**实验三预习报告**

**201914020128 计科1903 陈旭**

* **函数发生器(ALU)**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

entity alu is

    port(

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

        M: in std\_logic;

        inx, iny: in std\_logic\_vector(7 downto 0);

        Cf, Zf: out std\_logic:='0';

        outbus: out std\_logic\_vector(7 downto 0):="00000000");

end alu;

architecture alu of alu is

    signal result:std\_logic\_vector(8 downto 0):="000000000";

    signal sa, sb:std\_logic\_vector(8 downto 0);

    begin

        sa<='0'&inx;

        sb<='0'&iny;

        process(S, M, inx, iny)

        begin

            if (M='0') then --math compute or zhitong

                if (S="1001") then

                    outbus<=inx+iny;

                    result<=sa+sb;

                    Cf<=result(8);

                    if (result="000000000") then

                        Zf<='1';

                    else Zf<='0';

                    end if;

                elsif(S="0110") then

                    outbus<=iny-inx;

                    result<=sb-sa;

                    Cf<=result(8);

                    if(result="000000000") then

                        Zf<='1';

                    else Zf<='0';

                    end if;

                elsif(S="1010") then

                    Cf<='0';

                    Zf<='0';

                    outbus<=iny;

                elsif(S="0100") then

                    Cf<='0';

                    Zf<='0';

                    outbus<=iny;

                else

                    outbus<=iny;

                    Cf<='0';

                    Zf<='0';

                end if;

            else --logic compute

                if(S="1011") then

                    outbus<= inx or iny;

                    Cf<='0';

                    Zf<='0';

                elsif(S="0101") then

                    outbus<=not iny;

                    Cf<='0';

                    Zf<='0';

                end if;

            end if;

        end process;

    end architecture alu;

* **移位逻辑**

library ieee;

use ieee.std\_logic\_1164.all;

entity rslogic is

    port(

        fbus, frlbus, frrbus:in std\_logic;

        inbus:in std\_logic\_vector(7 downto 0);

        outbus2: out std\_logic\_vector(7 downto 0);

        Cf:out std\_logic:='0'

    );

end rslogic;

architecture rslogic of rslogic is

    begin

        process(fbus, frlbus, frrbus)

            begin

            if(fbus='0') then

                outbus2<="ZZZZZZZZ";

                Cf<='0';

            else

                if (frlbus='1') then

                    outbus2<=inbus(6 downto 0)&inbus(7);

                    Cf<=inbus(7);

                elsif(frrbus='1') then

                    outbus2<=inbus(0)&inbus(7 downto 1);

                    Cf<=inbus(0);

                else

                    outbus2<=inbus;

                    Cf<='0';

                end if;

            end if;

            end process;

    end architecture rslogic;

* **控制信号产生逻辑**

library ieee;

use ieee.std\_logic\_1164.all;

entity signal\_generate is

    port(

        sm, mov1, mov2, mov3, alu, not1, rsr, rsl, jmp, jz, jc, in1, Z,  C, nop, halt: in std\_logic;

        ir:in std\_logic\_vector(7 downto 0);

        ra, wa, madd:out std\_logic\_vector(1 downto 0);

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

        ldpc, inc, we, xl, dl, m, fbus, frr, frl, ld, Cf, Zf, Sm\_en

: out std\_logic

    );

end signal\_generate;

architecture signal\_generate of signal\_generate is

    signal in0: std\_logic:='1';

    begin

        process(sm, mov1, mov2, mov3, alu, not1, rsr, rsl, jmp, jz, jc,  in1, Z, C, nop, halt, ir)

        begin

            ld<=not sm;

            ldpc<=(jc and C) or (jz and Z) or jmp;

            inc <= (jz and (not Z)) or (jc and (not C)) or nop or (not sm);

            if (ir(7 downto 4)="0010") then

                in0<='1';

            else

                in0<='0';

            end if;

            we <= not(mov1 or mov3 or alu or not1 or rsr or rsl or in0)  or (not sm);

            ra <= ir(1 downto 0);

            wa <= ir(3 downto 2);

            if (sm='1' and mov3='1') then

                madd<="01";

            elsif (sm='1' and mov2='1') then

                madd<="10";

            else

                madd<="00";

            end if;

            s<=ir(7 downto 4);

            xl<=mov2;

            dl<=mov3 or jmp or (Z and jz) or (Z and jc) or (not sm);

            if ((ir(7 downto 4)="1001") or (ir(7 downto 4)="0110")

or(ir(7 downto 4)="1011")or(ir(7 downto 4)="0101")) then

                m<='1';

            else

                m<='0';

            end if;

            frl<=rsl;

            frr<=rsr;

            fbus<= mov1 or mov2 or alu or not1 or rsr or rsl;

            Cf<=alu or rsr or rsl;

            Zf<=alu;

            Sm\_en<=not halt;

        end process;

    end architecture signal\_generate;