3TB4 Lab 4 Prelab Report

Rachel Tsang #400388338

Bowen Wang #400325152

Tutorial Questions

Q1: The nios2_qsys is the master device.

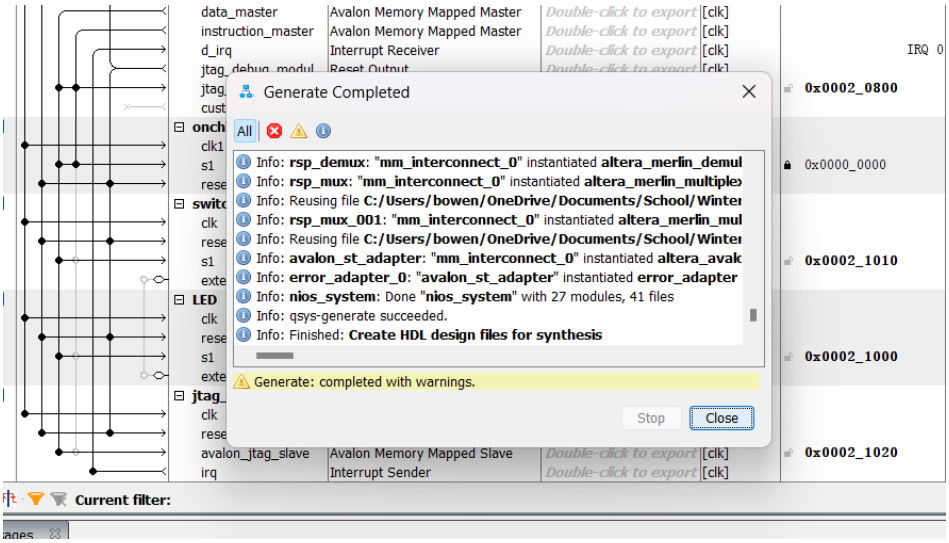
The onchip memory, LED, jtag_uart and switch are slave devices .

Q2:

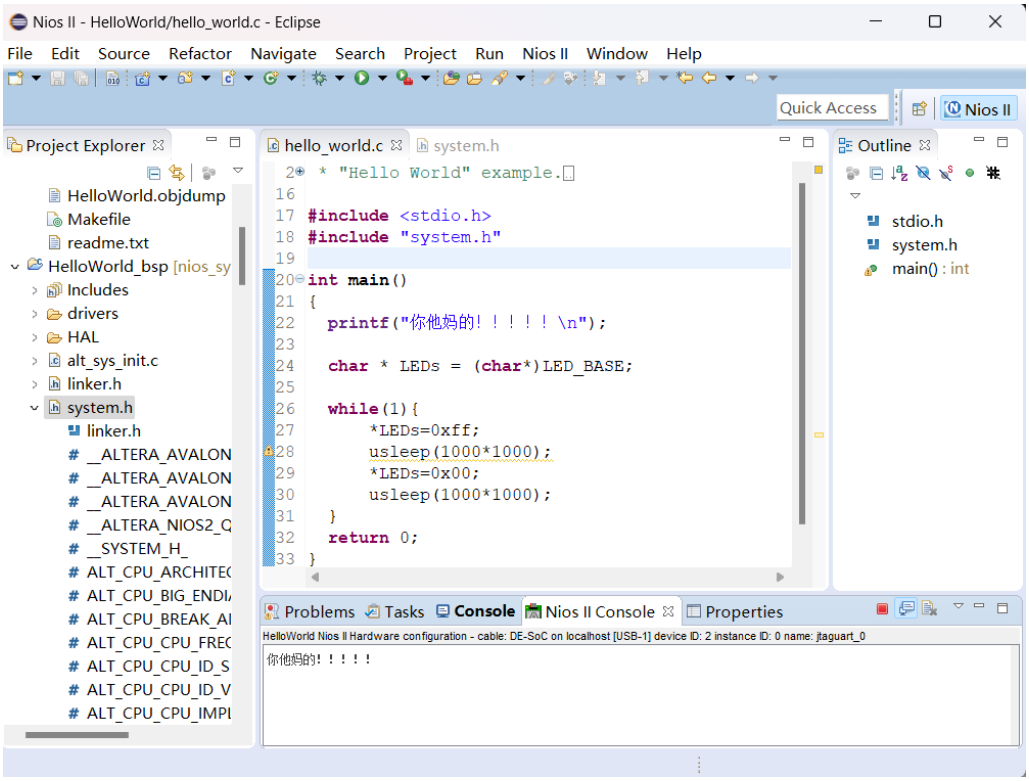
- Quartus: Both software and hardware, Quartus is a software development tool for hardware systems
- Qsys (Platform Designer): hardware: this is used to simulated hardware by making wire connections and working with memory data.
- Nios SBT for Eclipse: software: this is used to compile/edit c codes on mcu which facilitates software development.
- Signal Tap Logic Analyzer: hardware: used to visually see the hardware output signals in binary

Tutorial screenshots:

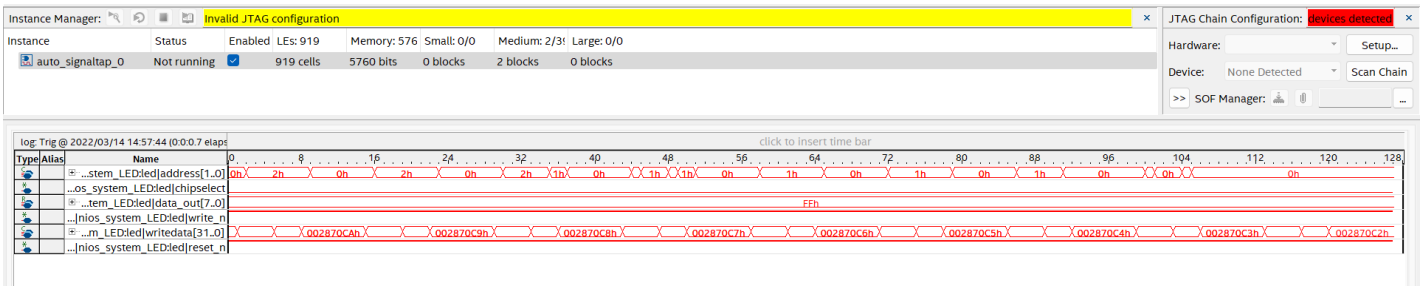
Part 1



Part 2



Part 3



Prelab Questions:

Q1: Considering that the SDRAM chip you will be using has the capacity of 64M bytes, and assuming that the lowest address of the SRDRAM chip is 0x00000000, what will be the highest byte address?

1Mb = 1048576 bytes, 64 Mb = 67108864 Bytes

In hexadecimal: 67108864-1 = 0x003FFFFFF

Q2:

The system assign 16 bit to one word, the address of the lowest word is 0x00000000, and the address of the highest word is 0x003FFFFFE

Q3: when the second byte is accessed, it will appear 0x000 on the SDRAM address line.

Q4:

Code added onto lab4.v starter file-

```
sopc_system  controller (
       // example ports
       .clk_clk(CLOCK_50),
       .reset_reset_n(KEY[0]),
       .sdram_clk_clk(DRAM_CLK),
       // more ports
       .sdram_addr_export(DRAM_ADDR),
       .sdram_ba_export(DRAM_BA),
       .sdram_cas_n_export(DRAM_CAS_N),
       .sdram_cke_export(DRAM_CKE),
       .sdram_cs_n_export(DRAM_CS_N),
       .sdram_dq_export(DRAM_DQ),
       .sdram_ldaqm_export(DRAM_LDQM),
       .sdram_ras_n_export(DRAM_RAS_N),
       .sdram_udqm_export(DRAM_UDQM),
       .sdram_we_n_export(DRAM_WE_N),
       /*
       .sram_controller_0_conduit_end_export(SRAM_DQ),
       .sram_controller_0_conduit_end_1_export(SRAM_ADDR),
       .sram_controller_0_conduit_end_2_export(SRAM_CE_N_wire),
       .sram_controller_0_conduit_end_3_export(SRAM_WE_N_wire),
       .sram_controller_0_conduit_end_4_export(SRAM_OE_N_wire),
       .sram_controller_0_conduit_end_5_export(SRAM_UB_N_wire),
       .sram_controller_0_conduit_end_6_export(SRAM_LB_N_wire),
       */
   );



endmodule
```

Q5:

Schematic of circuit –

Q6:

Code added onto SDRAM_Controller.v starter file -

```
  //  ==========Make  more  necessary connections=================
  //
   //
     assign DRAM_ADDR=DRAM_ADDR_wire;
     assign DRAM_BA=DRAM_BA_wire;
     assign DRAM_CKE=DRAM_CKE_wire;
     assign DRAM_WE_N=DRAM_WE_N_wire;
     assign DRAM_CAS_N=DRAM_CAS_N_wire;
     assign DRAM_RAS_N=DRAM_RAS_N_wire;
     assign DRAM_CS_N=DRAM_CS_N_wire;


endmodule
```

Q7:

C file -

```
#include "system.h"
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int i;
//-------------------------------------char test---------------------------------
//
char read_char[5];
char write_char[5] = {'a', 'b', 'c', 'd', 'e'};
char *shart = (char *)SRAM_CONTROLLER_0_BASE;
void test_char(){
for (i=0; i<5; i++){
    *shart = write_char[i];
    printf("writing char: %c \n", *shart);
    shart++;
}
return;
}
//-------------------------------------short test---------------------------------
//
short read_short[5];
short write_short[5] = {'100', '200', '300', '400', '500'};
short *shart = (char *)SRAM_CONTROLLER_0_BASE;
void test_short(){
for (i=0; i<5; i++){
    *shart = write_short[i];
    printf("writing short: %c \n", *shart);
    shart++;
```

```
}
return;
}
//-------------------------------------int test--------------------------------//
int read_int[5];
int write_int[5] = {'1', '2', '3', '4', '5'};
int *shart = (char *)SRAM_CONTROLLER_0_BASE;
void test_int(){
for (i=0; i<5; i++){
    *shart = write_int[i];
    printf("writing int: %c \n", *shart);
    shart++;
}
return;
}
```