Recent Innovations in Mechatronics (RIiM) Vol. 5. (2018). No. SI.
DOI: 10.17667/riim.2018.si/25.

Hangreaktív mood lamp tervezése & vezérlése Android Linux okostelefonnal

Lós Balázs
Debreceni Egyetem,
Informatikai Kar
Debrecen, Magyarország
los.balazs94@gmail.com

Erdei Erika
Debreceni Egyetem,
Tanárképző Központ
Debrecen, Magyarország
erika2322@gmail.com

Erdei Timotei István
Mechatronikai Tanszék
Debreceni Egyetem, Műszaki Kar
Debrecen, Magyarország
timoteierdei@eng.unideb.hu

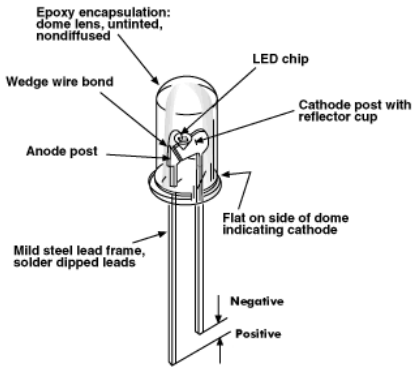
***Absztrakt*— A LED technológiát a világítástechnika mellett a különböző kisebb-nagyobb kijelzőkben is használják háttérvilágításként, továbbá a szórakoztató elektronikai cégek és perifériagyártók is meglátták benne a piaci potenciált. Ezekben az iparágakban, az utóbbi években ugrásszerűen megnőtt a termékek LED világítással való díszítése a könnyebb eladhatóság és a nagyobb profit érdekében. A projektben egy hang reaktív LED lámpa lesz bemutatva, mely a technológia, szórakoztatóipar felől betöltött szerepét hivatott bemutatni. Napjainkban már alapvető igény, hogy az ilyen eszközök mobilról vezérelhetők legyenek, így ez az elvárás kritikus tényező volt a fejlesztés során.**

Kulcsszavak—Arduino; RGB; HSV; LED; Blynk; hangreaktív

I. BEVEZETŐ

A fénykibocsátó diódák (LED-ek) évtizedek óta részei a mindennapi életnek, kezdve az 1960-as években az indikátorokkal és az infravörös távirányítókkal [3]. Az eleinte forgalomirányító lámpákban, hirdetőtáblák kijelzőiben és elektronikai paneleken állapotjelzőként használt LED technológia az utóbbi időben rendkívüli fejlődésen ment át. A technológia fejlődésével napjainkra a világítástechnikában is felfedezték pozitív tulajdonságait, mint például az energiatakarékosság vagy a környezetbarátság. A világítástechnika mellett a szórakoztató elektronikai cégek is meglátták a benne rejlő lehetőségeket, így az akkori kezdetleges technológia mára teljesen más területeken is megjelenik, mint azt akkor gondolták volna [1].

A projektben a LED szerepe lesz bemutatva a szórakoztató elektronika irányából megközelítve. Egy olyan Arduino alapú hangra aktiválódó lámpa kerül bemutatásra, amely Android Linux [16] okostelefonról vezérelhető. A fő funkciók, azaz a hangra aktiválódás és a világítás mellett a mobilos applikáció rendelkezik pár extra funkcióval, amelyek a témától kissé eltérnek, mégis hasznos funkciónak bizonyultak. A kor elvárásainak híven az eszköz okos telefonnal vezérelhető, vezeték nélküli kapcsolaton keresztül, így megvalósítja a hálózati kommunikáció legújabb igényeit. A kommunikáció legutóbbi paradigmája, amit népszerűen csak Tárgyak Internetének (IoT) hívnak; ahol mindennapi életünk különböző típusú tárgyai, kezdve az okostelefonokkal, érzékelőkkel vagy hálózati tárgyakkal összekapcsolt eszközökkel (például RFID); kommunikálni tudnak egymással és részei az internetnek [2]. A kutatás/fejlesztésnek a Debreceni Egyetem adott otthont [18].



1. ábra: A LED dióda felépítése [17]

II. TERVEZÉSI SZEMPONTOK

A tervezés legelején, a koncepció szintjén meghatározásra kerültek azok a tényezők, amelyeket az eszköznek tudnia kell. Ezek közé tartozott a kiterjedtebb színállítási lehetőség, a hangra reagáló működés, valamint a vezetéknélküli kapcsolat. Szempont volt még, hogy a vezérléshez használt okostelefonos alkalmazás könnyen kezelhető és letisztult legyen.

A kiterjedt színállítási lehetőség iránti igény egyből meghatározta, hogy egy olyan Arduino könyvtárra lesz szükség, amely több színmódot támogat és ennek kihasználására milyen további eszközökre lesz szükség.

A munka elkezdése előtt a lámpa kinézetét is meg kellett tervezni. Könnyen kivitelezhető, de mégis esztétikus és funkcióját betöltő eszköz tervezése volt a cél, így már az elején kialakult, hogy milyen formájú legyen az eszköz. Egy belső henger adja a lámpa vázát, a külsejét pedig egy újabb szélesebb, átlátszó henger alkotja.

DOI: 10.17667/riim.2018.si/25.

III. HARDVER ÉS ALKATRÉSZEK

A tervezési fázisban eldőlt, hogy a lámpa Arduino alapokra fog épülni. Az Arduino egy nyílt forráskódú elektronikai platform, amely könnyen használható hardveren és szoftveren alapul. Pontosan egy Arduino Uno panel a választott mikrokontroller, az egész projekt alapja. A panel a Microchip (Atmel) ATmega328P mikrovezérlőre épül, 6 darab analóg és 14 digitális ki- és bemenettel rendelkezik, amelyből 6 PWM-képes. Ezen kívül található még rajta egy 3.3V-os és egy 5V-os csatlakozó, amelyek a különböző modulok tápellátásáért felelnek, tovább 3 darab GND csatlakozó, melyek pedig a földpotenciálú pont szerepét töltik be egy áramkörben [4].

A világításért egy digitális LED-szalag felel, a választás egy WS2812B vezérlőchipes WorldSemi termékre esett. Ebben a szalagban a vezérlő áramkör és az RGB chip egy csomagba van integrálva. Minden egyes LED csomagban külön-külön van egy vezérlő lapka, így akár minden egyes LED csomagnak más színt lehet beállítani, ez különbözteti meg a digitális szalagokat az analóg társaiktól. A három alapszín egyenként 256 féle fényerőt képes megjeleníteni, így összesen több mint 16 millió különböző szín kapható. Maximális fényerőn, fehér fény mellett, egy LED működéséhez 60 mA áramerősség szükséges, 5 V feszültség mellett.

2. ábra: Az elkészült eszköz

A projektben egy 80 darab LED-ből álló szalag került felhasználásra. A szalag tápellátásáról egy 5 V-os 5 A-es Mean Well tápegység gondoskodik, amely egy próbapanelra szerelhető 2.1 mm-es jack csatlakozós tápegység aljzathoz csatlakozik.

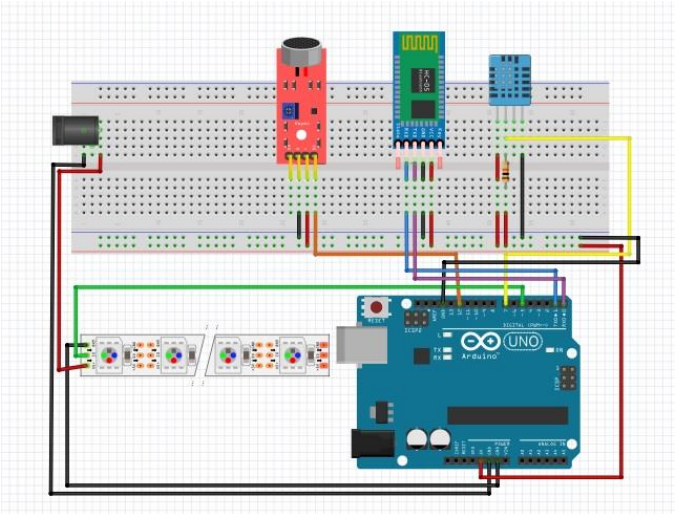
$$I_{max} = n * I_{LED} \qquad (1.)$$

Az (1.) képletet alkalmazva könnyen kiszámítható a maximális áramerősség (I_{max}) a LED csomagok számából (n) és egy LED működéséhez szükséges maximális áramerősségből (I_{LED}). A maximális áramerősség alapján megkapható, hogy milyen áramerősségű tápegységre lesz szükség.

A mikrokontroller és az okostelefon közötti kommunikáció vezetéknélküli kapcsolaton történik. A vezetéknélküli technológiák közül a Bluetooth tökéletesen elegendő a jelenlegi igényekhez, ezért a csatlakozás egy HC-05 típusú modullal történik. Az eszköz 5 V-os feszültség alatt üzemel és maximális áramerőssége 35 mA. Természetesen a Bluetooth kapcsolattal az eszköz még nem teljesíti az IoT egyik legfőbb, követelményét, hogy az eszköz része legyen az internetnek. Az okostelefonos applikáció viszont folyamatos kapcsolatot létesít egy távoli felhőszerverrel, amellyel így már az eszköz is része lesz az internetnek a később részletesebben taglalt módon.

Beszerzésre került egy digitális és analóg kimenettel rendelkező Keyes KY-038 típusú hangérzékelő, amely a környezetből érkező zajokat fogja feldolgozni. Jelen projektben csak az érzékelő digitális kimenete van használatban, a munka digitális jelekkel fog zajlani. A hangérzékelő 5 V-os feszültség mellett működik és a maximális áramerőssége 5 mA.

Extra funkciók ellátására beépítésre került egy DHT-11 hőmérséklet és páratartalom érzékelő, amely folyamatosan figyeli a hőmérsékleti és relatív páratartalom értékeket és ezeket az adatokat továbbítja a mobil applikációnak. Ez a szenzor szintén 5 V-os feszültség alatt működik 2,5 mA-es maximális áramerősség mellett. A DHT-11 szenzorból kétféle típus létezik, az egyik 4 tűs, a másik egy 3 tűvel rendelkező modell. A 3 tűvel ellátott modell egy kis nyomtatott áramkörre van szerelve. Az utóbbi, 3 tűs változat egy felületre szerelt 10 kΩ-os felhúzó ellenállást tartalmaz a jelvezetékhez. A 4 tűs változathoz egy 10 kΩ-os felhúzó ellenállás szükséges a jelvonal és az 5 V-os vonal közé, hogy a jel szintje biztosan magas legyen alapesetben is [5].



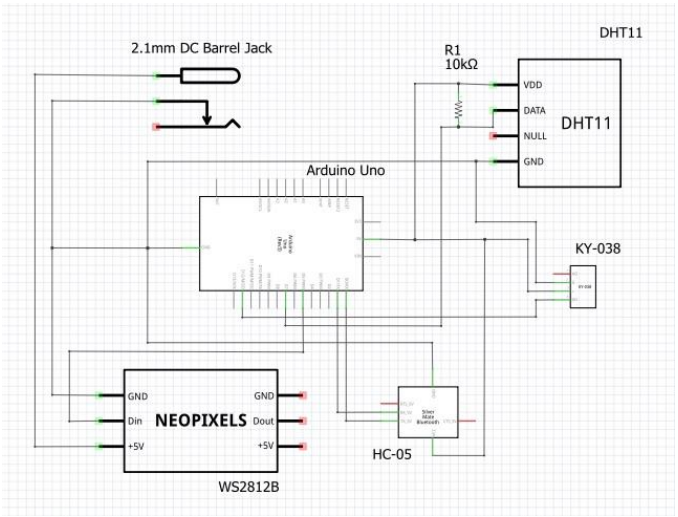
3. ábra: A fizikai kapcsolás

IV. A LÁMPA FELÉPÍTÉSE

A lámpa belső vázát egy PVC cső alkotja, ami kellően erős és stabil. Erre a csőre van körkörösen felragasztva az 1,33 méretes LED szalag. A belső henger magassága 14 cm, átmérője pedig 5 cm. A külső borítást egy átlátszó műanyag henger biztosítja, melynek a hátulján vannak kivezetve a szalaghoz használt kábelek. A külső henger magassága 17 cm és az átmérője 8 cm. A mikrokontroller és a próbapanelhez kapcsolt áramköri elemek a lámpa mögött helyezkednek el, ahogy az a 2. ábrán látható.

V. MEGTERVEZETT ÁRAMKÖR

Az áramkör a Fritzing nevű programban került megtervezésre. A programban lehetőség nyílik fizikai és sematikus kapcsolás létrehozására, nyomtatott áramkör tervezésére, továbbá Arduino programkód írására és a kód feltöltésére is. A Fritzing egy nyílt forráskódú hardverkezdeményezés, amely az elektronikát, mint kreatív tárgyat hozzáférhetővé teszi bárki számára [6]. Az áramkör fő része az Arduino Uno mikrokontroller, aminek feladata a WS2812B LED szalag, a szenzorok és a HC-05 Bluetooth modul vezérlése.

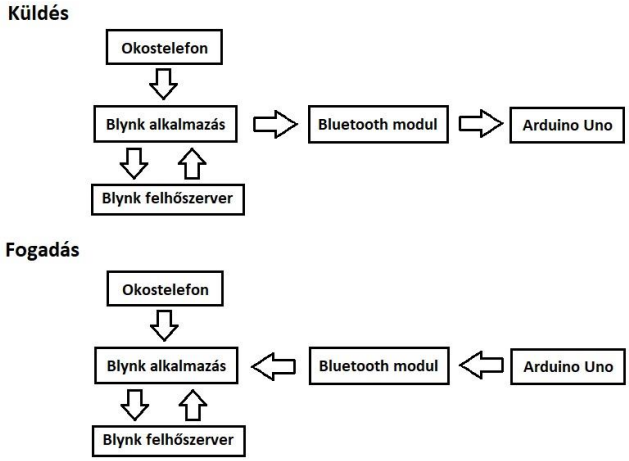


4. ábra: A sematikus kapcsolás

VI. OKOSTELEFONOS APPLIKÁCIÓ

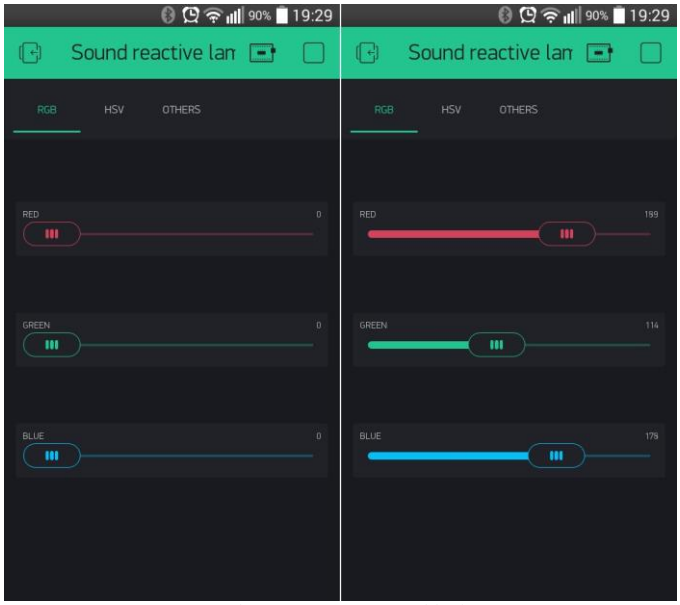
A tervezés előtt fontos követelmény volt, hogy a mobilos interfész könnyen használható és esztétikus legyen. A felhasználói felület kifejlesztése után a felhasználók egyedül maradnak a kezelőfelületen lévő problémák kezelésében [7], ezért kritikus tényező volt a könnyen kezelhető, tanulható és egyszerű kezelőfelület. Többféle Androidos applikáció létezik, melyek segítségével egyedi, könnyen programozható irányítópultok hozhatók létre mikrokontrollerek vezérlésére, Android programozói tudás nélkül. Ez rendkívül nagy segítség egy kezdő programozó számára, aki mikrokontrollerekre szeretne programot fejleszteni, mivel így jóval több időt tud fordítani az Arduino programnyelvének megértésére, gyakorlására.

Jelen projektben a Blynk nevű program van használatban. A Blynk egy olyan okostelefonos alkalmazással rendelkező platform, amellyel mikrokontrollerek és mikroszámítógépek vezérelhetők az interneten keresztül. Ez egy olyan digitális műszerfal, ahol grafikus felületet lehet létrehozni egy projekt számára, olyan módon, hogy a különböző modulok egyszerű „drag and drop” módszerrel felhelyezhetők az interfészre. A Blynk lehetővé teszi számunkra, hogy alkalmazásokat hozzunk létre, majd ezután egy internet-hozzáféréssel rendelkező PC-hez csatlakoztatott Arduino panel irányítására használjuk (például LED-ek, szervók vezérlése; adatok fogadása stb.) a világ bármely pontjáról egy okostelefonnal [8], amennyiben a kapcsolódás számítógépen keresztül történik.



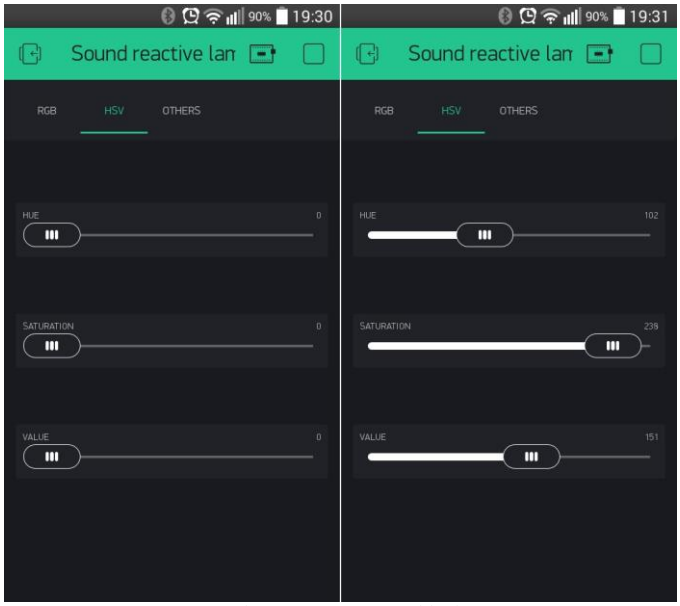
5. ábra: Adatküldés és adatfogadás

A Blynk működéséhez kötelezően szükség van egy szerverre, ahol a felületre felvitt modulok értékei tárolódnak. Ez lehet a Blynk ingyenes felhőszervere, de a lehetőség adott egy saját Blynk [8] szerver konfigurálására és annak használatára is. Jelen esetben a mobil és a mikrokontroller Bluetooth kapcsolaton fog kommunikálni, viszont a Blynk megköveteli a szerver elérését, így az alkalmazás fogja megoldani mind a Bluetooth, mind pedig a WiFi kapcsolaton való adatküldést, ahogyan az 5. ábrán látható.



6. ábra: Az RGB színmód lapja

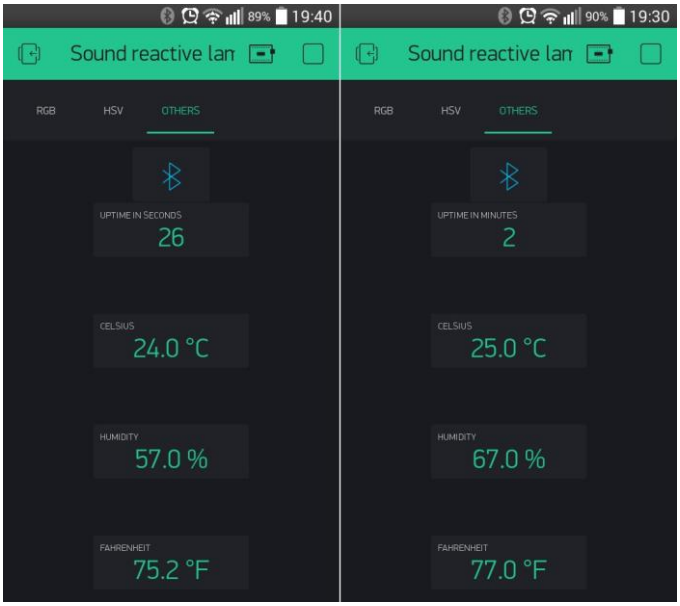
Az alkalmazáson belüli utasítások küldésénél az applikáció elküldi az úgynevezett „widgetek” értékeit a szervernek WiFi kapcsolaton keresztül, a kiszolgáló ezeket az adatokat eltárolja. Az applikáció Bluetooth kapcsolaton továbbítja az adatokat a HC-05 modulnak és végül az Arduino panel megkapja és feldolgozza a kapott adatokat és végrehajtja az utasításokat. Amennyiben a mikrokontroller küld adatokat a grafikus felületen levő moduloknak, az adat útja fordított, de ebben az esetben is el kell küldeni a „widgetek” értékeit a szervernek. Mivel csak a Bluetooth modul hatótávján belül történik az eszköz vezérlése, így felesleges dolognak tűnhet a szerver jelenléte, de ez a Blynk használata során elkerülhetetlen, viszont különböző „widget” szinkronizálásokhoz, értékeik lementéséhez és előhívásához rendkívül hasznos tud lenni.



7. ábra: A HSV színmód lapja

Az alkalmazással készített grafikus felület 3 lapból áll. 1-1 lap az RGB és HSV színmódnak, amelyeken egyaránt 3-3 csúszka található. Az RGB lapon a csúszkák a piros, zöld és kék színek értékeit állítják be és ezen értékek alapján történik a színkeverés, ahogyan az a 6. ábrán látható. A HSV lapon a csúszkák a színárnyalatot, a telítettséget és fényerőt állítják be, ez alapján történik a színek kinyerése, ahogy ez a 7. ábrán is látható. Az utóbbi mód sokkal felhasználó közelibb, több képszerkesztő program is ezzel vagy hasonló színmóddal valósítja meg a színkeverést.

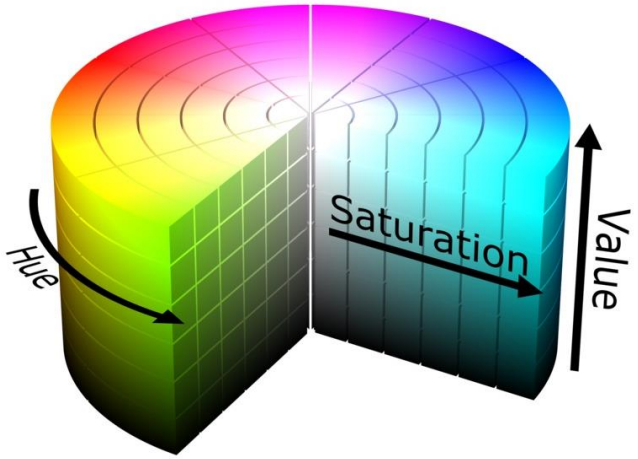
A 3. lapon található a HC-05 modulhoz való csatlakozásért felelős Bluetooth „widget”, amellyel több Bluetooth eszköz esetén kiválasztható melyikhez történjen a csatlakozás. További extra funkciók találhatók még ezen a lapon, ilyen például a hőmérsékleti értékek kijelzésére alkalmas 2 darab címkézett érték kijelző, amelyek Fahrenheitben és Celsiusban mutatják a folyamatosan monitorozott értékeket. A projektben használt DHT-11 modul a relatív páratartalmat is képes meghatározni, így egy újabb címkézett érték kijelzőn ez az érték látható. A legfelső kijelző „widgetről” az olvasható le, hogy az Arduino eszköz és a szerver között a kapcsolat mennyi ideje valósult meg.



8. ábra: A Bluetooth „widget” és az extra funkciók

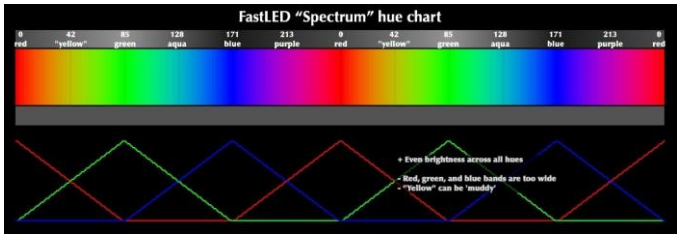
VII. HSV SZÍNTÉR

A színterek vagy másnéven színmodellek a színek matematikai ábrázolásának módjai. A leggyakoribb színmodellek az RGB (piros, zöld, kék), a CMYK (cián, magenta, sárga, fekete), az YCbCr (fénysűrűség komponens, kroma kék különbség, kroma piros különbség), a HSB (színárnyalat, telítettség, fényerő) és a HSI (színárnyalat, telítettség, intenzitás), amelyek az RGB színtérből vannak származtatva és olyan módon ábrázolják a színeket, ahogyan az emberi szem érzékeli és értelmezi a színeket [9].



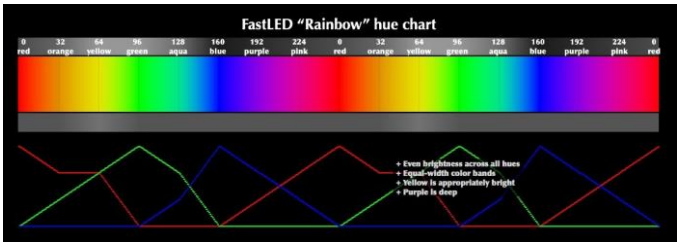
9. ábra: A HSV színmodell grafikai ábrázolása [10]

A színeket általában három réteg határozza meg, ezek a rétegek írják le a szín helyzetét a használt színmodellben [11]. A HSV színtér megegyezik a HSB-vel, szakirodalmanként eltérő melyiket használják. Létezik még az úgynevezett HSL színtér is (színárnyalat, telítettség, fényesség), de ez nem összekeverendő a HSB/HSV színmodellel, annak ellenére sem, hogy a két modell által adott színskála között kicsi az eltérés. A HSB/HSV színmodell tehát egy olyan rendszer, amely az emberi szem színfelismerésén alapul. 3 alapvető tulajdonsága a szín sajátos átmenete, a színárnyalat élénksége és szín fényerejének mértéke, ahogyan az a 9. ábrán látható. A színárnyalatot fokban szokták megadni 0-360-ig, a telítettséget és a fényerőt (vagy értéket) pedig százalékban 0-100-ig.



10. ábra: A „spektrum” színtérkép [12]

Mivel a HSV színmodell alternatív reprezentációja az RGB színtérnek, ezért formulák használatával elvégezhető a konverzió. A konverzióhoz a projektben a FastLED könyvtár van használatban, amelynek segítségével az algoritmus mély ismerete nélkül is könnyen létrehozhatók a színek. A könyvtár fejlesztői kiemelik, hogy két fontos különbség van a hagyományos számítógépes HSV színmodell és a FastLED által használt modell között. Az első a színek ábrázolásához használt numerikus értéktartománybeli különbségek (itt minden egy egybájtos érték 0-255-ig), a második a színárnyalatok értékeinek a színekhez való hozzárendelése (a FastLED alapértelmezetten egy gazdagabb „szivárvány” színtérképet használ a hagyományos „spektrum” színtérkép helyett) [13].



11. ábra: A „szivárvány” színtérkép [14]

A könyvtár fejlesztői kiemelik, hogy az egy bájtos értékek használatával a kód kisebb, gyorsabb és hatékonyabb lett. A méréseik szerint a kód így kevesebb, mint feleakkora lett és a színárnyalat számítást 20-szor gyorsabban tudja elvégezni így a program, ami jelentős gyorsulást eredményez. Habár a FastLED könyvtár biztosít egy függvényt a „spektrum” használatához, a projektben az alapértelmezett „szivárvány” színtérkép van használatban. Ennek az az előnye, hogy egyenletesen elosztott színsávokat biztosít. Ide tartozik a sárga szín is, ami így ugyanakkora szélességű sávval és színintenzitással rendelkezik, mint a többi szín. A sárga szín a „spektrum” térkép esetén észrevehetően keskenyebb színsávokkal rendelkezik és így zavarosabban jelenik meg.

VIII. ARDUINO FORRÁSKÓD

A forráskód az Arduino IDE programban került megírásra, amely egy integrált fejlesztői környezet. A helyes Blynk programozási paradigma kissé eltér az Arduino programozástól. A legszembetűnőbb dolog a loop() függvényben levő kevés utasítás. Ez azért van, mert a Blynk folyamatosan adatot küld a felhőszerverre és az egy másodperc alatt küldhető adatok száma korlátozva van 100 darabra, átlépés esetén pedig a szerver automatikusan bontja a kapcsolatot. A loop()-ban a nagy késleltetések használata sem ajánlott, mert gyakori le- illetve felcsatlakozások lehetnek az eredményei, ezért ha adott időközönként elvégzendő eseményt kell lekezelni, akkor a késleltetés helyett időzítőket kell alkalmazni.

A programfejlesztés elején hozzáadásra kerül a Blynk, a FastLED, valamit a DHT11 könyvtár. A forráskód elején a dht11.h illetve a FastLED.h fejlécállományok beemelése történik. Ezek után a DateTime.h állományból kiszedett nevesített konstansok következnek, amelyek az idő átváltása miatt fontosak. A LED-ek száma mellett nevesített konstansként deklarálva lettek azok a digitális tűk, amelyeken keresztül a hangérzékelő, a hőmérséklet- és páratartalom szenzor, valamint a LED szalag csatlakozik a mikrokontrollerhez.

DOI: 10.17667/riim.2018.si/25.

```
//Bluetooth connection from Blynk examples
#include <SoftwareSerial.h>
SoftwareSerial SwSerial(0, 1);  // RX, TX

#include <BlynkSimpleSerialBLE.h>
#include <SoftwareSerial.h>

SoftwareSerial SerialBLE(0, 1); // RX, TX

//Authentication token
char auth[] = "********************************";

void setup()
{
 //Bluetooth connection from Blynk examples
 Serial.begin(9600);

 SerialBLE.begin(9600);
 Blynk.begin(SerialBLE, auth);
}
```

12. ábra: a Bluetooth kapcsolat forráskódja

A Bluetooth kapcsolatot megvalósító kód, amely a 12. ábrán látható, a Blynk oldaláról származik, amely kivitelezi, hogy a lámpa Bluetooth kapcsolaton keresztül vezérelhető legyen a Blynk applikációval. Fontos a forráskódban megadni az autentikációs tokent, amelyet a Blynk emailen küld el. Az autentikációs token egy projektenként egyedi 32 karakter hosszúságú angol kisbetűkből és számjegyekből álló sztring. Mivel a kódban az RX és TX szoftveres soros kommunikációt a digitális 0 és digitális 1 tűk látják el, ezért amikor az Arduino kódot töltjük fel, addig le kell választani a HC-05-öt az Arduino panelről [15].

A Bluetooth kód után a kódban a különböző objektum-, tömb- és változódeklarációk illetve inicializálások következnek. A setup() függvényben a Serial.begin(9600); illetve a SerialBLE.begin(9600); utasítások a soros adatátvitelek sebességének beállításáért felelősek. A Blynk.begin(SerialBLE, auth); utasítás először kiépíti a kapcsolatot az Arduino és az applikáció között a Bluetooth segítségével. Ezután beállítja az autentikációs tokent és az alapértelmezett szerver címét és végül megpróbál csatlakozni a szerverhez. A hangérzékelő tűjének bemeneti módba állítása is itt kap helyet. A FastLED.addLeds<WS2812B, DATA_PIN, GRB>(leds, NUM_LEDS); utasítás megmondja a FastLED könyvtárnak, hogy a 7-es tűn egy WS2812B típusú digitális LED szalag adatvezetéke van, amely GRB (zöld, piros, kék) színkiosztású és ezek a LED-ek a leds tömböt fogják használni és 80 darab van belőlük. A timer.setInterval(25L, SoundEvent); utasítással az időzítő 25 milliszekundumonként fogja meghívni a SoundEvent függvényt.

```
//Sound reactive function
void SoundEvent(){
  int sound=digitalRead(SOUND_PIN);
  if(sound==1){
    for(int i=0;i<NUM_LEDS;i++){
      //leds[i] = CHSV(hue, sat, val);
      leds[i]= CRGB(red, green, blue);
    }
    FastLED.show();
  }
  else{
    for(int i=0;i<NUM_LEDS;i++){
      leds[i] = CRGB::Black;
    }
    FastLED.show();
  }
}
```

13. ábra: A SoundEvent függvény forráskódja

A SoundEvent függvény fogja megvalósítani a hang reaktív szerepkört. Először egy változóba kerül a hangszenzorról beolvasott digitális érték. Ha a változó értéke 1, akkor a leds tömb elemei felveszik az RGB-ben vagy HSV-ben megadott színt. A FastLED.show(); utasítás az összes LED vezérlőnek kiad egy parancsot, hogy a jelenlegi színt jelenítsék meg. A red, green, blue, illetve a hue, val, sat változók értékei a csúszkák segítségével állíthatók be, értéküket alapértelmezetten 0. Ha a 13. ábrán látható sound változó értéke 0, akkor a leds tömb elemei először felveszik a fekete színt, azaz mindhárom komponens 0 lesz. Így, ha a hangérzékelő megfelelő erősségű hangot érzékel, akkor a lámpa a beállított színnel felvillan, amint pedig a hangok erőssége egy bizonyos küszöbérték alá süllyed, abban az esetben pedig a lámpa lekapcsol. A mintavételezés 25 milliszekundumonként történik, azaz másodpercenként 40-szer.

A loop() függvényben csak két utasítás található, az egyik a Blynk.run(); amely a bejövő parancsok feldolgozását és a kapcsolatok kezelését végzi. A timer.run(); a BlynkTimer típusú timer objektum futásáért felelős.

IX. Összegzés

Az Arduino alapú hang reaktív LED lámpa megépítése és vezetéknélküli kapcsolaton keresztüli vezérlése megvalósításra került. A lámpa tesztelése külön-külön funkciónként, majd egész rendszerként történt. Az eszköz a teszteket sikerrel teljesítette, így a hangra aktiválódó funkció, valamint a színállítási lehetőségek mellett a hőmérsékleti- és páratartalom értékek monitorozására és az úgynevezett „uptime” figyelésére is képes az eszköz.

A további fejlesztések között szerepel a hangérzékelő analóg jelekkel való használata, ezáltal lehetőség nyílik arra, hogy különböző küszöbértékek beállításával vizualizálni lehessen a hangot erőssége alapján. Mivel a belső csőre körkörösen felragasztott LED szalag 9 sort alkot, így 9 erősségi szintet lehetne megkülönböztetni. A Blynk könyvtárt és applikációt folyamatosan fejlesztik, így új „widgetek” is kerülhetnének a mobilos felületre, továbbá a forráskód is folyamatosan újabb funkciókkal és megoldásokkal bővülne.

X. KÖSZÖNETNYÍLVÁNÍTÁS

A publikáció elkészítését az EFOP-3.6.1-16-2016-00022 számú projekt támogatta. A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

XI. HIVATKOZÁSOK

[1] B. Gayral, „LEDs for lighting: Basic physics and prospects for energy savings”, Comptes Rendus Physique Volume 18 Issues 7–8, September–October 2017, pp. 453-461

[2] S. Mukherjee, G. P. Biswas, „Networking for IoT and applications using existing communication technology”, Egyptian Informatics Journal, 6 December 2017

[3] PowerSecure Lighting, (2018, May 14). [Online]. Available: http://www.powersecurelighting.com/Blog-44-LED_Lighting___Separating_Fact_from_Fiction

[4] Arduino, (2018, May 14.). [Online]. Available: https://www.arduino.cc/en/Guide/Introduction

[5] Circuit Basics, (2018, May 15). [Online]. Available: http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

[6] Fritzing, (2018, May 15). [Online]. Available: http://fritzing.org/home/

[7] D. Zühlke, N. Thiels, „Useware engineering: A methodology for the development of user-friendly interfaces”, Library Hi Tech Volume 26 Issue 1, March 2008, pp. 126-140

[8] M. Todica, „Controlling Arduino board with smartphone and Blynk via internet”, November 2016

[9] Q. M. Hussein, A. Saadi, N. Q. Mohammed, „The efficiency of Color Models layers at Color Images as Cover in text hiding”, April 2016

[10] J. Vargas, codeitdown, (2018, May 17). [Online]. Available: https://codeitdown.com/hsl-hsb-hsv-color/

[11] Q. M. Hussein, A. Saadi, N. Q. Mohammed, „The efficiency of Color Models layers at Color Images as Cover in text hiding”, April 2016

[12] kriegsman, GitHub, (2018, May 29). [Online]. Available: https://github.com/FastLED/FastLED/wiki/FastLED-HSV-Colors

[13] kriegsman, GitHub, (2018, May 17). [Online]. Available: https://github.com/FastLED/FastLED/wiki/FastLED-HSV-Colors

[14] kriegsman, GitHub, (2018, May 29). [Online]. Available: https://github.com/FastLED/FastLED/wiki/FastLED-HSV-Colors

[15] M. Todica, „Controlling Arduino with Blynk via Bluetooth”, December 2016

[16] Android Linux, (2018, May 17). [Online]. Available: https://www.android.com/

[17] LED Diode, (2018, May 17). [Online]. Available: https://megaled.hu/hu/webshop/led-dioda

[18] T. I. Erdei, Zs. Molnár, N. C. Obinna, G. Husi, „Cyber Physical Systems In Mechatronic Research Centre,” IMTU Oradea - Proceedings of the Annual Seesion of Scientific Papers May 25th - May 27th – 2017.