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:

Јован Радовановић 85/2018

Професор:

др Александар Пеулић

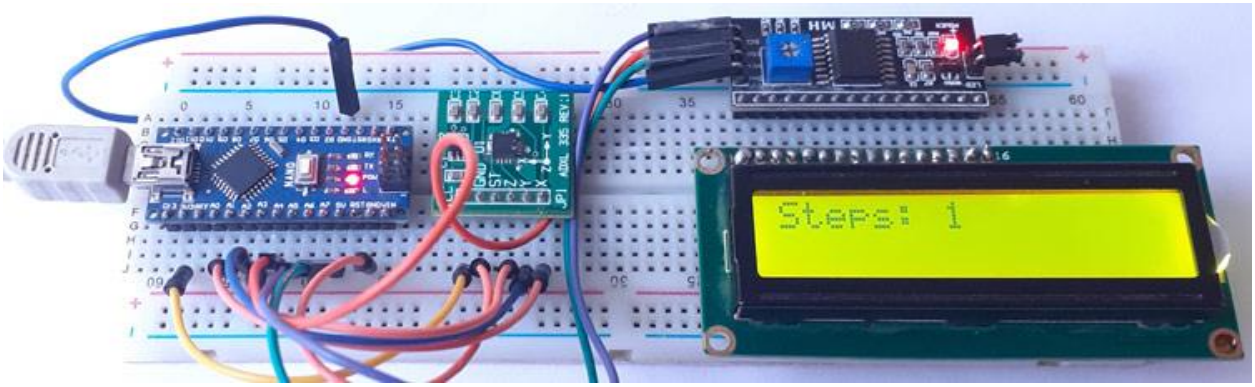
Фебруар 2023.

Садржај

1. Увод

Потреба за бројачем корака појавила се из жеље корисника да прате статистике својих свакодневних активности у циљу побољшања здравља или личне евиденције. Циљ овог пројекта јесте омогућити корисницима да на једноставан начин сазнају пређени број корака за одређени интервал мерења.

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. Педометар конструисан на ардуино плочи

2. Објашњавање алгоритма и кода

Пројекат садржи 1 LCD дисплеј који се користи за приказ пређеног броја корака, 2 потенциометра који симулирају сензор убрзања и жироскоп, 1 дугмета које се користи за симулацију помераја ноге корисника (симулира потенциометар жироскопа).

Функције везане за ЛЦД: LCD_Init и LCD. LCD_Init функција служи за иницијализацију екрана. Као што је на пример чишћење екрана, подешавање курсора на почетну позицију за неки испис, итд. LCD функција се користи за испис одређених вредности на ЛЦД екрану.

```
145 void LCD_init()
146 {
147     LCD(0x38, 0); //2 lines, 5*7 matrix
148     LCD(0x0C, 0); //Display on, cursor off
149     LCD(0x06, 0); //Increment cursor (shift to right)
150     LCD(0x01, 0); //Clear display screen
151     LCD(0x80, 0); //Forse cursos to beginning (1st line)
152 }
153
154 void LCD(uint8_t val_1, uint8_t cmd)
155 {
156     uint8_t data1;
157
158     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_9, cmd); // set RS = cmd; (cmd=0)=>Command; (cmd=1) => data
159
160     data1 = val_1 & 0x01;
161     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, data1);
162
163     data1 = (val_1 >> 1) & 0x01;
164     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, data1);
165
166     data1 = (val_1 >> 2) & 0x01;
167     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, data1);
168
169     data1 = (val_1 >> 3) & 0x01;
170     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, data1);
171
172     data1 = (val_1 >> 4) & 0x01;
173     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_5, data1);
174
175     data1 = (val_1 >> 5) & 0x01;
176     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_6, data1);
177
178     data1 = (val_1 >> 6) & 0x01;
179     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_7, data1);
180
181     data1 = (val_1 >> 7) & 0x01;
182     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, data1);
183
184         //Enable
185     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_10, GPIO_PIN_SET);
186     HAL_Delay(5);
187     HAL_GPIO_WritePin(GPIOA, GPIO_PIN_10, GPIO_PIN_RESET);
188 }
```

Слика 2. LCD функције у main.c

Дугме користимо за симулацију жироскопа. Када је дугме притиснуто знамо да је корисник направио корак који је већи од 20 степени, што је минимални угао којим би препознали корак корисника. Жироскоп је такође могуће прецизније симулирати мерењем вредности са потенциометра(преко 10% се рачуна као угао довољан за корак).

Целокупна логика програма смештена је у HAL_GPIO_EXTI_Callback функцију која се активира на сваки притисак тастера и симулира довољан угао жироскопа за рачунање корака корисника. Унутар функције проверава се и вредност сензора убрзања која се очитава са потенциометра који симулира убрзање.

HAL_GPIO_EXTI_Callback функција се активира на сваки притисак дугмета. Да би корак успешно био направљен и број корака увећан потребно је да се пре тога провери које је убрзање корисника. Убрзање меримо са потенциометра који симулира сензор убрзања и

минимална вредност јесте 31% вредности потенциометра што је еквивалентно убрзању 1 метар по секунди. Ако су минимални услови задовољени, кориснику се додаје корак.

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    if(IzracunatoUbrzanje >= 2.0)// preko 31% na potenciometru
    {
        brojKoraka++;
    }
}
```

Слика 3. Функција која је задужена за прекид при притиску дугмета

Почетне вредности су: Укупан број корака 0, ИзрачунатоУбрзање 0. На основу промена стања потенциометра који симулира сензор убрзања и притиска дугмета, ове вредности се увећавају или смањују у даљем коду.

У главом делу програма налази следећи код:

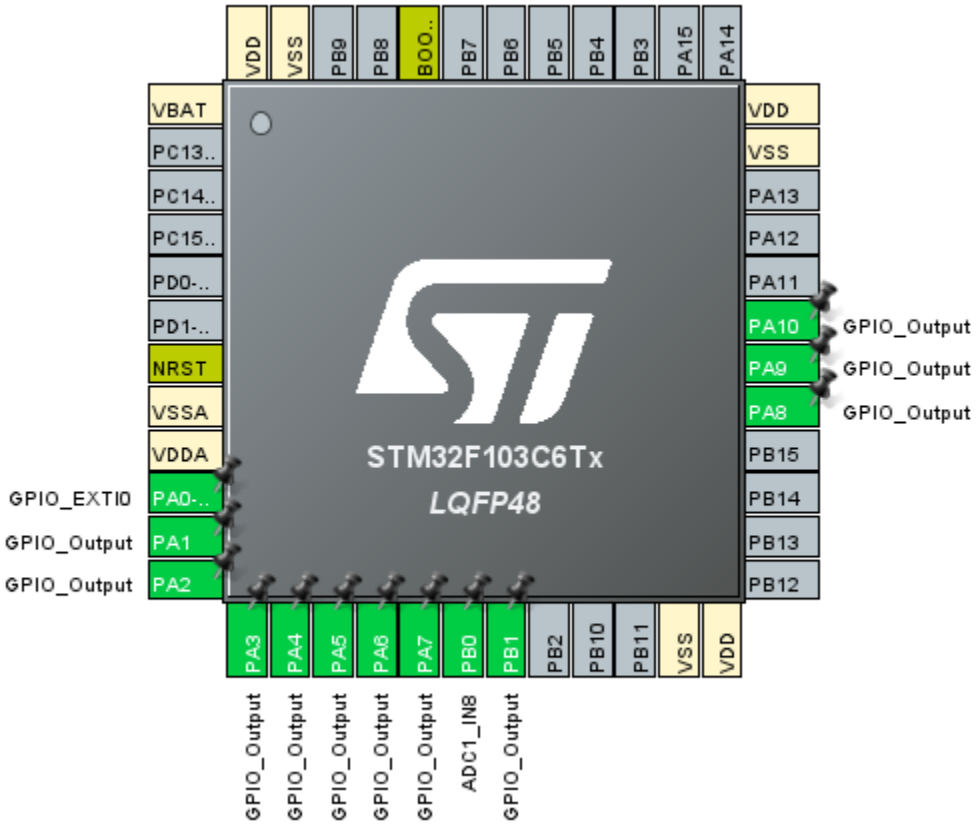
```c
int main(void)
{

  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_ADC1_Init();
  LCD_init();
  float res,rezultat;

  char stringBrKoraka[] = "Broj koraka: ";
  char buffer[10];

  while (1)
  {
          HAL_Delay(100);
          HAL_ADC_Start(&hadc1);
          HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
          res = HAL_ADC_GetValue(&hadc1);

          HAL_Delay(100);


          IzracunatoUbrzanje = (res*10)/4095;

          HAL_ADC_Stop(&hadc1);
          LCD(0x01, 0); /* cistimo ekran */

          int i = 0;

          while(stringBrKoraka[i])
          {
              LCD(stringBrKoraka[i],1);
              i++;
          }
          sprintf(buffer,"%d",brojKoraka);
          i=0;
          while(buffer[i]){
              LCD(buffer[i],1);
              i++;
          }

          LCD('  ',1);
          HAL_Delay(200);

  }
  /* USER CODE END 3 */
}
```

Слика 4. main.c главни део програма

Програм се одвија у бесконачној петљи. Прво покрећемо АДЦ конверзију за потенциометар, меримо вредности са потенциометра убрзања и претварамо у реалан број у интервалу између 0 и 10. Након тога заустављамо АДЦ конверзију и чистимо ЛЦД дисплеј. Затим исписујемо текст на ЛЦД екрану “Број корака је: ” и након тога целобројну вредност промењљиве бројКорака. Програм заустављамо на кратак временски период и након тога бесконачна петља поново обавља исти редослед операција.

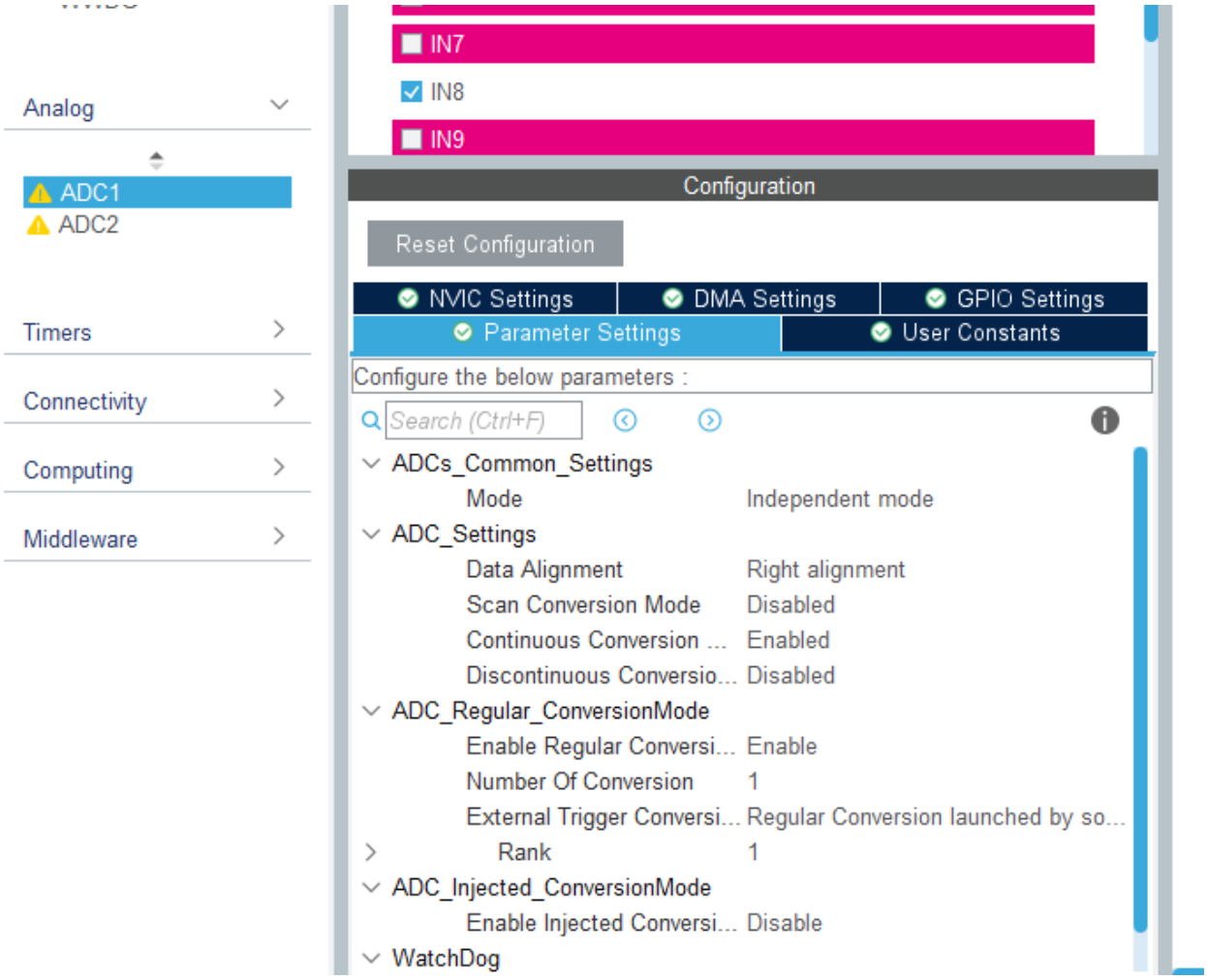
Дакле, main.c функција нашег програма нам служи да прочитамо вредност са потенциометра помоћу АДЦ конверзије и да на ЛЦД екрану прикажемо одговарајући текст и број корака које је корисник прешао за интервал од покретња програма до садашњег тренутка. Пошто су вредности бројКорака и ИзрачунатоУбрзање глобалне, функција HAL_GPIO_EXTI_Callback , која се активира на притисак дугмета које симулира жироскоп, проверава да ли су вредности са сензора убрзања и жироскопа добре. У случају да су услови за минимални корак задовољени(20 степени на жироскопу и 1 мс убрзања) увећава се вредност укупних корака корисника, у супротном се ништа не дешава.

Подешавања везана за хардвер микроконтролера:



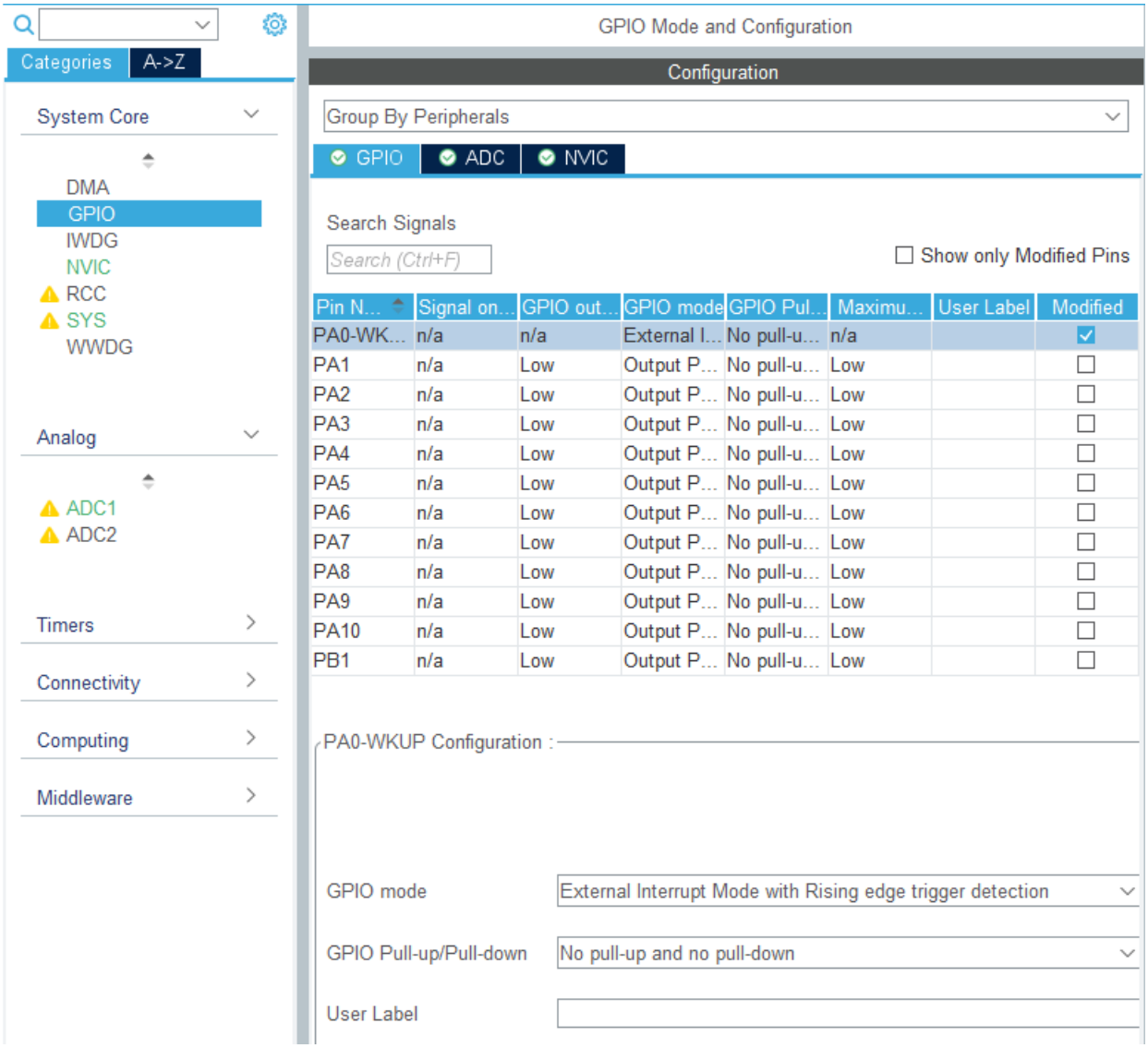
Слика 5. Поглед дефинисаних пинова за микроконтролер

За аналогно-дигиталну конверзију смо користили три пина, PB0. У подешавањима смо дозволили сталну конверзију.



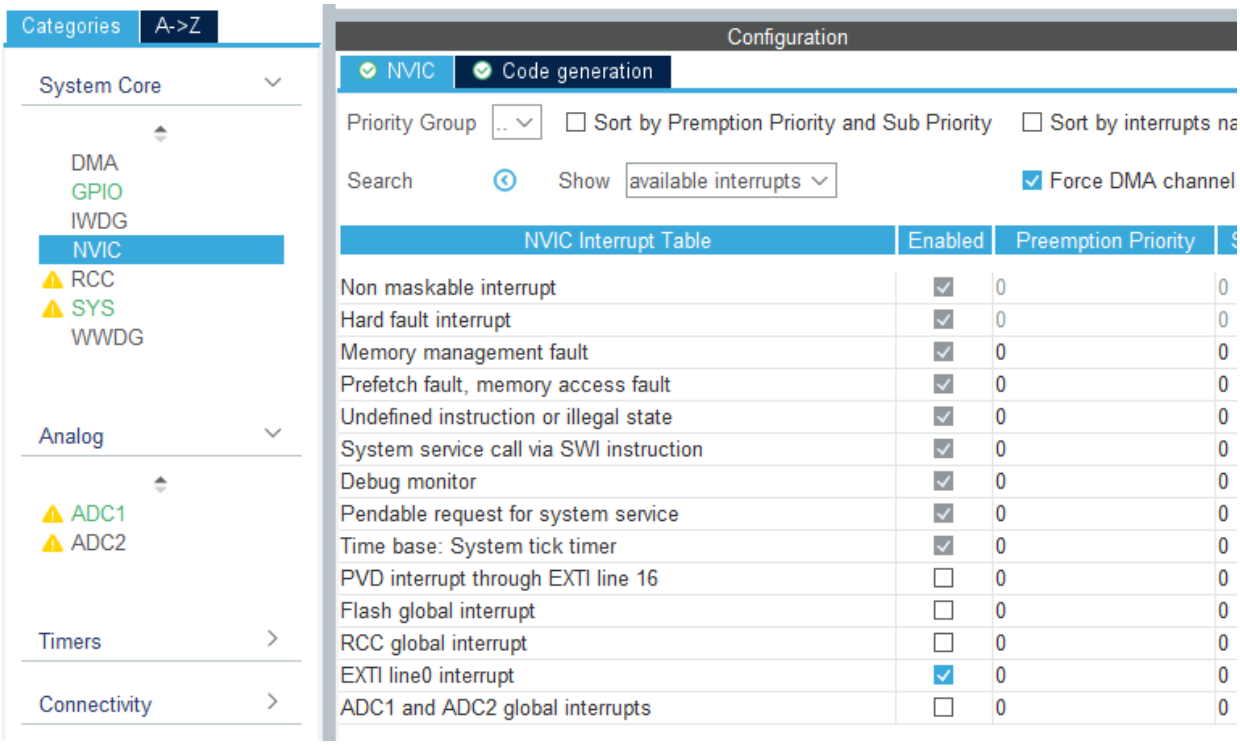
Слика 6. подешавање АДЦ конверзије

Дугме коришћено за симулацију жироскопа на пину 0 порта А подесили смо овако



Слика 7. подешавање дугмента симулације жироскопа

Такође смо у подешавањима дозволили и прекид за дугме.

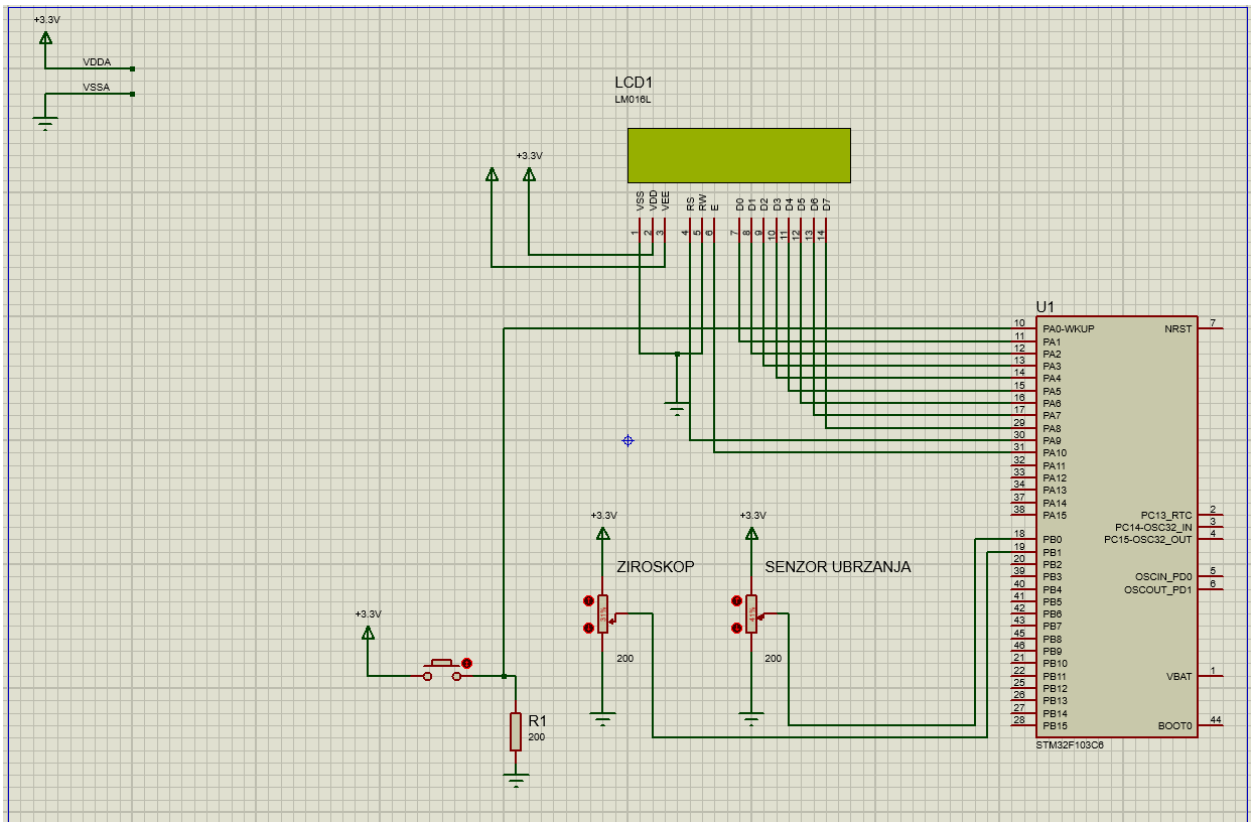


Слика 8. подешавање прекида

3. Симулација програма у Proteus 8

Од електричних компоненти смо користили следеће:

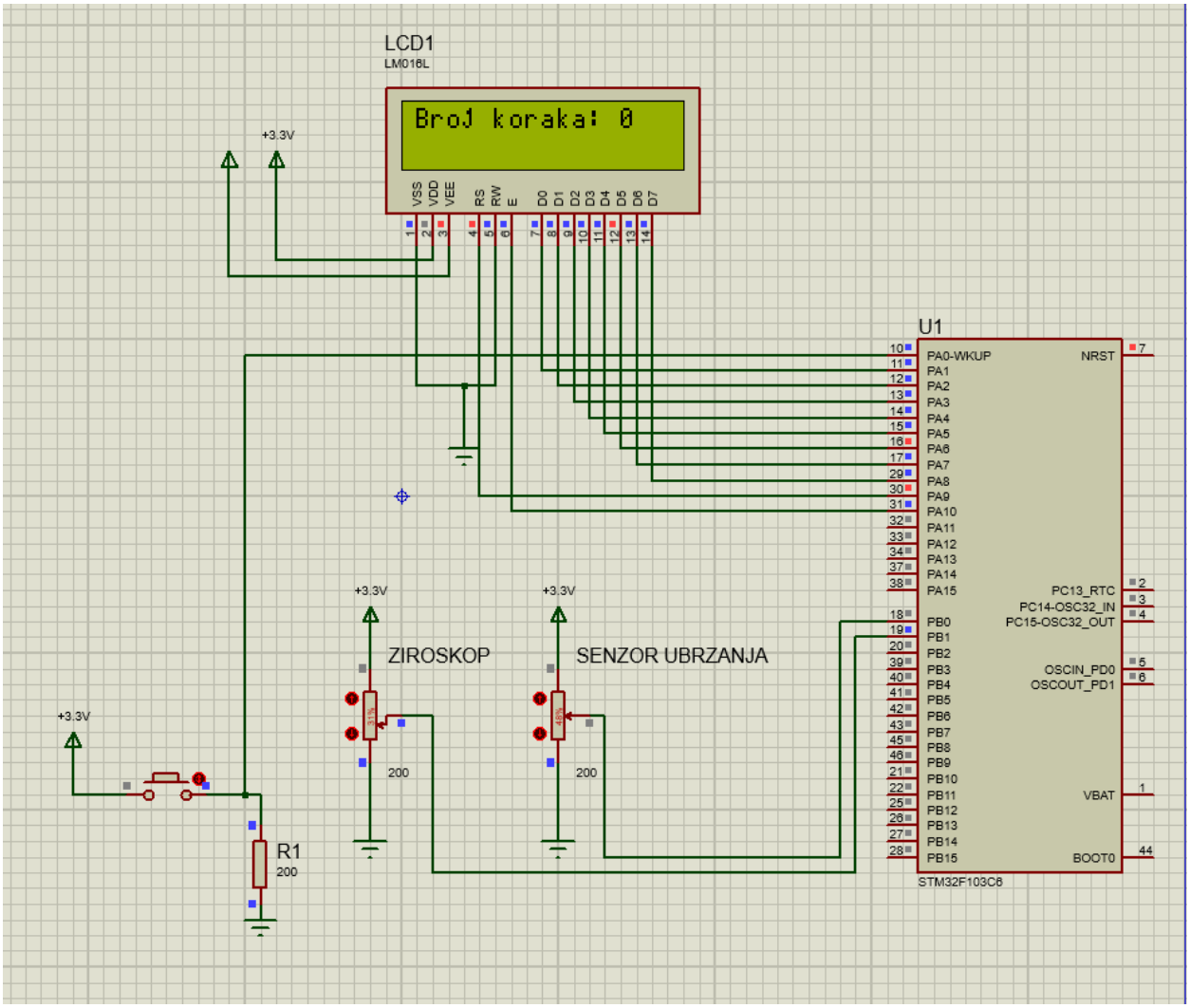
1. Једно дугме (симулира жироскоп)
2. Три отпорника (200 ома)
3. Два потенциометра (симулира сензор убрзања и жироскоп)
4. LCD
5. STM32F103C6 плочу



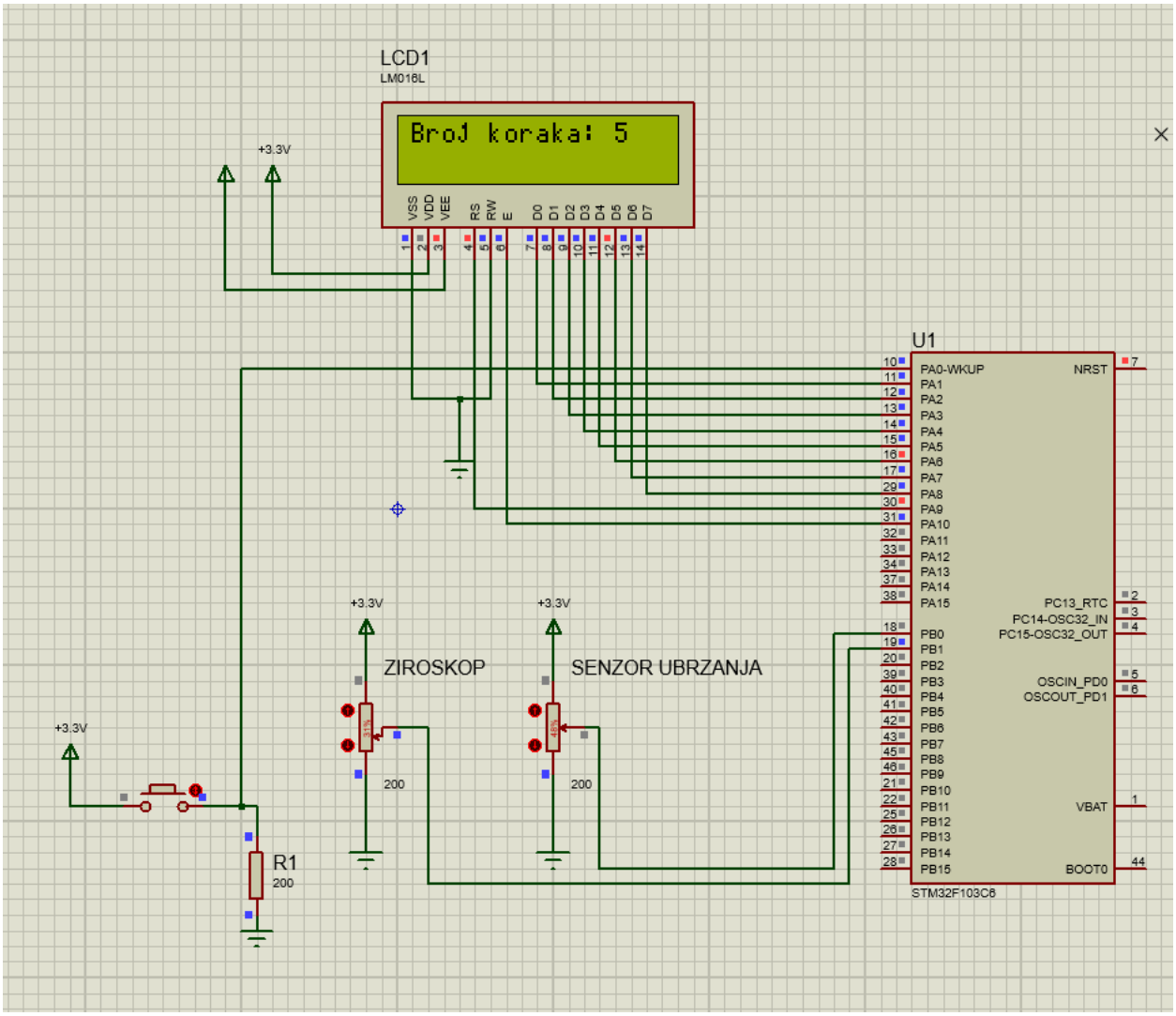
Слика 9. Шема свих компоненти на плочи

4. Корисничко упутство

При покретању симулације број корака који је очитан је 0. Када сензор убрзања померимо за вредност већу од 31% и притиснемо дугме број корака ће бити увећан.



Слика 10. Почетно стање симулације



Слика 11. Стање симулације након начињених 5 корака