Mitchell Jonker
Hunter Frady
CSCE-313-002 Embedded Systems
02/03/2023

Project 2

1.0

A system on a chip (SoC) is an integrated circuit (IC) that contains all the necessary components for a computer system, such as a processor, memory, and peripherals, on a single chip. It is used in embedded systems and other applications where space and power consumption are important considerations.

2.0

The Avalon Memory Mapped interface is a high-performance, low-latency, point-to-point connection protocol. It is used to connect the sub-components of a system-on-chip (SoC). It is used to transfer data between components, such as the processor cores, memory locations, and peripheral devices. In Project 2 we used the peripheral devices such as the LEDs and Switches, and the memory locations to run store data, and the processor unit to manipulate and work with the data.

3.0

Bare metal applications are applications that run directly on the hardware without an operating system. These kinds of applications are common in embedded systems. Bare Metal apps are written in low-level languages such as assembly code or C. Bare metal applications can control hardware directly and provide a high level of control. Performance from bare metal applications is improved due to not requiring an intermediary (operating system). In Project 2 the C hardware code is directed through the HAL and is run directly on the hardware. This is why reconfiguring the board once or twice may be required.

4.0

Screenshot below shows the hardware C code.

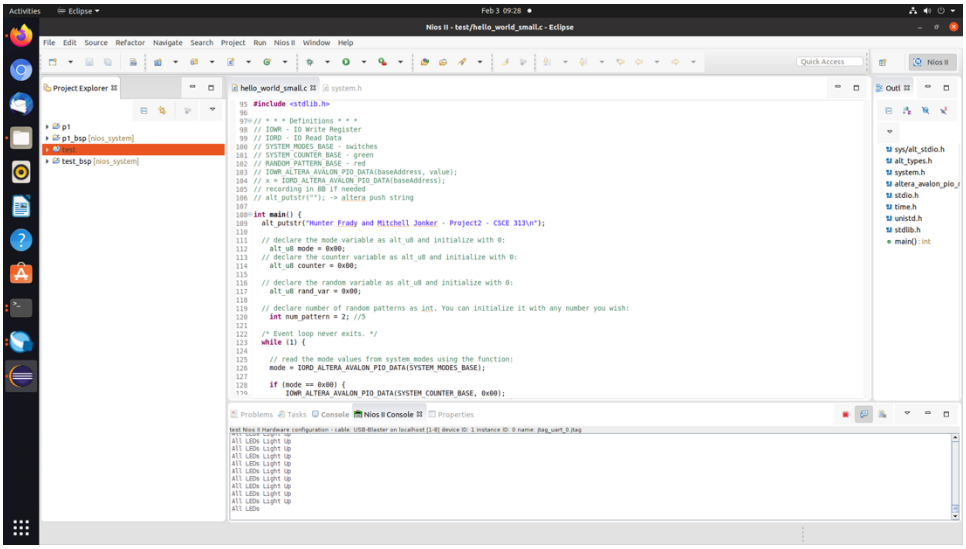
```c
// Hardware Libraries
#include "sys/alt_stdio.h"
#include "alt_types.h"
#include "system.h"
#include "altera_avalon_pio_regs.h"

// C Libraries
#include <stdio.h>
#include <time.h>
#include <unistd.h>
#include <stdlib.h>

int main() {

  alt_putstr("\n");

  alt_u8 a = 0x00; // switches
  alt_u8 b = 0x00; // green LEDs

  while (1) {

    if (a == 0x00) {
      alt_putstr("no output\n");

    } else if (a == 0x01) {

      b = 0x0A;
      IOWR_ALTERA_AVALON_PIO_DATA(SYSTEM_COUNTER_BASE, b);
    }

    usleep(1000000);
  }

  return 0;
}

```

5.0

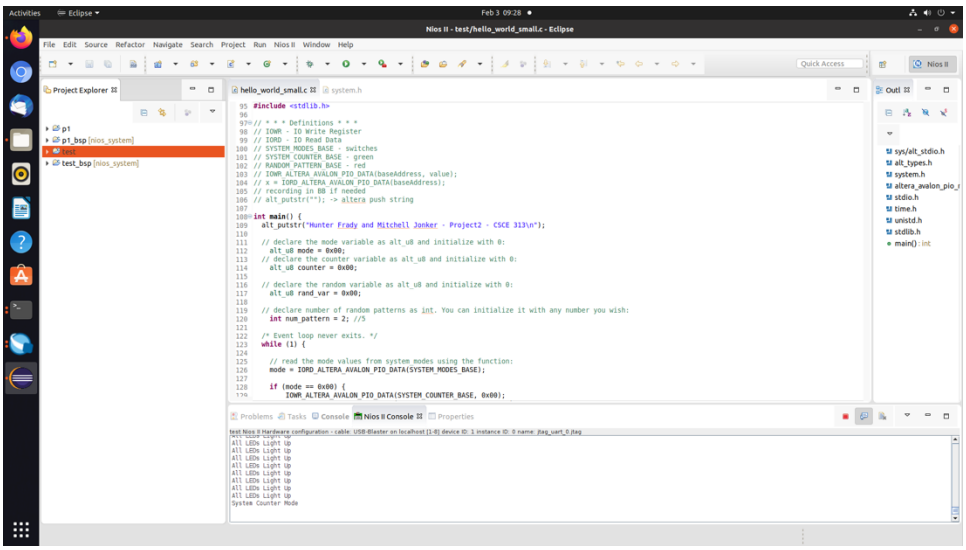
Logical Elements	Registers	Total Pins	Memory Bits
3183	1889	31	604672

6.0

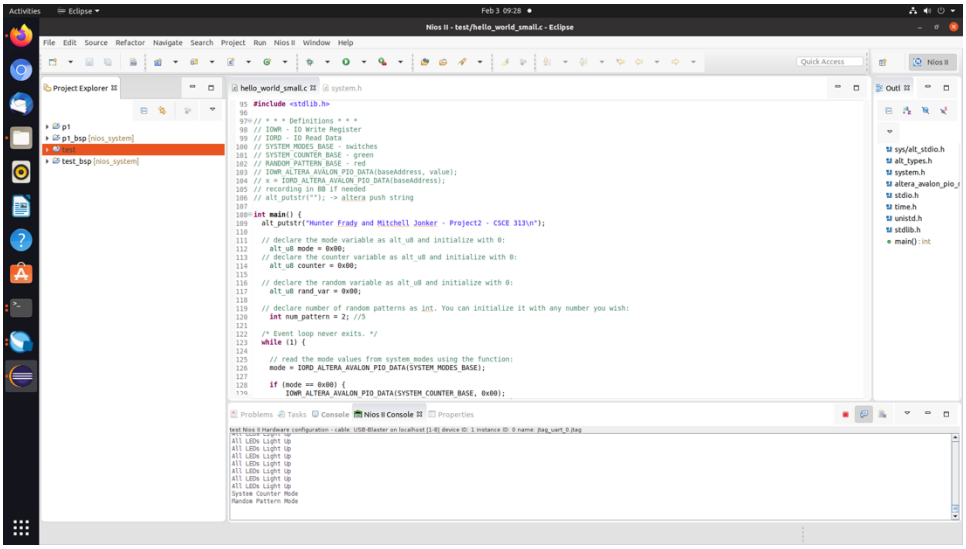
Mode 1 -



Mode 2 -



Mode 3 -


Activities
Eclipse
Feb 3 09:28
Nios II - test/hello_world_small.c - Eclipse
File Edit Source Refactor Navigate Search Project Run Nios II Window Help
Project Explorer
p1
p1_bsp [nios_system]
test
test_bsp [nios_system]
hello_world_small.c
system.h
#include <stdlib.h>
// * * * Definitions * * *
// IOWR - IO Write Register
// IORD - IO Read Data
// SYSTEM_MODES_BASE - switches
// SYSTEM_COUNTER_BASE - green
// RANDOM_PATTERN_BASE - red
// IOWR_ALTERA_AVALON_PIO_DATA(baseAddress, value);
// x = IORD_ALTERA_AVALON_PIO_DATA(baseAddress);
// recording in BB if needed
// alt_putstr(""); -> altera push string
int main() {
alt_putstr("Hunter Frady and Mitchell Jonker - Project2 - CSCE 313\n");
// declare the mode variable as alt_u8 and initialize with 0:
alt_u8 mode = 0x00;
// declare the counter variable as alt_u8 and initialize with 0:
alt_u8 counter = 0x00;
// declare the random variable as alt_u8 and initialize with 0:
alt_u8 rand_var = 0x00;
// declare number of random patterns as int. You can initialize it with any number you wish:
int num_pattern = 2; //5
/* Event loop never exits. */
while (1) {
// read the mode values from system_modes using the function:
mode = IORD_ALTERA_AVALON_PIO_DATA(SYSTEM_MODES_BASE);
if (mode == 0x00) {
IOWR_ALTERA_AVALON_PIO_DATA(SYSTEM_COUNTER_BASE, 0x00);
Problems Tasks Console Nios II Console Properties
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
All LEDs Light Up
System Counter Mode
Random Pattern Mode
Outl
sys/alt_stdio.h
alt_types.h
system.h
stdio.h
time.h
unistd.h
stdlib.h
main() : int