Институт за математику и информатику

Природно-математички факултет

Универзитет у Крагујевцу

Завршни пројекат из предмета Микропроцесорски системи

Тема: Паметни радијатор који омогућава задавање температуре и временски период укључења и искључења

Студент:

Лука Ристовић 83/2018

Професор:

др Александар Пеулић

Фебруар 2023.

Садржај

1. Увод

Живимо у времену где је веома важно штедети електричну као и било које друге енергију. Самим тим се производи све више уређаја са оптималном потрошњом истих. Циљ овог пројекта (Паметни радијатор који омогућава задавање температуре и временски период укључења и искључења) је концептуални развој и реализација радијатора зарад оптималнијег коришћења електричне енергије а притом да на што ефикаснији начин људе згреју у својим домовима у хладним данима.

За реализацију овог пројекта је коришћен програм Proteus који служи за симулацију и софтвер STM32CubeIDE у коме је писан код за микроконтролер STM32F103C6.

Слика 1. Илустрован приказ паметниг радијатора

2. Објашњавање алгоритма и кода

У овом поглављу ћемо детаљно описати логику и сам код програма. Овај радиатор у себи садржи један LCD, један термостат, један сензор који мери тренутну температуру просторије,дугме за паљење и гашење самог радијатора и један тимер за паљење/гашење радијатора у одређеном тренутку.

Кренућемо од кода за LCD (Liquid Crystal Display).

```
71 void LCD(uint8_t val_1, uint8_t cmd)
72 {
73
74     uint8_t data;
75
76     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, cmd);
77
78     data = val_1 & 0x01;
79     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_10, data);
80
81     data = (val_1 >> 1) & 0x01;
82     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, data);
83
84     data = (val_1 >> 2) & 0x01;
85     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, data);
86
87     data = (val_1 >> 3) & 0x01;
88     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, data);
89
90     data = (val_1 >> 4) & 0x01;
91     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, data);
92
93     data = (val_1 >> 5) & 0x01;
94     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, data);
95
96     data = (val_1 >> 6) & 0x01;
97     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, data);
98
99     data = (val_1 >> 7) & 0x01;
00     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_11, data);
01
02
03     HAL_GPIO_WritePin(GPIOA,GPIO_PIN_9,GPIO_PIN_SET);
04     HAL_Delay(50);
05     HAL_GPIO_WritePin(GPIOA,GPIO_PIN_9,GPIO_PIN_RESET);
06 }
07
08 void LCD_init()
09 {
10     LCD(0x38,0);
11     LCD(0x0C,0);
12     LCD(0x06,0);
13     LCD(0x01,0);
14     LCD(0x80,0);
15 }
```

Слика 2. Изглед LCD функција

У овом делу кода смо направили две функције: LCD_Init и LCD. LCD_Init функција служи за иницијализацију само екрана. Као што је на пример чишћење екрана, подешавање курсора на почетну позицију за неки испис, итд. Друга функција служи да извршава послате наредбе што од малопређашње функције тако и касније у коду за разне исписе.

Сам радијатор се може укључити на стандардан ON/OFF прекидач и када се он притисне, покреће се такозвани прекид у програму који извршава следећи код.

```
16
17 int prekidac = 0;
18 int flag = 1;
19
20 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
21 {
22     if(prekidac == 0) // pali se radiator
23     {
24         prekidac = 1;
25         flag = 0;
26         //counter= 0;
27         HAL_GPIO_WritePin(GPIOB, GPIO_PIN_10, GPIO_PIN_RESET);
28         HAL_GPIO_WritePin(GPIOB, GPIO_PIN_11, GPIO_PIN_SET);
29     }
30     else // gasi se radiator
31     {
32         HAL_GPIO_WritePin(GPIOB, GPIO_PIN_10, GPIO_PIN_SET);
33         HAL_GPIO_WritePin(GPIOB, GPIO_PIN_11, GPIO_PIN_RESET);
34         prekidac = 0;
35         flag = 1;
36     }
37 }
38
```

Слика 3. Функција која је задужена за прекид када се притисне дугме

На почетку програма променљива prekidac има вредност 0, што значи када се први пут дугме притисне, радијатор се пали, симулирано зеленом диодом која је повезана на микроконтполер преко порта B на пин 10. Та зелена диода се укључује као и малопре описан екран и радијатор креће са радом. Поред зелене ту је и црвена диода која симулира угашен систем (на почетку извршавања програма оно је укључена).

У главом делу програма у главној беспоначној петљи `while(1)` се налази следећи код:

```
190     while(1)
191     {
192         if(flag == 1)
193         {
194           LCD(0x01,0); // cistimo ekran
195           while(flag);
196         }
197
198       HAL_Delay(100);
199       HAL_ADC_Start(&hadc1);
200       HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
201       rez = HAL_ADC_GetValue(&hadc1);
202
203       zadataTemperatura = (rez*18)/4095 + 16;
204       HAL_Delay(1000);
205       HAL_ADC_Stop(&hadc1);
206
207       LCD(0x01,0); // cistimo ekran
208
209       int i = 0;
210       sprintf(buffer,"%d",zadataTemperatura);
211       while(temp[i])
212       {
213           LCD(temp[i],1);
214           i++;
215       }
216       i=0;
217       while(buffer[i])
218       {
219           LCD(buffer[i],1);
220           HAL_Delay(10);
221           i++;
222       }
223       HAL_Delay(500);
224
225       //citamo temperaturu sa senzora
226       HAL_Delay(100);
227       HAL_ADC_Start(&hadc1);
228       HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
229       rez = HAL_ADC_GetValue(&hadc1);
230       trenutnaTemperatura = (rez*45)/4095;
231
232       trenutnaTemperatura = 23; // recimo da smo procitali 23
233       LCD(0x01,0); // cistimo ekran
```

Слика 4. Први део петље у главном делу програма

Први део петље у главном делу програма почиње тако што смо убацили код у бесконачно чекање. То бесконачно чекање се прекида када корисник покрене систем притиском на дугме. Променљива flag има на почетку вредност 1 и та вредност се мења тек

покретањем функције за прекид (Слика 3.). Након тога програм прво чита вредност са термостата (у нашем случају симулације то је потенциометар). Ту се налази температура коју је корисник задао радијатору да достигне и одржава. Минимална температура је 16 а максимална 34 степени докле радијатор може грејати. Затим се та задата температура исписује на екран, са следећом поруком: „Задата Темп: X“, где је X прочитана температура са термометра. Након тога систем мери тренутну температуру просторије и смешта је у променљиву trenutnaTemperatura (због ограниченог броја пинова који раде у програму Протеус, ова температура је ручно постављена на 23 степена целзијуса).

```
231
232         trenutnaTemperatura = 23; // recimo da smo procitali 23
233         LCD(0x01,0); // cistimo ekran
234         i=0;
235         while(trenutna[i])
236         {
237             LCD(trenutna[i],1);
238             i++;
239         }
240         sprintf(buffer,"%d",trenutnaTemperatura);
241         i=0;
242         while(buffer[i])
243         {
244             LCD(buffer[i],1);
245             HAL_Delay(10);
246             i++;
247         }
248
249         HAL_Delay(1000);
250
251         //pitamo da li je veca temp od zadate
252         if(trenutnaTemperatura >= zadataTemperatura)
253         {
254             LCD(0x01,0);
255             flag = 1;
256             i = 0;
257             while(gasi[i])
258             {
259                 LCD(gasi[i],1);
260                 i++;
261             }
262             HAL_Delay(400);
263             counter = 9;
264             while(counter > 0)
265             {
266               sprintf(buffer,"%d",counter);
267               HAL_Delay(200);
268               LCD(0x01,0);
269               LCD(buffer[0],1);
270               counter--;
271             }
272             HAL_GPIO_WritePin(GPIOB, GPIO_PIN_10, GPIO_PIN_SET);
273             HAL_GPIO_WritePin(GPIOB, GPIO_PIN_11, GPIO_PIN_RESET);
274             prekidac = 0;
275         }
276
277     }
278 }
```

Слика 5. Други део петље у главном делу програма

Потом се тренутна температура собе исписује на екран. И последњи део while петље се односи на проверу да ли је тренутна температура просторије већа или једнака од задате температуре на термостату. Уколико је мања, радијатор наставља са грејањем и исписивањем тренутне и задате температуре у просторији, док уколико је једна или виша, радијатор исписује на екрану поруку о гашењу која гласи „Gasim se...“ и почиње са одбројавањем (10секунди) до гашења, након чега се и угаси.

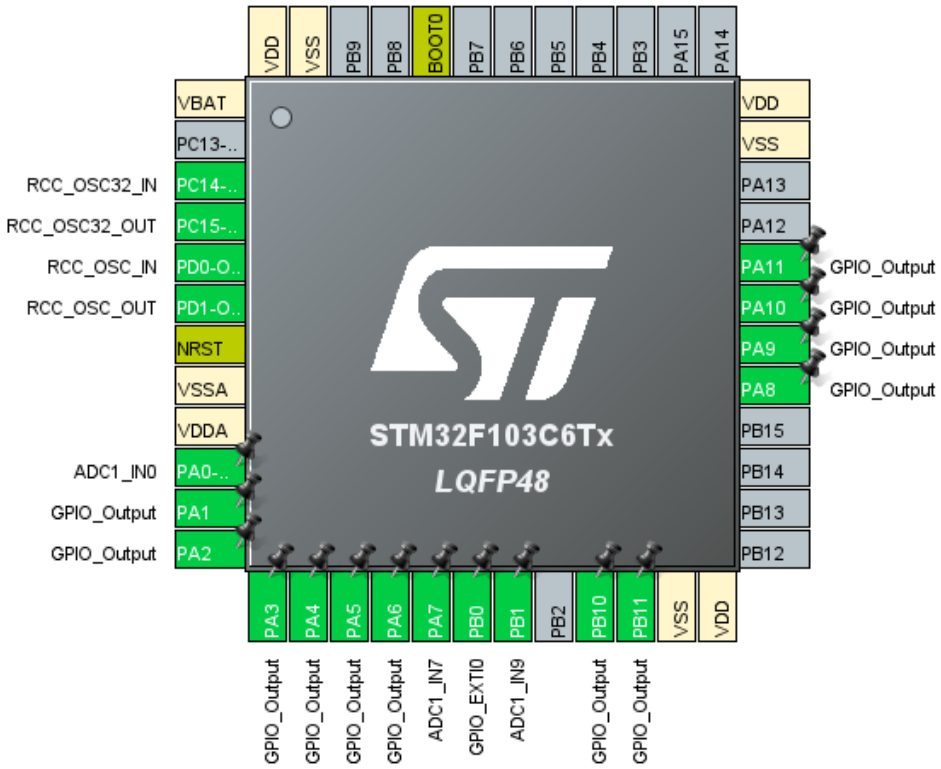
Овај радијатор поседује и тајмерски прекид, што значи да на одређено време проверава да ли треба укључити/искључити радијатор. Он се активирао у подешавањима које ћемо ускоро описати, имплементацијом функције:

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)

као и дозвољавањем његовог коришћења следећом линијом кода у главном делу програма изнад главне while(1) петље:

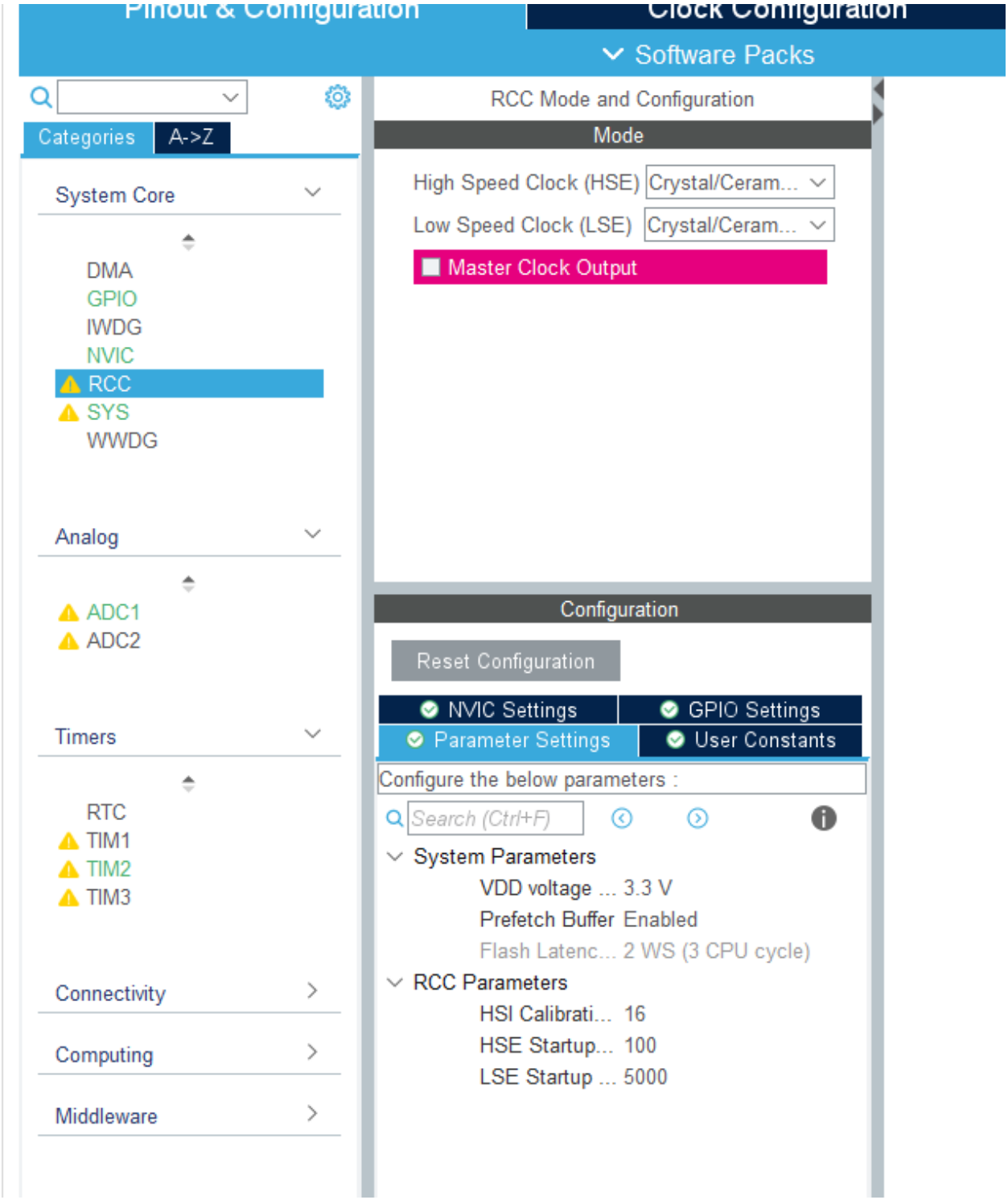
HAL_TIM_Base_Start_IT(&htim2);

Подешавања везана за сам хардвер микроконтролера смо обавили на следећи начин:



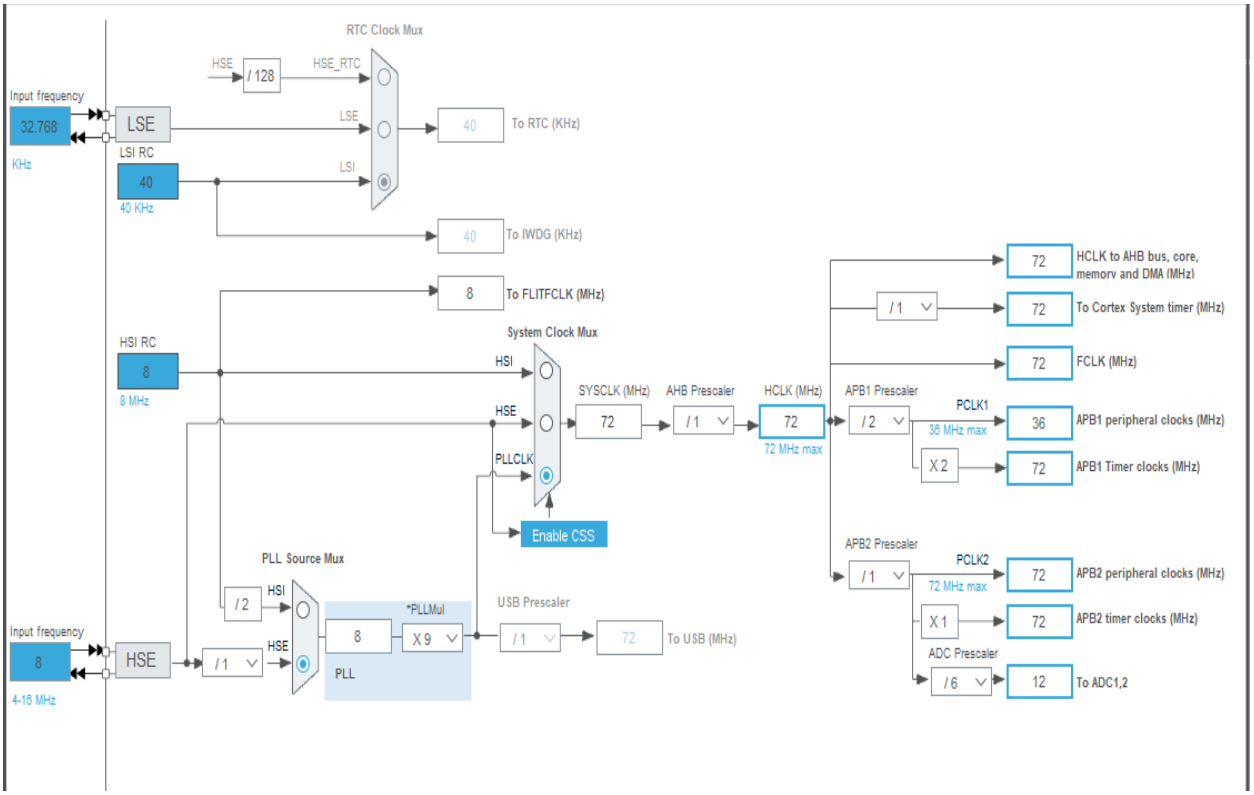
Слика 6. Поглед дефинисаних пинова за микроконтролер

На самом почетку ћемо истакнути да смо користили екстерни клок, због тајмера и то се може видети на слици испод:



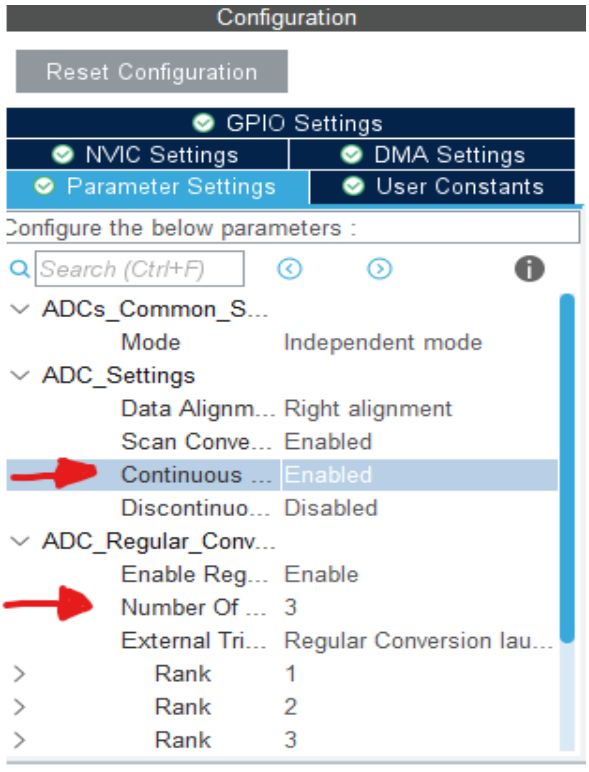
Слика 7. RCC Mode and Configuration

Радни такт износи максималних 72MHz за овај процесор, као и за ABP1 Timer.



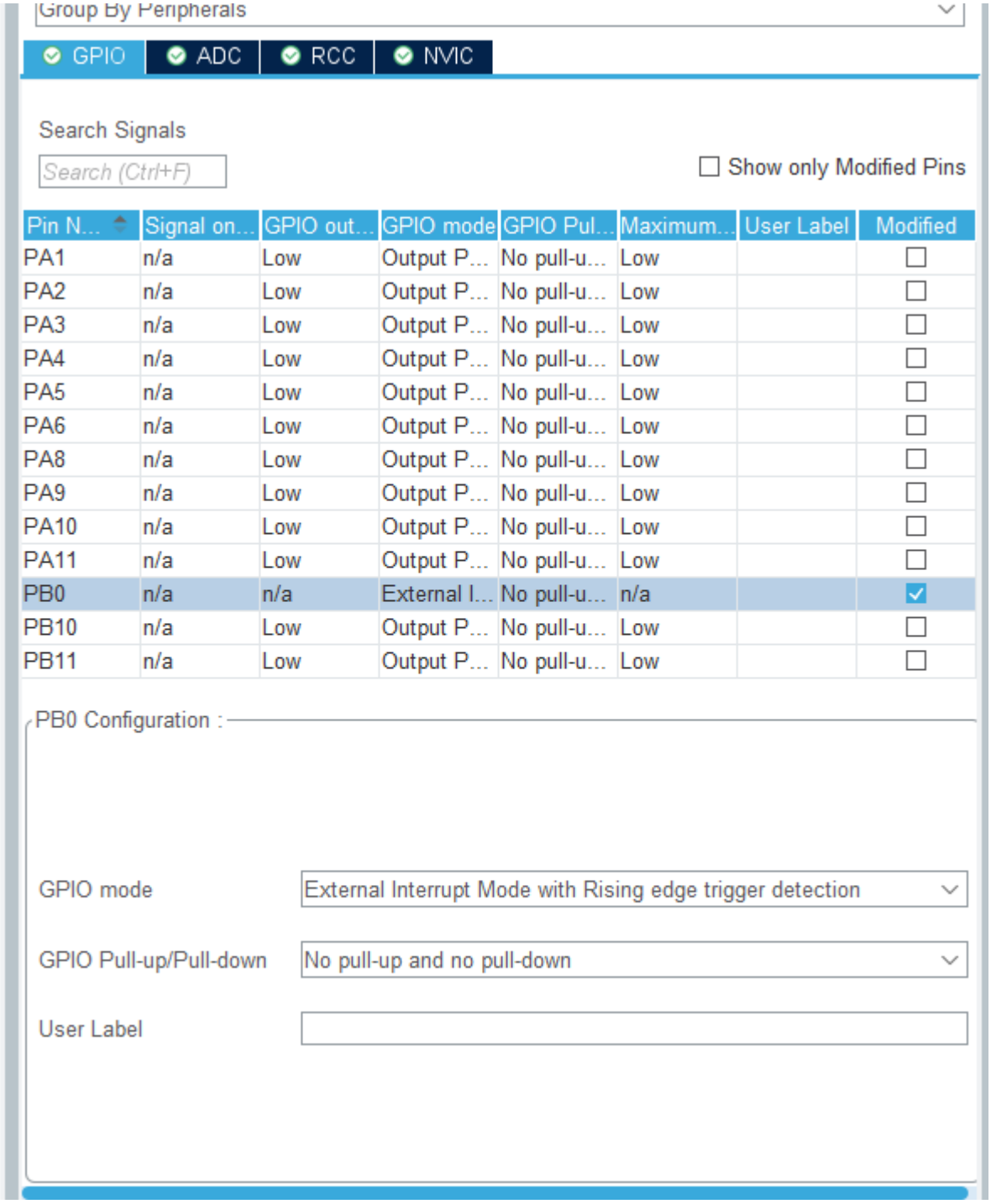
Слика 8. Clock Configuration

За аналогно-дигиталну конверзију смо користили три пина, PA0, PB1 и PA7. У подешавањима смо дозволили сталну конверзију и број конверзија наместили на три.



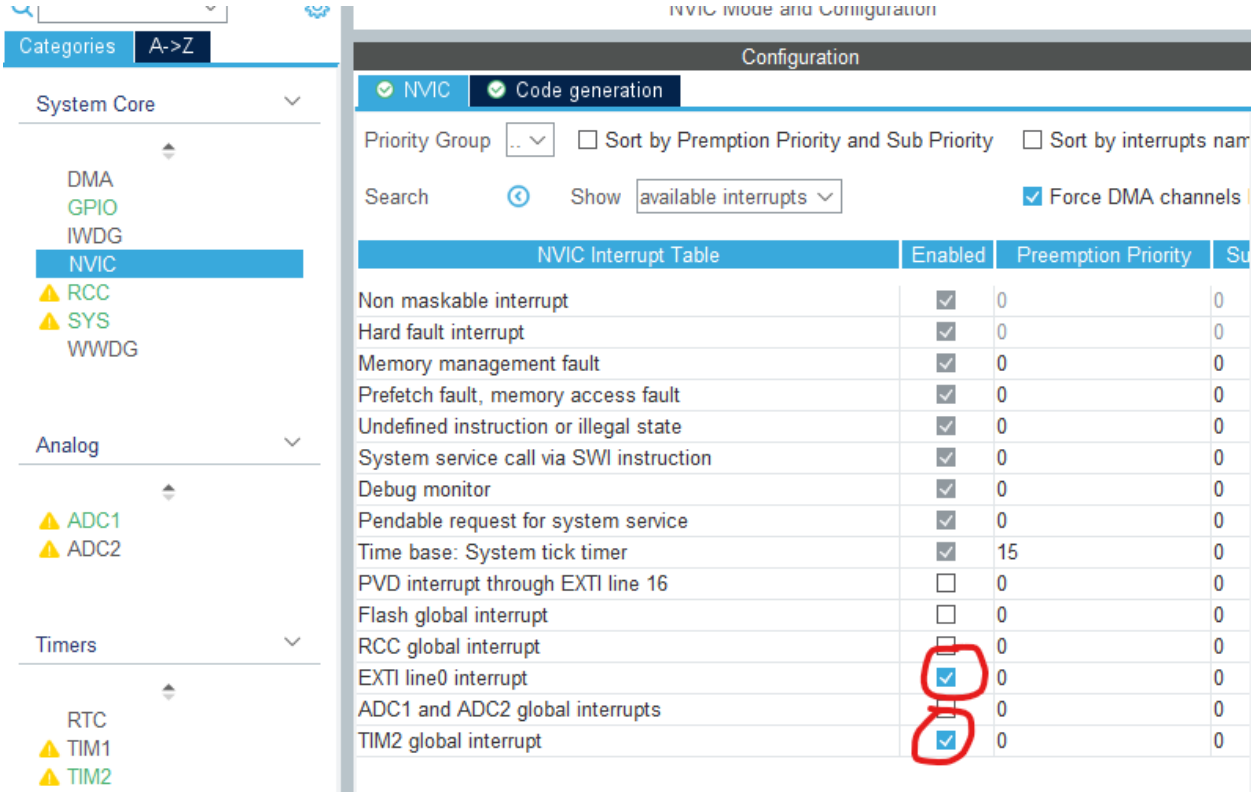
Слика 9. Конфигурација за АДЦ

За наше On/Off дугме смо користили пин 0 на порту Б.



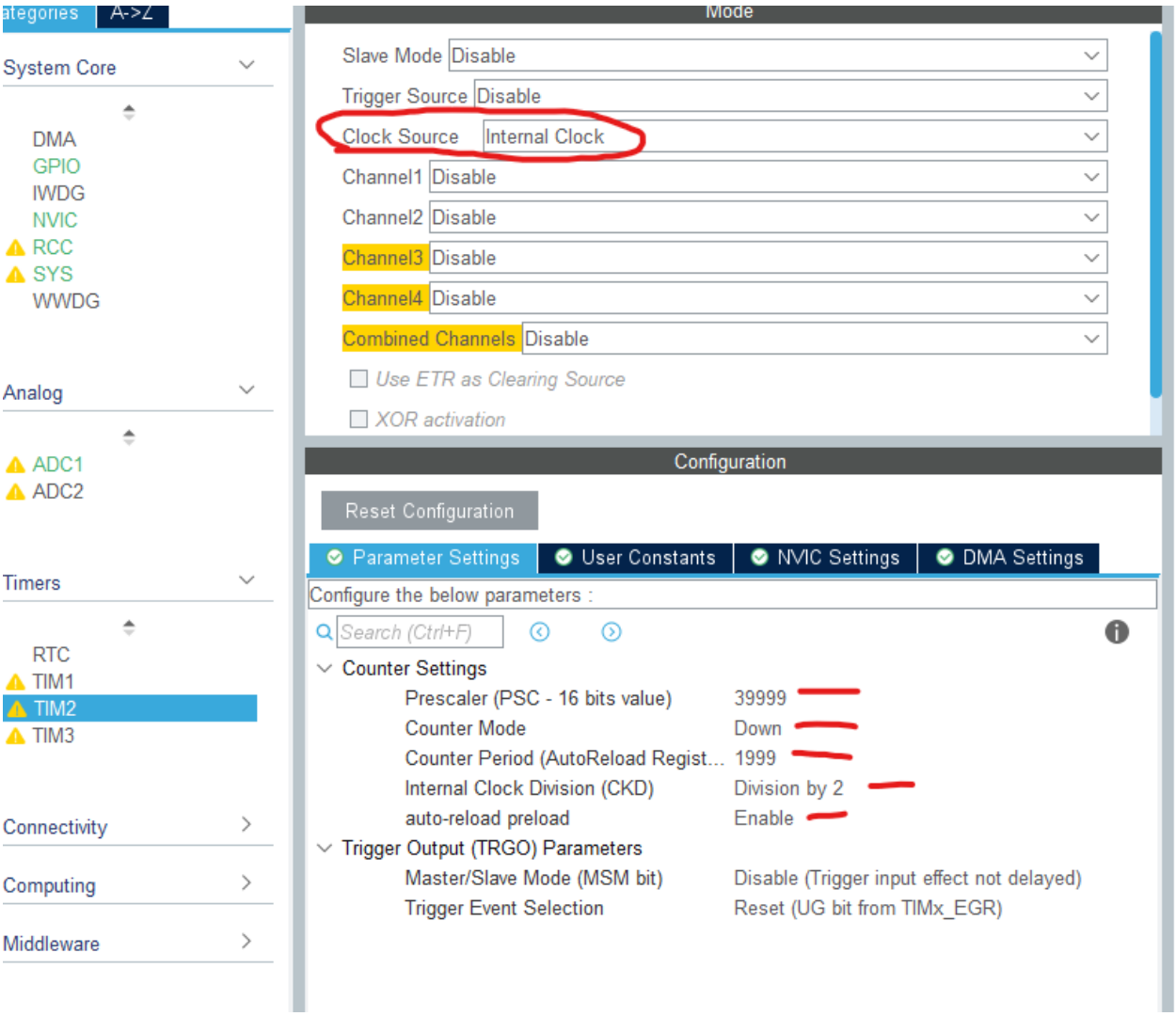
Слика 10. Конфигурација за дугме које је задужено за укључивање/искључивање

Такође смо у подешавањима дозволили и прекид за дугме као и за тимер.



Слика 11. Конфигурација за оба прекида

Што се тиче временског прекида, ту смо поставили следеће вредности:



Слика 12. Конфигурација за временски прекид

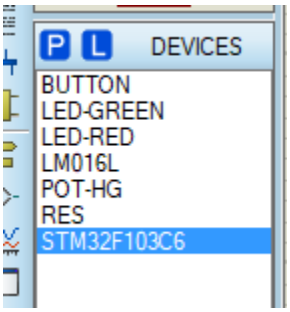
Clock Source смо поставили на Internal Clock, док смо Prescaler поставили на 39999 а Counter Period на 1999. Додатно смо изменили још три подешавања која су обележена на слици 12.

На овај начин смо добили временски прекид на сваких 1,1111 секунду.

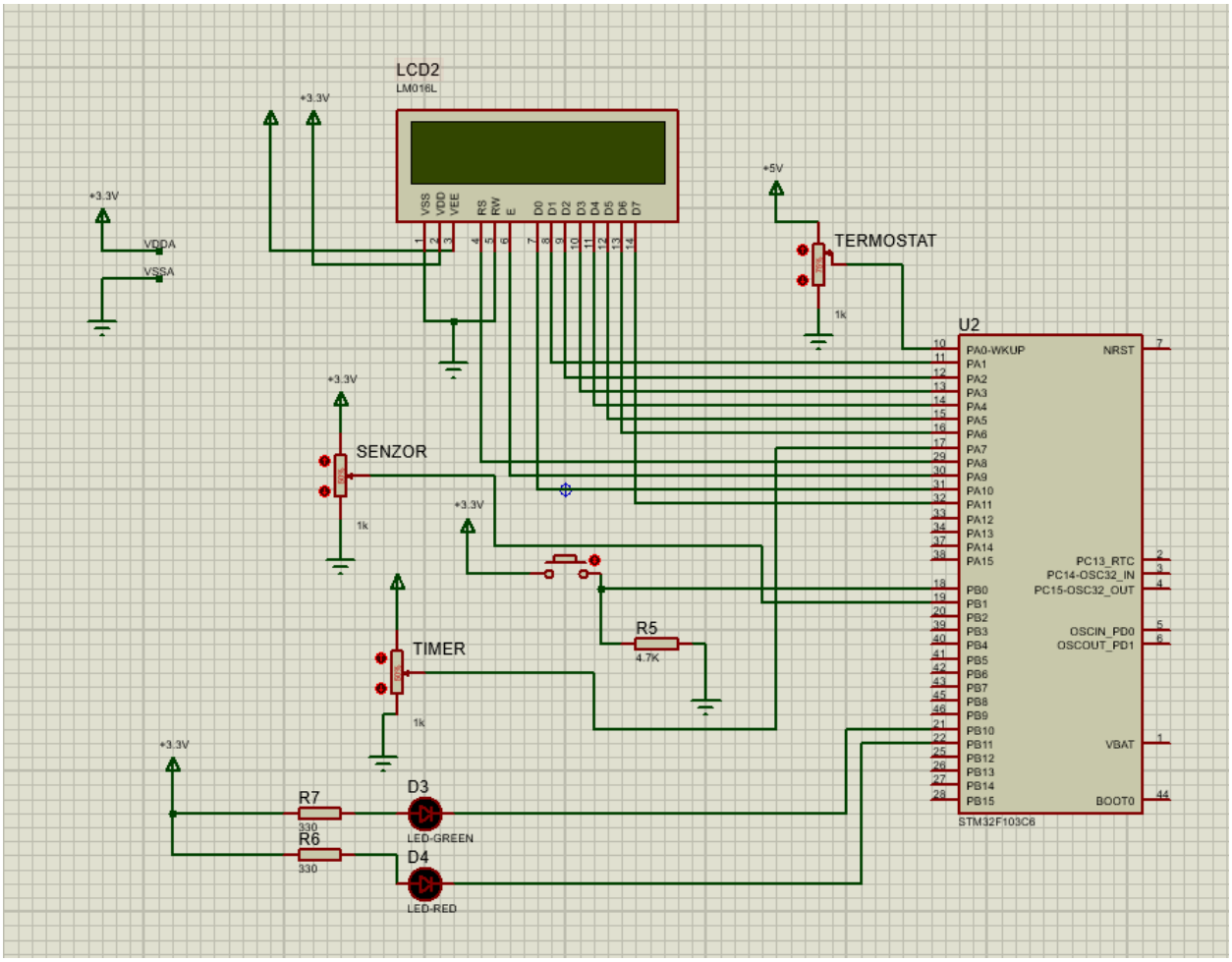
3. Симулација програма у Proteus 8 програму

Од електричних компоненти смо користили следеће:

1. Једно дугме
2. Лед диоде (црвену и зелену)
3. Три отпорника (два од 330 ома и један од 4,7к)
4. Три потенциометра (симулирају сензор топлоте, термостат и тајмер)
5. Један LCD
6. STM32F103C6 плочу



Слика 13. Компоненте



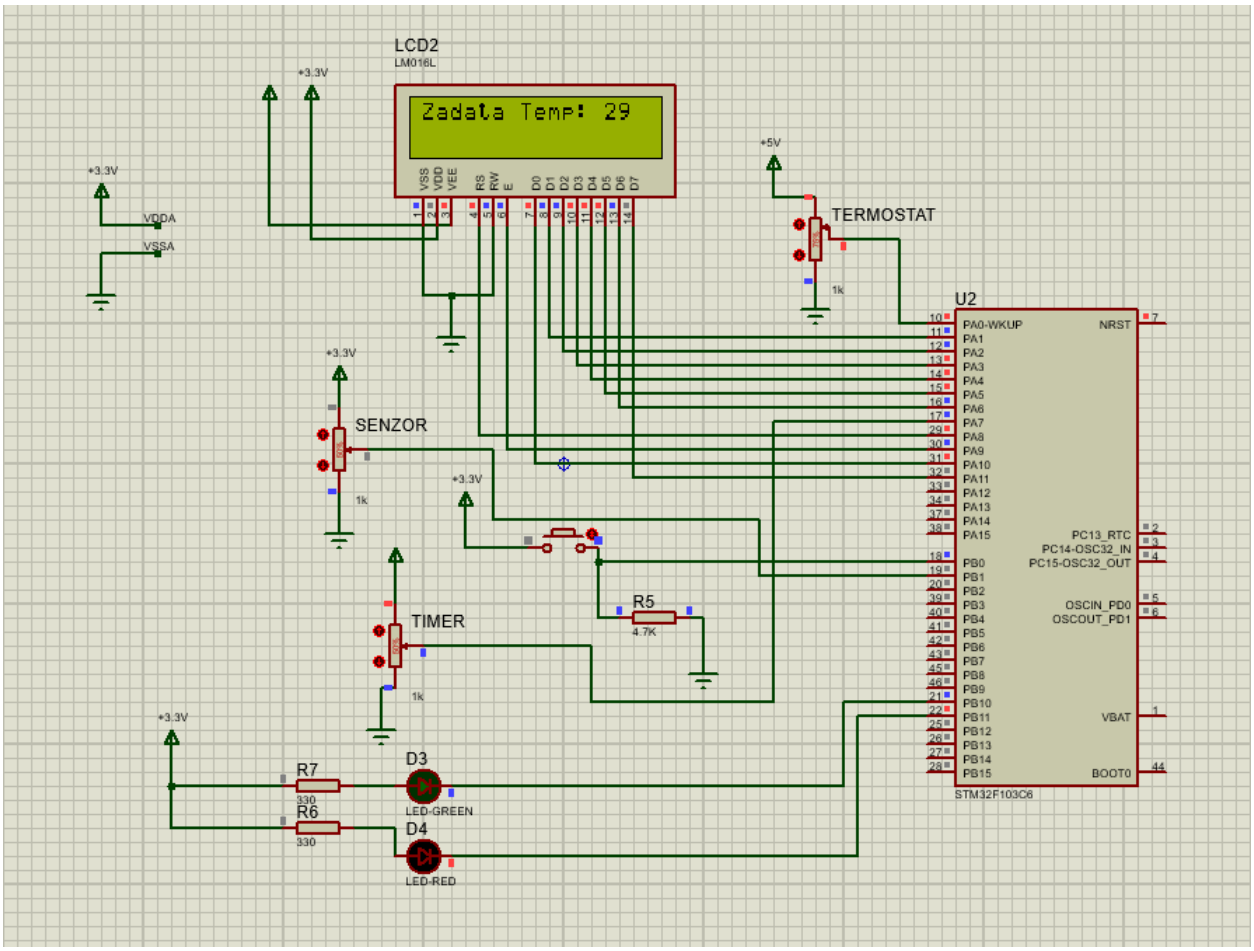
Слика 14. Изглед електричне шеме

4. Корисничко упутство

При самом покретању симулације, радијатор је искључен док светли црвена лед диода.

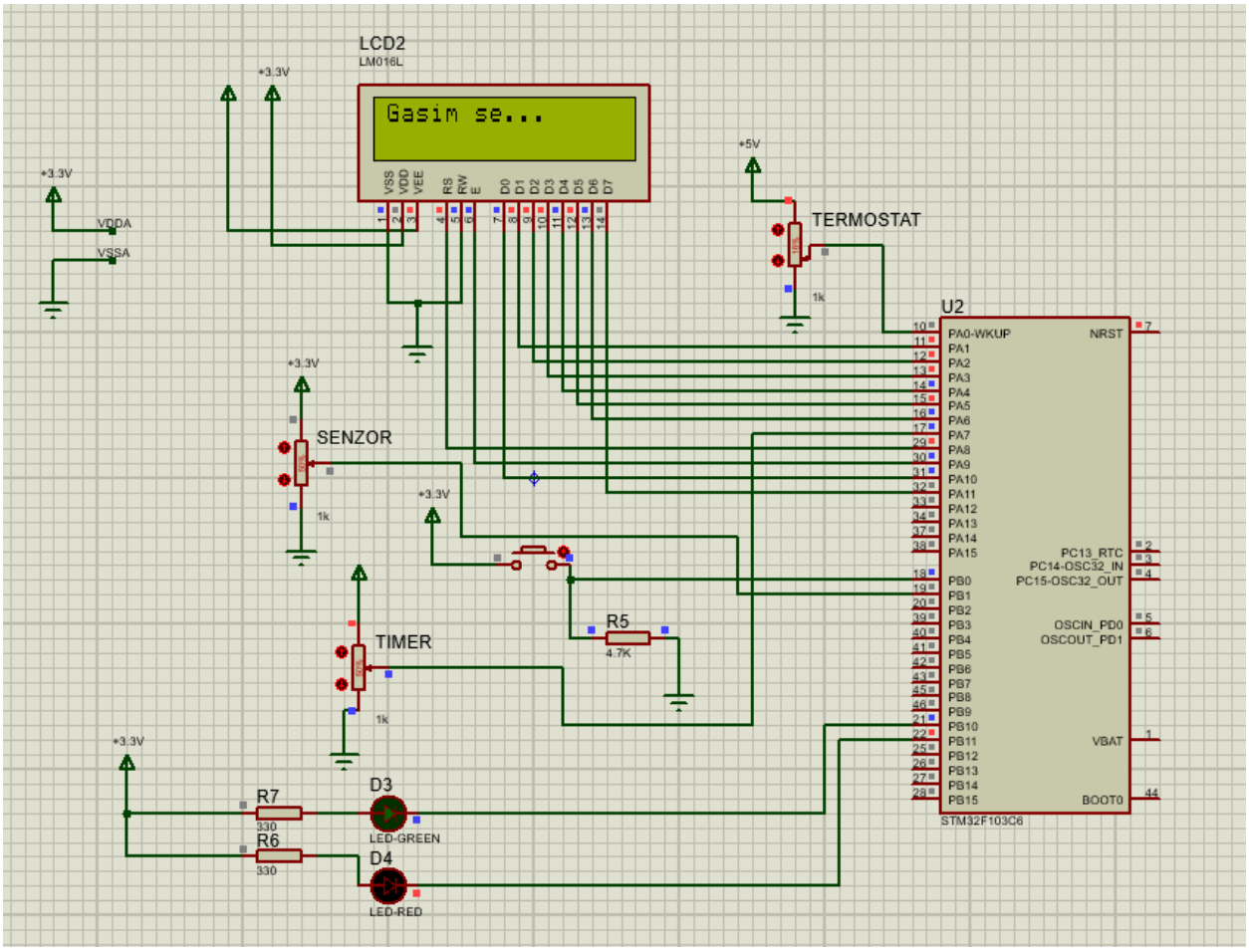
Радијатор се покреће као што смо већ и описали притиском на дугме које је повезано на PB0.

Након притиска на дугме, радијатор очитава задату температуру на термостату као и тренутну температуру са сензона и исписује их на екран.



Слика 15. Испис задата температуре са термостата

Уколико радијатор достигне жељену температуру, исписује поруку за гашење и након 10 секунди се гаси.



Слика 16. Радијатор је достигао жељену температуру

На самом крају, корисник има могућност да подеси који жељени дан у недељи жели да радијатор буде укључен тако што ће на потенциометру ТИМЕР да подеси жељени опсег. У зависности од подешеног радијатор ће се палити у то време уколико је тада искључен.

НАПОМЕНА: Због ограничења тимер интерупта у Протеусу, ову могућност није могуће визуелно имплементирати. Остале споменуте могућности се могу тестирати уколико се искључе опције за тимер прекид у самом STM32CubeIDE програму.