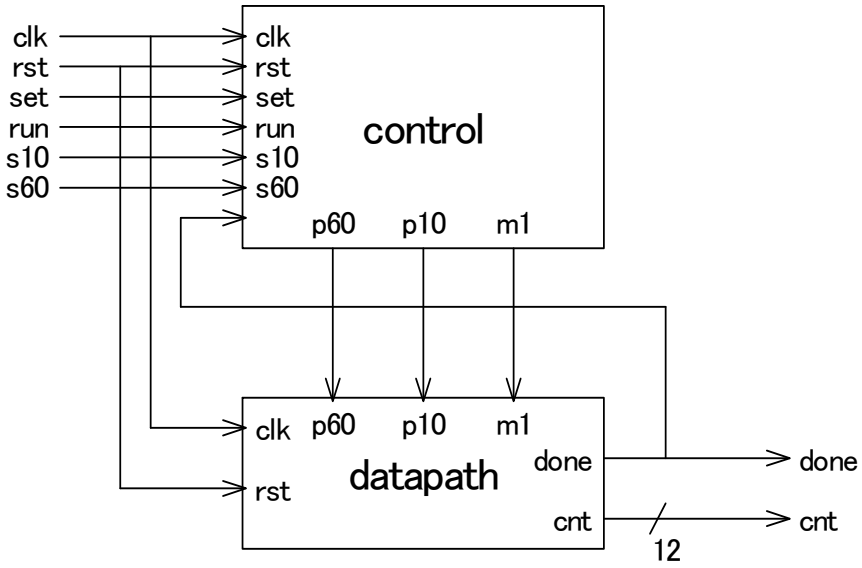
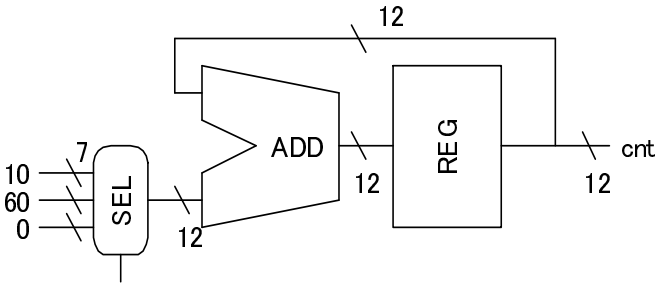
第１０回配布資料

タイマの全体構成

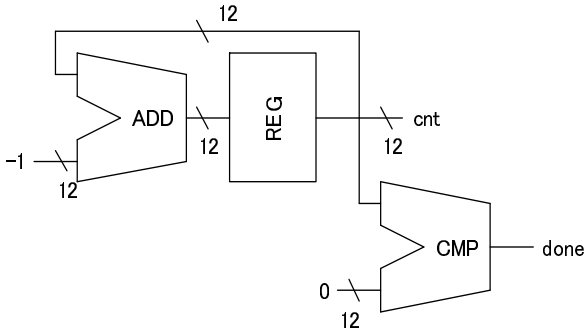


データパス部

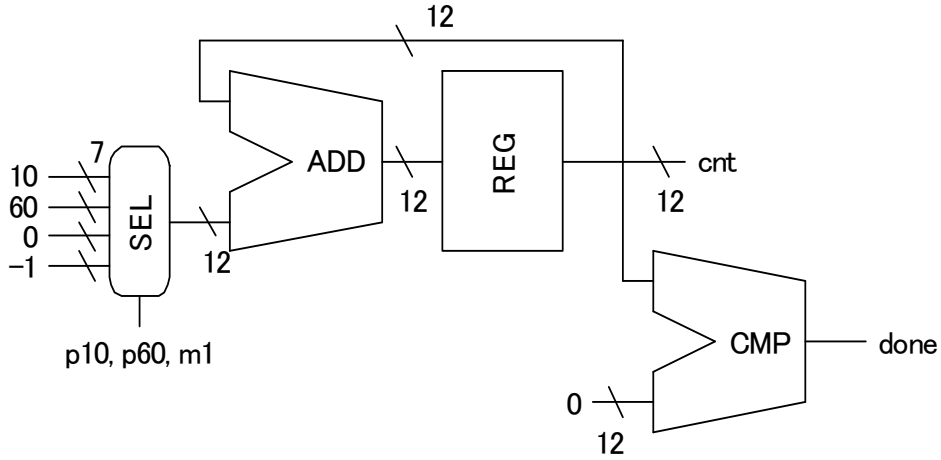
時間設定用データパス



カウントダウン用データパス



統合したデータパス



データパス部の VHDL 記述

```
entity datapath is
  port(
    clk, rst, p10, p60, m1: in std_logic;
    done: out std_logic;
    cnt: out std_logic_vector(11 downto 0));
end datapath

architecture arch of datapath is
  signal reg12_out, reg12_in: std_logic_vetor(11 downto 0);
  signal sel_out: std_logic_vector(6 downto 0);
  function sign_extend (a: std_logic_vector(6 downto 0))
    return std_logic_vector is
  begin
    if (a(6) = '1') then return("11111" & a);
    else return("00000" & a);
    end if
  end sign_extend

begin
  process (p10, p60, m1) begin -- selector
    if (p10 = '1') then sel_out = "0001010";    -- "10"
    elsif (p60 = '1') then sel_out = "0111100"; -- "60"
    elsif (m1 = '1') then sel_out = "1111111";  -- "-1"
    else sel_out = "0000000";
    end if;
  end process;

  process (reg12_out, sel_out) begin -- adder
    reg12_in <= reg12_out + sign_extend(sel_out);
  end process;

  process (clk, rst) begin -- register
    if (rst = '0') then reg12_out = "000000000000";
    elsif (clk'event and clk = '1') then re12_out <= reg_in;
    end if;
  end process;

  process (reg12_out) begin -- comparator
    if (reg12_out = "000000000000" then done <='1';
    else done <= '0';
    end if
  end process;
  cnt <= reg12_out;
end arch;
```

制御部の VHDL 記述

```
entity control is
  port(
    clk, rst, set, run, s10, s60, done: in std_logic;
    p10, p60, m1: out std_logic);
end control;

architecture arch of control is
  type t_state is (IDLE_ST, SET_ST, RUN_ST, DONE_ST);
  signal state, next_state: t_state;
begin
  process (state, set, run, done) begin -- next state calc
    case state is
      when IDLE_ST =>
        if (set = '1') then next_state <= SET_ST;
        else next_state <= IDLE_ST;
        end if;
      when SET_ST =>
```

```
      if (run = '1') then next_state <= RUN_ST;
      else next_state <= SET_ST;
      end if;
    when RUN_ST =>
      if (done = '1') then next_state <= DONE_ST;
      else next_state <= RUN_ST;
      end if;
    when DONE_ST => next_state <= IDLE_ST;
  end case;
 end process;

  process (clk, rst) begin -- state register
    if (rst = '0') then state <= IDLE_ST;
    elsif (clk'event and clk = '1') then state <= next_state;
    end if;
  end process;

  process (state, s10, s60, done) begin -- control signals
    p10 <= '0'; p60 <= '0'; m1 <= '0';
    case state is
      when IDLE_ST => null;
      when SET_ST =>
        if (s10 = '1') then p10 <= '1';
        elsif (s60 = '1') then p60 <= '1';
        end if;
      when RUN_ST =>
        if (done /= '0') then m1 <= '1'; -- /= : Not Equal
        end if;
      when DONE_ST => null;
    end case;
  end process;
end arch;
```

全体構造（トップ階層）の VHDL 記述

```
entity timer is
  port(
    clk, rst, set, run, s10, s60: in std_logic;
    done: out std_logic;
    cnt: std_logic_vector(11 downto 0));
end timer;

architecture arch of timer is
  component control
    port(
      clk, rst, set, run, s10, s60, done: in std_logic;
      p10, p60, m1: out std_logic);
  end component;

  component datapath
    port(
      clk, rst, p10, p60, m1: in std_logic;
      done: out std_logic;
      cnt: out std_logic_vector(11 downto 0));
  end component;

  signal p10, p60, m1, done_tmp: std_logic;
begin
  i0: control port map(
    clk, rst, set, run, s10, s60, done_tmp, p10, p60, m1);
  i1: datapath port map(
    clk, rst, p10, p60, m1, done_tmp, cnt);
  done <= done_tmp;
end arch;
```

構成を意識しない VHDL 記述

```
architecture arch2 of timer is
  type t_state is (IDLE_ST, SET_ST, RUN_ST, DONE_ST);
  signal state: t_state;
  signal cnt_tmp: std_logic_vector(11 downto 0);
  signal sel_out: std_logic_vector(6 downto 0);
begin
  process (clk, rst) begin
    if (rst = '0') then
      cnt_tmp <= "000000000000";
      done <= '0';
      state <= IDLE_ST;
    elsif (clk'event and clk = '1') then
      case state is
        when IDLE_ST =>
          if (set = '1') then state <= SET_ST;
          else state <= IDLE_ST;
          end if;
        when SET_ST =>
          if (s10 = '1') then
            cnt_tmp <= cnt_tmp + "000000001010";
          elsif (s60 = '1') then
            cnt_tmp <= cnt_tmp + "000000111100";
          end if;
          if (run = '1') then state <= RUN_ST;
          else state <= SET_ST;
          end if;
        when RUN_ST =>
          cnt_tmp <= cnt_tmp - "000000000001";
          if (cnt_tmp = "000000000001") then
            done <= '1';
            state <= DONE_ST;
          else state <= RUN_ST;
          end if;
        when DONE_ST =>
          done = '0';
          state <= IDLE_ST;
      end case;
    end if;
  end process;
  cnt <= cnt_tmp;
end arch2;
```