```
entity processor is
    port (
        clock, reset : in std_logic
    );
end processor;

architecture behavioral of processor is
    signal PC, Instr : std_logic_vector (31 downto 0);
    signal Zflag, predicate : std_logic;

-- Register File
    type RF_type is array (0 to 15) of std_logic_vector (31 downto 0);
    signal RF : RF_type;

-- Program and Data Memories and adresses
    type memory_type is array (0 to 63) of std_logic_vector (31 downto 0);
    signal PM, DM : memory_type;
    signal PMadr, DMadr : integer range 0 to 63;
    signal DMadr32 : signed (31 downto 0);

-- Sign extension for branch address
    signal Sext : std_logic_vector (5 downto 0);

-- Instruction fields
    signal F, OP, Cond : std_logic_vector (1 downto 0);
    signal Ubit, Lbit : std_logic;
    signal Imm : std_logic_vector (7 downto 0);
    signal Offset : std_logic_vector (11 downto 0);
    signal S_offset : std_logic_vector (23 downto 0);
    signal Rd, Rn, Rm : integer range 0 to 15;

begin
-- concurrent assignments for extracting instruction fields
    F <= Instr (27 downto 26);
    OP <= Instr (24 downto 23);
    Cond <= Instr (29 downto 28);
    Ubit <= Instr (23);
    Lbit <= Instr (20);
    Imm <= Instr (7 downto 0);
    Offset <= Instr (11 downto 0);
    S_offset <= Instr (23 downto 0);

    Rd <= to_integer (unsigned(Instr (15 downto 12)));
    Rn <= to_integer (unsigned(Instr (19 downto 16)));
    Rm <= to_integer (unsigned(Instr (3 downto 0)));

-- Extracting PM address from PC and getting the instruction
    PMadr <= to_integer (unsigned(PC (7 downto 2)));
    Instr <= PM (PMadr);

-- Data memory address
    DMadr32 <= (signed(RF(Rn)) + signed(X"00000" & Offset)) when (Ubit = '1')
                else (signed(RF(Rn)) – signed(X"00000" & Offset));
    DMadr <= to_integer(DMadr32(7 downto 2));

-- Sign extension for branch address
    S_ext <= "111111" when (Instr(23) = '1') else "000000";

-- Checking condition for branch
    with Cond select
        predicate <= '1'          when "10",
                     Zflag        when "00",
                     not Zflag when "01",
                     '0'          when others;
```

```
-- Process that interprets instructions
    process (reset, clock)
    begin
        if (reset = '1') then PC <= X"00000000";
        elsif (rising_edge(clock)) then
            -- Default address for next instruction
           PC <= std_logic_vector (signed(PC) + 4);
           case F is
              -- DP instructions sub, add, cmp, mov
              when "00" =>
                 case OP is
                    -- Instruction sub
                    when "00" => RF(Rd) <= std_logic_vector(signed(RF(Rn)) – signed(RF(Rm)));
                    -- Instruction add
                    when "01" => RF(Rd) <= std_logic_vector(signed(RF(Rn)) + signed(RF(Rm)));
                    -- Instruction cmp
                    when "10" => if (RF(Rn) = RF(Rm)) then Zflag <= '1'; else Zflag <= '1'; end if;
                    -- Instruction mov
                    when "11" => RF(Rd) <= X"000000" & Imm;
                 end case;
              -- DT instructions ldr, str
              when "01" =>
                 if (Lbit = '1') then
                    RF(Rd) <= DM(DMadr);
                 else
                   DM(DMAdr) <= RF(Rd);
                 end if;
              -- Branch instructions b, beq, bne
              when "10" =>
                 if (predicate = '1') then
                   -- This assignment to PC would override earlier assignment
                   PC <= std_logic_vector (signed(PC) + signed(S_ext & S_offset & "00") + 8);
                 end if;
           end case;
        end if;
    end process;
end behavioral;
```