Multidiszciplináris tudományok, 11. kötet. (2021) 4 sz. pp. 53-59 **https://doi.org/10.35925/j.multi.2021.4.5**

Mérőrendszer, kommunikáció és adatfeldolgozás kialakítása vezető nélküli targoncán

Cservenák Ákos
tudományos segédmunkatárs, Miskolci Egyetem, Logisztikai Intézet
3515 Miskolc, Miskolc-Egyetemváros, e-mail: cservenak.akos@uni-miskolc.hu

Absztrakt

A cikk bemutatja egy vezető nélküli szállítótargonca vezérléséhez szükséges feszültség- és áramerősség adatok mérését, továbbítását és feldolgozását. A mérés Arduino Uno fejlesztőplatform segítségével történik, amelynek bemenő feszültségértéke legfeljebb 5V lehet. A cikk bemutatja a mérőrendszer elektronikai kapcsolását, és Arduino-ra megvalósított szoftveres hátterét. A targonca vezérléséhez való illesztéshez szükség van a fejlesztőplatform és a PC közötti kommunikáció kialakítására, és azon történő adatfogadásra- és feldolgozásra, a cikk kitér ennek az informatikai résznek a megvalósítására is.

Kulcsszavak: *mechatronika, AGV, adatgyűjtés, autonóm rendszer, mérés*

Abstract

This paper describes the measurement, sending and processing of voltage and current data required to control an AGV (Automated Guided Vehicle). The measurement is performed using an Arduino Uno microcontroller with an 5V input maximal allowed voltage. The paper presents the electronic circuit of the measuring system and its software background for Arduino. Adaptation to the control of the AGV requires the establishment of communication between the microcontroller and the PC and the reception and processing of data thereon, the paper also covers the implementation of this IT part.

Keywords: *mechatronics, AGV, data collection, autonomous system, measurement*

1. Bevezetés

Manapság az Ipar 4.0 meghirdetése révén az automatizálás és mobil eszközök terjedése felgyorsult, és ez a gyártástámogatás terén is megjelent [1]. Egy logisztikai folyamat tervezése során mindig a modernizálásra kell törekedni [2]. A robotika egyre inkább teret hódít az automatizálási eszközök terén, mind az ipari robotok [3], mind a mobil robotok terén [4].

A Miskolci Egyetem Logisztikai Intézetének High-Tech Logisztikai Rendszerek Laboratóriumában található egy prototípus AGV [5]. Ezen AGV a laboratórium költözése után már nem tudott automatikusan működni, mivel a korábbi helyéhez kötötten alakították ki a mozgásvezérlést [6]. Korábbi kutatás témája egy olyan új pálya-és trajektória tervező megoldást mutat be, amely nem helyiséghez kötött, hanem áttelepítés után újra használható.

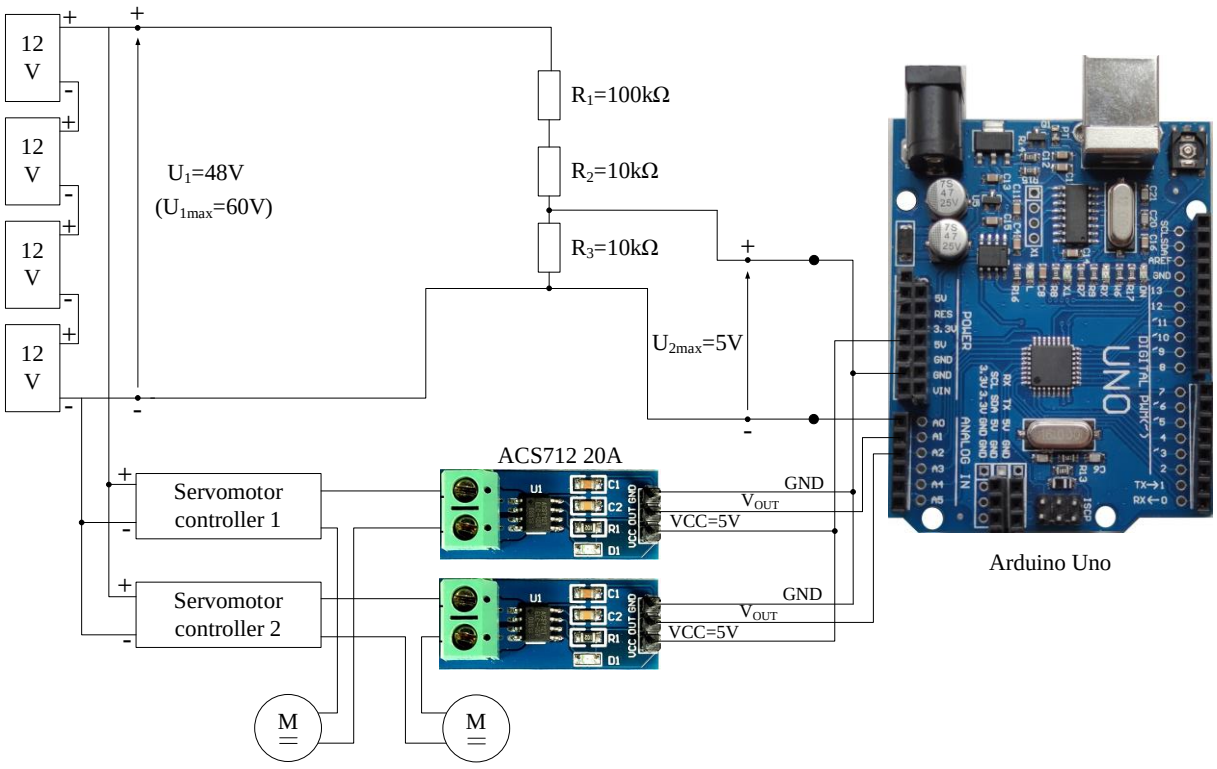
A két pont közötti mozgásszabályozáshoz és szimulációhoz a következő modulokra van szükség: 1. útvonaltervező modul, 2. pályatervező modul, 3. sebesség-feszültség átalakító modul a pályatervezőtől kapott sebességek felhasználásával, 4. mozgásvezérlés és motordinamikai modell szimulálása feszültségek felhasználásával az átalakítóból, 5. az út szimulációja és 6. adatfeldolgozás. A 3., 5. és 6. modulok a [7] irodalomban, míg a 4. modul a [8] irodalomban került megvalósításra.

Jelen cikk egy olyan mérőrendszer kialakítását mutatja be, amely a 4. és 5. modulokhoz valósít meg visszacsatolást. A cikk 2. fejezete a mérőrendszer áramerősség- és feszültség mérésre vonatkozó részét mutatja be. A 3. fejezet a mérési eredmények PC-n történő felhasználását foglalja össze. Végül a 4. fejezet összefoglaló megállapításokat tesz a cikkre vonatkozóan.

2. Mérőrendszer kialakítása - Arduino rész

2.1. Az Arduino Uno fejlesztőplatformhoz kapcsolt elektronikai rész

A feszültség- és áramerősség mérésekhez szükség van Arduino Uno fejlesztőplatformra illeszthető elektronikai elemekre [10], amelyet például a [11] értekezésben is felhasználtak. A feszültségmérés egy feszültségosztó, míg az árammérés az ACS712 eszközzel történik.



1. ábra: *Feszültség- és áramerősségmérő kapcsolás az Arduino Uno fejlesztőplatformmal*

A feszültségmérés a 4db sorba kötött, egyenként 12V-os névleges feszültségű akkumulátorra történik. Névleges feszültségük sorba kötve $U_1 = 48V$, maximális feszültségük elérheti az 57,6V-ot, így érdemes $U_{1_{max}} = 60V$ feszültség konvertálását $U_{2_{max}} = 5V$-ra kiszámítani. Egy feszültségosztó már kialakításra került korábban [9] erre a vezető nélküli targoncára, azonban akkor STM32F4 Discovery fejlesztőplatform alkalmazásával, ami 3,3V bemeneti feszültséget tolerált. A kialakítást és összefüggéseket tartalmazó cikket felhasználva a névleges szekunder feszültség [9]:

$$U_{2\max} = \frac{R_3}{R_1+R_2+R_3} \cdot U_1 = \frac{10k\Omega}{100k\Omega+10k\Omega+10k\Omega} \cdot 60V = \frac{1}{12} \cdot 60V = 5V \qquad (1)$$

Az ellenállások valóságos értékei miatt a szekunder feszültség nem lesz azonos.

$$U_2 = \frac{R_3}{R_1+R_2+R_3} \cdot U_1 = \frac{10{,}00k\Omega}{100{,}1k\Omega+10{,}01k\Omega+10{,}00k\Omega} \cdot 60V = \frac{1}{12{,}011} \cdot 60V = 4{,}99542V \quad (2)$$

A viszonyszámra (12,011) a PC-n való programozás során van szükség. Az Arduino Uno fejlesztőlatform 10 bites A/D átalakítóval rendelkezik, így az analóg feszültségértéket $2^{10} = 1024$ felbontással tudja digitális jellé alakítani. Így a mért feszültségérték a számítógépes programba történő átalakítási folyamata:

$$U_{Arduino} = 0 \div 5V \rightarrow value_{Arduino}: 0 \div 1023 \rightarrow U_{program} = 0 \div 60V \quad (3)$$

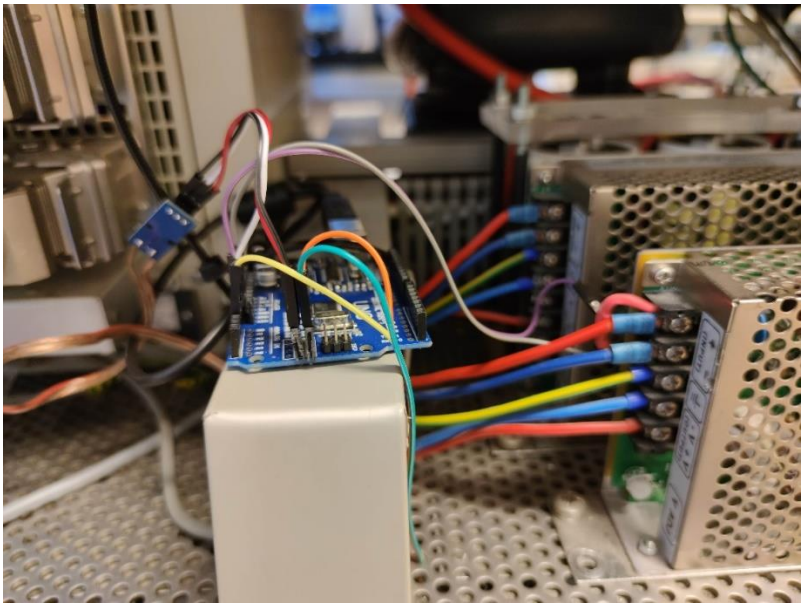
A mérési pontosság az Arduino és a program részéről:

$$Arduino: \frac{5V}{1024} = 0{,}0049V \quad (4)$$

$$program: \frac{60V}{1024} = 0{,}0586V \quad (5)$$

Azaz a programban a következő számítás történik az Arduino értékét felhasználva:

$$U_{program} = \frac{value_{Arduino}}{1024} \cdot 5V \cdot 12{,}011 \quad (6)$$



2. ábra: *Feszültség- és áramerősségmérő kapcsolás beszerelve a targoncába*

Az árammérés 2db ACS712 eszközzel történik, az egyik a bal oldali kereket hajtó motor, míg a másik a jobb oldali kereket hajtó motor áramerősségét méri. Mivel ezek az elektronikai eszközök is az Arduino fejlesztőplatformra vannak kötve, így a valós áramértékből a következő átalakítási úton lesz felhasználható érték a program számára:

$$i_{left,Arduino} = -20A \div 20A \rightarrow value_{Arduino}: 0 \div 1023 \rightarrow i_{left,program} = 0A \div 40A \rightarrow$$
$$\rightarrow i_{left,program,converted} = -20A \div 20A \quad (7)$$

A konvertálás a következőképpen történik:

$$i_{left,program} = \frac{value_{Arduino}}{1024} \cdot 40A - 20\text{A} \tag{8}$$

Az (5) és (6) összefüggések igazak a másik árammérőre is. Az elektronikai kapcsolást a 1. ábra, míg a targoncába helyezett kapcsolást a 2. ábra mutatja. Az egyik árammérőhöz, ami a 2. ábra bal oldalán látható, a vezetékek a targonca túloldalán levő szervomotor szabályzóról érkeznek, míg a másik árammérő a targonca ábra szerinti közelebbik oldalán levő szervomotor szabályzó mellett helyezkedik el.

2.2. Az Arduino Uno fejlesztőplatformra készített szoftveres rész

Az Arduino megírt program a feszültségosztóból érkező arányos feszültségjel, és az árammérőkből érkező feszültségjelek átalakításáért felelős a PC-re történő tovább küldés érdekében.

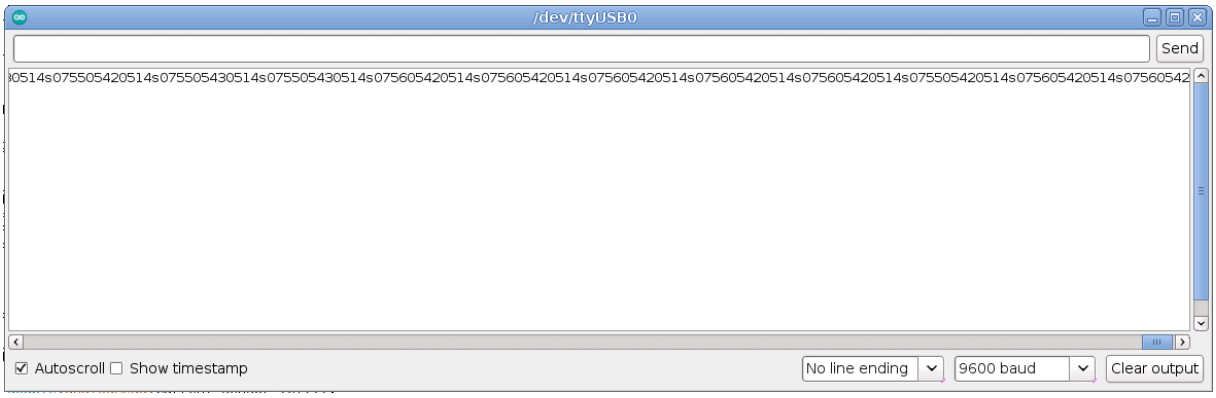
A program első részében a változók deklarálása történik meg, mint az elektronikai elemek pin-jének hozzárendelése, és további segédváltozók. A második részben a beállítások történnek meg, mint a soros port nyitása, az előző deklarált pin-ek módjának beállítása.

A főprogram előtt még találhatók további alprogramok:

- u_average1024(): a feszültségosztóról beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartománnyá
- i_left_1024(): az egyik árammérőről beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartománnyá
- i_right_1024(): a másik árammérőről beérkező feszültségjel mintavételezése 100-szor, majd az ebből képzett átlag átalakítása 0-1023 tartománnyá

A soros portos üzenet küldéshez még egy „Serial_0000” alprogram került megírásra. Önmagában az előző 3 alprogram 1 és 4 közötti helyiértékű értéket ad vissza, amely a PC oldaláról az üzenet figyelése miatt bizonytalanná válhat, így érdemes fix 4 helyiértékű értéket, azaz 0000 és 1023 közötti értéket továbbítani. Ennek megoldása az adott változó értékének figyelése, és ha 10-nél, 100-nál vagy 1000-nél kisebb értékű, akkor a tényleges érték elé 3, 2 vagy 1 db „0” -t helyez el a soros portos kiíratásnál, sorrendben.

A főprogram ezek után először egy „s” értéket írat ki a soros portra, ezzel jelezve, hogy az üzenet elkezdődik, majd ezután a feszültségosztóról, mindkét árammérőről érkező feszültségjel „Serial_0000” alprogrammal történő soros portos kiíratás.



3. ábra: *Feszültség- és áramerősségmérők értékeinek továbbítása a PC-re*

A működésre egy példa a 3. ábrán látható. Megfigyelhető, hogy a feszültségértékre „0755” adódik, amiből a (4) összefüggést felhasználva $U_{program} = 42{,}729V$, a kettő árammérőnél „0542” és „0514” adódik, amiből a (6) összefüggést felhasználva $i_{left,program} = 1{,}172A$ és $i_{left,program} = 0{,}078A$ értékek keletkeznek.

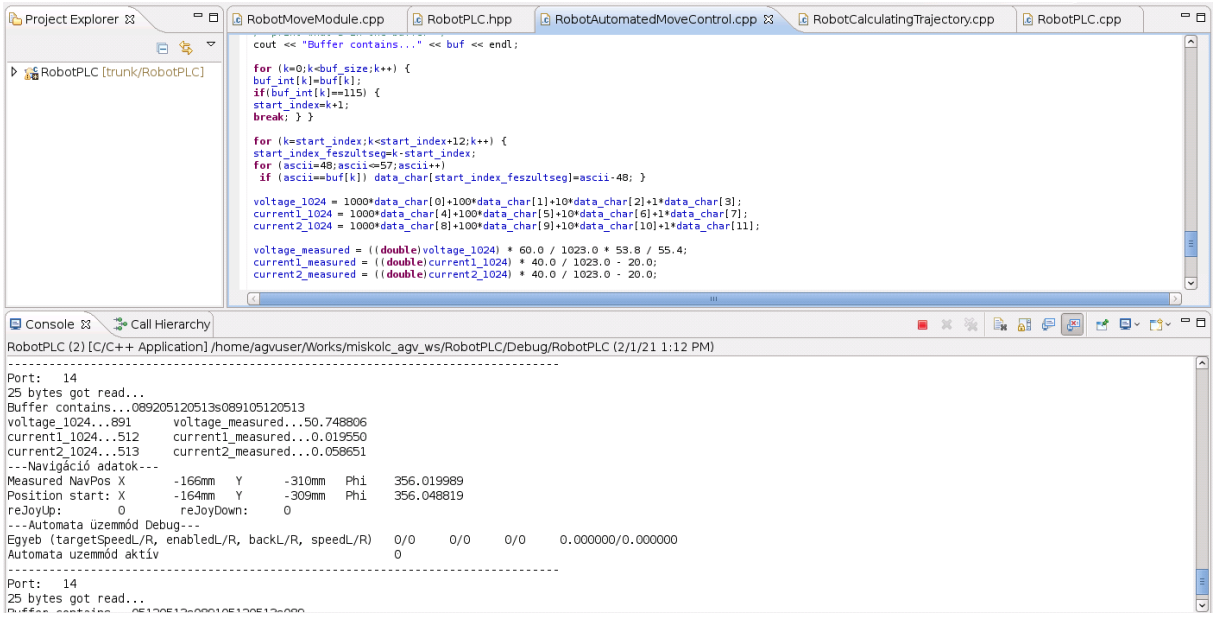
3. Mérőrendszer kialakítás - PC informatikai rész

A PC oldaláról az Arduino fejlesztőplatformról érkező üzenet feldolgozása történik meg.

Első körben az Arduino fejlesztőplatformról való kommunikációt kell megvalósítani. A PC-re az Arduino USB porton keresztül csatlakozik, így a PC ezt a portot kell kezelje. A targoncavezérlő programban először ezt az USB portot kell megnyitni, ami Linux alapú rendszeren a „ttyUSB0” lesz. A port megnyitása után több beállítást is kell eszközölni, majd ezután fogadhatja az üzenetet.

Mivel az Arduino és PC közötti kommunikáció aszinkron jellegű, azaz a PC-re megérkező üzenet nem biztos, hogy az „s” karakterrel kezdődik. Ennek érdekében egyszerre annyi karaktert kell beolvasni, hogy az „s” és az utána következő 12 karaktert biztosan beolvassa, még ha az „s” a 13. helyen is van. Ennélfogva összesen 25 karaktert kell kezelnie a PC programjának.

A 25 karakteres változóból a programnak karakterről karakterre léptetve meg kell keresnie az „s” karaktert, és rögzítenie a karakter sorszámát, ami 1 és 13 között lehet. Ezután tovább léptetve a karaktereken négyesével a karaktereket át kell alakítani egész számmá, amelyhez az ASCII-tábla került felhasználásra. Az egész számmá alakítás után történhet meg a (4) és (6) összefüggések szerinti átszámítás.



4. ábra: *Feszültség- és áramerősségmérők értékeinek feldolgozás a PC-n*

A 4. ábra mutatja ennek a feldolgozásnak az eredményét. Az ablak felső részén a programkód egy része látható, míg az alsó részén a kiíratott rész. A legelső sorban lehet látni, hogy a 14. számú porton 25 byte-ot olvasott be, ezután a beolvasott karaktersorozat látható, hasonlóan a 3. ábrán láthatóhoz és itt a 13. karakterre érkezett az üzenete kezdetét jelző „s” karakter. A következő 3 sorban a feszültség- és

árammérőkről érkező értékek és annak átalakított verziói olvashatók. Az ezután levő sorok már a navigációadatokról és üzemmódállapotokról ad tájékoztatást, amelyek most nem relevánsak.

4. Összefoglalás

Ez a cikk bemutatta egy vezető nélküli szállítójármű szabályozásához szükséges feszültség- és áramerősség adatok mintavételezését, Arduino Uno fejlesztőplatformon át PC-re továbbítását, majd PC-n történő információ feldolgozását.

Az első rész bemutatta az Arduino fejlesztőplatformhoz kapcsolt elektronikai kapcsolást, amely tartalmazza a 4db sorba kötött akkumulátor feszültségét mérő feszültségosztót, valamint a 2db ACS712 árammérőket, amelyek a targonca meghajtásáért felelős szervomotorok áramfelvételét méri. Ezután a fejezet kitért még az Arduino fejlesztőplatformra megírt program felépítésére, amely mintavételezi a 3 egység bemeneti feszültségjelét, majd előre megadott formában továbbítja a PC számára.

A második rész a PC-re megírt programrészlet felépítését részletezi. Elsőkörben az Arduino fejlesztőplatformmal való kommunikáció létrejöttét, majd az Arduino-ról érkező aszinkron adatok fogadását, szétválasztását, és karaktersorozatból egész számú értékek előállítását írja a fejezet. Az egész számú adatokat különböző összefüggések segítségével átalakítja a vezérlőprogram számára használható adattá. A jövőben ezen adatok felhasználása történik a targonca áramfogyasztás mérése érdekében.

5. Köszönetnyilvánítás

A cikkben ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése” projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

Irodalom

[1] He, W.: *Production allocation technologies for industrial product assembly lines based on the Internet of Things*, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp 158-163 (2020)

[2] Liu, W.: *Production scheduling and equipment matching of flexible workshops based on multi-objective and multi-process hybrid optimization algorithm*, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp 151-157 (2020)

[3] Benotsmane, R., Dudás, L, Kovács, Gy.: *Simulation and trajectory optimization of collaborating robots by application of Solidworks and Matlab software in Industry 4.0*, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp 191-197 (2020)

[4] Wang, X., Gao, J.: *An AGV scheduling algorithm for smart workshops with limited logistics capacity*, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp 23-27 (2020)

[5] Tamás, P., Bányai, T., Illés, B., Tollár, S., Veres, P., Cservenák, Á., Hardai, I., Skapinyecz, R.: *Development Possibilities of the High-tech Logistics Laboratory Established at the Institute of Logistics of the University of Miskolc*, Journal of Engineering Research and Reports, Vol 13, No. 3, pp. 60-68. (2020) **https://doi.org/10.9734/jerr/2020/v13i317127**

[6] Papp, Á., Szilassy, L., & Sárosi, J.: *Navigation of differential drive mobile robot on predefined, software designed path*. Recent Innovations in Mechatronics (RIiM), 3, 1-2. (2016) **https://doi.org/10.17667/riim.2016.1-2/3**

[7] Cservenák, Á. *Simulation of a mobile robot's motion*, Academic Journal of Manufacturing Engineering, elfogadva, megjelenés alatt (March 2021)
[8] Cservenák, Á. *Simulation and modelling of a DC motor used in a mobile robot*, Academic Journal of Manufacturing Engineering, Vol. 18, No. 4, pp 183-190 (2020)
[9] Cservenák, Á.: *Further development of an AGV control system*, Lecture Notes in Mechanical Engineering, pp. 376-384., 9 p. (2018) **https://doi.org/10.1007%2F978-3-319-75677-6_32**
[10] ATMEL:*8-bit AVR Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash*, datasheet, Rev. 8025I–AVR–02/09, 2009
[11] Rónai, L.: *Robotok heptikus tulajdonságának fejlesztése gyártási és szerelési folyamatok automatizálására*, 102 p., Sályi István Gépészeti Tudományok Doktori Iskola, Disszertáció (2020)