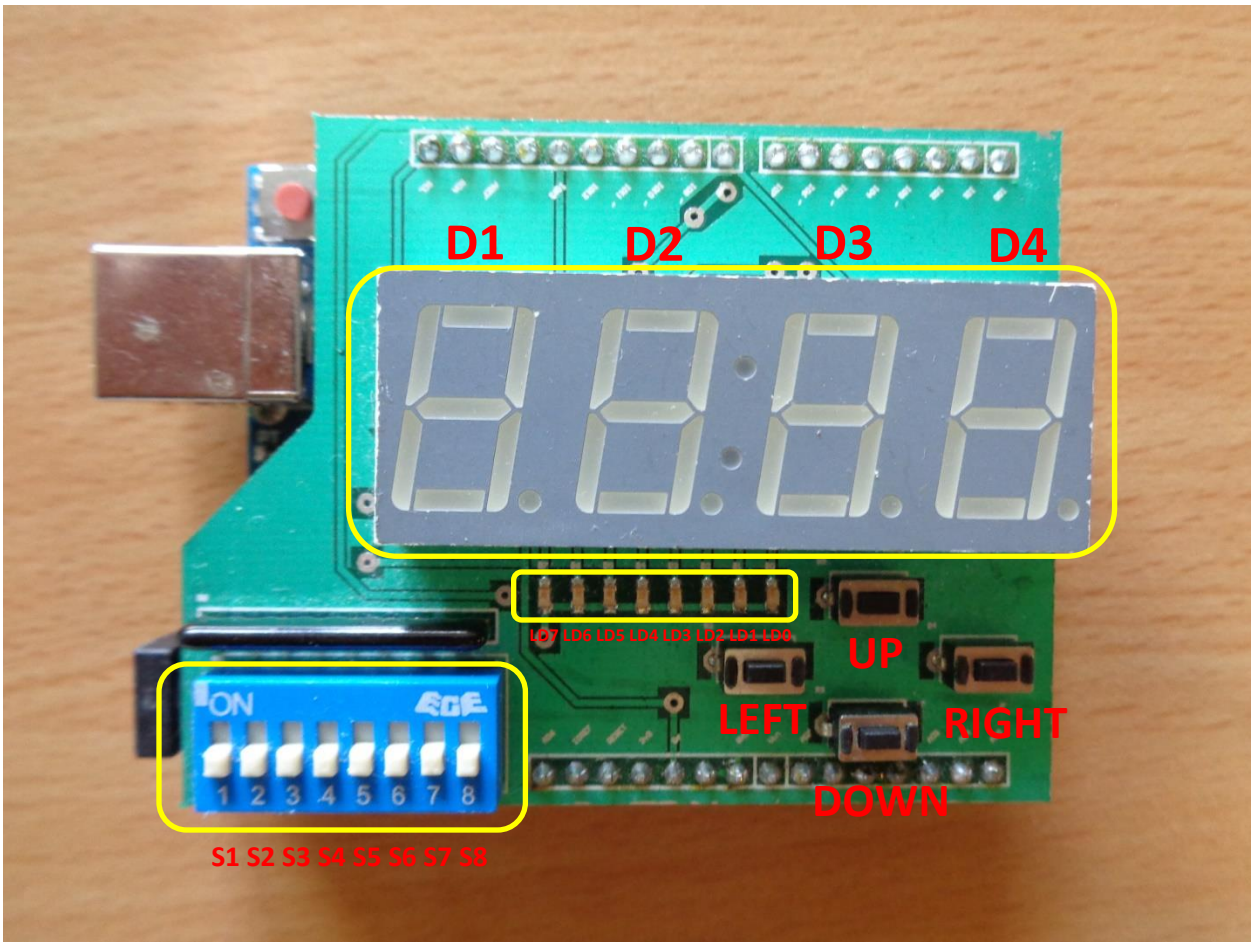
Arduino PLS7 ekspanziona ploča (4x7SEG + LED+ tasteri + prekidači)

Izgled PLS7 ekspanzione ploče



PLS7 ekspanziona ploča sadrži sledeće periferijske uređaje:

- Niz od 4 sedmosegmentna displeja (**D1 - D4**)
- Niz od 8 LED dioda (**LD7 – LD0**)
- 4 tastera (**UP, DOWN, LEFT, RIGHT**)
- Niz od 8 prekidača (**S1 – S8**)

Biblioteka *PLS7shield.h*

Upravljanje periferijama na ekspanzionoj ploči omogućeno je pomoću funkcija biblioteke *PLS7shield.h* koju je moguće preuzeti na adresi:

http://www.elektronika.ftn.uns.ac.rs/images/UDIME/PLS7shield.zip

Biblioteka se uključuje u *Arduino* razvojno okruženje pomoću opcije *Sketch -> Import Library -> Add Library*. Nakon otvaranja dijaloga, potrebno je odabrati snimljenu datoteku *PLS7shield.zip*, nakon čega je biblioteka spremna za korišćenje.

Program koji koristi ovu biblioteku mora da sadrži njen poziv, putem direktive **#include**. Sve funkcije za rad sa periferijama realizovane su u okviru klase **PLS7shield**, koju je potrebno instancirati na početku programa, kreiranjem odgovarajućeg objekta. Prilikom kreiranja objekta, obavljaju se sve potrebne inicijalizacije hardverskih modula mikrokontrolera koji se koriste za upravljanjem periferijama[1].

Kostur programa:

```
#include <PLS7shield.h>

PLS7shield shield;   //kreira se objekat shield, preko kojeg
                     //se pozivaju funkcije za upravljanje periferijama

void setup()
{
    //...
}

void loop()
{
    //...
}
```

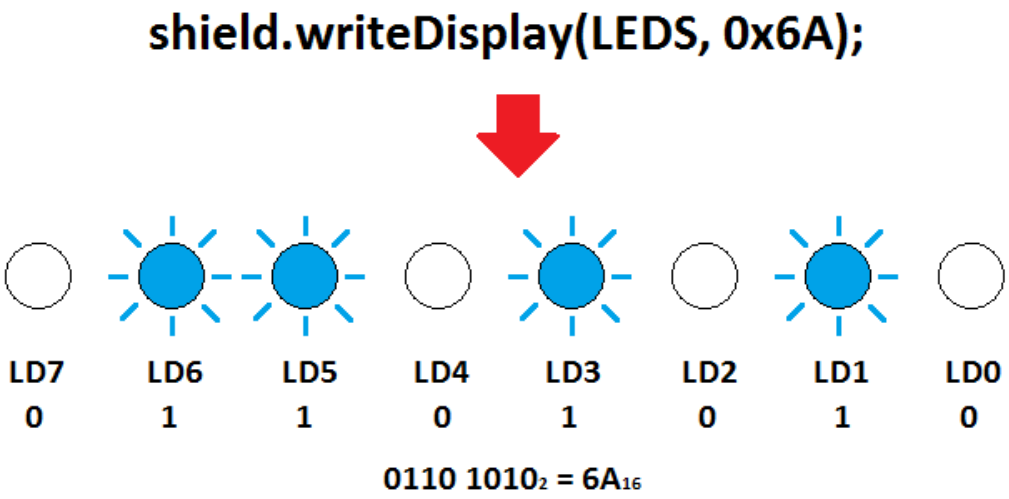
[1] Nakon inicijalizacije periferija prilikom instanciranja klase *KELshield*, oba pina koja se koriste za serijsku komunikaciju postaju zauzeta, tako da je od tog momenta serijska komunikacija onemogućena.

LED diode

Nizom od 8 LED dioda (***LD7*** - ***LD0***) upravlja se pozivom funkcije:

```
shield.writeDisplay(LEDS, value);
```

Prvi parametar (**LEDS**) je numerička konstanta sa vrednošću 0, koja daje mikrokontroleru do znanja da upiše vrednost **value** u bafer koji upravlja stanjem LED dioda. Vrednost parametra **value** je tipa byte (8-bitni neoznačeni tip), pri čemu svaki pojedinačan bit određuje stanje LED diode na odgovarajućoj poziciji. Na slici je prikazan primer poziva funkcije koja istovremeno uključuje diode **LD6**, **LD5**, **LD3** i **LD1**[2].



Vrednost koja je upisana u bafer koji čuva stanja LED dioda zadržava se sve do upisa nove vrednosti. Ovu vrednost je moguće očitati pozivom funkcije:

```
value = shield.readDisplay(LEDS);
```

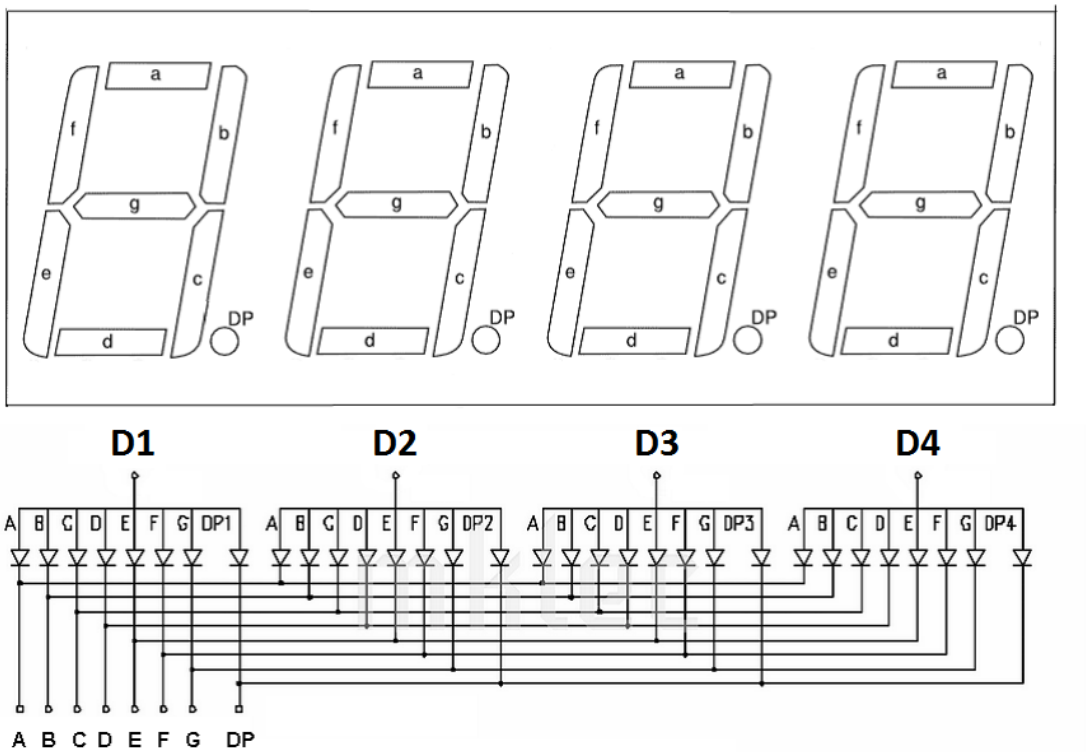
[2] Parametar value je moguće zadati i u binarnom formatu. U primeru sa slike, ekvivalentan poziv funkcije bio bi *shield.writeDisplay(LEDS, B01101010);*

Primer:

```
#include <PLS7shield.h>

PLS7shield shield; //instanciranje klase PLS7shield

void setup()
{
    shield.writeDisplay(LEDS, B01010101);   //ukljucuju se LED diode
                                            //na parnim pozicijama
}

void loop()
{
    byte stanje_LED = shield.readDisplay(LEDS); //ocitavanje sadrzaja bafera
    shield.writeDisplay(LEDS, ~stanje_LED);     //upis invertovane vrednosti
    delay(1000);                                //pauza 1s
}
```

LED displej 4x7SEG

Na ploči se nalazi komponenta koja sadrži četiri sedmosegmentna displeja sa decimalnom tačkom, integrisana u zajedničko kućište. LED diode koje čine segmente displeja povezane su u spoju sa zajedničkom anodom, kao što je prikazano na internoj šemi displeja:



Na različitim displejima (***D1-D4***), anode odgovarajućih segmenata (A-DP) su kratko spojene, a katode su razdvojene. Na red sa anodama displeja ***D1-D4*** povezani su prekidači pomoću kojih je moguće prekidati struju i time uključivati/isključivati svaki displej zasebno. Katode su povezane na iste izlazne pinove Arduina koji se koriste za upravljanje nizom LED dioda. Logika upravljanja realizuje se tzv. vremenskim multipleksiranjem: osvežavanje displeja vrši se u pravilnim vremenskim intervalima. Perioda osvežavanja displeja T podeljena je na 5 intervala jednakog trajanja: T0 = T1 = T2 = T3 = T4 = T/5. Tokom svakog od ovih intervala, zatvoren je tačno jedan od prekidača, koji dozvoljava uključenje odgovarajućeg displeja, odnosno niza LED dioda. Sistemski softver u okviru biblioteke *PLS7shield.h* obavlja osvežavanje automatski u intervlima trajanja 5ms (odnosno frekvencijom 200Hz), što je dovoljno visoka frekvencija da se stekne utisak da su displeji uključeni istovremeno.

Za svaki displej postoji zaseban bafer koji sadrži osmobitnu vrednost koja definiše stanja pojedinih segmenata. Upis, odnosno čitanje vrednosti ovih bafera vrši se pomoću istih funkcija kao i u slučaju niza LED dioda:

```
shield.writeDisplay(display, value);

value = shield.readDisplay(display);
```

Razlika u načinu korišćenja ovih funkcija je u tome što se kao parametar **display**, umesto **LEDS** prosleđuje odgovarajuća numerička konstanta **D1**, **D2**, **D3** ili **D4**, u zavisnosti od toga koji displej se adresira. Parametar **value** u pozivu funkcije *writeDisplay* određuje stanja pojedinih segmenata na displeju, u skladu sa sledećom tabelom:

parametar **display**		parametar **value**							
numerička konstanta	vrednost	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
LEDS	0	LD7	LD6	LD5	LD4	LD3	LD2	LD1	LD0
D1	1	A1	F1	B1	E1	D1	DP1	C1	G1
D2	2	A2	F2	B2	E2	D2	DP2	C2	G2
D3	3	A3	F3	B3	E3	D3	DP3	C3	G3
D4	4	A4	F4	B4	E4	D4	DP4	C4	G4

Primer:

```
#include <PLS7shield.h>

PLS7shield shield;

const byte simboli[] = {
    0xf3, 0x5b, 0xd8, 0x3b
};  //look-up tabela sa simbolima za ispis na displej (A,b,C,d)

void setup()
{
    shield.writeDisplay(D1, simboli[0]);    //D1 <- 'A'
    shield.writeDisplay(D2, simboli[1]);    //D2 <- 'b'
    shield.writeDisplay(D3, simboli[2]);    //D3 <- 'C'
    shield.writeDisplay(D4, simboli[3]);    //D4 <- 'd'
}

void loop()
{
}
```

Tasteri

Tasteri **LEFT**, **RIGHT**, **UP** i **DOWN** povezani su direktno na pinove Arduina. Očitavanje njihovih stanja vrši se pozivom funkcije:

```
value = shield.buttonState(button);
```

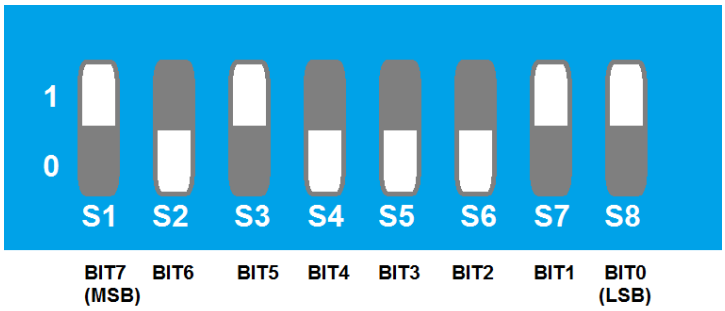
Kao parametar **button** prosleđuje se odgovarajuća numerička konstanta (**LEFT**, **RIGHT**, **UP** ili **DOWN**), u zavisnosti od toga koji se taster očitava. Povratna vrednost koja se u ovom primeru dodeljuje promenljivoj **value** može biti **HIGH** ukoliko je taster pritisnut u trenutku poziva funkcije, odnosno **LOW** ako je taster tada bio pušten.

Primer:

```
#include <PLS7shield.h>

PLS7shield shield;

void setup()
{
    byte i;

    for (i = D1; i <= D4; i++)
        shield.writeDisplay(i, 0x01); //'-'
}

void loop()
{
    if (shield.buttonState(LEFT) == HIGH)
        shield.writeDisplay(D1, 0xf3);  //'A'
    else
        shield.writeDisplay(D1, 0x01);  //'-'

    if (shield.buttonState(UP) == HIGH)
        shield.writeDisplay(D2, 0x5b);  //'b'
    else
        shield.writeDisplay(D2, 0x01);  //'-'

    if (shield.buttonState(DOWN) == HIGH)
        shield.writeDisplay(D3, 0xd8);  //'C'
    else
        shield.writeDisplay(D3, 0x01);  //'-'

    if (shield.buttonState(RIGHT) == HIGH)
        shield.writeDisplay(D4, 0x3b);  //'d'
    else
        shield.writeDisplay(D4, 0x01);  //'-'
}
```

Prekidači

Komponenta koja sadrži 8 prekidača u zajedničkom kućištu nije direktno povezana na pinove Arduina, pošto bi za tako nešto bilo potrebno 8 pinova, a povezivanjem već opisanih komponenti preostala su samo 3 raspoloživa pina. Stoga su prekidači povezani na 8-bitni pomerački registar sa mogućnošću paralelnog upisa. Komunikacija između Arduina i pomeračkog registra obavlja se putem serijskog protokola, koji podrazumeva prvo paralelni upis stanja prekidača u registar, a potom čitanje sadržaja registra bit po bit, tokom 8 uzastopnih perioda takta.



Na slici je prikazan fizički izgled kućišta komponente koja sadrži niz prekidača. Prekidači koji stoje u donjem položaju predstavljaju bite koji su u stanju logičke nule (**LOW**), a oni koji su u gornjem položaju su u stanju logičke jedinice (**HIGH**). Njihova stanja mogu biti očitana na dva načina. Prvi način omogućava očitavanje stanja pojedinačnih prekidača pozivom funkcije:

```
value = shield.switchState(sw);
```

Parametar **sw** je jedna od numeričkih konstanti (**S1**, **S2**, ..., **S8**) koja odgovara oznaci prekidača na kućištu komponente. Povratna vrednost koja će biti upisana u promenljivu **value** je **LOW** ili **HIGH**, u zavisnosti od položaja prekidača.

Drugi način omogućava istovremeno očitavanje stanja svih 8 prekidača:

```
value = shield.readSwitches();
```

U pitanju je funkcija koja vraća 8-bitnu vrednost u kojoj svaki bit predstavlja stanje prekidača na odgovarajućoj poziciji i koja se u ovom slučaju upisuje u promenljivu **value**. Pri tome treba voditi računa o tome da pozicije bita ne odgovaraju oznakama na kućištu komponente: Stanje krajnjeg levog prekidača označenog sa **S1** predstavljeno je najvišim bitom (tj. bitom na poziciji 7). Analogno tome, stanje krajnjeg levog prekidača označenog sa **S8** predstavljeno je najnižim bitom, koji se nalazi na na poziciji 0. Veza između oznaka prekidača na kućištu i njihovih bitskih pozicija u okviru povratne vrednosti funkcije takođe je prikazana na slici.

Primer:

```
#include <PLS7shield.h>

PLS7shield shield;

void setup()
{
}

void loop()
{
  byte sw = shield.readSwitches();  //citanje stanja prekidaca
  shield.writeDisplay(LEDS, sw);    //prikaz ocitanog stanja prekidaca
                                    //na LED diodama
}
```