# 最经工作工作总结1.0

--王馨

一、工作总结

目前毕设已完成的工作有：

1. 主逻辑判断函数状态图化简
2. 二进制编码模块编写
3. BCD译码模块编写

二、主逻辑状态图化简

结合状态图化简原则和自动售货机的实际情况，经过仔细研究初始计划的状态图可以进一步化简从而降低编码的复杂度。

主要化简思想为：将三种不同状态的投币情况进行化简，结合NFA状态图加减原则，进行化简后，投币状态可以完全合并成一个，最终化简后的主逻辑状态图如下：



其中：Sa表示自动售货机起始状态 Sb表示用户选择模块 S0表示用户投币状态 S1表示出货状态 S2表示找零状态 S3表示交易失败模块。

即主逻辑的流程情况大体如下：

当入户选择开始选择货物时候，进入Sb状态，用户可以选择自己需要的各种货物，当用户在此状态下长期不选择货物或者选择货物失败的情况下自动售货机回归到Sa起始状态。当用户结束选择货物时候进入用户投币模块即S0，用户在正常投币结束后进入出货状态即S1，当用户投币异常情况下，进入交易失败模块S3。当出货完成后，自动售货机根据实际情况选择是否进行找零，当需要找零时进入S2状态，当不需要找零时直接返回到Sa状态。最后所有状态均会回到最初始状态Sa。

三、二进制编码模块

二进制译码模块主要实现一个四位二进制转换成两个四位BCD码。其转换规则如下：

|  |  |  |
| --- | --- | --- |
| 二进制 | 第一个BCD码 | 第二个BCD码 |
| 0000 | 0000 | 0000 |
| 0001 | 0000 | 0001 |
| 0010 | 0000 | 0010 |
| 0011 | 0000 | 0011 |
| 0100 | 0000 | 0100 |
| 0101 | 0000 | 0101 |
| 0110 | 0000 | 0110 |
| 0111 | 0000 | 0111 |
| 1000 | 0000 | 1000 |
| 1001 | 0000 | 1000 |
| 1010 | 0001 | 0000 |
| 1011 | 0001 | 0001 |
| 1100 | 0001 | 0010 |
| 1101 | 0001 | 0011 |
| 1110 | 0001 | 0100 |
| 1111 | 0001 | 0101 |

实现代码如下：

Library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity binarycoder is

port(

b:in std\_logic\_vector(3 downto 0);

bcd0:out std\_logic\_vector(3 downto 0);

bcd1:out std\_logic\_vector(3 downto 0)

);

end binarycoder;

architecture one of binarycoder is

begin

process(b)

begin

case b is

when"0000"=>bcd0<="0000";bcd1<="0000";

when"0001"=>bcd0<="0001";bcd1<="0000";

when"0010"=>bcd0<="0010";bcd1<="0000";

when"0011"=>bcd0<="0011";bcd1<="0000";

when"0100"=>bcd0<="0100";bcd1<="0000";

when"0101"=>bcd0<="0101";bcd1<="0000";

when"0110"=>bcd0<="0110";bcd1<="0000";

when"0111"=>bcd0<="0111";bcd1<="0000";

when"1000"=>bcd0<="1000";bcd1<="0000";

when"1001"=>bcd0<="1001";bcd1<="0000";

when"1010"=>bcd0<="0000";bcd1<="0001";

when"1011"=>bcd0<="0001";bcd1<="0001";

when"1100"=>bcd0<="0010";bcd1<="0001";

when"1101"=>bcd0<="0011";bcd1<="0001";

when"1110"=>bcd0<="0100";bcd1<="0001";

when"1111"=>bcd0<="0101";bcd1<="0001";

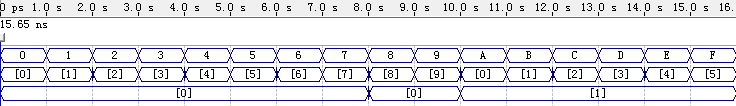
when others=>null;

end case;

end process;

end one;

仿真图如下：



四、BCD译码模块

BCD译码模块主要是将得到的BCD码转换成可以用数码管显示的格式，B数字与数码管对应编码如下：

|  |  |  |
| --- | --- | --- |
| 数字 | 编码 | 十六进制 |
| 0 | 1111110 | 7E |
| 1 | 0110000 | 30 |
| 2 | 1101101 | 6D |
| 3 | 1111001 | 79 |
| 4 | 0110011 | 33 |
| 5 | 1011011 | 5B |
| 6 | 1011111 | 5F |
| 7 | 1110000 | 70 |
| 8 | 1111111 | 7F |
| 9 | 1110011 | 73 |

代码实现如下：

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity bcdcoder is

port(d:in std\_logic\_vector(3 downto 0);

q:out std\_logic\_vector(6 downto 0));

end bcdcoder;

architecture one of bcdcoder is

begin

process(d)

begin

case d is

when"0000"=>q<="1111110";

when"0001"=>q<="0110000";

when"0010"=>q<="1101101";

when"0011"=>q<="1111001";

when"0100"=>q<="0110011";

when"0101"=>q<="1011011";

when"0110"=>q<="1011111";

when"0111"=>q<="1110000";

when"1000"=>q<="1111111";

when"1001"=>q<="1110011";

when others=>q<="0000000";

end case;

end process;

end one;

仿真图如下：

