Mikroelektromechanikai rendszerek (GKLB\_INTM020)

Fejlesztői dokumentáció  
Beléptetőrendszer

**Tartalomjegyzék**

[Fejlesztői dokumentáció Beléptetőrendszer 1](#_Toc122186927)

[Tervezési fázis 1](#_Toc122186928)

[Felmerülő problémák 1](#_Toc122186929)

[Áramköri rajz 2](#_Toc122186930)

[Felhasznált elemek 3](#_Toc122186931)

[Hardver specifikációk 4](#_Toc122186932)

[Szoftver specifikációk 4](#_Toc122186933)

[Rendszerkövetelmények 4](#_Toc122186934)

[Fejlesztett kód részletezése 5](#_Toc122186935)

[Képek a kész projektről 9](#_Toc122186936)

# Tervezési fázis

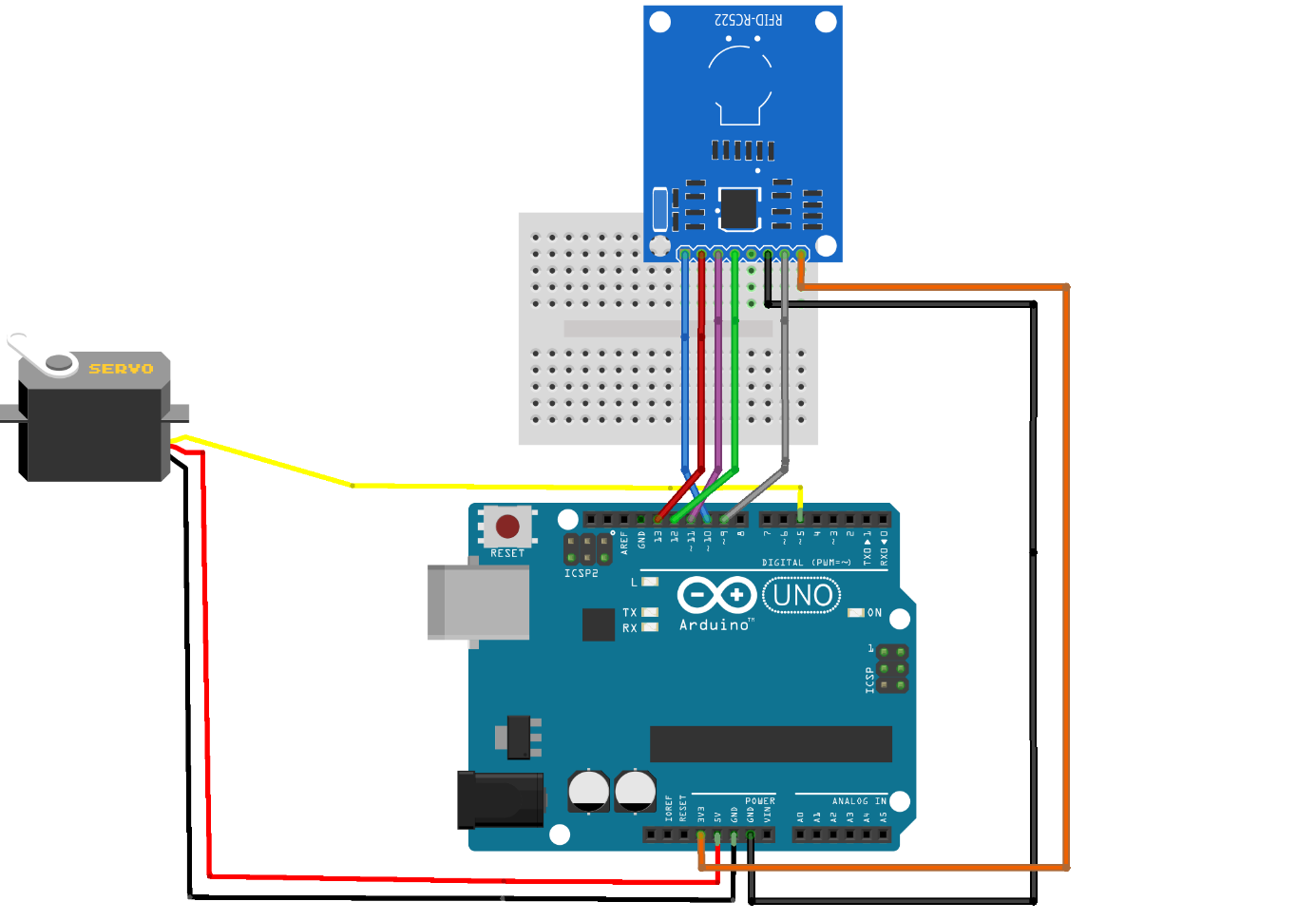
Megadott problémára szerettem volna egy gyors és praktikus megoldást találni. Ezért építettem meg azt a beléptetőrendszert, amely segítségével egyetlen olvasó és kártya összeérintésével megforgat egy szervót és ennek segítségével demonstrálok egy ajtózár kinyitást. A megalkotott koncepció felvázolása és az alkatrészek beszerzése után következett az összeépítés.

# Felmerülő problémák

Szükségesnek láttam az olvasó forrasztását a hozzá vásárolt lábakkal a stabil adatátvitel érdekében.  
Az első olvasó, amelyet vásároltam, utólag kiderült, hogy gyári hibás volt, így egy 3 hetes levelezést követően a gyártóval, sikerült egy új olvasót kapnom, amit természetesen ismét meg kellett forrasztanom.

A kód megírása során a megfelelő UID-kat ki kellett olvasni, máskülönben nem működött a program.

# Áramköri rajz



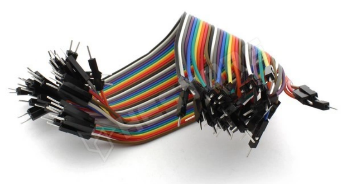
**Összekötésük:**

Olvasó (SDA) -> Arduino (~10)  
Olvasó (SCK) -> Arduino (13)  
Olvasó (MOSI) -> Arduino (~11)  
Olvasó (MISO) -> Arduino (12)  
Olvasó (GND) -> Arduino (GND)  
Olvasó (RST) -> Arduino (~9)  
Olvasó (3.3V) -> Arduino (3.3V)

Servo (Sárga kábel) -> Arduino (~5)  
Servo (Piros kábel) -> Arduino (5V)  
Servo (Fekete kábel) -> Arduino (GND)

# Felhasznált elemek

**A megépítés során felhasznált elemek:**

* AR-UNOR3  
  UNO R3 fejlesztői panel + USB kábel
* RC522-MFRC  
  MFRC522 RFID Mifare író/olvasó szett   
  (Modul 13.56MHz + kulcs tag + kártya tag)
* MS/SG-90 servo  
  Micro szervó (SG90), 9gr, analóg, 180°
* BB-170W  
  Próbapanel 170p, hófehér
* RC-40-20/MM  
  Szalagkábel csatlakozóval, 20cm, 40p,  
  male-male (jumper)

# Hardver specifikációk

Azon hardverek összessége, melyek segítségével futtattam a programot és demonstráltam a beléptetőrendszert.

Lenovo IdeaPad Gaming 3

* CPU: Intel Core I7-12650H
* RAM: DDR4 16 GB 3200 MHz
* GPU: nVidia Geforce RTX 3050 Ti 4GB GDDR6
* Storage: 512 GB PCIe SSD

# Szoftver specifikációk

Azon szoftverek, melyek segítségével futtattam a programot és demonstráltam a beléptetőrendszert.

* Windows 10 Ultimate
* Arduino IDE

# Rendszerkövetelmények

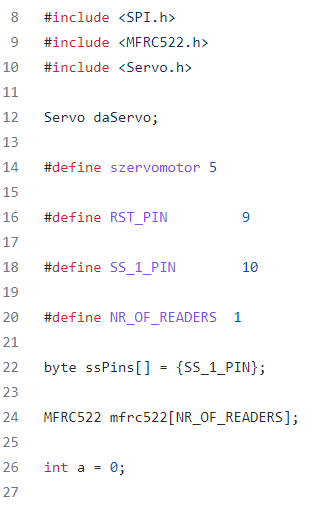
Minimum rendszerkövetelmények a megfelelő működéshez:

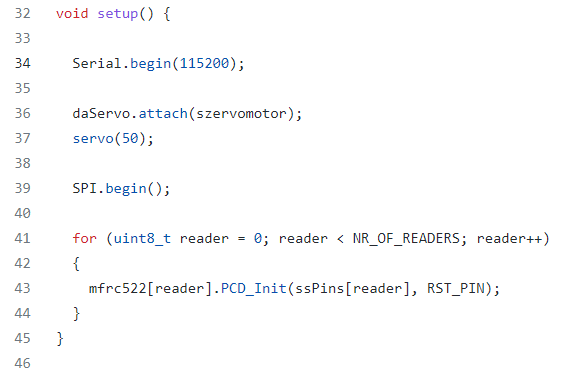
* GPU: Intel Core I3
* RAM: DDR4 4GB
* GPU: nVidia Geforce GTX 1060 / AMD Radeon HD 7000
* Storage: 10 GB

Ajánlott rendszerkövetelmény a megfelelő működéshez:

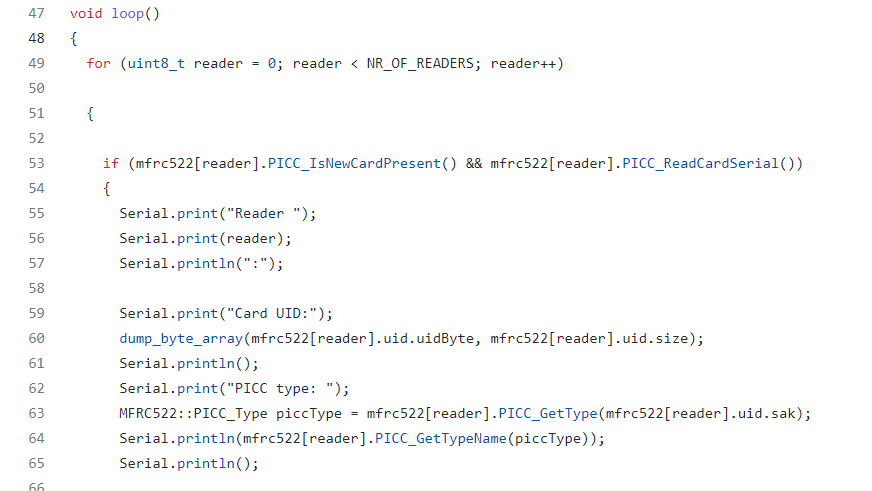
* GPU: Intel Core I5
* RAM: DDR4 8GB
* GPU: nVidia Geforce GTX 1080 Ti / AMD Radeon 500
* Storage: 10 GB

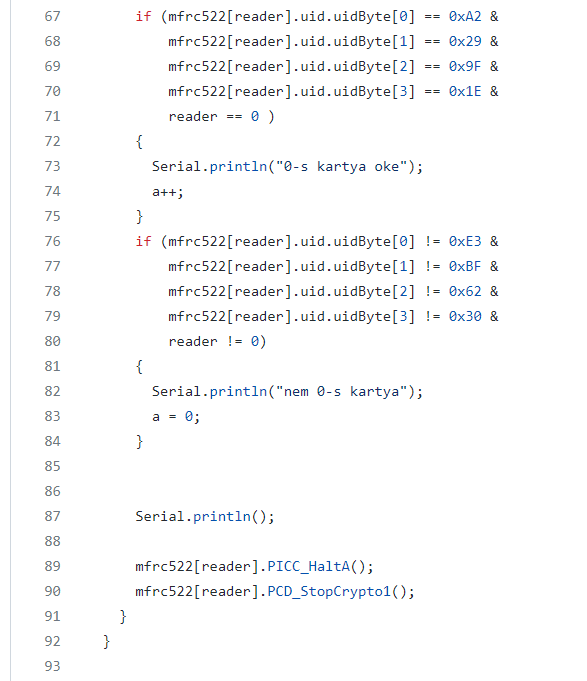
# Fejlesztett kód részletezése

Az első sorokban betöltjük a szükséges fájlokat a működéshez. (SPI,MFRC522,Servo)  
Majd létrehozzuk a saját szervó prototípusunkat daServo néven. A következő sorokban definiáljuk a szervómotor, az RST\_PIN, SS\_1\_PIN makrókat, majd utána a hozzá kötött Arduino láb számát írjuk.  
Az utolsó makróval azt adjuk meg, hogy hány darab RFID olvasót használunk.  
A következő sorban egy byte típusú tömbben megadjuk az SS pinjeit, ha van több, mindet bele kell írni vesszővel elválasztva. Majd létrehozzuk a saját MFRC522-es prototípusunkat, az olvasó nevét beleírva szögletes zárójelbe. A definiálások végén még egy változót létrehozunk, aminek később lesz szerepe.

A setup-ban elindítjuk a soros kommunikációt 115200-as bit/sec sebességgel. Utána megadom, hogy az én szervómotorom, szervómotor néven van megadva. Majd alapállásba állítom a szervómat. Azért valósítható meg így, mert a program végére készítettem erre egy funkciót). Ezután az SPI buszt inicializálom. A for ciklus azért van, ha több olvasót építek hozzá, akkor mindet egyesével tudja inicializálni.

A loop egy for ciklussal kezdődik, azért, hogy ha több olvasónk van, akkor annyiszor lefusson a program. Rögtön utána egy feltétel található. Ha érzékel egy kártyát és le is olvassa, akkor az első sorokban kiiratjuk, hogy hányas olvasóval olvastuk le a kártyát, majd a hozzá tartozó UID-ját decimálisan. Ez a decimális kiírás is egy külön függvénnyel működik. Ezután a TAG típusát is kiírjuk tájékoztatásként.



A következő 2 if elágazás a kiolvasott kártya UID-jának vizsgálata. Megnézzük négyfelé osztva, hogy megegyeznek-e, valamint azt, hogy 0-s olvasóval lett leolvasva. Ha igen, akkor kiírja a kimenetre, hogy „0-s kártya, oké”. Majd növeljük a program elején deklarált **a** változót. A for ciklus miatt úgy épülnek fel a feltételek, hogy szögletes zárójelben meg kell adni, hogy hányas olvasó és hányas byte-ját olvasta ki. A másik if elágazás ennek az ellentéte. Tehát ha nem egyezik valamelyik byte, vagy nem a 0-s olvasóval olvastuk le a TAG-et, akkor kiiratjuk, hogy „nem 0-s kártya”, majd lenullázzuk az **a** változót.  
Majd leállítjuk az olvasót és a kódolást.

Ezután ismét egy if elágazás következik, ahol ellenőrizzük, hogy jó volt a leolvasás. Ha az **a** nagyobb, mint 0, akkor forgassa a szervót 150-es pozícióba és írja ki a soros monitorra, hogy „nyitva”. Majd 3 mp késleltetés után visszaállítjuk az 50-es pozícióba a szervót és kiíratjuk, hogy „zárva” és a végén lenullázzuk az **a** változót. Ezután maradt még két segédfüggvény.  
A „SZERVÓ” nevű egyetlen attribútuma az int típusú datPos változó, amivel megadjuk a szervó pozícióját. A daServo.Write paranccsal kiadjuk a datPos értékét a szervómotornak.

# Képek a kész projektről

