```vhdl
--
-- Component : mp3_types
-- Name: Jeff Gomberg
-- Name: Gregg Miskelly
--
-- Generated by System Architect version v8.4_3.7 by greenlaw on Mar 14, 1996
--
LIBRARY arithmetic ;
USE arithmetic.std_logic_arith.all;

LIBRARY ieee ;
USE ieee.std_logic_1164.all;

PACKAGE mp3_types IS
-- Datapath signal types
    subtype int IS std_logic_vector (15 downto 0);
    subtype word IS std_logic_vector (31 downto 0);
    subtype uword IS std_ulogic_vector (31 downto 0);
    subtype immediate_value is std_logic_vector (25 downto 0);

    type shift_amount_select is (EV_IR_10to6, EV_16);
    type reg_dest_select is (EV_IR_20to16, EV_IR_15to11, EV_31);

-- MIPS IR types
    subtype opcode IS std_logic_vector (5 downto 0);
    subtype alu_func IS std_logic_vector (5 downto 0);

-- MIPS BTB types
    subtype history IS std_logic_vector (7 downto 0);
    subtype counter IS std_logic_vector (1 downto 0);
    subtype BTBentry IS std_logic_vector (32 downto 0);

-- Control signal types
    type register_bypass_select is (EV_Invalid, EV_RegFile, EV_dResult_3, EV_dRes
    subtype register_address is integer range 31 downto 0;
    subtype shift_amount is integer range 0 to 31;
    type alu_src_2 is (EV_rB, EV_Immediate);
    type alu_op is (EV_Add, EV_ShiftLeft, EV_ShiftRight, EV_Sub, EV_And, EV_Or, E
    type branch_comparitor_select is (EV_rB, EV_dPC_2);
    type did_branch_select is (EV_FALSE, EV_TRUE, EV_fALUzero, EV_fAlUzeroNOT);
    type exe_result_select is (EV_ALU, EV_PcPlus4);
    type branch_target_select is (EV_JType, EV_rA, EV_BranchTarget);
    type guess_wrong_select is (EV_FixedBranchAddress, EV_JumpReg);


-- Types and constants for the control word at each pipeline stage
    type control_word_5 is record
        iRegDest : register_address;
        fRegWriteOp : std_logic;
    end record;

    -- Constant value that containing default values
    constant DEFAULT_CW5 : control_word_5 := (0, '0');

    type control_word_4 is record
        iRegDest : register_address;
        fRegWriteOp : std_logic;

        fBranchOp, fMemWriteOp, fMemReadOp : std_logic;
        dPC : word;
    end record;

    -- Constant value that containing default values
    constant DEFAULT_CW4 : control_word_4 := (0, '0', '0', '0', '0', "00000000000

    type control_word_3 is record
        iRegDest : register_address;
        fRegWriteOp : std_logic;

        fBranchOp, fMemWriteOp, fMemReadOp : std_logic;
        dPC : word;
        dPCplus4 : word;

        iShiftAmount : shift_amount;
        eALUop : alu_op;
        eALUsrc2 : alu_src_2;
        fALUsignExtend : std_logic;

        -- Selects whether or not the instruction actually branches
        eBranchComparitorSelect : branch_comparitor_select;
        eDidBranchSelect : did_branch_select;
        eExeResultSelect : exe_result_select;
        eGuessWrongSelect : guess_wrong_select;
        eUpdatedPCSelect : branch_target_select;
    end record;

    -- Constant value that containing default values
    constant DEFAULT_CW3 : control_word_3 := (0, '0', '0', '0', '0', "00000000000
        "00000000000000000000000000000000", 0, EV_Add, EV_rB, 'X', EV_rB, EV_FALS
        EV_FixedBranchAddress, EV_JType);


    type control_word_2 is record
        eRegDestSelect : reg_dest_select;
        fRegWriteOp : std_logic;

        fBranchOp, fMemWriteOp, fMemReadOp : std_logic;
        dPC : word;
        dPCplus4 : word;

        eShiftAmountSelect : shift_amount_select;
        eALUop : alu_op;
        eALUsrc2 : alu_src_2;
        fALUsignExtend : std_logic;

        -- Selects whether or not the instruction actually branches
        eBranchComparitorSelect : branch_comparitor_select;
        eDidBranchSelect : did_branch_select;
        eExeResultSelect : exe_result_select;
        eGuessWrongSelect : guess_wrong_select;
        eUpdatedPCSelect : branch_target_select;
    end record;

    -- Constant value that containing default values
    constant DEFAULT_CW2 : control_word_2 := (EV_IR_20to16, '0', '0', '0', '0',
        "00000000000000000000000000000000", "00000000000000000000000000000000", E
        EV_rB, 'X', EV_rB, EV_FALSE, EV_ALU,EV_FixedBranchAddress, EV_JType);

-- Instruction definitions
    constant op_alu          : opcode := "000000"; -- 0x00
    constant op_addi         : opcode := "001000"; -- 0x08
    constant op_andi         : opcode := "001100"; -- 0x0c
    constant op_beq          : opcode := "000100"; -- 0x04
    constant op_bne          : opcode := "000101"; -- 0x05
    constant op_j            : opcode := "000010"; -- 0x02
    constant op_jal          : opcode := "000011"; -- 0x03
    constant op_lui          : opcode := "001111"; -- 0x0f
    constant op_lw           : opcode := "100011"; -- 0x23
    constant op_ori          : opcode := "001101"; -- 0x0d
    constant op_slti         : opcode := "001010"; -- 0x0A
    constant op_sw           : opcode := "101011"; -- 0x2B
```

```
    constant alufunc_jr       : alu_func := "001000";
    constant alufunc_addu     : alu_func := "100001";
    constant alufunc_add      : alu_func := "100000";
    constant alufunc_sub      : alu_func := "100010";
    constant alufunc_subu     : alu_func := "100011";
    constant alufunc_and      : alu_func := "100100";
    constant alufunc_or       : alu_func := "100101";
    constant alufunc_sll      : alu_func := "000000";
    constant alufunc_srl      : alu_func := "000010";
    constant alufunc_slt      : alu_func := "101010";
    constant alu_func_sltu    : alu_func := "101011";

    constant ZERO_WORD        : word := "00000000000000000000000000000000";
    constant UNKNOWN_WORD     : word := "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX";
END mp3_types;
```

```vhdl
--
-- VHDL Architecture mp3.ALU.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 11:26:49 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of ALU is
    function ShiftRight(num : word; amt : integer) return word is
        variable i : integer;
        variable ret : word;
    begin
        for i in 0 to 31 loop
            if i > (31 - amt) then
                ret(i) := '0';
            else
                ret(i) := num(i + amt);
            end if;
        end loop;
        return ret;
    end ShiftRight;

    function ShiftLeft(num : word; amt: integer) return word is
        variable i : integer;
        variable ret : word;
    begin
        for i in 0 to 31 loop
            if i < amt then
                ret(i) := '0';
            else
                ret(i) := num(i - amt);
            end if;
        end loop;
        return ret;
    end ShiftLeft;

begin

    compute_res : process (sControlWord_3, dALUsrc1_3, dALUsrc2_3)
        variable result : word;
    begin
        if (fReset = '0') then
            case sControlWord_3.eALUop is
                when EV_Add =>
                    result := dALUsrc1_3 + dALUsrc2_3;

                when EV_Sub =>
                    result := dALUsrc1_3 - dALUsrc2_3;

                when EV_ShiftLeft =>
                    result := ShiftLeft(dALUsrc2_3,sControlWord_3.iShiftAmount);

                when EV_ShiftRight =>
                    result := ShiftRight(dALUsrc2_3,sControlWord_3.iShiftAmount);

                when EV_And =>
                    result := dALUsrc1_3 and dALUsrc2_3;

                when EV_OR =>
                    result := dALUsrc1_3 or dALUsrc2_3;

                when EV_SetLessThan =>
                    if to_integer(dALUsrc1_3) < to_integer(dALUsrc2_3) then
                        result(0) := '1';
                    else
                        result(0) := '0';
                    end if;

                    result(31 downto 1) := ZERO_WORD(31 downto 1);

                when others =>
                    result := ZERO_WORD;
                    ASSERT false
                        report "Error in MUX select"
                        severity failure;

            end case;

            dALU_3 <= result after 9 ns;
        else
            dALU_3 <= ZERO_WORD;
        end if;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.Comparator_with_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:59:41 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of Comparator_with_MUX is
    function to_stdlogic(cond : boolean) return std_logic is
    begin
        if cond then
            return '1';
        else
            return '0';
        end if;
    end to_stdlogic;
begin

    output : process(sControlWord_3, dB_3, dA_3, dPC_2)
        variable result : std_logic;
    begin
        if (sControlWord_3.eBranchComparitorSelect = EV_rB) then
            result := to_stdlogic(dA_3 = dB_3);
        else
            -- check to make sure that the select is valid
            assert sControlWord_3.eBranchComparitorSelect = EV_dPC_2
                severity failure;

            result := to_stdlogic(dA_3 = dPC_2);
        end if;

        fCMPEqual_3 <= result after 3 ns;
    end process;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.CounterMem.default
--
-- Created:
--          by - gomberg.stdt (eesn35.ews.uiuc.edu)
--          at - 16:49:04 11/22/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of CounterMem is
    TYPE counter_array IS array (0 to 255) of counter;
    SIGNAL mem : counter_array;
begin

    write : PROCESS (fReset, clk, fBranchOp_4, update_address_BTB, fDidBranch_4)
        VARIABLE int_address : integer;
    BEGIN
        IF fReset = '1' then
            for int_address in 0 to 255 loop
                mem(int_address) <= "10";
            end loop;
        ELSE
            int_address := to_integer('0' & update_address_BTB);
            ASSERT (int_address >= 0 and int_address <= 255) severity failure;

            if (clk'event and clk = '0' and clk'last_value = '1') then
                if fBranchOp_4 = '1' then
                    if fDidBranch_4 = '1' then
                        if "11" /= mem(int_address) then
                            mem(int_address) <= mem(int_address) + "01" after 12
                        end if;
                    end if;
                    if fDidBranch_4 = '0' then
                        if mem(int_address) /= "00" then
                            mem(int_address) <= mem(int_address) - "01" after 12
                        end if;
                    end if;
                end if; -- if (fBranchOp)
            end if; -- if (Negative Edge)
        end if;
    END PROCESS write;

    read : process (clk, read_address_BTB)
    begin
        if (clk'event and clk = '0' and clk'last_value = '1') then
            MSBcounter <= mem(to_integer('0' & read_address_BTB))(1) after 12 ns;
        end if;
    end process read;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.OtherLogic.default
--
-- Created:
--          by - gomberg.stdt (eesn45.ews.uiuc.edu)
--          at - 15:43:53 11/22/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of OtherLogic is

begin
    process1: PROCESS(fBranchOp_4, MSBcounter, valid_out_1, fDidBranch_4)
    BEGIN
        fGuessTake_BTB <= MSBcounter AND valid_out_1;

        writeTarget_enable <=fDidBranch_4 AND fBranchOp_4;
        writePC_enable <=fDidBranch_4 AND fBranchOp_4;
    END PROCESS process1;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.RegFile.default
--
-- Created:
--          by - gomberg.stdt (eesn34.ews.uiuc.edu)
--          at - 13:38:07 11/23/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of regfile is
    type register_array is array (31 downto 0) of word;

    -- Internal Signals
    signal RegFile : register_array;

begin

    -- Output from register file
    dOutput1_RF <=
        ZERO_WORD when (fReset /= '0' or iReg1Select_2 = 0) else
        dResult_5 after 4 ns when (iRegDest_5 = iReg1Select_2 and fRegWrite_5 = '
        RegFile(iReg1Select_2) after 4 ns;

    dOutput2_RF <=
        ZERO_WORD when (fReset /= '0' or iReg2Select_2 = 0) else
        dResult_5 after 4 ns when (iRegDest_5 = iReg2Select_2 and fRegWrite_5 = '
        RegFile(iReg2Select_2) after 4 ns;

    ---------------------------------------------------------
    -- Process to write data to a register on the negative edge
    regfile_write : process (clk, dResult_5, dResult_5, fRegWrite_5)
    ---------------------------------------------------------
        VARIABLE i : integer;
    begin
        if (fReset = '0') then
            if ((falling_edge(clk)) and (fRegWrite_5 = '1')) then
                ASSERT false
                    report "Writing to register file"
                    severity note;

                if iRegDest_5 /= 0 then
                    RegFile(iRegDest_5) <= dResult_5 after 8 ns;
                end if;
            end if;
        else
            for i in 0 to 31 loop
                RegFile(i) <= ZERO_WORD;
            end loop;
        end if;
    end process regfile_write;
    ---------------------------------------------------------
```

```vhdl
--
-- VHDL Architecture mp3.Result_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:03:16 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

architecture default of Result3_MUX is
begin
    output : process (sControlWord_3, dALU_3)
    variable result : word;
    begin
        case ( sControlWord_3.eExeResultSelect ) is
            when EV_ALU => result := dALU_3;
            when EV_PCPlus4 => result := sControlWord_3.dPCplus4;
            when others =>
                result := UNKNOWN_WORD;
                assert false severity failure; -- illegal mux select
        end case;
        dResult_3 <= result after 1 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.Result_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:03:16 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

architecture default of Result_MUX_3 is
begin
    output : process (sControlWord_3, dALU_3)
    variable result : word;
    begin
        case ( sControlWord_3.eExeResultSelect ) is
            when EV_ALU => result := dALU_3;
            when EV_PCPlus4 => result := sControlWord_3.dPCplus4;
            when others =>
                result := UNKNOWN_WORD;
                assert false severity failure; -- illegal mux select
        end case;
        dResult_3 <= result after 1 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.Result_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:03:16 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

architecture default of Result_MUX is
begin
    output : process (sControlWord_3, dALU_3)
    variable result : word;
    begin
        case ( sControlWord_3.eExeResultSelect ) is
            when EV_ALU => result := dALU_3;
            when EV_PCPlus4 => result := sControlWord_3.dPCplus4;
            when others =>
                result := UNKNOWN_WORD;
                assert false severity failure; -- illegal mux select
        end case;
        dResult_3 <= result after 1 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.ShiftReg.flow
--
-- Created:
--          by - miskelly.stdt (eesn1.ews.uiuc.edu)
--          at - 16:46:43 12/11/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

ARCHITECTURE flow OF ShiftReg IS

-- Architecture declarations
SIGNAL update_address_buff1, update_address_buff2, update_address_buff3 : history

BEGIN

   ---------------------------------------------------------------------------
   process1 : PROCESS (clk, fReset)
   ---------------------------------------------------------------------------

   -- Process declarations
   Variable HistoryData : history := "00000000";

   BEGIN
      -- Asynchronous Reset
      IF (fReset = '1') THEN
         -- Reset Actions
         HistoryData := "00000000";
         update_address_buff1 <= "00000000";
         update_address_buff2 <="00000000";
         update_address_buff3 <="00000000";

      ELSIF (clk'EVENT AND clk = '0') THEN
         if fBranchOp_4 = '1' then
           HistoryData(6 downto 0) := HistoryData(7 downto 1);
           HistoryData(7) := fDidBranch_4;
         end if;
         read_address_BTB <= HistoryData after 2 ns;
         update_address_buff1 <= HistoryData after 2 ns;
         update_address_buff2 <= update_address_buff1 after 2 ns;
         update_address_buff3 <= update_address_buff2 after 2 ns;
         update_address_BTB <= update_address_buff3 after 2 ns;
      END IF;
   END PROCESS process1;


END flow;
```

```vhdl
--
-- VHDL Architecture mp3.TargetMem.default
--
-- Created:
--          by - gomberg.stdt (eesn35.ews.uiuc.edu)
--          at - 17:26:04 11/22/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of TargetMem is
        TYPE BTB_array IS array (0 to 2048) of BTBentry;
        signal mem : BTB_array;

begin

  read : PROCESS (fReset, dPC_1)
    VARIABLE int_address_one : integer;
  begin
    if (fReset = '0') then
        int_address_one := to_integer('0' & dPC_1(12 downto 2));
        valid_out_1 <= mem(int_address_one)(32) after 12 ns;
        dNextAddressGuess_BTB <= mem(int_address_one)(31 downto 0) after 12 ns;
    else
        valid_out_1 <= '0';
    end if;
  end process read;

  write : PROCESS (fReset, clk, writePC_enable, writeTarget_enable, dUpdatedPC_4)
    VARIABLE int_add_two : integer;
  begin
    if (fReset /= '0') then
        for int_add_two in 0 to 2048 loop
            mem(int_add_two) <= "000000000000000000000000000000000";
        end loop;
    else
        int_add_two := to_integer('0' & dCurrentAddress_4(12 downto 2));
        ASSERT (int_add_two >= 0 and int_add_two <= 1048) severity failure;

        if (clk'event and clk = '0' and clk'last_value = '1') then
            if writePC_enable = '1' then
                    mem(int_add_two)(32) <= '1' after 12 ns;
                end if;
                if writeTarget_enable = '1' then
                    mem(int_add_two)(31 downto 0) <= dUpdatedPC_4 after 12 ns;
                end if;
        end if;
    end if;
  end process write;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.UpdatedPC_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn12.ews.uiuc.edu)
--          at - 13:28:25 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of UpdatedPC3_MUX is
begin
    output : process (sControlWord_3,fDidBranch_3,dImmediate_3,dA_3,dBranchTarget_
    variable result : word;
    begin
        if (fDidBranch_3 /= '1') then
            -- If the branch is not taken, output PC + 4
            result := sControlWord_3.dPCplus4;
        else
            -- The branch was taken
            case ( sControlWord_3.eUpdatedPCSelect ) is
                when EV_JTYPE => -- jal and j
                                        result (31 downto 28) := sControlWord_3.d
                                        result (27 downto 2) := dImmediate_3;
                                        result (1 downto 0) := "00";

                when EV_rA => -- jr
                    result := dA_3;

                when EV_BranchTarget => -- bne and beq
                    result := dBranchTarget_3;

                when others =>
                    result := UNKNOWN_WORD;
                    assert false severity failure; -- illegal mux select
            end case;
        end if;

        dUpdatedPC_3 <= result after 2 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.UpdatedPC_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn12.ews.uiuc.edu)
--          at - 13:28:25 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of UpdatedPC_MUX is
begin
    output : process (sControlWord_3,fDidBranch_3,dImmediate_3,dA_3,dBranchTarget_
    variable result : word;
    begin
        if (fDidBranch_3 /= '1') then
            -- If the branch is not taken, output PC + 4
            result := sControlWord_3.dPCplus4;
        else
            -- The branch was taken
            case ( sControlWord_3.eUpdatedPCSelect ) is
                when EV_JTYPE => -- jal and j
                                        result (31 downto 28) := sControlWord_3.d
                                        result (27 downto 2) := dImmediate_3;
                                        result (1 downto 0) := "00";

                when EV_rA => -- jr
                    result := dA_3;

                when EV_BranchTarget => -- bne and beq
                    result := dBranchTarget_3;

                when others =>
                    result := UNKNOWN_WORD;
                    assert false severity failure; -- illegal mux select
            end case;
        end if;

        dUpdatedPC_3 <= result after 2 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.add4.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:43:13 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of add4 is
begin
    dPCplus4_1 <= dPC_1 + "00000000000000000000000000000100" after 4 ns;
end default;
```

```
--
-- VHDL Architecture mp3.alias_block.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:47:27 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of alias_block is
begin
    dOutputToMem_3 <= dB_3;
end default;
```

```
--
-- VHDL Architecture mp3.alias_block.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:47:27 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of alias_dB_3 is
begin
    dOutputToMem_3 <= dB_3;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.alias_result.default
--
-- Created:
--          by - gomberg.stdt (eesn33.ews.uiuc.edu)
--          at - 16:45:42 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of alias_result_4 is
begin
    dDataAddress_4 <= dResult_4;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.alias_result.default
--
-- Created:
--          by - gomberg.stdt (eesn33.ews.uiuc.edu)
--          at - 16:45:42 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of alias_result is
begin
    dDataAddress_4 <= dResult_4;
end default;
```

```
--
-- VHDL Architecture mp3.bt_adder.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:04:19 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

architecture default of bt_adder is
  function encodeDelta(num : std_logic_vector(15 downto 0)) return word is
     variable i : integer;
     variable ret : word;
  begin
     for i in 31 downto 18 loop
        ret(i) := num(15);
     end loop;
     ret(17 downto 2) := num;
     ret(1 downto 0) := "00";

     return ret;
  end;

begin
    dBranchTarget_2 <= dPCplus4_2 + encodeDelta(dIR_2(15 downto 0)) after 4 ns;
end default;
```

```
--
-- VHDL Architecture mp3.bypass_logic.table
--
-- Created:
--          by - miskelly.stdt (eesn1.ews.uiuc.edu)
--          at - 15:42:17 12/11/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ARCHITECTURE table OF bypass_logic IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(fEqu3, fEqu4, fMemRead4, fRegWrite3, fRegWrite4)   BEGI
      -- Block 1
      IF (fEqu3 = '0') AND (fRegWrite3 = '0') AND (fEqu4 = '0') THEN
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_dResult_4;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_DataCache;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '1') AND (fEqu4 = '0') THEN
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '1') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '1') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_dResult_4;
      ELSIF (fEqu3 = '0') AND (fRegWrite3 = '1') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_DataCache;
      ELSIF (fEqu3 = '1') AND (fRegWrite3 = '0') AND (fEqu4 = '0') THEN
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '1') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_RegFile;
      ELSIF (fEqu3 = '1') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_dResult_4;
      ELSIF (fEqu3 = '1') AND (fRegWrite3 = '0') AND (fEqu4 = '1') AND (fRegWrite
         eBypassSelect <= EV_DataCache;
      ELSIF (fEqu3 = '1') AND (fRegWrite3 = '1') THEN
         eBypassSelect <= EV_dResult_3;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```vhdl
--
-- VHDL Architecture mp3.cw_mux.default
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 16:09:30 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of cw_mux is
begin

    write : process(sInternalCW_2, fReset, fGuessWrong_3)
    begin
        if (fReset = '0' and fGuessWrong_3 = '0') then
            sControlWord_2.eRegDestSelect <= sInternalCW_2.eRegDestSelect after 1
            sControlWord_2.fRegWriteOp <= sInternalCW_2.fRegWriteOp after 1 ns;
            sControlWord_2.fBranchOp <= sInternalCW_2.fBranchOp after 1 ns;
            sControlWord_2.fMemWriteOp <= sInternalCW_2.fMemWriteOp after 1 ns;
            sControlWord_2.fMemReadOp <= sInternalCW_2.fMemReadOp after 1 ns;
            sControlWord_2.eShiftAmountSelect <= sInternalCW_2.eShiftAmountSelect
            sControlWord_2.eALUop <= sInternalCW_2.eALUop after 1 ns;
            sControlWord_2.eALUsrc2 <= sInternalCW_2.eALUsrc2 after 1 ns;
            sControlWord_2.fALUsignExtend <= sInternalCW_2.fALUsignExtend after 1
            sControlWord_2.eBranchComparitorSelect <= sInternalCW_2.eBranchCompar
            sControlWord_2.eDidBranchSelect <= sInternalCW_2.eDidBranchSelect aft
            sControlWord_2.eExeResultSelect <= sInternalCW_2.eExeResultSelect aft
            sControlWord_2.eGuessWrongSelect <= sInternalCW_2.eGuessWrongSelect a
            sControlWord_2.eUpdatedPCSelect <= sInternalCW_2.eUpdatedPCSelect aft
            sControlWord_2.dPC <= sInternalCW_2.dPC after 1 ns;
            sControlWord_2.dPCplus4 <= sInternalCW_2.dPCplus4 after 1 ns;
        else
            sControlWord_2 <= DEFAULT_CW2 after 1 ns;
        end if;
    end process;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.delay_rom.untitled
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 16:27:26 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture untitled of delay_rom is
begin
    sInternalCW_2.eRegDestSelect <= ROMout.eRegDestSelect after 2 ns;
    sInternalCW_2.fRegWriteOp <= ROMout.fRegWriteOp after 2 ns;
    sInternalCW_2.fBranchOp <= ROMout.fBranchOp after 2 ns;
    sInternalCW_2.fMemWriteOp <= ROMout.fMemWriteOp after 2 ns;
    sInternalCW_2.fMemReadOp <= ROMout.fMemReadOp after 2 ns;
    sInternalCW_2.eShiftAmountSelect <= ROMout.eShiftAmountSelect after 2 ns;
    sInternalCW_2.eALUop <= ROMout.eALUop after 2 ns;
    sInternalCW_2.eALUsrc2 <= ROMout.eALUsrc2 after 2 ns;
    sInternalCW_2.fALUsignExtend <= ROMout.fALUsignExtend after 2 ns;
    sInternalCW_2.eBranchComparitorSelect <= ROMout.eBranchComparitorSelect after
    sInternalCW_2.eDidBranchSelect <= ROMout.eDidBranchSelect after 2 ns;
    sInternalCW_2.eExeResultSelect <= ROMout.eExeResultSelect after 2 ns;
    sInternalCW_2.eGuessWrongSelect <= ROMout.eGuessWrongSelect after 2 ns;
    sInternalCW_2.eUpdatedPCSelect <= ROMout.eUpdatedPCSelect after 2 ns;

    sInternalCW_2.dPC <= dPC_2;
    sInternalCW_2.dPCplus4 <= dPCplus4_2;
end untitled;
```

```vhdl
--
-- VHDL Architecture mp3.Comparator_with_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 12:59:41 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of fCMPEqual_3_gen is
    function to_stdlogic(cond : boolean) return std_logic is
    begin
        if cond then
            return '1';
        else
            return '0';
        end if;
    end to_stdlogic;
begin

    output : process(sControlWord_3, dB_3, dA_3, dPC_2)
        variable result : std_logic;
    begin
        if (sControlWord_3.eBranchComparitorSelect = EV_rB) then
            result := to_stdlogic(dA_3 = dB_3);
        else
            -- check to make sure that the select is valid
            assert sControlWord_3.eBranchComparitorSelect = EV_dPC_2
                severity failure;

            result := to_stdlogic(dA_3 = dPC_2);
        end if;

        fCMPEqual_3 <= result after 3 ns;
    end process;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.fDidBranch3_MUX.table
--
-- Created:
--          by - miskelly.stdt (eesn41.ews.uiuc.edu)
--          at - 16:32:57 12/12/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ARCHITECTURE table OF fDidBranch3_MUX IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(fCMPEqual_3, sControlWord_3)   BEGIN
      -- Block 1
      IF (sControlWord_3.eDidBranchSelect = EV_FALSE) THEN
         fDidBranch_3 <= '0' after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_TRUE) THEN
         fDidBranch_3 <= '1' after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_fALUzero) THEN
         fDidBranch_3 <= fCMPEqual_3 after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_fALUzeroNot) THEN
         fDidBranch_3 <= NOT fCMPEqual_3 after 2 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```
--
-- VHDL Architecture mp3.fDidBranch_MUX.table
--
-- Created:
--          by - miskelly.stdt (eesn1.ews.uiuc.edu)
--          at - 15:42:24 12/11/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ARCHITECTURE table OF fDidBranch_MUX IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(fCMPEqual_3, sControlWord_3)   BEGIN
      -- Block 1
      IF (sControlWord_3.eDidBranchSelect = EV_FALSE) THEN
         fDidBranch_3 <= '0' after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_TRUE) THEN
         fDidBranch_3 <= '1' after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_fALUzero) THEN
         fDidBranch_3 <= fCMPEqual_3 after 2 ns;
      ELSIF (sControlWord_3.eDidBranchSelect = EV_fALUzeroNot) THEN
         fDidBranch_3 <= NOT fCMPEqual_3 after 2 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```
--
-- VHDL Architecture mp3.fGuessWrong3_MUX.table
--
-- Created:
--          by - miskelly.stdt (eesn41.ews.uiuc.edu)
--          at - 16:52:10 12/12/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ARCHITECTURE table OF fGuessWrong3_MUX IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(fCMPEqual_3, fDidBranch_3, fGuessTake_3, sControlWord_3
      -- Block 1
      IF (sControlWord_3.eGuessWrongSelect = EV_FixedBranchAddress) THEN
         fGuessWrong_3 <= (fDidBranch_3 xor fGuessTake_3) after 1 ns;
      ELSIF (sControlWord_3.eGuessWrongSelect = EV_JumpReg) THEN
         fGuessWrong_3 <= (fCMPEqual_3 nor fGuessTake_3) after 1 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```vhdl
--
-- VHDL Architecture mp3.fGuessWrong_MUX.table
--
-- Created:
--          by - miskelly.stdt (eesn1.ews.uiuc.edu)
--          at - 15:42:24 12/11/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

ARCHITECTURE table OF fGuessWrong_MUX IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(fCMPEqual_3, fDidBranch_3, fGuessTake_3, sControlWord_3
      -- Block 1
      IF (sControlWord_3.eGuessWrongSelect = EV_FixedBranchAddress) THEN
         fGuessWrong_3 <= not (fDidBranch_3 xor fGuessTake_3) after 2 ns;
      ELSIF (sControlWord_3.eGuessWrongSelect = EV_JumpReg) THEN
         fGuessWrong_3 <= (fCMPEqual_3 nor fGuessTake_3) after 2 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```vhdl
--
-- VHDL Architecture mp3.ir_mux.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:00:33 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of ir_mux is
begin
    IrInput_2 <= ZERO_WORD after 1 ns when (fGuessWrong_3 /= '0') else
        dOutput_IC after 1 ns;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.latch.default_latch
--
-- Created:
--          by - miskelly.stdt (eesn40.ews.uiuc.edu)
--          at - 01:41:15 11/02/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default_latch of latch is
    signal internal_store : std_logic;

begin
    fOutput <=
        '0' when (fReset /= '0') else
           internal_store after 2 ns;

    internal_store <=
        '0' when (fReset /= '0') else
        fInput when falling_edge(clk) else
        internal_store;
end default_latch;
```

```vhdl
--
-- VHDL Architecture mp3.rA_MUX_2.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:29:51 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rA_MUX_2 is
begin
    output : process ( dOutput1_RF, dOutput_DC, dResult_3, dResult_4, eASelect_2)
        variable result : word;
    begin
        case (eASelect_2) is
            when EV_RegFile =>   result := dOutput1_RF;
            when EV_dResult_3 => result := dResult_3;
            when EV_dResult_4 => result := dResult_4;
            when EV_DataCache => result := dOutput_DC;
            when others =>
        end case;

        dA_2 <= result after 2 ns;
    end process;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.rB_MUX_2.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:36:57 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rB_MUX_2 is
begin
    output : process ( dOutput2_RF, dOutput_DC, dResult_3, dResult_4, eBSelect_2)
        variable result : word;
    begin
        case (eBSelect_2) is
            when EV_RegFile =>   result := dOutput2_RF;
            when EV_dResult_3 => result := dResult_3;
            when EV_dResult_4 => result := dResult_4;
            when EV_DataCache => result := dOutput_DC;
            when others =>
        end case;

        dB_2 <= result after 2 ns;
    end process;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.rControlWord_3.default
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 17:06:34 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rControlWord_3 is
    signal sInput, internal_store : control_word_3;

begin
    sInput.iRegDest <= iRegDest_2;
    sInput.fRegWriteOp <= sControlWord_2.fRegWriteOp;

    sInput.fBranchOp <= sControlWord_2.fBranchOp;
    sInput.fMemWriteOp <= sControlWord_2.fMemWriteOp;
    sInput.fMemReadOp <= sControlWord_2.fMemReadOp;

    sInput.dPC <= sControlWord_2.dPC;
    sInput.dPCplus4 <= sControlWord_2.dPCplus4;

    sInput.iShiftAmount <= iShiftAmount_2;
    sInput.eALUop <= sControlWord_2.eALUop;
    sInput.eALUsrc2 <= sControlWord_2.eALUsrc2;
    sInput.fALUsignExtend <= sControlWord_2.fALUsignExtend;

    sInput.eBranchComparitorSelect <= sControlWord_2.eBranchComparitorSelect;
    sInput.eDidBranchSelect <= sControlWord_2.eDidBranchSelect;
    sInput.eExeResultSelect <= sControlWord_2.eExeResultSelect;
    sInput.eGuessWrongSelect <= sControlWord_2.eGuessWrongSelect;
    sInput.eUpdatedPCSelect <= sControlWord_2.eUpdatedPCSelect;

    sControlWord_3 <=
        DEFAULT_CW3 when (fReset /= '0') else
           internal_store after 2 ns;

    internal_store <=
        DEFAULT_CW3 when (fReset /= '0') else
        sInput when falling_edge(clk) else
        internal_store;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.rControlWord_4.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:47:52 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rControlWord_4 is
    signal sInput, internal_store : control_word_4;

begin
    -- Create input
    sInput.iRegDest <= sControlWord_3.iRegDest;
    sInput.fRegWriteOp <= sControlWord_3.fRegWriteOp;

    sInput.fBranchOp <= sControlWord_3.fBranchOp;
    sInput.fMemWriteOp <= sControlWord_3.fMemWriteOp;
    sInput.fMemReadOp <= sControlWord_3.fMemReadOp;
    sInput.dPC <= sControlWord_3.dPC;

    -- Assign to output
    sControlWord_4 <= internal_store after 2 ns;
    fMemRead_4 <= internal_store.fMemReadOp after 2 ns;
    fMemWrite_4 <= internal_store.fMemWriteOp after 2 ns;
    dCurrentAddress_4 <= internal_store.dPC after 2 ns;
    fBranchOp_4 <= internal_store.fBranchOp after 2 ns;

    -- Assign to internal store on negative edge
    internal_store <=
        DEFAULT_CW4 when (fReset /= '0') else
        sInput when falling_edge(clk) else
        internal_store;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.rControlWord_5.default
--
-- Created:
--          by - gomberg.stdt (eesn34.ews.uiuc.edu)
--          at - 14:25:03 11/23/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rControlWord_5 is
    signal sInput, internal_store : control_word_5;
begin
    -- Create input
    sInput.iRegDest <= sControlWord_4.iRegDest;
    sInput.fRegWriteOp <= sControlWord_4.fRegWriteOp;

    -- Assign to output
    iRegDest_5 <= internal_store.iRegDest after 2 ns;
    fRegWrite_5 <= internal_store.fRegWriteOp after 2 ns;

    -- Assign to internal store on negative edge
    internal_store <=
        DEFAULT_CW5 when (fReset /= '0') else
        sInput when falling_edge(clk) else
        internal_store;
```

```
--
-- VHDL Architecture mp3.rImmediate_3.default_reg
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 17:24:26 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default_reg of rImmediate_3 is
    signal internal_store : immediate_value;
begin
    dImmediate_3 <=
        ZERO_WORD(25 downto 0) when (fReset /= '0') else
           internal_store after 2 ns;

    internal_store <=
        ZERO_WORD(25 downto 0) when (fReset /= '0') else
        dImmediate_2 when falling_edge(clk) else
        internal_store;
end default_reg;
```

```
--
-- VHDL Architecture mp3.rPC_1_MUX.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:40:06 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of rPC_1_MUX is
begin
    int_pc_1 <=
        dUpdatedPC_3 after 2 ns when (fGuessWrong_3 = '1') else
        dNextAddressGuess_BTB after 2 ns when (fGuessTake_BTB = '1') else
        dPCplus4_1 after 2 ns;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.reg_bypass_comparitors.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:15:30 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of reg_bypass_comparitors is
    function to_stdlogic(cond : boolean) return std_logic is
    begin
        if cond then
            return '1';
        else
            return '0';
        end if;
    end to_stdlogic;
begin
    -- output flags from the control registers. We need to correct fRegWriteOp if
    -- destination register is zero because fRegWriteOp is really a lie
    fRegWrite3_2 <= sControlWord_3.fRegWriteOp and
        to_stdlogic(sControlWord_3.iRegDest /= 0) after 2 ns;
    fRegWrite4_2 <= sControlWord_4.fRegWriteOp and
        to_stdlogic(sControlWord_4.iRegDest /= 0) after 2 ns;
    fMemRead4_2 <= sControlWord_4.fMemReadOp;

    -- output from four comparitors
    fAEqu3_2 <= to_stdlogic(sControlWord_3.iRegDest = iReg1Select_2) after 2 ns;
    fBEqu3_2 <= to_stdlogic(sControlWord_3.iRegDest = iReg2Select_2) after 2 ns;
    fAEqu4_2 <= to_stdlogic(sControlWord_4.iRegDest = iReg1Select_2) after 2 ns;
    fBEqu4_2 <= to_stdlogic(sControlWord_4.iRegDest = iReg2Select_2) after 2 ns;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.reg.default_reg
--
-- Created:
--          by - miskelly.stdt (eesn40.ews.uiuc.edu)
--          at - 00:32:08 11/02/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default_reg of reg is
    signal internal_store : word;

begin
    dOutput <=
        ZERO_WORD when (fReset /= '0') else
           internal_store after 2 ns;

    internal_store <=
        ZERO_WORD when (fReset /= '0') else
        dInput when ((falling_edge(clk)) and (fWrite = '1')) else
        internal_store;
end default_reg;
```

```
--
-- VHDL Architecture mp3.reg_dest_mux.default
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 16:12:22 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of reg_dest_mux is
begin
    output : process(sInternalCW_2, dIR_2)
        variable result : register_address;
    begin
        case (sInternalCW_2.eRegDestSelect) is
            when EV_IR_20to16 =>
                result := to_integer('0' & dIR_2(20 downto 16));

            when EV_IR_15to11 =>
                result := to_integer('0' & dIR_2(15 downto 11));

            when EV_31 =>
                result := 31;

            when others =>
                assert false severity failure; -- illegal select
        end case;

        iRegDest_2 <= result after 2 ns;
    end process;

end default;
```

```
--
-- VHDL Architecture mp3.reg_select_splices.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:11:59 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of reg_select_splices is
begin
    iReg1Select_2 <= to_integer('0' & dIR_2(25 downto 21));
    iReg2Select_2 <= to_integer('0' & dIR_2(20 downto 16));
end default;
```

```
--
-- VHDL Architecture mp3.result_mux_5.table
--
-- Created:
--          by - miskelly.stdt (eesn41.ews.uiuc.edu)
--          at - 17:08:13 12/12/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

ARCHITECTURE table OF result_mux_5 IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(dOutput_DC, dResult_4, sControlWord_4)   BEGIN
      -- Block 1
      IF (sControlWord_4.fMemReadOp = '1') THEN
         dMuxOutput_5 <= dOutput_DC after 1 ns;
      ELSIF (sControlWord_4.fMemReadOp = '0') THEN
         dMuxOutput_5 <= dResult_4 after 1 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```
--
-- VHDL Architecture mp3.result_mux.table
--
-- Created:
--          by - miskelly.stdt (eesn1.ews.uiuc.edu)
--          at - 15:42:29 12/11/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--

ARCHITECTURE table OF result_mux IS

-- Architecture declarations


BEGIN

   truth_process: PROCESS(dOutput_DC, dResult_4, sControlWord_4)   BEGIN
      -- Block 1
      IF (sControlWord_4.fMemWriteOp = '1') THEN
         dMuxOutput_5 <= dOutput_DC after 1 ns;
      ELSIF (sControlWord_4.fMemWriteOp = '0') THEN
         dMuxOutput_5 <= dResult_4 after 1 ns;
      END IF;

   END PROCESS truth_process;

-- Architecture concurrent statements


END table;
```

```vhdl
--
-- VHDL Architecture mp3.shift_amt_mux.default
--
-- Created:
--          by - miskelly.stdt (glsn4.ews.uiuc.edu)
--          at - 16:53:13 11/14/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of shift_amt_mux is
begin
    output : process(sInternalCW_2, dIR_2)
        variable result : shift_amount;
    begin
        case (sInternalCW_2.eShiftAmountSelect) is
            when EV_IR_10to6 =>
                result := to_integer('0' & dIR_2(10 downto 6));

            when EV_16 =>
                result := shift_amount(16);

            when others =>
                assert false severity failure; -- illegal select
        end case;

        iShiftAmount_2 <= result after 1 ns;
    end process;

end default;
```

```vhdl
--
-- VHDL Architecture mp3.splice_immediate.default
--
-- Created:
--          by - miskelly.stdt (eesn45.ews.uiuc.edu)
--          at - 14:09:53 11/16/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of splice_immediate is
begin
    dImmediate_2 <= dIR_2(25 downto 0);
end default;
```

```vhdl
--
-- VHDL Architecture mp3.src1_alias.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 11:22:03 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of src1_alias is
begin
    dALUsrc1_3 <= dA_3;
end default;
```

```vhdl
--
-- VHDL Architecture mp3.src2_MUX.default
--
-- Created:
--          by - miskelly.stdt (glsn17.ews.uiuc.edu)
--          at - 11:23:06 11/13/98
--
-- Generated by Mentor Graphics' Renoir(TM) 98.2 (Build 4)
--
architecture default of src2_MUX is
begin
    output : process(sControlWord_3, dB_3, dImmediate_3)
        variable i : integer;
    begin
        case ( sControlWord_3.eALUsrc2 ) is
            when EV_rB =>
                dALUsrc2_3 <= dB_3 after 2 ns;

            when EV_Immediate =>
                -- Sign or zero extend the immediate
                for i in 31 downto 16 loop
                    if (sControlWord_3.fALUsignExtend = '1') then
                        dALUsrc2_3(i) <= dImmediate_3(15) after 2 ns;
                    else
                        dALUsrc2_3(i) <= '0' after 2 ns;
                    end if;
                end loop;
                dALUsrc2_3(15 downto 0) <= dImmediate_3(15 downto 0) after 2 ns;

            when others =>
                -- It should not be possible to have a select value not listed
                ASSERT false severity failure;
        end case;
    end process;
end default;
```

ns	delta	dresult_5	fregwrite_5	iregdest_5
102	+0	55550000	1	1A
119	+0	00000000	1	5
136	+0	12345678	1	5
153	+0	00000000	1	0
170	+0	00000000	1	0
187	+0	AAAA0000	1	A
204	+0	4320A988	1	B
221	+0	10140000	1	C
238	+0	57755678	1	D
255	+0	56780000	1	E
272	+0	00012345	1	F
289	+0	00000908	1	10
306	+0	4320BBBB	1	11
323	+0	AAAA1234	1	12
340	+0	4320A984	1	13
357	+0	00000001	1	14
374	+0	00000000	1	15
391	+0	00000001	1	16
408	+0	00000001	1	17
425	+0	44440000	1	A
442	+0	44452345	1	B
459	+0	44440908	1	C
476	+0	44440001	1	D
493	+0	88880000	1	14
510	+0	99990000	1	14
527	+0	DDDD0908	1	15
544	+0	DDDE2345	1	19
561	+0	34340000	1	1A
578	+0	00000000	1	0
595	+0	56560000	1	1A
612	+0	9A9A0908	1	19
629	+0	71710000	1	1B
646	+0	82820000	1	1B
663	+0	93930000	1	1B
680	+0	D7D82345	1	18
697	+0	55550000	1	A
714	+0	55562345	1	B
731	+0	55550908	1	C
748	+0	33320908	1	D
765	+0	88880000	1	16
782	+0	99990000	1	16
799	+0	EEEE0908	1	17
816	+0	EEEF2345	1	19
833	+0	95950000	1	1A
850	+0	00000000	1	0
867	+0	B4B40000	1	1A
884	+0	0A090908	1	1D
901	+0	ABAB0000	1	16
918	+0	BCBC0000	1	16
935	+0	99990000	1	16
952	+0	EEEF2345	1	1E
969	+0	00010000	1	C
986	+0	AAAB2345	1	A
1003	+0	AAAC2345	1	F
1020	+0	AAAC2345	1	10
1037	+0	0001468A	1	12
1054	+0	AAAC2345	1	A
1071	+0	00020000	1	A
1088	+0	999B0000	1	11
1105	+0	DDDF0908	1	13
1122	+0	AAAD2345	1	E
1139	+0	00000000	1	0
1156	+0	A3A20908	1	E
1173	+0	F8F82C4D	1	18
1190	+0	D6D41210	1	F
1207	+0	77760908	1	F
1224	+0	88870908	1	F
1241	+0	88880908	1	19
1258	+0	55550000	1	A
1275	+0	AAAB2345	1	A
1292	+0	AAAC2345	1	F
1309	+0	AAAC2345	1	10
1326	+0	0001468A	1	12
1343	+0	AAAC2345	1	A
1360	+0	00020000	1	A
1377	+0	999B0000	1	11
1394	+0	DDDF0908	1	13
1411	+0	A3A30908	1	F
1428	+0	00000000	1	0
1445	+0	A3A20908	1	F
1462	+0	47441210	1	18
1479	+0	D6D41210	1	10
1496	+0	77760908	1	10
1513	+0	88870908	1	10
1530	+0	88880908	1	19
1547	+0	55582345	1	0
1564	+0	00000000	1	A
1581	+0	00000000	1	A
1598	+0	00000000	1	A
1615	+0	50000000	1	A
1632	+0	00001234	1	8
1649	+0	123468AC	1	8
1666	+0	FFFF0000	1	A
1683	+0	33330000	1	8
1700	+0	CCCC0000	1	B
1717	+0	00000000	1	0
1734	+0	99990000	1	C
1751	+0	CCCC0000	1	A
1768	+0	44440000	1	8
1785	+0	44440000	1	B
1802	+0	00000000	1	0
1819	+0	44440000	1	C
1836	+0	12340000	1	A
1853	+0	ABCD0000	1	8
1870	+0	00000000	1	0
1887	+0	BBFD0000	1	B
1904	+0	BBFD0000	1	C
1921	+0	55560000	1	B
1938	+0	00010000	1	C
1955	+0	00000001	1	A
1972	+0	00000000	1	B
1989	+0	00000001	1	C
2006	+0	00000000	1	0
2023	+0	00000001	1	D
2040	+0	12340000	1	A
2057	+0	00001234	1	B
2074	+0	00012340	1	C
2091	+0	00000000	1	0
2108	+0	00123400	1	D
2125	+0	01234000	1	E
2142	+0	00000220	1	F
2159	+0	00000220	1	10
2176	+0	00000020	1	12
2193	+0	00000220	1	11
2210	+0	000076B4	1	18
2227	+0	00001321	1	12
2244	+0	000033E3	1	13
2261	+0	000044F4	1	14
2278	+0	000067E8	1	15
2295	+0	FFFFDCA7	1	16
2312	+0	FFFF0000	1	15

2329	+0	00000001	1	B
2346	+0	00000001	1	C
2363	+0	00000001	1	D
2380	+0	00000000	1	C
2397	+0	00000468	1	C
2414	+0	00000000	1	A
2431	+0	00000450	1	A
2448	+0	00000000	1	B
2465	+0	00000454	1	B
2482	+0	00000000	1	0
2499	+0	00000004	1	5
2516	+0	00000000	1	0
2533	+0	00000454	1	7
2550	+0	00000004	1	8
2567	+0	0000046C	1	C
2584	+0	00000000	1	0
2601	+0	00000000	1	0
2618	+0	34234753	1	9
2635	+0	39482742	1	16
2652	+0	00000000	1	0
2669	+0	39483976	1	19
2686	+0	3947B9B3	1	1A
2703	+0	00000000	1	A
2720	+0	00000474	1	A
2737	+0	45670000	1	15
2754	+0	00000474	0	0
2771	+0	00000478	0	0
2788	+0	0000047C	0	0
2805	+0	00000478	1	A
2822	+0	00000480	0	0
2839	+0	00000484	0	0
2856	+0	00000488	0	0
2873	+0	45670000	1	18
2890	+0	45670000	1	18
2907	+0	45670000	1	18
2924	+0	000044F4	1	18
2941	+0	000044F4	1	18
2958	+0	000044F4	1	18
2975	+0	00000008	1	6
2992	+0	0000000C	1	6
3009	+0	00000000	0	0
3026	+0	00000010	0	6
3043	+0	00000000	1	0
3060	+0	00000008	1	7
3077	+0	00000000	0	0
3094	+0	00000004	0	8
3111	+0	00000000	1	0
3128	+0	00000010	1	6
3145	+0	000002B8	1	1F
3162	+0	00000018	0	0
3179	+0	00000000	1	0
3196	+0	600D0000	1	8
3213	+0	00000014	1	7
3230	+0	000002B8	0	0
3247	+0	00000000	0	0
3264	+0	00000000	1	0
3281	+0	00000024	0	0
3298	+0	00000000	1	A
3315	+0	00000334	1	A
3332	+0	00000000	1	0
3349	+0	00000334	1	1F
3366	+0	00000334	1	F
3383	+0	00000334	0	0
3400	+0	00000334	0	A
3417	+0	00000000	1	0
3434	+0	00000000	1	A

3451	+0	000002D4	1	A
3468	+0	00000000	1	0
3485	+0	000002D4	1	1F
3502	+0	000002D4	1	10
3519	+0	000002D4	0	0
3536	+0	00000000	0	0
3553	+0	00000000	1	0
3570	+0	00000000	1	A
3587	+0	000002F4	1	A
3604	+0	00000000	1	0
3621	+0	000002F4	1	1F
3638	+0	000002F4	1	10
3655	+0	00000000	1	0
3672	+0	000002F4	0	0
3689	+0	00000000	0	0
3706	+0	00000000	1	0
3723	+0	0BCD0000	1	11
3740	+0	0CDE0000	1	12
3757	+0	01110000	1	13
3774	+0	18AB0000	0	0
3791	+0	01110000	1	14
3808	+0	0DEF0000	0	0
3825	+0	00000000	1	0
3842	+0	02220000	1	15
3859	+0	02220000	0	0
3876	+0	00000000	0	0
3893	+0	00000000	1	0
3910	+0	98760000	1	11
3927	+0	23450000	1	12
3944	+0	98760000	1	13
3961	+0	BBBB0000	1	14
3978	+0	30EC0000	0	0
3995	+0	468A0000	1	15
4012	+0	77760000	0	0
4029	+0	00000000	1	0
4046	+0	54310000	1	16
4063	+0	468A0000	0	1F
4080	+0	00000000	0	0
4097	+0	00000000	1	0
4114	+0	000002F4	0	0
4131	+0	00000000	0	0
4148	+0	00000000	1	0
4165	+0	00000000	1	5
4182	+0	0000043C	1	5
4199	+0	00000000	1	0
4216	+0	0000043C	0	0
4233	+0	00000000	0	0
4250	+0	00000000	1	0
4267	+0	00000000	1	6
4284	+0	0000039C	1	6
4301	+0	00000000	1	0
4318	+0	000003A0	1	6
4335	+0	000003A0	0	0
4352	+0	54310000	0	A
4369	+0	00000000	1	0
4386	+0	61610000	1	6
4403	+0	51510000	1	5
4420	+0	00000000	1	0
4437	+0	00000000	1	0
4454	+0	00000000	1	0
4471	+0	00000000	1	5
4488	+0	61610000	1	5
4505	+0	00000000	1	0
4522	+0	C2C20000	0	0
4539	+0	00000000	0	0
4556	+0	00000000	1	0

4573	+0	00004341	1	6
4590	+0	00004343	1	7
4607	+0	00000000	1	0
4624	+0	00000000	1	0
4641	+0	00004343	1	6
4658	+0	00008686	0	1F
4675	+0	00000000	0	0
4692	+0	00000000	1	0
4709	+0	68330000	1	3
4726	+0	68330000	1	4
4743	+0	00000000	1	0
4760	+0	00000000	1	0
4777	+0	00000000	1	0
4794	+0	00000000	1	3
4811	+0	ABCDABCD	1	3
4828	+0	00000000	1	0
4845	+0	1400ABCD	0	0
4862	+0	00000000	0	0
4879	+0	00000000	1	0
4896	+0	45630000	1	9
4913	+0	45630000	1	8
4930	+0	00000000	1	0
4947	+0	00000000	1	0
4964	+0	45634213	1	9
4981	+0	C2C20000	1	5
4998	+0	8AC64213	0	0
5015	+0	00000000	0	0
5032	+0	00000000	1	0
5049	+0	600D0000	1	A
5066	+0	00000000	0	0
5083	+0	00004343	0	6
5100	+0	00000000	1	0
5117	+0	600D0000	1	A
5134	+0	00000000	0	0
5151	+0	00004343	0	6
5168	+0	00000000	1	0
5185	+0	600D0000	1	A
5202	+0	00000000	0	0
5219	+0	600D0000	1	A
5236	+0	00000000	0	0
5253	+0	600D0000	1	A
5270	+0	00000000	0	0
5287	+0	600D0000	1	A
5304	+0	00000000	0	0
5321	+0	600D0000	1	A
5338	+0	00000000	0	0
5355	+0	600D0000	1	A