

模型机的 8 重 3-1 多路复用器

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_unsigned.all;
4  use ieee.std_logic_arith.all;
5
6  entity zjw_8_1 is
7  port(
8      in1:in std_logic_vector(7 downto 0);
9      enable:in std_logic_vector(3 downto 0);
10     out1:out std_logic);
11 end zjw_8_1;
12
13 architecture struct of zjw_8_1 is
14     component zjw_canshuhua
15     generic(n:integer:=4;
16            m:integer:=2);
17     port(
18         in1:in std_logic_vector(n-1 downto 0);
19         enable:in std_logic_vector(m-1 downto 0);
20         out1:out std_logic
21     );
22 end component;
23
24 begin
25     g0:zjw_canshuhua generic map(8,4) port map(in1,enable,out1);
26 end struct;
```

参数化多路复用器

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_unsigned.all;
4  use ieee.std_logic_arith.all;
5
6  entity zjw_canshuhua is
7  generic(n:integer:=4;
8          m:integer:=2);
9  port(
10     in1:in std_logic_vector(n-1 downto 0);
11     enable:in std_logic_vector(m-1 downto 0);
12     out1:out std_logic
13 );
14 end zjw_canshuhua;
15
16 architecture struct of zjw_canshuhua is
17 begin
18     process(in1,enable)
19     begin
20         out1<=in1(conv_integer(enable));
21     end process;
22 end struct;
```

参数化多路复用器定制为 8-1 多路复用器

```
1  library IEEE;
2  use ieee.std_logic_1164.all;
3
4  entity zjw_8_3_1 is
5  port(
6      JUMP,MOVB,MOVCI:in std_logic;
7      in0,in1,in2:in std_logic_vector(7 downto 0);
8      out1:out std_logic_vector(7 downto 0)
9  );
10 end zjw_8_3_1;
11
12 architecture struct of zjw_8_3_1 is
13 begin
14     process(in0,in1,in2,JUMP,MOVB,MOVCI)
15     begin
16         if JUMP='1' then
17             out1<=in0;
18         elsif MOVCI='1' then
19             out1<=in1;
20         elsif MOVB='1' then
21             out1<=in2;
22         else
23             out1<=in0;
24         end if;
25     end process;
26 end struct;
```

模型机的控制信号产生逻辑

```
1  library ieee;
2  use ieee.std_logic_1164.all;
3  use ieee.std_logic_unsigned.all;
4  use ieee.std_logic_arith.all;
5
6  entity zjw_kongzhixinhao is
7  port(
8      IR:in std_logic_vector(7 downto 0);
9      MOVA,MOVB,MOVC,ALU,NOT0,SHR,SHL,JMP,JZ,Z,JC,C,NOP,HALT,SM:in std_logic;
10     SME,LD_IR,DL,XL,F,FRL,FRR,CF,ZF,M,LD_PC,IN_PC,N_WE,N_CS,CFE,ZFE:out std_logic;
11     MADD,RAA,RWBA:out std_logic_vector(1 downto 0):="00";
12     S:out std_logic_vector(3 downto 0)
13 );
14 end zjw_kongzhixinhao;
15
16 architecture struct of zjw_kongzhixinhao is
17     signal command:std_logic_vector(3 downto 0);
18     begin
19         command<=IR(7 downto 4);
20         S(3)<=IR(7);
21         S(2)<=IR(6);
22         S(1)<=IR(5);
23         S(0)<=IR(4);
24         process (MOVA,MOVB,MOVC,ALU,NOT0,SHR,SHL,JMP,JZ,Z,JC,C,NOP,HALT,SM,IR)
25         begin
26             if SM='0' then
27                 LD_IR<='1';
28                 DL<='1';
29                 XL<='0';
30                 N_CS<='0';
31                 MADD<="00";
32                 LD_PC<='0';
33                 IN_PC<='1';
34                 N_WE<='1';
35             elsif SM='1' then
36                 if MOVA='1' then
37                     RAA<=IR(1 downto 0);
38                     RWBA<=IR(3 downto 2);
39                     MADD<="00";
40                     LD_PC<='0';
41                     IN_PC<='0';
42                     N_WE<='0';
43
44                     N_WE<='0';
45                     XL<='0';
46                     DL<='1';
47                     M<='1';
48                     N_CS<='0';
49                     F<='1';
50                     FRR<='0';
51                     FRL<='0';
52                     LD_IR<='0';
53                     CFE<='0';
54                     ZFE<='0';
55                     SME<='1';
56                 elsif MOVB='1' then
57                     RAA<=IR(1 downto 0);
58                     RWBA<="11";
59                     MADD<="10";
60                     LD_PC<='0';
61                     IN_PC<='0';
62                     N_WE<='1';
63                     XL<='1';
64                     DL<='0';
65                     M<='1';
66                     N_CS<='0';
67                     F<='1';
68                     FRR<='0';
69                     FRL<='0';
70                     LD_IR<='0';
71                     CFE<='0';
72                     ZFE<='0';
73                     SME<='1';
74                 elsif MOVC='1' then
```

```

71         SME<='1';
72     elsif MOV<='1' then
73         RAA<="11";
74         RWBA<=IR(3 downto 2);
75         MADD<="01";
76         LD_PC<='0';
77         IN_PC<='0';
78         N_WE<='0';
79         XL<='0';
80         DL<='1';
81         M<='1';
82         N_CS<='0';
83         F<='1';
84         FRR<='0';
85         FRL<='0';
86         LD_IR<='0';
87         CFE<='0';
88         ZFE<='0';
89         SME<='1';
90     elsif ALU<='1' then
91         RAA<=IR(1 downto 0);
92         RWBA<=IR(3 downto 2);
93         MADD<="00";
94         LD_PC<='0';
95         IN_PC<='0';
96         N_WE<='0';
97         XL<='0';
98         DL<='0';
99         if command="1011" then
100             M<='0';
101         else
102             M<='1';
103         end if;
104         N_CS<='1';
105         F<='1';
106         FRR<='0';
107         FRL<='0';
108         LD_IR<='0';
109         if command="1001" then
110             CFE<='1';
111             ZFE<='0';
112         elsif command="0110" then
113             ZFE<='0';
114             elsif command="0110" then
115                 CFE<='0';
116                 ZFE<='1';
117             else
118                 ZFE<='0';
119                 CFE<='0';
120             end if;
121             SME<='1';
122         elsif NOT0<='1' then
123             RAA<=IR(1 downto 0);
124             RWBA<=IR(3 downto 2);
125             MADD<="00";
126             LD_PC<='0';
127             IN_PC<='0';
128             N_WE<='0';
129             XL<='0';
130             DL<='0';
131             M<='1';
132             N_CS<='1';
133             F<='1';
134             FRR<='0';
135             FRL<='0';
136             LD_IR<='0';
137             CFE<='0';
138             ZFE<='0';
139             SME<='1';
140         elsif SHR<='1' then
141             RAA<=IR(1 downto 0);

```

```

139      RAA<=IR(1 downto 0);
140      RWBA<=IR(3 downto 2);
141      MADD<="00";
142      LD_PC<='0';
143      IN_PC<='0';
144      N_WE<='0';
145      XL<='0';
146      DL<='0';
147      M<='1';
148      N_CS<='1';
149      F<='0';
150      FRR<='1';
151      FRL<='0';
152      LD_IR<='0';
153      CFE<='0';
154      ZFE<='0';
155      SME<='1';
156  ── elseif SHL='1' then
157      RAA<=IR(1 downto 0);
158      RWBA<=IR(3 downto 2);
159      MADD<="00";
160      LD_PC<='0';
161      IN_PC<='0';
162      N_WE<='0';
163      XL<='0';
164      DL<='0';
165      M<='1';
166      N_CS<='1';
167      F<='0';
168      FRR<='0';
169      FRL<='1';
170      LD_IR<='0';
171      CFE<='0';
172      ZFE<='0';
173      SME<='1';
174  ── elseif JMP='1' or (JZ='1' and Z='1') or (JC='1'and C='1') then
175      RAA<=IR(1 downto 0);
176      RWBA<=IR(3 downto 2);
177      MADD<="00";
178      LD_PC<='1';
179      IN_PC<='0';
180      N_WE<='1';

```

```

180      N_WE<='1';
181      XL<='0';
182      DL<='1';
183      M<='0';
184      N_CS<='0';
185      F<='0';
186      FRR<='0';
187      FRL<='0';
188      LD_IR<='0';
189      CFE<='0';
190      ZFE<='0';
191      SME<='1';
192  ── elseif NOP='1' or (JZ='1' and Z='0') or (JC='1'and C='0') then
193      RAA<=IR(1 downto 0);
194      RWBA<=IR(3 downto 2);
195      MADD<="00";
196      LD_PC<='1';
197      IN_PC<='1';
198      N_WE<='1';
199      XL<='0';
200      DL<='0';
201      M<='0';
202      N_CS<='0';
203      F<='0';
204      FRR<='0';
205      FRL<='0';
206      LD_IR<='0';
207      CFE<='0';
208      ZFE<='0';
209      SME<='1';
210  ── elseif HALT='1' then

```

```

210         elsif HALT='1' then
211             RAA<=IR(1 downto 0);
212             RWBA<=IR(3 downto 2);
213             MADD<="00";
214             LD_PC<='0';
215             IN_PC<='0';
216             N_WE<='1';
217             XL<='0';
218             DL<='0';
219             M<='0';
220             N_CS<='1';
221             F<='0';
222             FRR<='0';
223             FRL<='0';
224             LD_IR<='0';
225             CFE<='0';
226             ZFE<='0';
227             SME<='0';
228         end if;
229     end if;
230 end process;
231 end struct;

```