# Általános kommunikációs protokoll

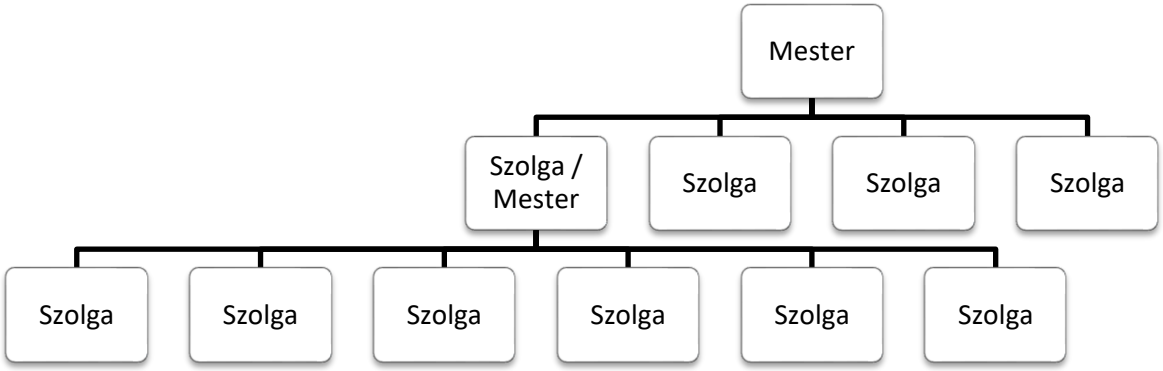
## I. Bevezetés

Jelen dokumentum azt az általános kommunikációs protokollt írja le, amelyet a jelenleg fejlesztett, és a jövőben fejlesztendő eszközeinkben használunk.

A kommunikáció buszos szerkezetű, azaz az egyes eszközök egy azonos buszra vannak felfűzve. A busz fizikai kialakítása elsősorban RS232, azonban lehetséges az ettől való eltérés is (például XBee, RS485, stb).

A protokoll mester-szolga alapú, ahol egy buszon mindig csak egy mester lehet, azonban szolgából több is található. A kommunikáció mindig a mester kezdeményezi, a szolgák csak a mester kérésére válaszolnak.

Igény esetén fa szerkezetű rendszerek építhetőek úgy, hogy egy adott eszköz több buszra is csatlakozik az 1. ábrán is látható módon.




1. ábra

Itt a középső sorban a bal oldalon lévő eszköz az egyik buszon szolga, míg a másikon mester.

A mester a szolgákkal adatcsomagok küldése révén kommunikál. Ezen csomagok leírása a következő fejezetben olvasható.

Egyes rendszereknél előfordulhat, hogy a szolgák alapesetben kikapcsolt állapotban vannak. Ezeknél a rendszereknél a kommunikáció az, amely „felébreszti” a szolgákat, ugyanis a hardveres kialakításuk olyan, hogy az első kommunikáció kapcsolja be a hardvert. Azoknak a mestereknek, amelyek ilyen kialakítású rendszerekben találhatóak, a bekapcsolás után el kell küldeniük egy kezdeti üzenetet, amelynek hatására a szolgák is aktivizálódnak. A szolgák általában rendelkeznek egy időzítővel is, amely az utolsó kommunikáció alkalmával nullázódik. Ha az időzítő elér egy bizonyos értéket, akkor a szolga automatikusan lekapcsolja önmagát. A mesternek tehát azon felül, hogy induláskor bekapcsolja a szolgákat, arra is figyelnie kell, hogy a megfelelő időközönként kommunikáljon a szolgákkal ahhoz, hogy „ébren”, azaz bekapcsolt állapotban tartsa őket. Az időzítés hossza rendszerenként eltérhet.

## II. A csomagok leírása

A kommunikáció adatcsomagok küldésével zajlik. Az adatcsomagok bájtokból állnak. Minden csomag egy startbájttal kezdődik, és egy stopbájttal végződik. A többi bájt ezen keretek között található:

| START | … | STOP |
|---|---|---|

A startbájt és a stopbájt értéke mindig fix, azonban különbözik egymástól attól függően, hogy a csomagot a mester küldi a szolgának, vagy pedig fordítva.

Ha a csomagot a mester küldi a szolgának, akkor a startbájt és a stopbájt értéke:

- [START] = 0x3A
- [STOP] = 0x1B

Ha a csomagot a szolga küldi a mesternek, akkor a startbájt és a stopbájt értéke:

- [START] = 0xDE
- [STOP] = 0xA3

A csomag teljes felépítése a következő:

| START | PID | ID | HEADER | DATA1 | … | DATAn | CHK | STOP |
|---|---|---|---|---|---|---|---|---|

A PID bájt a csomagazonosító, amely bájt azonosítja az adott kommunikációt (csomagváltást). Ez egy véletlenszerű szám, amelyet a mester határoz meg. A szolga az adott kommunikáció alatt a válaszában köteles ugyanezt a PID bájtot használni. Ennek segítségével a mester beazonosíthatja, hogy melyik kérdésére melyik válasz érkezett, több szolga esetén nem keverednek össze a csomagok.

Az ID bájt az eszközazonosító, ez azonosítja a szolgát. Minden egyes szolgának megvan a saját címe, illetve van egy általános cím, amelynek értéke 0x00. Amennyiben egy szolga megkap egy adott csomagot, akkor ellenőrzi, hogy ez az eszközazonosító megegyezik-e a saját azonosítójával, vagy pedig az általános címmel. Amennyiben nem egyezik meg, akkor a szolga eldobja az adatcsomagot, és nem válaszol rá. Amennyiben viszont a kapott csomagban szereplő eszközazonosító megegyezik a sajátjával, akkor a csomagot értelmezi, végrehajtja a benne foglalt parancsot, és ha nem az általános címre érkezett a parancs, akkor válaszol rá. Ez utóbbi feltétel alól az egyetlen kivétel a hívás parancs, amelyre a szolga akkor is válaszolhat, ha az az általános címre érkezett.

A HEADER bájt azonosítja a csomag tartalmát, azaz azt, hogy hány adatbájtot (DATA1 … DATAn) tartalmaz a csomag. A HEADER bájtokat a felső 4 bit alapján a következő csoportokba soroljuk:

- 0x0? - Nullabájtos parancsok.
- 0x1? - Nullabájtos parancsok.
- 0x2? - Egybájtos parancsok.

- 0x3? - Egybájtos parancsok.
- 0x4? - Kétbájtos parancsok.
- 0x5? - Kétbájtos parancsok.
- 0x6? - Négybájtos parancsok.
- 0x7? - Négybájtos parancsok.
- 0x8? - Nyolcbájtos parancsok.
- 0x9? - Nyolcbájtos parancsok.
- 0xA? - Speciális parancsok.
- 0xB? - Speciális parancsok.
- 0xC? - Speciális parancsok.
- 0xD? - Speciális parancsok.
- 0xE? - Speciális parancsok.
- 0xF? - Speciális parancsok.

A nulla-, egy-, kettő-, négy- és nyolcbájtos parancsok mindig, 0, 1, 2, 4, illetve 8 adatbájtot (DATAn) tartalmaznak. A speciális parancsok teljesen egyediek lehetnek, a bájtok számát az adott rendszer parancs-protokollja határozza meg.

Az adatbájtok (DATA1 … DATAn) – amennyiben léteznek – a HEADER bájttal együtt az üzenet lényegét tartalmazzák (beállítandó paraméterek, egyéb lekérdezéshez használt adatok, stb).

Az adatbájtok (DATA1 … DATAn) után egy ellenőrző összeg, a CHK jön. Az ellenőrző összeg egy bájt, melynek számítási módja a következő:

- [CHK] = 0xFF - ([ID] + [HEADER] + [DATA1] + … + [DATAn]) + 1

Amennyiben a számítás során a bájt túlcsordul, akkor a felső (MSB) bájtot el kell dobni, és csak az alsó bájtot (LSB) kell használni.

Az ellenőrző összeget mindig a küldő fél – szolga vagy mester – számítja ki, a fogadó fél pedig a teljes csomag beérkezése után ellenőrzi azt. Az ellenőrzés során a fogadó félnek ki kell számolnia a következő összeget:

- [ID] + [HEADER] + [DATA1] + … + [DATAn] + [CHK]

A számításnak a fentiekkel megegyezően egy bájton kell történnie, azaz ha a bájt túlcsordul, akkor a felső (MSB) bájtot el kell dobni, és csak az alsó bájtot (LSB) kell használni. Amennyiben a fogadó félnél az ellenőrző összeg helyes, azaz a fenti számítás nullát ad eredményül, akkor a kapott csomag érvényes, amennyiben az ellenőrző összeg helytelen, akkor a kapott csomag érvénytelen.

## III. Parancsok

A csomagban küldött parancsokat megkülönböztetjük az alapján, hogy a csomag hány adatbájtot tartalmaz (nullabájtos, egybájtos, stb parancsok).

A parancsok jelentése rendszerenként eltérő lehet, azonban követniük kell a csomagok leírásánál található, a HEADER bájt felső négy bitje szerinti csoportosítást. Ezen felül minden rendszernek meg kell valósítania a következő fejezetekben található parancsokat. Amennyiben az eszköz kialakítása miatt egy parancs nem megvalósítható, akkor az adott parancsot tilos másra használni.

## 1. Nullabájtos parancsok

A nullabájtos parancsok felépítése a következő:

| START | PID | ID | HEADER | CHK | STOP |
|---|---|---|---|---|---|

A parancsot a HEADER bájt határozza meg. A következő pontokban ezeknek a leírása található. A szolga válaszának HEADER bájtja megegyezik a küldött parancs HEADER bájtjával, valamint az ID bájtja megegyezik a szolga eszközazonosítójával. A szolga által küldött adatbájtok száma az adott parancstól függ.

### *a. 0x00 – A szolga hívása*

Ezzel a paranccsal hívható a megadott című szolga. Ez a parancs lényegében annak ellenőrzésére szolgál, hogy az adott szolga elérhető-e a rendszerben. Ez az egyetlen olyan parancs, amelyre a szolga akkor is válaszolhat, ha a parancs az általános címre érkezett. Ez abban az esetben használható ki, ha tudjuk, hogy csak egy szolga van a rendszerre kötve, azonban nem tudjuk ennek a szolgának a címét.

A kapott válasz egy nullabájtos csomag.

A megvalósítás során törekedni kell arra, hogy a szolga minél hamarabb válaszoljon erre a parancsra. Ennek oka az, hogy egyes rendszerekben (például a klímarendszerben) a mester nem tudja, hogy éppen hány szolga kapcsolódik hozzá, ezért a kommunikáció során először mindig ezzel a paranccsal ellenőrzi, hogy az adott szolga elérhető-e.

### *b. 0x01 – A szolga újraindítása*

Ezzel a paranccsal újraindítható a megadott című egység.

A kapott válasz egy egybájtos csomag, amelyben az adatbájt (DATA1) lehetséges értékei, illetve azok jelentése a következő:

- 0x00 - A parancs sikeresen végre lett hajtva (azaz ezen üzenet visszaküldése után a szolga újraindul).
- 0x10 - A parancs nem hajtható végre.

### *c. 0x02 – A szolga típus- illetve működési információjának a lekérdezése*

Ezzel a paranccsal olyan információ kérdezhető le, amely az adott szolga típusáról/működéséről ad felvilágosítást. A kapott bájt egyértelműen meghatározza azt, hogy miként kell az adott szolgával kommunikálni. Ennek a parancsnak az elsődleges funkciója az, hogy a kézi konfigurátor azonosítani tudja az eszközt anélkül, hogy ezt a felhasználónak kéne kiválasztania.

A kapott válasz egy egybájtos csomag, amelyben az adatbájt (DATA1) értéke a következő

- DATA1 - A szolga típusát/működését leíró bájt.

*d. 0x03 – A szolga firmware verziószámának a lekérdezése*

Ezzel a paranccsal lekérdezhető a szolga firmware verziószáma. A verziószám egy négybájtos előjel nélküli egész, amely egy hat számjegyű számot tartalmaz. A hat számjegy jelentése éévhnn (é: év, h: hónap, n: nap). Tehát ha a kapott szám értéke 130204, akkor az azt jelenti, hogy a firmware verziószáma 130204, azaz a 2013. február 4-ei verzió.

A kapott válasz egy négybájtos csomag, amelyben az adatbájtok (DATA1 … DATA4) értéke a következő:

- DATA1 - A verziószámnak a felső 8 bitje (31 … 24).
- DATA2 - A verziószámnak a felső 8 bitje (23 … 16).
- DATA3 - A verziószámnak a felső 8 bitje (15 … 8).
- DATA4 - A verziószámnak az alsó 8 bitje (7 … 0).

**2. Egybájtos parancsok**

Az egybájtos parancsok felépítése a következő:

| START | PID | ID | HEADER | DATA1 | CHK | STOP |
|---|---|---|---|---|---|---|

A parancsot a HEADER bájt határozza meg. A következő pontokban ezeknek a leírása található. A szolga válaszának PID bájtja megegyezik a küldött parancs PID bájtjával, a HEADER bájtja megegyezik a küldött parancs HEADER bájtjával, valamint az ID bájtja megegyezik a szolga eszközazonosítójával. A szolga által küldött adatbájtok száma az adott parancstól függ.

*a. 0x20 – A szolga címének a beállítása*

Ezzel a paranccsal beállítható a szolga címe. A címet egy 8 bites előjel nélküli szám tartalmazza. A szolga a válaszcsomagban mindig a régi címét küldi el.

A küldött csomagban található adatbájt (DATA1) értéke a következő:

- DATA1 - A szolga új címe.

A kapott válasz egy egybájtos csomag, amelyben az adatbájt (DATA1) lehetséges értékei, illetve azok jelentése a következő:

- 0x00 - A parancs sikeresen végre lett hajtva.
- 0x11 - A parancs nem hajtható végre, mert az érték kívül esik a megengedett tartományon (a cím nem lehet 0, mert az az általános cím).

# IV. Adatformátumok

## 1. Standard adatformátumok

Az általános kommunikációs protokollban használt standard adatformátumok, és azok értéktartományai a következőek:

- 1 bájtos előjel nélküli egész: 0 … 255
- 2 bájtos előjel nélküli egész: 0 … 65535
- 4 bájtos előjel nélküli egész: 0 … 4294967295
- 1 bájtos előjeles egész: -128 … 127
- 2 bájtos előjeles egész: -32768 … 32767
- 4 bájtos előjeles egész: -2147483648 … 2147483647

A több-bájtos értékeket mindig „big endian” formátumban kell elküldeni, azaz a legnagyobb helyiértékű bájtot kell először elküldeni, és a legkisebb helyiértékű bájtot utoljára. Egy bájton belül a bitsorrend szintén „big endian”, azaz a legnagyobb helyiértékű bit az első, és a legkisebb helyiértékű bit az utolsó.

Amennyiben tört számot kell elküldeni, akkor törekedni kell arra, hogy az szám egész számként legyen továbbítva úgy, hogy abban fix számú tizedesjegy legyen. Például ha egy két bájtos előjel nélküli egészet küldünk el úgy, hogy az 2 tizedesjegyet tartalmaz, akkor az 1200 jelentése 12.00, a 28 jelentése 0.28, stb.

## 2. Dátum/idő

Egyes esetekben előfordulhat, hogy a mesternek vagy a szolgának el kell küldenie valamilyen dátumot és időt. Ennek elküldése kétféle módon történhet. A kétféle módszer oka az, hogy a kommunikációs protokoll jól illeszkedjen az Universal Logger programhoz. Az egyik – általában preferált – módszernél a dátum/idő BCD (binárisan kódolt decimális, lásd lentebb) formátumban kerül átküldésre, míg a másik – csak szükség esetén alkalmazandó – módszernél pedig binárisan.

Azt, hogy melyik módszert kell használni, az határozza meg, hogy a rendszerünk – ha egyáltalán kommunikál, akkor – milyen módon kommunikál az Universal Logger szoftverrel.

Ha az átküldött dátumot/időt le kell menteni a log fájlokba úgy, hogy a szoftver ezt az átküldött időt használja az aktuális lekérdezés dátuma/idejeként, akkor a dátumot/időt BCD formátumban kell elküldeni. Ez az eset általában akkor áll fenn, ha például az eszköz tartalmaz egy valós idejű órát, amely óra szolgáltatja a dátumot/időt.

Abban az esetben azonban, ha az átküldött dátumot/időt le kell ugyan menteni, de nem szeretnénk, hogy ez legyen az aktuális lekérdezés dátuma/ideje, akkor az értéket bináris formátumban kell elküldeni. Ez az eset akkor áll fenn, ha például az eszköz lement egy korábbi dátumot/időt (például a bekapcsolás dátumát/idejét).

Abban az esetben, ha a dátumot/időt nem kell lementeni a log fájlokba, hanem csak a kezelőfelületen kell megjeleníteni, akkor bármely formátum használható.

Szintén bármely formátum használható, ha a kommunikációban résztvevő mester nem az Universal Logger szoftver.

A két formátum bájtonkénti felépítése azonos, mégpedig a következő: az év 16 bites előjel nélküli egész, a hónap, nap, óra, perc, másodperc pedig 8 bites előjel nélküli egészek. Ezek együtt egy hét bájtos összetartozó adathalmazt alkotnak, a következő sorrendben:

- DATA1 - Az év felső 8 bitje (bit 15 … bit 8).
- DATA2 - Az év alsó 8 bitje (bit 7 … bit 0).
- DATA3 - A hónap.
- DATA4 - A nap.
- DATA5 - Az óra.
- DATA6 - A perc.
- DATA7 - A másodperc.

A BCD formátum esetén 4 bit tárol egy számjegyet, míg bináris formátum esetén a számok ábrázolása a szokásos módon történik. Az alábbi példa azt mutatja, hogy a 2014. augusztus 28. 10:14:37 hogyan néz ki BCD, illetve bináris formában. A bal oldali oszlop mutatja a BCD formátumot, a jobb oldali oszlop pedig a bináris formátumot.

| BCD formátum | Bájt sorszáma | Bináris formátum |
|---|---|---|
| 0x20 | DATA1 | 0x07 |
| 0x14 | DATA2 | 0xDE |
| 0x08 | DATA3 | 0x08 |
| 0x28 | DATA4 | 0x1C |
| 0x10 | DATA5 | 0x0A |
| 0x14 | DATA6 | 0x0E |
| 0x37 | DATA7 | 0x25 |

## V. A standard válasz

A kommunikáció során a beállító parancsok zömére a szolga egy egybájtos csomaggal válaszol. Ezt az egybájtos csomagot standard válasznak nevezzük. A standard válasz felépítése a következő:

| START | PID | ID | HEADER | DATA1 | CHK | STOP |
|---|---|---|---|---|---|---|

A szolga válaszának PID bájtja megegyezik a küldött parancs PID bájtjával, a HEADER bájtja megegyezik a küldött parancs HEADER bájtjával, valamint az ID bájtja megegyezik a szolga eszközazonosítójával.

A DATA1 bájt a szolga válasza a beállító parancsra. Amennyiben a beállító parancs sikeres volt, akkor a bájtnak 0x00-nak kell lennie. Minden ettől eltérő bájt valamilyen hibát jelez. Ezzel a módszerrel jelezni lehet a mester felé, hogy a parancs végrehajtása miért nem sikerült. A jelenleg létező DATA1 bájtokat a következő táblázat foglalja össze:

| **DATA1 bájt** | **Jelentés** |
|---|---|
| 0x00 | A parancs sikeresen végre lett hajtva. |
| 0x10 | A parancs nem hajtható végre. |
| 0x11 | A parancs nem hajtható végre, mert az érték kívül esik a megengedett tartományon. |