Actuating, Sensing and Control Mechatronic Systems

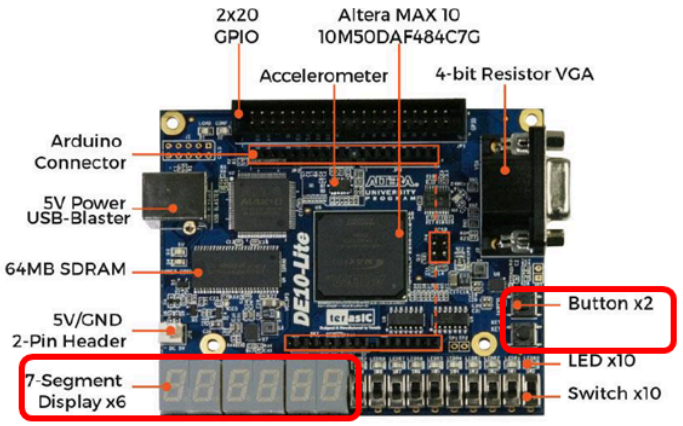
Laboratory instruction

System on chip – create program for soft-processor NiosII

Requirements for the exercise (issues and skills necessary to complete the task)
● representation of numbers in decimal, binary, hexadecimal and BCD systems; ● setting up a new project in Quartus Prime; ● creating a hardware module (symbol) in Quartus Prime based on a schematic file (*.bdf); ● include a hardware module in Quartus Prime; ● creating a hardware system using Platform Designer tool – Qsys ● basics of programming C language

INTRODUCTION

The next step in designing the SopC system is to write a program for the Nios processor. We use a tool - Nios II SBT for Eclipse at the level of software to write program code, compilation and implementation.



Development Board (top view)

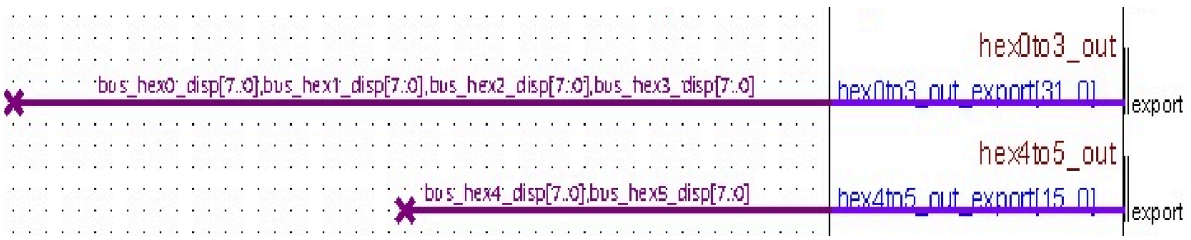
Functional description of task 4

- While the button (KEY0) is pressed, the counter value is incremented.
- While the button (KEY1) is pressed, the counter value is decremented.
- Pressing the buttons simultaneously does not change the counter value.
- The counter value is displayed on 7-segment displays

Hardware connection of elements on board (task 4)

- Buttons (KEY0, KEY1) in the normal state (not pressed) generate a high level on the FPGA input
- Displays (7-seg. display x6) are set high by default (pull-up connection), which means the LED segment will turn off when set high on the FPGA port

The configuration of SOPC ports and bus connections - program hardware layer (task 4)



TASK 1. Create new project in Nios II SBT for Eclipse from template - ***Hello World Small***

STEP 1: Download Hardware Design to Target FPGA !

NOTE: information (lecture 3, side:4-6): Narzędzia_software_v1.pdf

STEP 2: Create software project of Nios II SBT for Eclipse

NOTE: information (lecture 3, side:7-14): Narzędzia_software_v1.pdf
use project template: *Hello word small*

STEP 3: Implementation - Run configurations - Nios II programming

NOTE: information (lecture 3, side:15-17): Narzędzia_software_v1.pdf

TASK 2. Write the button press test program - example of using the *IOWR()* function

STEP 1: Rewrite code from (lecture 3, side:23): Narzędzia_software_v1.pdf

NOTE: ***Write code in hello_world_small.c file - don't change project and don't create new blank project or new *.c file !!!***

STEP 2: Make sure that the component (IP-CORE) of your PIO-LEDS system has the same label as defined in the "system.h" library (e.g. *LEDS_BASE*) and which you will use as the input argument of the IOWR() function.

NOTE: information (lecture 3, side: 23-24): Narzędzia_software_v1.pdf

STEP 3: Build project and run program.

NOTE: ***Observe changes on the board and analyze code execution.***

TASK 3. Write the button press test program - example of using the *IORD()* function

STEP 1: Rewrite code from (lecture 3, side:27): Narzędzia_software_v1.pdf

NOTE: ***Write code in hello_world_small.c file - don't change project and don't create new blank project or new *.c file !!!***

STEP 2: Make sure that the component (IP-CORE) of your PIO-KEY system has the same label as defined in the "system.h" library (e.g. *KEY_BASE*) and which you will use as the input argument of the IORD() function.

STEP 3: Build project and run program.

NOTE: ***Observe changes on the board and analyze code execution***

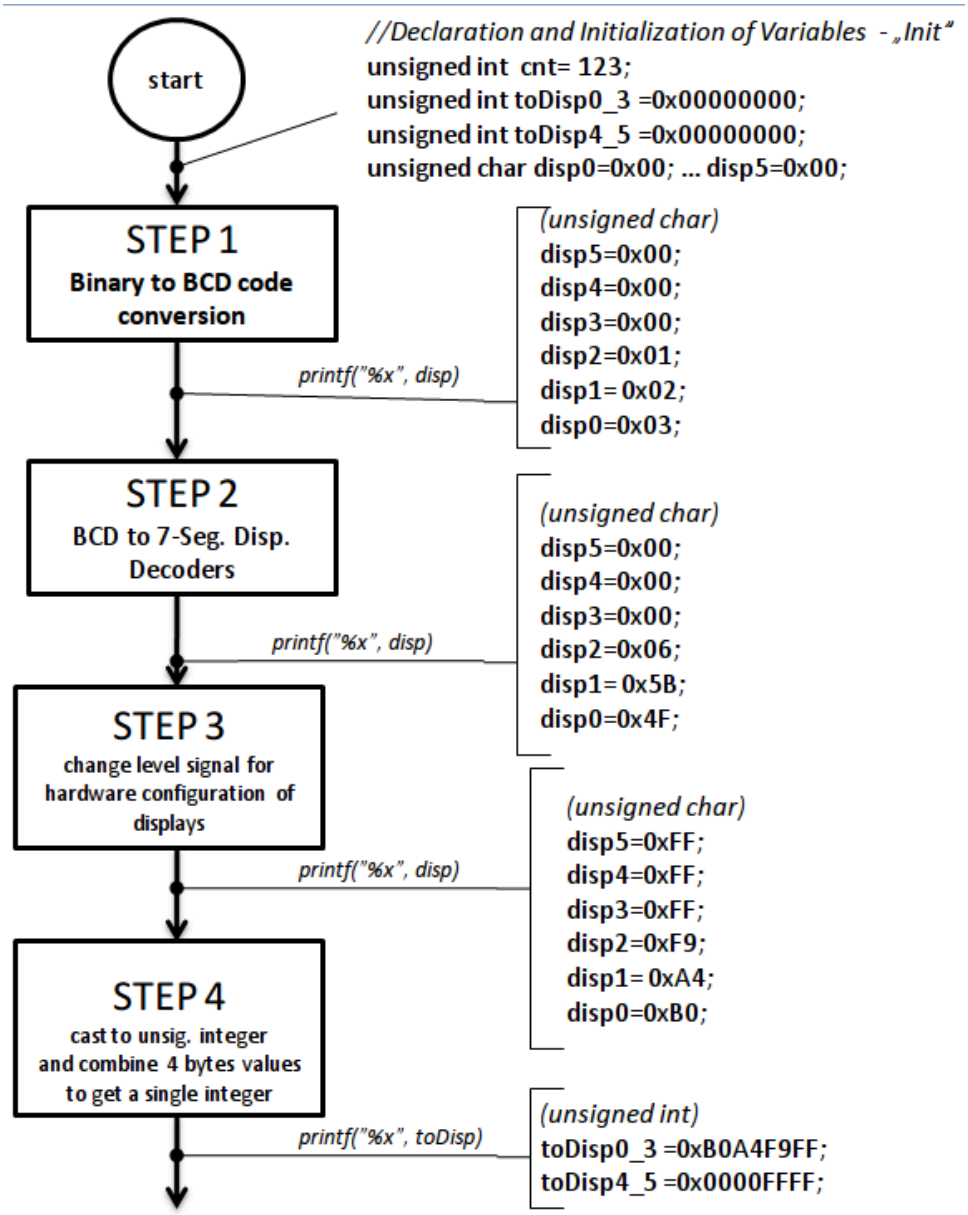
TASK 4. Write the display of a counter value program - - *task to solve on your own*

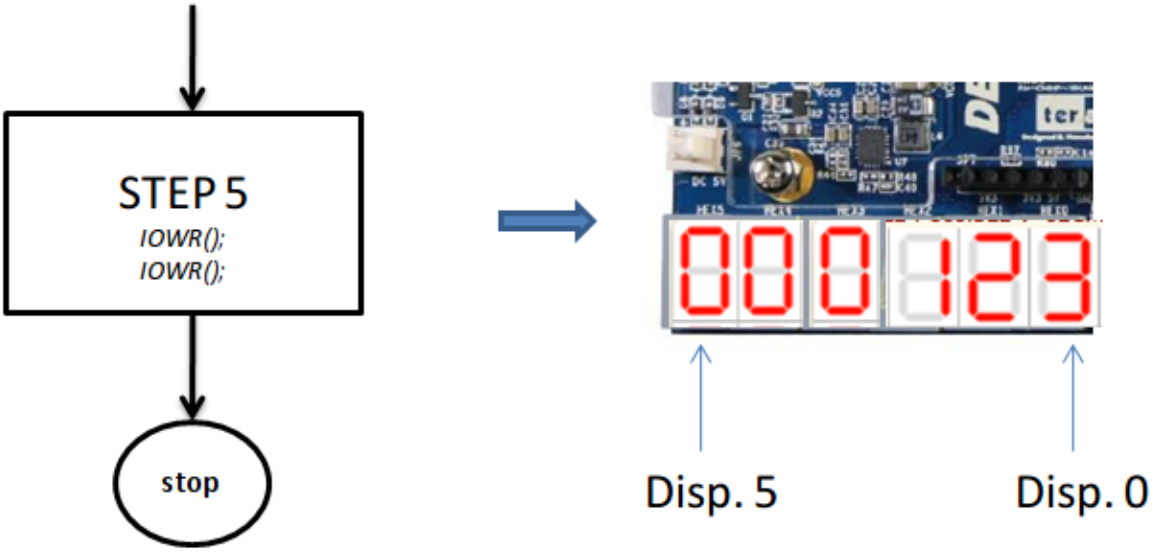
STEP 1: Read the first chapter of this lab instruction - Introduction.
Analyze the functional purpose of the project and the hardware structure.

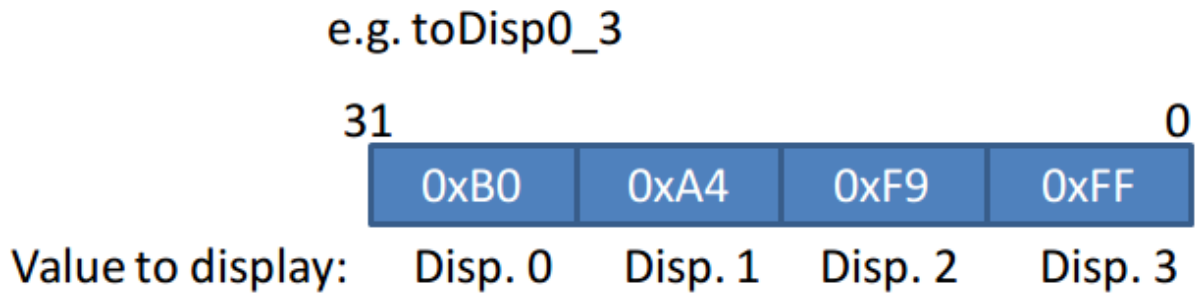
STEP 2: Use your functions written in Project 1 (Project_1.pdf): **Binary to BCD code conversion** and **BCD to 7-Seg. Disp. Decoders**

STEP 3: Build project and run program.

NOTE: Below is a graph with information that may be helpful in dividing the algorithm into functions. The presented graph is for illustrative purposes only.

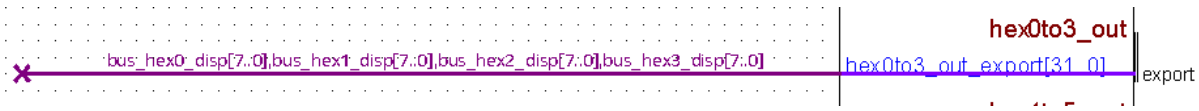






because configuration of the bus is:

bus_hex0_disp[7..0],bus_hex1_disp[7..0],bus_hex2_disp[7..0],bus_hex3_disp[7..0],



NOTE**: *(for advance programmer)*

- *reduce the size of code - use function and loop.*
- *optimal allocation size of memory for your code -use pointer and matrix. Minimize the number of variables necessary to execute a program.*
- *comments are part of the code - use them*
- *the final version of code should not contain printf() functions - comment them*
- *find the optimal value of the usleep() function argument so that the user can observe the change of counter value on displays*
- *experiment with code, let writing programs be fun for you, not just a task to solve*