Ryerson University

Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Title:	Systems-on-Chip Design
Course Number:	COE 838
Semester/Year (e.g.F2016)	W2022

Instructor:	Dr. Gul Khan

Assignment/Lab Number:	4
Assignment/Lab Title:	Interfacing Custom IP Cores to HPS/FPGA SOC Platform

Submission Date:	Thursday, March 17th, 2022
Due Date:	Thursday, March 24th, 2022

Student LAST Name	Student FIRST Name	Student Number	Section	Signature*
Pano	Toni	500822828	01	T. P.

*By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a “0” on the work, an “F” in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: http://www.ryerson.ca/senate/current/pol60.pdf

Contents

Design of Custom IP

Two modules have been selected to be used in the custom IP for Lab 4, the “LPM_MULT” and “LPM_DIVIDE” modules from the Altera based IP Core catalog in Quartus II 14.0. Each module was created as “mult.vhd” and “div.vhd”, respectively.

The “mult.vhd” module multiplies two 8 bit input numbers together to produce a 16 bit output number. It has asynchronous input signals for enabling and clearing the module. It has a pipeline that takes 3 cycles to process an instruction. The “mult_unit.vhd” module produces a “done” signal after 3 clock cycles have passed from starting an operation with the “mult.vhd” module. The “mult_control.vhd” and “mult_data.vhd” modules are AXI bridge devices that interact with the respective control and data signals of “mult_unit.vhd”.

The ”div.vhd” module divides an 8 bit numerator by an 8 bit denominator to produce an 8 bit quotient and an 8 bit remainder. It has asynchronous input signals for enabling and clearing the module. It has a pipeline that takes 4 cycles to process an instruction. The “div_unit.vhd” module produces a “done” signal after 4 clock cycles have passed from starting an operation with the “div.vhd” module. The “div_control.vhd” and “div_data.vhd” modules are AXI bridge devices that interact with the respective control and data signals of “div_unit.vhd”.

The C program interacts with the devices on the AXI bridge to perform 255 multiplication and division tests. It tests if each output matches an expected result for all possible 8 bit inputs.

Appendix A: C Code for Custom IP

main.c

```
/* COE838 - System-on-Chip
 * Lab 4 - Custom IP for HPS/FPGA Systems
 * main.c
 *
 *  Created on: 2014-11-15
 *  Author: Anita Tino
 */

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <time.h>
#include <sys/mman.h>
#include "hwlib.h"
#include "socal/socal.h"
#include "socal/hps.h"
#include "socal/alt_gpio.h"
#include "hps_0.h"

#define LW_SIZE 0x00200000
```

```
#define LWHPS2FPGA_BASE 0xff200000

volatile uint32_t *mult_control = NULL;
volatile uint32_t *mult_data = NULL;
volatile uint32_t *div_control = NULL;
volatile uint32_t *div_data = NULL;
int mult_success, mult_total;
int div_success, div_total;

void reset_system(){
	alt_write_word(mult_control+1, 0x1); //assert mult reset
	while(!(alt_read_word(mult_control+1) & 0x1));

	printf("MULT Reset done. Deasserting signal\n");
	while((alt_read_word(mult_control+1) & 0x1));//deassert mult reset

	alt_write_word(div_control+1, 0x1); //assert div reset
	while(!(alt_read_word(div_control+1) & 0x1));

	printf("DIV Reset done. Deasserting signal\n");
	while((alt_read_word(div_control+1) & 0x1));//deassert div reset
}

void copy_to_input(uint32_t a, uint32_t b){

	alt_write_word(mult_data, a); //write input A to MULT unit
	alt_write_word(mult_data+1, b); //write input B to MULT unit

	//start conversion
	alt_write_word(mult_control, 0x00000001);

	//double check that div start was asserted
	while(!(alt_read_word(mult_control) & 0x1));

	printf("MULT Start successful\n");

	alt_write_word(div_data, b); //write denominator to DIV unit
	alt_write_word(div_data+1, a); //write numerator to DIV unit

	//start conversion
	alt_write_word(div_control, 0x00000001);     //write 1 to DIV start

	//double check that div start was asserted
	while(!(alt_read_word(div_control) & 0x1));

	printf("DIV Start successful\n");
}

void copy_output(){
	uint32_t word, op1, op2;
	//wait for MULT done
	printf("waiting for MULT done\n");
	while(!(alt_read_word(mult_control+2) & 0x1));

	printf("MULT conversion done\n");
	word = alt_read_word(mult_data+0);
	op1 = alt_read_word(mult_data+1);
	op2 = alt_read_word(mult_data+2);
	printf("0x%08x * 0x%08x = 0x%08x. [Expected] 0x%08x\n",  op1, op2, word, (op1*op2));
	if(word == (op1*op2)){
		printf("[SUCCESSFUL]\n");
		mult_success++;
	}else{
		printf("[FAILED]\n");
	}
	mult_total++;
	printf("------\n");

	uint32_t quotient, remainder, denominator, numerator;
	uint32_t expected_q, expected_r;
	//wait for DIV done
```

```
        printf("waiting for DIV done\n");
        while(!(alt_read_word(div_control+2) & 0x1));

        printf("DIV conversion done\n");
        quotient = alt_read_word(div_data+0);
        remainder = alt_read_word(div_data+1);
        denominator = alt_read_word(div_data+2);
        numerator = alt_read_word(div_data+3);

        if(denominator == 0){
                expected_q = 0x0000FFFF; //max value for 16 bit word
                expected_r = numerator;
        }
        else{
                expected_q = ((uint16_t)numerator) / ((uint16_t)denominator);
                expected_r = numerator % denominator;
        }

        printf("0x%08x / 0x%08x = 0x%08x + r0x%08x. [Expected] 0x%08x + r0x%08x\n",  numerator,
denominator, quotient, remainder, expected_q, expected_r);
        if((quotient == expected_q) && (remainder == expected_r)){
                printf("[SUCCESSFUL]\n");
                div_success++;
        }else{
                printf("[FAILED]\n");
        }
        div_total++;
        printf("------------------------------------------------\n");
}

int main(int argc, char **argv){
        int fd, i, j;
        void *virtual_base;
        mult_success = 0; mult_total = 0;

        //map address space of fpga for software to access here
        if((fd = open("/dev/mem", ( O_RDWR | O_SYNC ) ) ) == -1 ) {
                printf( "ERROR: could not open \"/dev/mem\"...\n" );
                return( 1 );
        }

        virtual_base =  mmap( NULL, LW_SIZE, ( PROT_READ | PROT_WRITE ), MAP_SHARED, fd,
LWHPS2FPGA_BASE);

        if( virtual_base == MAP_FAILED ) {
                printf( "ERROR: mmap() failed...\n" );
                close( fd );
                return(1);
        }

        //initialize the addresses
        mult_control = virtual_base + ((uint32_t)(MULT_CONTROL_0_BASE));
        mult_data = virtual_base + ((uint32_t)(MULT_DATA_0_BASE));
        div_control = virtual_base + ((uint32_t)(DIV_CONTROL_0_BASE));
        div_data = virtual_base + ((uint32_t)(DIV_DATA_0_BASE));

        printf("------>Finished initializing HPS/FPGA system<-------\n");

        for(i = 0; i < 16; i++){
                for(j = 0; j < 16; j++){
                        printf("----------------- Iteration [%d, %d] ------------------\n", i, j);
                        reset_system();
                        copy_to_input((uint32_t)i, (uint32_t)j);
                        copy_output();
                }
        }
        printf("[MULT TEST PASSED] %d/%d\n", mult_success, mult_total);
        printf("[DIV TEST PASSED]  %d/%d\n", div_success, div_total);

        // clean up our memory mapping and exit
        if( munmap( virtual_base, LW_SIZE) != 0 ) {
```

```
                printf( "ERROR: munmap() failed...\n" );
                close( fd );
                return( 1 );
        }

        close( fd );


        return 0;

}
```

hps_0.h

```
#ifndef _ALTERA_HPS_0_H_
#define _ALTERA_HPS_0_H_

/*
 * This file was automatically generated by the swinfo2header utility.
 *
 * Created from SOPC Builder system 'soc_system' in
 * file './soc_system.sopcinfo'.
 */

/*
 * This file contains macros for module 'hps_0' and devices
 * connected to the following masters:
 *   h2f_axi_master
 *   h2f_lw_axi_master
 *
 * Do not include this header file and another header file created for a
 * different module or master group at the same time.
 * Doing so may result in duplicate macro names.
 * Instead, use the system header file which has macros with unique names.
 */

/*
 * Macros for device 'div_data_0', class 'div_data'
 * The macros are prefixed with 'DIV_DATA_0_'.
 * The prefix is the slave descriptor.
 */
#define DIV_DATA_0_COMPONENT_TYPE div_data
#define DIV_DATA_0_COMPONENT_NAME div_data_0
#define DIV_DATA_0_BASE 0x0
#define DIV_DATA_0_SPAN 64
#define DIV_DATA_0_END 0x3f

/*
 * Macros for device 'div_control_0', class 'div_control'
 * The macros are prefixed with 'DIV_CONTROL_0_'.
 * The prefix is the slave descriptor.
 */
#define DIV_CONTROL_0_COMPONENT_TYPE div_control
#define DIV_CONTROL_0_COMPONENT_NAME div_control_0
#define DIV_CONTROL_0_BASE 0x40
#define DIV_CONTROL_0_SPAN 64
#define DIV_CONTROL_0_END 0x7f

/*
 * Macros for device 'mult_control_0', class 'mult_control'
 * The macros are prefixed with 'MULT_CONTROL_0_'.
 * The prefix is the slave descriptor.
 */
#define MULT_CONTROL_0_COMPONENT_TYPE mult_control
#define MULT_CONTROL_0_COMPONENT_NAME mult_control_0
#define MULT_CONTROL_0_BASE 0x80
#define MULT_CONTROL_0_SPAN 64
#define MULT_CONTROL_0_END 0xbf

/*
 * Macros for device 'mult_data_0', class 'mult_data'
 * The macros are prefixed with 'MULT_DATA_0_'.
 * The prefix is the slave descriptor.
 */
#define MULT_DATA_0_COMPONENT_TYPE mult_data
#define MULT_DATA_0_COMPONENT_NAME mult_data_0
#define MULT_DATA_0_BASE 0xc0
#define MULT_DATA_0_SPAN 64
#define MULT_DATA_0_END 0xff


#endif /* _ALTERA_HPS_0_H_ */
```

Appendix B: VHDL Code and Setup for Custom IP

custom_ip.vhdl

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

ENTITY custom_ip IS
       PORT( CLOCK_50, HPS_DDR3_RZQ,HPS_ENET_RX_CLK, HPS_ENET_RX_DV               : IN
STD_LOGIC;
                HPS_DDR3_ADDR                                                     : OUT
STD_LOGIC_VECTOR(14 DOWNTO 0);
                HPS_DDR3_BA                                                       : OUT
STD_LOGIC_VECTOR(2 DOWNTO 0);
                HPS_DDR3_CS_N                                                            : OUT
STD_LOGIC;
                HPS_DDR3_CK_P, HPS_DDR3_CK_N, HPS_DDR3_CKE                        : OUT STD_LOGIC;
                HPS_USB_DIR, HPS_USB_NXT, HPS_USB_CLKOUT                          : IN STD_LOGIC;
                HPS_ENET_RX_DATA                                                  : IN
STD_LOGIC_VECTOR(3 DOWNTO 0);
                HPS_SD_DATA, HPS_DDR3_DQS_N, HPS_DDR3_DQS_P                              : INOUT
STD_LOGIC_VECTOR(3 DOWNTO 0);
                HPS_ENET_MDIO                                                            : INOUT
STD_LOGIC;
                HPS_USB_DATA                                                      : INOUT
STD_LOGIC_VECTOR(7 DOWNTO 0);
                HPS_DDR3_DQ                                                       : INOUT
STD_LOGIC_VECTOR(31 DOWNTO 0);
                HPS_SD_CMD                                                        : INOUT STD_LOGIC;
                HPS_ENET_TX_DATA, HPS_DDR3_DM                                            : OUT
STD_LOGIC_VECTOR(3 DOWNTO 0);
                HPS_DDR3_ODT, HPS_DDR3_RAS_N, HPS_DDR3_RESET_N                    : OUT STD_LOGIC;
                HPS_DDR3_CAS_N, HPS_DDR3_WE_N                                            : OUT
STD_LOGIC;
                HPS_ENET_MDC, HPS_ENET_TX_EN                                             : OUT
STD_LOGIC;
                HPS_USB_STP, HPS_SD_CLK, HPS_ENET_GTX_CLK                         : OUT STD_LOGIC);

END custom_ip;

ARCHITECTURE Behaviour OF custom_ip IS

       --instantiate the soc_systtem component here

    component soc_system is
        port (
            clk_clk                           : in    std_logic                     := 'X';
-- clk
            hps_0_h2f_reset_reset_n           : out   std_logic;
-- reset_n
            hps_io_hps_io_emac1_inst_TX_CLK   : out   std_logic;
-- hps_io_emac1_inst_TX_CLK
            hps_io_hps_io_emac1_inst_TXD0     : out   std_logic;
-- hps_io_emac1_inst_TXD0
            hps_io_hps_io_emac1_inst_TXD1     : out   std_logic;
-- hps_io_emac1_inst_TXD1
            hps_io_hps_io_emac1_inst_TXD2     : out   std_logic;
-- hps_io_emac1_inst_TXD2
            hps_io_hps_io_emac1_inst_TXD3     : out   std_logic;
-- hps_io_emac1_inst_TXD3
            hps_io_hps_io_emac1_inst_RXD0     : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD0
            hps_io_hps_io_emac1_inst_MDIO     : inout std_logic                     := 'X';
-- hps_io_emac1_inst_MDIO
            hps_io_hps_io_emac1_inst_MDC      : out   std_logic;
-- hps_io_emac1_inst_MDC
            hps_io_hps_io_emac1_inst_RX_CTL   : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RX_CTL
            hps_io_hps_io_emac1_inst_TX_CTL   : out   std_logic;
-- hps_io_emac1_inst_TX_CTL
```

```
            hps_io_hps_io_emac1_inst_RX_CLK     : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RX_CLK
            hps_io_hps_io_emac1_inst_RXD1       : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD1
            hps_io_hps_io_emac1_inst_RXD2       : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD2
            hps_io_hps_io_emac1_inst_RXD3       : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD3
            hps_io_hps_io_sdio_inst_CMD         : inout std_logic                     := 'X';
-- hps_io_sdio_inst_CMD
            hps_io_hps_io_sdio_inst_D0          : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D0
            hps_io_hps_io_sdio_inst_D1          : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D1
            hps_io_hps_io_sdio_inst_CLK         : out   std_logic;
-- hps_io_sdio_inst_CLK
            hps_io_hps_io_sdio_inst_D2          : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D2
            hps_io_hps_io_sdio_inst_D3          : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D3
            hps_io_hps_io_usb1_inst_D0          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D0
            hps_io_hps_io_usb1_inst_D1          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D1
            hps_io_hps_io_usb1_inst_D2          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D2
            hps_io_hps_io_usb1_inst_D3          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D3
            hps_io_hps_io_usb1_inst_D4          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D4
            hps_io_hps_io_usb1_inst_D5          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D5
            hps_io_hps_io_usb1_inst_D6          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D6
            hps_io_hps_io_usb1_inst_D7          : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D7
            hps_io_hps_io_usb1_inst_CLK         : in    std_logic                     := 'X';
-- hps_io_usb1_inst_CLK
            hps_io_hps_io_usb1_inst_STP         : out   std_logic;
-- hps_io_usb1_inst_STP
            hps_io_hps_io_usb1_inst_DIR         : in    std_logic                     := 'X';
-- hps_io_usb1_inst_DIR
            hps_io_hps_io_usb1_inst_NXT         : in    std_logic                     := 'X';
-- hps_io_usb1_inst_NXT
            memory_mem_a                        : out   std_logic_vector(14 downto 0);
-- mem_a
            memory_mem_ba                       : out   std_logic_vector(2 downto 0);
-- mem_ba
            memory_mem_ck                       : out   std_logic;
-- mem_ck
            memory_mem_ck_n                     : out   std_logic;
-- mem_ck_n
            memory_mem_cke                      : out   std_logic;
-- mem_cke
            memory_mem_cs_n                     : out   std_logic;
-- mem_cs_n
            memory_mem_ras_n                    : out   std_logic;
-- mem_ras_n
            memory_mem_cas_n                    : out   std_logic;
-- mem_cas_n
            memory_mem_we_n                     : out   std_logic;
-- mem_we_n
            memory_mem_reset_n                  : out   std_logic;
-- mem_reset_n
            memory_mem_dq                       : inout std_logic_vector(31 downto 0) := (others
=> 'X'); -- mem_dq
            memory_mem_dqs                      : inout std_logic_vector(3 downto 0)  := (others
=> 'X'); -- mem_dqs
            memory_mem_dqs_n                    : inout std_logic_vector(3 downto 0)  := (others
=> 'X'); -- mem_dqs_n
```

```
            memory_mem_odt                      : out   std_logic;
-- mem_odt
            memory_mem_dm                       : out   std_logic_vector(3 downto 0);
-- mem_dm
            memory_oct_rzqin                    : in    std_logic                     := 'X';
-- oct_rzqin
            reset_reset_n                       : in    std_logic                     := 'X';
-- reset_n
            mult_data_0_mult_data_m_in1         : out   std_logic_vector(31 downto 0);
-- m_in1
            mult_data_0_mult_data_m_in2         : out   std_logic_vector(31 downto 0);
-- m_in2
            mult_data_0_mult_data_m_result      : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- m_result
            mult_control_0_mult_control_m_start : out   std_logic_vector(31 downto 0);
-- m_start
            mult_control_0_mult_control_m_reset : out   std_logic_vector(31 downto 0);
-- m_reset
            mult_control_0_mult_control_m_done  : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- m_done
                                div_control_0_div_control_d_start   : out   std_logic_vector(31
downto 0);                      -- d_start
            div_control_0_div_control_d_reset   : out   std_logic_vector(31 downto 0);
-- d_reset
            div_control_0_div_control_d_done    : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- d_done
            div_data_0_div_data_d_denom         : out   std_logic_vector(31 downto 0);
-- d_denom
            div_data_0_div_data_d_numer         : out   std_logic_vector(31 downto 0);
-- d_numer
            div_data_0_div_data_d_quot          : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- d_quot
            div_data_0_div_data_d_rem           : in    std_logic_vector(31 downto 0) := (others
=> 'X')  -- d_rem
        );
    end component soc_system;

        component mult_unit is
                PORT( clk, reset, enable             : IN STD_LOGIC;
                                mult_a, mult_b                                  : IN
STD_LOGIC_VECTOR(15 DOWNTO 0);
                                mult_done                                               : OUT
STD_LOGIC;
                                mult_result                                             : OUT
STD_LOGIC_VECTOR(31 DOWNTO 0):= (others => '0')
                        );
        end component mult_unit;

        component div_unit is
                PORT(
                        clk, reset, enable     : IN STD_LOGIC;
                        div_denom, div_numer   : IN STD_LOGIC_VECTOR(15 DOWNTO 0);
                        div_done                                        : OUT STD_LOGIC;
                        div_quot, div_rem             : OUT STD_LOGIC_VECTOR(15 DOWNTO 0)
                         );
        end component div_unit;

        --SIGNALS instantiated here
        SIGNAL reset_reset_n : STD_LOGIC;

        SIGNAL mult_in1, mult_in2, mult_output_result       : STD_LOGIC_VECTOR(31 DOWNTO 0);
        SIGNAL mult_input_start, mult_input_reset           : STD_LOGIC_VECTOR(31 DOWNTO 0);
        SIGNAL mult_done
        : STD_LOGIC;

        SIGNAL div_in_d, div_in_n, div_out_q, div_out_r : STD_LOGIC_VECTOR(31 DOWNTO 0);
        SIGNAL div_input_start, div_input_reset                     : STD_LOGIC_VECTOR(31 DOWNTO
0);
        SIGNAL div_done
        : STD_LOGIC;
```

```
BEGIN

--port map soc_system here
    u0 : component soc_system
 port map (
     clk_clk => CLOCK_50,
                        reset_reset_n => '1',memory_mem_a => HPS_DDR3_ADDR,
                        memory_mem_ba => HPS_DDR3_BA,
                        memory_mem_ck => HPS_DDR3_CK_P,
                        memory_mem_ck_n => HPS_DDR3_CK_N,
                        memory_mem_cke => HPS_DDR3_CKE,
                        memory_mem_cs_n => HPS_DDR3_CS_N,
                        memory_mem_ras_n => HPS_DDR3_RAS_N,
                        memory_mem_cas_n => HPS_DDR3_CAS_N,
                        memory_mem_we_n => HPS_DDR3_WE_N,
                        memory_mem_reset_n => HPS_DDR3_RESET_N,
                        memory_mem_dq => HPS_DDR3_DQ,
                        memory_mem_dqs => HPS_DDR3_DQS_P,
                        memory_mem_dqs_n => HPS_DDR3_DQS_N,
                        memory_mem_odt => HPS_DDR3_ODT,
                        memory_mem_dm => HPS_DDR3_DM,
                        memory_oct_rzqin => HPS_DDR3_RZQ,
                        hps_io_hps_io_emac1_inst_TX_CLK => HPS_ENET_GTX_CLK,
                        hps_io_hps_io_emac1_inst_TXD0 => HPS_ENET_TX_DATA(0),
                        hps_io_hps_io_emac1_inst_TXD1 => HPS_ENET_TX_DATA(1),
                        hps_io_hps_io_emac1_inst_TXD2 => HPS_ENET_TX_DATA(2),
                        hps_io_hps_io_emac1_inst_TXD3 => HPS_ENET_TX_DATA(3),
                        hps_io_hps_io_emac1_inst_RXD0 => HPS_ENET_RX_DATA(0),
                        hps_io_hps_io_emac1_inst_MDIO => HPS_ENET_MDIO,
                        hps_io_hps_io_emac1_inst_MDC => HPS_ENET_MDC,
                        hps_io_hps_io_emac1_inst_RX_CTL => HPS_ENET_RX_DV,
                        hps_io_hps_io_emac1_inst_TX_CTL => HPS_ENET_TX_EN,
                        hps_io_hps_io_emac1_inst_RX_CLK => HPS_ENET_RX_CLK,
                        hps_io_hps_io_emac1_inst_RXD1 => HPS_ENET_RX_DATA(1),
                        hps_io_hps_io_emac1_inst_RXD2 => HPS_ENET_RX_DATA(2),
                        hps_io_hps_io_emac1_inst_RXD3 => HPS_ENET_RX_DATA(3),
                        hps_io_hps_io_sdio_inst_CMD => HPS_SD_CMD,
                        hps_io_hps_io_sdio_inst_D0 => HPS_SD_DATA(0),
                        hps_io_hps_io_sdio_inst_D1 => HPS_SD_DATA(1),
                        hps_io_hps_io_sdio_inst_CLK => HPS_SD_CLK,
                        hps_io_hps_io_sdio_inst_D2 => HPS_SD_DATA(2),
                        hps_io_hps_io_sdio_inst_D3 => HPS_SD_DATA(3),
                        hps_io_hps_io_usb1_inst_D0 => HPS_USB_DATA(0),
                        hps_io_hps_io_usb1_inst_D1 => HPS_USB_DATA(1),
                        hps_io_hps_io_usb1_inst_D2 => HPS_USB_DATA(2),
                        hps_io_hps_io_usb1_inst_D3 => HPS_USB_DATA(3),
                        hps_io_hps_io_usb1_inst_D4 => HPS_USB_DATA(4),
                        hps_io_hps_io_usb1_inst_D5 => HPS_USB_DATA(5),
                        hps_io_hps_io_usb1_inst_D6 => HPS_USB_DATA(6),
                        hps_io_hps_io_usb1_inst_D7 => HPS_USB_DATA(7),
                        hps_io_hps_io_usb1_inst_CLK => HPS_USB_CLKOUT,
                        hps_io_hps_io_usb1_inst_STP => HPS_USB_STP,
                        hps_io_hps_io_usb1_inst_DIR => HPS_USB_DIR,
                        hps_io_hps_io_usb1_inst_NXT => HPS_USB_NXT,
                        hps_0_h2f_reset_reset_n => reset_reset_n,
            mult_data_0_mult_data_m_in1         => mult_in1,         --
mult_data_0_mult_data.m_in1
            mult_data_0_mult_data_m_in2         => mult_in2,         --
.m_in2
            mult_data_0_mult_data_m_result      => mult_output_result,      --
.m_result
            mult_control_0_mult_control_m_start => mult_input_start, --
mult_control_0_mult_control.m_start
            mult_control_0_mult_control_m_reset => mult_input_reset, --
.m_reset
            mult_control_0_mult_control_m_done  => "00000000000000000000000000000000"  &
mult_done, --                              .m_done
                div_control_0_div_control_d_start   => div_input_start,   --
div_control_0_div_control.d_start
```

```
            div_control_0_div_control_d_reset   => div_input_reset,   --
.d_reset
            div_control_0_div_control_d_done    => "00000000000000000000000000000000"  & div_done,
--                              .d_done
            div_data_0_div_data_d_denom         => div_in_d,         --
div_data_0_div_data.d_denom
            div_data_0_div_data_d_numer         => div_in_n,         --
.d_numer
            div_data_0_div_data_d_quot          => div_out_q,          --
.d_quot
            div_data_0_div_data_d_rem           => div_out_r            --
.d_rem
        );

               m0 : component mult_unit
                      port map (
                             clk => CLOCK_50,
                             reset => mult_input_reset(0),
                             enable => mult_input_start(0),
                             mult_a => mult_in1(15 DOWNTO 0),
                             mult_b => mult_in2(15 DOWNTO 0),
                             mult_done => mult_done,
                             mult_result => mult_output_result
                      );

               d0 : component div_unit
                      port map (
                             clk => CLOCK_50,
                             reset => div_input_reset(0),
                             enable => div_input_start(0),
                             div_denom => div_in_d(15 DOWNTO 0),
                             div_numer => div_in_n(15 DOWNTO 0),
                             div_done => div_done,
                             div_quot => div_out_q(15 DOWNTO 0),
                             div_rem => div_out_r(15 DOWNTO 0)
                      );

End Behaviour;
```

soc_system.vhd

```
-- soc_system.vhd

-- Generated using ACDS version 14.0 209 at 2022.03.10.17:09:44

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity soc_system is
        port (
                clk_clk                             : in    std_logic                     := '0';
--                          clk.clk
                hps_0_h2f_reset_reset_n             : out   std_logic;
--              hps_0_h2f_reset.reset_n
                hps_io_hps_io_emac1_inst_TX_CLK     : out   std_logic;
--                       hps_io.hps_io_emac1_inst_TX_CLK
                hps_io_hps_io_emac1_inst_TXD0       : out   std_logic;
--                             .hps_io_emac1_inst_TXD0
                hps_io_hps_io_emac1_inst_TXD1       : out   std_logic;
--                             .hps_io_emac1_inst_TXD1
                hps_io_hps_io_emac1_inst_TXD2       : out   std_logic;
--                             .hps_io_emac1_inst_TXD2
                hps_io_hps_io_emac1_inst_TXD3       : out   std_logic;
--                             .hps_io_emac1_inst_TXD3
                hps_io_hps_io_emac1_inst_RXD0       : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RXD0
                hps_io_hps_io_emac1_inst_MDIO       : inout std_logic                     := '0';
--                             .hps_io_emac1_inst_MDIO
                hps_io_hps_io_emac1_inst_MDC        : out   std_logic;
--                             .hps_io_emac1_inst_MDC
                hps_io_hps_io_emac1_inst_RX_CTL     : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RX_CTL
                hps_io_hps_io_emac1_inst_TX_CTL     : out   std_logic;
--                             .hps_io_emac1_inst_TX_CTL
                hps_io_hps_io_emac1_inst_RX_CLK     : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RX_CLK
                hps_io_hps_io_emac1_inst_RXD1       : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RXD1
                hps_io_hps_io_emac1_inst_RXD2       : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RXD2
                hps_io_hps_io_emac1_inst_RXD3       : in    std_logic                     := '0';
--                             .hps_io_emac1_inst_RXD3
                hps_io_hps_io_sdio_inst_CMD         : inout std_logic                     := '0';
--                             .hps_io_sdio_inst_CMD
                hps_io_hps_io_sdio_inst_D0          : inout std_logic                     := '0';
--                             .hps_io_sdio_inst_D0
                hps_io_hps_io_sdio_inst_D1          : inout std_logic                     := '0';
--                             .hps_io_sdio_inst_D1
                hps_io_hps_io_sdio_inst_CLK         : out   std_logic;
--                             .hps_io_sdio_inst_CLK
                hps_io_hps_io_sdio_inst_D2          : inout std_logic                     := '0';
--                             .hps_io_sdio_inst_D2
                hps_io_hps_io_sdio_inst_D3          : inout std_logic                     := '0';
--                             .hps_io_sdio_inst_D3
                hps_io_hps_io_usb1_inst_D0          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D0
                hps_io_hps_io_usb1_inst_D1          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D1
                hps_io_hps_io_usb1_inst_D2          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D2
                hps_io_hps_io_usb1_inst_D3          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D3
                hps_io_hps_io_usb1_inst_D4          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D4
                hps_io_hps_io_usb1_inst_D5          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D5
                hps_io_hps_io_usb1_inst_D6          : inout std_logic                     := '0';
--                             .hps_io_usb1_inst_D6
```

```
              hps_io_hps_io_usb1_inst_D7          : inout std_logic                     := '0';
--                            .hps_io_usb1_inst_D7
              hps_io_hps_io_usb1_inst_CLK         : in    std_logic                     := '0';
--                            .hps_io_usb1_inst_CLK
              hps_io_hps_io_usb1_inst_STP         : out   std_logic;
--                            .hps_io_usb1_inst_STP
              hps_io_hps_io_usb1_inst_DIR         : in    std_logic                     := '0';
--                            .hps_io_usb1_inst_DIR
              hps_io_hps_io_usb1_inst_NXT         : in    std_logic                     := '0';
--                            .hps_io_usb1_inst_NXT
              memory_mem_a                        : out   std_logic_vector(14 downto 0);
--                       memory.mem_a
              memory_mem_ba                       : out   std_logic_vector(2 downto 0);
--                            .mem_ba
              memory_mem_ck                       : out   std_logic;
--                            .mem_ck
              memory_mem_ck_n                     : out   std_logic;
--                            .mem_ck_n
              memory_mem_cke                      : out   std_logic;
--                            .mem_cke
              memory_mem_cs_n                     : out   std_logic;
--                            .mem_cs_n
              memory_mem_ras_n                    : out   std_logic;
--                            .mem_ras_n
              memory_mem_cas_n                    : out   std_logic;
--                            .mem_cas_n
              memory_mem_we_n                     : out   std_logic;
--                            .mem_we_n
              memory_mem_reset_n                  : out   std_logic;
--                            .mem_reset_n
              memory_mem_dq                       : inout std_logic_vector(31 downto 0) :=
(others => '0'); --                           .mem_dq
              memory_mem_dqs                      : inout std_logic_vector(3 downto 0)  :=
(others => '0'); --                           .mem_dqs
              memory_mem_dqs_n                    : inout std_logic_vector(3 downto 0)  :=
(others => '0'); --                           .mem_dqs_n
              memory_mem_odt                      : out   std_logic;
--                            .mem_odt
              memory_mem_dm                       : out   std_logic_vector(3 downto 0);
--                            .mem_dm
              memory_oct_rzqin                    : in    std_logic                     := '0';
--                            .oct_rzqin
              reset_reset_n                       : in    std_logic                     := '0';
--                      reset.reset_n
              mult_data_0_mult_data_m_in1         : out   std_logic_vector(31 downto 0);
--       mult_data_0_mult_data.m_in1
              mult_data_0_mult_data_m_in2         : out   std_logic_vector(31 downto 0);
--                            .m_in2
              mult_data_0_mult_data_m_result      : in    std_logic_vector(31 downto 0) :=
(others => '0'); --                           .m_result
              mult_control_0_mult_control_m_start : out   std_logic_vector(31 downto 0);
-- mult_control_0_mult_control.m_start
              mult_control_0_mult_control_m_reset : out   std_logic_vector(31 downto 0);
--                            .m_reset
              mult_control_0_mult_control_m_done  : in    std_logic_vector(31 downto 0) :=
(others => '0'); --                           .m_done
              div_control_0_div_control_d_start   : out   std_logic_vector(31 downto 0);
--   div_control_0_div_control.d_start
              div_control_0_div_control_d_reset   : out   std_logic_vector(31 downto 0);
--                            .d_reset
              div_control_0_div_control_d_done    : in    std_logic_vector(31 downto 0) :=
(others => '0'); --                           .d_done
              div_data_0_div_data_d_denom         : out   std_logic_vector(31 downto 0);
--         div_data_0_div_data.d_denom
              div_data_0_div_data_d_numer         : out   std_logic_vector(31 downto 0);
--                            .d_numer
              div_data_0_div_data_d_quot          : in    std_logic_vector(31 downto 0) :=
(others => '0'); --                           .d_quot
              div_data_0_div_data_d_rem           : in    std_logic_vector(31 downto 0) :=
(others => '0')  --                           .d_rem
       );
```

```
end entity soc_system;

architecture rtl of soc_system is
        component soc_system_hps_0 is
                generic (
                        F2S_Width : integer := 2;
                        S2F_Width : integer := 2
                );
                port (
                        mem_a                    : out   std_logic_vector(14 downto 0);
-- mem_a
                        mem_ba                   : out   std_logic_vector(2 downto 0);
-- mem_ba
                        mem_ck                   : out   std_logic;
-- mem_ck
                        mem_ck_n                 : out   std_logic;
-- mem_ck_n
                        mem_cke                  : out   std_logic;
-- mem_cke
                        mem_cs_n                 : out   std_logic;
-- mem_cs_n
                        mem_ras_n                : out   std_logic;
-- mem_ras_n
                        mem_cas_n                : out   std_logic;
-- mem_cas_n
                        mem_we_n                 : out   std_logic;
-- mem_we_n
                        mem_reset_n              : out   std_logic;
-- mem_reset_n
                        mem_dq                   : inout std_logic_vector(31 downto 0) := (others
=> 'X'); -- mem_dq
                        mem_dqs                  : inout std_logic_vector(3 downto 0)  := (others
=> 'X'); -- mem_dqs
                        mem_dqs_n                : inout std_logic_vector(3 downto 0)  := (others
=> 'X'); -- mem_dqs_n
                        mem_odt                  : out   std_logic;
-- mem_odt
                        mem_dm                   : out   std_logic_vector(3 downto 0);
-- mem_dm
                        oct_rzqin                : in    std_logic                     := 'X';
-- oct_rzqin
                        hps_io_emac1_inst_TX_CLK : out   std_logic;
-- hps_io_emac1_inst_TX_CLK
                        hps_io_emac1_inst_TXD0   : out   std_logic;
-- hps_io_emac1_inst_TXD0
                        hps_io_emac1_inst_TXD1   : out   std_logic;
-- hps_io_emac1_inst_TXD1
                        hps_io_emac1_inst_TXD2   : out   std_logic;
-- hps_io_emac1_inst_TXD2
                        hps_io_emac1_inst_TXD3   : out   std_logic;
-- hps_io_emac1_inst_TXD3
                        hps_io_emac1_inst_RXD0   : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD0
                        hps_io_emac1_inst_MDIO   : inout std_logic                     := 'X';
-- hps_io_emac1_inst_MDIO
                        hps_io_emac1_inst_MDC    : out   std_logic;
-- hps_io_emac1_inst_MDC
                        hps_io_emac1_inst_RX_CTL : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RX_CTL
                        hps_io_emac1_inst_TX_CTL : out   std_logic;
-- hps_io_emac1_inst_TX_CTL
                        hps_io_emac1_inst_RX_CLK : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RX_CLK
                        hps_io_emac1_inst_RXD1   : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD1
                        hps_io_emac1_inst_RXD2   : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD2
                        hps_io_emac1_inst_RXD3   : in    std_logic                     := 'X';
-- hps_io_emac1_inst_RXD3
                        hps_io_sdio_inst_CMD     : inout std_logic                     := 'X';
-- hps_io_sdio_inst_CMD
```

```
                      hps_io_sdio_inst_D0      : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D0
                      hps_io_sdio_inst_D1      : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D1
                      hps_io_sdio_inst_CLK     : out   std_logic;
-- hps_io_sdio_inst_CLK
                      hps_io_sdio_inst_D2      : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D2
                      hps_io_sdio_inst_D3      : inout std_logic                     := 'X';
-- hps_io_sdio_inst_D3
                      hps_io_usb1_inst_D0      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D0
                      hps_io_usb1_inst_D1      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D1
                      hps_io_usb1_inst_D2      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D2
                      hps_io_usb1_inst_D3      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D3
                      hps_io_usb1_inst_D4      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D4
                      hps_io_usb1_inst_D5      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D5
                      hps_io_usb1_inst_D6      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D6
                      hps_io_usb1_inst_D7      : inout std_logic                     := 'X';
-- hps_io_usb1_inst_D7
                      hps_io_usb1_inst_CLK     : in    std_logic                     := 'X';
-- hps_io_usb1_inst_CLK
                      hps_io_usb1_inst_STP     : out   std_logic;
-- hps_io_usb1_inst_STP
                      hps_io_usb1_inst_DIR     : in    std_logic                     := 'X';
-- hps_io_usb1_inst_DIR
                      hps_io_usb1_inst_NXT     : in    std_logic                     := 'X';
-- hps_io_usb1_inst_NXT
                      h2f_rst_n                : out   std_logic;
-- reset_n
                      h2f_axi_clk              : in    std_logic                     := 'X';
-- clk
                      h2f_AWID                 : out   std_logic_vector(11 downto 0);
-- awid
                      h2f_AWADDR               : out   std_logic_vector(29 downto 0);
-- awaddr
                      h2f_AWLEN                : out   std_logic_vector(3 downto 0);
-- awlen
                      h2f_AWSIZE               : out   std_logic_vector(2 downto 0);
-- awsize
                      h2f_AWBURST              : out   std_logic_vector(1 downto 0);
-- awburst
                      h2f_AWLOCK               : out   std_logic_vector(1 downto 0);
-- awlock
                      h2f_AWCACHE              : out   std_logic_vector(3 downto 0);
-- awcache
                      h2f_AWPROT               : out   std_logic_vector(2 downto 0);
-- awprot
                      h2f_AWVALID              : out   std_logic;
-- awvalid
                      h2f_AWREADY              : in    std_logic                     := 'X';
-- awready
                      h2f_WID                  : out   std_logic_vector(11 downto 0);
-- wid
                      h2f_WDATA                : out   std_logic_vector(63 downto 0);
-- wdata
                      h2f_WSTRB                : out   std_logic_vector(7 downto 0);
-- wstrb
                      h2f_WLAST                : out   std_logic;
-- wlast
                      h2f_WVALID               : out   std_logic;
-- wvalid
                      h2f_WREADY               : in    std_logic                     := 'X';
-- wready
```

```
h2f_BID                 : in    std_logic_vector(11 downto 0) := (others
=> 'X'); -- bid
h2f_BRESP               : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- bresp
h2f_BVALID              : in    std_logic                     := 'X';
-- bvalid
h2f_BREADY              : out   std_logic;
-- bready
h2f_ARID                : out   std_logic_vector(11 downto 0);
-- arid
h2f_ARADDR              : out   std_logic_vector(29 downto 0);
-- araddr
h2f_ARLEN               : out   std_logic_vector(3 downto 0);
-- arlen
h2f_ARSIZE              : out   std_logic_vector(2 downto 0);
-- arsize
h2f_ARBURST             : out   std_logic_vector(1 downto 0);
-- arburst
h2f_ARLOCK              : out   std_logic_vector(1 downto 0);
-- arlock
h2f_ARCACHE             : out   std_logic_vector(3 downto 0);
-- arcache
h2f_ARPROT              : out   std_logic_vector(2 downto 0);
-- arprot
h2f_ARVALID             : out   std_logic;
-- arvalid
h2f_ARREADY             : in    std_logic                     := 'X';
-- arready
h2f_RID                 : in    std_logic_vector(11 downto 0) := (others
=> 'X'); -- rid
h2f_RDATA               : in    std_logic_vector(63 downto 0) := (others
=> 'X'); -- rdata
h2f_RRESP               : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- rresp
h2f_RLAST               : in    std_logic                     := 'X';
-- rlast
h2f_RVALID              : in    std_logic                     := 'X';
-- rvalid
h2f_RREADY              : out   std_logic;
-- rready
f2h_axi_clk             : in    std_logic                     := 'X';
-- clk
f2h_AWID                : in    std_logic_vector(7 downto 0)  := (others
=> 'X'); -- awid
f2h_AWADDR              : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- awaddr
f2h_AWLEN               : in    std_logic_vector(3 downto 0)  := (others
=> 'X'); -- awlen
f2h_AWSIZE              : in    std_logic_vector(2 downto 0)  := (others
=> 'X'); -- awsize
f2h_AWBURST             : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- awburst
f2h_AWLOCK              : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- awlock
f2h_AWCACHE             : in    std_logic_vector(3 downto 0)  := (others
=> 'X'); -- awcache
f2h_AWPROT              : in    std_logic_vector(2 downto 0)  := (others
=> 'X'); -- awprot
f2h_AWVALID             : in    std_logic                     := 'X';
-- awvalid
f2h_AWREADY             : out   std_logic;
-- awready
f2h_AWUSER              : in    std_logic_vector(4 downto 0)  := (others
=> 'X'); -- awuser
f2h_WID                 : in    std_logic_vector(7 downto 0)  := (others
=> 'X'); -- wid
f2h_WDATA               : in    std_logic_vector(63 downto 0) := (others
=> 'X'); -- wdata
f2h_WSTRB               : in    std_logic_vector(7 downto 0)  := (others
=> 'X'); -- wstrb
```

```
                        f2h_WLAST               : in    std_logic                     := 'X';
-- wlast
                        f2h_WVALID              : in    std_logic                     := 'X';
-- wvalid
                        f2h_WREADY              : out   std_logic;
-- wready
                        f2h_BID                 : out   std_logic_vector(7 downto 0);
-- bid
                        f2h_BRESP               : out   std_logic_vector(1 downto 0);
-- bresp
                        f2h_BVALID              : out   std_logic;
-- bvalid
                        f2h_BREADY              : in    std_logic                     := 'X';
-- bready
                        f2h_ARID                : in    std_logic_vector(7 downto 0)  := (others
=> 'X'); -- arid
                        f2h_ARADDR              : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- araddr
                        f2h_ARLEN               : in    std_logic_vector(3 downto 0)  := (others
=> 'X'); -- arlen
                        f2h_ARSIZE              : in    std_logic_vector(2 downto 0)  := (others
=> 'X'); -- arsize
                        f2h_ARBURST             : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- arburst
                        f2h_ARLOCK              : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- arlock
                        f2h_ARCACHE             : in    std_logic_vector(3 downto 0)  := (others
=> 'X'); -- arcache
                        f2h_ARPROT              : in    std_logic_vector(2 downto 0)  := (others
=> 'X'); -- arprot
                        f2h_ARVALID             : in    std_logic                     := 'X';
-- arvalid
                        f2h_ARREADY             : out   std_logic;
-- arready
                        f2h_ARUSER              : in    std_logic_vector(4 downto 0)  := (others
=> 'X'); -- aruser
                        f2h_RID                 : out   std_logic_vector(7 downto 0);
-- rid
                        f2h_RDATA               : out   std_logic_vector(63 downto 0);
-- rdata
                        f2h_RRESP               : out   std_logic_vector(1 downto 0);
-- rresp
                        f2h_RLAST               : out   std_logic;
-- rlast
                        f2h_RVALID              : out   std_logic;
-- rvalid
                        f2h_RREADY              : in    std_logic                     := 'X';
-- rready
                        h2f_lw_axi_clk          : in    std_logic                     := 'X';
-- clk
                        h2f_lw_AWID             : out   std_logic_vector(11 downto 0);
-- awid
                        h2f_lw_AWADDR           : out   std_logic_vector(20 downto 0);
-- awaddr
                        h2f_lw_AWLEN            : out   std_logic_vector(3 downto 0);
-- awlen
                        h2f_lw_AWSIZE           : out   std_logic_vector(2 downto 0);
-- awsize
                        h2f_lw_AWBURST          : out   std_logic_vector(1 downto 0);
-- awburst
                        h2f_lw_AWLOCK           : out   std_logic_vector(1 downto 0);
-- awlock
                        h2f_lw_AWCACHE          : out   std_logic_vector(3 downto 0);
-- awcache
                        h2f_lw_AWPROT           : out   std_logic_vector(2 downto 0);
-- awprot
                        h2f_lw_AWVALID          : out   std_logic;
-- awvalid
                        h2f_lw_AWREADY          : in    std_logic                     := 'X';
-- awready
```

```
                        h2f_lw_WID              : out   std_logic_vector(11 downto 0);
-- wid
                        h2f_lw_WDATA            : out   std_logic_vector(31 downto 0);
-- wdata
                        h2f_lw_WSTRB            : out   std_logic_vector(3 downto 0);
-- wstrb
                        h2f_lw_WLAST            : out   std_logic;
-- wlast
                        h2f_lw_WVALID           : out   std_logic;
-- wvalid
                        h2f_lw_WREADY           : in    std_logic                     := 'X';
-- wready
                        h2f_lw_BID              : in    std_logic_vector(11 downto 0) := (others
=> 'X'); -- bid
                        h2f_lw_BRESP            : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- bresp
                        h2f_lw_BVALID           : in    std_logic                     := 'X';
-- bvalid
                        h2f_lw_BREADY           : out   std_logic;
-- bready
                        h2f_lw_ARID             : out   std_logic_vector(11 downto 0);
-- arid
                        h2f_lw_ARADDR           : out   std_logic_vector(20 downto 0);
-- araddr
                        h2f_lw_ARLEN            : out   std_logic_vector(3 downto 0);
-- arlen
                        h2f_lw_ARSIZE           : out   std_logic_vector(2 downto 0);
-- arsize
                        h2f_lw_ARBURST          : out   std_logic_vector(1 downto 0);
-- arburst
                        h2f_lw_ARLOCK           : out   std_logic_vector(1 downto 0);
-- arlock
                        h2f_lw_ARCACHE          : out   std_logic_vector(3 downto 0);
-- arcache
                        h2f_lw_ARPROT           : out   std_logic_vector(2 downto 0);
-- arprot
                        h2f_lw_ARVALID          : out   std_logic;
-- arvalid
                        h2f_lw_ARREADY          : in    std_logic                     := 'X';
-- arready
                        h2f_lw_RID              : in    std_logic_vector(11 downto 0) := (others
=> 'X'); -- rid
                        h2f_lw_RDATA            : in    std_logic_vector(31 downto 0) := (others
=> 'X'); -- rdata
                        h2f_lw_RRESP            : in    std_logic_vector(1 downto 0)  := (others
=> 'X'); -- rresp
                        h2f_lw_RLAST            : in    std_logic                     := 'X';
-- rlast
                        h2f_lw_RVALID           : in    std_logic                     := 'X';
-- rvalid
                        h2f_lw_RREADY           : out   std_logic
-- rready
                );
        end component soc_system_hps_0;

        component mult_data is
                port (
                        avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => 'X'); -
- address
                        avs_s0_read      : in  std_logic                     := 'X';             -
- read
                        avs_s0_write     : in  std_logic                     := 'X';             -
- write
                        avs_s0_readdata  : out std_logic_vector(31 downto 0);                    -
- readdata
                        avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => 'X'); -
- writedata
                        clk              : in  std_logic                     := 'X';             -
- clk
                        reset            : in  std_logic                     := 'X';             -
- reset
```

```
                        mult_in1         : out std_logic_vector(31 downto 0);                    -
- m_in1
                        mult_in2         : out std_logic_vector(31 downto 0);                    -
- m_in2
                        mult_result      : in  std_logic_vector(31 downto 0) := (others => 'X')  -
- m_result
                );
        end component mult_data;

        component mult_control is
                port (
                        avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => 'X'); -
- address
                        avs_s0_write     : in  std_logic                     := 'X';             -
- write
                        avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => 'X'); -
- writedata
                        avs_s0_read      : in  std_logic                     := 'X';             -
- read
                        avs_s0_readdata  : out std_logic_vector(31 downto 0);                    -
- readdata
                        clk              : in  std_logic                     := 'X';             -
- clk
                        reset            : in  std_logic                     := 'X';             -
- reset
                        mult_start       : out std_logic_vector(31 downto 0);                    -
- m_start
                        mult_reset       : out std_logic_vector(31 downto 0);                    -
- m_reset
                        mult_done        : in  std_logic_vector(31 downto 0) := (others => 'X')  -
- m_done
                );
        end component mult_control;

        component div_control is
                port (
                        avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => 'X'); -
- address
                        avs_s0_write     : in  std_logic                     := 'X';             -
- write
                        avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => 'X'); -
- writedata
                        avs_s0_read      : in  std_logic                     := 'X';             -
- read
                        avs_s0_readdata  : out std_logic_vector(31 downto 0);                    -
- readdata
                        clk              : in  std_logic                     := 'X';             -
- clk
                        reset            : in  std_logic                     := 'X';             -
- reset
                        div_start        : out std_logic_vector(31 downto 0);                    -
- d_start
                        div_reset        : out std_logic_vector(31 downto 0);                    -
- d_reset
                        div_done         : in  std_logic_vector(31 downto 0) := (others => 'X')  -
- d_done
                );
        end component div_control;

        component div_data is
                port (
                        avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => 'X'); -
- address
                        avs_s0_read      : in  std_logic                     := 'X';             -
- read
                        avs_s0_write     : in  std_logic                     := 'X';             -
- write
                        avs_s0_readdata  : out std_logic_vector(31 downto 0);                    -
- readdata
                        avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => 'X'); -
- writedata
```

```
                        clk             : in  std_logic                     := 'X';             -
- clk
                        reset           : in  std_logic                     := 'X';             -
- reset
                        div_denom       : out std_logic_vector(31 downto 0);                    -
- d_denom
                        div_numer       : out std_logic_vector(31 downto 0);                    -
- d_numer
                        div_quot        : in  std_logic_vector(31 downto 0) := (others => 'X'); -
- d_quot
                        div_rem         : in  std_logic_vector(31 downto 0) := (others => 'X')  -
- d_rem
                );
        end component div_data;

        component soc_system_mm_interconnect_0 is
                port (
                        hps_0_h2f_lw_axi_master_awid                                    : in
std_logic_vector(11 downto 0) := (others => 'X'); -- awid
                        hps_0_h2f_lw_axi_master_awaddr                                  : in
std_logic_vector(20 downto 0) := (others => 'X'); -- awaddr
                        hps_0_h2f_lw_axi_master_awlen                                   : in
std_logic_vector(3 downto 0)  := (others => 'X'); -- awlen
                        hps_0_h2f_lw_axi_master_awsize                                  : in
std_logic_vector(2 downto 0)  := (others => 'X'); -- awsize
                        hps_0_h2f_lw_axi_master_awburst                                 : in
std_logic_vector(1 downto 0)  := (others => 'X'); -- awburst
                        hps_0_h2f_lw_axi_master_awlock                                  : in
std_logic_vector(1 downto 0)  := (others => 'X'); -- awlock
                        hps_0_h2f_lw_axi_master_awcache                                 : in
std_logic_vector(3 downto 0)  := (others => 'X'); -- awcache
                        hps_0_h2f_lw_axi_master_awprot                                  : in
std_logic_vector(2 downto 0)  := (others => 'X'); -- awprot
                        hps_0_h2f_lw_axi_master_awvalid                                 : in
std_logic                     := 'X';             -- awvalid
                        hps_0_h2f_lw_axi_master_awready                                 : out
std_logic;                                        -- awready
                        hps_0_h2f_lw_axi_master_wid                                     : in
std_logic_vector(11 downto 0) := (others => 'X'); -- wid
                        hps_0_h2f_lw_axi_master_wdata                                   : in
std_logic_vector(31 downto 0) := (others => 'X'); -- wdata
                        hps_0_h2f_lw_axi_master_wstrb                                   : in
std_logic_vector(3 downto 0)  := (others => 'X'); -- wstrb
                        hps_0_h2f_lw_axi_master_wlast                                   : in
std_logic                     := 'X';             -- wlast
                        hps_0_h2f_lw_axi_master_wvalid                                  : in
std_logic                     := 'X';             -- wvalid
                        hps_0_h2f_lw_axi_master_wready                                  : out
std_logic;                                        -- wready
                        hps_0_h2f_lw_axi_master_bid                                     : out
std_logic_vector(11 downto 0);                    -- bid
                        hps_0_h2f_lw_axi_master_bresp                                   : out
std_logic_vector(1 downto 0);                     -- bresp
                        hps_0_h2f_lw_axi_master_bvalid                                  : out
std_logic;                                        -- bvalid
                        hps_0_h2f_lw_axi_master_bready                                  : in
std_logic                     := 'X';             -- bready
                        hps_0_h2f_lw_axi_master_arid                                    : in
std_logic_vector(11 downto 0) := (others => 'X'); -- arid
                        hps_0_h2f_lw_axi_master_araddr                                  : in
std_logic_vector(20 downto 0) := (others => 'X'); -- araddr
                        hps_0_h2f_lw_axi_master_arlen                                   : in
std_logic_vector(3 downto 0)  := (others => 'X'); -- arlen
                        hps_0_h2f_lw_axi_master_arsize                                  : in
std_logic_vector(2 downto 0)  := (others => 'X'); -- arsize
                        hps_0_h2f_lw_axi_master_arburst                                 : in
std_logic_vector(1 downto 0)  := (others => 'X'); -- arburst
                        hps_0_h2f_lw_axi_master_arlock                                  : in
std_logic_vector(1 downto 0)  := (others => 'X'); -- arlock
                        hps_0_h2f_lw_axi_master_arcache                                 : in
std_logic_vector(3 downto 0)  := (others => 'X'); -- arcache
```

```
                        hps_0_h2f_lw_axi_master_arprot                                      : in
std_logic_vector(2 downto 0)  := (others => 'X'); -- arprot
                        hps_0_h2f_lw_axi_master_arvalid                                     : in
std_logic                     := 'X';             -- arvalid
                        hps_0_h2f_lw_axi_master_arready                                     : out
std_logic;                                        -- arready
                        hps_0_h2f_lw_axi_master_rid                                         : out
std_logic_vector(11 downto 0);                    -- rid
                        hps_0_h2f_lw_axi_master_rdata                                       : out
std_logic_vector(31 downto 0);                    -- rdata
                        hps_0_h2f_lw_axi_master_rresp                                       : out
std_logic_vector(1 downto 0);                     -- rresp
                        hps_0_h2f_lw_axi_master_rlast                                       : out
std_logic;                                        -- rlast
                        hps_0_h2f_lw_axi_master_rvalid                                      : out
std_logic;                                        -- rvalid
                        hps_0_h2f_lw_axi_master_rready                                      : in
std_logic                     := 'X';             -- rready
                        clk_0_clk_clk                                                       : in
std_logic                     := 'X';             -- clk
                        hps_0_h2f_lw_axi_master_agent_clk_reset_reset_bridge_in_reset_reset : in
std_logic                     := 'X';             -- reset
                        mult_data_0_reset_reset_bridge_in_reset_reset                       : in
std_logic                     := 'X';             -- reset
                        div_control_0_s0_address                                            : out
std_logic_vector(3 downto 0);                     -- address
                        div_control_0_s0_write                                              : out
std_logic;                                        -- write
                        div_control_0_s0_read                                               : out
std_logic;                                        -- read
                        div_control_0_s0_readdata                                           : in
std_logic_vector(31 downto 0) := (others => 'X'); -- readdata
                        div_control_0_s0_writedata                                          : out
std_logic_vector(31 downto 0);                    -- writedata
                        div_data_0_s0_address                                               : out
std_logic_vector(3 downto 0);                     -- address
                        div_data_0_s0_write                                                 : out
std_logic;                                        -- write
                        div_data_0_s0_read                                                  : out
std_logic;                                        -- read
                        div_data_0_s0_readdata                                              : in
std_logic_vector(31 downto 0) := (others => 'X'); -- readdata
                        div_data_0_s0_writedata                                             : out
std_logic_vector(31 downto 0);                    -- writedata
                        mult_control_0_s0_address                                           : out
std_logic_vector(3 downto 0);                     -- address
                        mult_control_0_s0_write                                             : out
std_logic;                                        -- write
                        mult_control_0_s0_read                                              : out
std_logic;                                        -- read
                        mult_control_0_s0_readdata                                          : in
std_logic_vector(31 downto 0) := (others => 'X'); -- readdata
                        mult_control_0_s0_writedata                                         : out
std_logic_vector(31 downto 0);                    -- writedata
                        mult_data_0_s0_address                                              : out
std_logic_vector(3 downto 0);                     -- address
                        mult_data_0_s0_write                                                : out
std_logic;                                        -- write
                        mult_data_0_s0_read                                                 : out
std_logic;                                        -- read
                        mult_data_0_s0_readdata                                             : in
std_logic_vector(31 downto 0) := (others => 'X'); -- readdata
                        mult_data_0_s0_writedata                                            : out
std_logic_vector(31 downto 0)                     -- writedata
                );
        end component soc_system_mm_interconnect_0;

        component altera_reset_controller is
                generic (
                        NUM_RESET_INPUTS          : integer := 6;
                        OUTPUT_RESET_SYNC_EDGES   : string  := "deassert";
```

```
                SYNC_DEPTH                : integer := 2;
                RESET_REQUEST_PRESENT     : integer := 0;
                RESET_REQ_WAIT_TIME       : integer := 1;
                MIN_RST_ASSERTION_TIME    : integer := 3;
                RESET_REQ_EARLY_DSRT_TIME : integer := 1;
                USE_RESET_REQUEST_IN0     : integer := 0;
                USE_RESET_REQUEST_IN1     : integer := 0;
                USE_RESET_REQUEST_IN2     : integer := 0;
                USE_RESET_REQUEST_IN3     : integer := 0;
                USE_RESET_REQUEST_IN4     : integer := 0;
                USE_RESET_REQUEST_IN5     : integer := 0;
                USE_RESET_REQUEST_IN6     : integer := 0;
                USE_RESET_REQUEST_IN7     : integer := 0;
                USE_RESET_REQUEST_IN8     : integer := 0;
                USE_RESET_REQUEST_IN9     : integer := 0;
                USE_RESET_REQUEST_IN10    : integer := 0;
                USE_RESET_REQUEST_IN11    : integer := 0;
                USE_RESET_REQUEST_IN12    : integer := 0;
                USE_RESET_REQUEST_IN13    : integer := 0;
                USE_RESET_REQUEST_IN14    : integer := 0;
                USE_RESET_REQUEST_IN15    : integer := 0;
                ADAPT_RESET_REQUEST       : integer := 0
        );
        port (
                reset_in0      : in  std_logic := 'X'; -- reset
                clk            : in  std_logic := 'X'; -- clk
                reset_out      : out std_logic;        -- reset
                reset_req      : out std_logic;        -- reset_req
                reset_req_in0  : in  std_logic := 'X'; -- reset_req
                reset_in1      : in  std_logic := 'X'; -- reset
                reset_req_in1  : in  std_logic := 'X'; -- reset_req
                reset_in2      : in  std_logic := 'X'; -- reset
                reset_req_in2  : in  std_logic := 'X'; -- reset_req
                reset_in3      : in  std_logic := 'X'; -- reset
                reset_req_in3  : in  std_logic := 'X'; -- reset_req
                reset_in4      : in  std_logic := 'X'; -- reset
                reset_req_in4  : in  std_logic := 'X'; -- reset_req
                reset_in5      : in  std_logic := 'X'; -- reset
                reset_req_in5  : in  std_logic := 'X'; -- reset_req
                reset_in6      : in  std_logic := 'X'; -- reset
                reset_req_in6  : in  std_logic := 'X'; -- reset_req
                reset_in7      : in  std_logic := 'X'; -- reset
                reset_req_in7  : in  std_logic := 'X'; -- reset_req
                reset_in8      : in  std_logic := 'X'; -- reset
                reset_req_in8  : in  std_logic := 'X'; -- reset_req
                reset_in9      : in  std_logic := 'X'; -- reset
                reset_req_in9  : in  std_logic := 'X'; -- reset_req
                reset_in10     : in  std_logic := 'X'; -- reset
                reset_req_in10 : in  std_logic := 'X'; -- reset_req
                reset_in11     : in  std_logic := 'X'; -- reset
                reset_req_in11 : in  std_logic := 'X'; -- reset_req
                reset_in12     : in  std_logic := 'X'; -- reset
                reset_req_in12 : in  std_logic := 'X'; -- reset_req
                reset_in13     : in  std_logic := 'X'; -- reset
                reset_req_in13 : in  std_logic := 'X'; -- reset_req
                reset_in14     : in  std_logic := 'X'; -- reset
                reset_req_in14 : in  std_logic := 'X'; -- reset_req
                reset_in15     : in  std_logic := 'X'; -- reset
                reset_req_in15 : in  std_logic := 'X'  -- reset_req
        );
    end component altera_reset_controller;

    signal hps_0_h2f_reset_reset                        : std_logic;                    --
hps_0:h2f_rst_n -> [hps_0_h2f_reset_reset_n, hps_0_h2f_reset_reset_n:in]
    signal hps_0_h2f_lw_axi_master_awvalid              : std_logic;                    --
hps_0:h2f_lw_AWVALID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awvalid
    signal hps_0_h2f_lw_axi_master_arsize               : std_logic_vector(2 downto 0); --
hps_0:h2f_lw_ARSIZE -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arsize
    signal hps_0_h2f_lw_axi_master_arlock               : std_logic_vector(1 downto 0); --
hps_0:h2f_lw_ARLOCK -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arlock
```

```
        signal hps_0_h2f_lw_axi_master_awcache              : std_logic_vector(3 downto 0);  --
hps_0:h2f_lw_AWCACHE -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awcache
        signal hps_0_h2f_lw_axi_master_arready              : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_arready -> hps_0:h2f_lw_ARREADY
        signal hps_0_h2f_lw_axi_master_arid                 : std_logic_vector(11 downto 0); --
hps_0:h2f_lw_ARID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arid
        signal hps_0_h2f_lw_axi_master_rready               : std_logic;                     --
hps_0:h2f_lw_RREADY -> mm_interconnect_0:hps_0_h2f_lw_axi_master_rready
        signal hps_0_h2f_lw_axi_master_bready               : std_logic;                     --
hps_0:h2f_lw_BREADY -> mm_interconnect_0:hps_0_h2f_lw_axi_master_bready
        signal hps_0_h2f_lw_axi_master_awsize               : std_logic_vector(2 downto 0);  --
hps_0:h2f_lw_AWSIZE -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awsize
        signal hps_0_h2f_lw_axi_master_awprot               : std_logic_vector(2 downto 0);  --
hps_0:h2f_lw_AWPROT -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awprot
        signal hps_0_h2f_lw_axi_master_arvalid              : std_logic;                     --
hps_0:h2f_lw_ARVALID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arvalid
        signal hps_0_h2f_lw_axi_master_arprot               : std_logic_vector(2 downto 0);  --
hps_0:h2f_lw_ARPROT -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arprot
        signal hps_0_h2f_lw_axi_master_bid                  : std_logic_vector(11 downto 0); --
mm_interconnect_0:hps_0_h2f_lw_axi_master_bid -> hps_0:h2f_lw_BID
        signal hps_0_h2f_lw_axi_master_arlen                : std_logic_vector(3 downto 0);  --
hps_0:h2f_lw_ARLEN -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arlen
        signal hps_0_h2f_lw_axi_master_awready              : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_awready -> hps_0:h2f_lw_AWREADY
        signal hps_0_h2f_lw_axi_master_awid                 : std_logic_vector(11 downto 0); --
hps_0:h2f_lw_AWID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awid
        signal hps_0_h2f_lw_axi_master_bvalid               : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_bvalid -> hps_0:h2f_lw_BVALID
        signal hps_0_h2f_lw_axi_master_wid                  : std_logic_vector(11 downto 0); --
hps_0:h2f_lw_WID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_wid
        signal hps_0_h2f_lw_axi_master_awlock               : std_logic_vector(1 downto 0);  --
hps_0:h2f_lw_AWLOCK -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awlock
        signal hps_0_h2f_lw_axi_master_awburst              : std_logic_vector(1 downto 0);  --
hps_0:h2f_lw_AWBURST -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awburst
        signal hps_0_h2f_lw_axi_master_bresp                : std_logic_vector(1 downto 0);  --
mm_interconnect_0:hps_0_h2f_lw_axi_master_bresp -> hps_0:h2f_lw_BRESP
        signal hps_0_h2f_lw_axi_master_wstrb                : std_logic_vector(3 downto 0);  --
hps_0:h2f_lw_WSTRB -> mm_interconnect_0:hps_0_h2f_lw_axi_master_wstrb
        signal hps_0_h2f_lw_axi_master_rvalid               : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_rvalid -> hps_0:h2f_lw_RVALID
        signal hps_0_h2f_lw_axi_master_wdata                : std_logic_vector(31 downto 0); --
hps_0:h2f_lw_WDATA -> mm_interconnect_0:hps_0_h2f_lw_axi_master_wdata
        signal hps_0_h2f_lw_axi_master_wready               : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_wready -> hps_0:h2f_lw_WREADY
        signal hps_0_h2f_lw_axi_master_arburst              : std_logic_vector(1 downto 0);  --
hps_0:h2f_lw_ARBURST -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arburst
        signal hps_0_h2f_lw_axi_master_rdata                : std_logic_vector(31 downto 0); --
mm_interconnect_0:hps_0_h2f_lw_axi_master_rdata -> hps_0:h2f_lw_RDATA
        signal hps_0_h2f_lw_axi_master_araddr               : std_logic_vector(20 downto 0); --
hps_0:h2f_lw_ARADDR -> mm_interconnect_0:hps_0_h2f_lw_axi_master_araddr
        signal hps_0_h2f_lw_axi_master_arcache              : std_logic_vector(3 downto 0);  --
hps_0:h2f_lw_ARCACHE -> mm_interconnect_0:hps_0_h2f_lw_axi_master_arcache
        signal hps_0_h2f_lw_axi_master_awlen                : std_logic_vector(3 downto 0);  --
hps_0:h2f_lw_AWLEN -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awlen
        signal hps_0_h2f_lw_axi_master_awaddr               : std_logic_vector(20 downto 0); --
hps_0:h2f_lw_AWADDR -> mm_interconnect_0:hps_0_h2f_lw_axi_master_awaddr
        signal hps_0_h2f_lw_axi_master_rid                  : std_logic_vector(11 downto 0); --
mm_interconnect_0:hps_0_h2f_lw_axi_master_rid -> hps_0:h2f_lw_RID
        signal hps_0_h2f_lw_axi_master_wvalid               : std_logic;                     --
hps_0:h2f_lw_WVALID -> mm_interconnect_0:hps_0_h2f_lw_axi_master_wvalid
        signal hps_0_h2f_lw_axi_master_rresp                : std_logic_vector(1 downto 0);  --
mm_interconnect_0:hps_0_h2f_lw_axi_master_rresp -> hps_0:h2f_lw_RRESP
        signal hps_0_h2f_lw_axi_master_wlast                : std_logic;                     --
hps_0:h2f_lw_WLAST -> mm_interconnect_0:hps_0_h2f_lw_axi_master_wlast
        signal hps_0_h2f_lw_axi_master_rlast                : std_logic;                     --
mm_interconnect_0:hps_0_h2f_lw_axi_master_rlast -> hps_0:h2f_lw_RLAST
        signal mm_interconnect_0_mult_data_0_s0_writedata   : std_logic_vector(31 downto 0); --
mm_interconnect_0:mult_data_0_s0_writedata -> mult_data_0:avs_s0_writedata
        signal mm_interconnect_0_mult_data_0_s0_address     : std_logic_vector(3 downto 0);  --
mm_interconnect_0:mult_data_0_s0_address -> mult_data_0:avs_s0_address
```

```
        signal mm_interconnect_0_mult_data_0_s0_write        : std_logic;                     --
mm_interconnect_0:mult_data_0_s0_write -> mult_data_0:avs_s0_write
        signal mm_interconnect_0_mult_data_0_s0_read         : std_logic;                     --
mm_interconnect_0:mult_data_0_s0_read -> mult_data_0:avs_s0_read
        signal mm_interconnect_0_mult_data_0_s0_readdata     : std_logic_vector(31 downto 0); --
mult_data_0:avs_s0_readdata -> mm_interconnect_0:mult_data_0_s0_readdata
        signal mm_interconnect_0_mult_control_0_s0_writedata : std_logic_vector(31 downto 0); --
mm_interconnect_0:mult_control_0_s0_writedata -> mult_control_0:avs_s0_writedata
        signal mm_interconnect_0_mult_control_0_s0_address   : std_logic_vector(3 downto 0);  --
mm_interconnect_0:mult_control_0_s0_address -> mult_control_0:avs_s0_address
        signal mm_interconnect_0_mult_control_0_s0_write     : std_logic;                     --
mm_interconnect_0:mult_control_0_s0_write -> mult_control_0:avs_s0_write
        signal mm_interconnect_0_mult_control_0_s0_read      : std_logic;                     --
mm_interconnect_0:mult_control_0_s0_read -> mult_control_0:avs_s0_read
        signal mm_interconnect_0_mult_control_0_s0_readdata  : std_logic_vector(31 downto 0); --
mult_control_0:avs_s0_readdata -> mm_interconnect_0:mult_control_0_s0_readdata
        signal mm_interconnect_0_div_control_0_s0_writedata  : std_logic_vector(31 downto 0); --
mm_interconnect_0:div_control_0_s0_writedata -> div_control_0:avs_s0_writedata
        signal mm_interconnect_0_div_control_0_s0_address    : std_logic_vector(3 downto 0);  --
mm_interconnect_0:div_control_0_s0_address -> div_control_0:avs_s0_address
        signal mm_interconnect_0_div_control_0_s0_write      : std_logic;                     --
mm_interconnect_0:div_control_0_s0_write -> div_control_0:avs_s0_write
        signal mm_interconnect_0_div_control_0_s0_read       : std_logic;                     --
mm_interconnect_0:div_control_0_s0_read -> div_control_0:avs_s0_read
        signal mm_interconnect_0_div_control_0_s0_readdata   : std_logic_vector(31 downto 0); --
div_control_0:avs_s0_readdata -> mm_interconnect_0:div_control_0_s0_readdata
        signal mm_interconnect_0_div_data_0_s0_writedata     : std_logic_vector(31 downto 0); --
mm_interconnect_0:div_data_0_s0_writedata -> div_data_0:avs_s0_writedata
        signal mm_interconnect_0_div_data_0_s0_address       : std_logic_vector(3 downto 0);  --
mm_interconnect_0:div_data_0_s0_address -> div_data_0:avs_s0_address
        signal mm_interconnect_0_div_data_0_s0_write         : std_logic;                     --
mm_interconnect_0:div_data_0_s0_write -> div_data_0:avs_s0_write
        signal mm_interconnect_0_div_data_0_s0_read          : std_logic;                     --
mm_interconnect_0:div_data_0_s0_read -> div_data_0:avs_s0_read
        signal mm_interconnect_0_div_data_0_s0_readdata      : std_logic_vector(31 downto 0); --
div_data_0:avs_s0_readdata -> mm_interconnect_0:div_data_0_s0_readdata
        signal rst_controller_reset_out_reset                : std_logic;                     --
rst_controller:reset_out -> [div_control_0:reset, div_data_0:reset,
mm_interconnect_0:mult_data_0_reset_reset_bridge_in_reset_reset, mult_control_0:reset,
mult_data_0:reset]
        signal rst_controller_001_reset_out_reset            : std_logic;                     --
rst_controller_001:reset_out ->
mm_interconnect_0:hps_0_h2f_lw_axi_master_agent_clk_reset_reset_bridge_in_reset_reset
        signal hps_0_h2f_reset_reset_n_ports_inv             : std_logic;                     --
hps_0_h2f_reset_reset_n:inv -> rst_controller_001:reset_in0
        signal reset_reset_n_ports_inv                       : std_logic;                     --
reset_reset_n:inv -> rst_controller:reset_in0

begin

        hps_0 : component soc_system_hps_0
                generic map (
                        F2S_Width => 2,
                        S2F_Width => 2
                )
                port map (
                        mem_a                   => memory_mem_a,                    --
memory.mem_a
                        mem_ba                  => memory_mem_ba,                   --
.mem_ba
                        mem_ck                  => memory_mem_ck,                   --
.mem_ck
                        mem_ck_n                => memory_mem_ck_n,                 --
.mem_ck_n
                        mem_cke                 => memory_mem_cke,                  --
.mem_cke
                        mem_cs_n                => memory_mem_cs_n,                 --
.mem_cs_n
                        mem_ras_n               => memory_mem_ras_n,                --
.mem_ras_n
```

```
                        mem_cas_n                => memory_mem_cas_n,                --
.mem_cas_n
                        mem_we_n                 => memory_mem_we_n,                 --
.mem_we_n
                        mem_reset_n              => memory_mem_reset_n,              --
.mem_reset_n
                        mem_dq                   => memory_mem_dq,                   --
.mem_dq
                        mem_dqs                  => memory_mem_dqs,                  --
.mem_dqs
                        mem_dqs_n                => memory_mem_dqs_n,                --
.mem_dqs_n
                        mem_odt                  => memory_mem_odt,                  --
.mem_odt
                        mem_dm                   => memory_mem_dm,                   --
.mem_dm
                        oct_rzqin                => memory_oct_rzqin,                --
.oct_rzqin
                        hps_io_emac1_inst_TX_CLK => hps_io_hps_io_emac1_inst_TX_CLK, --
hps_io.hps_io_emac1_inst_TX_CLK
                        hps_io_emac1_inst_TXD0   => hps_io_hps_io_emac1_inst_TXD0,   --
.hps_io_emac1_inst_TXD0
                        hps_io_emac1_inst_TXD1   => hps_io_hps_io_emac1_inst_TXD1,   --
.hps_io_emac1_inst_TXD1
                        hps_io_emac1_inst_TXD2   => hps_io_hps_io_emac1_inst_TXD2,   --
.hps_io_emac1_inst_TXD2
                        hps_io_emac1_inst_TXD3   => hps_io_hps_io_emac1_inst_TXD3,   --
.hps_io_emac1_inst_TXD3
                        hps_io_emac1_inst_RXD0   => hps_io_hps_io_emac1_inst_RXD0,   --
.hps_io_emac1_inst_RXD0
                        hps_io_emac1_inst_MDIO   => hps_io_hps_io_emac1_inst_MDIO,   --
.hps_io_emac1_inst_MDIO
                        hps_io_emac1_inst_MDC    => hps_io_hps_io_emac1_inst_MDC,    --
.hps_io_emac1_inst_MDC
                        hps_io_emac1_inst_RX_CTL => hps_io_hps_io_emac1_inst_RX_CTL, --
.hps_io_emac1_inst_RX_CTL
                        hps_io_emac1_inst_TX_CTL => hps_io_hps_io_emac1_inst_TX_CTL, --
.hps_io_emac1_inst_TX_CTL
                        hps_io_emac1_inst_RX_CLK => hps_io_hps_io_emac1_inst_RX_CLK, --
.hps_io_emac1_inst_RX_CLK
                        hps_io_emac1_inst_RXD1   => hps_io_hps_io_emac1_inst_RXD1,   --
.hps_io_emac1_inst_RXD1
                        hps_io_emac1_inst_RXD2   => hps_io_hps_io_emac1_inst_RXD2,   --
.hps_io_emac1_inst_RXD2
                        hps_io_emac1_inst_RXD3   => hps_io_hps_io_emac1_inst_RXD3,   --
.hps_io_emac1_inst_RXD3
                        hps_io_sdio_inst_CMD     => hps_io_hps_io_sdio_inst_CMD,     --
.hps_io_sdio_inst_CMD
                        hps_io_sdio_inst_D0      => hps_io_hps_io_sdio_inst_D0,      --
.hps_io_sdio_inst_D0
                        hps_io_sdio_inst_D1      => hps_io_hps_io_sdio_inst_D1,      --
.hps_io_sdio_inst_D1
                        hps_io_sdio_inst_CLK     => hps_io_hps_io_sdio_inst_CLK,     --
.hps_io_sdio_inst_CLK
                        hps_io_sdio_inst_D2      => hps_io_hps_io_sdio_inst_D2,      --
.hps_io_sdio_inst_D2
                        hps_io_sdio_inst_D3      => hps_io_hps_io_sdio_inst_D3,      --
.hps_io_sdio_inst_D3
                        hps_io_usb1_inst_D0      => hps_io_hps_io_usb1_inst_D0,      --
.hps_io_usb1_inst_D0
                        hps_io_usb1_inst_D1      => hps_io_hps_io_usb1_inst_D1,      --
.hps_io_usb1_inst_D1
                        hps_io_usb1_inst_D2      => hps_io_hps_io_usb1_inst_D2,      --
.hps_io_usb1_inst_D2
                        hps_io_usb1_inst_D3      => hps_io_hps_io_usb1_inst_D3,      --
.hps_io_usb1_inst_D3
                        hps_io_usb1_inst_D4      => hps_io_hps_io_usb1_inst_D4,      --
.hps_io_usb1_inst_D4
                        hps_io_usb1_inst_D5      => hps_io_hps_io_usb1_inst_D5,      --
.hps_io_usb1_inst_D5
```

```
                    hps_io_usb1_inst_D6      => hps_io_hps_io_usb1_inst_D6,      --
.hps_io_usb1_inst_D6
                    hps_io_usb1_inst_D7      => hps_io_hps_io_usb1_inst_D7,      --
.hps_io_usb1_inst_D7
                    hps_io_usb1_inst_CLK     => hps_io_hps_io_usb1_inst_CLK,     --
.hps_io_usb1_inst_CLK
                    hps_io_usb1_inst_STP     => hps_io_hps_io_usb1_inst_STP,     --
.hps_io_usb1_inst_STP
                    hps_io_usb1_inst_DIR     => hps_io_hps_io_usb1_inst_DIR,     --
.hps_io_usb1_inst_DIR
                    hps_io_usb1_inst_NXT     => hps_io_hps_io_usb1_inst_NXT,     --
.hps_io_usb1_inst_NXT
                    h2f_rst_n                => hps_0_h2f_reset_reset,           --
h2f_reset.reset_n
                    h2f_axi_clk              => clk_clk,                         --
h2f_axi_clock.clk
                    h2f_AWID                 => open,                            --
h2f_axi_master.awid
                    h2f_AWADDR               => open,                            --
.awaddr
                    h2f_AWLEN                => open,                            --
.awlen
                    h2f_AWSIZE               => open,                            --
.awsize
                    h2f_AWBURST              => open,                            --
.awburst
                    h2f_AWLOCK               => open,                            --
.awlock
                    h2f_AWCACHE              => open,                            --
.awcache
                    h2f_AWPROT               => open,                            --
.awprot
                    h2f_AWVALID              => open,                            --
.awvalid
                    h2f_AWREADY              => open,                            --
.awready
                    h2f_WID                  => open,                            --
.wid
                    h2f_WDATA                => open,                            --
.wdata
                    h2f_WSTRB                => open,                            --
.wstrb
                    h2f_WLAST                => open,                            --
.wlast
                    h2f_WVALID               => open,                            --
.wvalid
                    h2f_WREADY               => open,                            --
.wready
                    h2f_BID                  => open,                            --
.bid
                    h2f_BRESP                => open,                            --
.bresp
                    h2f_BVALID               => open,                            --
.bvalid
                    h2f_BREADY               => open,                            --
.bready
                    h2f_ARID                 => open,                            --
.arid
                    h2f_ARADDR               => open,                            --
.araddr
                    h2f_ARLEN                => open,                            --
.arlen
                    h2f_ARSIZE               => open,                            --
.arsize
                    h2f_ARBURST              => open,                            --
.arburst
                    h2f_ARLOCK               => open,                            --
.arlock
                    h2f_ARCACHE              => open,                            --
.arcache
```

```
                    h2f_ARPROT              => open,                        --
.arprot
                    h2f_ARVALID             => open,                        --
.arvalid
                    h2f_ARREADY             => open,                        --
.arready
                    h2f_RID                 => open,                        --
.rid
                    h2f_RDATA               => open,                        --
.rdata
                    h2f_RRESP               => open,                        --
.rresp
                    h2f_RLAST               => open,                        --
.rlast
                    h2f_RVALID              => open,                        --
.rvalid
                    h2f_RREADY              => open,                        --
.rready
                    f2h_axi_clk             => clk_clk,                     --
f2h_axi_clock.clk
                    f2h_AWID                => open,                        --
f2h_axi_slave.awid
                    f2h_AWADDR              => open,                        --
.awaddr
                    f2h_AWLEN               => open,                        --
.awlen
                    f2h_AWSIZE              => open,                        --
.awsize
                    f2h_AWBURST             => open,                        --
.awburst
                    f2h_AWLOCK              => open,                        --
.awlock
                    f2h_AWCACHE             => open,                        --
.awcache
                    f2h_AWPROT              => open,                        --
.awprot
                    f2h_AWVALID             => open,                        --
.awvalid
                    f2h_AWREADY             => open,                        --
.awready
                    f2h_AWUSER              => open,                        --
.awuser
                    f2h_WID                 => open,                        --
.wid
                    f2h_WDATA               => open,                        --
.wdata
                    f2h_WSTRB               => open,                        --
.wstrb
                    f2h_WLAST               => open,                        --
.wlast
                    f2h_WVALID              => open,                        --
.wvalid
                    f2h_WREADY              => open,                        --
.wready
                    f2h_BID                 => open,                        --
.bid
                    f2h_BRESP               => open,                        --
.bresp
                    f2h_BVALID              => open,                        --
.bvalid
                    f2h_BREADY              => open,                        --
.bready
                    f2h_ARID                => open,                        --
.arid
                    f2h_ARADDR              => open,                        --
.araddr
                    f2h_ARLEN               => open,                        --
.arlen
                    f2h_ARSIZE              => open,                        --
.arsize
```

```
                    f2h_ARBURST             => open,                            --
.arburst
                    f2h_ARLOCK              => open,                            --
.arlock
                    f2h_ARCACHE             => open,                            --
.arcache
                    f2h_ARPROT              => open,                            --
.arprot
                    f2h_ARVALID             => open,                            --
.arvalid
                    f2h_ARREADY             => open,                            --
.arready
                    f2h_ARUSER              => open,                            --
.aruser
                    f2h_RID                 => open,                            --
.rid
                    f2h_RDATA               => open,                            --
.rdata
                    f2h_RRESP               => open,                            --
.rresp
                    f2h_RLAST               => open,                            --
.rlast
                    f2h_RVALID              => open,                            --
.rvalid
                    f2h_RREADY              => open,                            --
.rready
                    h2f_lw_axi_clk          => clk_clk,                         --
h2f_lw_axi_clock.clk
                    h2f_lw_AWID             => hps_0_h2f_lw_axi_master_awid,    --
h2f_lw_axi_master.awid
                    h2f_lw_AWADDR           => hps_0_h2f_lw_axi_master_awaddr,  --
.awaddr
                    h2f_lw_AWLEN            => hps_0_h2f_lw_axi_master_awlen,   --
.awlen
                    h2f_lw_AWSIZE           => hps_0_h2f_lw_axi_master_awsize,  --
.awsize
                    h2f_lw_AWBURST          => hps_0_h2f_lw_axi_master_awburst, --
.awburst
                    h2f_lw_AWLOCK           => hps_0_h2f_lw_axi_master_awlock,  --
.awlock
                    h2f_lw_AWCACHE          => hps_0_h2f_lw_axi_master_awcache, --
.awcache
                    h2f_lw_AWPROT           => hps_0_h2f_lw_axi_master_awprot,  --
.awprot
                    h2f_lw_AWVALID          => hps_0_h2f_lw_axi_master_awvalid, --
.awvalid
                    h2f_lw_AWREADY          => hps_0_h2f_lw_axi_master_awready, --
.awready
                    h2f_lw_WID              => hps_0_h2f_lw_axi_master_wid,     --
.wid
                    h2f_lw_WDATA            => hps_0_h2f_lw_axi_master_wdata,   --
.wdata
                    h2f_lw_WSTRB            => hps_0_h2f_lw_axi_master_wstrb,   --
.wstrb
                    h2f_lw_WLAST            => hps_0_h2f_lw_axi_master_wlast,   --
.wlast
                    h2f_lw_WVALID           => hps_0_h2f_lw_axi_master_wvalid,  --
.wvalid
                    h2f_lw_WREADY           => hps_0_h2f_lw_axi_master_wready,  --
.wready
                    h2f_lw_BID              => hps_0_h2f_lw_axi_master_bid,     --
.bid
                    h2f_lw_BRESP            => hps_0_h2f_lw_axi_master_bresp,   --
.bresp
                    h2f_lw_BVALID           => hps_0_h2f_lw_axi_master_bvalid,  --
.bvalid
                    h2f_lw_BREADY           => hps_0_h2f_lw_axi_master_bready,  --
.bready
                    h2f_lw_ARID             => hps_0_h2f_lw_axi_master_arid,    --
.arid
```

```
                        h2f_lw_ARADDR            => hps_0_h2f_lw_axi_master_araddr,  --
.araddr
                        h2f_lw_ARLEN             => hps_0_h2f_lw_axi_master_arlen,   --
.arlen
                        h2f_lw_ARSIZE            => hps_0_h2f_lw_axi_master_arsize,  --
.arsize
                        h2f_lw_ARBURST           => hps_0_h2f_lw_axi_master_arburst, --
.arburst
                        h2f_lw_ARLOCK            => hps_0_h2f_lw_axi_master_arlock,  --
.arlock
                        h2f_lw_ARCACHE           => hps_0_h2f_lw_axi_master_arcache, --
.arcache
                        h2f_lw_ARPROT            => hps_0_h2f_lw_axi_master_arprot,  --
.arprot
                        h2f_lw_ARVALID           => hps_0_h2f_lw_axi_master_arvalid, --
.arvalid
                        h2f_lw_ARREADY           => hps_0_h2f_lw_axi_master_arready, --
.arready
                        h2f_lw_RID               => hps_0_h2f_lw_axi_master_rid,     --
.rid
                        h2f_lw_RDATA             => hps_0_h2f_lw_axi_master_rdata,   --
.rdata
                        h2f_lw_RRESP             => hps_0_h2f_lw_axi_master_rresp,   --
.rresp
                        h2f_lw_RLAST             => hps_0_h2f_lw_axi_master_rlast,   --
.rlast
                        h2f_lw_RVALID            => hps_0_h2f_lw_axi_master_rvalid,  --
.rvalid
                        h2f_lw_RREADY            => hps_0_h2f_lw_axi_master_rready   --
.rready
                );

        mult_data_0 : component mult_data
                port map (
                        avs_s0_address   => mm_interconnect_0_mult_data_0_s0_address,   --
s0.address
                        avs_s0_read      => mm_interconnect_0_mult_data_0_s0_read,      --
.read
                        avs_s0_write     => mm_interconnect_0_mult_data_0_s0_write,     --
.write
                        avs_s0_readdata  => mm_interconnect_0_mult_data_0_s0_readdata,  --
.readdata
                        avs_s0_writedata => mm_interconnect_0_mult_data_0_s0_writedata, --
.writedata
                        clk              => clk_clk,                                    --
clock.clk
                        reset            => rst_controller_reset_out_reset,             --
reset.reset
                        mult_in1         => mult_data_0_mult_data_m_in1,                --
mult_data.m_in1
                        mult_in2         => mult_data_0_mult_data_m_in2,                --
.m_in2
                        mult_result      => mult_data_0_mult_data_m_result              --
.m_result
                );

        mult_control_0 : component mult_control
                port map (
                        avs_s0_address   => mm_interconnect_0_mult_control_0_s0_address,   --
s0.address
                        avs_s0_write     => mm_interconnect_0_mult_control_0_s0_write,     --
.write
                        avs_s0_writedata => mm_interconnect_0_mult_control_0_s0_writedata, --
.writedata
                        avs_s0_read      => mm_interconnect_0_mult_control_0_s0_read,      --
.read
                        avs_s0_readdata  => mm_interconnect_0_mult_control_0_s0_readdata,  --
.readdata
                        clk              => clk_clk,                                       --
clock.clk
```

```
                        reset            => rst_controller_reset_out_reset,               --
reset.reset
                        mult_start       => mult_control_0_mult_control_m_start,          --
mult_control.m_start
                        mult_reset       => mult_control_0_mult_control_m_reset,          --
.m_reset
                        mult_done        => mult_control_0_mult_control_m_done            --
.m_done
                );

        div_control_0 : component div_control
                port map (
                        avs_s0_address   => mm_interconnect_0_div_control_0_s0_address,   --
s0.address
                        avs_s0_write     => mm_interconnect_0_div_control_0_s0_write,     --
.write
                        avs_s0_writedata => mm_interconnect_0_div_control_0_s0_writedata, --
.writedata
                        avs_s0_read      => mm_interconnect_0_div_control_0_s0_read,      --
.read
                        avs_s0_readdata  => mm_interconnect_0_div_control_0_s0_readdata,  --
.readdata
                        clk              => clk_clk,                                      --
clock.clk
                        reset            => rst_controller_reset_out_reset,               --
reset.reset
                        div_start        => div_control_0_div_control_d_start,            --
div_control.d_start
                        div_reset        => div_control_0_div_control_d_reset,            --
.d_reset
                        div_done         => div_control_0_div_control_d_done              --
.d_done
                );

        div_data_0 : component div_data
                port map (
                        avs_s0_address   => mm_interconnect_0_div_data_0_s0_address,   --
s0.address
                        avs_s0_read      => mm_interconnect_0_div_data_0_s0_read,      --
.read
                        avs_s0_write     => mm_interconnect_0_div_data_0_s0_write,     --
.write
                        avs_s0_readdata  => mm_interconnect_0_div_data_0_s0_readdata,  --
.readdata
                        avs_s0_writedata => mm_interconnect_0_div_data_0_s0_writedata, --
.writedata
                        clk              => clk_clk,                                   --
clock.clk
                        reset            => rst_controller_reset_out_reset,            --
reset.reset
                        div_denom        => div_data_0_div_data_d_denom,               --
div_data.d_denom
                        div_numer        => div_data_0_div_data_d_numer,               --
.d_numer
                        div_quot         => div_data_0_div_data_d_quot,                --
.d_quot
                        div_rem          => div_data_0_div_data_d_rem                  --
.d_rem
                );

        mm_interconnect_0 : component soc_system_mm_interconnect_0
                port map (
                        hps_0_h2f_lw_axi_master_awid                                    =>
hps_0_h2f_lw_axi_master_awid,                   --
hps_0_h2f_lw_axi_master.awid
                        hps_0_h2f_lw_axi_master_awaddr                                  =>
hps_0_h2f_lw_axi_master_awaddr,                 --
.awaddr
                        hps_0_h2f_lw_axi_master_awlen                                   =>
hps_0_h2f_lw_axi_master_awlen,                  --
.awlen
```

```
                    hps_0_h2f_lw_axi_master_awsize                    =>
hps_0_h2f_lw_axi_master_awsize,                --
.awsize
                    hps_0_h2f_lw_axi_master_awburst                   =>
hps_0_h2f_lw_axi_master_awburst,               --
.awburst
                    hps_0_h2f_lw_axi_master_awlock                    =>
hps_0_h2f_lw_axi_master_awlock,                --
.awlock
                    hps_0_h2f_lw_axi_master_awcache                   =>
hps_0_h2f_lw_axi_master_awcache,               --
.awcache
                    hps_0_h2f_lw_axi_master_awprot                    =>
hps_0_h2f_lw_axi_master_awprot,                --
.awprot
                    hps_0_h2f_lw_axi_master_awvalid                   =>
hps_0_h2f_lw_axi_master_awvalid,               --
.awvalid
                    hps_0_h2f_lw_axi_master_awready                   =>
hps_0_h2f_lw_axi_master_awready,               --
.awready
                    hps_0_h2f_lw_axi_master_wid                       =>
hps_0_h2f_lw_axi_master_wid,                   --
.wid
                    hps_0_h2f_lw_axi_master_wdata                     =>
hps_0_h2f_lw_axi_master_wdata,                 --
.wdata
                    hps_0_h2f_lw_axi_master_wstrb                     =>
hps_0_h2f_lw_axi_master_wstrb,                 --
.wstrb
                    hps_0_h2f_lw_axi_master_wlast                     =>
hps_0_h2f_lw_axi_master_wlast,                 --
.wlast
                    hps_0_h2f_lw_axi_master_wvalid                    =>
hps_0_h2f_lw_axi_master_wvalid,                --
.wvalid
                    hps_0_h2f_lw_axi_master_wready                    =>
hps_0_h2f_lw_axi_master_wready,                --
.wready
                    hps_0_h2f_lw_axi_master_bid                       =>
hps_0_h2f_lw_axi_master_bid,                   --
.bid
                    hps_0_h2f_lw_axi_master_bresp                     =>
hps_0_h2f_lw_axi_master_bresp,                 --
.bresp
                    hps_0_h2f_lw_axi_master_bvalid                    =>
hps_0_h2f_lw_axi_master_bvalid,                --
.bvalid
                    hps_0_h2f_lw_axi_master_bready                    =>
hps_0_h2f_lw_axi_master_bready,                --
.bready
                    hps_0_h2f_lw_axi_master_arid                      =>
hps_0_h2f_lw_axi_master_arid,                  --
.arid
                    hps_0_h2f_lw_axi_master_araddr                    =>
hps_0_h2f_lw_axi_master_araddr,                --
.araddr
                    hps_0_h2f_lw_axi_master_arlen                     =>
hps_0_h2f_lw_axi_master_arlen,                 --
.arlen
                    hps_0_h2f_lw_axi_master_arsize                    =>
hps_0_h2f_lw_axi_master_arsize,                --
.arsize
                    hps_0_h2f_lw_axi_master_arburst                   =>
hps_0_h2f_lw_axi_master_arburst,               --
.arburst
                    hps_0_h2f_lw_axi_master_arlock                    =>
hps_0_h2f_lw_axi_master_arlock,                --
.arlock
```

```
                      hps_0_h2f_lw_axi_master_arcache                                    =>
hps_0_h2f_lw_axi_master_arcache,               --
.arcache
                      hps_0_h2f_lw_axi_master_arprot                                     =>
hps_0_h2f_lw_axi_master_arprot,                --
.arprot
                      hps_0_h2f_lw_axi_master_arvalid                                    =>
hps_0_h2f_lw_axi_master_arvalid,               --
.arvalid
                      hps_0_h2f_lw_axi_master_arready                                    =>
hps_0_h2f_lw_axi_master_arready,               --
.arready
                      hps_0_h2f_lw_axi_master_rid                                        =>
hps_0_h2f_lw_axi_master_rid,                   --
.rid
                      hps_0_h2f_lw_axi_master_rdata                                      =>
hps_0_h2f_lw_axi_master_rdata,                 --
.rdata
                      hps_0_h2f_lw_axi_master_rresp                                      =>
hps_0_h2f_lw_axi_master_rresp,                 --
.rresp
                      hps_0_h2f_lw_axi_master_rlast                                      =>
hps_0_h2f_lw_axi_master_rlast,                 --
.rlast
                      hps_0_h2f_lw_axi_master_rvalid                                     =>
hps_0_h2f_lw_axi_master_rvalid,                --
.rvalid
                      hps_0_h2f_lw_axi_master_rready                                     =>
hps_0_h2f_lw_axi_master_rready,                --
.rready
                      clk_0_clk_clk                                                      =>
clk_clk,                                       --
clk_0_clk.clk
                      hps_0_h2f_lw_axi_master_agent_clk_reset_reset_bridge_in_reset_reset =>
rst_controller_001_reset_out_reset,            --
hps_0_h2f_lw_axi_master_agent_clk_reset_reset_bridge_in_reset.reset
                      mult_data_0_reset_reset_bridge_in_reset_reset                      =>
rst_controller_reset_out_reset,                --
mult_data_0_reset_reset_bridge_in_reset.reset
                      div_control_0_s0_address                                           =>
mm_interconnect_0_div_control_0_s0_address,    --
div_control_0_s0.address
                      div_control_0_s0_write                                             =>
mm_interconnect_0_div_control_0_s0_write,      --
.write
                      div_control_0_s0_read                                              =>
mm_interconnect_0_div_control_0_s0_read,       --
.read
                      div_control_0_s0_readdata                                          =>
mm_interconnect_0_div_control_0_s0_readdata,   --
.readdata
                      div_control_0_s0_writedata                                         =>
mm_interconnect_0_div_control_0_s0_writedata,  --
.writedata
                      div_data_0_s0_address                                              =>
mm_interconnect_0_div_data_0_s0_address,       --
div_data_0_s0.address
                      div_data_0_s0_write                                                =>
mm_interconnect_0_div_data_0_s0_write,         --
.write
                      div_data_0_s0_read                                                 =>
mm_interconnect_0_div_data_0_s0_read,          --
.read
                      div_data_0_s0_readdata                                             =>
mm_interconnect_0_div_data_0_s0_readdata,      --
.readdata
                      div_data_0_s0_writedata                                            =>
mm_interconnect_0_div_data_0_s0_writedata,     --
.writedata
```

```
                        mult_control_0_s0_address                                => mm_interconnect_0_mult_control_0_s0_address,   -- mult_control_0_s0.address
                        mult_control_0_s0_write                                  => mm_interconnect_0_mult_control_0_s0_write,     -- .write
                        mult_control_0_s0_read                                   => mm_interconnect_0_mult_control_0_s0_read,      -- .read
                        mult_control_0_s0_readdata                               => mm_interconnect_0_mult_control_0_s0_readdata,  -- .readdata
                        mult_control_0_s0_writedata                              => mm_interconnect_0_mult_control_0_s0_writedata, -- .writedata
                        mult_data_0_s0_address                                   => mm_interconnect_0_mult_data_0_s0_address,      -- mult_data_0_s0.address
                        mult_data_0_s0_write                                     => mm_interconnect_0_mult_data_0_s0_write,        -- .write
                        mult_data_0_s0_read                                      => mm_interconnect_0_mult_data_0_s0_read,         -- .read
                        mult_data_0_s0_readdata                                  => mm_interconnect_0_mult_data_0_s0_readdata,     -- .readdata
                        mult_data_0_s0_writedata                                 => mm_interconnect_0_mult_data_0_s0_writedata     -- .writedata
                );

        rst_controller : component altera_reset_controller
                generic map (
                        NUM_RESET_INPUTS          => 1,
                        OUTPUT_RESET_SYNC_EDGES   => "deassert",
                        SYNC_DEPTH                => 2,
                        RESET_REQUEST_PRESENT     => 0,
                        RESET_REQ_WAIT_TIME       => 1,
                        MIN_RST_ASSERTION_TIME    => 3,
                        RESET_REQ_EARLY_DSRT_TIME => 1,
                        USE_RESET_REQUEST_IN0     => 0,
                        USE_RESET_REQUEST_IN1     => 0,
                        USE_RESET_REQUEST_IN2     => 0,
                        USE_RESET_REQUEST_IN3     => 0,
                        USE_RESET_REQUEST_IN4     => 0,
                        USE_RESET_REQUEST_IN5     => 0,
                        USE_RESET_REQUEST_IN6     => 0,
                        USE_RESET_REQUEST_IN7     => 0,
                        USE_RESET_REQUEST_IN8     => 0,
                        USE_RESET_REQUEST_IN9     => 0,
                        USE_RESET_REQUEST_IN10    => 0,
                        USE_RESET_REQUEST_IN11    => 0,
                        USE_RESET_REQUEST_IN12    => 0,
                        USE_RESET_REQUEST_IN13    => 0,
                        USE_RESET_REQUEST_IN14    => 0,
                        USE_RESET_REQUEST_IN15    => 0,
                        ADAPT_RESET_REQUEST       => 0
                )
                port map (
                        reset_in0      => reset_reset_n_ports_inv,        -- reset_in0.reset
                        clk            => clk_clk,                        --       clk.clk
                        reset_out      => rst_controller_reset_out_reset, -- reset_out.reset
                        reset_req      => open,                           -- (terminated)
                        reset_req_in0  => '0',                            -- (terminated)
                        reset_in1      => '0',                            -- (terminated)
                        reset_req_in1  => '0',                            -- (terminated)
                        reset_in2      => '0',                            -- (terminated)
                        reset_req_in2  => '0',                            -- (terminated)
                        reset_in3      => '0',                            -- (terminated)
                        reset_req_in3  => '0',                            -- (terminated)
```

```
                reset_in4      => '0',                              -- (terminated)
                reset_req_in4  => '0',                              -- (terminated)
                reset_in5      => '0',                              -- (terminated)
                reset_req_in5  => '0',                              -- (terminated)
                reset_in6      => '0',                              -- (terminated)
                reset_req_in6  => '0',                              -- (terminated)
                reset_in7      => '0',                              -- (terminated)
                reset_req_in7  => '0',                              -- (terminated)
                reset_in8      => '0',                              -- (terminated)
                reset_req_in8  => '0',                              -- (terminated)
                reset_in9      => '0',                              -- (terminated)
                reset_req_in9  => '0',                              -- (terminated)
                reset_in10     => '0',                              -- (terminated)
                reset_req_in10 => '0',                              -- (terminated)
                reset_in11     => '0',                              -- (terminated)
                reset_req_in11 => '0',                              -- (terminated)
                reset_in12     => '0',                              -- (terminated)
                reset_req_in12 => '0',                              -- (terminated)
                reset_in13     => '0',                              -- (terminated)
                reset_req_in13 => '0',                              -- (terminated)
                reset_in14     => '0',                              -- (terminated)
                reset_req_in14 => '0',                              -- (terminated)
                reset_in15     => '0',                              -- (terminated)
                reset_req_in15 => '0'                               -- (terminated)
        );

rst_controller_001 : component altera_reset_controller
        generic map (
                NUM_RESET_INPUTS          => 1,
                OUTPUT_RESET_SYNC_EDGES   => "deassert",
                SYNC_DEPTH                => 2,
                RESET_REQUEST_PRESENT     => 0,
                RESET_REQ_WAIT_TIME       => 1,
                MIN_RST_ASSERTION_TIME    => 3,
                RESET_REQ_EARLY_DSRT_TIME => 1,
                USE_RESET_REQUEST_IN0     => 0,
                USE_RESET_REQUEST_IN1     => 0,
                USE_RESET_REQUEST_IN2     => 0,
                USE_RESET_REQUEST_IN3     => 0,
                USE_RESET_REQUEST_IN4     => 0,
                USE_RESET_REQUEST_IN5     => 0,
                USE_RESET_REQUEST_IN6     => 0,
                USE_RESET_REQUEST_IN7     => 0,
                USE_RESET_REQUEST_IN8     => 0,
                USE_RESET_REQUEST_IN9     => 0,
                USE_RESET_REQUEST_IN10    => 0,
                USE_RESET_REQUEST_IN11    => 0,
                USE_RESET_REQUEST_IN12    => 0,
                USE_RESET_REQUEST_IN13    => 0,
                USE_RESET_REQUEST_IN14    => 0,
                USE_RESET_REQUEST_IN15    => 0,
                ADAPT_RESET_REQUEST       => 0
        )
        port map (
                reset_in0      => hps_0_h2f_reset_reset_n_ports_inv,  -- reset_in0.reset
                clk            => clk_clk,                            --       clk.clk
                reset_out      => rst_controller_001_reset_out_reset, -- reset_out.reset
                reset_req      => open,                               -- (terminated)
                reset_req_in0  => '0',                                -- (terminated)
                reset_in1      => '0',                                -- (terminated)
                reset_req_in1  => '0',                                -- (terminated)
                reset_in2      => '0',                                -- (terminated)
                reset_req_in2  => '0',                                -- (terminated)
                reset_in3      => '0',                                -- (terminated)
                reset_req_in3  => '0',                                -- (terminated)
                reset_in4      => '0',                                -- (terminated)
                reset_req_in4  => '0',                                -- (terminated)
                reset_in5      => '0',                                -- (terminated)
                reset_req_in5  => '0',                                -- (terminated)
                reset_in6      => '0',                                -- (terminated)
                reset_req_in6  => '0',                                -- (terminated)
```

```
                        reset_in7      => '0',                                  -- (terminated)
                        reset_req_in7  => '0',                                  -- (terminated)
                        reset_in8      => '0',                                  -- (terminated)
                        reset_req_in8  => '0',                                  -- (terminated)
                        reset_in9      => '0',                                  -- (terminated)
                        reset_req_in9  => '0',                                  -- (terminated)
                        reset_in10     => '0',                                  -- (terminated)
                        reset_req_in10 => '0',                                  -- (terminated)
                        reset_in11     => '0',                                  -- (terminated)
                        reset_req_in11 => '0',                                  -- (terminated)
                        reset_in12     => '0',                                  -- (terminated)
                        reset_req_in12 => '0',                                  -- (terminated)
                        reset_in13     => '0',                                  -- (terminated)
                        reset_req_in13 => '0',                                  -- (terminated)
                        reset_in14     => '0',                                  -- (terminated)
                        reset_req_in14 => '0',                                  -- (terminated)
                        reset_in15     => '0',                                  -- (terminated)
                        reset_req_in15 => '0'                                   -- (terminated)
                );

        hps_0_h2f_reset_reset_n_ports_inv <= not hps_0_h2f_reset_reset;

        reset_reset_n_ports_inv <= not reset_reset_n;

        hps_0_h2f_reset_reset_n <= hps_0_h2f_reset_reset;

end architecture rtl; -- of soc_system
```

div.qip

```
set_global_assignment -name IP_TOOL_NAME "LPM_DIVIDE"
set_global_assignment -name IP_TOOL_VERSION "14.0"
set_global_assignment -name IP_GENERATED_DEVICE_FAMILY "{Cyclone V}"
set_global_assignment -name VHDL_FILE [file join $::quartus(qip_path) "div.vhd"]
set_global_assignment -name MISC_FILE [file join $::quartus(qip_path) "div.cmp"]
```

div.vhd

```
-- megafunction wizard: %LPM_DIVIDE%
-- GENERATION: STANDARD
-- VERSION: WM1.0
-- MODULE: LPM_DIVIDE

-- ============================================================
-- File Name: div.vhd
-- Megafunction Name(s):
--                     LPM_DIVIDE
--
-- Simulation Library Files(s):
--                     lpm
-- ============================================================
-- ************************************************************
-- THIS IS A WIZARD-GENERATED FILE. DO NOT EDIT THIS FILE!
--
-- 14.0.2 Build 209 09/17/2014 SJ Full Version
-- ************************************************************


--Copyright (C) 1991-2014 Altera Corporation. All rights reserved.
--Your use of Altera Corporation's design tools, logic functions
--and other software and tools, and its AMPP partner logic
--functions, and any output files from any of the foregoing
--(including device programming or simulation files), and any
--associated documentation or information are expressly subject
--to the terms and conditions of the Altera Program License
--Subscription Agreement, the Altera Quartus II License Agreement,
--the Altera MegaCore Function License Agreement, or other
--applicable license agreement, including, without limitation,
--that your use is for the sole purpose of programming logic
--devices manufactured by Altera and sold by Altera or its
--authorized distributors.  Please refer to the applicable
--agreement for further details.


LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY lpm;
USE lpm.all;

ENTITY div IS
	PORT
	(
		aclr		: IN STD_LOGIC ;
		clken		: IN STD_LOGIC ;
		clock		: IN STD_LOGIC ;
		denom		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		numer		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		quotient		: OUT STD_LOGIC_VECTOR (15 DOWNTO 0);
		remain		: OUT STD_LOGIC_VECTOR (15 DOWNTO 0)
	);
END div;


ARCHITECTURE SYN OF div IS

	SIGNAL sub_wire0	: STD_LOGIC_VECTOR (15 DOWNTO 0);
	SIGNAL sub_wire1	: STD_LOGIC_VECTOR (15 DOWNTO 0);
```

```
        COMPONENT lpm_divide
        GENERIC (
                lpm_drepresentation           : STRING;
                lpm_hint              : STRING;
                lpm_nrepresentation           : STRING;
                lpm_pipeline          : NATURAL;
                lpm_type              : STRING;
                lpm_widthd            : NATURAL;
                lpm_widthn            : NATURAL
        );
        PORT (
                        aclr    : IN STD_LOGIC ;
                        clken   : IN STD_LOGIC ;
                        clock   : IN STD_LOGIC ;
                        denom   : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
                        numer   : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
                        quotient       : OUT STD_LOGIC_VECTOR (15 DOWNTO 0);
                        remain  : OUT STD_LOGIC_VECTOR (15 DOWNTO 0)
        );
        END COMPONENT;

BEGIN
        quotient    <= sub_wire0(15 DOWNTO 0);
        remain    <= sub_wire1(15 DOWNTO 0);

        LPM_DIVIDE_component : LPM_DIVIDE
        GENERIC MAP (
                lpm_drepresentation => "UNSIGNED",
                lpm_hint => "LPM_REMAINDERPOSITIVE=TRUE",
                lpm_nrepresentation => "UNSIGNED",
                lpm_pipeline => 4,
                lpm_type => "LPM_DIVIDE",
                lpm_widthd => 16,
                lpm_widthn => 16
        )
        PORT MAP (
                aclr => aclr,
                clken => clken,
                clock => clock,
                denom => denom,
                numer => numer,
                quotient => sub_wire0,
                remain => sub_wire1
        );



END SYN;

-- ============================================================
-- CNX file retrieval info
-- ============================================================
-- Retrieval info: PRIVATE: INTENDED_DEVICE_FAMILY STRING "Cyclone V"
-- Retrieval info: PRIVATE: PRIVATE_LPM_REMAINDERPOSITIVE STRING "TRUE"
-- Retrieval info: PRIVATE: PRIVATE_MAXIMIZE_SPEED NUMERIC "-1"
-- Retrieval info: PRIVATE: SYNTH_WRAPPER_GEN_POSTFIX STRING "0"
-- Retrieval info: PRIVATE: USING_PIPELINE NUMERIC "1"
-- Retrieval info: PRIVATE: VERSION_NUMBER NUMERIC "2"
-- Retrieval info: PRIVATE: new_diagram STRING "1"
-- Retrieval info: LIBRARY: lpm lpm.lpm_components.all
-- Retrieval info: CONSTANT: LPM_DREPRESENTATION STRING "UNSIGNED"
-- Retrieval info: CONSTANT: LPM_HINT STRING "LPM_REMAINDERPOSITIVE=TRUE"
-- Retrieval info: CONSTANT: LPM_NREPRESENTATION STRING "UNSIGNED"
-- Retrieval info: CONSTANT: LPM_PIPELINE NUMERIC "4"
-- Retrieval info: CONSTANT: LPM_TYPE STRING "LPM_DIVIDE"
-- Retrieval info: CONSTANT: LPM_WIDTHD NUMERIC "16"
-- Retrieval info: CONSTANT: LPM_WIDTHN NUMERIC "16"
-- Retrieval info: USED_PORT: aclr 0 0 0 0 INPUT NODEFVAL "aclr"
```

```
-- Retrieval info: USED_PORT: clken 0 0 0 0 INPUT NODEFVAL "clken"
-- Retrieval info: USED_PORT: clock 0 0 0 0 INPUT NODEFVAL "clock"
-- Retrieval info: USED_PORT: denom 0 0 16 0 INPUT NODEFVAL "denom[15..0]"
-- Retrieval info: USED_PORT: numer 0 0 16 0 INPUT NODEFVAL "numer[15..0]"
-- Retrieval info: USED_PORT: quotient 0 0 16 0 OUTPUT NODEFVAL "quotient[15..0]"
-- Retrieval info: USED_PORT: remain 0 0 16 0 OUTPUT NODEFVAL "remain[15..0]"
-- Retrieval info: CONNECT: @aclr 0 0 0 0 aclr 0 0 0 0
-- Retrieval info: CONNECT: @clken 0 0 0 0 clken 0 0 0 0
-- Retrieval info: CONNECT: @clock 0 0 0 0 clock 0 0 0 0
-- Retrieval info: CONNECT: @denom 0 0 16 0 denom 0 0 16 0
-- Retrieval info: CONNECT: @numer 0 0 16 0 numer 0 0 16 0
-- Retrieval info: CONNECT: quotient 0 0 16 0 @quotient 0 0 16 0
-- Retrieval info: CONNECT: remain 0 0 16 0 @remain 0 0 16 0
-- Retrieval info: GEN_FILE: TYPE_NORMAL div.vhd TRUE
-- Retrieval info: GEN_FILE: TYPE_NORMAL div.inc FALSE
-- Retrieval info: GEN_FILE: TYPE_NORMAL div.cmp TRUE
-- Retrieval info: GEN_FILE: TYPE_NORMAL div.bsf FALSE
-- Retrieval info: GEN_FILE: TYPE_NORMAL div_inst.vhd FALSE
-- Retrieval info: LIB_FILE: lpm
```

div_unit.vhd

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

ENTITY div_unit IS
	PORT(
		clk, reset, enable	: IN STD_LOGIC;
		div_denom, div_numer	: IN STD_LOGIC_VECTOR(15 DOWNTO 0);
		div_done                                        : OUT STD_LOGIC;
		div_quot, div_rem             : OUT STD_LOGIC_VECTOR(15 DOWNTO 0)
	);
END ENTITY div_unit;

ARCHITECTURE Behaviour OF div_unit IS
	COMPONENT div
		PORT(
			aclr		: IN STD_LOGIC ;
			clken		: IN STD_LOGIC ;
			clock		: IN STD_LOGIC ;
			denom		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
			numer		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
			quotient		: OUT STD_LOGIC_VECTOR (15 DOWNTO 0);
			remain		: OUT STD_LOGIC_VECTOR (15 DOWNTO 0)
		);
	END COMPONENT;

	SIGNAL done0, done1, done2, done3 : STD_LOGIC;
	SIGNAL temp_q, temp_r		: STD_LOGIC_VECTOR(15 DOWNTO 0);

BEGIN
	divider : div
	PORT MAP(
		aclr => reset,
		clken => enable,
		clock => clk,
		denom => div_denom,
		numer => div_numer,
		quotient => temp_q,
		remain => temp_r
	);

	PROCESS(clk, reset, enable)
	BEGIN
		IF(reset = '1')THEN
			done0 <= '0';
			done1 <= '0';
			done2 <= '0';
			done3 <= '0';

		ELSIF(rising_edge(clk))THEN
			IF(enable = '1')THEN
				done0 <= '1';
			END IF;

			--pipeline cycle counting
			done1 <= done0;
			done2 <= done1;
			done3 <= done2;

		END IF;
	END PROCESS;

	div_quot <= temp_q;
	div_rem <= temp_r;
	div_done <= done3;

END Behaviour;
```

div_control.vhd

```
-- div_control.vhd

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity div_control is
	port (
		avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => '0'); --
s0.address
		avs_s0_write     : in  std_logic                     := '0';             --
.write
		avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => '0'); --
.writedata
		avs_s0_read      : in  std_logic                     := '0';             --
.read
		avs_s0_readdata  : out std_logic_vector(31 downto 0);                    --
.readdata
		clk              : in  std_logic                     := '0';             --
clock.clk
		reset            : in  std_logic                     := '0';             --
reset.reset
		div_start       : out std_logic_vector(31 downto 0);                    --
.m_start
		div_reset       : out std_logic_vector(31 downto 0);                    --
.m_reset
		div_done        : in  std_logic_vector(31 downto 0) := (others => '0')  --
.m_done
	);
end entity div_control;

architecture rtl of div_control is

	SIGNAL start, reset2 : STD_logic_vector(31 DOWNTO 0);

begin

		PROCESS(clk, reset, avs_s0_address, avs_s0_write, avs_s0_writedata, avs_s0_read)
	BEGIN
		IF(reset = '1')THEN
			start <= (OTHERS => '0');
			reset2 <= (OTHERS => '0');
		ELSIF(rising_edge(clk))THEN

			IF(avs_s0_write = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						start <= avs_s0_writedata;
					WHEN "0001" =>
						reset2 <= avs_s0_writedata;
					WHEN OTHERS =>

				END CASE;
			ELSIF(avs_s0_read = '1')THEN
				reset2 <= (OTHERS => '0');
				CASE avs_s0_address IS
					WHEN "0000" =>
						avs_s0_readdata <= start;
					WHEN "0001" =>
						avs_s0_readdata <= reset2;
					WHEN "0010" =>
						avs_s0_readdata <= div_done;
					WHEN OTHERS =>

				END CASE;
			END IF;
		END IF;
	END PROCESS;
```

```
        div_start <= start;
        div_reset <= reset2;

end architecture rtl;
```

div_data.vhd

```
-- div_data.vhd
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity div_data is
	port (
		avs_s0_address  : in  std_logic_vector(3 downto 0)  := (others => '0'); --
s0.address
		avs_s0_read     : in  std_logic                     := '0';             --
.read
		avs_s0_write    : in  std_logic                     := '0';             --
.write
		avs_s0_readdata : out std_logic_vector(31 downto 0);                    --
.readdata
		avs_s0_writedata  : in  std_logic_vector(31 downto 0)  := (others => '0'); --
s0.address
		clk             : in  std_logic                     := '0';             --
clock.clk
		reset           : in  std_logic                     := '0';             --
reset.reset
		div_denom       : out std_logic_vector(31 downto 0);                    --
mult_input.m_in1
		div_numer       : out std_logic_vector(31 downto 0);                    --
mult_input.m_in2
		div_quot        : in  std_logic_vector(31 downto 0) := (others => '0'); --
mult_output.m_result
		div_rem         : in  std_logic_vector(31 downto 0) := (others => '0') --
mult_output.m_result
	);
end entity div_data;

architecture rtl of div_data is

	SIGNAL in1, in2 : STD_logic_vector(31 DOWNTO 0);

begin
	PROCESS(clk, reset, avs_s0_read, avs_s0_write, avs_s0_address, avs_s0_writedata)
	BEGIN
		IF(reset = '1')THEN
			avs_s0_readdata <= (OTHERS => '0');
			in1 <= (OTHERS => '0');
			in2 <= (OTHERS => '0');

		ELSIF(rising_edge(clk))THEN
			IF(avs_s0_read = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						avs_s0_readdata <= div_quot;
					WHEN "0001" =>
						avs_s0_readdata <= div_rem;
					WHEN "0010" =>
						avs_s0_readdata <= in1;
					WHEN "0011" =>
						avs_s0_readdata <= in2;
					WHEN OTHERS =>
						avs_s0_readdata <= (OTHERS => '0');
				END CASE;
			ELSIF(avs_s0_write = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						in1 <= "0000000000000000" & avs_s0_writedata(15
DOWNTO 0);
					WHEN "0001" =>
						in2 <= "0000000000000000" & avs_s0_writedata(15
DOWNTO 0);
					WHEN OTHERS =>

					END CASE;
```

```
                    END iF;
                END IF;
        END PROCESS;

        div_denom <= in1;
        div_numer <= in2;

end architecture rtl; -- of mult_output
```

Appendix C: Supporting Files (given by lab)

pin_assignment_DE1-SoC.tcl

```
set_global_assignment -name FAMILY "Cyclone V"
set_global_assignment -name DEVICE 5CSEMA5F31C6
set_global_assignment -name DEVICE_FILTER_PACKAGE FBGA

#============================================================
# I/O Standard
#============================================================

# ADC
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to ADC_CS_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to ADC_DIN
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to ADC_DOUT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to ADC_SCLK

# AUD
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_ADCDAT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_ADCLRCK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_BCLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_DACDAT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_DACLRCK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to AUD_XCK

# CLOCK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to CLOCK_50

# CLOCK2
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to CLOCK2_50

# CLOCK3
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to CLOCK3_50

# CLOCK4
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to CLOCK4_50

# DRAM
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[9]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[10]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[11]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_ADDR[12]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_BA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_BA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_CAS_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_CKE
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_CS_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[9]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[10]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[11]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[12]
```

```
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[13]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[14]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_DQ[15]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_LDQM
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_RAS_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_UDQM
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to DRAM_WE_N

# FAN
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to FAN_CTRL

# FPGA
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to FPGA_I2C_SCLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to FPGA_I2C_SDAT

# GPIO
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[9]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[10]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[11]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[12]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[13]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[14]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[15]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[16]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[17]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[18]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[19]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[20]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[21]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[22]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[23]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[24]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[25]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[26]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[27]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[28]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[29]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[30]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[31]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[32]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[33]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[34]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_0[35]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[9]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[10]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[11]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[12]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[13]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[14]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[15]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[16]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[17]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[18]
```

```
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[19]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[20]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[21]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[22]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[23]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[24]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[25]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[26]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[27]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[28]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[29]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[30]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[31]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[32]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[33]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[34]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to GPIO_1[35]

# HEX0
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX0[6]

# HEX1
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX1[6]

# HEX2
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX2[6]

# HEX3
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX3[6]

# HEX4
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX4[6]

# HEX5
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HEX5[6]
```

```
# HPS
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_CONV_USB_N
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[0]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[1]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[2]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[3]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[4]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[5]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[6]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[7]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[8]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[9]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[10]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[11]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[12]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[13]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_A[14]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_BA[0]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_BA[1]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_BA[2]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_CAS_n
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_CKE
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_CK_n
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_CK_p
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_CS_n
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DM[0]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DM[1]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DM[2]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DM[3]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[0]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[1]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[2]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[3]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[4]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[5]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[6]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[7]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[8]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[9]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[10]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[11]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[12]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[13]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[14]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[15]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[16]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[17]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[18]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[19]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[20]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[21]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[22]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[23]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[24]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[25]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[26]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[27]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[28]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[29]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[30]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_DQ[31]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_n[0]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_n[1]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_n[2]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_n[3]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_p[0]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_p[1]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_p[2]
set_instance_assignment -name IO_STANDARD "Differential 1.5-V SSTL Class I" -to HPS_DDR3_DQS_p[3]
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_ODT
```

```
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_RAS_n
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_RESET_n
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_RZQ
set_instance_assignment -name IO_STANDARD "SSTL-15 Class I" -to HPS_DDR3_WE_n
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_GTX_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_INT_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_MDC
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_MDIO
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_RX_DV
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_TX_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_TX_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_TX_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_TX_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_ENET_TX_EN
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_DCLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_FLASH_NCSO
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_GSENSOR_INT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_I2C1_SCLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_I2C1_SDAT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_I2C2_SCLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_I2C2_SDAT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_I2C_CONTROL
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_KEY
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_LED
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_LTC_GPIO
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_CMD
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SD_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SPIM_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SPIM_MISO
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SPIM_MOSI
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_SPIM_SS
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_UART_RX
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_UART_TX
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_CLKOUT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DATA[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_DIR
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_NXT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_USB_STP
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_GPIO[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to HPS_GPIO[1]

# IRDA
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to IRDA_RXD
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to IRDA_TXD

# KEY
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to KEY[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to KEY[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to KEY[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to KEY[3]
```

```
# LEDR
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to LEDR[9]

# PS2
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to PS2_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to PS2_CLK2
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to PS2_DAT
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to PS2_DAT2

# SW
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[8]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to SW[9]

# TD
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_CLK27
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_DATA[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_HS
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_RESET_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to TD_VS


# VGA
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_B[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_BLANK_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_CLK
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[5]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_G[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_HS
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[0]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[1]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[2]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[3]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[4]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[5]
```

```
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[6]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_R[7]
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_SYNC_N
set_instance_assignment -name IO_STANDARD "3.3-V LVTTL" -to VGA_VS

#============================================================================
#====================================== Pins ================================
#============================================================================

set_global_assignment -name CYCLONEII_RESERVE_NCEO_AFTER_CONFIGURATION "USE AS REGULAR IO"
set_location_assignment PIN_AJ4 -to ADC_CS_N
set_location_assignment PIN_AK4 -to ADC_DIN
set_location_assignment PIN_AK3 -to ADC_DOUT
set_location_assignment PIN_AK2 -to ADC_SCLK
set_location_assignment PIN_K7 -to AUD_ADCDAT
set_location_assignment PIN_K8 -to AUD_ADCLRCK
set_location_assignment PIN_H7 -to AUD_BCLK
set_location_assignment PIN_J7 -to AUD_DACDAT
set_location_assignment PIN_H8 -to AUD_DACLRCK
set_location_assignment PIN_G7 -to AUD_XCK
set_location_assignment PIN_AF14 -to CLOCK_50
set_location_assignment PIN_AA16 -to CLOCK2_50
set_location_assignment PIN_Y26 -to CLOCK3_50
set_location_assignment PIN_K14 -to CLOCK4_50
set_location_assignment PIN_AK14 -to DRAM_ADDR[0]
set_location_assignment PIN_AH14 -to DRAM_ADDR[1]
set_location_assignment PIN_AG15 -to DRAM_ADDR[2]
set_location_assignment PIN_AE14 -to DRAM_ADDR[3]
set_location_assignment PIN_AB15 -to DRAM_ADDR[4]
set_location_assignment PIN_AC14 -to DRAM_ADDR[5]
set_location_assignment PIN_AD14 -to DRAM_ADDR[6]
set_location_assignment PIN_AF15 -to DRAM_ADDR[7]
set_location_assignment PIN_AH15 -to DRAM_ADDR[8]
set_location_assignment PIN_AG13 -to DRAM_ADDR[9]
set_location_assignment PIN_AG12 -to DRAM_ADDR[10]
set_location_assignment PIN_AH13 -to DRAM_ADDR[11]
set_location_assignment PIN_AJ14 -to DRAM_ADDR[12]
set_location_assignment PIN_AF13 -to DRAM_BA[0]
set_location_assignment PIN_AJ12 -to DRAM_BA[1]
set_location_assignment PIN_AF11 -to DRAM_CAS_N
set_location_assignment PIN_AK13 -to DRAM_CKE
set_location_assignment PIN_AH12 -to DRAM_CLK
set_location_assignment PIN_AG11 -to DRAM_CS_N
set_location_assignment PIN_AK6 -to DRAM_DQ[0]
set_location_assignment PIN_AJ7 -to DRAM_DQ[1]
set_location_assignment PIN_AK7 -to DRAM_DQ[2]
set_location_assignment PIN_AK8 -to DRAM_DQ[3]
set_location_assignment PIN_AK9 -to DRAM_DQ[4]
set_location_assignment PIN_AG10 -to DRAM_DQ[5]
set_location_assignment PIN_AK11 -to DRAM_DQ[6]
set_location_assignment PIN_AJ11 -to DRAM_DQ[7]
set_location_assignment PIN_AH10 -to DRAM_DQ[8]
set_location_assignment PIN_AJ10 -to DRAM_DQ[9]
set_location_assignment PIN_AJ9 -to DRAM_DQ[10]
set_location_assignment PIN_AH9 -to DRAM_DQ[11]
set_location_assignment PIN_AH8 -to DRAM_DQ[12]
set_location_assignment PIN_AH7 -to DRAM_DQ[13]
set_location_assignment PIN_AJ6 -to DRAM_DQ[14]
set_location_assignment PIN_AJ5 -to DRAM_DQ[15]
set_location_assignment PIN_AB13 -to DRAM_LDQM
set_location_assignment PIN_AE13 -to DRAM_RAS_N
set_location_assignment PIN_AK12 -to DRAM_UDQM
set_location_assignment PIN_AA13 -to DRAM_WE_N
set_location_assignment PIN_AA12 -to FAN_CTRL
set_location_assignment PIN_J12 -to FPGA_I2C_SCLK
set_location_assignment PIN_K12 -to FPGA_I2C_SDAT
set_location_assignment PIN_AC18 -to GPIO_0[0]
set_location_assignment PIN_Y17 -to GPIO_0[1]
set_location_assignment PIN_AD17 -to GPIO_0[2]
set_location_assignment PIN_Y18 -to GPIO_0[3]
set_location_assignment PIN_AK16 -to GPIO_0[4]
```

```
set_location_assignment PIN_AK18 -to GPIO_0[5]
set_location_assignment PIN_AK19 -to GPIO_0[6]
set_location_assignment PIN_AJ19 -to GPIO_0[7]
set_location_assignment PIN_AJ17 -to GPIO_0[8]
set_location_assignment PIN_AJ16 -to GPIO_0[9]
set_location_assignment PIN_AH18 -to GPIO_0[10]
set_location_assignment PIN_AH17 -to GPIO_0[11]
set_location_assignment PIN_AG16 -to GPIO_0[12]
set_location_assignment PIN_AE16 -to GPIO_0[13]
set_location_assignment PIN_AF16 -to GPIO_0[14]
set_location_assignment PIN_AG17 -to GPIO_0[15]
set_location_assignment PIN_AA18 -to GPIO_0[16]
set_location_assignment PIN_AA19 -to GPIO_0[17]
set_location_assignment PIN_AE17 -to GPIO_0[18]
set_location_assignment PIN_AC20 -to GPIO_0[19]
set_location_assignment PIN_AH19 -to GPIO_0[20]
set_location_assignment PIN_AJ20 -to GPIO_0[21]
set_location_assignment PIN_AH20 -to GPIO_0[22]
set_location_assignment PIN_AK21 -to GPIO_0[23]
set_location_assignment PIN_AD19 -to GPIO_0[24]
set_location_assignment PIN_AD20 -to GPIO_0[25]
set_location_assignment PIN_AE18 -to GPIO_0[26]
set_location_assignment PIN_AE19 -to GPIO_0[27]
set_location_assignment PIN_AF20 -to GPIO_0[28]
set_location_assignment PIN_AF21 -to GPIO_0[29]
set_location_assignment PIN_AF19 -to GPIO_0[30]
set_location_assignment PIN_AG21 -to GPIO_0[31]
set_location_assignment PIN_AF18 -to GPIO_0[32]
set_location_assignment PIN_AG20 -to GPIO_0[33]
set_location_assignment PIN_AG18 -to GPIO_0[34]
set_location_assignment PIN_AJ21 -to GPIO_0[35]
set_location_assignment PIN_AB17 -to GPIO_1[0]
set_location_assignment PIN_AA21 -to GPIO_1[1]
set_location_assignment PIN_AB21 -to GPIO_1[2]
set_location_assignment PIN_AC23 -to GPIO_1[3]
set_location_assignment PIN_AD24 -to GPIO_1[4]
set_location_assignment PIN_AE23 -to GPIO_1[5]
set_location_assignment PIN_AE24 -to GPIO_1[6]
set_location_assignment PIN_AF25 -to GPIO_1[7]
set_location_assignment PIN_AF26 -to GPIO_1[8]
set_location_assignment PIN_AG25 -to GPIO_1[9]
set_location_assignment PIN_AG26 -to GPIO_1[10]
set_location_assignment PIN_AH24 -to GPIO_1[11]
set_location_assignment PIN_AH27 -to GPIO_1[12]
set_location_assignment PIN_AJ27 -to GPIO_1[13]
set_location_assignment PIN_AK29 -to GPIO_1[14]
set_location_assignment PIN_AK28 -to GPIO_1[15]
set_location_assignment PIN_AK27 -to GPIO_1[16]
set_location_assignment PIN_AJ26 -to GPIO_1[17]
set_location_assignment PIN_AK26 -to GPIO_1[18]
set_location_assignment PIN_AH25 -to GPIO_1[19]
set_location_assignment PIN_AJ25 -to GPIO_1[20]
set_location_assignment PIN_AJ24 -to GPIO_1[21]
set_location_assignment PIN_AK24 -to GPIO_1[22]
set_location_assignment PIN_AG23 -to GPIO_1[23]
set_location_assignment PIN_AK23 -to GPIO_1[24]
set_location_assignment PIN_AH23 -to GPIO_1[25]
set_location_assignment PIN_AK22 -to GPIO_1[26]
set_location_assignment PIN_AJ22 -to GPIO_1[27]
set_location_assignment PIN_AH22 -to GPIO_1[28]
set_location_assignment PIN_AG22 -to GPIO_1[29]
set_location_assignment PIN_AF24 -to GPIO_1[30]
set_location_assignment PIN_AF23 -to GPIO_1[31]
set_location_assignment PIN_AE22 -to GPIO_1[32]
set_location_assignment PIN_AD21 -to GPIO_1[33]
set_location_assignment PIN_AA20 -to GPIO_1[34]
set_location_assignment PIN_AC22 -to GPIO_1[35]
set_location_assignment PIN_AE26 -to HEX0[0]
set_location_assignment PIN_AE27 -to HEX0[1]
set_location_assignment PIN_AE28 -to HEX0[2]
set_location_assignment PIN_AG27 -to HEX0[3]
```

```
set_location_assignment PIN_AF28 -to HEX0[4]
set_location_assignment PIN_AG28 -to HEX0[5]
set_location_assignment PIN_AH28 -to HEX0[6]
set_location_assignment PIN_AJ29 -to HEX1[0]
set_location_assignment PIN_AH29 -to HEX1[1]
set_location_assignment PIN_AH30 -to HEX1[2]
set_location_assignment PIN_AG30 -to HEX1[3]
set_location_assignment PIN_AF29 -to HEX1[4]
set_location_assignment PIN_AF30 -to HEX1[5]
set_location_assignment PIN_AD27 -to HEX1[6]
set_location_assignment PIN_AB23 -to HEX2[0]
set_location_assignment PIN_AE29 -to HEX2[1]
set_location_assignment PIN_AD29 -to HEX2[2]
set_location_assignment PIN_AC28 -to HEX2[3]
set_location_assignment PIN_AD30 -to HEX2[4]
set_location_assignment PIN_AC29 -to HEX2[5]
set_location_assignment PIN_AC30 -to HEX2[6]
set_location_assignment PIN_AD26 -to HEX3[0]
set_location_assignment PIN_AC27 -to HEX3[1]
set_location_assignment PIN_AD25 -to HEX3[2]
set_location_assignment PIN_AC25 -to HEX3[3]
set_location_assignment PIN_AB28 -to HEX3[4]
set_location_assignment PIN_AB25 -to HEX3[5]
set_location_assignment PIN_AB22 -to HEX3[6]
set_location_assignment PIN_AA24 -to HEX4[0]
set_location_assignment PIN_Y23 -to HEX4[1]
set_location_assignment PIN_Y24 -to HEX4[2]
set_location_assignment PIN_W22 -to HEX4[3]
set_location_assignment PIN_W24 -to HEX4[4]
set_location_assignment PIN_V23 -to HEX4[5]
set_location_assignment PIN_W25 -to HEX4[6]
set_location_assignment PIN_V25 -to HEX5[0]
set_location_assignment PIN_AA28 -to HEX5[1]
set_location_assignment PIN_Y27 -to HEX5[2]
set_location_assignment PIN_AB27 -to HEX5[3]
set_location_assignment PIN_AB26 -to HEX5[4]
set_location_assignment PIN_AA26 -to HEX5[5]
set_location_assignment PIN_AA25 -to HEX5[6]
set_location_assignment PIN_D25 -to HPS_CLOCK1_25
set_location_assignment PIN_F25 -to HPS_CLOCK2_25
set_location_assignment PIN_B15 -to HPS_CONV_USB_N
set_location_assignment PIN_F26 -to HPS_DDR3_A[0]
set_location_assignment PIN_G30 -to HPS_DDR3_A[1]
set_location_assignment PIN_F28 -to HPS_DDR3_A[2]
set_location_assignment PIN_F30 -to HPS_DDR3_A[3]
set_location_assignment PIN_J25 -to HPS_DDR3_A[4]
set_location_assignment PIN_J27 -to HPS_DDR3_A[5]
set_location_assignment PIN_F29 -to HPS_DDR3_A[6]
set_location_assignment PIN_E28 -to HPS_DDR3_A[7]
set_location_assignment PIN_H27 -to HPS_DDR3_A[8]
set_location_assignment PIN_G26 -to HPS_DDR3_A[9]
set_location_assignment PIN_D29 -to HPS_DDR3_A[10]
set_location_assignment PIN_C30 -to HPS_DDR3_A[11]
set_location_assignment PIN_B30 -to HPS_DDR3_A[12]
set_location_assignment PIN_C29 -to HPS_DDR3_A[13]
set_location_assignment PIN_H25 -to HPS_DDR3_A[14]
set_location_assignment PIN_E29 -to HPS_DDR3_BA[0]
set_location_assignment PIN_J24 -to HPS_DDR3_BA[1]
set_location_assignment PIN_J23 -to HPS_DDR3_BA[2]
set_location_assignment PIN_E27 -to HPS_DDR3_CAS_n
set_location_assignment PIN_L29 -to HPS_DDR3_CKE
set_location_assignment PIN_L23 -to HPS_DDR3_CK_n
set_location_assignment PIN_M23 -to HPS_DDR3_CK_p
set_location_assignment PIN_H24 -to HPS_DDR3_CS_n
set_location_assignment PIN_K28 -to HPS_DDR3_DM[0]
set_location_assignment PIN_M28 -to HPS_DDR3_DM[1]
set_location_assignment PIN_R28 -to HPS_DDR3_DM[2]
set_location_assignment PIN_W30 -to HPS_DDR3_DM[3]
set_location_assignment PIN_M19 -to HPS_DDR3_DQS_n[0]
set_location_assignment PIN_N24 -to HPS_DDR3_DQS_n[1]
set_location_assignment PIN_R18 -to HPS_DDR3_DQS_n[2]
```

```
set_location_assignment PIN_R21 -to HPS_DDR3_DQS_n[3]
set_location_assignment PIN_N18 -to HPS_DDR3_DQS_p[0]
set_location_assignment PIN_N25 -to HPS_DDR3_DQS_p[1]
set_location_assignment PIN_R19 -to HPS_DDR3_DQS_p[2]
set_location_assignment PIN_R22 -to HPS_DDR3_DQS_p[3]
set_location_assignment PIN_K23 -to HPS_DDR3_DQ[0]
set_location_assignment PIN_K22 -to HPS_DDR3_DQ[1]
set_location_assignment PIN_H30 -to HPS_DDR3_DQ[2]
set_location_assignment PIN_G28 -to HPS_DDR3_DQ[3]
set_location_assignment PIN_L25 -to HPS_DDR3_DQ[4]
set_location_assignment PIN_L24 -to HPS_DDR3_DQ[5]
set_location_assignment PIN_J30 -to HPS_DDR3_DQ[6]
set_location_assignment PIN_J29 -to HPS_DDR3_DQ[7]
set_location_assignment PIN_K26 -to HPS_DDR3_DQ[8]
set_location_assignment PIN_L26 -to HPS_DDR3_DQ[9]
set_location_assignment PIN_K29 -to HPS_DDR3_DQ[10]
set_location_assignment PIN_K27 -to HPS_DDR3_DQ[11]
set_location_assignment PIN_M26 -to HPS_DDR3_DQ[12]
set_location_assignment PIN_M27 -to HPS_DDR3_DQ[13]
set_location_assignment PIN_L28 -to HPS_DDR3_DQ[14]
set_location_assignment PIN_M30 -to HPS_DDR3_DQ[15]
set_location_assignment PIN_U26 -to HPS_DDR3_DQ[16]
set_location_assignment PIN_T26 -to HPS_DDR3_DQ[17]
set_location_assignment PIN_N29 -to HPS_DDR3_DQ[18]
set_location_assignment PIN_N28 -to HPS_DDR3_DQ[19]
set_location_assignment PIN_P26 -to HPS_DDR3_DQ[20]
set_location_assignment PIN_P27 -to HPS_DDR3_DQ[21]
set_location_assignment PIN_N27 -to HPS_DDR3_DQ[22]
set_location_assignment PIN_R29 -to HPS_DDR3_DQ[23]
set_location_assignment PIN_P24 -to HPS_DDR3_DQ[24]
set_location_assignment PIN_P25 -to HPS_DDR3_DQ[25]
set_location_assignment PIN_T29 -to HPS_DDR3_DQ[26]
set_location_assignment PIN_T28 -to HPS_DDR3_DQ[27]
set_location_assignment PIN_R27 -to HPS_DDR3_DQ[28]
set_location_assignment PIN_R26 -to HPS_DDR3_DQ[29]
set_location_assignment PIN_V30 -to HPS_DDR3_DQ[30]
set_location_assignment PIN_W29 -to HPS_DDR3_DQ[31]
set_location_assignment PIN_H28 -to HPS_DDR3_ODT
set_location_assignment PIN_D30 -to HPS_DDR3_RAS_n
set_location_assignment PIN_P30 -to HPS_DDR3_RESET_n
set_location_assignment PIN_D27 -to HPS_DDR3_RZQ
set_location_assignment PIN_C28 -to HPS_DDR3_WE_n
set_location_assignment PIN_H19 -to HPS_ENET_GTX_CLK
set_location_assignment PIN_C19 -to HPS_ENET_INT_N
set_location_assignment PIN_B21 -to HPS_ENET_MDC
set_location_assignment PIN_E21 -to HPS_ENET_MDIO
set_location_assignment PIN_E18 -to HPS_ENET_RESET_N
set_location_assignment PIN_G20 -to HPS_ENET_RX_CLK
set_location_assignment PIN_A21 -to HPS_ENET_RX_DATA[0]
set_location_assignment PIN_B20 -to HPS_ENET_RX_DATA[1]
set_location_assignment PIN_B18 -to HPS_ENET_RX_DATA[2]
set_location_assignment PIN_D21 -to HPS_ENET_RX_DATA[3]
set_location_assignment PIN_K17 -to HPS_ENET_RX_DV
set_location_assignment PIN_F20 -to HPS_ENET_TX_DATA[0]
set_location_assignment PIN_J19 -to HPS_ENET_TX_DATA[1]
set_location_assignment PIN_F21 -to HPS_ENET_TX_DATA[2]
set_location_assignment PIN_F19 -to HPS_ENET_TX_DATA[3]
set_location_assignment PIN_A20 -to HPS_ENET_TX_EN
set_location_assignment PIN_B22 -to HPS_GSENSOR_INT
set_location_assignment PIN_E23 -to HPS_I2C1_SCLK
set_location_assignment PIN_C24 -to HPS_I2C1_SDAT
set_location_assignment PIN_H23 -to HPS_I2C2_SCLK
set_location_assignment PIN_A25 -to HPS_I2C2_SDAT
set_location_assignment PIN_H17 -to HPS_LTC_GPIO
set_location_assignment PIN_A16 -to HPS_SD_CLK
set_location_assignment PIN_F18 -to HPS_SD_CMD
set_location_assignment PIN_G18 -to HPS_SD_DATA[0]
set_location_assignment PIN_C17 -to HPS_SD_DATA[1]
set_location_assignment PIN_D17 -to HPS_SD_DATA[2]
set_location_assignment PIN_B16 -to HPS_SD_DATA[3]
set_location_assignment PIN_C23 -to HPS_SPIM_CLK
```

```
set_location_assignment PIN_E24 -to HPS_SPIM_MISO
set_location_assignment PIN_D22 -to HPS_SPIM_MOSI
set_location_assignment PIN_D24 -to HPS_SPIM_SS
set_location_assignment PIN_B25 -to HPS_UART_RX
set_location_assignment PIN_C25 -to HPS_UART_TX
set_location_assignment PIN_N16 -to HPS_USB_CLKOUT
set_location_assignment PIN_E16 -to HPS_USB_DATA[0]
set_location_assignment PIN_G16 -to HPS_USB_DATA[1]
set_location_assignment PIN_D16 -to HPS_USB_DATA[2]
set_location_assignment PIN_D14 -to HPS_USB_DATA[3]
set_location_assignment PIN_A15 -to HPS_USB_DATA[4]
set_location_assignment PIN_C14 -to HPS_USB_DATA[5]
set_location_assignment PIN_D15 -to HPS_USB_DATA[6]
set_location_assignment PIN_M17 -to HPS_USB_DATA[7]
set_location_assignment PIN_E14 -to HPS_USB_DIR
set_location_assignment PIN_A14 -to HPS_USB_NXT
set_location_assignment PIN_G17 -to HPS_USB_RESET
set_location_assignment PIN_C15 -to HPS_USB_STP
set_location_assignment PIN_J12 -to I2C_SCLK
set_location_assignment PIN_K12 -to I2C_SDAT
set_location_assignment PIN_AA30 -to IRDA_RXD
set_location_assignment PIN_AB30 -to IRDA_TXD
set_location_assignment PIN_AA14 -to KEY[0]
set_location_assignment PIN_AA15 -to KEY[1]
set_location_assignment PIN_W15 -to KEY[2]
set_location_assignment PIN_Y16 -to KEY[3]
set_location_assignment PIN_V16 -to LEDR[0]
set_location_assignment PIN_W16 -to LEDR[1]
set_location_assignment PIN_V17 -to LEDR[2]
set_location_assignment PIN_V18 -to LEDR[3]
set_location_assignment PIN_W17 -to LEDR[4]
set_location_assignment PIN_W19 -to LEDR[5]
set_location_assignment PIN_Y19 -to LEDR[6]
set_location_assignment PIN_W20 -to LEDR[7]
set_location_assignment PIN_W21 -to LEDR[8]
set_location_assignment PIN_Y21 -to LEDR[9]
set_location_assignment PIN_AD7 -to PS2_CLK
set_location_assignment PIN_AD9 -to PS2_CLK2
set_location_assignment PIN_AE7 -to PS2_DAT
set_location_assignment PIN_AE9 -to PS2_DAT2
set_location_assignment PIN_AB12 -to SW[0]
set_location_assignment PIN_AC12 -to SW[1]
set_location_assignment PIN_AF9 -to SW[2]
set_location_assignment PIN_AF10 -to SW[3]
set_location_assignment PIN_AD11 -to SW[4]
set_location_assignment PIN_AD12 -to SW[5]
set_location_assignment PIN_AE11 -to SW[6]
set_location_assignment PIN_AC9 -to SW[7]
set_location_assignment PIN_AD10 -to SW[8]
set_location_assignment PIN_AE12 -to SW[9]
set_location_assignment PIN_H15 -to TD_CLK27
set_location_assignment PIN_D2 -to TD_DATA[0]
set_location_assignment PIN_B1 -to TD_DATA[1]
set_location_assignment PIN_E2 -to TD_DATA[2]
set_location_assignment PIN_B2 -to TD_DATA[3]
set_location_assignment PIN_D1 -to TD_DATA[4]
set_location_assignment PIN_E1 -to TD_DATA[5]
set_location_assignment PIN_C2 -to TD_DATA[6]
set_location_assignment PIN_B3 -to TD_DATA[7]
set_location_assignment PIN_A5 -to TD_HS
set_location_assignment PIN_F6 -to TD_RESET_N
set_location_assignment PIN_A3 -to TD_VS
set_location_assignment PIN_B13 -to VGA_B[0]
set_location_assignment PIN_G13 -to VGA_B[1]
set_location_assignment PIN_H13 -to VGA_B[2]
set_location_assignment PIN_F14 -to VGA_B[3]
set_location_assignment PIN_H14 -to VGA_B[4]
set_location_assignment PIN_F15 -to VGA_B[5]
set_location_assignment PIN_G15 -to VGA_B[6]
set_location_assignment PIN_J14 -to VGA_B[7]
set_location_assignment PIN_F10 -to VGA_BLANK_N
```

```
set_location_assignment PIN_A11 -to VGA_CLK
set_location_assignment PIN_J9 -to VGA_G[0]
set_location_assignment PIN_J10 -to VGA_G[1]
set_location_assignment PIN_H12 -to VGA_G[2]
set_location_assignment PIN_G10 -to VGA_G[3]
set_location_assignment PIN_G11 -to VGA_G[4]
set_location_assignment PIN_G12 -to VGA_G[5]
set_location_assignment PIN_F11 -to VGA_G[6]
set_location_assignment PIN_E11 -to VGA_G[7]
set_location_assignment PIN_B11 -to VGA_HS
set_location_assignment PIN_A13 -to VGA_R[0]
set_location_assignment PIN_C13 -to VGA_R[1]
set_location_assignment PIN_E13 -to VGA_R[2]
set_location_assignment PIN_B12 -to VGA_R[3]
set_location_assignment PIN_C12 -to VGA_R[4]
set_location_assignment PIN_D12 -to VGA_R[5]
set_location_assignment PIN_E12 -to VGA_R[6]
set_location_assignment PIN_F13 -to VGA_R[7]
set_location_assignment PIN_C10 -to VGA_SYNC_N
set_location_assignment PIN_D11 -to VGA_VS

############################################################################
################################### Other ##################################
############################################################################

set_global_assignment -name PARTITION_NETLIST_TYPE SOURCE -section_id Top
set_global_assignment -name PARTITION_FITTER_PRESERVATION_LEVEL PLACEMENT_AND_ROUTING -section_id
Top
set_global_assignment -name PARTITION_COLOR 16764057 -section_id Top
set_global_assignment -name STRATIX_DEVICE_IO_STANDARD "2.5 V"
set_global_assignment -name MIN_CORE_JUNCTION_TEMP 0
set_global_assignment -name MAX_CORE_JUNCTION_TEMP 85
set_instance_assignment -name PARTITION_HIERARCHY root_partition -to | -section_id Top
```

mult.qip

```
set_global_assignment -name IP_TOOL_NAME "LPM_MULT"
set_global_assignment -name IP_TOOL_VERSION "14.0"
set_global_assignment -name IP_GENERATED_DEVICE_FAMILY "{Cyclone V}"
set_global_assignment -name VHDL_FILE [file join $::quartus(qip_path) "mult.vhd"]
set_global_assignment -name MISC_FILE [file join $::quartus(qip_path) "mult.cmp"]
```

mult.vhd

```
-- megafunction wizard: %LPM_MULT%
-- GENERATION: STANDARD
-- VERSION: WM1.0
-- MODULE: lpm_mult

-- ============================================================
-- File Name: mult.vhd
-- Megafunction Name(s):
--                     lpm_mult
--
-- Simulation Library Files(s):
--                     lpm
-- ============================================================
-- ************************************************************
-- THIS IS A WIZARD-GENERATED FILE. DO NOT EDIT THIS FILE!
--
-- 14.0.2 Build 209 09/17/2014 SJ Full Version
-- ************************************************************


--Copyright (C) 1991-2014 Altera Corporation. All rights reserved.
--Your use of Altera Corporation's design tools, logic functions
--and other software and tools, and its AMPP partner logic
--functions, and any output files from any of the foregoing
--(including device programming or simulation files), and any
--associated documentation or information are expressly subject
--to the terms and conditions of the Altera Program License
--Subscription Agreement, the Altera Quartus II License Agreement,
--the Altera MegaCore Function License Agreement, or other
--applicable license agreement, including, without limitation,
--that your use is for the sole purpose of programming logic
--devices manufactured by Altera and sold by Altera or its
--authorized distributors.  Please refer to the applicable
--agreement for further details.


LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY lpm;
USE lpm.all;

ENTITY mult IS
	PORT
	(
		aclr		: IN STD_LOGIC ;
		clken		: IN STD_LOGIC ;
		clock		: IN STD_LOGIC ;
		dataa		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		datab		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		result		: OUT STD_LOGIC_VECTOR (31 DOWNTO 0)
	);
END mult;


ARCHITECTURE SYN OF mult IS

	SIGNAL sub_wire0	: STD_LOGIC_VECTOR (31 DOWNTO 0);
```

```
	COMPONENT lpm_mult
	GENERIC (
		lpm_hint		: STRING;
		lpm_pipeline		: NATURAL;
		lpm_representation		: STRING;
		lpm_type		: STRING;
		lpm_widtha		: NATURAL;
		lpm_widthb		: NATURAL;
		lpm_widthp		: NATURAL
	);
	PORT (
			aclr	: IN STD_LOGIC ;
			clken	: IN STD_LOGIC ;
			clock	: IN STD_LOGIC ;
			dataa	: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
			datab	: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
			result	: OUT STD_LOGIC_VECTOR (31 DOWNTO 0)
	);
	END COMPONENT;

BEGIN
	result    <= sub_wire0(31 DOWNTO 0);

	lpm_mult_component : lpm_mult
	GENERIC MAP (
		lpm_hint => "MAXIMIZE_SPEED=5",
		lpm_pipeline => 3,
		lpm_representation => "UNSIGNED",
		lpm_type => "LPM_MULT",
		lpm_widtha => 16,
		lpm_widthb => 16,
		lpm_widthp => 32
	)
	PORT MAP (
		aclr => aclr,
		clken => clken,
		clock => clock,
		dataa => dataa,
		datab => datab,
		result => sub_wire0
	);


END SYN;

-- ============================================================
-- CNX file retrieval info
-- ============================================================
-- Retrieval info: PRIVATE: AutoSizeResult NUMERIC "0"
-- Retrieval info: PRIVATE: B_isConstant NUMERIC "0"
-- Retrieval info: PRIVATE: ConstantB NUMERIC "0"
-- Retrieval info: PRIVATE: INTENDED_DEVICE_FAMILY STRING "Cyclone V"
-- Retrieval info: PRIVATE: LPM_PIPELINE NUMERIC "3"
-- Retrieval info: PRIVATE: Latency NUMERIC "1"
-- Retrieval info: PRIVATE: SYNTH_WRAPPER_GEN_POSTFIX STRING "0"
-- Retrieval info: PRIVATE: SignedMult NUMERIC "0"
-- Retrieval info: PRIVATE: USE_MULT NUMERIC "1"
-- Retrieval info: PRIVATE: ValidConstant NUMERIC "0"
-- Retrieval info: PRIVATE: WidthA NUMERIC "16"
-- Retrieval info: PRIVATE: WidthB NUMERIC "16"
-- Retrieval info: PRIVATE: WidthP NUMERIC "32"
-- Retrieval info: PRIVATE: aclr NUMERIC "1"
-- Retrieval info: PRIVATE: clken NUMERIC "1"
-- Retrieval info: PRIVATE: new_diagram STRING "1"
-- Retrieval info: PRIVATE: optimize NUMERIC "0"
-- Retrieval info: LIBRARY: lpm lpm.lpm_components.all
-- Retrieval info: CONSTANT: LPM_HINT STRING "MAXIMIZE_SPEED=5"
-- Retrieval info: CONSTANT: LPM_PIPELINE NUMERIC "3"
-- Retrieval info: CONSTANT: LPM_REPRESENTATION STRING "UNSIGNED"
```

```
-- Retrieval info: CONSTANT: LPM_TYPE STRING "LPM_MULT"
-- Retrieval info: CONSTANT: LPM_WIDTHA NUMERIC "16"
-- Retrieval info: CONSTANT: LPM_WIDTHB NUMERIC "16"
-- Retrieval info: CONSTANT: LPM_WIDTHP NUMERIC "32"
-- Retrieval info: USED_PORT: aclr 0 0 0 0 INPUT NODEFVAL "aclr"
-- Retrieval info: USED_PORT: clken 0 0 0 0 INPUT NODEFVAL "clken"
-- Retrieval info: USED_PORT: clock 0 0 0 0 INPUT NODEFVAL "clock"
-- Retrieval info: USED_PORT: dataa 0 0 16 0 INPUT NODEFVAL "dataa[15..0]"
-- Retrieval info: USED_PORT: datab 0 0 16 0 INPUT NODEFVAL "datab[15..0]"
-- Retrieval info: USED_PORT: result 0 0 32 0 OUTPUT NODEFVAL "result[31..0]"
-- Retrieval info: CONNECT: @aclr 0 0 0 0 aclr 0 0 0 0
-- Retrieval info: CONNECT: @clken 0 0 0 0 clken 0 0 0 0
-- Retrieval info: CONNECT: @clock 0 0 0 0 clock 0 0 0 0
-- Retrieval info: CONNECT: @dataa 0 0 16 0 dataa 0 0 16 0
-- Retrieval info: CONNECT: @datab 0 0 16 0 datab 0 0 16 0
-- Retrieval info: CONNECT: result 0 0 32 0 @result 0 0 32 0
-- Retrieval info: GEN_FILE: TYPE_NORMAL mult.vhd TRUE
-- Retrieval info: GEN_FILE: TYPE_NORMAL mult.inc FALSE
-- Retrieval info: GEN_FILE: TYPE_NORMAL mult.cmp TRUE
-- Retrieval info: GEN_FILE: TYPE_NORMAL mult.bsf FALSE
-- Retrieval info: GEN_FILE: TYPE_NORMAL mult_inst.vhd FALSE
-- Retrieval info: LIB_FILE: lpm
```

mult_unit.vhd

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

ENTITY mult_unit IS
	PORT( clk, reset, enable                    : IN STD_LOGIC;
		mult_a, mult_b                              : IN STD_LOGIC_VECTOR(15
DOWNTO 0);
		mult_done                                                   : OUT STD_LOGIC;
		mult_result                                                 : OUT
STD_LOGIC_VECTOR(31 DOWNTO 0):= (others => '0'));
END ENTITY mult_unit;

ARCHITECTURE Behaviour of mult_unit IS

COMPONENT mult
	PORT
	(
		aclr		: IN STD_LOGIC ;
		clken		: IN STD_LOGIC ;
		clock		: IN STD_LOGIC ;
		dataa		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		datab		: IN STD_LOGIC_VECTOR (15 DOWNTO 0);
		result		: OUT STD_LOGIC_VECTOR (31 DOWNTO 0)
	);
END COMPONENT;

SIGNAL done0, done1, done2 : STD_LOGIC;
SIGNAL temp_res : STD_LOGIC_VECTOR(31 DOWNTO 0);

BEGIN

multiplier : mult
PORT MAP(aclr => reset, clken => enable, clock => clk,
		dataa   => mult_a, datab => mult_b, result => temp_res);

PROCESS(clk, reset, enable)
BEGIN
	IF(reset = '1')THEN
		done0 <= '0'; done1 <= '0'; done2 <= '0';
	ELSIF(rising_edge(clk))THEN
		IF(enable = '1')THEN
				done0 <= '1';
		END IF;
		--pipeline the done signal til multiplication is complete
		done1 <= done0;
		done2 <= done1;
	END IF;
END PROCESS;

mult_result <= temp_res;
mult_done <= done2;

END Behaviour;
```

mult_control.vhd

```
-- mult_control.vhd

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity mult_control is
	port (
		avs_s0_address   : in  std_logic_vector(3 downto 0)  := (others => '0'); --
s0.address
		avs_s0_write     : in  std_logic                     := '0';             --
.write
		avs_s0_writedata : in  std_logic_vector(31 downto 0) := (others => '0'); --
.writedata
		avs_s0_read      : in  std_logic                     := '0';             --
.read
		avs_s0_readdata  : out std_logic_vector(31 downto 0);                    --
.readdata
		clk              : in  std_logic                     := '0';             --
clock.clk
		reset            : in  std_logic                     := '0';             --
reset.reset
		mult_start       : out std_logic_vector(31 downto 0);                    --
.m_start
		mult_reset       : out std_logic_vector(31 downto 0);                    --
.m_reset
		mult_done        : in  std_logic_vector(31 downto 0) := (others => '0')  --
.m_done
	);
end entity mult_control;

architecture rtl of mult_control is

	SIGNAL start, reset2 : STD_logic_vector(31 DOWNTO 0);

begin

		PROCESS(clk, reset, avs_s0_address, avs_s0_write, avs_s0_writedata, avs_s0_read)
	BEGIN
		IF(reset = '1')THEN
			start <= (OTHERS => '0');
			reset2 <= (OTHERS => '0');
		ELSIF(rising_edge(clk))THEN

			IF(avs_s0_write = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						start <= avs_s0_writedata;
					WHEN "0001" =>
						reset2 <= avs_s0_writedata;
					WHEN OTHERS =>

				END CASE;
			ELSIF(avs_s0_read = '1')THEN
				reset2 <= (OTHERS => '0');
				CASE avs_s0_address IS
					WHEN "0000" =>
						avs_s0_readdata <= start;
					WHEN "0001" =>
						avs_s0_readdata <= reset2;
					WHEN "0010" =>
						avs_s0_readdata <= mult_done;
					WHEN OTHERS =>

				END CASE;
			END IF;
		END IF;
	END PROCESS;
```

```
        mult_start <= start;
        mult_reset <= reset2;

end architecture rtl; -- of mult_input
```

mult_data.vhd

```
-- mult_data.vhd
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity mult_data is
	port (
		avs_s0_address  : in  std_logic_vector(3 downto 0)  := (others => '0'); --
s0.address
		avs_s0_read     : in  std_logic                     := '0';             --
.read
		avs_s0_write    : in  std_logic                     := '0';             --
.write
		avs_s0_readdata : out std_logic_vector(31 downto 0);                    --
.readdata
		avs_s0_writedata  : in  std_logic_vector(31 downto 0)  := (others => '0'); --
s0.address
		clk             : in  std_logic                     := '0';             --
clock.clk
		reset           : in  std_logic                     := '0';             --
reset.reset
		mult_in1        : out std_logic_vector(31 downto 0);                    --
mult_input.m_in1
		mult_in2        : out std_logic_vector(31 downto 0);                    --
mult_input.m_in2
		mult_result     : in  std_logic_vector(31 downto 0) := (others => '0') --
mult_output.m_result
	);
end entity mult_data;

architecture rtl of mult_data is

	SIGNAL in1, in2 : STD_logic_vector(31 DOWNTO 0);

begin
	PROCESS(clk, reset, avs_s0_read, avs_s0_write, avs_s0_address, avs_s0_writedata)
	BEGIN
		IF(reset = '1')THEN
			avs_s0_readdata <= (OTHERS => '0');
			in1 <= (OTHERS => '0');
			in2 <= (OTHERS => '0');

		ELSIF(rising_edge(clk))THEN
			IF(avs_s0_read = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						avs_s0_readdata <= mult_result;
					WHEN "0001" =>
						avs_s0_readdata <= in1;
					WHEN "0010" =>
						avs_s0_readdata <= in2;
					WHEN OTHERS =>
						avs_s0_readdata <= (OTHERS => '0');
				END CASE;
			ELSIF(avs_s0_write = '1')THEN
				CASE avs_s0_address IS
					WHEN "0000" =>
						in1 <= "0000000000000000" & avs_s0_writedata(15
DOWNTO 0);
					WHEN "0001" =>
						in2 <= "0000000000000000" & avs_s0_writedata(15
DOWNTO 0);
					WHEN OTHERS =>

					END CASE;

			END iF;
		END IF;
	END PROCESS;
```

```
        mult_in1 <= in1;
        mult_in2 <= in2;

end architecture rtl; -- of mult_output
```