**7 segment display**

[Diagram

Description automatically generated](https://osoyoo.com/wp-content/uploads/2017/08/4-digit-7-segment-LED-display-pinout.png)

[A picture containing text, clipart

Description automatically generated](https://osoyoo.com/wp-content/uploads/2017/08/7segment-1.jpg)

Vsak segment v prikazovalnem modulu je multipleksiran, kar pomeni, da si deli iste anodne priključne točke. Vsaka od štirih številk v modulu ima svojo skupno katodno povezovalno točko. To omogoča neodvisno vklop ali izklop vsake številke. Poleg tega ta tehnika multipleksiranja pretvori ogromno število pinov mikrokrmilnika, potrebnih za nadzor zaslona, v samo enajst ali dvanajst (namesto dvaintrideset)!

**Skupna anoda**

[Diagram

Description automatically generated](https://osoyoo.com/wp-content/uploads/2017/08/anode.jpg)

[A picture containing text, candelabrum

Description automatically generated](https://osoyoo.com/wp-content/uploads/2017/08/cathnode.jpg)**Skupna katoda**

**Tehnika multipleksiranja**

Torej, kako bomo na tem 4-mestnem zaslonu prikazali številko, kot je 1234? Za to bomo uporabili metodo, imenovano multipleksiranje. Multipleksiranje je preprosto – prikazujte eno številko naenkrat na prikazovalni enoti in zelo hitro preklapljajte med prikazovalnimi enotami. Zaradi fizioloških lastnosti vida človeško oko ne more ločiti, kateri zaslon je vklopljen/izklopljen. Človeško oko samo vizualizira, da so vse 4 zaslonske enote ves čas VKLOPLJENE. Recimo, da moramo prikazati 1234. Najprej vklopimo segmente, ki so pomembni za "1" in vklopimo 1. prikazovalno enoto. Nato pošljemo signale za prikaz "2", izklopimo 1. prikazovalno enoto in vklopimo 2. prikazovalno enoto. Ta postopek ponovimo za naslednji dve številki in preklapljanje med prikazovalnimi enotami mora biti opravljeno zelo hitro. Ker naše oči ne morejo izbrati spremembe, ki se večkrat zgodi na katerem koli predmetu v 1 sekundi, vidimo, da se na zaslonu istočasno pojavi 1234.

**Primer**

Vezje:

[A close-up of a computer

Description automatically generated with low confidence](https://osoyoo.com/wp-content/uploads/2017/08/4dss-74hc595-fritzing.jpg)

Vsi segmenti 7 segmentnega zaslona so povezani z vzporednimi podatkovnimi izhodnimi pini premičnega registra. Pini ure, latch in serijski podatki premičnega registra so povezani z digitalnimi pini arduino 6, 5 oziroma 4. Vsak od štirih skupnih anodnih pinov je povezan z ločenim pinom arduino (9,10,11 in 12) preko 200Ω upora za omejitev toka.

Algoritem:

#include "Timer.h" //include timer library

**Timer** t; // craete a timer object

long number = 0; //declear the variables

int first\_digit = 0;

int second\_digit = 0;

int third\_digit = 0;

int fourth\_digit = 0;

int timer\_event = 0;

int CA\_1 = 12;

int CA\_2 = 11;

int CA\_3 = 10;

int CA\_4 = 9;

int clk = 6;

int latch = 5;

int data = 4;

int count = 0;

int digits[4] ;

int CAS[4] = {12, 11, 10, 9};

byte numbers[10] {B11111100, B01100000, B11011010, B11110010, B01100110, B10110110, B10111110, B11100000, B11111110, B11110110};

//byte combinations for each number 0-9

void setup() {

**Serial**.begin(9600); //serial start and pin config

 pinMode(CA\_1, OUTPUT);

 pinMode(CA\_2, OUTPUT);

 pinMode(CA\_3, OUTPUT);

 pinMode(CA\_4, OUTPUT);

 pinMode(clk, OUTPUT);

 pinMode(latch, OUTPUT);

 pinMode(data, OUTPUT);

 digitalWrite(CA\_1, HIGH);

 digitalWrite(CA\_2, HIGH);

 digitalWrite(CA\_3, HIGH);

 digitalWrite(CA\_4, HIGH);

**Serial**.println("please Enter a number from 0 to 9999");

}

void loop() {

 t.update(); //timer update

 if (**Serial**.available()) { // read from serial

   t.stop(timer\_event); //stop timer if anythign to read

   cathode\_high(); // blank the screen

   String s = **Serial**.readString(); //read the serail value

   number = (long)s.toInt(); //convert it to int

   if (number > 9999) { //check the number is 0-9999

**Serial**.println("Please Enter Number Between 0 - 9999");

   } else {

     break\_number(number);

     timer\_event = t.every(1, display\_number); // start timer again

   }

 }

}

void break\_number(long num) { // seperate the input number into 4 single digits

 first\_digit = num / 1000;

 digits[0] = first\_digit;

 int first\_left = num - (first\_digit \* 1000);

 second\_digit = first\_left / 100;

 digits[1] = second\_digit;

 int second\_left = first\_left - (second\_digit \* 100);

 third\_digit = second\_left / 10;

 digits[2] = third\_digit;

 fourth\_digit = second\_left - (third\_digit \* 10);

 digits[3] = fourth\_digit;

}

void display\_number() { //scanning

 cathode\_high(); //black screen

 digitalWrite(latch, LOW); //put the shift register to read

 shiftOut(data, clk, LSBFIRST, numbers[digits[count]]); //send the data

 digitalWrite(CAS[count], LOW); //turn on the relevent digit

 digitalWrite(latch, HIGH); //put the shift register to write mode

 count++; //count up the digit

 if (count == 4) { // keep the count between 0-3

   count = 0;

 }

}

void cathode\_high() { //turn off all 4 digits

 digitalWrite(CA\_1, HIGH);

 digitalWrite(CA\_2, HIGH);

 digitalWrite(CA\_3, HIGH);

 digitalWrite(CA\_4, HIGH);

}

Pomembne vrstice kode so komentirane v kodi. V tej kodi lahko pošljete poljubno številko od 0 do 9999 prek serijskega monitorja (glejte spodnjo sliko). Za preklapljanje med številkami uporabljamo prekinitve časovnika arduino. Segmente je treba vklopiti in izklopiti za vsako številko, ki je shranjena v nizu bajtov. V loop() – serijske vrednosti se preberejo in pretvorijo v int in nato v dolge podatkovne vrste. Nato se ti dolgi podatki razčlenijo na enomestno število z metodo break\_number (). Multipleksiranje izvaja funkcija razreda časovnik every() in enkrat vsako milisekundo pokliče metodo display\_number(). Ta metoda uporablja funkcijo arduino shiftOut () za pošiljanje signalov v premični register. Upoštevajte, kako je pin latch postavljen v NIZKO stanje, preden pošljemo podatke, in postavljen v VISOKO po koncu pošiljanju podatkov. Funkcija, imenovana cathode\_high(), povzroči izklop zaslona.