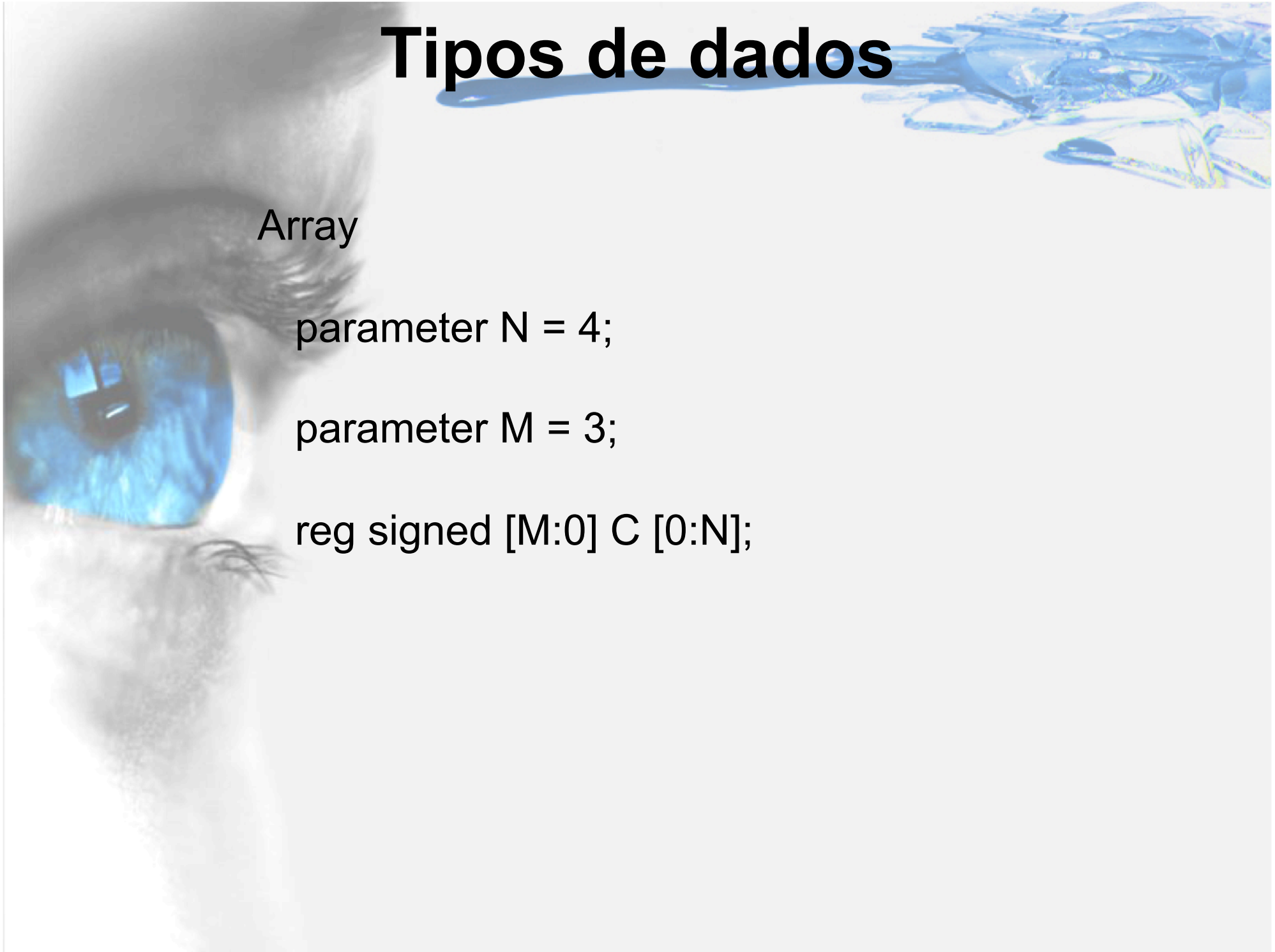
```
wire [3:0] a = 4'b1010;
reg [7:0] total = 8'd12;
```

integer , real

Parâmetro

```
module compr_genr (X, Y, xgty, xlty, xeqy);
parameter N = 3;
input [N:0] X, Y;
output xgty, xlty, xeqy;
wire [N:0] soma, Yb;
```

Tipos de dados



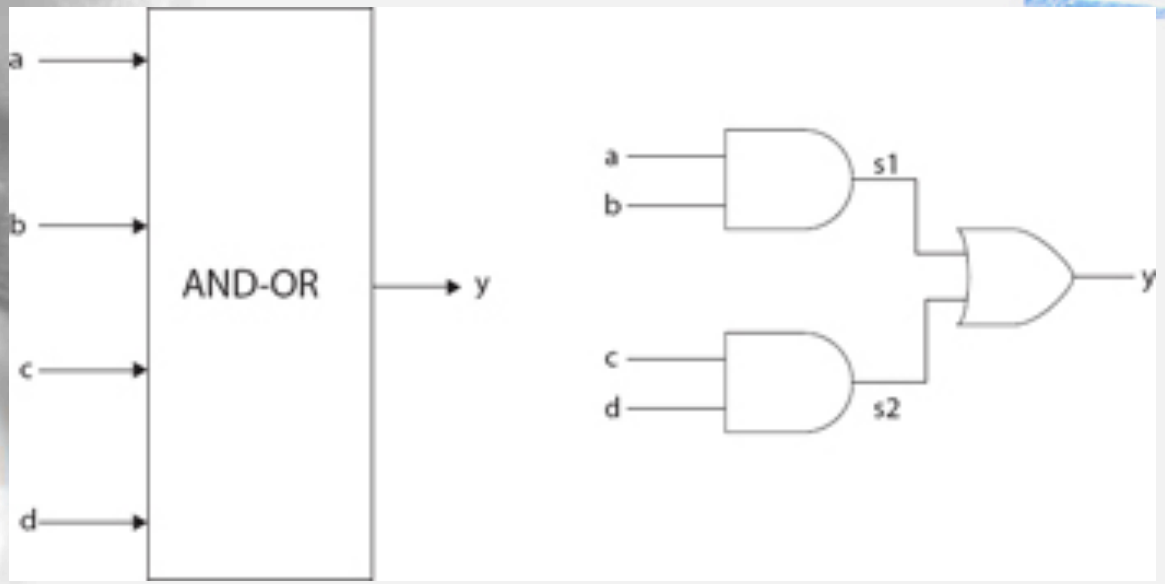
Array

parameter N = 4;

parameter M = 3;

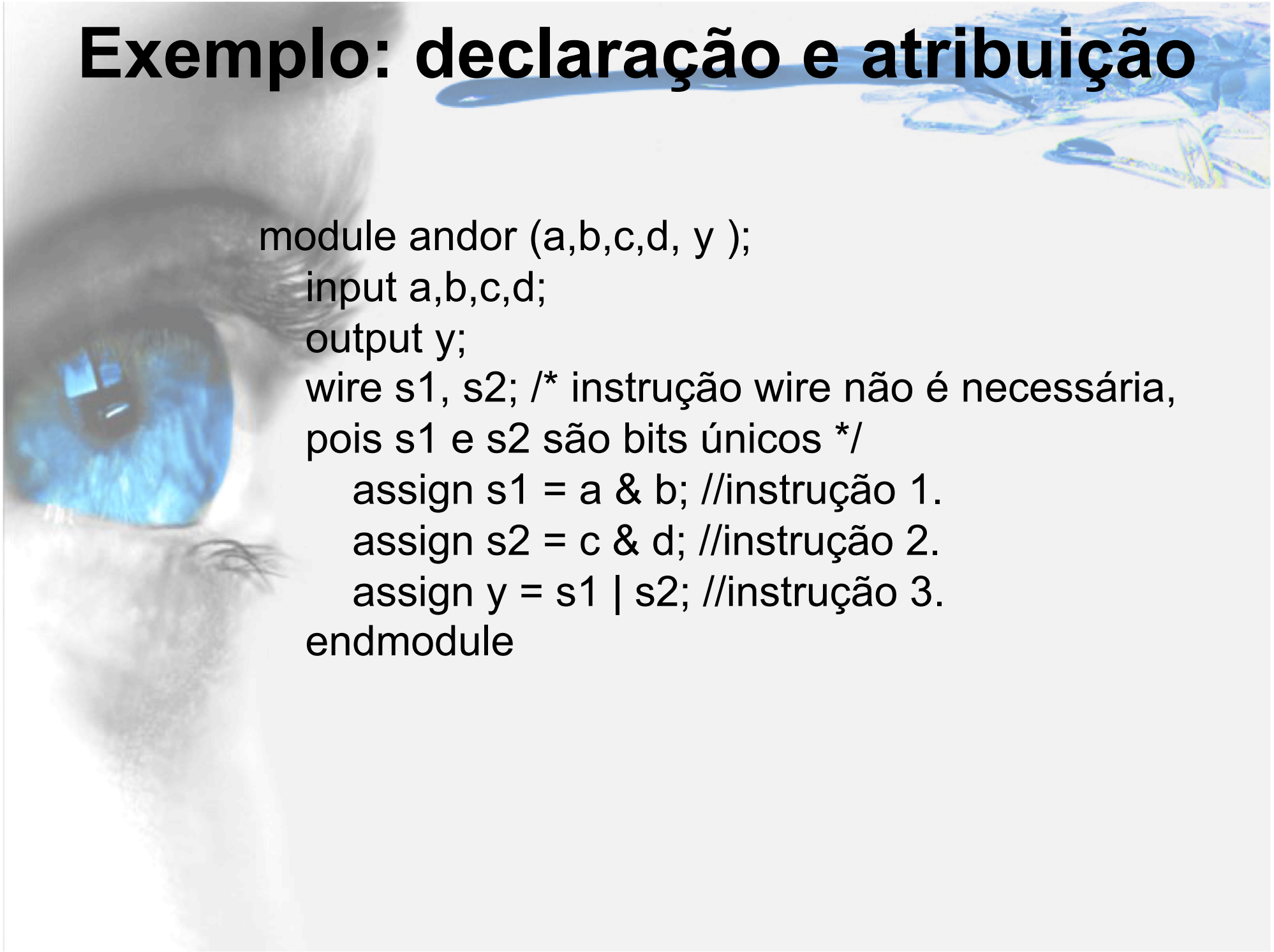
reg signed [M:0] C [0:N];

Tipos de dados



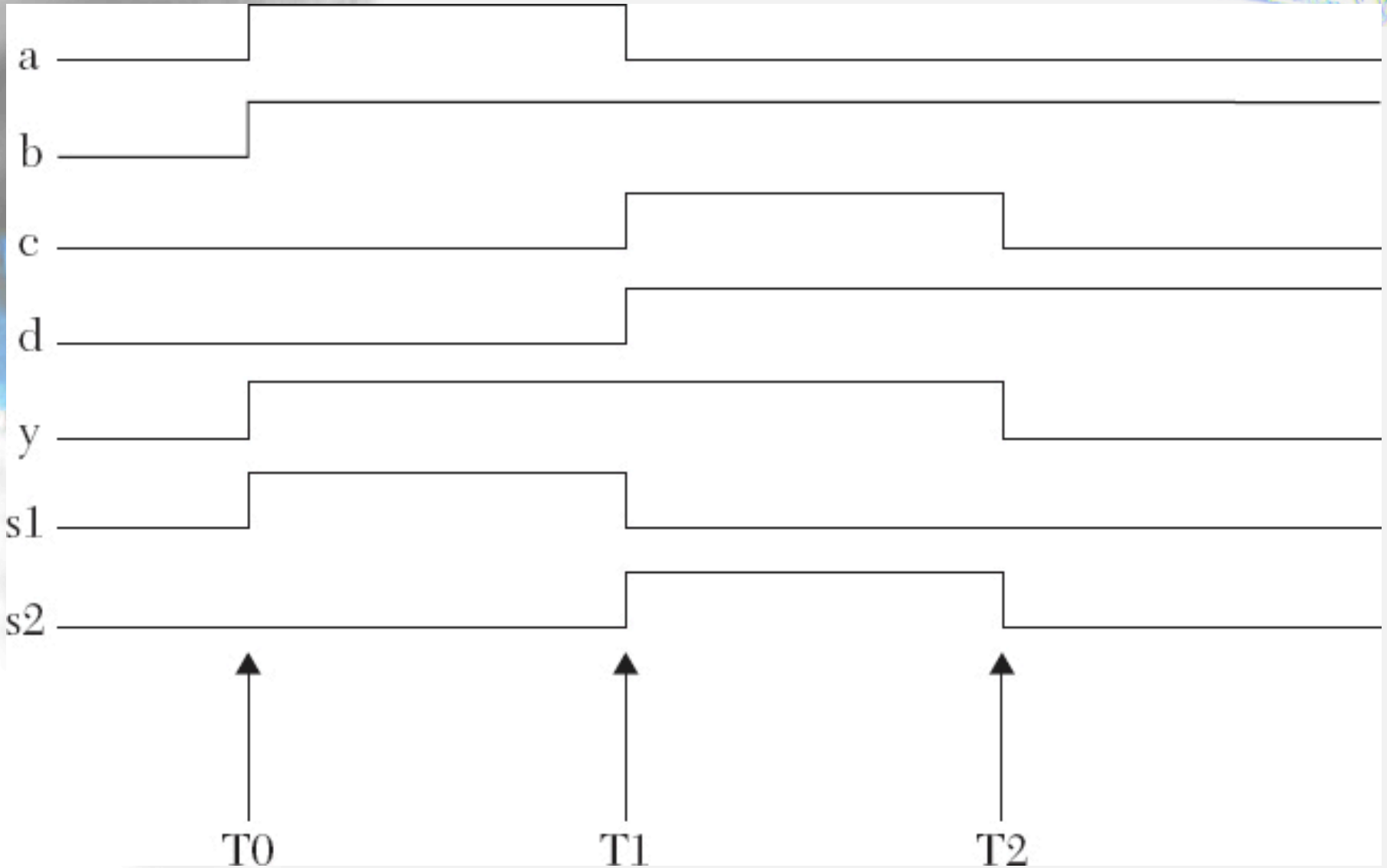
s1 = ab ; s2 = cd ;
y = s1 + s2
Y = ab + cd

Exemplo: declaração e atribuição



```
module andor (a,b,c,d, y );
    input a,b,c,d;
    output y;
    wire s1, s2; /* instrução wire não é necessária,
    pois s1 e s2 são bits únicos */
        assign s1 = a & b; //instrução 1.
        assign s2 = c & d; //instrução 2.
        assign y = s1 | s2; //instrução 3.
    endmodule
```

Exemplo: declaração e atribuição



Exemplo: tempo de espera

```
module and_orDlyVr( a,b,c,d, y );
input a,b,c,d;
output y;
time dly = 10;
wire s1, s2;
    assign # dly s1 = a & b; //statement 1.
    assign # dly s2 = c & d; //statement 2.
    assign # dly y = s1 | s2; //statement 3.
endmodule
```

Exemplo: tempo de espera