数字电压表设计

指导老师：郑海永

组长：江克洲 12020021025 电子信息科学与技术

组员：董铭杰 12020021020 电子信息科学与技术

郭伟博 12020021023 电子信息科学与技术

联系邮箱：925776878@qq.com

摘要：应用 VHDL 语言编程，进行了数字电压表的设计，并在 Xilinx ISE14.6 环境下通过了编译、仿真、调试。

关键词：VHDL 语言；FPGA 芯片；数字电压表；A/D 采集

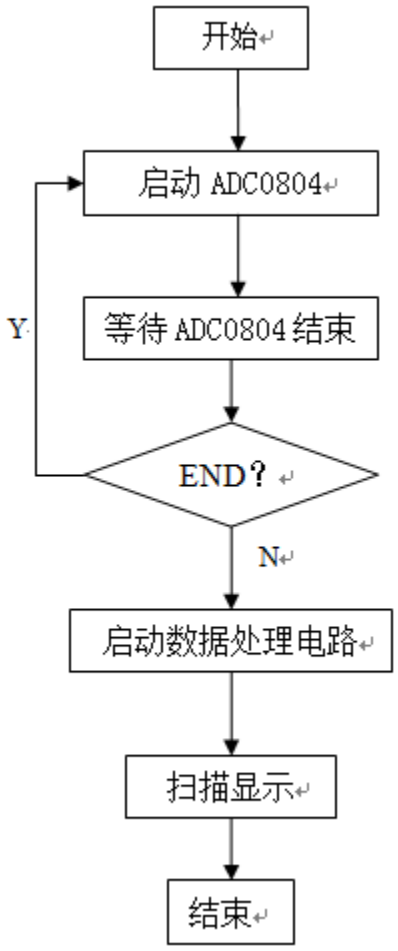
引言：数字电压表的高速发展，使它成为实现测量自动化、提高工作效率不可缺少的仪表。它具有显示清晰直观，读数准确；准确度高；分辨率高；测量范围宽；扩展能力强；测量速率快；抗干扰力强等优点。 近 40 年来，我国除了研制生产了大批量数字仪表外，还从国外进口了大量的高准确度的数字仪表，由于缺乏计量管理，使用不当，量值转递不统一，损坏返修和待测工作量很大，甚至造成了和大的经济损失。

数字电压表的应有十分广泛，其准确度也越来越高。因此，研究它的检定测试方法，分析掌握它的工作原理和技术指标，是计量测试部门、科研单位、生产厂家、使用和维修单位的一项重要技术工作。为了充分发挥数字电压表的许多突出优点，必须掌握它的技术性、鉴定方法和测量技术。 数字电子系统通常由传统的用 ASIC 和以微处理器为控制的 A/D 转换数字电压表其速度和灵活性不能满足日益发展的电子工业的需求。FPGA 最大的特点就是灵活，实现你想实现的任何数字电路，可以定制各种电路。减少受制于专用芯片的束缚。真正为自己的产品量身定做。在设计的过程中可以灵活的更改设计。而且它强大的逻辑资源和寄存器资源可以让你轻松的去发挥设计理念，其并行执行，硬件实现的方式可以应对设计中大量的高速电子线路设计需求。所以用 FPGA 芯片控制的 A/D 转换器其速度、灵活性可大大提高，得到日益广泛的应用。

实验流程：由于 FPGA 开发板有一定的限制，所以只是进行了简单的测试。我们分成三大模块来做：A/D 转换控制模块，即负责 A/D 转换器的启动、地址锁存、

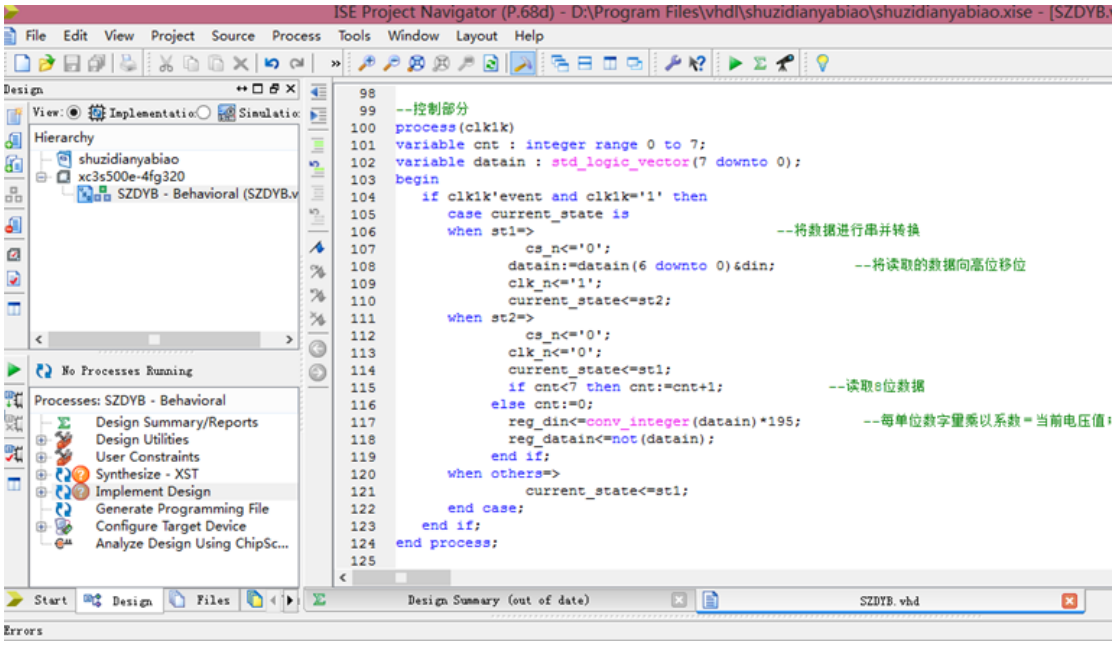
输入通道选择、数据读取等工作。数据转换模块，即将 8 位二进制数据转换成 16 位十进制 BCD 码送入动态扫描与译码模块。动态扫描与译码模块，即编译传输过来的代码，并向外部数码管显示电路输出数据。

下图是实验流程图

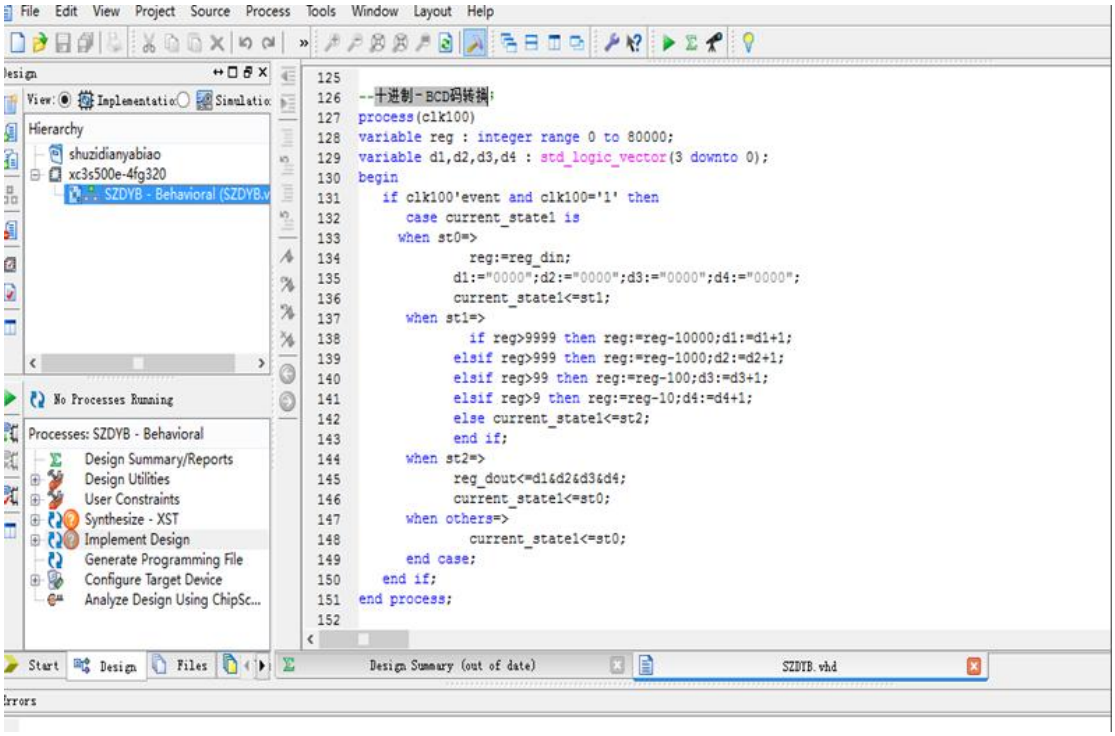


实验分析：三个模块的代码

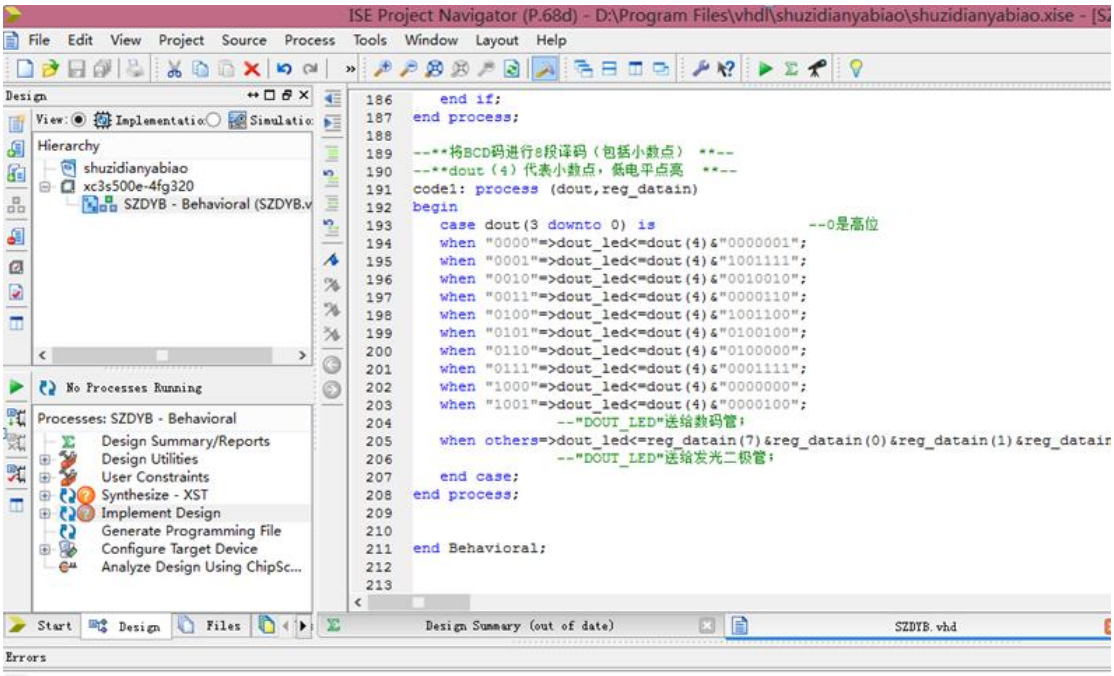
A/D 转换控制模块



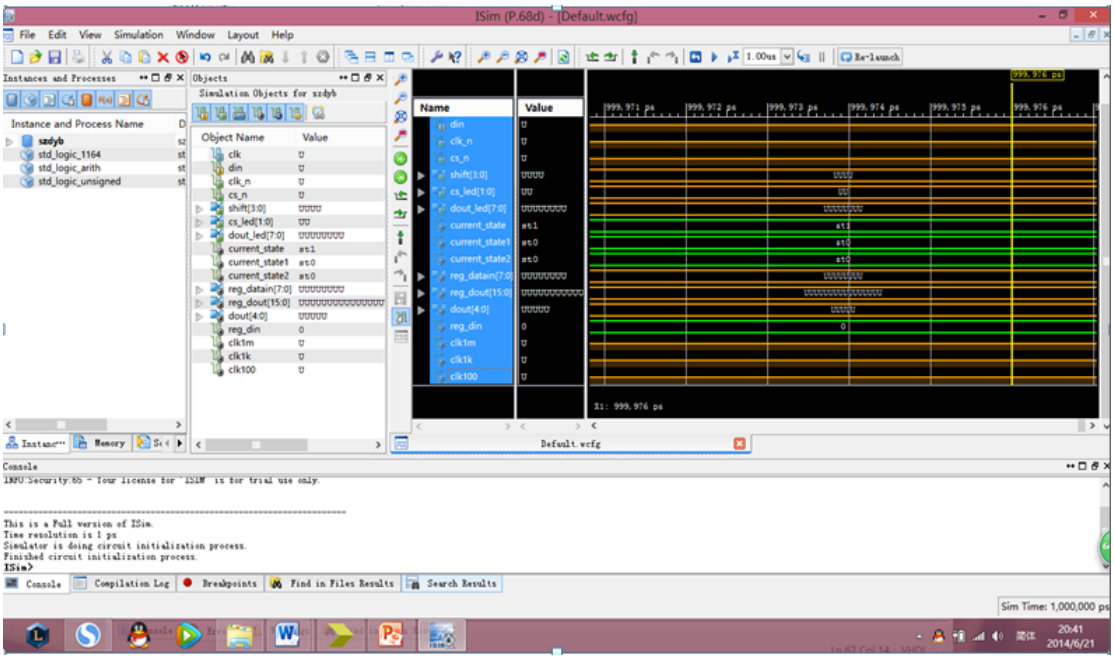
数据转换模块



动态扫描与译码模块



最后根据实验波形仿真



总结：

通过对 FPGA 芯片进行 VHDL 语言编程，实现了数字电压表的功能。通过设计表明，VHDL 具有很强的电路描述和建模能力，能从多个层次对数字系统进行建模和描述，从而大大简化了硬件设计任务，提高了设计效率和可靠性。这种设计方法必将在未来的数字系统设计中发挥越来越重要的作用。而且在这次的实践操作中我们也发现了我们的知识面还很狭隘，而且很多网上的事情并不能完全相信，需要我们去核对，整理。

致谢： 感谢老师给我们一个动手实践的机会，交给我们很多方法和许多课本上学不到

的知识，让我们更能认识到我们的不足之处。同时也感谢在平台上发 XiLinx 软件的同学。

附录：

```
LIBRARY ieee; --A/D0809
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity ad is
port(ST,eoc:instd_logic; --控制端口
d:in std_logic_vector(7 downto 0);
oe,sta,ale,adda:outstd_logic;
q:out std_logic_vector(7 downto 0));
end ad;

architecture a of ad is
type states is(st0,st1,st2,st3,st4,st5,st6); --7 个状态
signal c_state,n_state:states :=st0;
signal regl:std_logic_vector(7 downto 0);
signal lock:std_logic;
begin
adda<='1';
com:process(c_state,eoc)
begin
case c_state is
when st0=>ale<='0';sta<='0';oe<='0';lock<='0';
n_state<=st1;
when st1=>ale<='1';sta<='0';oe<='0';lock<='0';
n_state<=st2;
when st2=>ale<='0';sta<='1';oe<='0';lock<='0';
n_state<=st3;
when st3=>ale<='0';sta<='0';oe<='0';lock<='0';
if(eoc='1') then n_state<=st4;
else n_state<=st3; ——eoc 为‘1’转换结束进入下一状态
end if; ——否则继续转换
when st4=>ale<='0';sta<='0';oe<='1';lock<='0';
n_state<=st5;
when st5=>ale<='0';sta<='0';oe<='0';lock<='1';
```

```
n_state<=st6;
when st6=>ale<='0';sta<='0';oe<='0';lock<='1';
n_state<=st0;
when others=>n_state<=st0;
end case;
end process com;
reg:process(st)
begin
if(st'event and st='1') then
c_state<=n_state;
end if;
end process reg;
lo:process(lock) --锁存
begin
if(lock'event and lock='1') then
regl<=d;
end if;
end process lo;
q<=regl;
end a;


——BCD 8 位转 12
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY BCD IS
PORT (V:IN STD_LOGIC_VECTOR(7 DOWNTO 0);
HB,LB:BUFFER STD_LOGIC_VECTOR(11 DOWNTO 0);
BVALUE:BUFFER STD_LOGIC_VECTOR(11 DOWNTO 0);
BCD_L,BCD_M,BCD_H:OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END BCD;
ARCHITECTURE A OF BCD IS
BEGIN
P1:PROCESS(V(7 DOWNTO 4))
BEGIN --A/D 输出高 4 位转换分辨率 0.32V
IF V(7 DOWNTO 4)="1111" THEN HB<="010010000000"; --4.80V
ELSIF V(7 DOWNTO 4)= "1110" THEN HB<="010001001000";--4.48V
```

```
ELSIF V(7 DOWNTO 4)= "1101" THEN HB<="010000010110";--4.16V
ELSIF V(7 DOWNTO 4)= "1100" THEN HB<="001110000100";--3.84V
ELSIF V(7 DOWNTO 4)= "1011" THEN HB<="001101010010";--3.52V
ELSIF V(7 DOWNTO 4)= "1010" THEN HB<="001100100000";--3.20V
ELSIF V(7 DOWNTO 4)= "1001" THEN HB<="001010001000";--2.88V
ELSIF V(7 DOWNTO 4)= "1000" THEN HB<="001001010110";--2.56V
ELSIF V(7 DOWNTO 4)= "0111" THEN HB<="001000100100";--2.24V
ELSIF V(7 DOWNTO 4)= "0110" THEN HB<="000110010010";--1.92V
ELSIF V(7 DOWNTO 4)= "0101" THEN HB<="000101100000";--1.60V
ELSIF V(7 DOWNTO 4)= "0100" THEN HB<="000100101000";--1.28V
ELSIF V(7 DOWNTO 4)= "0011" THEN HB<="000010010110";--0.96V
ELSIF V(7 DOWNTO 4)= "0010" THEN HB<="000001100100";--0.64V
ELSIF V(7 DOWNTO 4)= "0001" THEN HB<="000000110010";--0.32V
ELSIF V(7 DOWNTO 4)= "0000" THEN HB<="000000000000";--0.00V
ELSE HB<="000000000000" ; --0.00V
END IF;
END PROCESS P1;

P2:PROCESS(V(3 DOWNTO 0))
BEGIN --A/D 输出低 4 位转换分辨率 0.02V
IF V(3 DOWNTO 0)= "1111" THEN LB<="000000110000"; --0.30V
ELSIF V(3 DOWNTO 0)= "1110" THEN LB<="000000101000";--0.28V
ELSIF V(3 DOWNTO 0)= "1101" THEN LB<="000000100110";--0.26V
ELSIF V(3 DOWNTO 0)= "1100" THEN LB<="000000100100";--0.24V
ELSIF V(3 DOWNTO 0)= "1011" THEN LB<="000000100010";--0.22V
ELSIF V(3 DOWNTO 0)= "1010" THEN LB<="000000100000";--0.20V
ELSIF V(3 DOWNTO 0)= "1001" THEN LB<="000000011000";--0.18V
ELSIF V(3 DOWNTO 0)= "1000" THEN LB<="000000010110";--0.16V
ELSIF V(3 DOWNTO 0)= "0111" THEN LB<="000000010100";--0.14V
ELSIF V(3 DOWNTO 0)= "0110" THEN LB<="000000010010";--0.12V
ELSIF V(3 DOWNTO 0)= "0101" THEN LB<="000000010000";--0.10V
ELSIF V(3 DOWNTO 0)= "0100" THEN LB<="000000001000";--0.08V
ELSIF V(3 DOWNTO 0)= "0011" THEN LB<="000000000110";--0.06V
ELSIF V(3 DOWNTO 0)= "0010" THEN LB<="000000000100";--0.04V
ELSIF V(3 DOWNTO 0)= "0001" THEN LB<="000000000010";--0.02V
ELSIF V(3 DOWNTO 0)= "0000" THEN LB<="000000000000";--0.00V
ELSE LB<="000000000000"; --0V
END IF;
END PROCESS P2;
```

```
BVALUE<=HB+LB;
P3:PROCESS(BVALUE)
VARIABLE JJ:STD_LOGIC_VECTOR(11 DOWNTO 0);
BEGIN
JJ:=BVALUE;
IF (JJ(3 DOWNTO 0)>"1001") THEN ——如果 12 位结果中，低 4 位
JJ:=JJ+"000000000110"; ——大于 9 则低 4 位加 6
END IF;
IF(JJ(7 DOWNTO 4)>"1001") THEN ——如果中间的 4 位大于 9
JJ:=JJ+"000001100000"; ——则中 4 位加 6
END IF;
BCD_L<=JJ(3 DOWNTO 0);
BCD_M<=JJ(7 DOWNTO 4);
BCD_H<=JJ(11 DOWNTO 8);
END PROCESS P3;
END A;

——3 选 1 数据选择器
LIBRARY ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity mux3_1 is
port(sel:instd_logic_vector(1 downto 0);
A,B,C:instd_logic_vector(3 downto 0);
Mselout:outstd_logic_vector(3 downto 0));
end mux3_1;

architecture a of mux3_1 is
begin
process(sel)
begin
if sel="10" then Mselout<=A;
elsifsel="01" then Mselout<=B;
elsifsel="00" then Mselout<=C;
else null;
end if;
end process;
end a;
```

——位选信号产生器（3 进制计数器）

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity c3 is
port(clk,clr:instd_logic;
qout:bufferstd_logic_vector(1 downto 0)
);
end c3;
architecture behave of c3 is
begin
process(clk,clr)
begin
if(clr='0')then qout<="00";
elsif(clk'event and clk='1')then
qout<=qout+1;
if(qout=2)then qout<="00";
end if;
end if;
end process;
end behave;
```

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
ENTITY del7 IS
PORT
( input : IN STD_LOGIC_vector(3 downto 0);
output : OUT STD_LOGIC_vector(6 downto 0)
);
END del7;

ARCHITECTURE a OF del7 IS
BEGIN
PROCESS (input)
BEGIN
CASE input IS
WHEN "0000" =>output<="1111110";
WHEN "0001" =>output<="0110000";
```

```
WHEN "0010" =>output<="1101101";
WHEN "0011" =>output<="1111001";
WHEN "0100" =>output<="0110011";
WHEN "0101" =>output<="1011011";
WHEN "0110" =>output<="1011111";
WHEN "0111" =>output<="1110000";
WHEN "1000" =>output<="1111111";
WHEN "1001" =>output<="1111011";
WHEN OTHERS=>NULL;
END CASE;
end process;
END a;
```

——小数点产生器

```
LIBRARY ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity DP is
port(SELDP:instd_logic_vector(1 downto 0);
DPout:outstd_logic);
end DP;

architecture a of DP is
begin
process(selDP)
begin
if selDP="10" then DPout<='0';
elsifselDP="01" then DPout<='0';
elsifselDP="00" then DPout<='1'; ——在高 4 位整数输出时，输出
else null; ——小数点 DP
end if;
end process;
end a;
```

——顶层文件

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
```

```
use ieee.std_logic_arith.all;
entity V_WATCH is
port( clkK,EOCC:instd_logic;
DD:IN std_logic_vector(7 downto 0);
clk3,CLR3:IN STD_LOGIC;
OEE,START,ALEE,ADDAA:OUT STD_LOGIC;
DPOUT:OUT STD_LOGIC;
Qoutput:outstd_logic_vector(6 downto 0);
Qselout:outstd_logic_vector(1 downto 0));
end V_WATCH;
architecture a of V_WATCH is ——元件例化
COMPONENT DP
port(SELDP:instd_logic_vector(1 downto 0);
DPout:outstd_logic);
END COMPONENT;
COMPONENT del7
PORT(input : IN STD_LOGIC_vector(3 downto 0);
output : OUT STD_LOGIC_vector(6 downto 0));
END COMPONENT;
COMPONENT mux3_1
port(sel:instd_logic_vector(1 downto 0);
A,B,C:instd_logic_vector(3 downto 0);
Mselout:outstd_logic_vector(3 downto 0));
END COMPONENT;
COMPONENT ad
port(st,eoc:instd_logic;
d:in std_logic_vector(7 downto 0);
oe,sta,ale,adda:outstd_logic;
q:out std_logic_vector(7 downto 0));
END COMPONENT;
COMPONENT C3
PORT
(clk,clr: IN STD_LOGIC;
qout: OUT std_logic_vector(1 downto 0));
END COMPONENT;
COMPONENT BCD
PORT (V:IN STD_LOGIC_VECTOR(7 DOWNTO 0);
HB,LB:BUFFER STD_LOGIC_VECTOR(11 DOWNTO 0);
BVALUE:BUFFER STD_LOGIC_VECTOR(11 DOWNTO 0);
```

```
BCD_L,BCD_M,BCD_H:OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END COMPONENT;

  signal selOUT: std_logic_vector(1 downto 0);
      SIGNAL     SBCD_L,SBCD_M,SBCD_H:     STD_LOGIC_VECTOR(3
      DOWNTO 0);
      SIGNAL DI: STD_LOGIC_VECTOR(3 DOWNTO 0);
      SIGNAL QQ: STD_LOGIC_VECTOR(7 DOWNTO 0);
      SIGNAL BBCD: STD_LOGIC_VECTOR(11 DOWNTO 0);
      BEGIN
      U1:AD PORT MAP(CLKK,EOCC,DD,OEE,START,ALEE,ADDAA,QQ);
      U2:BCD                                                PORT
      MAP(QQ,BCD_L=>SBCD_L,BCD_M=>SBCD_M,BCD_H=>SBCD_H);
      U3:C3 PORT MAP(CLK3,CLR3,SELOUT);
      U4:MUX3_1 PORT MAP(SELOUT,SBCD_L,SBCD_M,SBCD_H,DI);
      U5:DEL7 PORT MAP(DI,Qoutput);
      U6:DP PORT MAP(SELOUT,DPOUT);
      QSELOUT<=SELOUT;
      end a;
```