习题

1. 按照下列指令格式，写出可由微处理器执行的计算 X＝A＋（B*C）＋D 的代码。不能修改 A，B，C 和 D 的值。如果需要，可用临时单元 T 存储中间结果。

a) 三操作数指令
b) 二操作数指令
c) 一操作数指令
d) 零操作数指令

解：

a)
```
MUL X,B,C
ADD X,X,A
ADD X,X,D
```

b)
```
MOV X,B
MUL X,C
ADD X,A
ADD X,D
```

c)
```
LOAD B
MUL C
ADD A
ADD D
STORE X
```

d)
```
PUSH A
PUSH B
PUSH C
MUL
PUSH D
ADD
ADD
POP X
```

2. 为相对简单 CPU 写一段程序，将内存单元 1001H 到 100AH 中的 10 个数值相加并且将结果存在内存单元 1000H 中。假定结果总是小于 256。

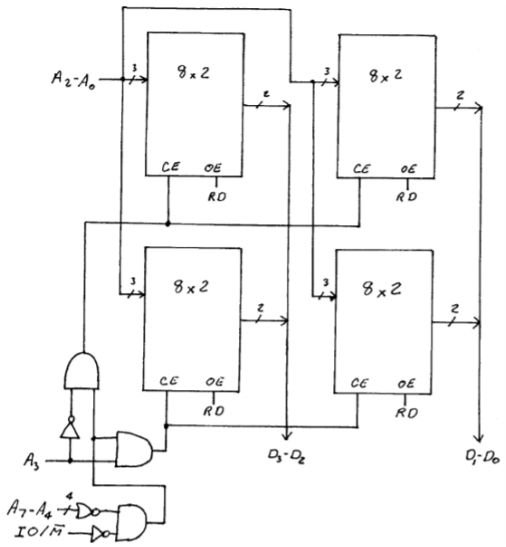
解：

```
LDAC 1001H
MVAC
LDAC 1002H
ADD
MVAC
LDAC 1003H
ADD
MVAC
LDAC 1004H
ADD
MVAC
LDAC 1005H
ADD
MVAC
```

```
LDAC 1006H
ADD
MVAC
LDAC 1007H
ADD
MVAC
LDAC 1008H
ADD
MVAC
LDAC 1009H
ADD
MVAC
LDAC 100AH
ADD
STAC 1000H
```

3. 用 8×2 的存储器芯片、为一个有 8 位地址总线的计算机设计一个 16×4 的存储器子系统，该子系统带高位交叉的地址。

解：



4. 当 X = 1001 1001 0000 0010 时，执行下列操作的结果分别为多少？
   a) shl(X)
   b) shr(X)
   c) cil(X)
   d) cir(X)
   e) ashl(X)
   f) ashr(X)
   g) dshl(X)
   h) dshr(X)

解：shl：逻辑左移，低位补 0；

shr ：逻辑右移，高位补 0；

cil:循环左移，移出的数据补到低位；

cir:循环右移，移出的数据补到高位；

ashl:算数左移，符号位保持不变，符号位之后的数据左移，低位补 0；

ashr:算数右移，符号位保持不变，符号位之后的数据右移，高位补符号位；

dshl:十进制左移，即左移 4 位，低位补 0；

dshr:十进制右移，即右移 4 位，高位补 0；

按照上述规则，得到的结果如下：

a) 0011 0010 0000 0100
b) 0100 1100 1000 0001
c) 0011 0010 0000 0101
d) 0100 1100 1000 0001
e) 1011 0010 0000 0100
f) 1100 1100 1000 0001
g) 1001 0000 0010 0000
h) 0000 1001 1001 0000

5. 设计作为数字电路的 VHDL 以实现收费站控制器。

解：

```
library IEEE;
use IEEE.std_logic_1164.all;

entity toll_booth_controller is
   port(
      I1,I0,C,clk: in std_logic;
      X3,X2,X1,X0: buffer std_logic;
            R,G,A: out std_logic
   );
end toll_booth_controller;

architecture a_toll_booth_controller of toll_booth_controller is
begin
   cct_toll_booth_controller: process(X3,X2,X1,X0,I1,I0,C,clk)
      begin
         if rising_edge(clk) then
            X3 <= not C;
            X2 <= ((not X3) and C and I1 and I0) or
                  ((not X3) and (X2 or X1) and C and I1 and (not I0)) or
                  ((not X3) and (X2 or (X1 and X0)) and C and (not I1) and I0)
                    or (X2 and C and (not I1) and (not I0));
            X1 <= ((not X3) and (X2 or X1 or X0) and C and I1 and I0)or
                  ((not X3) and (X2 or (not X1)) and C and I1 and (not I0)) or
                  ((not X3) and (((not X1) and X0) or (X1 and (not X0)) or
                   (X2 and X1 and X0)) and C and (not I1) and I0) or
                  (X1 and X0 and C and (not I1) and (not I0));
            X0 <= ((not X3) and (X2 or X1 or (not X0)) and C and I1 and I0) or
                  ((not X3) and (X0 or (X2 and X1)) and C and I1 and (not I0))
                   or ((not X3) and ((not X0) or (X2 and X1)) and C and
                   (not I1) and I0) or ((not X3) and X0 and C and (not I1) and
                   (not I0)) or (X3 and X0 and (not C)) or ((not X3) and
                   ((not X2) or (not X1) or (not X0)) and (not C));
         end if;
          R <= X3 or (not X2) or (not X1) or (not X0);
          G <= (not X3) and X2 and X1 and X0;
          A <= X3 and (not X2) and (not X1) and X0;
      end process cct_toll_booth_controller;
end a_toll_booth_controller;
```

6. 给出相对简单 CPU 用来产生寄存器 PC，DR，TR 和 IR 的控制信号所必需的逻辑。

解：

$$PCLOAD = JUMP3 \vee JMPZY3 \vee JPNZY3$$
$$PCINC = FETCH2 \vee LDAC1 \vee LDAC2 \vee STAC1 \vee STAC2 \vee JMPZN1 \vee JMPZN2 \vee JPNZN1 \vee JPNZN2$$
$$DRLOAD = FETCH2 \vee LDAC1 \vee LDAC2 \vee LDAC4 \vee STAC1 \vee STAC2 \vee STAC4 \vee JUMP1 \vee JUMP2 \vee JMPZY1 \vee JMPZY2 \vee JPNZY1 \vee JPNZY2$$
$$TRLOAD = LDAC2 \vee STAC2 \vee JUMP2 \vee JMPZY2 \vee JPNZY2$$
$$IRLOAD = FETCH3$$

7. 给出相对简单 CPU 用来产生 ALU 的控制信号所必需的逻辑。

解：

State	ALUS[1..7]
LDAC5	0 0 1 0 X X 0
MOVR1	0 0 1 0 X X 0
ADD1	1 0 1 0 X X 0
SUB1	1 1 0 0 X X 0
INAC1	1 0 0 1 X X 0
CLAC1	0 0 0 0 X X 0
AND1	X X X X 0 0 1
OR1	X X X X 0 1 1
XOR1	X X X X 1 0 1
NOT1	X X X X 1 1 1

$$ALUS1 = ADD1 \vee SUB1 \vee INAC1$$
$$ALUS2 = SUB1$$
$$ALUS3 = LDAC5 \vee MOVR1 \vee ADD1$$
$$ALUS4 = SUB1 \vee INAC1$$
$$ALUS5 = XOR1 \vee NOT1$$
$$ALUS6 = OR1 \vee NOT1$$
$$ALUS7 = AND1 \vee OR1 \vee XOR1 \vee NOT1$$

8. 给出下列非负数码操作的结果。
   a) 1011 0100－0111 0111
   b) 0011 1000＋1100 1101
   c) 1000 1011＋0111 0100
   d) 0111 0100－1000 1011

解：

a) 180-119=61

b) 56+205=261>2^8=256，溢出

c) 139+11=255

d) 116-139= -23，得到负数

9. 给出下列补码操作的结果
   a) 1011 0100－0111 0111
   b) 0011 1000＋1100 1101
   c) 1000 1011＋0111 0100
   d) 0111 0100－1000 1011

解：最高位作为符号位——“1”表示负数，“0”表示正数，所以得到：

a）-76-119=-193

b）56+（-51）=5

c）-117+116=-1

d）116-（-117）=233，由于最大正数为 2^7-1=128-1=127，所以 233 已经溢出

10. 一台基于相对简单 CPU 的计算机，有一个 16 个字的相联 cache，采用 FIFO 的替换策略。给出在执行下列代码过程中及执行完代码后 cache 的内容。同时求这段程序在这个系统中的命中率。

```
0：LDAC 4234
3:  STAC 4235
6:  MVAC
7:  INAC
8:  ADD
9:  JPNZ 0020
C:  JUMP 0010
F:  NOP
10: CLAC
11: JUMP 0020
20: LDAC 4235
23: JUMP 0029
26: JUMP 0000
29: AND
4235: 55
```

解：执行过程如下：

指令	Tag	Data	Valid	Dirty	命中情况
LDAC 4234	0000 0001	01 34	1 1	0 0	

	0002 4234	42 55	1 1	0 0	
STAC 4235	0003 0004 0005 4235	0B 35 42 55	1 1 1 1	0 0 0 1	
MVAC	0006	03	1	0	
INAC	0007	0A	1	0	
ADD	0008	08	1	0	
JPNZ 0020	0009 000A 000B	07 20 00	1 1 1	0 0 0	
LDAC 4235	0020 0021 0022 4235	01 35 42 55	1 1 1 1	0 0 0 1	命中
JUMP 0029	0023 0024 0025	05 29 00	1 1 1	0 0 0	
AND	0029	0C	1	0	

由此可知命中率为：1/23*100%=4.5%