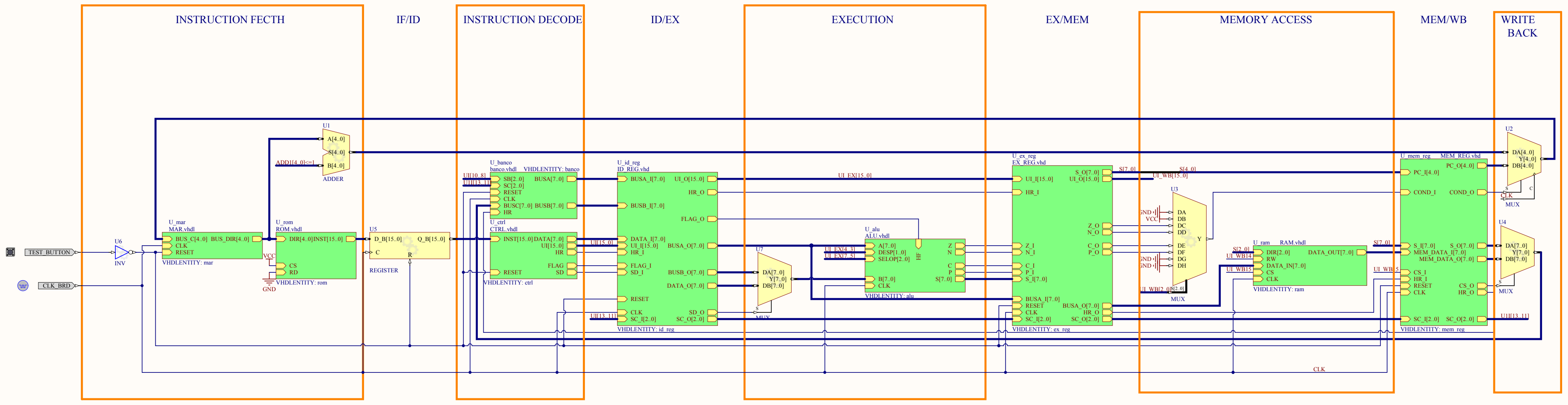

Title		
Size A3	Number	Revision
Date: 28/05/2014		Sheet of
File: C:\Users\..\TOP_PDUA_V2.SchDoc		Drawn By:

```
-- **********************************************
-- **  PROYECTO PDUA                           **
-- **  Modulo:  ALU                            **
-- **  Creacion:    Julio 07                               **
-- **  Revisiï¿½:   Marzo 08                               **
-- **  Por:     MGH-CMUA-UNIANDES              **
-- **********************************************
-- Descripcion:
-- ALU Bit_Slice de N Bits
--            A     B      Clk   HF (habilitador)
--          __|_  __|_      _|___|_
--          \   \/   /     |       |
--   SELOP-->\      / -->  |       |--> C,N,Z,P
--            \____/       |_______|    (Banderas)
--               |RES
--            ___|___
--   DESP -->|_______|
--              |
--              S
-- **********************************************

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ALU is
        Port (CLK,HF : in std_logic;
              A      : in     std_logic_vector(7 downto 0);
              B      : in    std_logic_vector(7 downto 0);
              SELOP  : in  std_logic_vector(2 downto 0);
              DESP   : in    std_logic_vector(1 downto 0);
              S      : out   std_logic_vector(7 downto 0);
             C,N,Z,P : out   std_logic
         );
end ALU;

architecture Bit_Slice of ALU is

component ALU_BIT is
    Port ( A         : in  STD_LOGIC;
           B         : in  STD_LOGIC;
           Cin   : in  STD_LOGIC;
           SELOP     : in  STD_LOGIC_VECTOR (2 downto 0);
              Cout   : out  STD_LOGIC;
           R         : out  STD_LOGIC);
end component;

signal RES  : std_logic_vector(7 downto 0);
signal Cr   : std_logic_vector(7 downto 0);
signal Cm1  : std_logic;

begin
Slices:
for i in 7 downto 0 generate
    BIT0:
    if i=0 generate
        B0: ALU_BIT port map (A(i),B(i),Cm1,SELOP,Cr(i),RES(i));
    end generate;
    BITN:
    if i /= 0 generate
        BN: ALU_BIT port map (A(i),B(i),Cr(i-1),SELOP,Cr(i),RES(i));
end generate;
end generate;

Cm1 <= SELOP(2) and SELOP(1);        -- Carry de entrada a la ALU ='1'
                                             -- para las operaciones
                                             -- 110  B+1
                                             -- 110  Complemento a 2 de B

Banderas:       -- Negativo, Paridad, Carry
```

```
process(clk)
begin
If (clk = '0' and clk'event) then
    If HF = '1' then
        N <= RES(7);
        If RES = "00000000" then Z <='1'; else Z <='0';
        end if;
        P <= not (RES(7) xor RES(6) xor RES(5) xor RES(4) xor RES(3) xor RES(2) xor RES(1) xor RES(0));
        C <= Cr(7);
    end if;
end if;
end process;

Desplazador:
process(DESP,RES)
begin
case DESP is
    when "00" => S <= RES;                              -- No desplaza
    when "01" => S <= '0' & RES(7 downto 1);     -- Desplaza a la derecha
    when "10" => S <= RES(6 downto 0) & '0';     -- Desplaza a la izquierda
    when others => S <= (others => 'X');
end case;
end process;

end Bit_Slice;
```

```vhdl
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  ALU SLICE                       **
-- **  Creacion:    Marzo 08                            **
-- **  Por:     MGH-CMUA-UNIANDES               **
-- ***********************************************
-- Descripcion:
-- ALU Slice de 1 Bit
--           A     B
--         __|_  __|_
--         \   \/   /
--   SELOP-->\      / -->Cout
--      Cin-->\____/
--              |
--              R
-- ***********************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ALU_BIT is
    Port ( A         : in  STD_LOGIC;
           B         : in  STD_LOGIC;
           Cin   : in  STD_LOGIC;
           SELOP     : in  STD_LOGIC_VECTOR (2 downto 0);
              Cout   : out  STD_LOGIC;
           R         : out  STD_LOGIC);
end ALU_BIT;

architecture Behavioral of ALU_BIT is
Begin

Process (A,B,Cin,SELOP)
Begin
 case SELOP is
    when "000" => R <= B;           -- R = B
        Cout <= 'X';
    when "001" => R <= not B;       -- R = /B
        Cout <= 'X';
    when "010" => R <= A and B; -- R = AB
        Cout <= 'X';
    when "011" => R <= A or B;      -- R = A or B
        Cout <= 'X';
    when "100" => R <= A xor B; -- R = A xor B
        Cout <= 'X';
    when "101" =>                       -- R = A + B
        R <= A xor B xor Cin;
        Cout <= (A and B)or (Cin and (A or B));
    when "110" =>                       -- R = B + 1
        R <= B xor Cin;
        Cout <= B and Cin;
    when "111" =>                       -- R = /B + 1
        R <= (not B) xor Cin;
        Cout <= (not B) and Cin;
    when others => R <= 'X';
        Cout <= 'X';
 end case;
end process;

end Behavioral;
```

```
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  BANCO                           **
-- **  Creacion:    Julio 07                    **
-- **  Revision:   Julio 09                    **
-- **  Por:     MGH-CMUA-UNIANDES               **
-- ***********************************************
-- Descripcion:
-- Banco de registros
--           reset_n    HR (Habilitador)
--                  _|___|_
--         clk -->|  PC    |
--                |  SP    |
--                |  DPTR  |
--                |  A     |--> BUSB
--        BUSC -->|  AVI    |--> BUSA
--                |  CTE1  |
--                |  ACC   |
--                |_______|
--                  |   |
--                 SC  SB
--  Selector de destino  Selector de Origen
--         reg <--BUSC  BUSB <-- reg
-- ***********************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity banco is
    Port ( RESET      : in    std_logic;
           HR         : in    std_logic;
           CLK        : in    std_Logic;
           SC,SB      : in    std_logic_vector(2 downto 0);
           BUSC       : in    std_logic_vector(7 downto 0);
           BUSA,BUSB  : out   std_logic_vector(7 downto 0)
         );
end banco;

architecture Behavioral of banco is

SIGNAL PC,SP,DPTR,A,AVI,TEMP,CTE1,ACC : std_logic_vector(7 downto 0);

begin
process (SC,HR,BUSC,RESET,CLK)
begin
if (rising_edge(clk)) then
    if RESET = '1' then
       PC   <= "00000000";
       SP   <= "10000000";  -- Primera posicion de RAM
       DPTR <= "00000000";
       A    <= "00000000";
       AVI  <= "00000000";  -- Apuntador al Vector de Interrupcion
       TEMP <= "00000000";
       CTE1 <= "11111111";  -- Constante Menos 1 (Compl. a 2)
       ACC  <= "00000000";
    elsif HR = '1' then
           case SC is
              when "000" => PC     <= BUSC;
              when "001" => SP     <= BUSC;
              when "010" => DPTR   <= BUSC;
              when "011" => A      <= BUSC;
           -- when "100" => B      <= BUSC; -- B es constante (vector de Int)
              when "101" => TEMP   <= BUSC;
           -- when "110" => CTE 1              -- Es constante (menos 1)
              when "111" => ACC    <= BUSC;
              when others => CTE1  <= "11111111";
           end case;
    end if;
end if;
end process;
```

```
   process(SB,PC,SP,DPTR,A,AVI,TEMP,ACC,CTE1)
    begin
       case SB is
          when "000" => BUSB <= PC;
          when "001" => BUSB <= SP;
          when "010" => BUSB <= DPTR;
          when "011" => BUSB <= A;
          when "100" => BUSB <= AVI;
          when "101" => BUSB <= TEMP;
          when "110" => BUSB <= CTE1;
          when "111" => BUSB <= ACC;
          when others=> BUSB <= ACC;
       end case;
   end process;

   BUSA <= ACC;

   end Behavioral;
```

C:\Users\Daares1\Documents\Dropbox\Pro_Arquitecturas\PDUA_V2\VHDL\CTRL.vhdl

```vhdl
-- ***********************************************
-- **  PROYECTO PDUA                           **
-- **  Modulo:  CONTROL                        **
-- **  Creacion:    Julio 07                   **
-- **  Por:         Mauricio Guerrero H.       **
-- **  Revisión:    Marzo 08                   **
-- **              Conjunto de Instrucciones   **
-- **  Por:        Mauricio Guerrero H.        **
-- **              Diego Mendez Chaves         **
-- ***********************************************
-- **  Bloque para decodificar la instrucción
-- **  15/05/2014 David Arévalo
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

--  UNIDAD DE CONTROL MG

entity CTRL is
    Port ( RESET    : in  std_logic;
           FLAG     : out std_Logic := '0';
           INST     : in  std_logic_vector(15 downto 0);
           HR       : out  std_Logic;
           SD       : out std_Logic;
           UI       : out std_Logic_vector(15 downto 0);
           DATA     : out std_Logic_vector(7 downto 0)
          );
end CTRL;

architecture Behavioral of CTRL is
signal hs,hr1,flag1     : std_logic;
signal data_s: std_Logic_vector(7 downto 0);
signal insts   : std_logic_vector(3 downto 0);

begin
 RI: process(RESET,INST,insts)
 begin
    if RESET = '1' then
       UI <= "0000000000000000";
    else
      hs      <= INST(15);
      hr1     <= INST(10);
      flag1   <= INST(8);
      insts   <= INST(14 downto 11);
      data_s <= INST(7 downto 0);
      case insts is

      --CS(1) RW(1) BUS_C(3) BUSC_B(3) OP(3) DESP (2) COND(3)
           when "0000" => UI <= "0011011000000000";
           when "0001" => UI <= "0011101100000000"; --00001   MOV ACC,A   B
           when "0010" => UI <= "0001111100000000"; --00010   MOV A,ACC   B
           when "0011" => UI <= "00111XXX00000000"; --00011   MOV ACC,CTE B
           when "0100" => UI <= "1011101000000000"; --MOV ACC,[DPTR]      B
           when "0101" => UI <= "0001011100000000"; --MOV DPTR,ACC        B
           when "0110" => UI <= "1111001000000000"; --MOV [DPTR],ACC      B
           when "0111" => UI <= "0011111100100000"; --INV ACC             B
           when "1000" => UI <= "0011101101000000"; --AND ACC,A           B
           when "1001" => UI <= "0011101110100000"; --ADD ACC,A           B
           when "1010" => UI <= "00000XXX00000001"; --JMP DIR             B
           when "1011" => UI <= "00000XXX00000010"; --JZ DIR              B
           when "1100" => UI <= "00000XXX00000011"; --JN DIR              B
           when "1101" => UI <= "00000XXX00000100"; --JC DIR              B
           when "1110" => UI <= "00110110000XXXXX"; --CALL DIR            ?
           when "1111" => UI <= "00110110000XXXXX"; --RET                 ?
           when others => UI <= (others => 'X');
      end case;
    end if;
 end process;
 DATA <= data_s;
 FLAG <= flag1;
```

```
    SD <= hs;
    HR <= hr1;
  end Behavioral;
```

```vhdl
-- ************************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  BANCO                           **
-- **  Creacion:    Julio 07                    **
-- **  Revision:   Julio 09                    **
-- **  Por:     MGH-CMUA-UNIANDES               **
-- ************************************************
-- Descripcion:
-- Banco de registros etapa ID
-- ************************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity EX_REG is
    Port ( RESET      : in    std_logic;
           CLK        : in    std_Logic;
           Z_I        : in    std_Logic := '0';
           N_I        : in    std_Logic := '0';
           C_I        : in    std_Logic := '0';
           P_I        : in    std_Logic := '0';
           HR_I       : in    std_Logic := '0';
           SC_I       : in    std_Logic_vector(2  downto 0) := (OTHERS => '0');
           BUSA_I     : in    std_Logic_vector(7  downto 0) := (OTHERS => '0');
           S_I        : in    std_Logic_vector(7  downto 0) := (OTHERS => '0');
           UI_I       : in    std_logic_vector(15 downto 0) := (OTHERS => '0');
           Z_O        : out   std_Logic := '0';
           N_O        : out   std_Logic := '0';
           C_O        : out   std_Logic := '0';
           P_O        : out   std_Logic := '0';
           HR_O       : out   std_Logic := '0';
           SC_O       : out   std_Logic_vector(2  downto 0) := (OTHERS => '0');
           BUSA_O     : out   std_Logic_vector(7  downto 0) := (OTHERS => '0');
           S_O        : out   std_Logic_vector(7  downto 0) := (OTHERS => '0');
           UI_O       : out   std_Logic_vector(15 downto 0) := (OTHERS => '0')
         );
end EX_REG;

architecture Behavioral of EX_REG is
begin
process (CLK)
begin
if (rising_edge(CLK)) then
    if RESET = '1' then
       Z_O    <= '0';
       N_O    <= '0';
       C_O    <= '0';
       P_O    <= '0';
       HR_O   <= '0';
       SC_O   <= (OTHERS => '0');
       BUSA_O <= (OTHERS => '0');
       S_O    <= (OTHERS => '0');
       UI_O   <= (OTHERS => '0');
    else
       Z_O    <= Z_I;
       N_O    <= N_I;
       C_O    <= C_I;
       P_O    <= P_I;
       HR_O   <= HR_I;
       SC_O   <= SC_I;
       BUSA_O <= BUSA_I;
       S_O    <= S_I;
       UI_O   <= UI_I;
    end if;
end if;
end process;
end Behavioral;
```

```
-- **********************************************
-- **  PROYECTO PDUA                          **
-- **  Modulo:  BANCO                         **
-- **  Creacion:    Julio 07                  **
-- **  Revision:   Julio 09                  **
-- **  Por:     MGH-CMUA-UNIANDES             **
-- **********************************************
-- Descripcion:
-- Banco de registros etapa ID
-- **********************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ID_REG is
    Port ( RESET       : in    std_logic;
           CLK         : in    std_Logic;
           SD_I,HR_I,FLAG_I  : in    std_Logic := '0';
           SD_O,HR_O,FLAG_O  : out   std_Logic := '0';
           SC_I        : in    std_logic_vector(2  downto 0)  := (OTHERS => '0');
           BUSA_I      : in    std_logic_vector(7  downto 0)  := (OTHERS => '0');
           BUSB_I      : in    std_logic_vector(7  downto 0)  := (OTHERS => '0');
           DATA_I      : in    std_logic_vector(7  downto 0)  := (OTHERS => '0');
           UI_I        : in    std_logic_vector(15 downto 0)  := (OTHERS => '0');
           SC_O        : out   std_logic_vector(2  downto 0)  := (OTHERS => '0');
           BUSA_O      : out   std_logic_vector(7  downto 0)  := (OTHERS => '0');
           BUSB_O      : out   std_Logic_vector(7  downto 0)  := (OTHERS => '0');
           DATA_O      : out   std_Logic_vector(7  downto 0)  := (OTHERS => '0');
           UI_O        : out   std_logic_vector(15 downto 0)  := (OTHERS => '0')
         );
end ID_REG;

ARCHITECTURE Behavioral OF ID_REG IS
BEGIN

process (CLK)
begin
if (rising_edge(CLK)) then
    if RESET = '1' then
       SD_O   <= '0';
       HR_O   <= '0';
       FLAG_O <= '0';
       SC_O   <= (OTHERS => '0');
       BUSA_O <= (OTHERS => '0');
       BUSB_O <= (OTHERS => '0');
       DATA_O <= (OTHERS => '0');
       UI_O   <= (OTHERS => '0');
    else
       SD_O   <= SD_I;
       HR_O   <= HR_I;
       FLAG_O <= FLAG_I;
       SC_O   <= SC_I;
       BUSA_O <= BUSA_I;
       BUSB_O <= BUSB_I;
       DATA_O <= DATA_I;
       UI_O   <= UI_I;
    end if;
end if;
end process;
end Behavioral;
```

```vhdl
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  MAR  (Registro de direcciones)  **
-- **  Creacion:    Julio 07                    **
-- **  Revisión:    Marzo 08                    **
-- **  Por:     MGH-CMUA-UNIANDES               **
-- ***********************************************
-- **  Single Cycle PDUA CPU by David Arévalo
-- Descripcion:
-- ALU Bit_Slice de N Bits
--             Clk //HMAR (habilitador)
--             _|___|_
--            |       |
--  BUS_DIR ->|       |--> BUS_C
--            |_______|
--
-- ***********************************************

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity MAR is
    Port ( CLK          : in std_logic;
           RESET        : in std_Logic;
           BUS_DIR      : out std_logic_vector(4 downto 0);
           BUS_C        : in std_logic_vector(4 downto 0)
           --HMAR         : in std_logic);
         );
end MAR;

architecture Behavioral of MAR is

begin
process(RESET,CLK)
begin
if RESET = '1'then
    BUS_DIR <= "00000";
else
    if (CLK'event and CLK ='1')then
    --if HMAR = '1' then
       BUS_DIR <= BUS_C;
    --end if;
    end if;
end if;
end process;
end Behavioral;
```

```
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  BANCO                           **
-- **  Creacion:    Julio 07                    **
-- **  Revision:   Julio 09                    **
-- **  Por:     MGH-CMUA-UNIANDES               **
-- ***********************************************
-- Descripcion:
-- Banco de registros etapa ID
-- ***********************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity MEM_REG is
    Port ( RESET       : in    std_logic;
           CLK         : in    std_Logic;
           HR_I,CS_I   : in    std_Logic := '0';
           HR_O,CS_O   : out   std_Logic := '0';
           COND_I      : in    std_Logic := '0';
           SC_I        : in    std_logic_vector(2  downto 0)  := (OTHERS => '0');
           PC_I        : in    std_logic_vector(4  downto 0)  := (OTHERS => '0');
           S_I         : in    std_logic_vector(7  downto 0)  := (OTHERS => '0');
           MEM_DATA_I  : in    std_logic_vector(7  downto 0)  := (OTHERS => '0');
           COND_O      : out   std_Logic := '0';
           SC_O        : out   std_logic_vector(2  downto 0)  := (OTHERS => '0');
           PC_O        : out   std_logic_vector(4  downto 0)  := (OTHERS => '0');
           MEM_DATA_O  : out   std_logic_vector(7  downto 0)  := (OTHERS => '0');
           S_O         : out   std_Logic_vector(7  downto 0)  := (OTHERS => '0')
         );
end MEM_REG;

architecture Behavioral of MEM_REG is
begin

process (CLK)
begin
if (rising_edge(CLK)) then
    if RESET = '1' then
       CS_O    <= '0';
       HR_O    <= '0';
       COND_O  <= '0';
       SC_O    <= (OTHERS => '0');
       PC_O    <= (OTHERS => '0');
       S_O     <= (OTHERS => '0');

    else
       CS_O    <= CS_I;
       HR_O    <= HR_I;
       COND_O  <= COND_I;
       SC_O    <= SC_I;
       PC_O    <= PC_I;
       S_O     <= S_I;
    end if;
end if;
end process;
end Behavioral;
```

```vhdl
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  RAM                             **
-- **  Creacion:    Julio 07                    **
-- **  Revisión:    Marzo 08                    **
-- **  Por :        MGH-DIMENDEZ-CMUA-UNIANDES  **
-- ***********************************************
-- Descripcion:
-- RAM (Buses de datos independientes in-out)
--                        cs
--                    _____|_
--             rw -->|       |
-- dir(direccion)-->|       |--> data_out
--        data_in -->|_______|
--
-- ***********************************************
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity RAM is
    Port ( CS,RW       : in std_logic;
           CLK         : in std_Logic;
           DIR     : in std_logic_vector(2 downto 0);
           DATA_IN  : in std_logic_vector(7 downto 0);
           DATA_OUT  : out std_logic_vector(7 downto 0));
end RAM;

architecture Behavioral of RAM is

type memoria is array (7 downto 0) of std_logic_vector(7 downto 0);
signal mem: memoria;

begin

-- Memory Write Block
MEM_WRITE:
process (CLK) begin
 if (rising_edge(clk)) then
   if (CS = '1' and RW = '1') then
       mem(conv_integer(DIR)) <= DATA_IN;
   end if;
 end if;
end process;

-- Memory Read Block
MEM_READ:
process (CLK) begin
 if (rising_edge(clk)) then
    if (CS = '1' and RW = '0')  then
         DATA_OUT <= mem(conv_integer(DIR));
    else DATA_OUT <= (others => 'Z');
    end if;
 end if;
end process;
end Behavioral;
```

```vhdl
-- ***********************************************
-- **  PROYECTO PDUA                            **
-- **  Modulo:  ROM                             **
-- **  Creacion:    Julio 07                    **
-- **  Revisión:   Marzo 08                    **
-- **  Por:        MGH-CMUA-UNIANDES            **
-- ***********************************************
-- **Revisión abril 2014 David Arévalo
-- ** Procesador PDUA de ciclo simple
-- Descripcion:
-- ROM (Solo lectura)
--                      cs
--                  ______|_
--             rd -->|        |
--            dir -->|        |--> data
--                  |________|
--
-- ***********************************************

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity ROM is
    Port ( CS,RD    : in std_logic;
           DIR      : in std_logic_vector(4 downto 0);
           INST     : out std_logic_vector(15 downto 0));
end ROM;

architecture Behavioral of ROM is

begin
process(CS,RD,DIR)
begin
if CS = '1' and RD = '0' then
       case DIR is
       --TIPO INSTRUCCIÓN (1) INSTRUCCIÓN (4) H_REGISTRO(1) X(1) FLAGS (1) DATO(8)
        when "00000" => INST <= "1101000000000101";  -- JMP MAIN     -- MAIN
        when "00001" => INST <= "0000000000000000";  -- STALL
        when "00010" => INST <= "0000000000000000";  -- STALL
        when "00011" => INST <= "0000000000000000";  -- STALL
        when "00100" => INST <= "0000100000000000";  -- RAI Vector de Interrupcion
        when "00101" => INST <= "1001110000000001";  -- MAIN: MOV ACC,CTE  -- CTE (0x01)
        when "00110" => INST <= "0000000000000000";  -- STALl
        when "00111" => INST <= "0000000000000000";  -- STALl
        when "01000" => INST <= "0000000000000000";  -- STALL
        when "01001" => INST <= "0000000000000000";  -- STALL
        when "01010" => INST <= "0001010100000000";  -- MOV A,ACC
        when "01011" => INST <= "1001110011111100";  -- MOV ACC,CTE  -- CTE (0xFC)
        when "01100" => INST <= "0000000000000000";  -- STALL
        when "01101" => INST <= "0000000000000000";  -- STALL
        when "01110" => INST <= "0000000000000000";  -- STALL
        when "01111" => INST <= "0000000000000000";  -- STALL
        when "10000" => INST <= "0100110100000000";  -- OTRA: ADD ACC,A
        when "10001" => INST <= "1110100000011100";  -- JC FIN       -- FIN
        when "10010" => INST <= "0000000000000000";  -- STALL
        when "10011" => INST <= "0000000000000000";  -- STALL
        when "10100" => INST <= "0000000000000000";  -- STALL
        when "10101" => INST <= "0000000000000000";  -- STALL
        when "10110" => INST <= "1101000000010000";  -- JMP OTRA     -- OTRA
        when "10111" => INST <= "0000000000000000";  -- STALL
        when "11000" => INST <= "0000000000000000";  -- STALL
        when "11001" => INST <= "0000000000000000";  -- STALL
        when "11010" => INST <= "0000000000000000";  -- STALL
        when "11011" => INST <= "0000000000000000";  -- STALL
        when "11100" => INST <= "1101000000011100";  -- FIN: JMP FIN -- FIN
        when "11101" => INST <= "0000000000000000";  --
        when "11110" => INST <= "0111100000000000";  -- RET
        when "11111" => INST <= "0000000000000000";  --
```

```
            when others => INST <= (others => 'X');
           end case;
    else INST <= (others => 'Z');
    end if;
    end process;
    end;
```