Arduino IR daljinski upravljač

Autor: Darko Golner, e-mail: dgolner@inet.hr

Nešto motivacije

U većini Arduino starter kitova nalaze se infrared (IR) senzor i daljinski upravljač koji su dobra kombinacija za igrarije s Arduino sketchevima koji najčešće prate takve starter kitove. Međutim, oni se mogu i korisnije upotrijebiti. Zanimalo me mogu li ih iskoristiti kako bi daljinski upravljač služio za upravljanje video playerima na PC-u. Mogu se iskoristiti, pa čemu onda stati samo na korištenju daljinskog iz Arduino starter kita kada se može upogoniti i neki drugi daljinski, pogotovo ako se ne koristi za originalni uređaj uz koji je došao. U principu, svaki se daljinski može iskoristiti za upravljanje video playerom na PC-u.

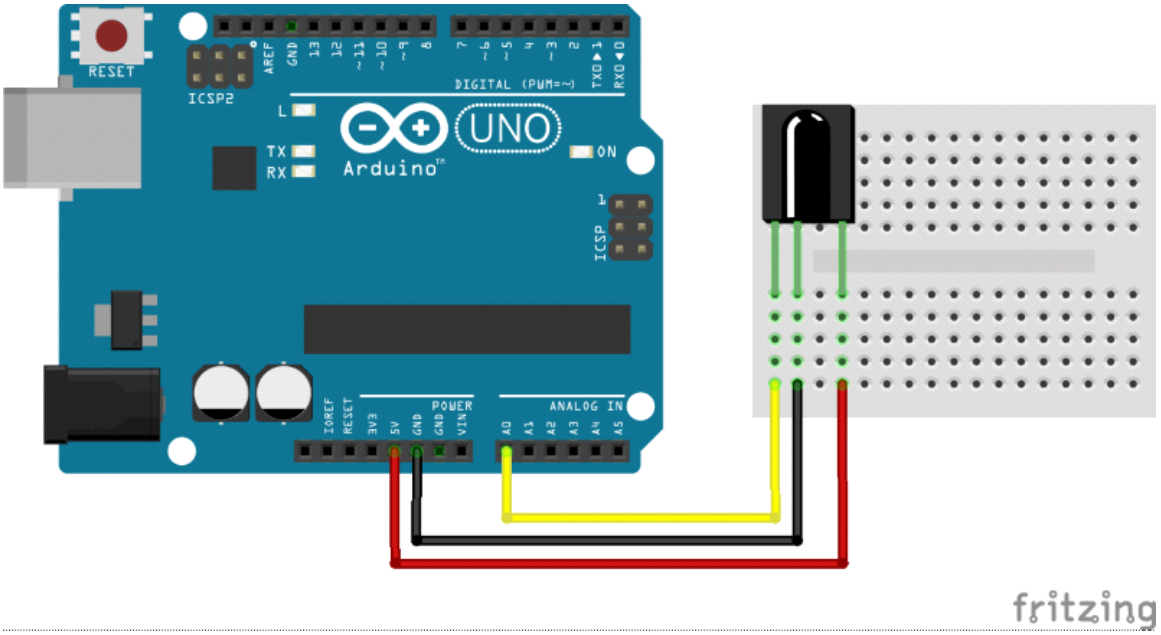
Kroz ovaj tutorial pretvorit ću kompatibilan Arduino UNO (ne koristim originalan, ali nema razlike) u USB HID (human interface device) tipkovnicu kojom će se slati naredbe, odnosno pritisnute tipke ili kombinacije tipki video playeru. Daljinskim će se poslati akcije (play, pause, stop, volume up, down i sve ostale) VLC-u, GOM-u, YouTubeu, Netflixu ili Kodiju u mojim primjerima, ali to ne sprečava korištenje bilo kojeg od playera, video, audio ili nekog potpuno drugačijeg programa. I to iz razloga što Arduino pretvaram u USB tipkovnicu. Da bi se to izvelo, bit će potrebno malo i reprogramirati Arduino, odnosno loadati prilagođeni firmware, ali naoko kompliciran posao može se jednostavno odraditi.

Komponente

Od samih korištenih komponenti nema neke specijalne nabave, sve se već nalazi u većini Arduino starter kitova:

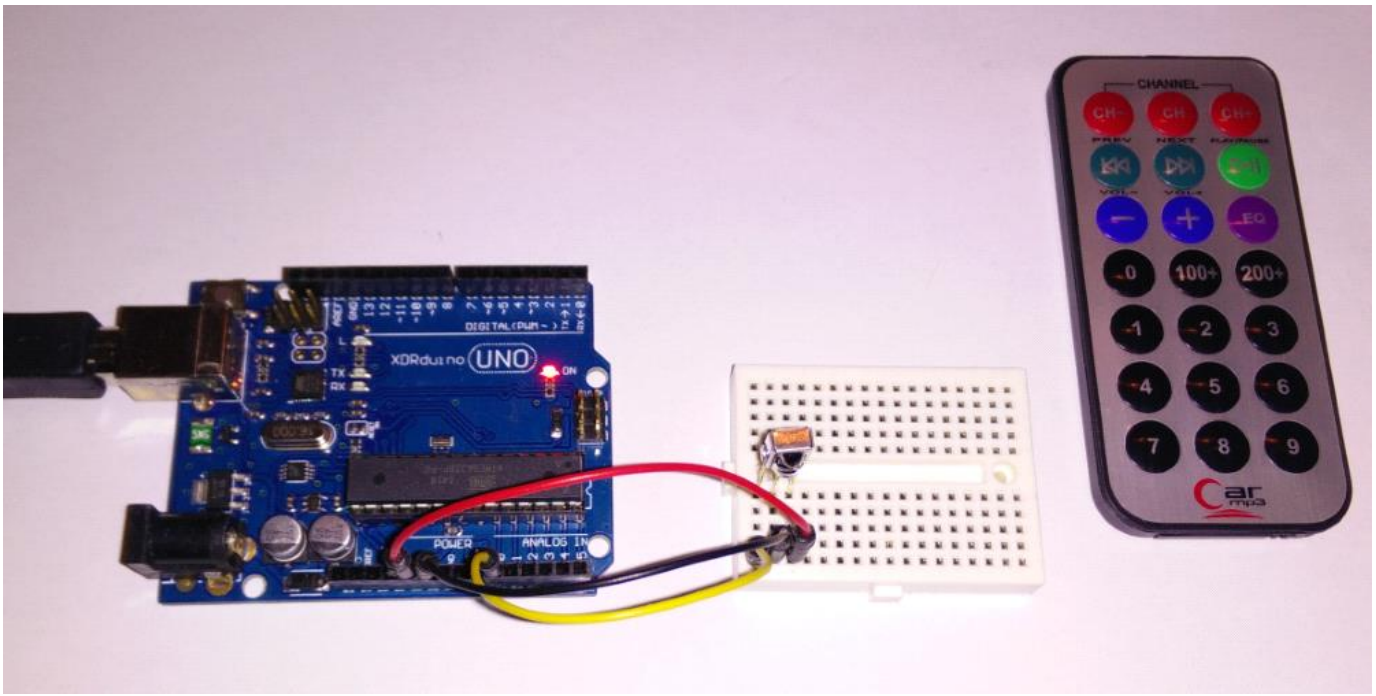
- Arduino UNO pločica (originalna ili kompatibilna)
- IR senzor (u starter kitu sam dobio senzor AX-1838HS)
- minijaturni breadboard
- tri kabla za spajanje breadboarda s Arduino UNO

Nabrojene komponente samo treba pospojiti prema slici:



Ovdje je potrebno jedino pripaziti na raspored pinova na IR senzoru. Nemaju svi isti raspored tako da je najbolje pronaći na netu datasheet korištenog senzora. U mom primjeru datasheet je na linku.

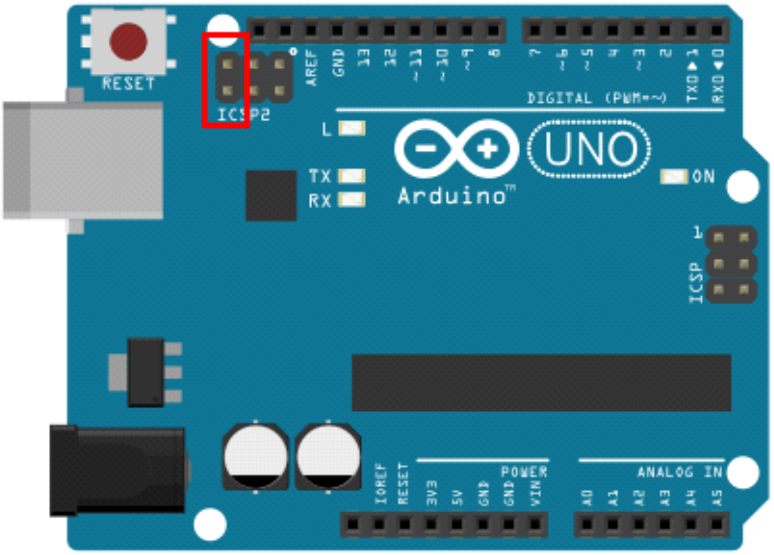
Konkretno, moje spojene komponente izgledaju ovako:



Hardverski smo gotovi i dalje preostaje programiranje za što koristim Arduino IDE.

Device Firmware Update

Prije nego se krene s programiranjem, potrebno je provjeriti može li se Arduino prebaciti u protokol putem kojeg je moguć update firmwarea. Prebacivanje na protokol DFU (Device Firmware Update) čime će biti omogućen update nije kompliciran. Dovoljno je prilikom uključenog Arduina kratko spojiti označene konektore na sekundu do dvije:



Primjer kako se to u praksi radi može se vidjeti u videu:

https://youtu.be/ZDwvmkamKnc

Nakon ove akcije, Arduino je prebačen u DFU i u Windows Device Manageru će se pojaviti novi uređaj. Ako se ne instaliraju automatski driveri, a vrlo je vjerojatno da se neće, moraju se instalirati ručno. Sami driveri dolaze u sklopu softwarea Atmel FLIP koji se mora instalirati zbog koraka koji tek slijede jer će se putem FLIP-a mijenjati firmware. Atmel FLIP može se skinuti s lokacije nakon čega ga je potrebno instalirati. Kada se instalira, mogu se ručno instalirati i driveri u Device Manageru, u mom slučaju instalirao sam Arduino UNO s ATmega16U2 chipom:

https://youtu.be/Ttz-R9KjJxI

Slično je i ukoliko se radi na Linuxu, a razlozi za sve akcije i dodatne upute mogu se naći na linku.

Ukoliko su gornje akcije bile uspješne, Arduino se može odspojiti s PC-a (izlazak iz DFU-a) jer još nismo krenuli sa zamjenom firmwarea, a pločica nam je dalje potrebna za testiranje sketcha.

Prvi testni program

Krenimo i s programiranjem u Arduino IDE-u.

Za početak, potreban je sketch koji će se uploadati na Arduino. Ovo je jednostavan sketch kojim će se provjeriti rad IR senzora i daljinskog upravljača.

```
/************************************************
*                                               *
*  Arduino IR daljinski upravljač               *
*                                               *
*  Autor: Darko Golner, 2016.                   *
*                                               *
************************************************/

// IRremote library
#include <IRremote.h>

// za indikator rada koristi se interni LED
int ledPin = 13;

// signal daljinskog očitava se na pinu A0
int Recv_PIN = A0;
```

```
IRrecv irrecv(Recv_PIN);
decode_results results;

// setup programa
void setup() {
  // definiranje korištenja interne LED-ice
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  // početak serijske komunikacije
  Serial.begin(9600);

  // pokretanje IR receivera
  irrecv.enableIRIn();
}

// glavna procedura programa
void loop() {
  // procesirati će se rezultati očitanja IR senzora
  if (irrecv.decode(&results)) {
    // ispis koda tipke na Serial Monitor
    Serial.println(results.value, HEX);

    // signaliziranje da je zaprimljen kod iz IR senzora
    digitalWrite(ledPin, HIGH);
    delay(100);
    digitalWrite(ledPin, LOW);

    // prihvaćanje sljedeće vrijednosti
    irrecv.resume();
  }

  delay(500);
}
```

U sketchu koristim library IRremote koji ne dolazi s Arduino IDE-om tako da je library potrebno skinuti s lokacije. Potrebno je skinuti cijeli library Arduino-IRremote-master.zip.
Nakon što se instalira library u IDE, može uzrokovati greške prilikom verifikacije i uploada ovog sketcha u Arduino IDE-u. IDE javlja grešku prilikom tih akcija jer je u konfliktu s instaliranim RobotIRremote libraryem. Greška s konfliktom možda će biti riješena s novim verzijama Arduino IDE-a (ovdje koristim verziju 1.6.3). Da bi do daljnjega izbjegli greške, library RobotIRremote mora se privremeno premjestiti, postoji i diskusija na tu temu.
Sada bi trebali biti uspješni Verify i Upload. Nakon uploada, može se odmah pokrenuti i Serial Monitor radi praćenja testiranja rada.

Pritiskom na tipku daljinskog upravljača, u Serial Monitoru morao bi biti vidljiv heksadecimalni kod tipke, a interni LED Arduina morao bi zatreperiti. Ako se to ne dogodi, potrebno je provjeriti konekciju pinova IR senzora i da li je output pin senzora spojen na pin A0 kao što je definirano u sketchu. U ovom primjeru dobro je provjeriti u Serial Monitoru i kako se drugi daljinski upravljači ponašaju jer svaki šalje svoje kodove tipki. Ovu osobinu iskoristit ću u kreiranju sketcha kojim će biti podržano više različitih daljinskih, a Arduino će moći prepoznati s kojim radi. Pritiskom na tipku, prebacit će se na mapiranje kodova tipki pripadnog daljinskog. Zbog toga je najbolje za svaku pritisnutu tipku daljinskog odmah zabilježiti koji je njen pripadni heksadecimalni kod. To ću iskoristiti u sljedećem sketchu i to će biti proof of concept prije nego krenem s konačnim sketchom.

Drugi testni program koji nešto i radi

Sljedeći sketch će za pritisnutu tipku Volume Up na daljinskom poslati pripadni kod tipke Up (strelica ↑) na USB HID tipkovnici. Isto će vrijediti za tipku Volume Down na daljinskom koja će poslati tipku Down (strelica ↓) na USB tipkovnici. Za tipku Play / Pause daljinskog poslat ću kod tipke Space na tipkovnici. One će već sada poslužiti za pravi test radi li daljinski s video playerom jer dosta njih tipku Space mapira s akcijama Play i Pause, a tipke Up i Down služe za reguliranje glasnoće.

Prvi je korak popisati heksadecimalne kodove tipki Volume Up, Volume Down i Play / Pause na daljinskom što znači pritisnite svaku od tipki daljinskog i kopirajte vrijednosti koje se vide u Serial Monitoru. Također, kopirajte i vrijednost koda koji se pojavi kada dugo držite pritisnutu tipku.
Drugi je korak pronaći pripadni kod tipke na tipkovnici. Ovdje se ne radi o ASCII kodu slova na tipkovnici nego o kodu tipke USB tipkovnice. Kodovi, odnosno brojevi tipki USB tipkovnice mogu se preuzeti ovdje. Za potrebe sketcha, broj iz dokumenta morat će se pretvoriti u heksadecimalan broj. Više detalja i objašnjenja o USB HID kodovima može se pronaći u originalnom dokumentu HID Usage Tables sa službenih stranica Universal Serial Bus (USB).
Konačno, kodovi daljinskog i tipkovnice moraju se mapirati u samom Arduino sketchu. Na kraju se kod USB HID-a šalje u sklopu buffera serijskom komunikacijom na USB prema PC-u. Sam sketch bi onda izgledao ovako:

```
/*************************************************
*                                                *
*  Arduino IR daljinski upravljač                *
*                                                *
*  Autor: Darko Golner, 2016.                    *
*                                                *
*************************************************/

// IRremote library
#include <IRremote.h>

// konstante kodova tipki USB tipkovnice
#define KEY_SPACE 0x2C // prema tablici je broj 44
#define KEY_UP 0x52 // prema tablici je broj 82
#define KEY_DOWN 0x51 // prema tablici je broj 81

// za indikator rada koristi se interni LED
int ledPin = 13;

// signal daljinskog očitava se na pinu A0
int Recv_PIN = A0;
IRrecv irrecv(Recv_PIN);
decode_results results;
// kod držanja pritisnute tipke na daljinskom
long keyPressed = 0x00000000;

// setup programa
void setup() {
  // definiranje korištenja interne LED-ice
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  // početak serijske komunikacije
  Serial.begin(9600);

  // pokretanje IR receivera
  irrecv.enableIRIn();
}

// glavna procedura programa
void loop() {
  // procesirati će se rezultati očitanja IR senzora
  if (irrecv.decode(&results)) {
    // buffer za slanje koda tipke
    uint8_t buf[8] = {0};

    // ako se ne radi o dugom pritisku tipke, uzima se očitan kod tipke
    // 0xFFFFFFFF je kod koji sam očitao prilikom dugog pritiska tipke
    if (results.value != 0xFFFFFFFF) {
      keyPressed = results.value;
    }

    // pritisnuta tipka Volume Up na daljinskom
```

```
    // 0xA05F20DF je kod koji sam očitao za Volume Up na daljinskom
    if (keyPressed == 0xA05F20DF) {
      // punjenje buffera s USB HID kodom tipkovnice
      buf[0] = 0;
      buf[2] = KEY_UP;
      // slanje buffera za pritisnute tipke (press key)
      Serial.write(buf, 8);
      buf[0] = 0;
      buf[2] = 0;
      // slanje buffera za otpuštanje tipke (release key)
      Serial.write(buf, 8);
    }

    // pritisnuta tipka Volume Down na daljinskom
    // 0xA05FA05F je kod koji sam očitao za Volume Down na daljinskom
    if (keyPressed == 0xA05FA05F) {
      // punjenje buffera s USB HID kodom tipkovnice
      buf[0] = 0;
      buf[2] = KEY_DOWN;
      // slanje buffera za pritisnute tipke (press key)
      Serial.write(buf, 8);
      buf[0] = 0;
      buf[2] = 0;
      // slanje buffera za otpuštanje tipke (release key)
      Serial.write(buf, 8);
    }

    // pritisnuta tipka Play / Pause na daljinskom
    // 0xA05F6897 je kod koji sam očitao za Play / Pause na daljinskom
    if (keyPressed == 0xA05F6897) {
      // punjenje buffera s USB HID kodom tipkovnice
      buf[0] = 0;
      buf[2] = KEY_SPACE;
      // slanje buffera za pritisnute tipke (press key)
      Serial.write(buf, 8);
      buf[0] = 0;
      buf[2] = 0;
      // slanje buffera za otpuštanje tipke (release key)
      Serial.write(buf, 8);
    }

    // signaliziranje da je zaprimljen kod iz IR senzora
    digitalWrite(ledPin, HIGH);
    delay(100);
    digitalWrite(ledPin, LOW);

    // prihvaćanje sljedeće vrijednosti
    irrecv.resume();
  }

  delay(500);
}
```

Potrebno je obratiti pozornost da se u sketchu ne koristi Serial Monitor jer u suprotnom šalje krive kodove kao USB HID tipkovnica (šalju se kodovi ispisa). Sketch se može probati uploadati na Arduino, ali neće kao takav odmah raditi. Razlog je što još nismo pretvorili Arduino u USB HID tipkovnicu uploadom firmwarea što je predmet sljedećih koraka.

Update Arduino firmwarea

Slijedi pretvaranje Arduina u USB HID tipkovnicu koja će biti pogonjena novim sketchom.

Za update firmwarea Arduina koristit ću ranije opisan Atmel FLIP. Arduino IDE koristit ću za upload sketcha u

međukoraku. Jedino nedostaje novi firmware, konkretno koristit ću dva. Na linku nalaze se potrebni firmwarei, potrebno je skinuti oba koji se u trenutku pisanja zovu Arduino-keyboard-0.3.hex i Arduino-usbserial-uno.hex.
Dalje se treba pridržavati sljedećih koraka kako bi se updateao prvi firmware, uploadao sketch i updateao drugi firmware:

1. kratkospojnikom prebaciti Arduino pločicu u DFU
2. pomoću Atmel FLIP-a uploadati firmware Arduino-usbserial-uno.hex
3. odspojiti Arduino pločicu, ponovno je spojiti i s Arduino IDE-om uploadati sketch
4. ponovno kratkospojnikom prebaciti Arduino pločicu u DFU
5. pomoću Atmel FLIP-a uploadati drugi firmware Arduino-keyboard-0.3.hex
6. odspojiti Arduino pločicu, ponovno je spojiti te smo time dobili Arduino USB HID tipkovnicu

Gornji koraci moraju se ponavljati svaki put kada se želi uploadati novi sketch kojim će Arduinom postati USB HID tipkovnica.

Upload firmwarea kroz Atmel FLIP može se vidjeti u videu:

https://youtu.be/DDnGKmqTCsQ

Sada je Arduino spreman da se daljinski upravljač koristi u nekom od video playera jer se Arduino ponaša kao USB tipkovnica.

Konkretan program IR daljinskog

Kao konačan sketch tutoriala dajem kompleksni program koji u sebi uključuje i nešto objektnog programiranja. Ideja je bila podržati više video playera, ali i daljinskih upravljača. Tu ću si pomoći koristeći klase (više o njima na linku). Popisao sam sve kodove tipki USB tipkovnice koje se koriste u playerima VLC, GOM i Kodi te video servisima YouTube i Netflix. Akcije u playerima okidaju se jednom ili kombinacijom tipki tako da unutar klase playera svaka metoda predstavlja jedno mapiranje akcije playera i tipke (ili njih dvije). To znači da sam za svaku metodu (player) popisao sve akcije i kombinacije tipki koje se odnose na VLC, GOM, YouTube, Netflix i Kodi. Klasa također sadržava i jedinstvenu metodu koja će se pozivati radi slanja kodova tipke (ili više njih) na USB kako bi simulirali tipkovnicu. Na sličan način je nastala i klasa daljinskog gdje sam mapirao korištene tipke daljinskog upravljača i njihove kodove. Svaki daljinski koji sam želio koristiti zbog toga je dobio vlastitu metodu. Kad se zbog svega gore opisanog obje klase instanciraju i razradi logika u glavnom programu, dobije se unificiranost prema kojoj je u principu nebitan player i daljinski jer osim što se postave na inicijalizaciji sketcha, tipkama na daljinskom ne samo da se može mijenjati odabrani player, nego se mogu mijenjati i daljinski upravljači. Konkretno, cijeli program je u priloženom:

```
/*************************************************
*                                                *
*  Arduino IR daljinski upravljač                *
*                                                *
*  Autor: Darko Golner, 2016.                    *
*                                                *
*************************************************/

// IRremote library
#include <IRremote.h>

// konstante kodova tipki USB tipkovnice
#define KEY_LEFT_CTRL 0x01
#define KEY_LEFT_SHIFT 0x02
#define KEY_LEFT_ALT 0x04
#define KEY_RIGHT_ALT 0x40
#define KEY_SPACE 0x2C
#define KEY_UP 0x52
#define KEY_DOWN 0x51
#define KEY_LEFT 0x50
#define KEY_RIGHT 0x4F
#define KEY_PLUS 0x57
#define KEY_MINUS 0x56
#define KEY_PAGEUP 0x4B
```

```
#define KEY_PAGEDOWN 0x4E
#define KEY_F 0x09
#define KEY_M 0x10
#define KEY_N 0x11
#define KEY_P 0x13
#define KEY_Q 0x14
#define KEY_S 0x16
#define KEY_X 0x1B
#define KEY_ENTER 0x28
#define KEY_BACKSLASH 0x31
#define KEY_F4 0x3D
#define KEY_F8 0x41

// true samo za debugiranje kroz Serial Monitor
bool debug = false;
// za indikator rada koristi se interni LED
int ledPin = 13;

// način upravljanja video playerom definira se klasom
class PlayerControls {
  public:
    // ime korištenog playera - nema praktičnu primjenu u radu
    String playerName;
    // univerzalne kontrole playera
    // po potrebi mogu se dodati nove kontrole
    uint8_t playStartPause[2] = {0x00, 0x00};
    uint8_t playStop[2] = {0x00, 0x00};
    uint8_t playForward[2] = {0x00, 0x00};
    uint8_t playRewind[2] = {0x00, 0x00};
    uint8_t playPrevious[2] = {0x00, 0x00};
    uint8_t playNext[2] = {0x00, 0x00};
    uint8_t volumeUp[2] = {0x00, 0x00};
    uint8_t volumeDown[2] = {0x00, 0x00};
    uint8_t volumeMute[2] = {0x00, 0x00};
    uint8_t fullScreen[2] = {0x00, 0x00};
    uint8_t exitPlayer[2] = {0x00, 0x00};
    // buffer za slanje koda tipke
    uint8_t buf[8] = {0};

    // pretvaranje kratice u naredbu za HID
    void sendKey(uint8_t chr[]) {
      // ispis u debug modu na Serial Monitor
      if (debug) {
        Serial.print(chr[0], HEX);
        Serial.println(chr[1], HEX);
      }
      // prazan kod
      if (chr[0] == 0x00 && chr[1] == 0x00) {
        buf[0] = 0;
        buf[2] = 0;
      }
      // poslan prvi kod
      if (chr[0] != 0x00 && chr[1] == 0x00) {
        buf[2] = chr[0];
      }
      // poslana oba koda
      if (chr[0] != 0x00 && chr[1] != 0x00) {
        // za oba ista koda šalje se samo jedan
        if (chr[0] == chr[1]) {
          buf[0] = 0;
        } else {
          buf[0] = chr[0];
        }
        buf[2] = chr[1];
```

```
  }
  // ispis u debug modu na Serial Monitor
  if (debug) {
    Serial.print(buf[0], HEX);
    Serial.println(buf[2], HEX);
  }
  // slanje buffera za pritisnute tipke (press key)
  Serial.write(buf, 8);
  buf[0] = 0;
  buf[2] = 0;
  // slanje buffera za otpuštanje tipke (release key)
  Serial.write(buf, 8);
  delay(100);
}

// tipkovničke kratice za VLC media player
void VLC() {
  playerName = "VLC";
  // Play / Pause: Space
  playStartPause[0] = KEY_SPACE;
  // Stop: s
  playStop[0] = KEY_S;
  // Very short forward jump: Shift + Right
  playForward[0] = KEY_LEFT_SHIFT;  playForward[1] = KEY_RIGHT;
  // Very short backwards jump: Shift + Left
  playRewind[0] = KEY_LEFT_SHIFT;  playRewind[1] = KEY_LEFT;
  // Previous: p
  playPrevious[0] = KEY_P;
  // Next: n
  playNext[0] = KEY_N;
  // Volume up: Ctrl + Up
  volumeUp[0] = KEY_LEFT_CTRL;  volumeUp[1] = KEY_UP;
  // Volume down: Ctrl + Down
  volumeDown[0] = KEY_LEFT_CTRL;  volumeDown[1] = KEY_DOWN;
  // Mute: m
  volumeMute[0] = KEY_M;
  // Fullscreen: f
  fullScreen[0] = KEY_F;
  // Quit: Ctrl + q
  exitPlayer[0] = KEY_LEFT_CTRL;  exitPlayer[1] = KEY_Q;
}

// tipkovničke kratice za GOM Player
void GOM() {
  playerName = "GOM";
  // Play / Pause: Space
  playStartPause[0] = KEY_SPACE;
  // Stop: Ctrl + Space
  playStop[0] = KEY_LEFT_CTRL;  playStop[1] = KEY_SPACE;
  // Fast Forward (10 sec): Right Arrow
  playForward[0] = KEY_RIGHT;
  // Rewind (10 sec): Left Arrow
  playRewind[0] = KEY_LEFT;
  // Play Previous File (Chapter): Page Up
  playPrevious[0] = KEY_PAGEUP;
  // Play Next File (Chapter): Page Down
  playNext[0] = KEY_PAGEDOWN;
  // Player Volume Up: Up Arrow
  volumeUp[0] = KEY_UP;
  // Player Volume Down: Down Arrow
  volumeDown[0] = KEY_DOWN;
  // Mute: m
  volumeMute[0] = KEY_M;
  // Full Screen: Alt + Enter
```

```
  fullScreen[0] = KEY_LEFT_ALT;  fullScreen[1] = KEY_ENTER;
  // Quit GOM Player: Alt + F4
  exitPlayer[0] = KEY_LEFT_ALT;  exitPlayer[1] = KEY_F4;
}

// tipkovničke kratice za YouTube Player
void YouTube() {
  playerName = "YouTube";
  // Play / Pause: Space
  playStartPause[0] = KEY_SPACE;
  // Very short forward jump: Right Arrow
  playForward[0] = KEY_RIGHT;
  // Very short backwards jump: Left Arrow
  playRewind[0] = KEY_LEFT;
  // Play Previous Video: Shift + p
  playPrevious[0] = KEY_LEFT_SHIFT;  playPrevious[1] = KEY_P;
  // Play Next Video: Shift + n
  playNext[0] = KEY_LEFT_SHIFT;  playNext[1] = KEY_N;
  // Volume up: Up Arrow
  volumeUp[0] = KEY_UP;
  // Volume down: Down Arrow
  volumeDown[0] = KEY_DOWN;
  // Mute: m
  volumeMute[0] = KEY_M;
  // Fullscreen: f
  fullScreen[0] = KEY_F;
}

// tipkovničke kratice za Netflix Movie Player
void Netflix() {
  // pošto je većina kratica ista kao i za YouTube, koristit će se njegove
  YouTube();
  playerName = "Netflix";
  // nekorištene kratice se inicijaliziraju
  playPrevious[0] = 0x00;
  playPrevious[1] = 0x00;
  playNext[0] = 0x00;
  playNext[1] = 0x00;
}

// tipkovničke kratice za Kodi
void Kodi() {
  playerName = "Kodi";
  // Play/Pause: Space
  playStartPause[0] = KEY_SPACE;
  // Stop: x
  playStop[0] = KEY_X;
  // Seek step forward: Right Arrow
  playForward[0] = KEY_RIGHT;
  // Seek step backward: Left Arrow
  playRewind[0] = KEY_LEFT;
  // Previous: Page Down
  playPrevious[0] = KEY_PAGEDOWN;
  // Next: Page Up
  playNext[0] = KEY_PAGEUP;
  // Volume up: +
  volumeUp[0] = KEY_PLUS;
  // Volume down: -
  volumeDown[0] = KEY_MINUS;
  // Mute: F8
  volumeMute[0] = KEY_F8;
  // Fullscreen: Right Alt + q
  fullScreen[0] = KEY_RIGHT_ALT;  fullScreen[1] = KEY_Q;
  // Quit: Alt + F4
```

```
      exitPlayer[0] = KEY_LEFT_ALT;  exitPlayer[1] = KEY_F4;
    }

};

// rad s daljinskim upravljačem definira se klasom
class RemoteControls {
  public:
    // naziv daljinskog upravljača - nema praktičnu primjenu u radu
    String playerName;
    // gumbi kontrola na daljinskom upravljaču
    // po potrebi mogu se dodati nove kontrole
    long onOff;
    long longPress;
    long playStartPause;
    long playStop;
    long playForward;
    long playRewind;
    long playPrevious;
    long playNext;
    long volumeUp;
    long volumeDown;
    long volumeMute;
    long fullScreen;
    long button0;
    long button1;
    long button2;
    long button3;
    long button4;
    long button5;
    long button6;
    long button7;
    long button8;
    long button9;
    // buffer za slanje koda tipke
    uint8_t buf[8] = {0};

    // blink interne LED-ice
    void blinkLED() {
      digitalWrite(ledPin, HIGH);
      delay(100);
      digitalWrite(ledPin, LOW);
    }

    // blink Caps Lock na tipkovnici - koristi se kao oznaka promjene daljinskog
    void blinkCapsLock() {
      buf[2] = 0x39;
      Serial.write(buf, 8);
      buf[0] = 0;
      buf[2] = 0;
      Serial.write(buf, 8);
      delay(500);
      buf[2] = 0x39;
      Serial.write(buf, 8);
      buf[0] = 0;
      buf[2] = 0;
      Serial.write(buf, 8);
      delay(100);
    }

    // mapiranje kodova za Arduino starter kit daljinski
    long switchToStarterKitRemote = 0xFF629D; // gumb CH
    void StarterKitRemote() {
      playerName = "StarterKitRemote";
```

```
      onOff = 0x00; // nije mapirano
      longPress = 0xFFFFFFFF;
      playStartPause = 0xFFC23D;
      playStop = 0x00; // nije mapirano
      playForward = 0xFF02FD;
      playRewind = 0xFF22DD;
      playPrevious = 0xFFA25D;
      playNext = 0xFFE21D;
      volumeUp = 0xFFA857;
      volumeDown = 0xFFE01F;
      volumeMute = 0x00; // nije mapirano
      fullScreen = 0x00; // nije mapirano
      button0 = 0xFF6897;
      button1 = 0xFF30CF;
      button2 = 0xFF18E7;
      button3 = 0xFF7A85;
      button4 = 0xFF10EF;
      button5 = 0xFF38C7;
      button6 = 0xFF5AA5;
      button7 = 0xFF42BD;
      button8 = 0xFF4AB5;
      button9 = 0xFF52AD;
    }

    // mapiranje kodova za AverMedia RM-KS daljinski
    long switchToAverMedia = 0xA05F10EF; // gumb Source
    void AverMedia() {
      playerName = "AverMedia";
      onOff = 0xA05F807F;
      longPress = 0xFFFFFFFF;
      playStartPause = 0xA05F6897;
      playStop = 0xA05F18E7;
      playForward = 0xA05FB847;
      playRewind = 0xA05F38C7;
      playPrevious = 0xA05FC03F;
      playNext = 0xA05F40BF;
      volumeUp = 0xA05F20DF;
      volumeDown = 0xA05FA05F;
      volumeMute = 0xA05F609F;
      fullScreen = 0xA05F6A95;
      button0 = 0xA05F48B7;
      button1 = 0xA05F906F;
      button2 = 0xA05F50AF;
      button3 = 0xA05FD02F;
      button4 = 0xA05F30CF;
      button5 = 0xA05FB04F;
      button6 = 0xA05F708F;
      button7 = 0xA05FF00F;
      button8 = 0xA05F08F7;
      button9 = 0xA05F8877;
    }

};

// signal daljinskog očitava se na pinu A0
int Recv_PIN = A0;
IRrecv irrecv(Recv_PIN);
decode_results results;
// kod držanja pritisnute tipke na daljinskom
long keyPressed = 0x00000000;

// deklaracija varijable playera
PlayerControls *player;
// deklaracija varijable daljinskog
```

```
RemoteControls *remote;

// setup programa
void setup() {
  // definiranje korištenja interne LED-ice
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  // početak serijske komunikacije
  Serial.begin(9600);

  // pokretanje IR receivera
  irrecv.enableIRIn();

  // instanciranje video playera
  player = new PlayerControls();
  // na inicijalizaciji Arduina koristim video player VLC
  player->VLC();

  // instanciranje daljinskog upravljača
  remote = new RemoteControls();
  // na inicijalizaciji Arduina koristim daljinski AverMedia
  remote->AverMedia();
}

// glavna procedura programa
void loop() {
  // procesirati će se rezultati očitanja IR senzora
  if (irrecv.decode(&results)) {
    // ispis u debug modu na Serial Monitor
    if (debug) {
      Serial.println(results.value, HEX);
    }
    // signaliziranje da je zaprimljen kod iz IR senzora
    remote->blinkLED();

    // ako se ne radi o dugom pritisku tipke na daljinskom, uzima se očitan kod
    if (results.value != remote->longPress) {
      keyPressed = results.value;
    }

    // svaka tipka na daljinskom ima mapiranje s akcijom video playera
    // pritisnuta tipka On / Off na daljinskom šalje video playeru akciju izlaza
    if (keyPressed == remote->onOff) {
      player->sendKey(player->exitPlayer);
    }
    // pritisnuta tipka na daljinskom šalje pripadnu akciju video playeru
    if (keyPressed == remote->playStartPause) {
      player->sendKey(player->playStartPause);
    }
    if (keyPressed == remote->playStop) {
      player->sendKey(player->playStop);
    }
    if (keyPressed == remote->playForward) {
      player->sendKey(player->playForward);
    }
    if (keyPressed == remote->playRewind) {
      player->sendKey(player->playRewind);
    }
    if (keyPressed == remote->playPrevious) {
      player->sendKey(player->playPrevious);
    }
    if (keyPressed == remote->playNext) {
      player->sendKey(player->playNext);
```

```
}
if (keyPressed == remote->volumeUp) {
  player->sendKey(player->volumeUp);
}
if (keyPressed == remote->volumeDown) {
  player->sendKey(player->volumeDown);
}
if (keyPressed == remote->volumeMute) {
  player->sendKey(player->volumeMute);
}
if (keyPressed == remote->fullScreen) {
  player->sendKey(player->fullScreen);
}

// proizvoljno su definirane akcije prema tipkama daljinskog
// tipka 0 na daljinskom instancira video player - reset odabranog playera
if (keyPressed == remote->button0) {
  player = new PlayerControls();
  remote->blinkCapsLock();
}
// tipka 1 na daljinskom postavlja mapiranje akcija koje se šalju VLC-u
if (keyPressed == remote->button1) {
  player = new PlayerControls();
  player->VLC();
  remote->blinkCapsLock();
}
// tipka 2 na daljinskom postavlja mapiranje akcija koje se šalju GOM-u
if (keyPressed == remote->button2) {
  player = new PlayerControls();
  player->GOM();
  remote->blinkCapsLock();
}
// tipka 3 na daljinskom postavlja mapiranje akcija koje se šalju YouTubeu
if (keyPressed == remote->button3) {
  player = new PlayerControls();
  player->YouTube();
  remote->blinkCapsLock();
}
// tipka 4 na daljinskom postavlja mapiranje akcija koje se šalju Netflixu
if (keyPressed == remote->button4) {
  player = new PlayerControls();
  player->Netflix();
  remote->blinkCapsLock();
}
// tipka 5 na daljinskom postavlja mapiranje akcija koje se šalju Kodiu
if (keyPressed == remote->button5) {
  player = new PlayerControls();
  player->Kodi();
  remote->blinkCapsLock();
}

// proizvoljno je omogućeno da se mogu mijenjati daljinski prema tipki
// tipkom na daljinskom iz Arduino starter kita mijenjaju se pripadni kodovi
if (keyPressed == remote->switchToStarterKitRemote) {
  remote = new RemoteControls();
  remote->StarterKitRemote();
  remote->blinkCapsLock();
}
// tipkom na daljinskom AverMedia TV kartice mijenjaju se pripadni kodovi
if (keyPressed == remote->switchToAverMedia) {
  remote = new RemoteControls();
  remote->AverMedia();
  remote->blinkCapsLock();
}
```

```
    // prihvaćanje sljedeće vrijednosti
    irrecv.resume();
  }

  delay(500);
}
```

Nakon novog uploada sketcha prema gornjim koracima updatea firmwarea (ponavljanje gornjih koraka 1. - 6.) imamo mnogo moćniju Arduino USB HID tipkovnicu koja će služiti kao daljinski za upravljanje video playerima.

Download materijali

PDF tutorial
USB_Keyboard_Keys.pdf
ir_controler_tutorial.ino
ir_controler_tutorial_poc.ino
ir_controler.ino

Dodatni linkovi

IR-RemoteControl
https://arduino-info.wikispaces.com/IR-RemoteControl

Updating the Atmega8U2 and 16U2 on an Uno or Mega2560 using DFU
https://www.arduino.cc/en/Hacking/DFUProgramming8U2

HID Information
http://www.usb.org/developers/hidpage/

Objavljeno 2016. g.