Univerzitet Singidunum Beograd,
Tehnički fakultet

Sistemi za rad u realnom vremenu

Izveštaj

Snake game
(EasyPick)

Student:
Đorđe Krstović

Beograd, 2020

Sadržaj

Uvod

Snake game – jednostavna verzija snake igre na EasyPic V7 platformi, uz korišćenje GLCD modula i tastera koji su dostupni na ploči RC4, RC5, RC6 i RC7. Ova verzija igre podržava mogućnost prolaza kroz zidove I ne raste, niti jede kockice kao u standardnoj verziji igre, već ima dve prepreke, jednu veću I jednu manju na određenim koordinatama. Zmiju je moguće kontrolisati tasterima koji su već navedeni, odnosno RC6 i RC7 su kontrolni tasteri za smer kretanja zmije, dok RC4 resetuje igru I vraća je na početak, a taster RC5 zamrzava igru u trenutku kada je pritisnut, takođe ponovnim pritiskom na taster RC5 igra se nastavlja od trenutka kada je pauzirana. Igra je napisana u C programskom jeziku.

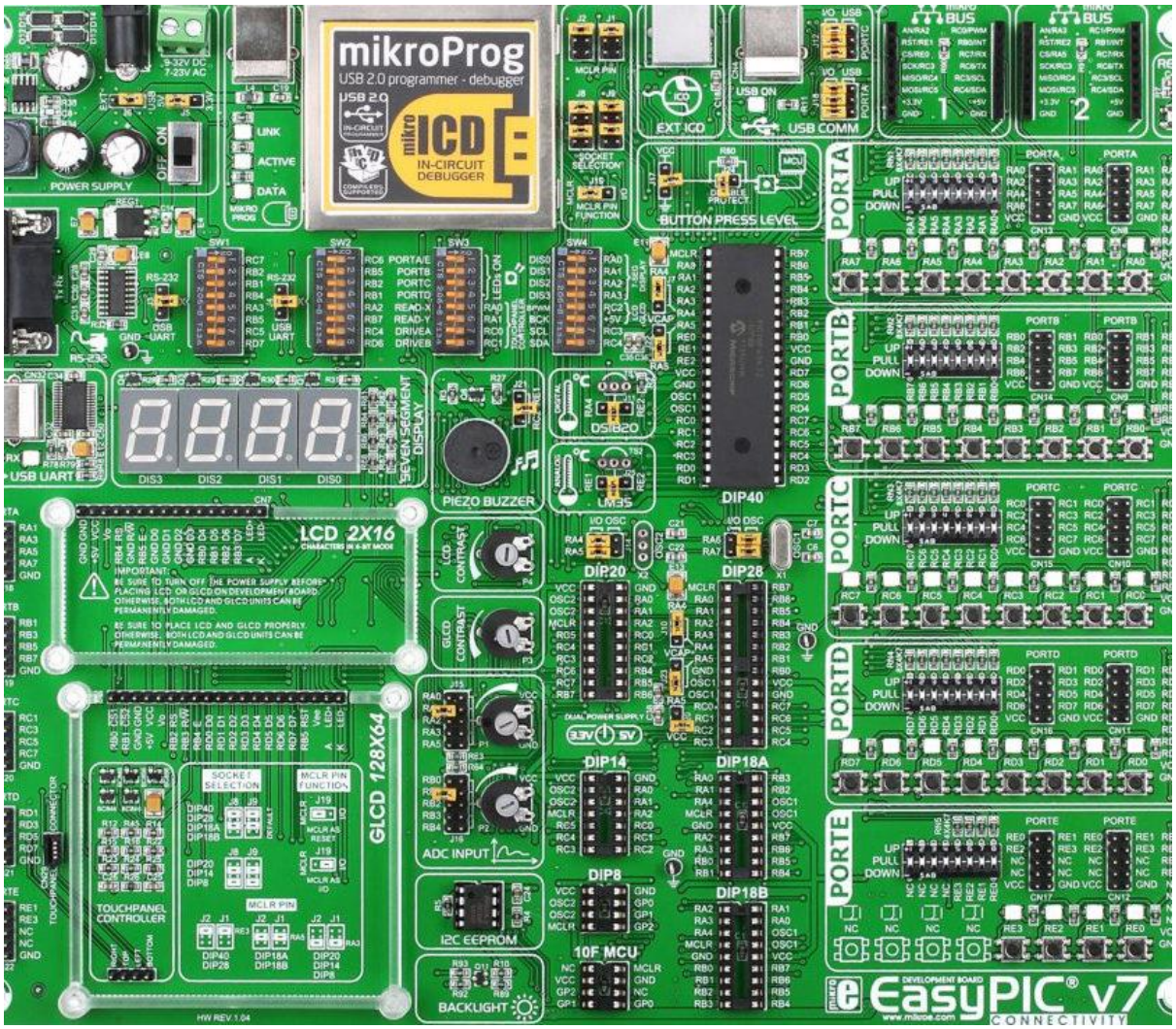
Komponente

Tabelarni prikaz svih komponenti koje su potrebne za realizaciju ovog projekta, uz podatke o količini i koloni „Opširnije“ koja šire opisuje komponente kao i njihovu namenu.

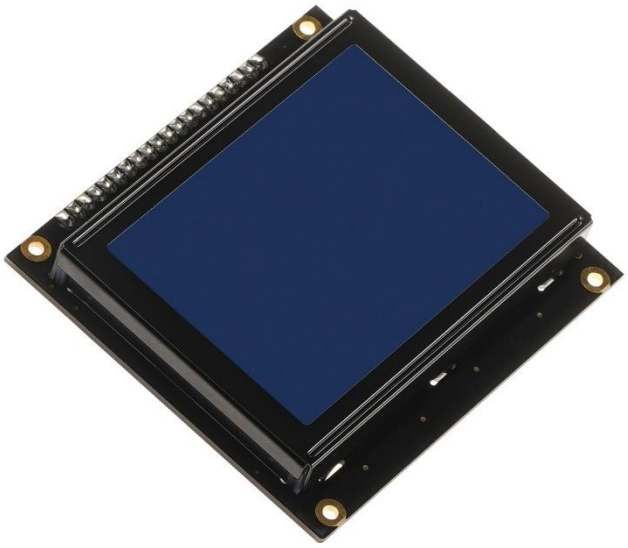
Komponenta	Količina	Opširnije
Easypic v7	1	PIC Development Board
GLCD	1	Graphic LCD 128x64

Komponente - tabela

EasyPic V7 + GLCD



EasyPic V7 - MikroE



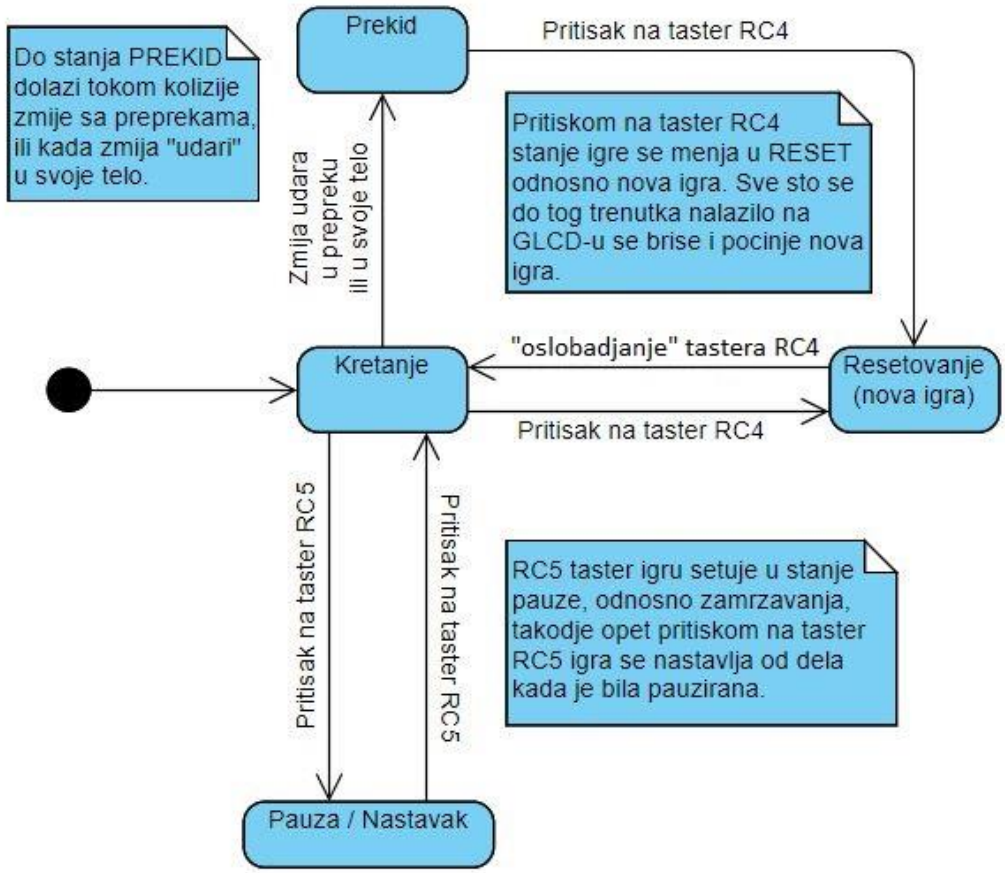
Graphic LCD 128x64

Dijagram stanja

Dijagram stanja koji ilustruje ideju realizacije programa uz sva stanja u kojima igra može da se nađe i napomene.

Stanja:

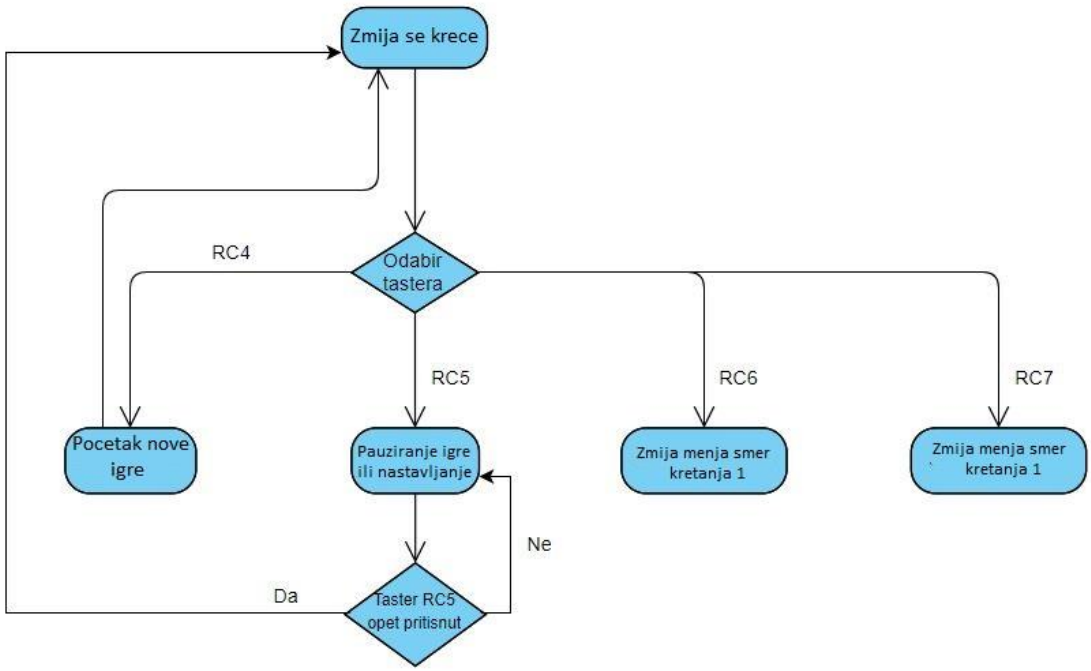
- Kretanje – u ovom stanju se zmija nalazi dok god ne dođe do prekida prilikom kolizije sa nekom od prepreka ili dok zmija ne udari u svoje telo. U ovom stanju je moguće upravljati zmijom tasterima RC6 i RC7.
- Pauza/Nastavak – ovo stanje je moguće pozvati iz stanja kretanja, pritiskom na taster RC5, čime se igra „zamrzava“ odnosno pauzira, za izlazak iz ovog stanja, odnosno nastavljanje od trenutka kada je pauzirana, koristi se isti taster – RC5.
- Prekid – igra prelazi u ovo stanje ukoliko je došlo do kolizije zmije sa preprekom ili sa telom. U ovom stanju koristniku se ispisuju poruke „Igra je zavrsena!“ i „Pocni novu igru pritiskom na RC4“
- Resetovanje(nova igra) – ovo stanje igru resetuje i zmiju vraća na početak tj početnu poziciju, odnosno praktično kreira novu igru. Prelazak igre u ovo stanje vrši se pritiskom na taster RC4.



Dijagram stanja

Dijagram aktivnosti

Dijagram aktivnosti – prikaz aktivnosti koje su moguće pri korišćenju tastera za kontrolu igre.



Dijagram aktivnosti

Kod implementacije

```
/**********************************
|   Konstante za smer kretanja      |
**********************************/
#define UP 0
#define RIGHT 1
#define DOWN 2
#define LEFT 3

/**********************************
|   Konfiguracija GLCD-a            |
|   Izvor mikroe.com - glcd library |
**********************************/
char GLCD_DataPort at PORTD;    // GLCD port za podatke
sbit GLCD_CS1 at LATB0_bit;     // Chip Select 1 line
sbit GLCD_CS2 at LATB1_bit;     // Chip Select 2 line
sbit GLCD_RS at LATB2_bit;      // Register select line
sbit GLCD_RW at LATB3_bit;      // Register Read/Write line
sbit GLCD_EN at LATB4_bit;      // Enable line
sbit GLCD_RST at LATB5_bit;     // Reset line
// Direction
sbit GLCD_CS1_Direction at TRISB0_bit;      // Direction of the Chip Select 1 pin
sbit GLCD_CS2_Direction at TRISB1_bit;      // Direction of the Chip Select 2 pin
sbit GLCD_RS_Direction at TRISB2_bit;       // Direction of the Register select pin
sbit GLCD_RW_Direction at TRISB3_bit;       // Direction of the Read/Write pin
sbit GLCD_EN_Direction at TRISB4_bit;       // Direction of the Enable pin
sbit GLCD_RST_Direction at TRISB5_bit;      // Direction of the Reset pin.

/**********************************
|    Inicijalizacija/deklaracija    |
**********************************/
// koordinate
int snakeCoordinateX[20];   // Vrednost koordinate X
int snakeCoordinateY[20];   // Vrednost koordinate Y
// fleg bitovi za tastere RC4, RC5, RC6, RC7
bit oldstate7,oldstate6,oldstate5,oldstate4,flag7,flag6,flag5,flag4;
// interrupt flag, prekid sa tajmera
int flag;
// stanje kretanja zmije -> UP, DOWN, RIGHT, LEFT
int direction;
// brojaci za for petlje
int i,j,k;
// x i y promenljive za proveru pozicije zmije
int x,y;

/**************************************
|   Interrupt rutina (prekid)          |
|   Tajmer 50ms generisan uz pomoc     |
|   app "Timer_Calculator_Build.4.0.0" |
**************************************/
// link za app https://libstock.mikroe.com/projects/view/398/timer-calculator
void interrupt() {              // glavni interrupt
    if (TMR0IF_bit) {
        TMR0IF_bit = 0;
        TMR0H = 0x3C;
        TMR0L = 0xB0;
        flag=1;
    }
}

```

```
/**********************************
|    Funkcija barrier_collision      |
|    proverava da li je doslo do     |
|    sudara sa nekom od prepreka     |
**********************************/
int barrier_collision(){

     // kriticne tacke - ivice prepreke u koju zmija moze da udari (po dokumentaciji za Glcd_Rectangle_Round_Edges_Fill treba nam
     // X,Y gore levo, i X,Y dole desno za crtanje tela, odnosno u ovom slucaju prepreke)
    int x_rect_1[11] = {10,11,12,13,14,15,16,17,18,19,20};
    int x_rect_2[21] = {54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74};
    int y_rect_1[11] = {10,11,12,13,14,15,16,17,18,19,20};
    int y_rect_2[21] = {22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42};

    /**********************************
    |          Manja prepreka          |
    **********************************/
    // proveravamo da li se desio sudar sa prvom(manjom) preprekom
    for(x = 0; x < 11; x++){
        for(y = 0; y < 11; y++){
              // usov koji proverava da li je doslo do sudara
              if(snakeCoordinateX[0] == x_rect_1[x] && snakeCoordinateY[0] == y_rect_1[y]){

                  return 1;
              }

        }
    }

    /**********************************
    |           Veca prepreka          |
    **********************************/
    // proveravamo da li se desio sudar sa drugom(vecom) preprekom
    for(x = 0; x < 21; x++){
        for(y = 0; y < 21; y++){
              // usov koji proverava da li je doslo do sudara
              if(snakeCoordinateX[0] == x_rect_2[x] && snakeCoordinateY[0] == y_rect_2[y]){

                  return 1;
              }

        }
          }
          return 0;
      }

      /*****************************************
      |   Funkcija draw_barrier                 |
      |   iscrtava prepreke na koordinatama     |
      |   gornjih levih temena (54,22) i (10,10)  |
      |   i dužinama stranica 20 i 10 respektivno |
      *****************************************/
      void draw_barrier(){
          // Manja prepreka
          Glcd_Rectangle_Round_Edges_Fill(10,10,20,20,0,1);  //gore levo X,Y, dole desno X,Y, radius = 0, crna boja => 10x10
          // Veca prepreka
          Glcd_Rectangle_Round_Edges_Fill(54,22,74,42,0,1);  //gore levo X,Y, dole desno X,Y, radius = 0, crna boja => 20x20
      }

```

```
117.        /*****************************************
118.        |   Funkcija state_reset                  |
119.        |   resetuje stanje zmije, odnosno niz sa |
120.        |   koordinatama postavlja na nule i vraca|
121.        |   zmiju na pocetnu poziciju.            |
122.        *****************************************/
123.        void state_reset() {
124.
125.            Glcd_Fill(0x00);   // "ciscenje" ekrana
126.            draw_barrier();    // crtanje prepreka nakon "ciscenja" ekrana
127.
128.            // birsanje piksel po piksel
129.            for ( i = 19; i > 0; i-- ){
130.                snakeCoordinateX[i]=0; snakeCoordinateY[i]=0;
131.            }
132.
133.             // setovanje na pocetnu poziciju
134.             snakeCoordinateX[0]=0;
135.             snakeCoordinateY[0]=0;
136.             // podrazumevano ide desno
137.             direction=RIGHT;
138.
139.        }
140.
141.
142.        /*****************************************
143.        |   Funkcija pause_and_continue           |
144.        |   pritiskom na taster RC5 - pauza,      |
145.        |   ponovnim pritiskom na RC5 igra se     |
146.        |   nastavlja od trenutka kada je pauzirana |
147.        *****************************************/
148.        void pause_and_continue(){
149.            if(flag5 == 1) {     // proveravamo da li postoji zahtev za prekid
150.                flag5 = 0;
151.            }
152.
153.            // provera - ukoliko je drugi put pritisnut taster RC5 zmija nastavlja sa kretan
    jem od trenutka kada je pauzirana
154.            do {
155.
156.                for(k=0; k<20; k++){
157.                    // crtanje zmije na poziciji gde se trenutno nalazi
158.                    // parametri: koordinate X i Y, 1 = boja (0 - brise tacku, 1 crta, 2 inv
    ertuje)
159.                    Glcd_Dot(snakeCoordinateX[k], snakeCoordinateY[k], 1);
160.
161.                }
162.
163.                // &PORTC, 5, 1, 0 - referenca na port &PORTC, pin 5, duzina za detektovanje
     pritiska - vremensko ogranicenje(ms),
164.                // koliko se drzi taster da se detektuje prekid, 0 ili 1 odnosno HIGH ili LO
    W tj. da li je pritisnut ili ne
165.                // prakticno provera da li je dugme pritisnuto
166.                if (Button(&PORTC, 5, 1, 1)) {
167.                    oldstate5 = 1;
168.                }
169.
170.                // &PORTC, 5, 1, 0 - referenca na port &PORTC, pin 5, duzina za detektovanje
     pritiska - vremensko ogranicenje(ms),
171.                // koliko se drzi taster da se detektuje prekid, 0 ili 1 odnosno HIGH ili LO
    W tj. da li je pritisnut ili ne
172.                // na ovaj nacin se obezbedjuje da se izvrsi samo ukoliko je taster pritisnu
    t SAMO jednom
173.                // oldstate se menja u odnosu na prethodni if kada je doslo do promene stanj
    a, dugme je promenilo stanje iz 1 u 0
174.                if (oldstate5 && Button(&PORTC, 5, 1, 0)) {
```

```
175.                    oldstate5 = 0;
176.                    flag5 = 1;
177.                }
178.
179.            }while(flag5==0); // dok se ne pritisne taster RC5 opet, fleg je setovan na 0
180.
181.        }
182.
183.        /***********************
184.        |   Funkcija main()    |
185.        ***********************/
186.        void main() {
187.
188.            // ANSEL => ansel=1 (port analogni), ansel=0 (port digitalni)
189.            // Svi portovi koje koristimo su digitalni ulazi/izlazi
190.            ANSELB = 0;
191.            ANSELC = 0;
192.            ANSELD = 0;
193.
194.            // registar TRISC u kom je PORTC postavljen u input rezim
195.            TRISC=0xFF;
196.
197.            // konfiguracija za tajmer, tajmer kalkulator generise conf deo, ovo je potrebno
    da bi se prvi put izvrsilo sve
198.            T0CON = 0x80;
199.            TMR0H = 0x3C;
200.            TMR0L = 0xB0;
201.            // interrupt enable
202.            TMR0IE_bit = 1;
203.
204.
205.            // general interrupt enable, omogucava da se prekid izvrsava
206.            GIE_bit = 1;
207.
208.            // mora da postoji inicijalizacija GLCD modula
209.            Glcd_Init();
210.
211.            // preventivno ciscenje ekrana da budemo sigurni da nije nista ostalo
212.            state_reset();
213.
214.            // inicijalizacija flegova, svi su pocetno setovani na 0 dok se ne desi potreba
   za prekidom
215.            flag7=0;
216.            flag6=0;
217.            flag5=0;
218.            flag4=0;
219.
220.
221.            do {
222.
223.                // Napomena: za svaki taster proveravamo uvek da li je taster pritisnut jedn
   om, odnosno da li menja stanje
224.
225.                /*******************************
226.                |   pritisak na  taster RC7     |
227.                |   taster za kontrolu kretanja |
228.                *******************************/
229.                if (Button(&PORTC, 7, 1, 1)) {
230.                    oldstate7 = 1;
231.                }
232.                if (oldstate7 && Button(&PORTC, 7, 1, 0)) {
233.                    flag7 =1;
234.                    oldstate7 = 0;
235.                }
236.
237.
```

```
238.                /*******************************
239.                |   pritisak na  taster RC6     |
240.                |   taster za kontrolu kretanja |
241.                *******************************/
242.                if (Button(&PORTC, 6, 1, 1)) {
243.                    oldstate6 = 1;
244.                }
245.                if (oldstate6 && Button(&PORTC, 6, 1, 0)) {
246.                    flag6 =1;
247.                    oldstate6 = 0;
248.                }
249.
250.
251.                /*******************************
252.                |   pritisak na  taster RC5     |
253.                |   pauza ili nastavak          |
254.                *******************************/
255.                if (Button(&PORTC, 5, 1, 1)) {
256.                    oldstate5 = 1;
257.                }
258.                if (oldstate5 && Button(&PORTC, 5, 1, 0)) {
259.                    oldstate5 = 0;
260.                    pause_and_continue();    // nije potrebno setovati fleg jer se provera d
    esava u funkciji "pause_and_continue"
261.                }
262.
263.
264.                /*******************************
265.                |   pritisak na  taster RC4     |
266.                |   ponovno pokretanje igre     |
267.                *******************************/
268.                if (Button(&PORTC, 4, 1, 1)) {
269.                    oldstate4 = 1;
270.                }
271.                if (oldstate4 && Button(&PORTC, 4, 1, 0)) {
272.                    flag4 =1;
273.                    oldstate4 = 0;
274.                }
275.
276.
277.
278.                // provera da li je doslo do prekida sa tajmera u interrupt rutini (bilo koj
    i fleg utice na to se desi prekid)
279.                // ukoliko jeste izvrsava sve dole navedeno
280.                if (flag){
281.
282.                    // brisanje zmije od repa da bi se videlo kretanje
283.                    Glcd_Dot(snakeCoordinateX[19], snakeCoordinateY[19], 0);
284.
285.                    // proveravamo da li je doslo do kolizije sa preprekom, ili telom zmije,
     za svaku koordinatu
286.                    for ( i = 19; i > 0; i-- ){
287.
288.                        /**********************
289.                        |   tip kolizije:      |
290.                        |   SA PREPREKOM       |
291.                        **********************/
292.                        if(barrier_collision()) {
293.                            // "ciscenje" ekrana
294.                            Glcd_fill(0x00);
295.                            // poruke                                          // 55 jer je GLC
    D 128x64, da bi bilo negde na sredini
296.                            Glcd_Write_Text("Igra je zavrsena!", 55, 2, 1); // parametri: po
    ruka, pozicija na ekranu, red, "boja"(0 -nista, 1- crno, 2-invertovano)
297.                            Glcd_Write_Text("Pocni novu igru pritiskom na RC4", 48, 4, 1);
298.
```

```
                        // program ocekuje pritisak na taster RC4 nakon sto je poruka is
    pisana
                        do {

                            if (Button(&PORTC, 4, 1, 1)) {
                                oldstate4 = 1;
                            }
                            if (oldstate4 && Button(&PORTC, 4, 1, 0)) {
                                flag4 =1;
                                oldstate4 = 0;
                            }


                        // program ocekuje promenu na flag4 kako bi nastavio dalje sa ra
    dom
                        } while(flag4 != 1);
                    }


                    /**********************
                    |   tip kolizije:      |
                    |   SA TELOM ZMIJE     |
                    **********************/
                    // provera da li je prva takca razlicita od poslednje, odnosno da li
     su koordinate glave i repa razlicite
                    // bez ovoga igra bi odmah detektovala koliziju
                   if(snakeCoordinateX[0] != snakeCoordinateX[19]){
                      if(snakeCoordinateX[0]==snakeCoordinateX[i] && snakeCoordinateY[0]
    ==snakeCoordinateY[i]){// provera koordinata glave i repa

                        // "ciscenje" ekrana
                      Glcd_fill(0x00);
                        // poruke
                      Glcd_Write_Text("Igra je zavrsena!", 55, 2, 1); // parametri:
    poruka, pozicija na ekranu, red, "boja"(0 -nista, 1- crno, 2-invertovano)
                        Glcd_Write_Text("Pocni novu igru pritiskom na RC4", 48, 4, 1);


                        do {
                            if (Button(&PORTC, 4, 1, 1)) {
                               oldstate4 = 1;
                            }
                            if (oldstate4 && Button(&PORTC, 4, 1, 0)) {
                               flag4 =1;
                               oldstate4 = 0;
                            }

                        } while(flag4 != 1);
                    }
                }
                       // kretanje zmije => prethodna koordinata se postavlja na trenutn
    u, trenutna na sledecu, npr 19 postaje 18 i td, tako se krece
                    snakeCoordinateX[i]=snakeCoordinateX[i-1];
                    snakeCoordinateY[i]=snakeCoordinateY[i-1];

            }

```

```
                    /**********************
                    |   Provera smera      |
                    |   u kom se trenutno  |
                    |   zmija krece        |
                    **********************/
                    switch (direction){


                        /************
                        |   GORE     |
                        ************/
                        // ukoliko se zmija krece u smeru ka GORE, onda se Y koordinata sman
    juje
                        case UP:
                            snakeCoordinateY[0]--;
                            // na ovaj nacin se omogucava da zmija prodje kroz zid, proverom
     trenutne koordinate i setovanjem sledece
                            if (snakeCoordinateY[0]<0) snakeCoordinateY[0]=63;
                            if (flag7){ // ako je doslo to pritiska na taster RC7 smer se me
    nja u levo
                                direction = LEFT;
                                flag7=0;
                            }
                            if (flag6){ // ako je doslo to pritiska na taster RC6 smer se me
    nja u desno
                                direction = RIGHT;
                                flag6 = 0;
                            }
                            if(flag4){ // ako je pritisnut taster RC4 pocinje nova igra
                                flag4 = 0;
                                state_reset();
                            }
                            break;

                        /************
                        |   DOLE     |
                        ************/
                        case DOWN:
                            snakeCoordinateY[0]++;
                            // visina glcd je 63
                            if (snakeCoordinateY[0]>63) snakeCoordinateY[0]=0;
                            if (flag7){
                                direction = RIGHT;
                                flag7=0;
                            }
                            if (flag6){
                                direction = LEFT;
                                flag6 = 0;
                            }
                            if(flag4){
                                flag4 = 0;
                                state_reset();
                            }
                            break;
```

```
400.
401.                         /************
402.                         |   DESNO    |
403.                         ************/
404.                         case RIGHT:
405.                             snakeCoordinateX[0]++;
406.                             // sirina glcd je 127
407.                             if (snakeCoordinateX[0]>127) snakeCoordinateX[0]=0;
408.                             if (flag7){
409.                                 direction = UP;
410.                                 flag7=0;
411.                             }
412.                             if (flag6){
413.                                 direction = DOWN;
414.                                 flag6 = 0;
415.                             }
416.                             if(flag4){
417.                                 flag4 = 0;
418.                                 state_reset();
419.                             }
420.                             break;
421.
422.                         /************
423.                         |   LEVO     |
424.                         ************/
425.                         case LEFT:
426.                             snakeCoordinateX[0]--;
427.                             if (snakeCoordinateX[0]<0) snakeCoordinateX[0]=127;
428.                             if (flag7){
429.                                 direction = DOWN;
430.                                 flag7=0;
431.                             }
432.                             if (flag6){
433.                                 direction = UP;
434.                                 flag6 = 0;
435.                             }
436.                             if(flag4){
437.                                 flag4 = 0;
438.                                 state_reset();
439.                             }
440.                             break;
441.
442.
443.                     }
444.
445.                     // resetuje se flag za TMR0
446.                     flag=0;
447.
448.                     for ( j = 0; j < 20; j++ ){
449.
450.                         // crtanje tela zmije nakon svih provera, na mestu gde se trenutno n
    alazi
451.                         Glcd_Dot(snakeCoordinateX[j], snakeCoordinateY[j], 1);
452.
453.                     }
454.
455.
456.                 }
457.
458.             } while(1);
459.
460.         }
```