UNIVERZITET U BEOGRADU
ELEKTROTEHNIČKI FAKULTET

Nataša Dimitrijević, 3375/2012

Merač temperature i relativne vlažnosti vazduha sa alarmom

Mentor:
Dr Dragan Vasiljević

Beograd, februar 2013.

SADRŽAJ

Slike

Tabele

1 UVOD

Zamisao predmetnog projekta jeste da se izvrši simulacija i testiranje zadatka na mikrokontroleru iz familije STM32 proizvođača ST Mi-croelectronics. Ovi 32-bitni mikrokontroleri su zasnovani na ARM Cortex-M3 jezgru koje odlikuje velika brzina rada (do 120 MHz procesorskog takta), veliki broj periferija, mala potrošnja, kao i poboljšan Thumb-2 instrukcijski set (koji obuhvata 16-bitne i 32-bitne instrukcije zajedno). Razvoj koda i testiranje je rađeno u softveru IAR Embeded Workbench for ARM 6.4.

Ideja ovog rada jeste da se napravi merač temperature i vlage vazduha sa alarmom. Da krajnji korisnik ne bi morao da poznaje programiranje mikrokontrolera, zadavanje kritične temperature omogućeno je preko računara. Podatak se mikrokontroleru prenosi preko usart-a. U te svrhe korišćen je modul USB-UART board firme Mikroelektronika. Senzor koji je korišćen jeste SHT11, firme Mikroelektronika, podaci su prikazani na LCD Board-u iste firme. Zvučni signal alarma obezbeđen je modulom BUZZ CLICK, već pomenute firme Mikroelektronika.

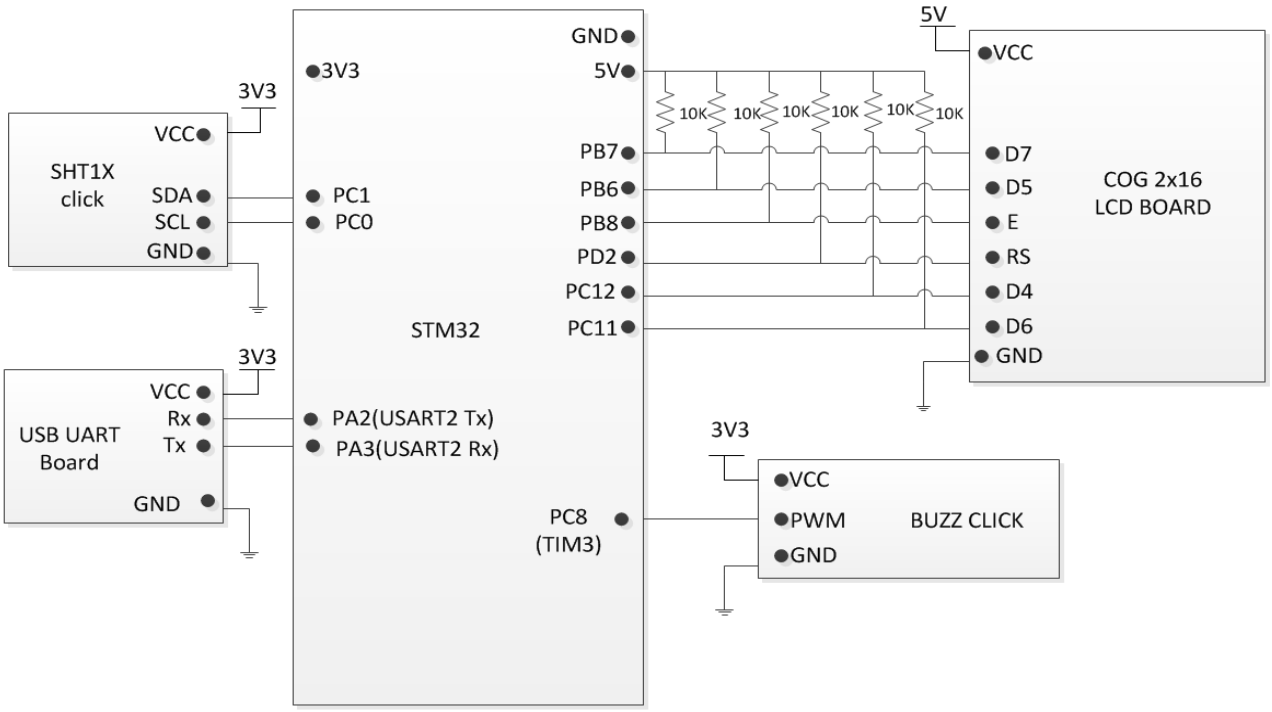
2 HARDVERSKA PLATFORMA

Kao osnova sistema korišćena je evaluciona ploča STM32VLDISCOVERY, proizvođača STMicroelectronics. Na njoj se nalazi mikrokontroler STM32F100RB, baziran na ARM Cortex-M3 jezgru. Napaja se preko USB-a, a može i sa 3.3V ili 5V eksternog napajanja. Poseduje i ST-Link debugger /programmer, preko koga se mikrokontroler programira i debaguje. Prikaz pomenute ploče, dat je na slici 1.
Mikrokontroler je zasnovan na ARM arhitekturi, sa 32-bitnom dužinom instrukcije. Radna frekvencija iznosi 24MHz, sa 128KB Flash memorije i 8KB SRAM-a. Sadrži tri USART periferije, jedan 12-bitni AD konvertor, dva 12-bitna DA konvertora, šest 16-bitnih tajmera opšte namene, kao i jedan sa naprednijim funkcijama. Nalazi se u 64-pinskom LQFP pakovanju.

Slika 1. Evaluciona pločica.

2.1 Realizacija sistema

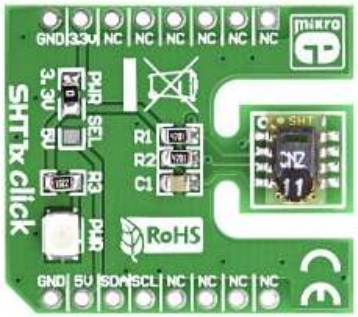
Na slici 2. prikazana je električna šema sistema. Od periferija su korišćeni tajmeri TIM2, TIM3 i USART2. Tajmer TIM2 radi u counter modu, pomoću njega se periodično generiše prekid i u prekidu se postavlja flag "meri" na jedinicu, čime startujemo merenje senzorom. Tajmer TIM3 radi u modu generisanja PWM signala. Ovim signalom pobuđujemo Buzz click board, odnosno uključujemo alarm. Periferiju USART2 koristimo za komunikaciju sa računarom. Komunikacija se obavlja preko modula USB UART board. Ideja za korišćenjem ovog modula jeste olakšano zadavanje granične temperature, nema potreba za stalnim učitavanjem celog programa na mikrokontroler, dovoljno je putem hiper terminala zadati novu graničnu temperaturu u zapisu STxxKR, pri čemu xx predstavlja novu temperaturu u stepenima celzijusa. Automatski ovaj podatak se prenosi preko uarta i postaje nova, validna granica. Modul SHT11 click na sebi ima senzor za merenje temperature i relativne vlažnosti vazduha.



Slika 2. Električna šema sistema.

2.2 SHT11 CLICK BOARD

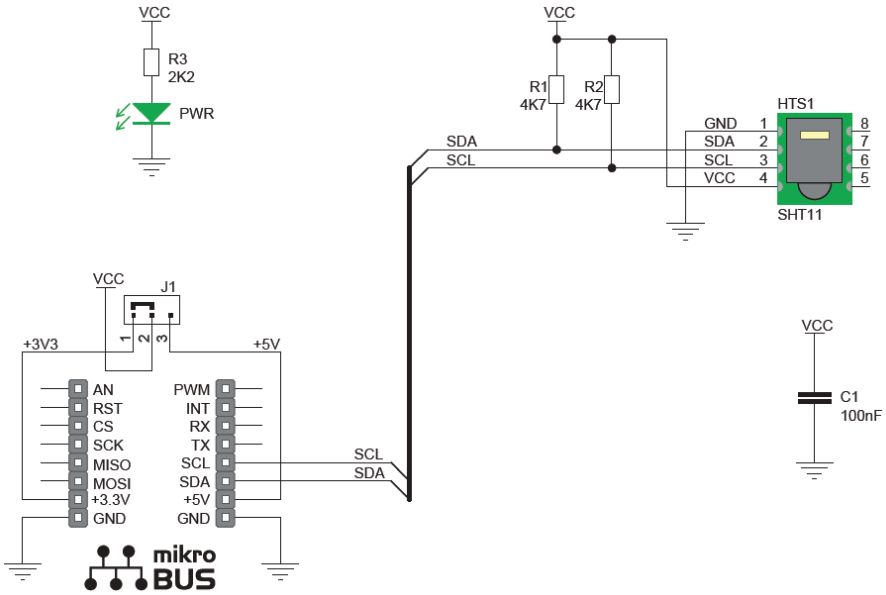
Na slici 3. prikazan je izgled modula SHT11 click Board.



Slika 3. SHT11 click board.

Digitalni senzor za merenje temperature i relativne vlažnosti vazduha SHT11 je proizvod firme Sensirion. Temperatura može biti prikazana u obliku 12-bitnog ili 14-bitnog podatka. Opseg temperature u kome ovaj senzor radi jeste od -40°C do +100 °C sa tačnošću od ±0,5 °C . Relativna vlažnost vazduha može biti predstavljena 8-bitnom rezolucijom, sa tačnošću od ±3%.

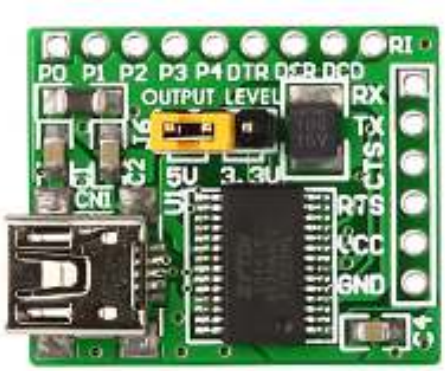
Ova komponenta može raditi na 3,3V ili na 5V. Preko linija SCL i SDA dovodi se clock signal, odnosno podaci.



Slika 4 .Električna šema modula SHT11.

2.3 USB UART board

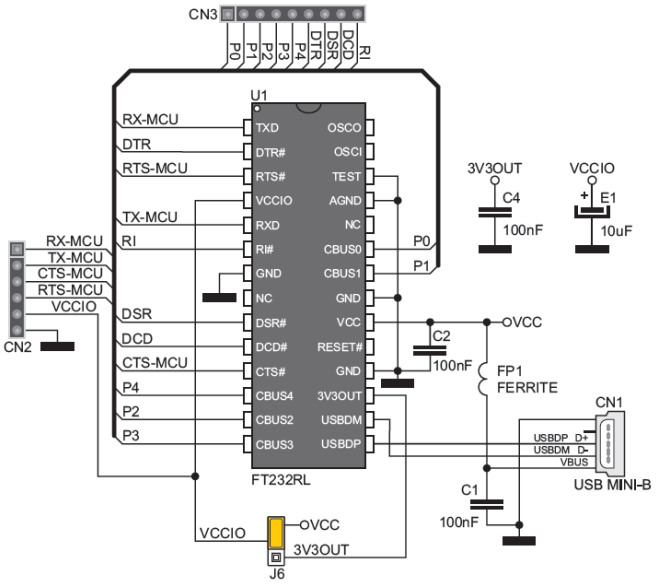
Na slici 4. prikazan je izgled ovog modula.



Slika 5. Usb Uart board.

Uz pomoć ovog modula povezuje se USB uredjaja i UART modul koji je implementiran u okviru mikrokontrolera. Obavlja se serijski transfer podataka putem UART komunikacije. Napajanje ovog modula može biti 5V ili 3,3V. Mali je potrošač.

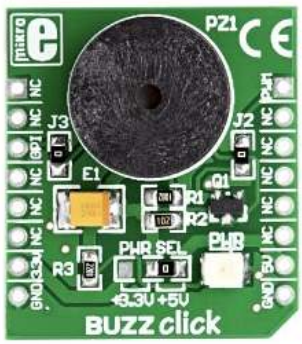
U ovom projektu je od značaja jer se preko njega obavlja komunikacija sa računarom. Ideja je da i osoba koja ne poznaje programiranje mikrokontrolera može da zadaje graničnu temperaturu. Dakle nema potrebe za stalnim učitavanjem celog programa na ploču.



Slika 6. Električna šema modula USB-UART.

2.4 BUZZ click board

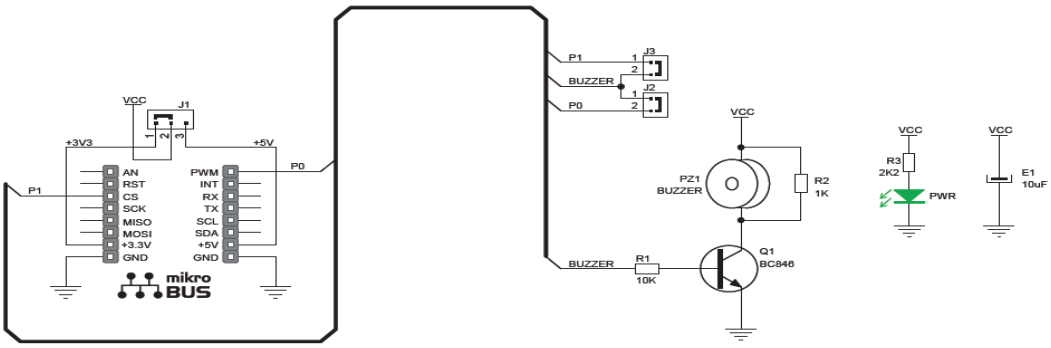
Na slici 7. Prikazan je izgled modula Buzz click board.



Slika 7. Buzz click board

Ovaj modul ima mogućnost emitovanja audio signala zahvaljujući piezo zvučniku koji poseduje. Rezonantna frekvencija Buzzera je 3,8 kHz, i na ovoj frekvenciji ima najbolje performanse.Može se povezati na mikrokontroler preko digitalne (GPI) ili PWM linije. Napajanje može biti 5V ili 3,3V.

U ovom projektu povezivanje je uradjeno preko PWM linije. Preko pina PC8 povezan je na tajmer TIM3, tajmerom se generiše PWM signal kojim se pobuđuje Buzz click.



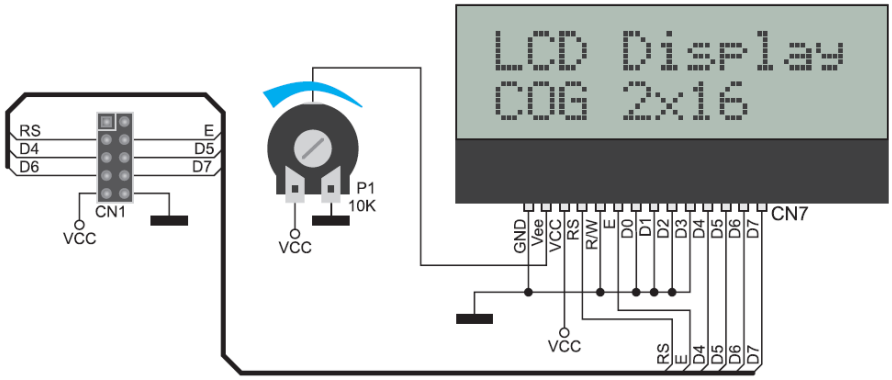
Slika 8. Električna šema modula buzz click

2.5 COG 2x16 LCD Board

Na slici 9. Prikazan je izgled ovog modula

Slika 9. LCD board

Ova pločica se koristi za prikaz informacija putem tekstualnih poruka. COG (Chip-on-Glass) 2x16 LCD Pločica se može lako povezati sa mikrokontrolerom. Na pločici se nalazi potenciometar za podešavanje kontrasta. Ispis na displeju se vrši u dva reda po šesnaest karaktera.



Slika 10. Električna šema LCD modula

Modul se povezuje preko 2x5 ženskog konektora CN1 na razvojni sistem na 2x5 muški konektor. Funkcije njegovih pinova su:

RS - Register sellection. Dovođenjem logičke jedinice na pin selektuje se data registar. Dovođenjem logičke nule na pin selektuje se instrukcijski registar.

Vee -Regulator kontrasta displeja, promenom položaja potenciometra P1 menja se napon napajanja.

R/W -Reading/Writing selekcija. Dovođenjem logičke jedinice na pin, vršiće se čitanje sa displeja. Dovođenjem logičke nule na pin podaci će se ispisivati na displej.

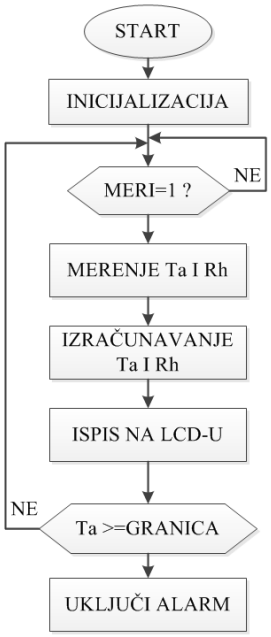
E -I/O dozvola.

D0-D1 -Pinovi koji se koriste za slanje podataka LCD-u.

Pinovi D4,D5,D6 i D7 primaju podatke poslate od strane mikrokontrolera kada su pinovi D0,D1.D2 i D3 na logičkoj nuli.

3 SOFTVERSKA REALIZACIJA

Program je realizovan u vidu glavnog programa i prekidne rutine. Na sledećoj slici prikazan je blok dijagram toka izvršavanja programa.



Slika 11. Blok dijagram glavnog programa.

Po stratu programa vrši se inicijalizacija periferija. U narednim tabelama dati su delovi koda programa.

```
1.    //Inicijalizujemo neke pinove porta B
2.    GPIO_StructInit(&GPIO_InitStructure); // Reset init structure
3.    GPIO_InitStructure.GPIO_Pin =  GPIO_Pin_6 + GPIO_Pin_7 + GPIO_Pin_8;     // pinovi 6,7 i 8 za displej
4.    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
5.    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_OD; // open Drain
6.    GPIO_Init( GPIOB, &GPIO_InitStructure );
7.
8.    //Inicijalizujemo neke pinove porta C
9.    GPIO_StructInit(&GPIO_InitStructure); // Reset init structure
10.   GPIO_InitStructure.GPIO_Pin =  GPIO_Pin_0 + GPIO_Pin_1 +GPIO_Pin_11 + GPIO_Pin_12; // pinovi 0,1,11 i 12 displej i
      senzor
11.   GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
12.   GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_OD;            // PC0,PC1,PC11 i PC12 open Drain
13.   GPIO_Init( GPIOC, &GPIO_InitStructure );
14.
15.  //Inicijalizujemo neke pinove porta D
16.   GPIO_StructInit(&GPIO_InitStructure); // Reset init structure
17.   GPIO_InitStructure.GPIO_Pin =  GPIO_Pin_2;//  pin 2 za displej
18.   GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
19.   GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_OD; // open Drain
20.   GPIO_Init( GPIOD, &GPIO_InitStructure );
21.
```

Tabela 1. Inicijalizacija portova za displej i senzor.

```
1.      //Inicijalizacija porta C, pina 8 koji se koristi za buzz-er
2.      GPIO_StructInit(&GPIO_InitStructure); // Reset init structure
3.      GPIO_InitStructure.GPIO_Pin =  GPIO_Pin_8;//buzzer
4.      GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
5.      GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;            // konfiguracija za alternativnu funkciju - Push Pull

6.      GPIO_Init( GPIOC, &GPIO_InitStructure );
7.    GPIO_PinRemapConfig( GPIO_FullRemap_TIM3, ENABLE );        // Map TIM3_CH3 to GPIOC.Pin8, TIM3_CH4 to GPIOC.Pin9
8.
9.      //Tajmer TI3 radi u modu generisanja PWM signala
10.
11.     //Inicijalizacija vremenske baze tajmera TIM3
12.
13.     TIM_TimeBaseStructInit( &TIM_TimeBaseInitStruct );
14.     TIM_TimeBaseInitStruct.TIM_ClockDivision = TIM_CKD_DIV4;
15.     TIM_TimeBaseInitStruct.TIM_Period = 849 - 1;   // 0..999
16.     TIM_TimeBaseInitStruct.TIM_Prescaler = 65 - 1; // Div 64
17.     TIM_TimeBaseInit( TIM3, &TIM_TimeBaseInitStruct );
18.
19.     //Inicijalizacija OC strukture tajmera TIM3
20.     TIM_OCStructInit( &TIM_OCInitStruct );
21.     TIM_OCInitStruct.TIM_OutputState = TIM_OutputState_Enable;
22.     TIM_OCInitStruct.TIM_OCPolarity = TIM_OCPolarity_Low;
23.     TIM_OCInitStruct.TIM_OCMode = TIM_OCMode_PWM1;
24.     //Initial duty cycle equals 0%. Value can range from zero to 1000.
25.     TIM_OCInitStruct.TIM_Pulse = 800; // 0 .. 1000 (0=Always Off, 1000=Always On)
26.     TIM_OC3Init( TIM3, &TIM_OCInitStruct ); // Channel 3 Blue LED
27.     TIM_Cmd( TIM3, ENABLE );//startovanje tajmera
```

Tabela 2 . Inicijalizacija porta C za rad sa Buzz click-om i tajmera TIM3 .

```
1.      //INICIJALIZACIJA TIMER2 PERIFERIJE KOJA CE DA GENERISE PERIODICNI PREKID
2.      //Konfiguracija vremenske baze za TIMER2 periferiju
3.      TIM_TimeBaseInitStruct.TIM_Period = 2700;
4.      TIM_TimeBaseInitStruct.TIM_Prescaler = 8880;
5.      TIM_TimeBaseInitStruct.TIM_ClockDivision = 0;
6.      TIM_TimeBaseInitStruct.TIM_CounterMode = TIM_CounterMode_Up;
7.      //Inicijalizacija TIMER2 periferije
8.      TIM_TimeBaseInit(TIM2, &TIM_TimeBaseInitStruct);
9.      //Dozvola generisanja prekida na nivou periferije
10.     TIM_ITConfig(TIM2,TIM_IT_Update, ENABLE);
11.     //Startovanje TIMER2 periferije
12.     TIM_Cmd(TIM2, ENABLE);
```

Tabela 3. Inicijalizacija tajmera Tim2.

```
1.    //Inicijalizacija Usart-a
2.        USART_DeInit(USART2);
3.
4.        USART_InitStructure.USART_BaudRate = 9600;
5.        USART_InitStructure.USART_WordLength = USART_WordLength_8b;
6.        USART_InitStructure.USART_StopBits = USART_StopBits_1;
7.        USART_InitStructure.USART_Parity = USART_Parity_No;
8.        USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
9.        USART_InitStructure.USART_Mode = USART_Mode_Rx + USART_Mode_Tx; //Omoguceno je slanje i primanje podataka
10.
11.       USART_Init(USART2, &USART_InitStructure);
12.       USART_Cmd(USART2, ENABLE);
```

Tabela 4 . Inicijalizacija Usart-a.

```
1.  //Inicijalizacija i podesavanje displeja
2.  LCD_Init();               //inicijalizacija displeja
3.  LCD_Send_Instr(0x28);     //Function Set - 4-bit, Dual Line
4.  LCD_Send_Instr(0x06);     //Entry Mode Set - Increment, Display Shift Off
5.  LCD_Send_Instr(0x10);     //Display/Cursor Shift - Move Cursor
6.  LCD_Send_Instr(0x01);     //Display Clear
7.  LCD_Send_Instr(0x02);     //Display/Cursor Home
8.  LCD_Send_Instr(0x0C);     //Display On, Underline Off, Blink Off
9.
10. Ta[11] = 223;
11. Ta[12] = 'C';                     // 'C' karakter
12. Rh[11] = '%';                     // '%' karakter
13.
14. LCD_Send_Instr(0x02);     //Display/Cursor Home
15. for (br=0; br<13; br++)
16.   LCD_Send_Data(Ta[br]);
17.
18. LCD_Send_Instr(0xC0);     //Drugi red
19. for (br=0; br<13; br++)
20.   LCD_Send_Data(Rh[br]);
```

Tabela 5. Inicijalizacija i podešavanje LCD-a.

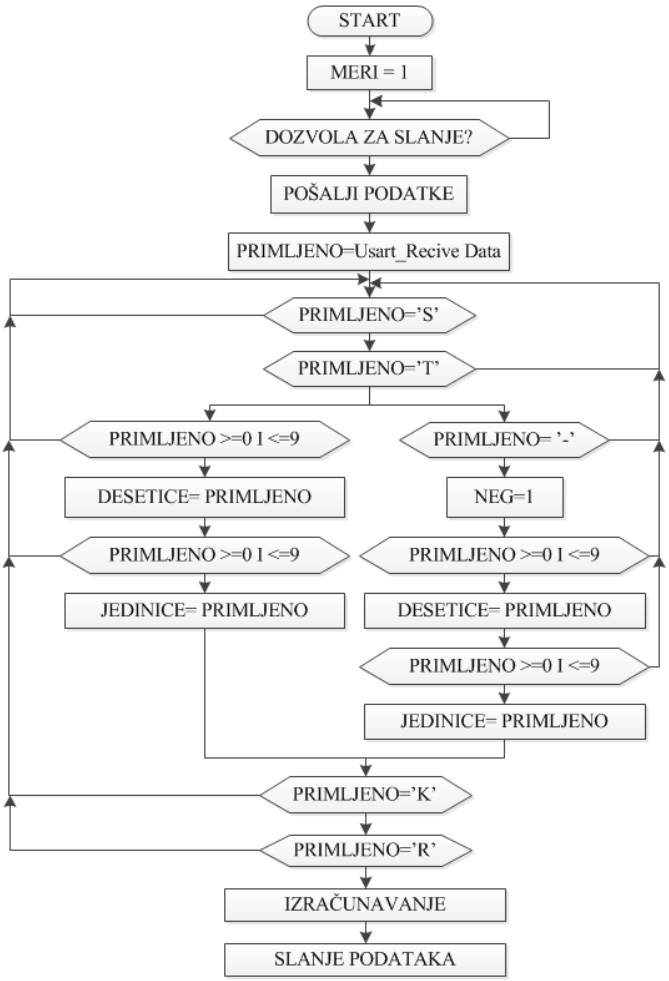
Potom se ispituje uslov za početak merenja, ako je uslov ispunjen pristupa se merenju. Uslov se resetuje. Upotrebom određenih konstanti , na osnovu izmerenih veličina izračunavaju se vrednosti izmerene temperature i relativne vlažnosti vazduha.Podaci se pripremaju za ispis na LCD-u. Ispituje se da li je izmerena temperatura veća od zadate kritične temperature, ako jeste uključuje se alarm, a ako nije - program se izvršava od početka, odnosno od ispitivanja uslova za merenje. Tabelom 6. dat je deo koda glavnog programa.

```
1.  while (1)
2.    {
3.     if (meri==1)  //Ako je zadat fleg da je potrebno izvrsiti merenje , ulazi u petlju
4.     {
5.       meri = 0;  //fleg se odmah resetuje
6.
7.       // Merenje temperature
8.      SOt = Measure(0x03);             // function for measuring (command 0x03 is for temperature)
9.
10.      // Merenje relativne vlaznosti vazduha
11.     SOrh = Measure(0x05);            // function for measuring (command 0x05 is for humidity)
12.
13. // Izracunavanje temperature
14.     Ta_res =(int)((D2*SOt+D1)*100);
15.
16. // Izracunavanje relativne vlaznosti vazduha
17.    Rh_res =(int)((C3*SOrh*SOrh+C2*SOrh+C1)*100);
18.
19. // Pripremanje podataka za ispis na LCD-u
20.     Ta[5] = Ta_res / 10000 + 48;
21.     Ta[6] = Ta_res % 10000 / 1000 + 48;
22.     Ta[7] = Ta_res % 1000 / 100 + 48;
23.     Ta[9] = Ta_res % 100 / 10 + 48;
24.     Ta[10] = Ta_res % 10 + 48;
25.
26. // Pripremanje podataka za ispis na LCD-u
27.     Rh[5] = Rh_res / 10000 + 48;
28.     Rh[6] = Rh_res % 10000 / 1000 + 48;
29.     Rh[7] = Rh_res % 1000 / 100 + 48;
30.     Rh[9] = Rh_res % 100 / 10 + 48;
31.     Rh[10] = Rh_res % 10 + 48;
32.
33. // Brisanje nepotrebnih nula
34.     if (Ta[5] == '0')                          // if Ta[5] = '0' then
35.       Ta[5] = ' ';                             // insert blank character to Ta[5]
36.     if (Ta[5] == ' ' && Ta[6] == '0')          // if Ta[5] is blank and Ta[6] = '0' then
37.       Ta[6] = ' ';                             // insert blank character to Ta[6]
38.
39.     if (Rh[5] == '0')                          // if Ta[5] = '0' then
40.       Rh[5] = ' ';                             // insert blank character to Ta[5]
```

```
41.      if (Rh[5] == ' ' && Rh[6] == '0')             // if Ta[5] is blank and Ta[6] = '0' then
42.        Rh[6] = ' ';                                // insert blank character to Ta[6]
43.
44.      LCD_Send_Instr(0x85);     // Prvi red
45.      for (br=5; br<11; br++)
46.        LCD_Send_Data(Ta[br]);
47.
48.      LCD_Send_Instr(0xC5);     // Drugi red
49.      for (br=5; br<11; br++)
50.        LCD_Send_Data(Rh[br]);
51.
52.      if (Ta_res > granica*100)// Startovanje buzzer-a
53.          TIM_Cmd( TIM3, ENABLE );
54.      else
55.          TIM_Cmd( TIM3, DISABLE );
56.    }
57.  }
```

Tabela 6. Glavni program.

Na narednoj slici prikazan je dijagram toka izvršavanja prekidne rutine.



Slika 12. Blok dijagram toka izvršavanja prekidne rutine.

Po ulasku u prekid,daje se dozvola za merenje, kako je prikazano u tabeli 7.

```
1.  extern char meri;
2.  void TIM2_IRQHandler(void)
3.  {
4.       meri = 1;  //postavlja se fleg koji startuje tajmer
5.
6.       //Brisemo prekidni fleg
7.       TIM_ClearFlag(TIM2, TIM_FLAG_Update);
8.  }
```

Tabela 7. Prekidna rutina-TIM2.

Prekidna rutina potom obrađuje podatke sa usart-a. Prvo se proverava da li postoji dozvola za slanje podataka, ako postoji, podaci se šalju i dozvola se resetuje. Podaci primljeni preko usart-a se smeštaju u pomoćnu promenljivu “primljeno“. Počinje se ispitivanje primljenih podataka. Podatak je validan ako je dobijen u formi STxxKR, pri čemu je xx dvocifreni broj od 00 do 99. Ispitivanje primljenih podataka realizovano je u vidu mašine stanja. Proverava se da li je prvi primljeni podatak karakter 'S', ako jeste prelazi se na sledeće stanje. Ako nije , ceo podatak se zanemaruje i čeka se naredna pošiljka. Pošto je prvi primljeni karakter bio 'S', proverava se da li je naredni 'T'. Ako nije,provera se poništava. Ako jeste prelazi se na sledeće stanje. Proverava se da li je sledeći podatak cifra od 0 do 9, ili znak minus '-'. Ako jeste cifra, upisuje se u promenljivu „desetice“. Ako je znak '-' promenljiva neg (negativno) se postavlja na '1'. Ako nije ništa od navedenog, provera se počinje od početka. Zatim, ako je je jedan od prethodna dva uslova ispunjen, proverava se da li je pristigla cifra od 0 do 9, ako jeste u prvom slučaju se upisuje u promenljivu „jedinice“ (pozitivan broj) odnosno u „desetice“ (negativan broj). U slučaju negativnog broja ostaje još jedna provera, ako je primljena još jedna cifra, ona se upisuje u jedinice. Zatim se ispituje da li je naredni podatak karakter 'K', ako nije poništava se ceo podatak,a ako jeste proverava se da li je naredni podatak karakter 'R', ako jeste podatak je validan i dobija dozvolu za prenos. Ako nije, ceo podatak se zanemaruje i pristupa se proveri naredne pošiljke.Ovaj deo koda prikazan je u tabeli 8.

```
1.  //Obrada podataka poslatih preko usart-a
2.  extern int i;
3.  extern char potvrda[], primljeno;
4.  char stanje=0, neg=0;
5.  char desetice, jedinice;
6.  extern int granica;
7.  char dozvola_za_slanje = 0;
8.
9.  void USART2_IRQHandler(void)
10. {
11.   if (USART2->SR & USART_FLAG_TXE)
12.   {
13.     if (dozvola_za_slanje)
14.     {
15.       USART_SendData(USART2, potvrda[i]);
16.       i++;
17.       if (potvrda[i] == 0)
18.       {
19.         USART_ITConfig(USART2, USART_IT_TXE, DISABLE);
20.         dozvola_za_slanje = 0;
21.       }
22.     }
```

```
23.       USART_ClearITPendingBit(USART2, USART_IT_TXE);
24.     }
25.
26.     if (USART2->SR & USART_FLAG_RXNE)
27.     {
28.       primljeno = USART_ReceiveData(USART2);
29.
30.       //  masina stanja
31.
32.       switch(stanje)
33.       {
34.       case 0 : if (primljeno == 'S') stanje = 1; //Ako je primljeno S predji na stanje 1
35.                break;
36.
37.       case 1 : if (primljeno == 'T') stanje = 2; //Ako je primljeno T predji na stanje 2, u suprotnom idi na stanje 0

38.                else stanje = 0;
39.                break;
40.
41.       case 2 : if ((primljeno >= '0') & (primljeno <= '9')) //Ako je primljena cifra od 0 do 9
42.                {
43.                  desetice = primljeno - '0'; //dodeli vrednost desetica
44.                  neg = 0;                    //vrednost podatka je pozitivna
45.                  stanje = 3;                 //predji na stanje3
46.                }
47.                else
48.                  if (primljeno == '-')       //ako je primljen karakter -
49.                  {
50.                    neg = 1;                  //vrednost podatka je negativna
51.                    stanje = 4;               //predji na stanje 4
52.                  }
53.                  else stanje = 0;            //ako nije nista od navedenog, predji na stanje 0
54.                break;
55.
56.       case 3 : if ((primljeno >= '0') & (primljeno <= '9')) //ako je primljena cifra 0 do 9
57.                {
58.                  jedinice = primljeno - '0';                 //dodeli vrednost jedinica
59.                  stanje = 5;                                 //predji na stanje 5
60.                }
61.                else stanje = 0;                              //ako nije nista od navedenog, predji na stanje 0
62.                break;
63.
64.       case 4 : if ((primljeno >= '0') & (primljeno <= '9')) //ako je primljena cifra 0 do 9
65.                {
66.                  desetice = primljeno - '0'; //dodeli vrednost desetica
67.                  stanje = 3;                 //predji na stanje3
68.                }
69.                else stanje = 0;              //ako nije nista od navedenog, predji na stanje 0
70.                break;
71.
72.       case 5 : if (primljeno == 'K') stanje = 6; //Ako je primljen karakter K predji na stanje 6
73.                else stanje = 0;
74.                break;
75.
76.       case 6 : if (primljeno == 'R') //Ako je primljen karakter R
77.                {
78.                  granica = desetice*10 + jedinice; //izracunava se vrednost
79.                  if (neg) granica = -granica;     //ako je broj negativan, dodaje se predznak
80.                  dozvola_za_slanje = 1;           //dozvoljava se prenos podataka
81.                  i = 0;
82.                  USART_ITConfig(USART2, USART_IT_TXE, ENABLE); //dozvoljava se prekid
83.                }
84.       default : stanje = 0;
85.       }
86.
87.       USART_ClearITPendingBit(USART2, USART_IT_RXNE);
88.     }
89.   }
```

Tabela 8. Prekidna rutina – Usart.

ZAKLJUČAK

U ovom projektu obradjen je problem digitalnog merača temperature i relativne vlažnosti vazduha sa alarmom. Ideja je bila napraviti sklop koji može imati primenu u realnom svetu. Odabir modula je takav da se merenjem temperature i vlažnosti može upravljati i preko hiper terminala, bez potrebe za otvaranjem koda programa.

Postoji mnogo mogućnosti za nadogradnju projekta, na primer, dodavanjem potrebnih modula rezultat bi mogao da se šalje bežično na neki drugi uređaj. Takođe, rezultat merenja mogao bi da kontroliše otvaranje ili zatvaranje ventila, zatim da kontroliše releje.. Mogućnosti su velike uz dodavanje željenog hardvera i manje promene u kodu.

LITERATURA

[1] http://www.st.com/internet/mcu/product/216844.jsp

[2] http://www.mikroe.com

[3]http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_LITERATURE/USER_MANUAL/CD00267113.pdf

[4]http://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/Humidity/Sensirion_Humidity_SHT1x_Datasheet_V5.pdf

[5] http://www.sparkfun.com/datasheets/LCD/HD44780.pdf