AHDL
训练教程

哈尔滨工业大学威海校区
戴伏生

什么是 AHDL

- ***A***ltera ***H***ardware ***D***escription ***L***anguage
  - 由 Altera企业开发
  - 集成到Altera的软件 Max+Plus II
  - 用语言描述硬件来替代图形
    - 修改更轻易
    - 易于维护
  - 非常适合于
    - 复杂的组合逻辑
      - BCD 到 7段转换
      - 地址解码
    - 状态机
    - 其他你想要的……..

继续...

- 就象图形输入一样轻易
  - 强有力的HDL (硬件描述语言)
  - VHDL, Verilog HDL 等等.

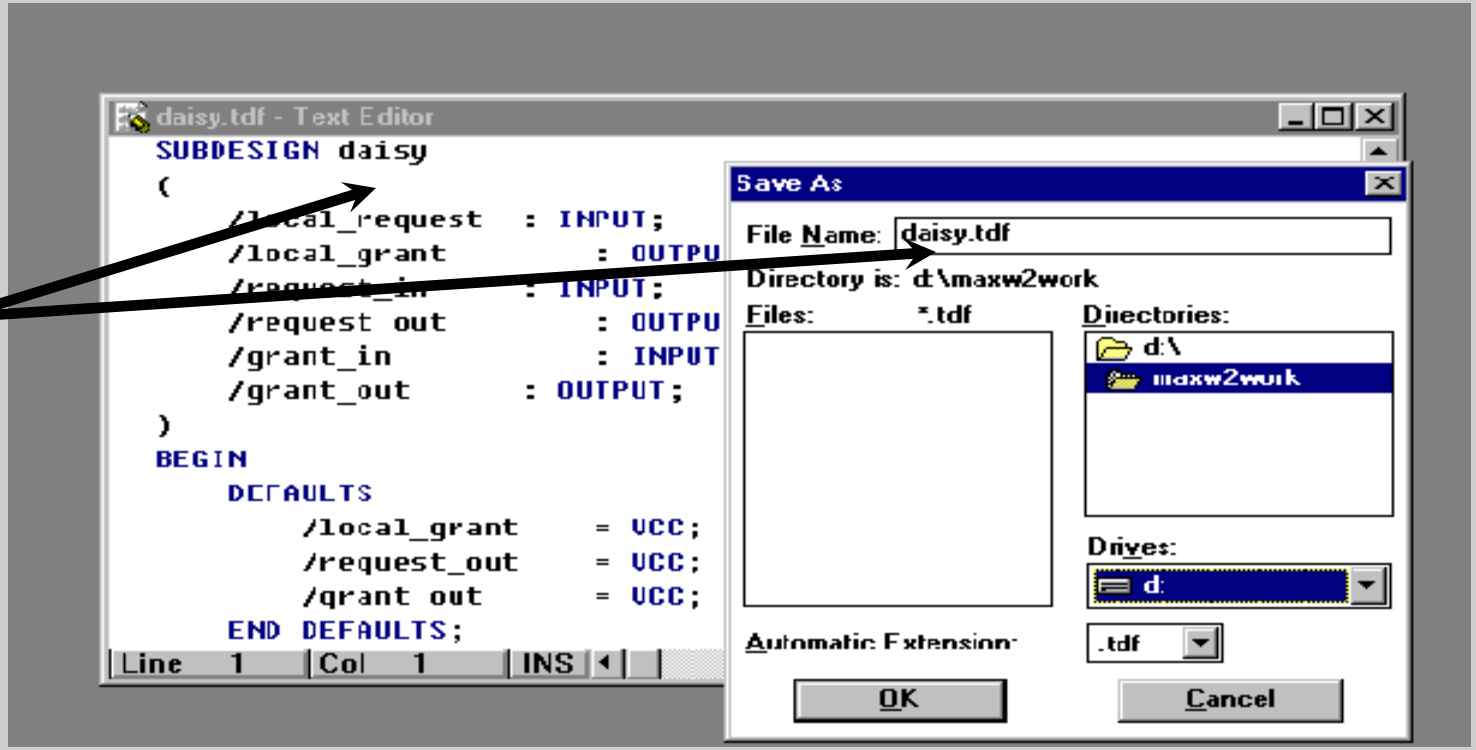
怎样用ADHL

- 用任何文本编辑器建立这个文件
  - Altera 软件 Max+Plus II 提供文本编辑器

继续.....

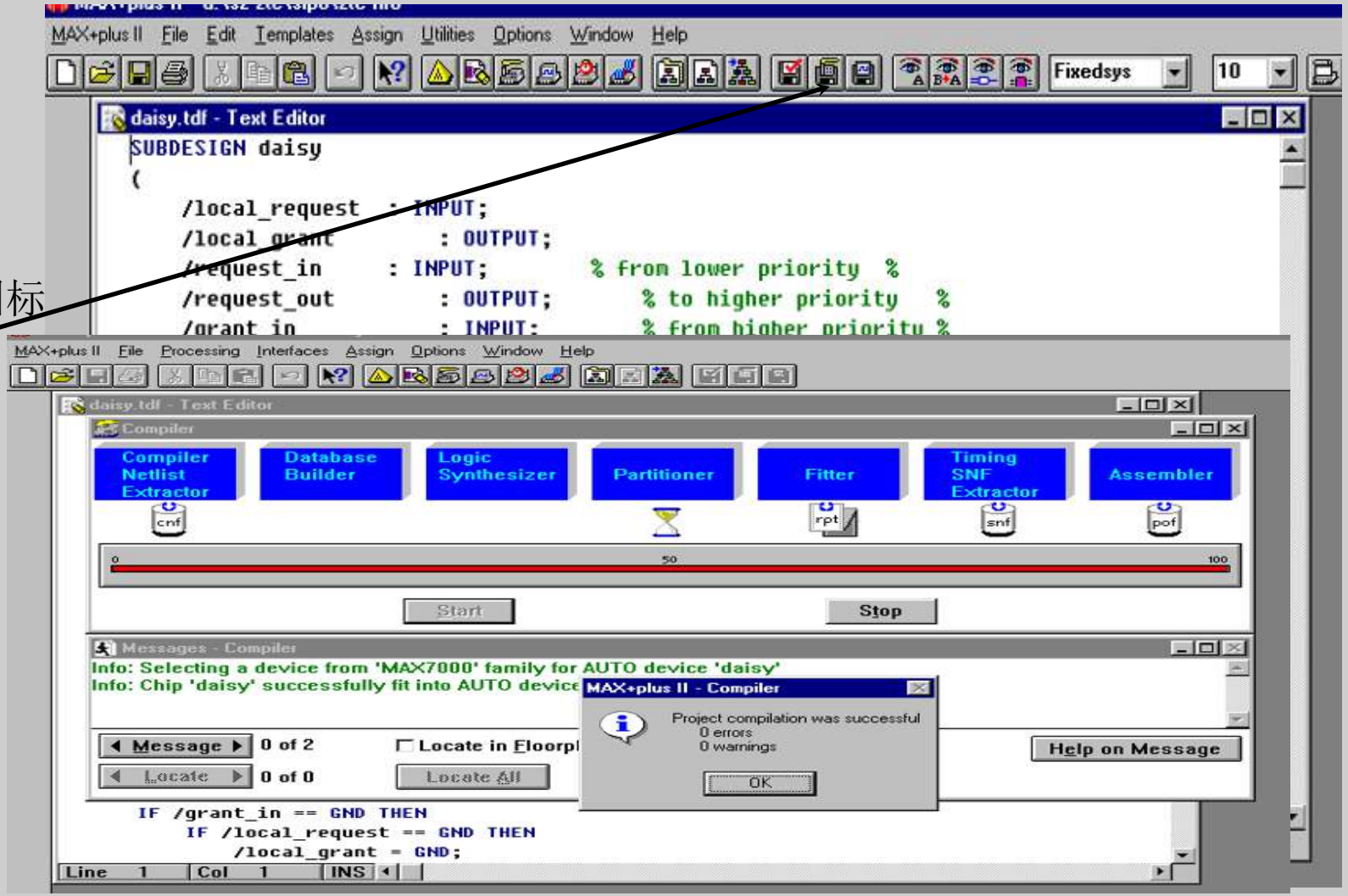
- 用 name.TDF保存你的ADHL文件

必须一样



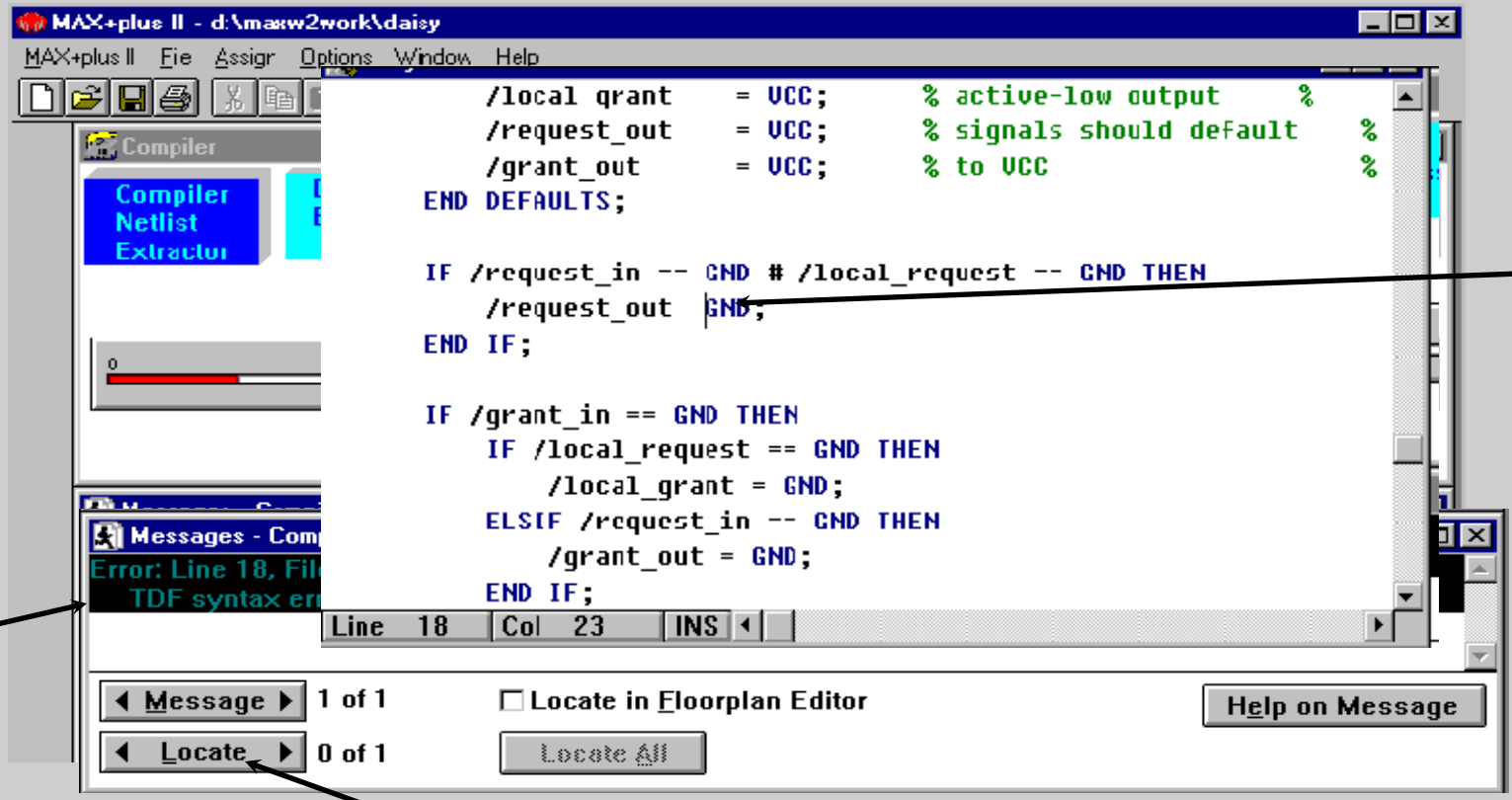
继续...

按这个图标


MAX+plus II File Edit Templates Assign Utilities Options Window Help
Fixedsys
10
daisy.tdf - Text Editor
SUBDESIGN daisy
(
/local_request : INPUT;
/local_grant : OUTPUT;
/request_in : INPUT; % from lower priority %
/request_out : OUTPUT; % to higher priority %
/grant_in : INPUT; % from higher priority %
MAX+plus II File Processing Interfaces Assign Options Window Help
daisy.tdf - Text Editor
Compiler
Compiler Netlist Extractor
Database Builder
Logic Synthesizer
Partitioner
Fitter
Timing SNF Extractor
Assembler
cnf
rpt
snf
pof
0
50
100
Start
Stop
Messages - Compiler
Info: Selecting a device from 'MAX7000' family for AUTO device 'daisy'
Info: Chip 'daisy' successfully fit into AUTO device
MAX+plus II - Compiler
Project compilation was successful
0 errors
0 warnings
OK
Message 0 of 2
Locate 0 of 0
Locate in Floorpl
Locate All
Help on Message
IF /grant_in == GND THEN
IF /local_request == GND THEN
/local_grant = GND;
Line 1
Col 1
INS

编译期间的错误定位

- 以便的错误定位



错误位置

点按错误信息

点按定位键

一般的AHDL格式

```
SUBDESIGN decode1
( input_pin_name : INPUT;
  input_bus_name[15..0] : INPUT;
  output_pin_name : OUTPUT;
  output_bus_name : OUTPUT;
)
BEGIN
ouptut_pin_name = input_pin_name;
output_bus_name = input_bus_name;
END;
```

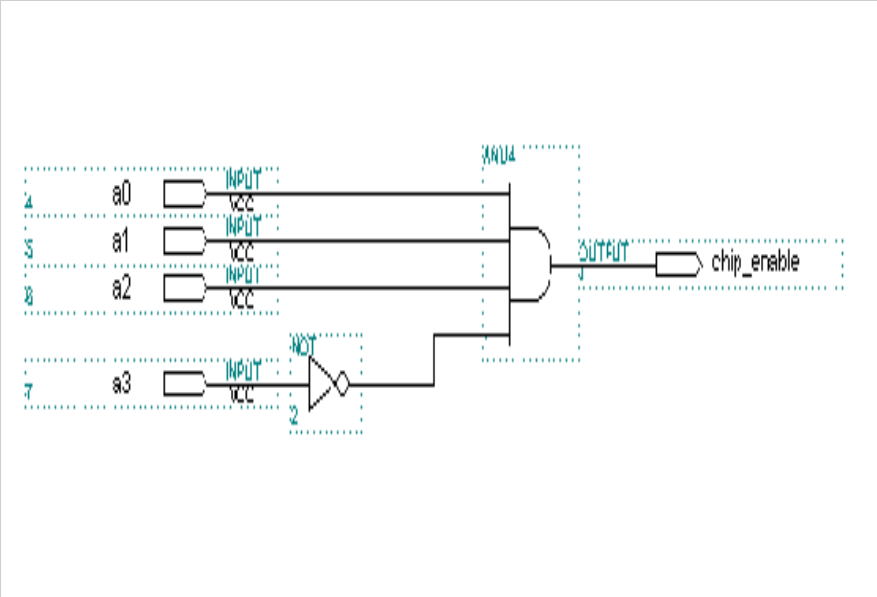
关键字

定义I/O口

逻辑

AHDL 格式

你的第一种AHDL设计 – 地址解码器



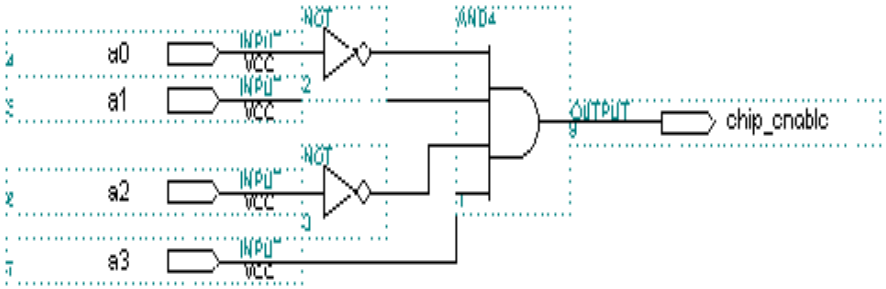

```
SUBDESIGN decode1
( a[3..0] : input;
  chip_enable : output;
)
begin
chip_enable = (a[3..0] == H"7");
end;
```

Chip_enable = a0 & a1 & a2 & !a3

为何我用AHDL替代图形

- 轻易编辑
- Document during the coding
- 我想解码 H"A" 不是 H"7"

Chip_enable = !a0 & a1 & !a2 & a3



需要更多的工作去修改

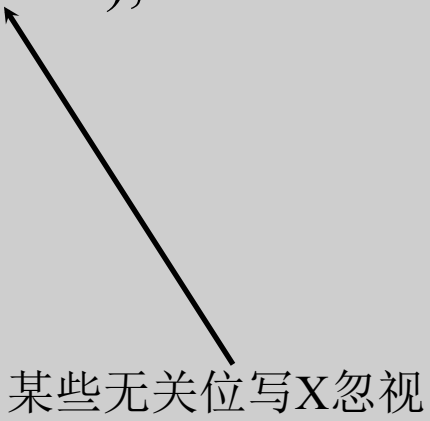
```
SUBDESIGN decode1
( a[3..0] : input;
  chip_enable : output;
)
begin
chip_enable = (a[3..0] == H"A");
end;
```

Self Explain the Function

唯一的改动

其他

```
SUBDESIGN decode1
( a[3..0] : input;
  chip_enable : output;
)
begin
chip_enable = (a[3..0] == B"1x0x");
end;
```



需要你懂得的某些事情

- 加： +
- 减：-
- 数字恒等：==
- 不等于：!=
- 不小于：>
- 不小于或等于：>=
- 不不小于：<
- 不不小于或等于：<=
- 逻辑或：#
- 逻辑与：&

用常数函数

- 假如相同的数值，字符串或数学体现式在一种文件出现几次就用常数
- 优点
  - 被变化只需申明一次

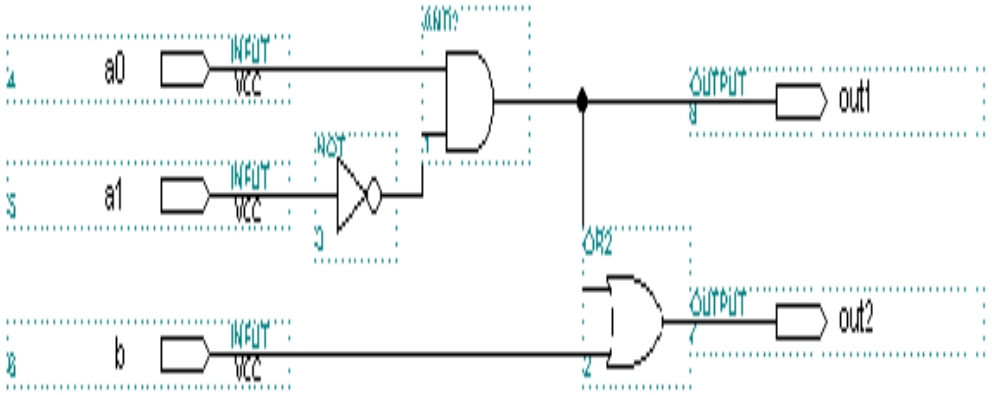
编辑一种地方就全部编辑

在 SUBDESIGN 关键字之前定义 CONSTANT

```
CONSTANT IO_ADDRESS = H"A";

SUBDESIGN decode1
( a[3..0] : input;
  chip_enable : output;
)
begin
chip_enable = (a[3..0] == IO_ADDRESS);
if (a[3..0] == IO_ADDRESS) then
……...
end;
```

实现组合逻辑



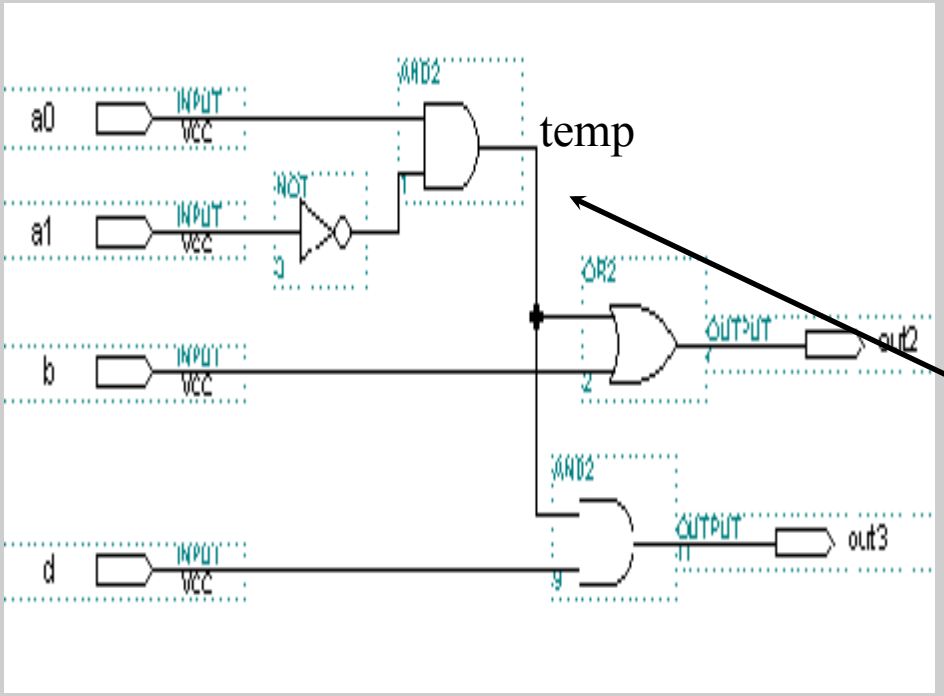
out1 = a0 & !a1

out2 = a0 & !a1 # b

AHDL ?

```
SUBSDESIGN decode1
( a0, a1, b : input;
  out1, out2 : output;
)
begin
out1 = a0 & !a1;
out2 = out1 # b;
end;
```

定义NODEs

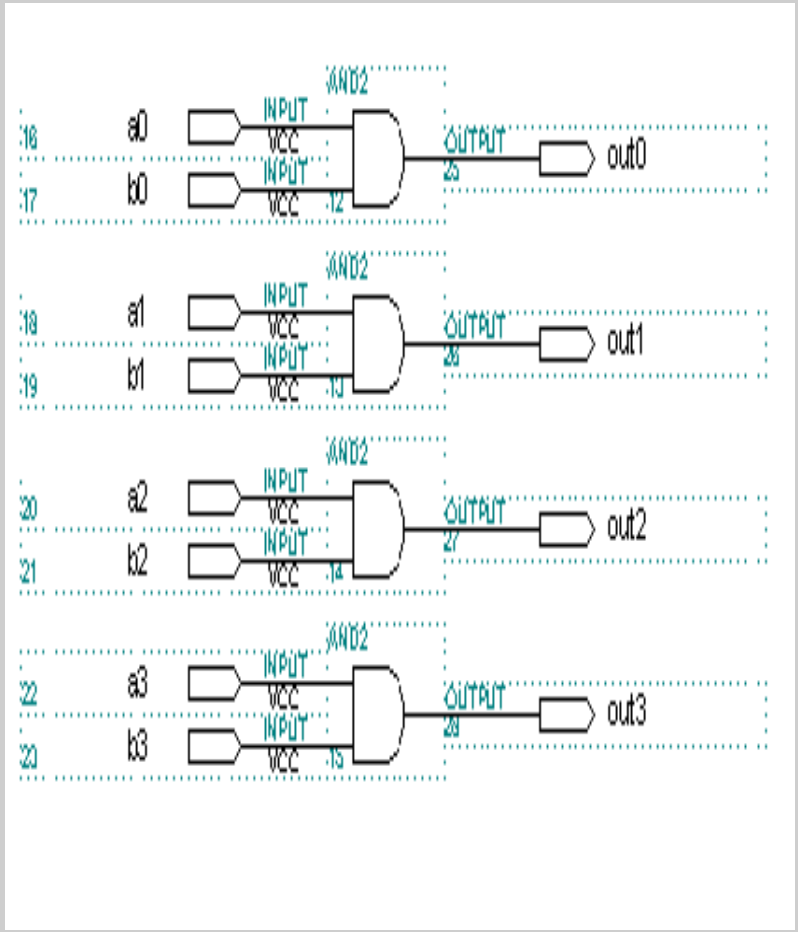


out2 = a0 & !a1 # b
out3 = a0 & !a1 & d

AHDL ?

```
SUBDESIGN decode1
( a0, a1, b, d: input;
  out2, out3 : output;
)
variable
temp : node;
begin
temp = a0 & !a1;
out2 = temp # b;
out3 = temp & d;
end;
```

总线操作




```
SUBDESIGN decode1
( a[3..0], b[3..0] : input;
   out[3..0] : output;
)
begin
out0 = a0 & b0;
out1 = a1 & b1;
out2 = a2 & b2;
out3 = a3 & b3;
end;
```

相同的功能
但是更轻易

```
SUBDESIGN decode1
( a[3..0], b[3..0] : input;
  out[3..0]: output;
)
begin
out1[] = a[] & b[];
end;
```

总线操作的更多内容

总线操作

a[9..0], b[9..0]

- a[] = b[];
- a[7..4] = b[9..6];
- a[9..8] = VCC;
- a[9..8] = 1;
- a[9..8] = 2;
- a[9..8] = 3;
- a[3..0] = GND
- a[3..0] = 0;
- temp = b0& b1;
  a[2..1] = temp

a7=b9, a6=b8, a5=b7, a4=b6

a[9..8] connect to VCC

a[9..8] = B”01”
a[9..8] = B”10”
a[9..8] = B”11”

a[3..0] connect to GND
a[3..0] = B”0000”

a2 = temp, a1 = temp

高级总线操作

- 总线Bus
  - b[3..0]
    - b3, b2, b1, b0 (有四个组员)
    - MSB 是 b3, LSB 是b0
- 阵列ARRAY BUS
  - a[3..0][2..0]
    - a3_2, a3_1, a3_0, a2_2, a2_1, a2_0, a1_2, a1_1, a1_0, a0_2, a0_1, a0_0 (有12个组员）
    - MSB 是 a3_2, LSB 是 a0_0

a[3..2][1..0] = b[];

这一种首先变化

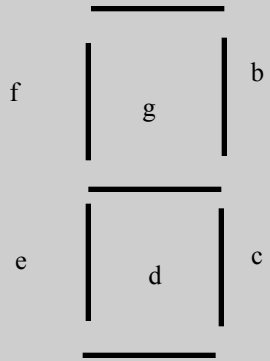
a3_1 = b3　　a3_0 = b2

a2_1 = b1　　a2_0 = b0

真值表

i[3..0]	Segment 7
0	0
1	1
2	2
F	F

a



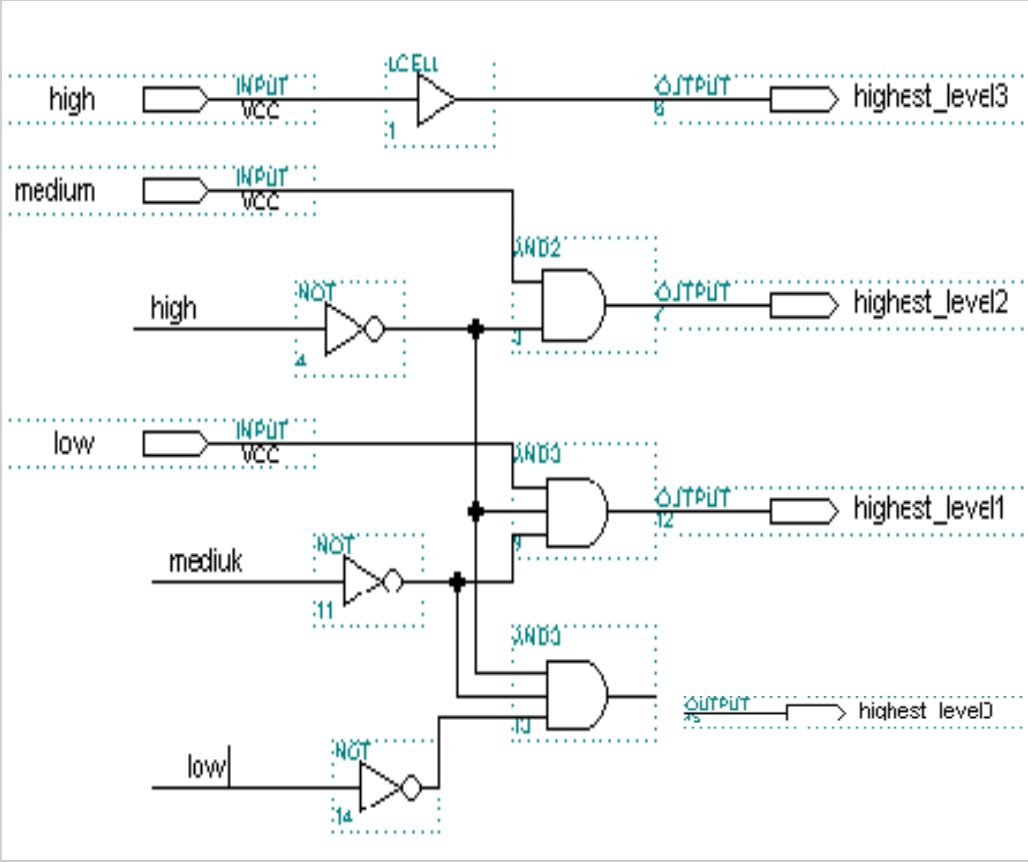

```
SUBDESIGN 7segment
(
    i[3..0]                    : INPUT;
    a, b, c, d, e, f, g : OUTPUT;
)
BEGIN
    TABLE
        i[3..0] =>  a, b, c, d, e, f, g;

        H"0"          =>  1, 1, 1, 1, 1, 1, 0;
        H"1"          =>  0, 1, 1, 0, 0, 0, 0;
        H"2"          =>  1, 1, 0, 1, 1, 0, 1;
        H"3"          =>  1, 1, 1, 1, 0, 0, 1;
        H"4"          =>  0, 1, 1, 0, 0, 1, 1;
        H"5"          =>  1, 0, 1, 1, 0, 1, 1;
        H"6"          =>  1, 0, 1, 1, 1, 1, 1;
        H"7"          =>  1, 1, 1, 0, 0, 0, 0;
        H"8"          =>  1, 1, 1, 1, 1, 1, 1;
        H"9"          =>  1, 1, 1, 1, 0, 1, 1;
        H"A"          =>  1, 1, 1, 0, 1, 1, 1;
        H"B"          =>  0, 0, 1, 1, 1, 1, 1;
        H"C"          =>  1, 0, 0, 1, 1, 1, 0;
        H"D"          =>  0, 1, 1, 1, 1, 0, 1;
        H"E"          =>  1, 0, 0, 1, 1, 1, 1;
        H"F"          =>  1, 0, 0, 0, 1, 1, 1;
    END TABLE;
END;
```

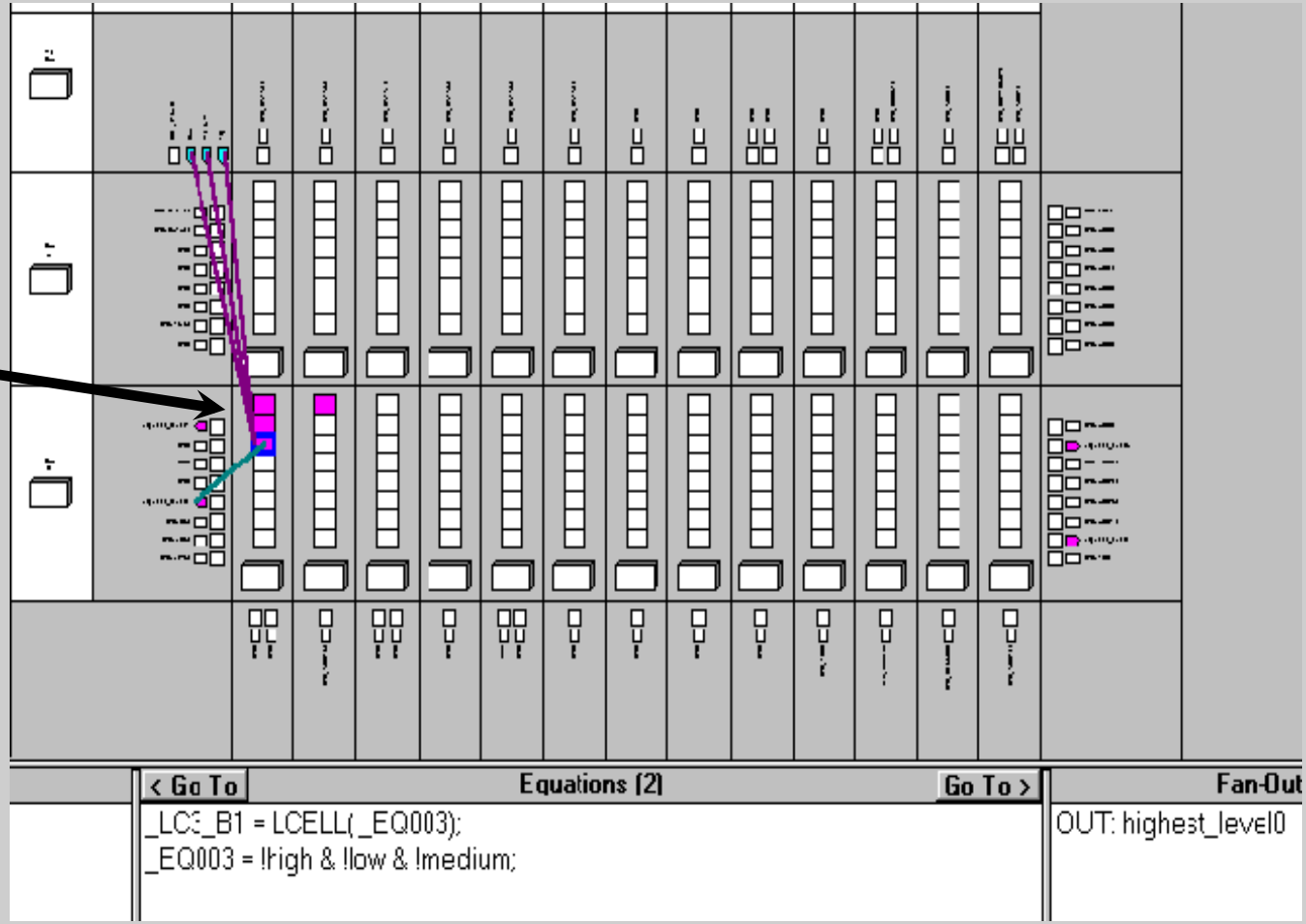
做任何改动都很轻易

IF-THEN-ELSE

```
SUBDESIGN priority
( low, medium, high : input;
   highest_level[3..0] : output;
)
begin
if ( high == B”1”) then
highest_level[] = B”1000”;
elsif (medium == B”1”) then
highest_level[] = B”0100”;
elsif (low == B”1”) then
highest_level[] = B”0010”;
else
highest_level[] = B”0001”;
end if;
end;
```

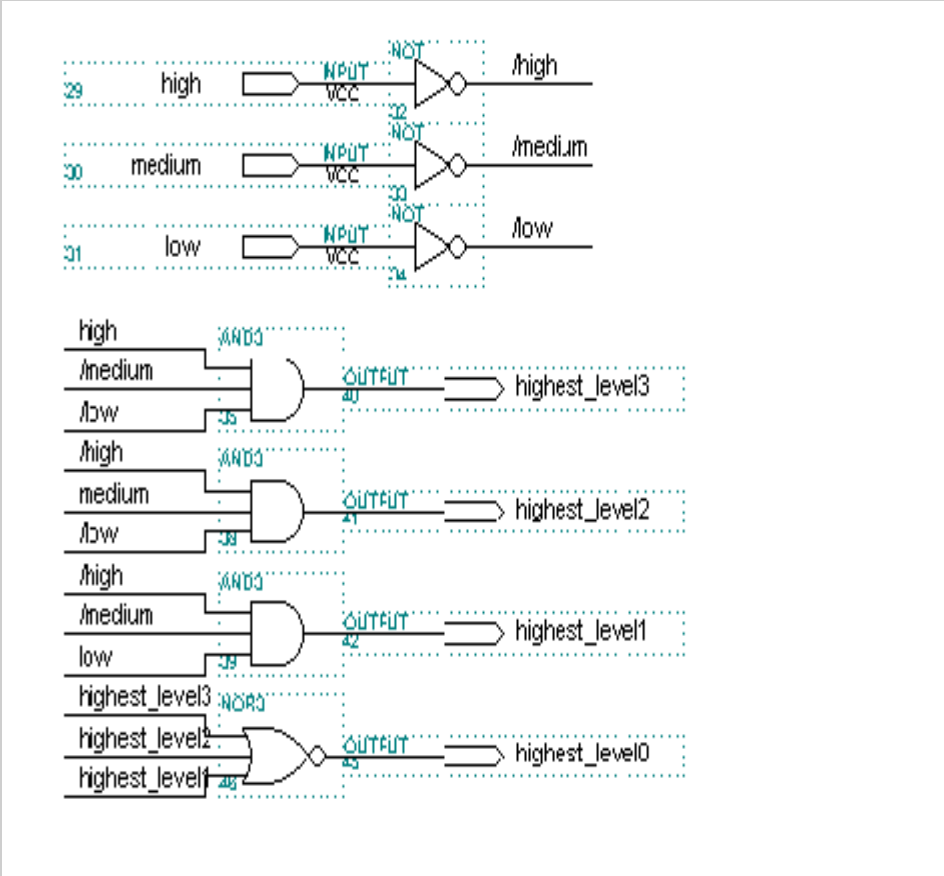


需要 4
LCELL


< Go To
Equations (2)
Go To >
Fan-Out
_LC3_B1 = LCELL(_EQ003);
_EQ003 = !high & !low & !medium;
OUT: highest_level0

CASE 段

```
SUBDESIGN decoder
(low, medium, high : input;
  highest_level[3..0] : output;
)
variable
code[2..0] : node;
begin
code2=high;
code1=medium;
code0=low;
case code[] is
when 4 => highest_level[] = B"1000";
when 2 => highest_level[] = B"0100";
when 1 => highest_level[] = B"0010";
when others => highest_level[] = B"0001";
end case;
end;
```



这个段落怎么用