Elektrotehnički fakultet u Sarajevu

Projektni zadatak iz predmeta
UGRADBENI SISTEMI
Kompletan listing koda

Članovi: Kenan Karahodžić i Edvin Teskeredžić
Grupa: 2
Naziv tima: Diffie-Hellmann's
Tema: Conway's Game of Life
Demonstrator: Nermin Čović

15. juni 2018. Sarajevo

Sadržaj

Kod - sa refaktoringom

```
/*
* PROJEKTNI ZADATAK IZ UGRADBENIH SISTEMA - AKADEMSKA GODINA 2017./2018.
* KREATORI: Edvin Teskeredzic i Kenan Karahodzic;
* Bilo kakvo koristenje ovog koda bez znanja kreatora nije dozvoljeno!
*/

#include "mbed.h"
#include "rtos.h"

// REGISTRI ZA DISPLAY - preuzeto sa datasheet-a i linka: https://os.mbed.com/users/andcor02/code/max7219ledarray/rev/
#define NO_OP_REG           0x00    // no operation registar
#define DIGIT_0_REG         0x01    // prvi red
#define DIGIT_1_REG         0x02    // drugi red
#define DIGIT_2_REG         0x03    // treci red
#define DIGIT_3_REG         0x04    // cetvrti red
#define DIGIT_4_REG         0x05    // peti red
#define DIGIT_5_REG         0x06    // sesti red
#define DIGIT_6_REG         0x07    // sedmi red
#define DIGIT_7_REG         0x08    // osmi red
#define DECODE_MODE_REG     0x09    // oznacava da li se koristi decode mode
#define INTENSITY_REG       0x0A    // intenzitet kojim diode svijetle
#define SCAN_LIMIT_REG      0x0B    // koliko smije dioda goriti istovremeno (8)
#define SHUTDOWN_REG        0x0C    // ako je setovan, displaj je upaljen


// KOMUNIKACIJA
Serial pc(USBTX, USBRX);

// DISPLAY-i
DigitalOut cs1(PTD0);
DigitalOut cs2(PTC16);
DigitalOut cs3(PTD5);
DigitalOut cs4(PTC13);
SPI spi(PTD2, PTD3, PTD1); // MOSI, MISO, SCLK

// MATRICE ZA GAME OF LIFE
char trenutno[16][16] = { {0} };
char staro[16][16] = { {0} };
unsigned char red[8] = {0}; // ovo koristimo za ispis u jednu matricu

// THREAD
Thread thread;

// MISC.
unsigned char pauza = 1;        // oznacava da li se vrsi prelazak u novu generaciju - na pocetku je 1 jer sistem miruje (ceka se input korisnika)
float vrijeme_cekanja = 0.3;    // oznacava koliko dugo cekamo na narednu generaciju
char a = 0;                     // za komsije

// ova funkcija vraca nasumican broj svaki put kada bude pozvana - pri tome ne koristeci rand funkciju, sto uklanja potrebu za cstdlib bibliotekom
unsigned int SEED = time(NULL);
int dajNasumicanBroj(unsigned int *pSEED, int granica)
{

*pSEED = (*pSEED * 48271) % 2147483647;

return *pSEED % granica;
```

```
}

// pise na odgovarajuci display zadan kao parametar
void pisi(unsigned char registar, unsigned char vrijednost, unsigned char redni)
{

    switch(redni)
    {
        case 1: // prvi display
            cs1 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs1 = 1;
            break;
        case 2: // drugi display
            cs2 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs2 = 1;
            break;
        case 3: // treci display
            cs3 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs3 = 1;
            break;
        case 4: // cetvrti display
            cs4 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs4 = 1;
            break;
        default:
            break;
    }
    wait(0.01);
}

// gasi matricu
void ocistiDisplej(int displej)
{
   for(int i = 1; i <= 8; ++i)
     pisi(i, 0, displej);
}

// cisti sve 4 matrica
void ocistiSve()
{
    for(int i = 1; i<=4; ++i)
        ocistiDisplej(i);

    for(int i = 0; i<16;++i)
        for(int j = 0; j<16;++j)
            trenutno[i][j] = 0;
}

// postavljanje defaultnih vrijednosti za sve 4 matrice (initialization routine - koristi se
    nakon Žto se sistem tek upali)
void init()
{
    for(int i = 1; i <= 4; ++i)
    {
```

```
118         pisi(DECODE_MODE_REG, 0x00, i);     // decode mode ne koristimo
119         pisi(INTENSITY_REG, 0x0f, i);       // stavi duty cycle na maksimalnu mogucu vrijednost (
    osvijetljenost)
120         pisi(SCAN_LIMIT_REG,0x07, i);       // upisemo broj 7 jer zelimo da palimo sve do reda
    indeksa 7 (a to je osmi red)
121         pisi(SHUTDOWN_REG,0x01, i);         // stavlja se u normal opmode, jer ne testiramo nista
122         pisi(0x0F, 0x0F, i);                // udji u display test mode
123         wait_ms(500);                       // radi 500 ms
124         ocistiDisplej(i);                   // ugasi sve
125         pisi(0x0F, 0x00, i);                // izadji iz display test mode
126         wait_ms(500);                       // cekaj 500ms
127     }
128 }
129
130 // spaja 8 charova u jedan
131 unsigned char spoji_red(unsigned char red[])
132 {
133     unsigned char rez = 0;
134     int i = 0;
135     for(; i < 8 ; ++i)
136         rez |= red[i] << 7-i;
137 return rez;
138 }
139
140
141 // spaja 8 charova u jedan, ali naopako
142 unsigned char spoji_red_naopako(unsigned char red[])
143 {
144     unsigned char rez = 0;
145     int i = 0;
146     for(i = 0; i<8; ++i)
147         rez |= red[i] << i;
148 return rez;
149 }
150
151 // iscrtava trenutnu konfiguraciju ploce na display
152 void osvjezi_displej()
153 {
154     unsigned char i = 0, j = 0, c = 0;
155     for(; i < 8; ++i) // gornja polovina
156     {
157         for(j = 0; j < 8; ++j)   // prva matrica (gore-lijevo)
158             red[j] = trenutno[i][j];
159
160         c = spoji_red(red);
161         pisi(i+1,c,1);
162
163         for(j = 8; j < 16; ++j) // druga matrica (gore-desno)
164             red[j-8] = trenutno[i][j];
165
166         c = spoji_red_naopako(red);
167         pisi(8-i,c,2); // pisemo od zadnjeg reda prema prvom
168     }
169
170     int b = 0;
171     for(; i < 16; ++i) // donja polovina
172     {
173
174         for(j = 0; j < 8; ++j) // treca matrica (dole-lijevo)
175             red[j] = trenutno[i][j];
176
177         c = spoji_red(red);
```

```
178         pisi(i-7,c,3);
179
180         for(j = 8; j < 16; ++j) // cetvrta matrica (dole-desno)
181             red[j-8] = trenutno[i][j];
182
183         c = spoji_red_naopako(red);
184         pisi(i-b,c,4); // pisemo od zadnjeg reda prema prvom
185         b+=2;
186     }
187 }
188
189 // racuna broj zivih komsija celije
190 int dajKomsije(short int x, short int y)
191 {
192     int rez = 0;
193     if (x != 0 && y != 0  && staro[x - 1][y - 1] == 1) // iznad-lijevo
194         ++rez;
195     if (x != 0 && staro[x - 1][y] == 1) // iznad
196         ++rez;
197     if (x != 0 && y != 15 && staro[x - 1][y + 1] == 1) // iznad-desno
198         ++rez;
199     if (y != 0  && staro[x][y - 1] == 1) // lijevo
200         ++rez;
201     if (y != 15 && staro[x][y + 1] == 1) // desno
202         ++rez;
203     if (x != 15 && y != 0 && staro[x + 1][y - 1] == 1) // ispod-lijevo
204         ++rez;
205     if (x != 15 && staro[x + 1][y] == 1) // ispod
206         ++rez;
207     if (x != 15 && y != 15 && staro[x + 1][y + 1] == 1) // ispod-desno
208         ++rez;
209     return rez;
210 }
211
212 // generise narednu generaciju celija na osnovu prethodne
213 void update_game()
214 {
215     for (int i = 0; i < 16; ++i)
216         for (int j = 0; j < 16; ++j)
217             staro[i][j] = trenutno[i][j]; // prepisuj
218
219
220     for (int i = 0; i < 16; ++i)
221     {
222         for (int j = 0; j < 16; ++j)
223         {
224                 a = dajKomsije(i, j);
225                 if (staro[i][j] == 1 && (a < 2 || a > 3))
226                     trenutno[i][j] = 0; // pravila 1 i 3
227                 else if (staro[i][j] == 0 && (a == 3))
228                     trenutno[i][j] = 1; // pravilo 4
229                 else if (staro[i][j] == 1 && (a == 2 || a == 3))
230                     trenutno[i][j] = 1; // pravilo 2
231         }
232     }
233     osvjezi_displej();
234
235     for (int i = 0; i < 16; ++i)
236         for (int j = 0; j < 16; ++j)
237             staro[i][j] = trenutno[i][j]; // prepisuj
238 }
239
```

```
240 // upravljanje igrom
241 void threadGameOfLife()
242 {
243     while(1) // vrti beskonacno update
244     {
245         wait(vrijeme_cekanja);
246         if(pauza == 1) continue; // ako je igra pauzirana, ne radi nista
247         update_game(); // radi update
248     }
249 }
250
251 // postavlja boju teksta u putty na zelenu (koju korisitmo za obavjestenja)
252 void zelena()
253 {
254  pc.printf("\033[0;32m");
255 }
256
257 // postavlja boju teksta u putty na crvenu (koju koristimo za upozorenja)
258 void crvena()
259 {
260     pc.printf("\033[0;31m");
261 }
262
263 // vraca boju teksta na pocetnu
264 void defaultna()
265 {
266  pc.printf("\033[0m");
267 }
268
269 // ispisuje informacije o kreatorima
270 void credits()
271 {
272     zelena();
273     pc.printf("\r\n==========================================================\r\n");
274     pc.printf("\r\nPravili: Edvin Teskeredzic (17333) i Kenan Karahodzic (17853)\r\nCitav kod
    dostupan na: https://github.com/eteskeredzic/Embedded-Game-of-Life\r\n");
275     pc.printf("\r\n==========================================================\r\n");
276     defaultna();
277 }
278
279 // generise pseudonasumicnu konfiguraciju celija i iscrtava to na display
280 void nasumicno()
281 {
282     int i  = 0, j = 0;
283     for(i = 0; i<16;++i)
284         for(j = 0; j<16; ++j)
285             if(dajNasumicanBroj(&SEED, 10) < 5) trenutno[i][j] = 0;
286             else trenutno[i][j] = 1;
287     osvjezi_displej();
288 }
289
290 // ubrzava mijenjanje generacija za 0.1 sec (max brzina = 0.1sec)
291 void ubrzaj()
292 {
293     vrijeme_cekanja-=0.1;
294     if(vrijeme_cekanja <= 0.1) vrijeme_cekanja = 0.1;
295 }
296
297 // usporava mijenjanje generacija za 0.1 sec (min brzina = 1.0sec)
298 void uspori()
299 {
300     vrijeme_cekanja+=0.1;
```

```
301     if(vrijeme_cekanja >= 1.0) vrijeme_cekanja = 1;
302 }
303
304 // provjerava da li je karakter cifra
305 int isNum(char c)
306 {
307     int a = c-48;
308     return (a >= 0 && a < 9) ? 1 : 0;
309 }
310
311 // otvara korisniku dijalog za unosenje koordinata diode koju mijenja (0,0 = gornji lijevi ugao,
        15,15 = donji desni ugao)
312 void mijenjajDiodu()
313 {
314     pc.printf("\r\nUnesite koordinate celije kojoj mijenjate stanje - U FORMATU 'X,Y;'\r\n");
315     int xcoord = 0, ycoord = 0, i = 0;
316     char c = '0';
317     char s[10] = {'\0'};
318     while(1)
319     {
320         c = pc.getc();
321         pc.putc(c);
322         if(c == ';' || i == 9) break;
323         s[i++] = c;
324     }
325     if(isNum(s[0]) == 1 && s[1] == ',' && isNum(s[2]) == 1 && s[3] == 0) // slucaj B,B
326     {
327         xcoord = s[0]-48;
328         ycoord = s[2]-48;
329         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
330         {
331             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
332         }
333         trenutno[xcoord][ycoord] ^= 1;
334         osvjezi_displej();
335         return;
336     }
337     if(isNum(s[0]) == 1 && s[1] == ',' && isNum(s[2]) == 1 && isNum(s[3]) == 1 && s[4] == 0) //
        slucaj B,BB
338     {
339         xcoord = s[0]-48;
340         ycoord = (s[2]-48) * 10 + (s[3]-48);
341         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
342         {
343             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
344         }
345         trenutno[xcoord][ycoord] ^= 1;
346         osvjezi_displej();
347         return;
348     }
349     if(isNum(s[0]) == 1 && isNum(s[1]) == 1 && s[2] == ',' && isNum(s[3]) == 1 && s[4] == 0) //
        slucaj BB,B
350     {
351         xcoord = (s[0]-48)*10 + (s[1]-48);
352         ycoord = s[3]-48;
353         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
354         {
355             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
356         }
357         trenutno[xcoord][ycoord] ^= 1;
358         osvjezi_displej();
359         return;
```

```
360     }
361     if(isNum(s[0]) == 1 && isNum(s[1]) == 1 && s[2] == ',' && isNum(s[3]) == 1 && isNum(s[4]) ==
        1 && s[5] == 0) // slucaj BB,BB
362     {
363         xcoord = (s[0]-48)*10 + (s[1]-48);
364         ycoord = (s[3]-48)*10 + (s[4]-48);
365         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
366         {
367             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
368         }
369         trenutno[xcoord][ycoord] ^= 1;
370         osvjezi_displej();
371         return;
372     }
373     crvena();
374     pc.printf("\r\nPogresan unos!\r\n");
375     defaultna();
376 }
377
378 // ispisuje glavni meni i daje korisniku opcije
379 void meni()
380 {
381
382     pc.printf("\r\n|-------------------------CONWAY'S GAME OF LIFE-------------------------|\r\
        n");
383     pc.printf("| Za mijenjanje stanja diode, pritisnite 1                                |\r\n");
         // gotovo
384     pc.printf("| Za povecanje brzine ispisa, pritisnite 2"); zelena(); pc.printf(" (trenutna
        brzina %.1f s)        ", vrijeme_cekanja); defaultna(); pc.printf("|\r\n"); // gotovo
385     pc.printf("| Za smanjenje brzine ispisa, pritisnite 3"); zelena(); pc.printf(" (trenutna
        brzina %.1f s)        ", vrijeme_cekanja); defaultna(); pc.printf("|\r\n");// gotovo
386     pc.printf("| Za pauziranje/ponovno pokretanje, pritisnite 4"); zelena(); pc.printf(" (
        trenutno pauzirano: %s) ", pauza == 1 ? "DA" : "NE"); defaultna(); pc.printf("|\r\n");//
        gotovo
387     pc.printf("| Za ciscenje ploce, pritisnite 5                                         |\r\n");
         // gotovo
388     pc.printf("| Za nasumicnu pocetnu konfiguraciju, pritisnite 6                        |\r\n");
         // gotovo
389     pc.printf("| Za informacije o kreatorima, pritisnite 7                               |\r\n");
         // gotovo
390     pc.printf("|-------------------------------------------------------------------------|\r\n");
391
392
393     while(1)
394     {
395         pc.printf("Unesite redni broj komande: ");
396         char c = pc.getc();
397         if(c == '1')
398         {
399             if(pauza == 0)
400             {
401                 crvena();
402                 pc.printf("\r\nDozvoljeno samo dok je igra pauzirana!\r\n");
403                 defaultna();
404                 break;
405             }
406                 mijenjajDiodu();
407                 break;
408             }
409             else if(c == '2')
410             {
411                 ubrzaj();
```

```
412                 break;
413             }
414             else if(c == '3')
415             {
416                 uspori();
417                 break;
418             }
419             else if(c == '4')
420             {
421                 if(pauza == 1) pauza = 0;
422                 else pauza = 1;
423                 break;
424             }
425             else if(c == '5')
426             {
427                 ocistiSve();
428                 break;
429             }
430             else if(c == '6')
431             {
432                 if(pauza == 0)
433                 {
434                     crvena();
435                     pc.printf("\r\nDozvoljeno samo dok je igra pauzirana!\r\n");
436                     defaultna();
437                     break;
438                 }
439                 nasumicno();
440                 break;
441             }
442             else if(c == '7')
443             {
444                 credits();
445                 break;
446             }
447             else
448             {
449                 crvena();
450                 pc.printf("\r\nNe postoji komanda! Pokusaj ponovo (i ovaj put unesi kako treba).
    . .\r\n");
451                 defaultna();
452             }
453     }
454 }
455
456 int main()
457 {
458     cs1 = 1;
459     cs2 = 1;
460     cs3 = 1;
461     cs4 = 1;
462     spi.format(8,0);
463     spi.frequency(1000000);
464     init();
465     thread.start(threadGameOfLife);
466     while(1)
467     {
468      meni();
469     }
470 }
```

Kod - bez refaktoringa

```
/*
* PROJEKTNI ZADATAK IZ UGRADBENIH SISTEMA - AKADEMSKA GODINA 2017./2018.
* KREATORI: Edvin Teskeredzic i Kenan Karahodzic;
* Bilo kakvo koristenje ovog koda bez znanja kreatora nije dozvoljeno!
*/

#include "mbed.h"
#include "rtos.h"
//#include <cstdio>
#define dp23 P0_0

// REGISTRI ZA DISPLAY
#define NO_OP_REG           0x00    // ???
#define DIGIT_0_REG         0x01    // prvi red
#define DIGIT_1_REG         0x02    // drugi red
#define DIGIT_2_REG         0x03    // treci red
#define DIGIT_3_REG         0x04    // cetvrti red
#define DIGIT_4_REG         0x05    // peti red
#define DIGIT_5_REG         0x06    // sesti red
#define DIGIT_6_REG         0x07    // sedmi red
#define DIGIT_7_REG         0x08    // osmi red
#define DECODE_MODE_REG     0x09    // ???
#define INTENSITY_MODE_REG  0x0A    // intenzitet kojim diode svijetle
#define SCAN_LIMIT_REG      0x0B    // koliko smije dioda goriti istovremeno (8)
#define SHUTDOWN_REG        0x0C    // ako je setovan, displaj je upaljen
#define DISPLAY_TEST_REG    0x0F    // odlucuje izmedju normal mode i test mode


// KOMUNIKACIJA
Serial pc(USBTX, USBRX);

// DISPLAY-i
DigitalOut cs1(PTD0);
DigitalOut cs2(PTC16);
DigitalOut cs3(PTD5);
DigitalOut cs4(PTC13);
SPI spi(PTD2, PTD3, PTD1); // MOSI, MISO, SCLK

// MATRICE ZA GAME OF LIFE
char trenutno[16][16] = { { 0 } };
char staro[16][16] = { { 0 } };
unsigned char red[8] = {0}; // ovo koristimo za ispis u jednu matricu

// TICKERI TIMER THREAD
Ticker tDisplej;/// OVO MOZDA NECE TREBATI OBZIROM DA DISPLAY IMA INTERNO REGISTRE KOJI CUVAJU
    STANJE
Timer tajmer;
Thread thread;

// MISC.
unsigned char pauza = 1; // oznacava da li se vrsi prelazak u novu generaciju - na pocetku je 1
    jer sistem miruje (ceka se input korisnika)
float vrijeme_cekanja = 0.3; // oznacava koliko dugo cekamo na narednu generaciju
unsigned char smijem_crtati = 1; // oznacava da li funkcija za pisanje po matricama smije raditi
char a = 0; // za komsije

// ova funkcija vraca nasumican broj svaki put kada bude pozvana - pri tome ne koristeci rand
    funkciju, sto uklanja potrebu za cstdlib bibliotekom!
unsigned int SEED = time(NULL);
```

```
int dajNasumicanBroj(unsigned int *pSEED, int granica){

*pSEED = (*pSEED * 48271) % 2147483647;

return *pSEED % granica;
}

// pise na odgovarajuci display zadan kao parametar
void pisi(unsigned char registar, unsigned char vrijednost, unsigned char redni){

    switch(redni){
        case 1: // prvi display
            cs1 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs1 = 1;
            break;
        case 2: // drugi display
            cs2 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs2 = 1;
            break;
        case 3: // treci display
            cs3 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs3 = 1;
            break;
        case 4: // cetvrti display
            cs4 = 0;
            spi.write(registar);
            spi.write(vrijednost);
            cs4 = 1;
            break;
        default:
            break;
    }
    wait(0.01);
}

// gasi matricu
void ocistiDisplej(int displej){

   for(int i = 1; i <= 8; ++i)
     pisi(i, 0, displej);
}

// cisti sve 4 matrica
void ocistiSve(){
    for(int i = 1; i<=4; ++i)
        ocistiDisplej(i);
    for(int i = 0; i<16;++i) for(int j = 0; j<16;++j) trenutno[i][j] = 0;
}

// postavljanje defaultnih vrijednosti za sve 4 matrice
void init(){
    for(int i = 1; i <= 4; ++i){
        pisi(DECODE_MODE_REG, 0x00, i);    // decode mode ne koristimo
        pisi(INTENSITY_MODE_REG, 0x0f, i); // stavi duty cycle na maksimalnu mogucu vrijednost (
    osvijetljenost)
```

```
117         pisi(SCAN_LIMIT_REG,0x07, i);       // upisemo broj 7 jer zelimo da palimo sve do reda
    indeksa 7 (a to je osmi red)
118         pisi(SHUTDOWN_REG,0x01, i);         // stavlja se u normal opmode, jer ne testiramo nista
119         //pisi(0xff,0, i);                    // ovo po datasheet mora biti 0
120         pisi(0x0F, 0x0F, i);  /// NOVO ENABLE DISPLAY TEST
121         wait_ms(500);         /// NOVO 500 ms delay
122         ocistiDisplej(i);                           //  ugasi sve diode
123         pisi(0x0F, 0x00, i);  /// NOVO DISABLE DISPLAY TEST
124         wait_ms(500);
125     }
126 }
127
128 // spaja 8 charova u jedan
129 unsigned char spoji_red(unsigned char red[]){
130     unsigned char rez = 0;
131     int i = 0;
132     for(; i < 8 ; ++i)
133         rez |= red[i] << 7-i;
134 return rez;
135 }
136
137
138 // spaja 8 charova u jedan, ali naopako
139 unsigned char spoji_red_naopako(unsigned char red[]){
140     unsigned char rez = 0;
141     int i = 0;
142     for(i = 0; i<8; ++i)
143         rez |= red[i] << i;
144 return rez;
145 }
146
147 void osvjezi_displej(){
148     unsigned char i = 0, j = 0, c = 0;
149     for(; i < 8; ++i){ // gornja polovina
150
151         for(j = 0; j < 8; ++j) red[j] = trenutno[i][j]; // prva matrica
152
153         c = spoji_red(red);
154         pisi(i+1,c,1);
155
156         for(j = 8; j < 16; ++j) red[j-8] = trenutno[i][j]; // druga matrica
157
158         c = spoji_red_naopako(red);
159         pisi(8-i,c,2); // pisemo od zadnjeg reda prema prvom
160     }
161     int b = 0;
162     for(; i < 16; ++i){ // donja polovina
163
164         for(j = 0; j < 8; ++j) red[j] = trenutno[i][j]; // treca matrica
165
166         c = spoji_red(red);
167         pisi(i-7,c,3);
168         for(j = 8; j < 16; ++j) red[j-8] = trenutno[i][j]; // cetvrta matrica
169
170         c = spoji_red_naopako(red);
171         pisi(i-b,c,4); // pisemo od zadnjeg reda prema prvom
172         b+=2;
173     }
174 }
175
176 // racuna broj komsija celije
177 int dajKomsije(short int x, short int y){
```

```
178     int rez = 0;
179     if (x != 0 && y != 0  && staro[x - 1][y - 1] == 1) // iznad-lijevo
180         ++rez;
181     if (x != 0 && staro[x - 1][y] == 1) // iznad
182         ++rez;
183     if (x != 0 && y != 15 && staro[x - 1][y + 1] == 1) // iznad-desno
184         ++rez;
185     if (y != 0  && staro[x][y - 1] == 1) // lijevo
186         ++rez;
187     if (y != 15 && staro[x][y + 1] == 1) // desno
188         ++rez;
189     if (x != 15 && y != 0 && staro[x + 1][y - 1] == 1) // ispod-lijevo
190         ++rez;
191     if (x != 15 && staro[x + 1][y] == 1) // ispod
192         ++rez;
193     if (x != 15 && y != 15 && staro[x + 1][y + 1] == 1) // ispod-desno
194         ++rez;
195     return rez;
196 }
197
198 // generise narednu generaciju celija na osnovu prethodne
199 void update_game(){
200
201     for (int i = 0; i < 16; ++i)
202         for (int j = 0; j < 16; ++j)
203             staro[i][j] = trenutno[i][j]; // prepisuj
204
205
206         for (int i = 0; i < 16; ++i) {
207             for (int j = 0; j < 16; ++j) {
208                     a = dajKomsije(i, j);
209                     if (staro[i][j] == 1 && (a < 2 || a > 3))
210                         trenutno[i][j] = 0; // pravila 1 i 3
211                     else if (staro[i][j] == 0 && (a == 3))
212                         trenutno[i][j] = 1; // pravilo 4
213                     else if (staro[i][j] == 1 && (a == 2 || a == 3))
214                         trenutno[i][j] = 1;
215             }
216         }
217       osvjezi_displej();
218
219         for (int i = 0; i < 16; ++i)
220             for (int j = 0; j < 16; ++j)
221                 staro[i][j] = trenutno[i][j]; // prepisuj
222
223
224 }
225
226 // upravljanje igrom
227 void threadGameOfLife(){
228     while(1){ // vrti beskonacno update
229        wait(vrijeme_cekanja);
230        if(pauza == 1) continue;
231        update_game(); // radi update
232     }
233 }
234
235
236 void zelena(){
237  pc.printf("\033[0;32m");
238 }
239
```

```
240 void crvena(){
241     pc.printf("\033[0;31m");
242 }
243
244 void defaultna(){
245  pc.printf("\033[0m");
246 }
247
248 void credits(){
249     zelena();
250     pc.printf("\r\n==========================================================================\r\n");
251     pc.printf("\r\nPravili: Edvin Teskeredzic (17333) i Kenan Karahodzic (17853)\r\nCitav kod
      dostupan na: https://github.com/eteskeredzic/Embedded-Game-of-Life\r\n");
252     pc.printf("\r\n==========================================================================\r\n");
253     defaultna();
254   /* pauza = 1;
255     trenutno[0][3] = trenutno[0][4]=trenutno[1][2]=trenutno[1][5]=trenutno[1][3]=trenutno[1][4]=
      trenutno[2][1]=trenutno[2][2]=trenutno[2][3]=trenutno[2][4]=
256 trenutno[2][5]=trenutno[2][6]=trenutno[3][0]=trenutno[3][1]=trenutno[3][3]=trenutno[3][4]=
      trenutno[3][6]=trenutno[3][7]=
257 trenutno[4][0]=trenutno[4][1]=trenutno[4][2]=trenutno[4][3]=trenutno[4][4]=trenutno[4][5]=
      trenutno[4][6]=trenutno[4][7]= trenutno[5][1] = trenutno[5][3] = trenutno[5][4]
258 =trenutno[5][6]=trenutno[6][0]=trenutno[6][7]=trenutno[7][1]=trenutno[7][6]=1;
259 osvjezi_displej();*/
260 }
261
262 void nasumicno(){
263     int i  = 0, j = 0;
264     for(i = 0; i<16;++i)
265         for(j = 0; j<16; ++j)
266             if(dajNasumicanBroj(&SEED, 10) < 5) trenutno[i][j] = 0;
267             else trenutno[i][j] = 1;
268     osvjezi_displej();
269 }
270
271 void ubrzaj(){
272     vrijeme_cekanja-=0.1;
273     if(vrijeme_cekanja <= 0.1) vrijeme_cekanja = 0.1;
274 }
275
276 void uspori(){
277     vrijeme_cekanja+=0.1;
278     if(vrijeme_cekanja >= 1.0) vrijeme_cekanja = 1;
279 }
280
281 int isNum(char c){
282     int a = c-48;
283     return (a >= 0 && a < 9) ? 1 : 0;
284 }
285
286 void mijenjajDiodu(){
287     pc.printf("\r\nUnesite koordinate celije kojoj mijenjate stanje - U FORMATU 'X,Y;'\r\n");
288     int xcoord = 0, ycoord = 0, i = 0;
289     char c = '0';
290     char s[10] = {'\0'};
291     while(1)
292     {
293         c = pc.getc();
294         pc.putc(c);
295         if(c == ';' || i == 9) break;
296         s[i++] = c;
297     }
```

```
298     if(isNum(s[0]) == 1 && s[1] == ',' && isNum(s[2]) == 1 && s[3] == 0) // slucaj B,B
299     {
300         xcoord = s[0]-48;
301         ycoord = s[2]-48;
302         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
303         {
304             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
305         }
306         trenutno[xcoord][ycoord] ^= 1;
307         osvjezi_displej();
308         return;
309     }
310     if(isNum(s[0]) == 1 && s[1] == ',' && isNum(s[2]) == 1 && isNum(s[3]) == 1 && s[4] == 0) //
        slucaj B,BB
311     {
312         xcoord = s[0]-48;
313         ycoord = (s[2]-48) * 10 + (s[3]-48);
314         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
315         {
316             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
317         }
318         trenutno[xcoord][ycoord] ^= 1;
319         osvjezi_displej();
320         return;
321     }
322     if(isNum(s[0]) == 1 && isNum(s[1]) == 1 && s[2] == ',' && isNum(s[3]) == 1 && s[4] == 0) //
        slucaj BB,B
323     {
324         xcoord = (s[0]-48)*10 + (s[1]-48);
325         ycoord = s[3]-48;
326         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
327         {
328             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
329         }
330         trenutno[xcoord][ycoord] ^= 1;
331         osvjezi_displej();
332         return;
333     }
334     if(isNum(s[0]) == 1 && isNum(s[1]) == 1 && s[2] == ',' && isNum(s[3]) == 1 && isNum(s[4]) ==
        1 && s[5] == 0) // slucaj BB,BB
335     {
336         xcoord = (s[0]-48)*10 + (s[1]-48);
337         ycoord = (s[3]-48)*10 + (s[4]-48);
338         if(xcoord < 0 || xcoord > 15 || ycoord < 0 || ycoord > 15)
339         {
340             crvena(); pc.printf("\r\nGreska!\r\n"); defaultna(); return;
341         }
342         trenutno[xcoord][ycoord] ^= 1;
343         osvjezi_displej();
344         return;
345     }
346     crvena();
347     pc.printf("\r\nPogresan unos!\r\n");
348     defaultna();
349 }
350
351 // ispisuje glavni meni i daje korisniku opcije
352 void meni(){
353
354     pc.printf("\r\n|----------CONWAY'S GAME OF LIFE----------|\r\
        n");
```

```
355     pc.printf("| Za mijenjanje stanja diode, pritisnite 1                                  |\r\n");
         //
356     pc.printf("| Za povecanje brzine ispisa, pritisnite 2"); zelena(); pc.printf(" (trenutna
        brzina %.1f s)        ", vrijeme_cekanja); defaultna(); pc.printf("|\r\n"); // gotovo
357     pc.printf("| Za smanjenje brzine ispisa, pritisnite 3"); zelena(); pc.printf(" (trenutna
        brzina %.1f s)        ", vrijeme_cekanja); defaultna(); pc.printf("|\r\n");// gotovo
358     pc.printf("| Za pauziranje/ponovno pokretanje, pritisnite 4"); zelena(); pc.printf(" (
        trenutno pauzirano: %s) ", pauza == 1 ? "DA" : "NE"); defaultna(); pc.printf("|\r\n");//
        gotovo
359     pc.printf("| Za ciscenje ploce, pritisnite 5                                           |\r\n");
360     pc.printf("| Za nasumicnu pocetnu konfiguraciju, pritisnite 6                          |\r\n");
         // gotovo
361     pc.printf("| Za informacije o kreatorima, pritisnite 7                                 |\r\n");
         // gotovo
362     pc.printf("|-------------------------------------------------------------------------------|\r\n");
363
364
365     while(1){
366         pc.printf("Unesite redni broj komande: ");
367         char c = pc.getc();
368             if(c == '1'){
369                 if(pauza == 0){
370                     crvena();
371                     pc.printf("\r\nDozvoljeno samo dok je igra pauzirana!\r\n");
372                     defaultna();
373                     break;
374                 }
375                 mijenjajDiodu();
376                 break;
377             }
378             else if(c == '2'){
379                 ubrzaj();
380                 break;
381             }
382             else if(c == '3'){
383                 uspori();
384                 break;
385             }
386             else if(c == '4'){
387                 if(pauza == 1) pauza = 0; else pauza = 1;
388                 break;
389             }
390             else if(c == '5'){ ocistiSve(); break; }
391             else if(c == '6'){
392                 if(pauza == 0){
393                     crvena();
394                     pc.printf("\r\nDozvoljeno samo dok je igra pauzirana!\r\n");
395                     defaultna();
396                     break;
397                 }
398                 nasumicno();
399                 break;
400             }
401             else if(c == '7'){
402                 credits();
403                 break;
404             }
405             else{
406                     crvena();
407                     pc.printf("\r\nNe postoji komanda! Pokusaj ponovo (i ovaj put unesi kako
    treba). . .\r\n");
408                     defaultna();
```

```
409                     }
410     }
411 }
412
413 int main()
414 {
415     cs1 = 1;
416     cs2 = 1;
417     cs3 = 1;
418     cs4 = 1;
419     spi.format(8,0);
420     spi.frequency(1000000);
421     init();
422     tajmer.start();
423     thread.start(threadGameOfLife);
424     while(1) meni();
425
426 }
```