Recent Innovations in Mechatronics (RIiM) Vol. 5. (2018). No. SI.
DOI: 10.17667/riim.2018.si/23.

Atmel mikrokontroller vezérelt kültéri meteorológia állomás tervezése

Lakatos Ádám
Mérnökinformatikus
Debreceni Egyetem, Informatikai Kar
Debrecen, Magyarország
addam.lakatos@gmail.com

Beatrix Papp
London South Bank University
School of Law and Social Sciences,
London, United Kingdom
pappb@lsbu.ac.uk

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

***Absztrakt*— A hőmérséklet mérése a mai világban elengedhetetlen, de legalább ugyanilyen fontosságú a páratartalom vagy a légnyomás pontos meghatározása. Szükség van ezen értékek pontos ismeretére a mindennapokban, például meteorológiai előrejelzések esetén. Mint ismert a klímaváltozás következtében komoly viharkárok keletkeznek, ennek megelőzése kulcsfontosságú. A hőmérséklet & páratartalom mérése az üvegházak üzemeltetésében is kiemelt prioritású a mezőgazdaságokban. Az Atmel mikrokontroller alapú kültéri mérő szenzor a meteorológiai értékek pontos meghatározásának a céljából készült el.**

Kulcsszavak—Arduino; Arduino UNO; Hőmérséklet; Páratartalom; Légnyomás; Atmel; Meteorológia állomás

I. BEVEZETŐ

Az Arduino egy olyan, széles tömegek számára elérhető fejlesztői platform, amely segítségével bárki könnyen hozzáférhetővé teheti a különböző elektronikai eszközök használatát projektekben vagy akár hobbi szinten is. Mindezt alacsony árával, nyílt forráskódjával, könnyen kezelhető fejlesztői platformjával teszi lehetővé. A hőmérő projektnek az alapját a jelenleg legnépszerűbb modell, az Arduino/Genuino UNO biztosította [1]. A kutatás/fejlesztésnek a Debreceni Egyetem adott otthont [11].

II. TERVEZÉSI SZEMPONTOK

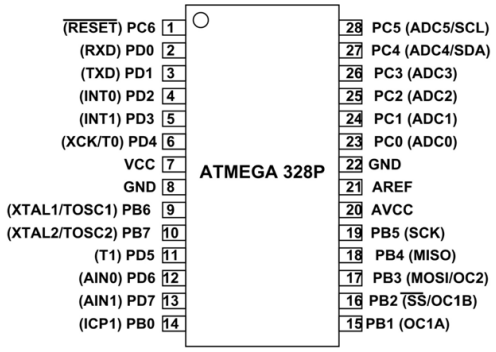
Az elkészítés folyamatát a tervezési szempontok kigondolása előzte meg. Fontos szempont volt, hogy a mért meteorológiai értékek valamilyen vizuális formában megjelenjenek a felhasználó számára. Mivel a külső környezeti tényezők tönkretennék a mérő szenzort, így fontos volt, hogy egy megfelelő védelmet nyújtó burok, doboz készüljön a meteorológiai állmáshoz. A rendszer kialakításánál kiemelt előnyt élvezett az is, hogy működtethető legyen mind vezetékes, mind vezeték nélküli áramforrásról.

Valamint szem előtt kellett tartani, hogy további fejlesztések elvégzésére is lehetőség legyen majd a későbbiekben.

III. A RENDSZER FELÉPÍTÉSE

A meteorológiai állomás alapját egy Arduino Uno szolgáltatja (1. ábra). Maga a hardver a jelenleg legszélesebb körben használt alapmodell modell, amely egy 8 bites, 16 MHz-es ATmega328P processzort és 32 kB flash memóriát használ a működéséhez. USB kapcsolat után van lehetőség a mikrovezérlő programozására, és a „nem felejtő” memóriának köszönhető, hogy biztosítva van a felhasználó számára, hogy az áramellátás után a feltöltött programkód nem vész el [1]. Természetesen az Arduino-ra küldött programkód a felhasználó kedve szerint módosítható. Az Arduino képes analóg, illetve digitális jelek küldésére és fogadására is. Az analóg jel időben folyamatos, amely lehet ciklikusan ismétlődő vagy véletlenszerű, fontos jellemzője az amplitúdó. Digitális jel esetében két állapotot különböztetünk meg: ha áram folyik át, akkor ezt az állapotot 1-es számmal jelöljük, ellenkező esetben az állapotot 0-val jelöljük (bináris jelek). Az Arduino úgy lett kialakítva, hogy bizonyos hibákkal szemben, úgymint fordított bekötés, rövidzárlat ellen védelmet nyújt.

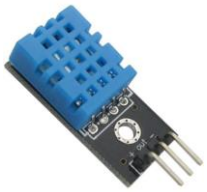
Az Arduino UNO által alkalmazott ATmega328P chip rendkívül sokoldalú, emiatt is alapozta az UNO alap modellt az Arduino coorporation.



1. ábra ATmega328P chip PinOut [9]

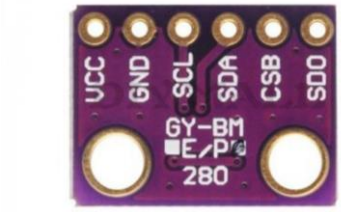
Az Arduino-hoz kapcsolódik egy breadboard, melyre miután kivezettük az 5V-nak és a Ground-nak megfelelő pinekről a vezetékeket, lehetőségünk van különböző modulokat elhelyezni és a mikrokontrollerhez kapcsolni. Így

lett beépítve a rendszerbe egy DHT11 digitális hőmérő, páratartalom mérő modul (2. ábra), a modul 20-80% közötti páratartalom mérésére alkalmas, 5%-os mérési pontossággal, illetve 0-50°C közötti hőmérséklet mérésre, -+ 2°C pontossággal [2]. A modul segítségével meghatározódik az aktuális hőmérséklet, illetve páratartalom. Ezek az értékek eltárolódnak a későbbi műveletek céljából.



2. ábra DHT11 digitális hőmérő, páratartalom mérő modul [2]

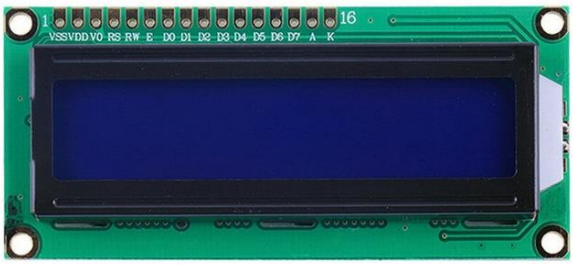
A légnyomás meghatározását egy BMP280 típusú digitális nyomás érzékelő végzi (3. ábra), melynek mérési intervalluma 300-1100 hPa, -+0.12 hPa pontossággal. Az előző modulhoz hasonlóan, az ezzel a modullal mért adatok is eltárolásra kerülnek a további munkavégzés szempontjából [3].



3. ábra BMP280 barométer [3]

További tényezők értékei is meghatározódnak, mint például a talajnedvesség és a jelenlegi esőzés mértéke. Azonban ez a két érték nem szolgál releváns információval, hiszen nagyon minimális területet lehet velük „lefogni”, megbízható és statisztika jellegű információ meghatározáshoz ezeknek a moduloknak a sokszorosa szükséges. A két modul: talajnedvesség érzékelő szenzor, eső érzékelő szenzor.

A különböző modulok által mért értékek visszajelzése a felhasználó felé egy 2x16 karakteres HD44780 LCD kijelzőn történik (4. ábra) [4]. A kijelzőn a felhasználó úgy jeleníti meg a kiírandó értékeket, ahogy szeretné.



4. ábra 2x16 karakteres HD44780 LCD kijelző [4]

Az Arduino, illetve a hozzákapcsolódó alkatrészek, elektronika megvédését, figyelembe véve a kész rendszer méretét, egy műanyag doboz szolgáltatja. Ezen kivágásra kerültek a vezetékek számára szükséges lyukak, a nyomógomb számára szükséges hely, illetve az LCD kijelző számára szükséges nyílás.

A mért adatok megőrzéséhez, azok későbbi felhasználáshoz szükség volt egy adatbázisra. Az értékek egy erre a célja elkészített adatbázisban tárolódnak. Az adatok megtekintésére lehetőség van az adatbázist tartalmazó szoftverben, ami jelen esetben a MySQL Workbench [10], illetve helyi hálózaton belül lehetőség van böngészőben is megjeleníteni az értékeket, egy táblázatban. Ahhoz, hogy helyi hálózaton belül szervert tudjunk létrehozni, az XAMPP Control Panel [5] elnevezésű szoftver alkalmazzuk. Az adatbázisba történő adattovábbításhoz szükség volt egy Ethernet Shield-re. Ez a Shiled az Arduino-ra helyezve látja el a feladatát, és ezen Shield pinjeire csatlakoznak a különböző modulok.

IV. A RENDSZER MŰKÖDÉSE

A rendszer működtetését az Arduino UNO végzi, a programkódban definiálásra kerülnek a különböző pin-ek, azaz, hogy a különböző modulok (digitális hőmérő, páratartalom mérő szenzor, LCD kijelző, barométer) melyik pin-eken csatlakoznak a breadboard-ra, az egyes modulok működéséhez szükséges könyvtárak meghívása végbemegy, az azokhoz szükséges paraméterek beállítása megtörténik. A loop részen belül leolvasásra kerül a DHT11-es modul segítségével az aktuális hőmérséklet, illetve páratartalom, majd kiírásra kerülnek az LCD kijelzőn (5. ábra), illetve a soros monitoron, így adva visszajelzést a felhasználó számára. A leolvasott hőmérséklet eltárolásra kerül, a további műveletek elvégzésének a céljából.

```
61  //* Hőmérséklet és pára *//
62  DHT.read22(szenzorpin);
63  double temp = DHT.temperature;
64  double hum = DHT.humidity;
65
66  // Temp LCD-re való kiíratása //
67  lcd.setCursor(3, 0);
68  lcd.print("Temperature: ");
69  lcd.setCursor(3, 1);
70  lcd.print(DHT.temperature);
71  delay(2000);
72  lcd.clear();
```

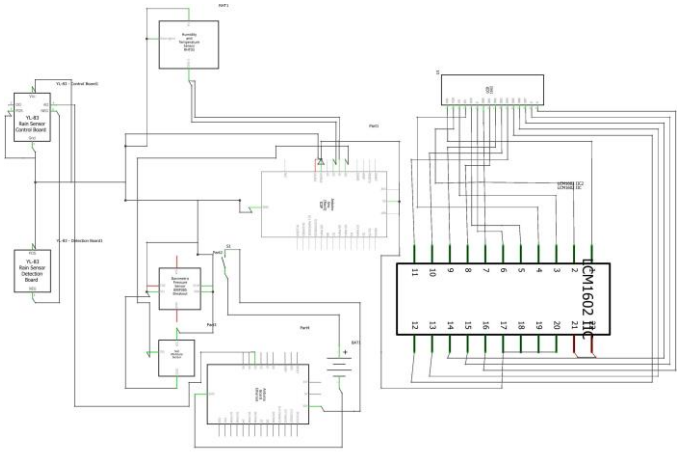
5. ábra Hőmérsékletet és páratartalom leolvasás/megjelenítés

Ezt követi hasonló formátumban a barométer segítségével a légnyomás meghatározása, illetve a talajnedvesség és az esőzés meghatározása. Minden egyes érték meghatározása után az adatok kiíródnak az LCD kijelzőre, így egymás után váltakozva közölve a felhasználóval az aktuális értékeket. Talajnedvesség és az esőzés mérő szenzorok 0 – 1023-ig tartó skálán adnak vissza mért értéket, ahol 1023 a teljesen száraz, míg a 0 a teljesen nedves értéket jelenti. Ezen értékek szemléletesebb megjelenítése érdekében egy skála segítségével könnyebben érthető adatot jelenít meg a program a felhasználó számára, ha a talajnedvesség értéke 800 felett található, akkor „Dry”, 800 és 500 között „Wet”, 500 alatt pedig „Watery” értéket jelenít meg az LCD monitoron. Ezek az értékek a felhasználó igényei szerint alakíthatóak, tetszés szerint.

A programkód következő lényegi része az Ethernet Shield-en keresztül történő adatátvitel, amelynek a főszereplője a PHP nyelven megírt programkód. Ez a programkód végzi a mikrokontroller segítségével mért értékek kinyerését, valamint azok továbbítását az adatbázis felé, továbbá egy szintén PHP nyelven megírt programkód teszi lehetővé a böngészőben való adatmegjelenítést (6. ábra). A delay változó segítségével meghatározhatjuk, hogy a szenzorok milyen időközönként végezzenek adat meghatározást, így például hosszabb ideig történő üzemeltetésnél elegendő lehet, ha csak minden 2. órában végzünk adatfeldolgozást. Ez az opció teljes mértékben a felhasználó igényei szerint alakítható.

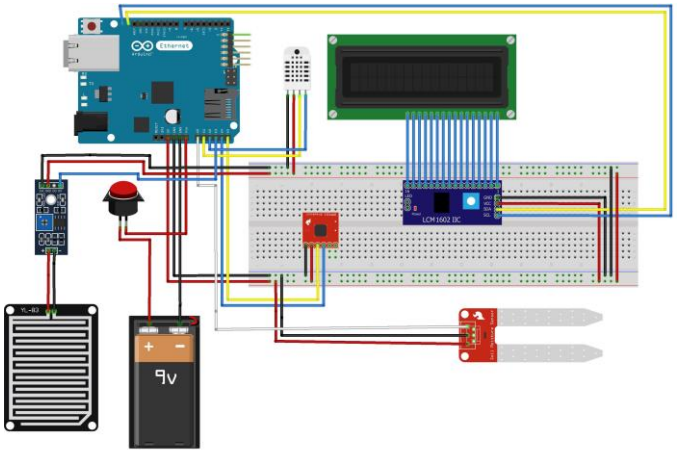
6. ábra Böngészőben történő adat megjelenítés

A programkód „loop” része ismét lefut, mindaddig, amíg az áramellátás meg nem szűnik, vagy a rendszer újra nem indul felhasználói beavatkozásnak köszönhetően. Az értékek feldolgozása a már említett delay függvényében történik. Fritzing [5] elnevezésű nyílt forráskódú szoftver segítségével a felhasználók egyszerűen elektronikus hardvereket tervezhetnek Arduino környezetben, majd ezt felhasználva léphetnek tovább a prototípus elkészítésének következő fázisába. Az előre betöltött áramkörök közül a felhasználó kiválaszthatja a számára legmegfelelőbbet, vagy akár létrehozhat saját változatot is, de rengeteg már előre, mások által elkészített könyvtárakat is megtalálhatóak az interneten. Jelen esetben a kapcsolási rajzhoz szükség volt, egy mások által elkészített könyvtárcsomag felhasználása. Tervezési rajzokból több is a felhasználó rendelkezésére állnak, ilyen például a sematikus, vázlatos rajz (7. ábra).



7. ábra Sematikus ábra

Ezen szoftver segítségével készült el a rendszer kapcsolási rajza (8. ábra).



8. ábra Fritzing szoftverben elkészített kapcsolási rajz

V. ARDUINO SZOFTVER (IDE)

Az Arduino IDE (Integrated Development Enviroment) egy nyílt forrás kódú, a hivatalos Arduino oldalról [6] ingyen letölthető, Java nyelven megírt szoftver, azonban a szoftveren belül megírt programkód leginkább C/C++ nyelvekre hasonlít. A szoftver kezelőfelülete nagyon egyszerű, átlátható. A szoftver beépített példák segítségével a kezdők számára egyszerű, érthető formában magyarázza el az alapvető dolgokat. Ilyen beépített példa például egy Led villogtatása ismétlésszerűen, ez az egyszerű kód a 9. ábrán látható. Az Arduino IDE a megírt programkódot egy hexadecimális formában kódolt szöveges fájllá alakítva, a vezetékes kapcsolat létrejötte után pár másodperc elteltével betölti a „Feltöltés” parancs segítségével az Arduino vezérlő részébe. A programkód feltöltése előtt lehetőség van annak ellenőrzésére az „Ellenőrzés” parancs segítségével, így kiszűrhetőek a szintaktikai, szemantikai hibák. Illetve egy úgynevezett „debugging monitor” is visszajelzést ad a felhasználó számára a programkódban található esetleges hibákról, azok pontos elhelyezkedéséről, így megkönnyítve azok megkeresését és kijavítását. Továbbá lehetőség van arra, hogy a soros monitor (10. ábra) segítségével a felhasználó visszajelzéseket kapjon arról, hogy az egyes modulok helyesen működnek-e, illetve ellenőrzés céljából például kiírassa a változók értékeit, így nyomon követve a program működésének a helyességét.

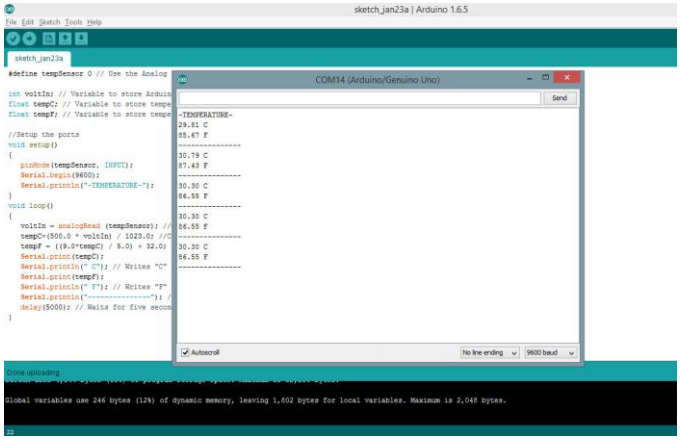
```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

9. ábra Indikátor LED működtetése

A legegyszerűbb, Arduino-ban megírt programkód két részből áll: az egyik a „setup” rész, ez egy úgy nevezetett vázlatot tartalmaz, amely az Arduino minden bekapcsolásakor, vagy újraindításakor betöltődik. Itt találhatóak az inicializált változók, a különböző pin-ek input és output meghatározásai.

A könyvtárak hozzáadása a programkódhoz elengedhetetlen, ezek további programozási eszközöket, metódusokat tartalmaznak. Más felhasználók által készített könyvtárak is elérhetőek, ezek szabadon felhasználhatóak. A másik rész a „loop”, ez a „setup” után található, és a fő programban ciklikusan fut mindaddig, amíg az Arduino működő állapotban van, vagy újra nem indul.

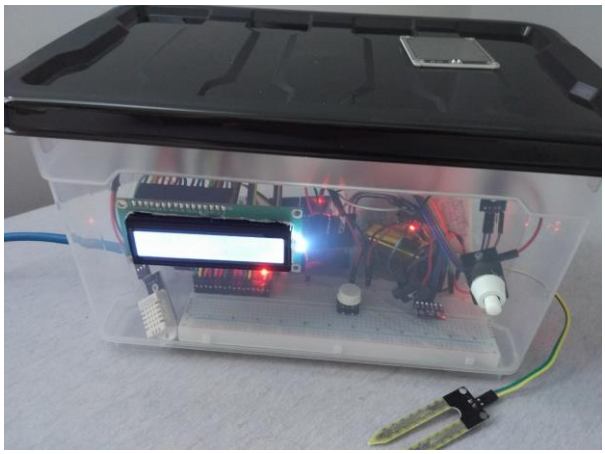


10. ábra Soros monitor az Arudino IDE-ben

VI. TESZTELÉS

A tesztelés első fázisa burkolat nélkül, szobában történt, így megvizsgálva, hogy a rendszer az elvárásoknak megfelelően működik-e. Első tesztelés után minimális módosításokra volt szükség, lényegében csak a felhasználó igényeit befolyásoló tényezők kerültek módosításra, ezek: a talajnedvesség és az esőzés értéke alapján megjelenített információ az LCD kijelzőn. A tesztelés második fázisa az adatbázis és az Arduino összekapcsolása volt. A tesztelések sikeresek voltak, azoban itt is szükséges volt néhány módosítás elvégzése, melyek ugyanúgy, mint az első tesztelés esetében a talajnedvesség és az esőzés által mért adatokat érintette. A localhost-ban megjelenített honlapon található táblázatban a 0-1023-ig tartó skálán elhelyezett mért érték helyett szöveges információ megjelenítése került módosításra.

Az Arduino mikrokontroller alapú meteorológiai állomás a 11. ábrán látható.



11. ábra Arduino alapú meteorológiai mérőszenzor

Tesztelésnél fontos volt, hogy esős időben is tesztelni lehessen, ez az elkészített tároló szempontjából volt szükséges. Meg kellett győződni róla, hogy a meteorológiai mérőszenzor védve van a külső tényezőktől. A tesztelések az előzetes elvárásoknak megfelelően zajlottak, illetve nem jelentkezett semmilyen váratlan meghibásodás.

VII. ÖSSZEGZÉS & FEJLESZTÉSI LEHETŐSÉGEK

Fejlesztési lehetőség, ami a legfontosabb a jövőben, hogy Ethernet Shield helyett egy WiFi Shield [7] szolgáltassa az adatátvitelt, ezzel biztosítva, hogy a rendszer mentesítve van minden vezetékes kapcsolattól. Egy domain cím bérlése lehetővé tenné, hogy bárhonnan elérhetőek legyenek a mérőszenzorral mért adatok, ezáltal bárki számára megtekinthetővé válva. Ha a WiFi-s modul beépítése megtörténne, akkor ezt követné az adott rendszer sokszorosítása, ezáltal nagyobb területen lenne lehetőség a meteorológiai értékek meghatározására. Az így kinyert értékek már nagyobb relevanciával bírnának, mint a jelenlegi rendszer esetében. Az így összegyűjtött adatokat lementve a már meglévő adatbázisba, lehetőség lenne statisztika jellegű adatfeldolgozásra, így akár régióra lebontva megjeleníteni a mért adatokat.

Fontos szerepe lenne a mobilitásnak, abban a tekintetben, hogy a felhasználó ne csak akkor kapjon figyelmeztetést, amikor közvetlen a rendszer mellett helyezkedik el és internet hozzáférése is van, hanem ezen tényezőktől függetlenül is. Ennek a megvalósításnak az alapját egy GSM Shield [8] biztosítaná, amely segítségével a felhasználó SMS formában értesülne a rendszer felől.

Az elkészült eszköz minden bekapcsolás után megkezdi az adott tényezők mérését, majd azokat felváltva megjeleníti az LCD kijelzőn, így adva visszajelzést a felhasználó számára. A háttérben pedig a felhasználó igénye szerint meghatározott időközönként megtörténik az Ethernet Shield-en keresztül az adatküldés. A hőmérséklet érzékelés lassúsága a hőmérséklet érzékelő szenzorból adódik, egy drágább szenzor gyorsabb változás érzékelést jelentene. Ez igaz a többi szenzor esetében is.

VIII. ÖSSZEGZÉS

A mikrokontroller alapú meteorológiai mérőszenzor az előzetes terveknek megfelelően elkészült. További fejlesztési szempontok kivitelezése a lehetőségekhez mérten elkészülnek a jövőben.

IX. KÖSZÖNETNYÍLVÁNITÁS

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. Köszönet illeti a Debreceni Egyetem Műszaki Kar Mechatronikai Tanszék oktatóját, Erdei Timotei Istvánt, a segítségnyújtásért.

X. HIVATKOZÁSOK

[1] Arduino Uno, (2018. 05. 07.) [Online]. https://www.arduino.cc/en/Products/Counterfeit

[2] DHT11 (2018. 05. 07.) [Online] https://www.inventelectronics.com/product/dht11-temperature-humidity-sensor/

[3] BMP280 (2018. 05. 07.) [Online] https://www.makerfabs.com/image/cache/makerfabs/BMP280%20Barometer/BMP280%20Barometer_1-1000x750.jpg

[4] 2x16 karakteres HD44780 LCD kijelző (2018. 05. 07.) [Online] https://www.aliexpress.com/item/LCD-Display-For-Raspberry-PI-LCD-1602-Display-LCD1602-HD44780-LCD-Module-16x2-DIY-KIT-5V/32473651550.html

[5] Fritzing (2018. 05. 08.) [Online] Avalaible: http://fritzing.org/home/

[6] Arduino (2018. 05. 08.) [Online] Avalaible: https://www.arduino.cc/

[7] WiFi Shield (2018. 05. 08.) [Online] https://store.arduino.cc/arduino-wifi-shield

[8] Arduino GSM Shield (2018. 05. 08.) [Online] https://www.arduino.cc/en/Guide/ArduinoGSMShield

[9] ATMEGA328P (2018. 05. 08.) [Online] https://www.circuito.io/blog/arduino-uno-pinout/

[10] MySQL Workbench (2018. 05. 08.) [Online] https://dev.mysql.com/downloads/workbench/6.1.html

[11] T. I. Erdei, Zs. Molnár, G. Husi, „Robot visual and virtual control technology In industrial environment," WoS (Web of Science) publication, International Symposium on Small-Scale Intelligent Manufacturing Systems (SIMS), Narvik, NORWAY- IEEE, Jun 21-24, 2016.