**2. Task**

The task is a timer. It is necessary to implement a timer using 8 switches as input, which gives

seconds in advance in binary code. The value should be visible on the 7-segment display

in hexadecimal. After the entry, the 7-segment display must show what is left

seconds. Extra points are obtained when the red LED starts flashing when the time reaches zero.

The functions handle the conversion of all number systems themselves.

In addition to the buttons and LEDs used in the previous task, you also need to implement the timer

7-segment display (Seven Segment Display or SSD) and RGB LEDs and switches.

SSD library: ssd.h

RGB-LED library: rgbled.h

Switch library: swt.h

Like other components on the board, the SSD, switches and RGB-LED must be initialized first

use, and the project must contain their respective Source and Header files.

For this we use the “SSD\_Init()”, ”SWT\_Init()” and “RGBLED\_Init()” commands.

Care must be taken with the RGBLED\_Init() command as it slows down the controller. So this

is the only Init command that should be used immediately before using the RGB-LED function

and immediately after using the functions it should be closed with RGBLED\_Close() command.

To display the SSD on the screen, it is necessary to use the command:

SSD\_WriteDigitsGrouped(int x, int y)

x is the displayed integer and y are the decimal places (not necessary in this task and you can simply enter 0).

You can use SWT\_GetValue(int x) and SWT\_GetGroupValue() to read switches

The difference between the two is that the first gives the status of only one switch, the second gives all in 8-bit

in the format (0 means all up, i.e. off, and 255 means all down, i.e. on).

The command to use RGB-LED is: RGBLED\_SetValue(int r, int g, int b)

Each integer represents an eight-bit part of one color in the LED. For example, for the color green, enter:

0, 255, 0

And for the purple color you have to enter: 255, 0, 255

2.Ülesanne

Ülesanne on taimer. Tuleb realiseerida taimer, kasutades sisendina 8’t lülitit, millega antakse

binaarkoodis ette sekundid. 7-segmendilise ekraani pealt peaks olema väärtus nähtav

kuueteistkümnendkoodis. Peale sisestust tuleb 7-segmendilisel ekraanil näidata allesjäänud

sekundeid. Lisa punkte saadakse, kui aja nulli jõudmisel hakkab vilkuma punane LED tuluke.

Kõikide arvsüsteemide konversiooniga tegelevad funktsioonid ise.

Lisaks eelmises ülesandes kasutatud nuppudele ja LED’dele, on taimeri realiseerimiseks vaja ka

7segmendilist ekraani(Seven Segment Display ehk SSD) ja RGB-LED’I ning lüliteid.

SSD teek: ssd.h

RGB-LED’I teek: rgbled.h

Lülitite teek: swt.h

Sarnaselt teistele plaadi komponentidele, tuleb ka SSD, lülitid ja RGB-LED initsialiseerida ennem

kasutamist ning project peab sisaldama nende vastavaid Source ja Header file.

Selleks kasutame “SSD\_Init()”, ”SWT\_Init()” ja “RGBLED\_Init()” käske.

RGBLED\_Init() käsuga peab olema ettevaatlik, kuna see aeglustab kontrolleri tööd. Seega see

on ainuke Init käsk mida peaks kasutama viivitamatult enne RGB-LED funktsiooni kasutamist

ning koheselt pärast funktsioonide kasutust peaks selle sulgema käsuga RGBLED\_Close().

SSD ekraanile kuvamiseks on vaja kasutada käsku:

SSD\_WriteDigitsGrouped(int x, int y)

x on kuvatav täisarv ja y on komakohad(siin ülesanndes mitte vajalik ja võib lihtsalt sisestada 0’i).

Lülitite lugemiseks saab kasutada SWT\_GetValue(int x) ning SWT\_GetGroupValue()

Nende kahe vahe on see, et esimene annab ainult ühe lüliti staatuse, teine annab kõikide 8-bitises

formaadis(0 tähendab kõik üleval, ehk väljalülitatud, ja 255, et kõik all, ehk sisselülitatud).

RGB-LED kasutamiseks on käsk: RGBLED\_SetValue(int r, int g, int b)

Iga täisarv näitab kaheksa bitiga ühe värvi osa LED’s. Näiteks rohelise värvi jaoks, tuleb sisestada:

0, 255, 0

Ja lilla värvi jaoks tuleb sisestada:

255, 0, 255