第9章 结构建模

本章讲述Verilog HDL中的结构建模方式。结构建模方式用以下三种实例语句描述：

- Gate实例语句
- UDP实例语句
- Module实例语句

第5章和第6章已经讨论了门级建模方式和UDP建模方式，本章讲述模块实例语句。

9.1 模块

Verilog HDL中，基本单元定义成模块形式，如下所示：

```
module module_name(port_list);
  Declarations_and_Statements
endmodule
```

端口队列*port_list*列出了该模块通过哪些端口与外部模块通信。

9.2 端口

模块的端口可以是输入端口、输出端口或双向端口。缺省的端口类型为线网类型（即wire类型）。但是，端口可被显式地指定为线网。输出或输入输出端口能够被重新声明为reg型寄存器。无论是在线网说明还是寄存器说明中，线网或寄存器必须与端口说明中指定的长度相同。下面是一些端口说明实例。

```
module Micro (PC, Instr, NextAddr);
  //端口说明
  input [3:1] PC;
  output [1:8] Instr;
  inout [16:1] NextAddr;

  //重新说明端口类型：
  wire [16:1] NextAddr; //该说明是可选的，但如果指定了，就必须与它的端口说明保持相同长度。

  reg [1:8] Instr;
  //Instr已被重新说明为reg类型，因此它能在always 语句或在initial语句中赋值。
  ...
endmodule
```

9.3 模块实例语句

一个模块能够在另外一个模块中被引用，这样就建立了描述的层次。模块实例语句形式如下：

```
module_name instance_name(port_associations);
```

信号端口可以通过位置或名称关联；但是关联方式不能够混合使用。端口关联形式如下：

```
port_expr                    //通过位置。
.PortName (port_expr)        //通过名称。
```

*port_expr*可以是以下的任何类型：

1) 标识符（reg或net）

2) 位选择

3) 部分选择

4) 上述类型的合并

5) 表达式（只适用于输入端口）

在位置关联中，端口表达式按指定的顺序与模块中的端口关联。在通过名称实现的关联中，模块端口和端口表达式的关联被显式地指定，因此端口的关联顺序并不重要。下例使用两个半加器模块构造全加器；逻辑图如图 9-1所示。

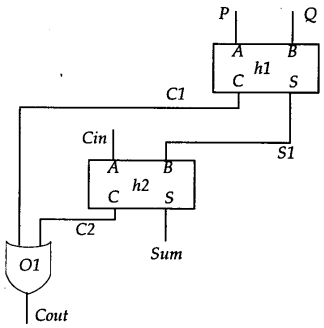


图9-1 使用两个半加器模块构造的全加器

```
module HA(A,B,S,C);
  input A,B;
  output S, C;
  parameter AND_DELAY = 1, XOR_DELAY = 2;

  assign #XOR_DELAY  S = A ^ B;
  assign #AND_DELAY  C = A & B;
endmodule

module FA(P, Q, Cin, Sum, Cout);
  input P, Q, Cin;
  output Sum, Cout;
  parameter OR_DELAY = 1;
  wire S1, C1, C2;

  //两个模块实例语句
  HA h1 (P, Q, S1, C1);  //通过位置关联。
  HA h2 (.A(Cin), .S(Sum), .B(S1), .C(C2)); //通过端口与信号的名字关联。
  //门实例语句：
  or #OR_DELAY O1 (Cout, C1, C2);
endmodule
```

在第一个模块实例语句中，*HA*是模块的名字，*h1*是实例名称，并且端口按位置关联，即信号*P*与模块（*HA*）的端口*A*连接，信号*Q*与端口*B*连接，*S1*与*S*连接，*C1*与模块端口*C*连接。在第二个实例中，端口按名称关联，即模块（ *HA*）和端口表达式间的连接是显示地定义的。

下例是使用不同端口表达式形式的模块实例语句。

```
Micro M1 (UdIn[3:0], {WrN, RdN}, Status[0], Status[1],
         & UdOut [0:7], TxData);
```

这个实例语句表示端口表达式可以是标识符（ *TxData*）、位选择（*Status*[0]）、部分位选择（*UdIn*[3:0]）、合并（{*WrN*,*RdN*}）或一个表达式（& *udOut*[0:7]）；表达式只能够连接到输入端口。

9.3.1 悬空端口

在实例语句中，悬空端口可通过将端口表达式表示为空白来指定为悬空端口，例如：

```
DFF d1 (.Q(QS), .Qbar(), .Data(D),
        .Preset(), .Clock(CK));   /名称对应方式。

DFF d2 (QS, , D, , CK);//位置对应方式。
//输出端口Qbar悬空。
//输入端口Preset打开，其值设定为z。
```

在这两个实例语句中，端口*Qbar*和*Preset*悬空。

模块的输入端悬空，值为高阻态z。模块的输出端口悬空，表示该输出端口废弃不用。

9.3.2 不同的端口长度

当端口和局部端口表达式的长度不同时，端口通过无符号数的右对齐或截断方式进行匹配。例如：

```
module Child(Pba, Ppy);
  input [5:0] Pba;
  output [2:0] Ppy;
  ...
endmodule

module Top;
  wire [1:2] Bdl;
  wire [2:6] Mpr;

  Child C1(Bdl, Mpr);
endmodule
```

在对*Child*模块的实例中，*Bdl*[2]连接到*Pba*[0]，*Bdl*[1] 连接到*Pba*[1]，余下的输入端口*Pba*[5]、*Pba*[4]和*Pba*[3]悬空，因此为高阻态z。与之相似，*Mpr*[6]连接到*Ppy*[0]，*Mpr*[5]连接到*Ppy*[1]，*Mpr*[4] 连接到*Ppy*[2]。参见图9-2。

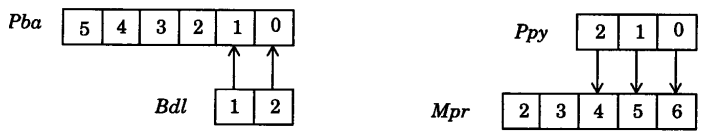


图9-2 端口匹配

9.3.3 模块参数值

当某个模块在另一个模块内被引用时，高层模块能够改变低层模块的参数值。模块参数值的改变可采用下述两种方式：

1) 参数定义语句（defparam）；

2) 带参数值的模块引用。

1. 参数定义语句

参数定义语句形式如下：

```
defparam hier_path_name1= value1,
         hier_path_name2= value2, ...;
```

较低层模块中的层次路径名参数可以使用如下语句显式定义（层次路径名在下一章中讲

述）。下面是一个例。模块*FA*和*HA*已在本节前面描述过。

```
module TOP (NewA,NewB,NewS,NewC);
  input NewA,NewB;
  output NewS,NewC;
  defparam  Ha1.XOR_DELAY = 5,
              //实例Ha1中的参数XOR_DELAY。
            Ha1.AND_DELAY = 2;
              //实例Ha1中参数的AND_DELAY。

HA Hal (NewA, NewB, NewS,NewC);
endmodule
module TOP2 (NewP, NewQ, NewCin,NewSum,NewCout);
  input NewP, NewQ, NewCin;
  output NewSum,NewCout;
  defparam  Fa1.h1.XOR_DELAY = 2,
              //实例Fa1的实例h1中的参数XOR_DELAY。
            Fa1.h1.AND_DELAY = 3,
              //实例Fa1的实例h1中的参数AND_DELAY。
            Fa1.h2.XOR_DELAY = 4,
              //实例Fa1的实例h2中的参数XOR_DELAY。
            Fa1.h2.AND_DELAY = 3;
              //实例Fa1的实例h2中的参数AND_DELAY。
  FA Fa1 (NewP, NewQ, NewCin, NewSum, NewCout);
endmodule
```

5.2(HAewB, NewS,NewC)\T 8 Tf w (= 2:)Ti -14.8 -3 T BT /F5 80.931N D D E L A Y 64 Tw (H a 1)Ti ET BT /F5 8 18

相似；但由于对复杂模块的引用时，其实例语句不能像对门实例语句那样指定时延，故此处不会导致混淆。

参数值还可以表示长度。下面是通用的M×N乘法器建模的实例。

```
module Multiplier(Opd_1,Opd_2,Result);
  parameter EM = 4,EN = 2;  //默认值
  input [EM:1] Opd_1;
  input [EN:1] Opd_2;
  output [EM+EN:1] Result;

  assign Result = Opd_1 * Opd_2;
endmodule
```

这个带参数的乘法器可在另一个设计中使用，下面是8×6乘法器模块的带参数引用方式：

```
wire [1:8] Pipe_Reg;
wire [1:6] Dbus;
wire [1:14] Addr_Counter;
...
Multiplier #(8,6) M1(Pipe_Reg,Dbus,Addr_Counter);
```

第1个值8指定了参数*EM*的新值，第2个值6指定了参数*EN*的新值。

9.4 外部端口

在迄今为止所见到的模块定义中，端口表列举出了模块外部可见的端口。例如，

```
module Scram_A(Arb,Ctrl,Mem_Blk,Byte);
  input[0:3] Arb;
  input Ctrl;
  input [8:0] Mem_Blk;
  output [0:3] Byte;
  ...
endmodule
```

Arb、*Ctrl*、*Mem_Blk*和*Byte*为模块端口。这些端口同时也是外部端口，即在实例中，当采用名称关联方式时，外部端口名称用于指定相互连接。下面是模块*Scram_A*的实例。

```
Scram_A SX(.Byte(B1),.Mem_Blk(M1),.Ctrl(C1),.Arb(A1));
```

在模块*Scram_A*中，外部端口名称隐式地指定。Verilog HDL中提供显式方式指定外部端口名称。这可以通过按如下形式指定一个端口来完成：

```
.external_port_name(internal_port_name)
```

下面是同一个例子，只不过是显式地指定外部端口。

```
module Scram_B (.Data(Arb),.Control(Ctrl),
                .Mem_Word(Mem_Blk),.Addr(Byte));
  input [0:3] Arb;
  input Ctrl;
  input [8:0]Mem_Blk;
  output [0:3] Byte;
  ...
endmodule
```

模块*Scram_B*在此实例中指定的外部端口是*Data*、*Control*、*Mem_Word*和*Addr*。端口表显式地表明了外部端口和内部端口之间的连接。注意外部端口无需声明，但是模块的内部端口

```
module Scram_E(.Data(), .Control (Ctrl),
               .Mem_Word ({ Mem_Blk[0], Mem_Blk [1]}),
               .Addr());
  input Ctrl;
  input [8:0] Mem_Blk;
  ...
endmodule
```

模块*Scram_E*有两个外部端口*Data*和*Addr*，这两个端口在使用时被悬空。

一个内部端口是否能与多个外部端口连接？Verilog HDL允许这样连接。例如，

```
module FanOut (.A(CtrlIn),.B(CondOut),.C(CondOut));
  input CtrlIn;
  output CondOut;

  assign CondOut = CtrlIn;
endmodule
```

内部端口*CondOut*与两个外部端口*B*和*C*连接，所以*CondOut*的值在*B*和*C*上都出现。

9.5 举例

下例采用结构模型描述十进制计数器。十进制计数器的逻辑图如图 9-3所示。

图9-3 十进制计数器

```
module Decade_Ctr (Clock,Z);
  input Clock;
  output [0:3] Z;
  wire S1,S2;

  and A1 (S1,Z[2],Z[1]); //基本门实例语句。

  // 4个模块实例语句：
  JK_FF JK1(.J(1'b1),.K(1'b1),.CK(Clock),.Q(Z[0]),.NQ()),
        JK2(.J(S2),.K(1'b1),.CK(Z[0]),.Q(Z[1]),.NQ()),
        JK3(.J(1'b1),.K(1'b1),.CK(Z[1]),.Q(Z[2]),.NQ()),
        JK4(.J(S1),.K(1'b1),.CK(Z[0]),.Q(Z[3]),.NQ(S2));
endmodule
```

注意常数作为输入端口信号的用法，以及悬空端口。

下面是另一个例子，3位可逆计数器的逻辑结构如图9-4所示，其结构描述如下：

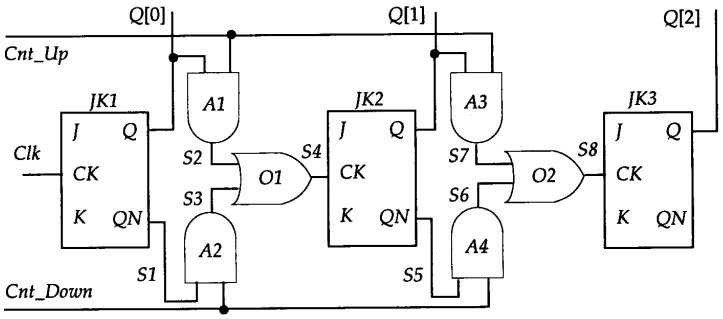
```
module Up_Down(Clk,Cnt_Up,Cnt_Down,Q);
  input Clk,Cnt_Up,Cnt_Down;
  output [0:2] Q;
```

```
  wire S1,S2,S3,S4,S5,S6,S7,S8

  JK_FF JK1(1'b1,1'b1,Clk,Q[0],S1),
        JK2(1'b1,1'b1,S4,Q[1],S5),
        JK3(1'b1,1'b1,S8,Q[2],);

  and A1(S2,Cnt_Up,Q[0]),
      A2(S3,S1,Cnt_Down),
      A3(S7,Q[1],Cnt_Up),
      A4(S6,S5,Cnt_Down);

  or O1(S4,S2,S3),
     O2(S8,S7,S6);
endmodule
```



所有触发器的J、 K输入端口与高电平相连

图9-4 位可逆计数器

习题

1.模块实例语句与门实例语句的区别是什么？

2.当端口悬空时，即端口没有被连接时，端口的值是什么？

3.对于9.3节中的模块*FA*，*OR_DELAY*值为4，*XOR_DELAY*值为7，AND_DELAY值为5，写出其结构描述形式。

4.用本章讲述的模块*FA*编写执行加法和减法的4位ALU的结构模型。

5.用5.11节中描述的*MUX4x1*模块编写16-1多路选择器的结构化模型。

6.用异步低电平复位描述通用 *N*位计数器。将通用计数器在实例语句中用作 5位计数器用测试验证程序验证这个5位计数器。