```
------------------------------------------------------------------------------
-- micro_assist_dft.vhd - entity/architecture pair
------------------------------------------------------------------------------
-- IMPORTANT:
-- DO NOT MODIFY THIS FILE EXCEPT IN THE DESIGNATED SECTIONS.
--
-- SEARCH FOR --USER TO DETERMINE WHERE CHANGES ARE ALLOWED.
--
-- TYPICALLY, THE ONLY ACCEPTABLE CHANGES INVOLVE ADDING NEW
-- PORTS AND GENERICS THAT GET PASSED THROUGH TO THE INSTANTIATION
-- OF THE USER_LOGIC ENTITY.
------------------------------------------------------------------------------
--
-- ***************************************************************************
-- ** Copyright (c) 1995-2008 Xilinx, Inc.  All rights reserved.            **
-- **                                                                       **
-- ** Xilinx, Inc.                                                          **
-- ** XILINX IS PROVIDING THIS DESIGN, CODE, OR INFORMATION "AS IS"         **
-- ** AS A COURTESY TO YOU, SOLELY FOR USE IN DEVELOPING PROGRAMS AND       **
-- ** SOLUTIONS FOR XILINX DEVICES.  BY PROVIDING THIS DESIGN, CODE,        **
-- ** OR INFORMATION AS ONE POSSIBLE IMPLEMENTATION OF THIS FEATURE,        **
-- ** APPLICATION OR STANDARD, XILINX IS MAKING NO REPRESENTATION           **
-- ** THAT THIS IMPLEMENTATION IS FREE FROM ANY CLAIMS OF INFRINGEMENT,     **
-- ** AND YOU ARE RESPONSIBLE FOR OBTAINING ANY RIGHTS YOU MAY REQUIRE      **
-- ** FOR YOUR IMPLEMENTATION.  XILINX EXPRESSLY DISCLAIMS ANY              **
-- ** WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE               **
-- ** IMPLEMENTATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR        **
-- ** REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF       **
-- ** INFRINGEMENT, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS       **
-- ** FOR A PARTICULAR PURPOSE.                                             **
-- **                                                                       **
-- ***************************************************************************
--
------------------------------------------------------------------------------
-- Filename:          micro_assist_dft.vhd
-- Version:           1.00.a
-- Description:       Top level design, instantiates library components and user logic.
-- Date:              Sat Apr 04 16:39:42 2009 (by Create and Import Peripheral Wizard)
-- VHDL Standard:     VHDL'93
------------------------------------------------------------------------------
-- Naming Conventions:
--   active low signals:                    "*_n"
--   clock signals:                         "clk", "clk_div#", "clk_#x"
--   reset signals:                         "rst", "rst_n"
--   generics:                              "C_*"
--   user defined types:                    "*_TYPE"
--   state machine next state:              "*_ns"
--   state machine current state:           "*_cs"
--   combinatorial signals:                 "*_com"
--   pipelined or register delay signals:   "*_d#"
--   counter signals:                       "*cnt*"
--   clock enable signals:                  "*_ce"
--   internal version of output port:       "*_i"
--   device pins:                           "*_pin"
--   ports:                                 "- Names begin with Uppercase"
--   processes:                             "*_PROCESS"
--   component instantiations:              "<ENTITY_>I_<#|FUNC>"
------------------------------------------------------------------------------

library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

library proc_common_v2_00_a;
use proc_common_v2_00_a.proc_common_pkg.all;
use proc_common_v2_00_a.ipif_pkg.all;

library plbv46_slave_single_v1_00_a;
use plbv46_slave_single_v1_00_a.plbv46_slave_single;

library micro_assist_dft_v1_00_a;
use micro_assist_dft_v1_00_a.user_logic;

------------------------------------------------------------------------------
-- Entity section
------------------------------------------------------------------------------
-- Definition of Generics:
```

```
--   C_BASEADDR                     -- PLBv46 slave: base address
--   C_HIGHADDR                     -- PLBv46 slave: high address
--   C_SPLB_AWIDTH                  -- PLBv46 slave: address bus width
--   C_SPLB_DWIDTH                  -- PLBv46 slave: data bus width
--   C_SPLB_NUM_MASTERS             -- PLBv46 slave: Number of masters
--   C_SPLB_MID_WIDTH               -- PLBv46 slave: master ID bus width
--   C_SPLB_NATIVE_DWIDTH           -- PLBv46 slave: internal native data bus width
--   C_SPLB_P2P                     -- PLBv46 slave: point to point interconnect scheme
--   C_SPLB_SUPPORT_BURSTS          -- PLBv46 slave: support bursts
--   C_SPLB_SMALLEST_MASTER         -- PLBv46 slave: width of the smallest master
--   C_SPLB_CLK_PERIOD_PS           -- PLBv46 slave: bus clock in picoseconds
--   C_INCLUDE_DPHASE_TIMER         -- PLBv46 slave: Data Phase Timer configuration; 0 = exclude timer, 1 = inc
lude timer
--   C_FAMILY                       -- Xilinx FPGA family
--
-- Definition of Ports:
--   SPLB_Clk                       -- PLB main bus clock
--   SPLB_Rst                       -- PLB main bus reset
--   PLB_ABus                       -- PLB address bus
--   PLB_UABus                      -- PLB upper address bus
--   PLB_PAValid                    -- PLB primary address valid indicator
--   PLB_SAValid                    -- PLB secondary address valid indicator
--   PLB_rdPrim                     -- PLB secondary to primary read request indicator
--   PLB_wrPrim                     -- PLB secondary to primary write request indicator
--   PLB_masterID                   -- PLB current master identifier
--   PLB_abort                      -- PLB abort request indicator
--   PLB_busLock                    -- PLB bus lock
--   PLB_RNW                        -- PLB read/not write
--   PLB_BE                         -- PLB byte enables
--   PLB_MSize                      -- PLB master data bus size
--   PLB_size                       -- PLB transfer size
--   PLB_type                       -- PLB transfer type
--   PLB_lockErr                    -- PLB lock error indicator
--   PLB_wrDBus                     -- PLB write data bus
--   PLB_wrBurst                    -- PLB burst write transfer indicator
--   PLB_rdBurst                    -- PLB burst read transfer indicator
--   PLB_wrPendReq                  -- PLB write pending bus request indicator
--   PLB_rdPendReq                  -- PLB read pending bus request indicator
--   PLB_wrPendPri                  -- PLB write pending request priority
--   PLB_rdPendPri                  -- PLB read pending request priority
--   PLB_reqPri                     -- PLB current request priority
--   PLB_TAttribute                 -- PLB transfer attribute
--   Sl_addrAck                     -- Slave address acknowledge
--   Sl_SSize                       -- Slave data bus size
--   Sl_wait                        -- Slave wait indicator
--   Sl_rearbitrate                 -- Slave re-arbitrate bus indicator
--   Sl_wrDAck                      -- Slave write data acknowledge
--   Sl_wrComp                      -- Slave write transfer complete indicator
--   Sl_wrBTerm                     -- Slave terminate write burst transfer
--   Sl_rdDBus                      -- Slave read data bus
--   Sl_rdWdAddr                    -- Slave read word address
--   Sl_rdDAck                      -- Slave read data acknowledge
--   Sl_rdComp                      -- Slave read transfer complete indicator
--   Sl_rdBTerm                     -- Slave terminate read burst transfer
--   Sl_MBusy                       -- Slave busy indicator
--   Sl_MWrErr                      -- Slave write error indicator
--   Sl_MRdErr                      -- Slave read error indicator
--   Sl_MIRQ                        -- Slave interrupt indicator
------------------------------------------------------------------------------

entity micro_assist_dft is
  generic
  (
    -- ADD USER GENERICS BELOW THIS LINE ---------------
    --USER generics added here
    -- ADD USER GENERICS ABOVE THIS LINE ---------------

    -- DO NOT EDIT BELOW THIS LINE ---------------------
    -- Bus protocol parameters, do not add to or delete
    C_BASEADDR                     : std_logic_vector     := X"FFFFFFFF";
    C_HIGHADDR                     : std_logic_vector     := X"00000000";
    C_SPLB_AWIDTH                  : integer              := 32;
    C_SPLB_DWIDTH                  : integer              := 128;
    C_SPLB_NUM_MASTERS             : integer              := 8;
    C_SPLB_MID_WIDTH               : integer              := 3;
    C_SPLB_NATIVE_DWIDTH           : integer              := 32;
    C_SPLB_P2P                     : integer              := 0;
    C_SPLB_SUPPORT_BURSTS          : integer              := 0;
```

```
    C_SPLB_SMALLEST_MASTER         : integer              := 32;
    C_SPLB_CLK_PERIOD_PS           : integer              := 10000;
    C_INCLUDE_DPHASE_TIMER         : integer              := 1;
    C_FAMILY                       : string               := "virtex5"
    -- DO NOT EDIT ABOVE THIS LINE ---------------------
  );
  port
  (
    -- ADD USER PORTS BELOW THIS LINE ------------------
    --USER ports added here
    -- ADD USER PORTS ABOVE THIS LINE ------------------

    -- DO NOT EDIT BELOW THIS LINE ---------------------
    -- Bus protocol ports, do not add to or delete
    SPLB_Clk                       : in  std_logic;
    SPLB_Rst                       : in  std_logic;
    PLB_ABus                       : in  std_logic_vector(0 to 31);
    PLB_UABus                      : in  std_logic_vector(0 to 31);
    PLB_PAValid                    : in  std_logic;
    PLB_SAValid                    : in  std_logic;
    PLB_rdPrim                     : in  std_logic;
    PLB_wrPrim                     : in  std_logic;
    PLB_masterID                   : in  std_logic_vector(0 to C_SPLB_MID_WIDTH-1);
    PLB_abort                      : in  std_logic;
    PLB_busLock                    : in  std_logic;
    PLB_RNW                        : in  std_logic;
    PLB_BE                         : in  std_logic_vector(0 to C_SPLB_DWIDTH/8-1);
    PLB_MSize                      : in  std_logic_vector(0 to 1);
    PLB_size                       : in  std_logic_vector(0 to 3);
    PLB_type                       : in  std_logic_vector(0 to 2);
    PLB_lockErr                    : in  std_logic;
    PLB_wrDBus                     : in  std_logic_vector(0 to C_SPLB_DWIDTH-1);
    PLB_wrBurst                    : in  std_logic;
    PLB_rdBurst                    : in  std_logic;
    PLB_wrPendReq                  : in  std_logic;
    PLB_rdPendReq                  : in  std_logic;
    PLB_wrPendPri                  : in  std_logic_vector(0 to 1);
    PLB_rdPendPri                  : in  std_logic_vector(0 to 1);
    PLB_reqPri                     : in  std_logic_vector(0 to 1);
    PLB_TAttribute                 : in  std_logic_vector(0 to 15);
    Sl_addrAck                     : out std_logic;
    Sl_SSize                       : out std_logic_vector(0 to 1);
    Sl_wait                        : out std_logic;
    Sl_rearbitrate                 : out std_logic;
    Sl_wrDAck                      : out std_logic;
    Sl_wrComp                      : out std_logic;
    Sl_wrBTerm                     : out std_logic;
    Sl_rdDBus                      : out std_logic_vector(0 to C_SPLB_DWIDTH-1);
    Sl_rdWdAddr                    : out std_logic_vector(0 to 3);
    Sl_rdDAck                      : out std_logic;
    Sl_rdComp                      : out std_logic;
    Sl_rdBTerm                     : out std_logic;
    Sl_MBusy                       : out std_logic_vector(0 to C_SPLB_NUM_MASTERS-1);
    Sl_MWrErr                      : out std_logic_vector(0 to C_SPLB_NUM_MASTERS-1);
    Sl_MRdErr                      : out std_logic_vector(0 to C_SPLB_NUM_MASTERS-1);
    Sl_MIRQ                        : out std_logic_vector(0 to C_SPLB_NUM_MASTERS-1)
    -- DO NOT EDIT ABOVE THIS LINE ---------------------
  );

  attribute SIGIS : string;
  attribute SIGIS of SPLB_Clk      : signal is "CLK";
  attribute SIGIS of SPLB_Rst      : signal is "RST";

end entity micro_assist_dft;

------------------------------------------------------------------------------
-- Architecture section
------------------------------------------------------------------------------

architecture IMP of micro_assist_dft is

  ---------------------------------------
  -- Array of base/high address pairs for each address range
  ---------------------------------------
  constant ZERO_ADDR_PAD                 : std_logic_vector(0 to 31) := (others => '0');
  constant USER_SLV_BASEADDR             : std_logic_vector     := C_BASEADDR;
  constant USER_SLV_HIGHADDR             : std_logic_vector     := C_HIGHADDR;
```

```
  constant IPIF_ARD_ADDR_RANGE_ARRAY      : SLV64_ARRAY_TYPE     :=
    (
      ZERO_ADDR_PAD & USER_SLV_BASEADDR,  -- user logic slave space base address
      ZERO_ADDR_PAD & USER_SLV_HIGHADDR   -- user logic slave space high address
    );

  ------------------------------------------
  -- Array of desired number of chip enables for each address range
  ------------------------------------------
  constant USER_SLV_NUM_REG               : integer              := 3;
  constant USER_NUM_REG                   : integer              := USER_SLV_NUM_REG;

  constant IPIF_ARD_NUM_CE_ARRAY          : INTEGER_ARRAY_TYPE   :=
    (
      0  => pad_power2(USER_SLV_NUM_REG)  -- number of ce for user logic slave space
    );

  ------------------------------------------
  -- Ratio of bus clock to core clock (for use in dual clock systems)
  -- 1 = ratio is 1:1
  -- 2 = ratio is 2:1
  ------------------------------------------
  constant IPIF_BUS2CORE_CLK_RATIO        : integer              := 1;

  ------------------------------------------
  -- Width of the slave data bus (32 only)
  ------------------------------------------
  constant USER_SLV_DWIDTH                : integer              := C_SPLB_NATIVE_DWIDTH;

  constant IPIF_SLV_DWIDTH                : integer              := C_SPLB_NATIVE_DWIDTH;

  ------------------------------------------
  -- Index for CS/CE
  ------------------------------------------
  constant USER_SLV_CS_INDEX              : integer              := 0;
  constant USER_SLV_CE_INDEX              : integer              := calc_start_ce_index(IPIF_ARD_NUM_CE_ARRAY
, USER_SLV_CS_INDEX);

  constant USER_CE_INDEX                  : integer              := USER_SLV_CE_INDEX;

  ------------------------------------------
  -- IP Interconnect (IPIC) signal declarations
  ------------------------------------------
  signal ipif_Bus2IP_Clk                 : std_logic;
  signal ipif_Bus2IP_Reset               : std_logic;
  signal ipif_IP2Bus_Data                : std_logic_vector(0 to IPIF_SLV_DWIDTH-1);
  signal ipif_IP2Bus_WrAck               : std_logic;
  signal ipif_IP2Bus_RdAck               : std_logic;
  signal ipif_IP2Bus_Error               : std_logic;
  signal ipif_Bus2IP_Addr                : std_logic_vector(0 to C_SPLB_AWIDTH-1);
  signal ipif_Bus2IP_Data                : std_logic_vector(0 to IPIF_SLV_DWIDTH-1);
  signal ipif_Bus2IP_RNW                 : std_logic;
  signal ipif_Bus2IP_BE                  : std_logic_vector(0 to IPIF_SLV_DWIDTH/8-1);
  signal ipif_Bus2IP_CS                  : std_logic_vector(0 to ((IPIF_ARD_ADDR_RANGE_ARRAY'length)/2)-1);
  signal ipif_Bus2IP_RdCE                : std_logic_vector(0 to calc_num_ce(IPIF_ARD_NUM_CE_ARRAY)-1);
  signal ipif_Bus2IP_WrCE                : std_logic_vector(0 to calc_num_ce(IPIF_ARD_NUM_CE_ARRAY)-1);
  signal user_Bus2IP_RdCE                : std_logic_vector(0 to USER_NUM_REG-1);
  signal user_Bus2IP_WrCE                : std_logic_vector(0 to USER_NUM_REG-1);
  signal user_IP2Bus_Data                : std_logic_vector(0 to USER_SLV_DWIDTH-1);
  signal user_IP2Bus_RdAck               : std_logic;
  signal user_IP2Bus_WrAck               : std_logic;
  signal user_IP2Bus_Error               : std_logic;

begin

  ------------------------------------------
  -- instantiate plbv46_slave_single
  ------------------------------------------
  PLBV46_SLAVE_SINGLE_I : entity plbv46_slave_single_v1_00_a.plbv46_slave_single
    generic map
    (
      C_ARD_ADDR_RANGE_ARRAY         => IPIF_ARD_ADDR_RANGE_ARRAY,
      C_ARD_NUM_CE_ARRAY             => IPIF_ARD_NUM_CE_ARRAY,
      C_SPLB_P2P                     => C_SPLB_P2P,
      C_BUS2CORE_CLK_RATIO           => IPIF_BUS2CORE_CLK_RATIO,
      C_SPLB_MID_WIDTH               => C_SPLB_MID_WIDTH,
      C_SPLB_NUM_MASTERS             => C_SPLB_NUM_MASTERS,
      C_SPLB_AWIDTH                  => C_SPLB_AWIDTH,
```

```
    C_SPLB_DWIDTH                  => C_SPLB_DWIDTH,
    C_SIPIF_DWIDTH                 => IPIF_SLV_DWIDTH,
    C_INCLUDE_DPHASE_TIMER         => C_INCLUDE_DPHASE_TIMER,
    C_FAMILY                       => C_FAMILY
  )
  port map
  (
    SPLB_Clk                       => SPLB_Clk,
    SPLB_Rst                       => SPLB_Rst,
    PLB_ABus                       => PLB_ABus,
    PLB_UABus                      => PLB_UABus,
    PLB_PAValid                    => PLB_PAValid,
    PLB_SAValid                    => PLB_SAValid,
    PLB_rdPrim                     => PLB_rdPrim,
    PLB_wrPrim                     => PLB_wrPrim,
    PLB_masterID                   => PLB_masterID,
    PLB_abort                      => PLB_abort,
    PLB_busLock                    => PLB_busLock,
    PLB_RNW                        => PLB_RNW,
    PLB_BE                         => PLB_BE,
    PLB_MSize                      => PLB_MSize,
    PLB_size                       => PLB_size,
    PLB_type                       => PLB_type,
    PLB_lockErr                    => PLB_lockErr,
    PLB_wrDBus                     => PLB_wrDBus,
    PLB_wrBurst                    => PLB_wrBurst,
    PLB_rdBurst                    => PLB_rdBurst,
    PLB_wrPendReq                  => PLB_wrPendReq,
    PLB_rdPendReq                  => PLB_rdPendReq,
    PLB_wrPendPri                  => PLB_wrPendPri,
    PLB_rdPendPri                  => PLB_rdPendPri,
    PLB_reqPri                     => PLB_reqPri,
    PLB_TAttribute                 => PLB_TAttribute,
    Sl_addrAck                     => Sl_addrAck,
    Sl_SSize                       => Sl_SSize,
    Sl_wait                        => Sl_wait,
    Sl_rearbitrate                 => Sl_rearbitrate,
    Sl_wrDAck                      => Sl_wrDAck,
    Sl_wrComp                      => Sl_wrComp,
    Sl_wrBTerm                     => Sl_wrBTerm,
    Sl_rdDBus                      => Sl_rdDBus,
    Sl_rdWdAddr                    => Sl_rdWdAddr,
    Sl_rdDAck                      => Sl_rdDAck,
    Sl_rdComp                      => Sl_rdComp,
    Sl_rdBTerm                     => Sl_rdBTerm,
    Sl_MBusy                       => Sl_MBusy,
    Sl_MWrErr                      => Sl_MWrErr,
    Sl_MRdErr                      => Sl_MRdErr,
    Sl_MIRQ                        => Sl_MIRQ,
    Bus2IP_Clk                     => ipif_Bus2IP_Clk,
    Bus2IP_Reset                   => ipif_Bus2IP_Reset,
    IP2Bus_Data                    => ipif_IP2Bus_Data,
    IP2Bus_WrAck                   => ipif_IP2Bus_WrAck,
    IP2Bus_RdAck                   => ipif_IP2Bus_RdAck,
    IP2Bus_Error                   => ipif_IP2Bus_Error,
    Bus2IP_Addr                    => ipif_Bus2IP_Addr,
    Bus2IP_Data                    => ipif_Bus2IP_Data,
    Bus2IP_RNW                     => ipif_Bus2IP_RNW,
    Bus2IP_BE                      => ipif_Bus2IP_BE,
    Bus2IP_CS                      => ipif_Bus2IP_CS,
    Bus2IP_RdCE                    => ipif_Bus2IP_RdCE,
    Bus2IP_WrCE                    => ipif_Bus2IP_WrCE
  );

------------------------------------------
-- instantiate User Logic
------------------------------------------
USER_LOGIC_I : entity micro_assist_dft_v1_00_a.user_logic
  generic map
  (
    -- MAP USER GENERICS BELOW THIS LINE ---------------
    --USER generics mapped here
    -- MAP USER GENERICS ABOVE THIS LINE ---------------

    C_SLV_DWIDTH                   => USER_SLV_DWIDTH,
    C_NUM_REG                      => USER_NUM_REG
  )
  port map
```

```
    (
      -- MAP USER PORTS BELOW THIS LINE ------------------
      --USER ports mapped here
      -- MAP USER PORTS ABOVE THIS LINE ------------------

      Bus2IP_Clk                    => ipif_Bus2IP_Clk,
      Bus2IP_Reset                  => ipif_Bus2IP_Reset,
      Bus2IP_Data                   => ipif_Bus2IP_Data,
      Bus2IP_BE                     => ipif_Bus2IP_BE,
      Bus2IP_RdCE                   => user_Bus2IP_RdCE,
      Bus2IP_WrCE                   => user_Bus2IP_WrCE,
      IP2Bus_Data                   => user_IP2Bus_Data,
      IP2Bus_RdAck                  => user_IP2Bus_RdAck,
      IP2Bus_WrAck                  => user_IP2Bus_WrAck,
      IP2Bus_Error                  => user_IP2Bus_Error
    );

  ----------------------------------------
  -- connect internal signals
  ----------------------------------------
  ipif_IP2Bus_Data <= user_IP2Bus_Data;
  ipif_IP2Bus_WrAck <= user_IP2Bus_WrAck;
  ipif_IP2Bus_RdAck <= user_IP2Bus_RdAck;
  ipif_IP2Bus_Error <= user_IP2Bus_Error;

  user_Bus2IP_RdCE <= ipif_Bus2IP_RdCE(USER_CE_INDEX to USER_CE_INDEX+USER_NUM_REG-1);
  user_Bus2IP_WrCE <= ipif_Bus2IP_WrCE(USER_CE_INDEX to USER_CE_INDEX+USER_NUM_REG-1);

end IMP;
```