Fok-Gyem protokoll ismertetése

Ezek a kijelzők RS485-ös soros kommunikációt használnak az adatok fogadására, tápfeszültségük, igazodva az alkalmazási környezethez 24V. A világítással rendelkező kijelzőkben a fénycső táplálása szintén 24 voltos.

A kommunikáció beállításai:

Baud rate (szimbólumsebesség): 19200
Parítás: szóköz
adatbitek: 8
stopbitek: 2
Minden kijelzőnek van egy címe, amit a kijelző vezérlőpanelén található DIP kapcsolókról lehet leolvasni.

A DIP kapcsolók kiosztása:

1	2	3	4	5	6	7	8
Címzés					Orientáció	Önteszt 1	Önteszt 2

1-5ig a címzést lehet beállítani, a felkapcsolt kapcsolókat bináris 1-nek, a lekapcsolatakat bináris 0-nak értelmezve. Ez a kijelző vezérlőn futó szoftververzióktól függően lehet invertálva is.

Az orientációval a kijelző modulokon a kép 90°-al elforgatható.

A DIP kapcsolók beállítása után kijelzőt újra kell indítani, vagy a vezérlőpanelen lévő reset gombot meg kell nyomni.

Az önteszt kapcsolókat felkapcsolva a kijelző bekapcsolás vagy reset után egy tesztprogramot fog futtatni, amivel könnyen kideríthetőek a hibás pixelek pontos helye, illetve a mechanikusan szoruló pixelek fellazíthatók. Egyes típusokon létezik önteszt is, mely során a beépített EEPROM memóriából kiolvasott képeket jelenít meg egymás után a kijelző.

Az adatstruktúra felépítése:

Fontos megérteni, hogy a kijelzők csak az ASCII tábla karaktereit tudják értelmezni. Tehát ha bármilyen karaktert küldünk a kijelző felé, akkor nem egyszerűen az adott karaktert küldjük ki soros porton, hanem a karakter ASCII tábla szerinti hexadecimális értékét. Az adatstruktúra két féle lehet, ebből az egyszerűbbik az, amikor nem küldünk ki képet, csak például a kijelző törlésére, vagy az önteszt elindítására utasítjuk a kijelzőt.

Ebben az esetben a felépítés a következő:

STX	ADDRESS	COMMAND	CHECKSUM	ETX

STX – start of text, az ASCII tábla szerinti 0x02. Minden üzenet ezzel kezd, egy byte hosszú.

ADDRESS – kijelző címzése, kiszámítási módját későbbiekben részletezem, két byte hosszú

COMMAND – Következők lehetnek:

Clear	Kijelzőkép törlése	0x30, 0x30, 0x30, 0x32, 0x30, 0x42
Self test	„hardveres” önteszt	0x30, 0x30, 0x30, 0x34, 0x31, 0x46, 0x30, 0x31
Picture test	kép teszt	0x30, 0x30, 0x30, 0x34, 0x31, 0x46, 0x30, 0x32
Image upload start	képfeltöltés kezdete	0x30, 0x30, 0x30, 0x32, 0x30, 0x43
Image upload end	képfeltöltés vége	0x30, 0x30, 0x30, 0x32, 0x31, 0x34

CHECKSUM – ellenőrző kód, a kiküldött üzenet alapján számolva, 2 byte hosszú

ETX – end of text, ASCII tábla szerinti 0x03, minden üzenet ezzel zárul, egy byte hosszú.

A kijelzőcímzés

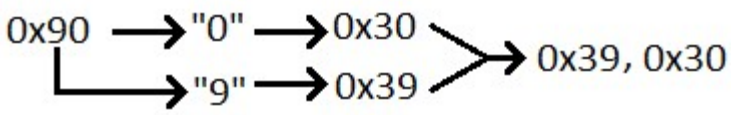
A kijelzőcímzés kiszámítása trükkös. Mint korábban írtam, a kijelző csak az ASCII tábla szerint gondolkodik, csak ezt ismeri. Ha a kijelző címe, a DIP kapcsolók alapján 4-es (általában belső, utastéri kijelző) akkor, ha csak simán a 4-et, mint egy byte (0x04) küldenénk ki, akkor az az ASCII tábla szerinti EOT, end of transmission lenne. Ez a kijelző számára értelmezhetetlen, emiatt a protokoll előírja a kijelző címek átszámítását.

A példában vett kijelzőnk fizikai címéhez (4) hozzá kell adni 32-t, majd a kapott számot meg kell szorozni 4-el.

(Fizikai cím + 32) * 4 = számított cím

(4 + 32) * 4 = 144 (decimális) = 0x90(hexadecimális)

Mivel ez az érték (90) két karakter hosszú, így ennek kiküldésére két btye-ra lesz szükség. A protokoll ezt a hexadecimális számot felbontja a két karakterre, és kikeresi a karakterekhez tartozó ASCII tábla szerinti hexadecimális értéket. Előbbi példánkkal élve, így a címzés két byte-ja a következők lesznek: 0x39, 0x30



Létezik egy közös címzés, nevezhetnénk „broadcast” címnek, melyre az RS485-ös buszon lévő összes kijelző reagál, ez pedig a 31-es fizikai cím:

(31+32) * 4 = 252(decimális) = 0xFC(hex) -> 0x46, 0x43 cím bájtok

A checksum (ellenőrzőösszeg) számítása

Az adatátviteli hibák kiküszöbölése érdekében, minden üzenetben, annak lezárása előtt a protokoll kiküld egy két byte hosszú ellenőrző összeget.

A cheksum-ba a STX és ETX byte-ok nem számítanak bele, csak a kijelző címe, és a parancs. Amikor kijelző képet küldünk akkor a címzés, illetve a blokk száma, és a blokkhoz tartozó képinformáció bytejai is beleszámítanak.

Az ellenőrző összeg számítása a következő:

Azokat a byteokat, amik a beleszámítanak az ellenőrzőösszegbe, összeadjuk és ezt az értéket 256-tal csökkentjük. Ha az így kapott (hexadecimális) szám hosszabb, mint két karakter, akkor a szám utolsó két karakterét (két legkisebb helyiérték) vesszük csak figyelembe. Ezeket a karaktereket a címzésnél bemutatott módon felbontjuk az ASCII tábla szerinti megfelelő értékre. Az így kapott két byte az ellenőrzőösszeg, melyet az üzenet végéhez, az ETX – end of text byte elé illesztünk be.

Példa:

31-es címre küldjünk egy törlés parancsot, így a teljes üzenet:

0x02, **0x46, 0x43, 0x30, 0x30, 0x30, 0x32, 0x30, 0x42,** [chksum 1. byte], [chksum 2. byte], 0x03

A checksum számításába csak a vastaggal jelölt byte-ok számítanak bele. Ezeket a számokat először összeadjuk, összegük hexadecimálisan 0x1BD, ezt csökkenteni kell decimális 256-al (hexa 0x100), így a checksum összege 0xBD. Ezt a két karakter hosszú számot felbontjuk az ASCII tábla alapján, így a checksum bytejai: 0x42, 0x44

Így a teljes üzenet cheksummal:

0x02, 0x46, 0x43, 0x30, 0x30, 0x30, 0x32, 0x30, 0x42, **0x42, 0x44**, 0x03

STX, --ADDRESS--, ---------- COMMAND ----------, **-Checksum-**, ETX

A kijelzőkép felépítése

A kijelzőkép felépítése a protokoll legbonyolultabb része. A kijelzőkép feltöltése során az adatstruktúra felépítése a következő:

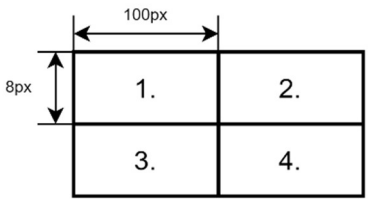
STX	ADDRESS	BLOCK ID	BLOCK LENGTH	IMAGE DATA	CHECKSUM	ETX

Az STX és az ADDRESS a korábban leírtaknak megfelel ez esetben is. Ezt követi a BLOCK ID és a BLOCK LENGTH adatok. Mindegyik adat két-két byte hosszú.

Blokkokra bontás

Ha a kijelző mérete (felbontása) túl nagy, akkor a kijelzőképet blokkokra kell bontani. A blokkokra bontást a következő esetekben kell megtenni:

1. Ha a kijelző vízszintes felbontása nagyobb, mint 96 képpont
2. Ha a kijelző függőleges felbontása nagyobb, mint 8 képpont



A blokkok elhelyezése táblázatos, az első blokk mindig bal fent, az utolsó mindig jobb lent helyezkedik el.

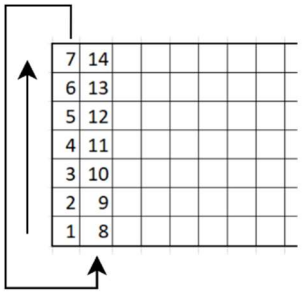
A blokkok számát a kijelzőkép kiküldése során meg kell jelölni. Az adatfolyamban az adott blokk száma a kijelzőcím mögött helyezkedik el. Itt nincs semmilyen átszámítás, az 1-es blokk értelemszerűen hexában 0x01, ennek az ASCII tábla szerinti felbontása a korábban ismertetettek alapján 0x30, 0x31, és így tovább a többi blokkal. A blokk száma után annak hosszát (horizontális felbontását) is meg kell jelölni, itt viszont a valós értéket **szorozni kell kettővel**, majd az ismert módon az ASCII átváltást elvégezni. A blokkok szélessége bármekkora lehet, de érdemes 96 alatt tartani, célszerűen érdemes a kijelző teljes horizontális felbontásának felénél meghúzni a két blokk közti határt.

Egyes esetekben, ha a kijelző több blokkot várna, tehát pl 16 pixel magas kijelzőre küldünk adatot, tehát a kijelzőképet két blokkra kéne osztani, akkor az egyik blokk elhagyható. A képfeltöltés kezdete után, az első blokk képinformációja után, a második blokk helyett egyből küldhető a képfelöltés vége üzenet, és a kijelző ez esetben is meg fogja jeleníteni a képet, és csak a kiküldött blokk helyén fog frissülni a megjelenített tartalom. FIGYELEM, ez hivatalosan nem támogatott funkció, melynek használata nem javasolt, egyes esetekben a kijelző lefagyhat.

Régebbi típusú, ún. „volános” kijelzők esetében, a teljes kép elrendezése más. A kijelző oszlopainak első 32 tagja kerül kiküldésre legkésőbb. Az első oszlop, amely elsőnek kiküldésre kerül, az a teljes kijelzőkép 33. oszlopa, innen halad a kép felépítése a kijelző legutolsó (jobb szélső) oszlopáig, majd visszaugrik és folytatja a kijelző első oszlopával.

A pixelmátrix felépítése

A minden blokkon belül a kijelzőkép első képpontja a bal alsó sarokban lesz. Innen felfele haladva épül fel az első oszlop. Ha a blokkon belüli első oszlop felépítése kész, akkor az oszlop számát egyel növelve, a második oszlop alsó képpontjával folytatódik a kép.



Ez így megy míg a blokk végére nem értünk, utána a blokk számát egyel növelve folytatódik a teljes kijelzőkép felépítése. A kijelzőkép adatfolyama hordozza a pixelek állását jelölő bájtokat. Ez a teljes protokoll leghosszabb része, mert minden esetben az összes képpont állását ki kell küldeni.

Mivel minden képpontnak csak két állása lehet, így kézre eső minden pixel állását egy bittel jelezni. Ha egy adott képpontot, a világos oldalára akarunk állítani, akkor az a protokoll szerint egy bináris egyesnek felel meg, ha a sötétre, akkor bináris 0. Az így kialakult bináris számot, az adott oszlop aljáról kezdve, felfele olvassuk ki.

Itt fontos megjegyezni, hogy a 7 pixel magas kijelzőknél, az alsó bájt csonka maradna. Ezt elkerülendő, a protokoll a legalsó alatti, valóságban nem létező, „fantom” pixelt mindig 0-nak értelmezi. Ezeket az ábrán szürkével jelöltem.

Az ábrán látható példával tehát, ha lentről felfele olvassuk ki a két bájtot, MSB (most significant bit) értelmézésben, a 8-as helyiérték lesz az első pixel, ami a képet felépíti, akkor az alsó bájt decimális értéke 7, az hexában szintén 0x07, az ASCII megfelelője pedig 0x37.

Helyiérték					
1	0	0	1	0	0
2	0	1	0	1	0
4	1	0	0	0	1
8	1	0	0	0	1
1	1	1	1	1	1
2	1	0	0	0	1
4	1	0	0	0	1
8	0	0	0	0	0
	↓	↓	↓	↓	↓
DEC	12	2	1	2	12
	7	1	1	1	7
Hex	C	2	1	2	C
	7	1	1	1	7
ASCII	0x43	0x32	0x31	0x32	0x43
	0x37	0x31	0x31	0x31	0x37

A felső bájté pedig decimális 12, tehát hexában 0x0C. A „C” karakter ASCII megfelelője pedig a 0x43-as hexadecimális szám.

A konkrét képinfromációt hordozó üzenetnek nincs külön parancsa, csak a szokásos STX, a kijelző címe átszámítva az ismertetett módon, majd a blokk száma és hossza, majd maga a képinformációt hordozó bájtok, majd checksum és ETX.

Ellenben, minden képfeltöltés kezdetét és végét külön paranccsal kell jelezni a kijelző felé. Ezeket a parancsokat egyenként, valamint minden képtartalmat hordozó üzenetet, ugyanúgy el kell látni STX és ETX bájtokkal, ellenőrzőösszegel és címzéssel.

Képfeltölés kezdete	0x30, 0x30, 0x30, 0x32, 0x30, 0x43
Képinformáció (blokk(ok))	lásd fent
Képfeltöltés vége	0x30, 0x30, 0x30, 0x32, 0x31, 0x34

Egy kép felöltése tehát legalább 3 különálló üzenet a kijelző felé, amennyiben csak egy blokkból áll a kijelző. Ha több blokkot is tartalmaz, a blokkokat ugyan különálló üzenetekre kell osztani, de közöttük nem kell ismételten más parancsot küldeni. Egyes üzenetek között legalább 0.2-0.3 másodperc szünetet illik tartani, ellenkező esetben a kijelző lefagyhat.

Szín	jelentés
Zöld	STX vagy ETX byte
világoskék	Címzés
piros	Blokk száma
sötétkék	Blokk hossza
citromsárga	Parancs
fehér	Képtartalom
rózsaszín	Ellenőrzőösszeg

Példaképként, egy 7x96-os (belső, utastéri) kijelzőre, 31-es („broadcast”) címmel, kiírjuk hogy FOK-GYEM: (a táblázatban szereplő számok mind hexadecimálisan értelmezendők, az üzenetek teljes értékűek)

képfeltöltés kezdete	02 46 43 30 30 30 32 30 43 42 45 03
Képadat	02 46 43 30 31 43 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 37 46 30 39 30 39 30 39 30 33 30 30 33 45 34 31 34 31 34 31 33 45 30 30 37 46 30 38 31 34 32 32 34 31 30 30 30 38 30 38 30 38 30 38 30 30 33 45 34 31 34 31 34 39 33 41 30 30 30 37 30 38 37 30 30 38 30 37 30 30 37 46 34 39 34 39 34 39 36 33 30 30 37 46 30 32 30 34 30 32 37 46 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 32 31 03
képfeltöltés vége	02 46 43 30 30 30 32 31 34 42 30 03

További példák:

Törlés a 31-es címre: 02 46 43 30 30 30 32 30 42 42 44 03