Arduino, mint PLC

„Kicsit sárgább, kicsit savanyúbb, de a mienk!”

Bacsó Péter

V 3.xx


ARDUINO, MINT PLC
RIASZT1
RIASZT2
RIASZT3
RIASZT4
BEKAPCSOLVA
NYUGTÁZÁS
KONFIGURÁLÁS
AkiNET KFT.

1. PLC alapfunkciók, kommunikáció

AZ „Arduino, mint PLC” az alábbi feladatok végrehajtására lett kifejlesztve: analóg és digitális bemenő jelek fogadása és feldolgozása, riasztások generálása, kommunikáció a megjelenítő egységekkel és a további feldolgozó – pl. bemenő adatokat tároló, grafikonokat rajzoló, eseményeket naplózó stb. – automata gépekkel.

A berendezés 8 db 4-20 mA-es analóg bemenettel, 16 digitális, kontaktus bemenettel és 4 relés, digitális kimenetel rendelkezik.

A 8 analóg bemenet a géptermekben elhelyezett hőmérők jelét fogadja, a 16 kontaktus bemenet általános célra használható – pl. klíma hibajel, UPS állapotjel, tűzjelző, középfeszültségű betápláló trafó megszakító állásjel stb. –, a 4 digitális kimenet pedig riasztások (duda, lámpa, átjelzés) kiadására alkalmas.

Az Arduino alapú PLC-nek nincs semmilyen kijelzője, legfeljebb néhány kigyulladó LED jelzi, hogy kap tápfeszültséget, s valamilyen programot hajt végre. Az Arduinon a PLC szoftver felett egy webszervert megvalósító program fut, így az adatokat Ethernet szabványt használva helyi hálózaton (LAN-on) továbbítja HTML formában, amelyet egy szokásos böngészővel[1] lehet nyomon követni. A módszer egyik előnye, hogy egy szabványos, mindenki által használt kliens elegendő a kommunikációhoz, a másik – ami veszélyeket is hordoz magában – pedig a távoli elérés lehetősége.

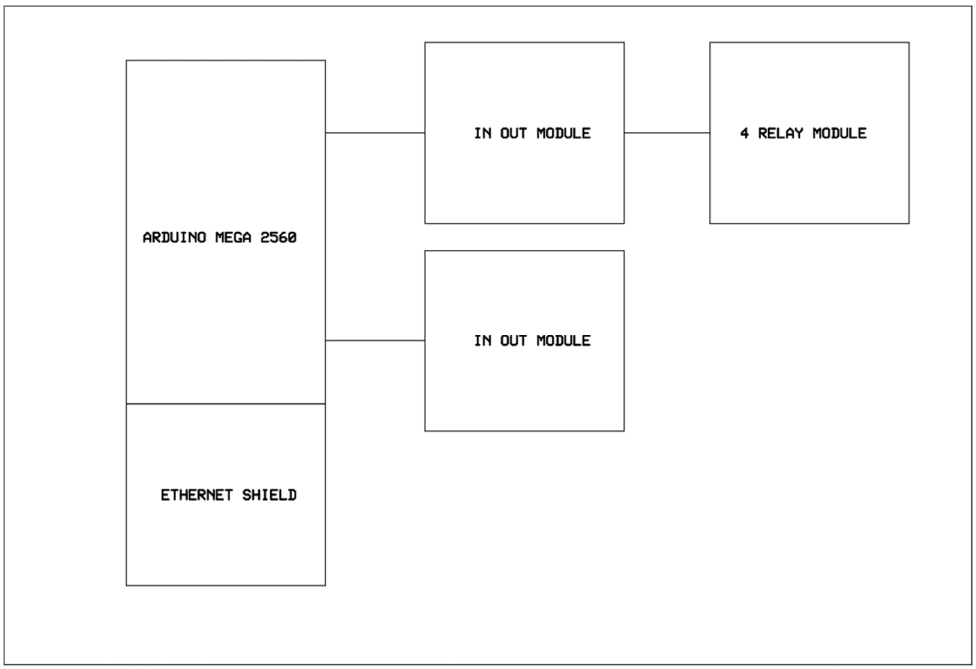
A szerver - kliens között kétirányú a kommunikáció, azaz nemcsak megjeleníteni lehet a pillanatnyi állapotokat, hanem a távoli beavatkozás, konfigurálás – pl. a riasztási hőmérséklet megváltoztatása – is lehetséges. Hogy a rosszindulatú, vagy téves módosításokat elkerüljük, ezek a beavatkozások csak az „Arduino, mint PLC” doboz előlapján elhelyezett piros nyomógomb megfelelő állásában engedélyezett (ezt az állapotot a nyomógomb világítás bekapcsolása és a képernyő háttérszín megváltozása jelzi), így egy távoli módosítás csak ekkor lehetséges. A fenti módszer előnye az, hogy nem csak helyben, hanem távolról, akár otthonról is kényelmesen nyomon követhető az alapinfrastruktúra, a kliens általi változtatások

[1] Az Arduino program a Google Crome, a Mozilla Firefox, a Microsoft Edge, az Apple Safari, az Opera cég Opera és a Brave cég Brave böngészőjével lett tesztelve.

érvényre jutása azonban csak a helyben, az „Arduino, mint PLC”-n lévő nyomógomb megfelelő állásában hatásos.

2. Rendszerfelépítés

Az „Arduino, mint PLC” több NYÁK összekötésével lett kialakítva (1. ábra). A rendszer legfontosabb eleme egy Arduino Mega 2560 alaplap Ethernet Shielddel kiegészítve (ez lehet az alaplapra integrálva, de lehet piggyback modul is), ehhez kell csatlakoztatni egy, vagy két „In out module” NYÁK-ot, valamint egy reléket tartalmazó kis „4 Relay Module” NYÁK-ot. A rendszer egy 24VDC/1A külső tápegységről működik, aminek a kimenetét sorkapcsokon fogadja. A NYÁK-ok különböző tápfeszültségeit belső DC/DC konverterek biztosítják.



1. ábra

Minden bemenet és kimenet sorkapcsokra van kivezetve és megfelelően feliratozva a könnyű bekötés érdekében.
A LAN összeköttetést egy szabványos RJ-45 8P8C csatlakozó biztosítja.
A dobozban található még egy átkötés – JPA jumper –, ami a fent említett paraméterállítási funkciókat engedélyezi, vagy tiltja. Az átkötés az előlapra egy piros nyomógombra van kivezetve, de – pár méteren belül – tetszés szerinti helyre kikábelezhető.[2]

3. Működés

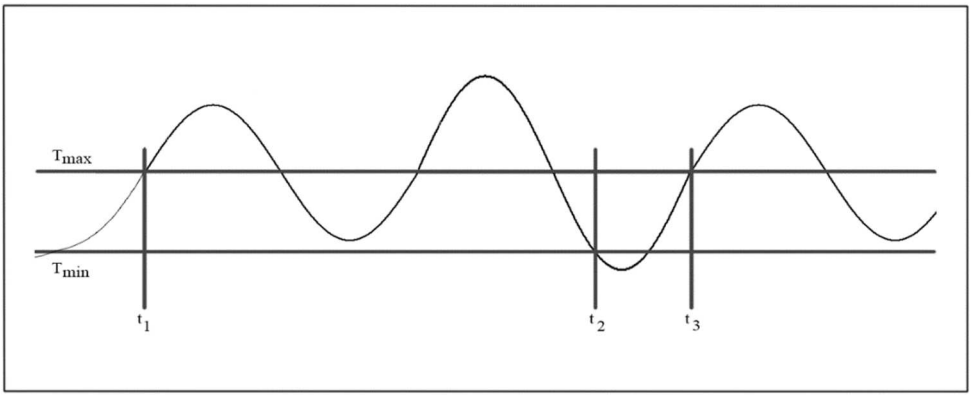
Az Arduino program egy PLC alapú géptermi felügyeletet emulál, analóg és digitális bemenetek függvényében kimeneteket aktivál (riaszt). A rendszernek nincs dedikált, helyi kijelzője, a felügyelt paraméterek megjelenítéséhez szabványos web szervert valósít meg és HTML segítségével táblázatos formában írja le az adatokat, így egy általános böngészővel a géptermi infrastruktúra paraméterei távolról leolvashatók, a kritikus értékek (riasztások) piros színnel kiemelve láthatók. Amennyiben a képernyőn nagyon kis méretben jelennek meg az adatok, akkor az a szokásos módszerekkel (pl. CTRL+, vagy CTRL egérgörgetés fel) nagyítható, vagy ha az egészet egy képernyőn egyben akarjuk áttekinteni, akkor (pl. CTRL-, vagy CTRL egérgörgetés le) kicsinyíthető.
A megjelenített adatok háromféle helyen tárolódhatnak: FLASH, EEPROM, vagy SRAM memória. A paraméterek egy része FLASH-ben van, ez azt jelenti, hogy módosítani csak a forrásprogramban lehet, amit aztán újra kell fordítani és letölteni, hogy érvényre jusson. RESET után, újrainduláskor mindig ezek az értékek fognak betöltődni (pl. táblázatok fejlécei).
Az EEPROM-ban tárolt értékeket változtatni csak a piros nyomógomb (JPA jumper) megfelelő állapotában lehet, azonnal – ont the fly – érvényre jut, s bootoláskor ez, a legutoljára beírt érték jelenik meg. (Pl. az első táblázatban a riasztási

[2] A nagyobb biztonság érdekében védett (pl. kulcsos) kapcsolóra is kivezethető.

hőmérsékletek megadása, második táblázatban a bemenetek invertálása, és engedélyezése.)

A paraméterek harmadik része csak SRAM-ban tárolódik, ezek a rendszerint pillanatnyi értékek nem maradnak meg, újrainduláskor más, aktuális állapotot vesznek fel (pl. harmadik táblázat adott riasztási kimenet állapota).
A PLC **program** 8 analóg és 16 digitális bemenet, valamint 4 digitális kimenet kezelésére alkalmas, azaz 8 gépterem felügyeletét tudja kényelmesen ellátni, az eredményeket pedig 3 táblázatba foglalva megjeleníteni egy weboldalon. (Lásd melléklet!)

Az első rész az „1. Hőmérséklet érzékelők" táblázat. Ennek első sorában az adott gépterem megnevezése szerepel (átírni csak a forrásprogramban lehet), a második sorában „Pillanatnyi érték", ezt a hőmérők adataiból számítja ki, s jeleníti meg a program. Ha ez eléri a harmadik „Riasztási érték" sor értékét, akkor pirosra vált – így emelve ki a riasztás okát – és riasztást generál. A táblázat harmadik sorában „Riasztási érték" az az érték, ami elérésekor az Arduino a riasztás kimeneteket aktivizálja. Ennek a sornak a módosítása csak a piros nyomógomb (JPA jumper) megfelelő állásában lehetséges. (A módosítás konkrét lépéseit a „Konfigurálás" fejezetben tárgyaljuk.) Ha átírjuk, akkor az azonnal – ont he fly – az EEPROM-ba tárolódik le, s bootolás esetén már ez az új érték fog megjelenni. Ha egy hőmérő (egy gépterem) állapota nem lényeges, akkor a riasztási értékét állítsuk maximumra (99 ^{o}C), így soha nem fog riasztást generálni (kivéve, ha valóban eléri a 99 Celsius fokot, ami tűz keletkezésére utal, vagy valami hardver hiba lépett fel).
A kényelmesebb üzemvitel érdekében a hőmérsékletfigyelésnél fix, 2 ^{o}C-os hiszterézis (a 2. ábrán a hiszterézis = T_{max} - T_{min}) van beállítva, ezért az ábrán látható esetben riasztás a t_1 időpontban keletkezik, s csak a t_2 időpontban szűnik meg,



2. ábra

majd a t_3 időpontban új riasztás generálódik. A böngésző ablak 1. táblázatában a „Riasztási érték” a T_{max}-nak felel meg.

A második rész a „2. Üzemállapot bemenetek” táblázat a 16 digitális bemenet beállításait és aktuális értékeit mutatja. Az első sor az adott bemenet megnevezése (módosítani csak a forrásprogramban lehet). A második „Pillanatnyi állapot” sor azt jelzi, hogy az adott bemenet riasztási állapotban van-e. A harmadik „Riasztás” sor azt jelzi, hogy az adott bemenet generálhat-e riasztást. Ha igen, akkor ténylegesen is aktívak lesznek a riasztás kimenetek, ha nem, akkor az adott bemenet állapota lényegtelen, soha nem vált ki riasztást. (Egy nem használt bemenetet pl. így lehet inaktívvá tenni.) Ennek a sornak a módosítása csak piros nyomógomb (JPA jumper) megfelelő állapotában lehet, s ez az érték azonnal beíródik az EEPROM dedikált helyére. (A módosítás konkrét lépéseit a „Konfigurálás” fejezetben tárgyaljuk.) A negyedik „Invertálás” sorban az adott bemeneti szintet tudjuk változtatni, azaz beállíthatjuk, hogy a magas (ponált, ebben az esetben a szakadás vált ki riasztást), vagy alacsony (negált, ebben az esetben a rövidzár vált ki riasztást) szint jelentsen-e riasztást.

A harmadik rész a „3. Riasztás kimenetek” táblázat a 4 digitális kimenet állapotát mutatja. Az első sor az adott kimenet megnevezése (módosítani csak a forrásprogramban lehet). A második „Pillanatnyi állapot” sor az adott kimenet aktuális állapotát tükrözi. Ez összegzi az eddigi riasztásokat, azaz ha bármelyik bemenet riasztása aktív (piros), akkor ez a sor is aktív, piros lesz. A harmadik

„Riasztás” sor a kimenetek nyugtázására, inaktívvá tételére szolgál. (A módosítás konkrét lépéseit a „Konfigurálás” fejezetben tárgyaljuk.)
Három megjegyzést kell tenni:
1. A böngésző a táblázat adatait az AJAX ajánlásnak[3] megfelelően frissíti kb. 1**0 másodpercenként**. Ez azt jelenti, hogy adott esetben maximum ennyit várni kell, hogy a böngésző ablakában egy cella értéke megváltozzon (ez nem azt jelenti, hogy ilyen lassú az Arduino, hanem azt, hogy csak egy frissítés után jelenik meg az új érték, függetlenül attól, hogy pl. a kimenet hardveresen már ezelőtt megváltozott). Ezt meggyorsíthatjuk, ha a böngészőnek egy frissítést (refresh) nyomunk.
2. Ha a kimeneteket nyugtázással inaktívvá tesszük, akkor azok természetesen elnémulnak egészen addig, amíg egy új hiba fel nem lép. Ha új hiba keletkezik, akkor hiába voltak nyugtázva a kimenetek, újra aktívak lesznek (s pl. újra dudálni fog), így hívva fel a figyelmet egy új eseményre. **Ez nagyon fontos tulajdonság, hogy ne maradhasson rejtve egy-egy újabb hiba megjelenése.**
3. A „3. Riasztás kimenetek” táblázatban a (piros) riasztás **csak akkor** szűnik meg, ha a nagy „gombbal” (cellával) nyugtáztuk. A riasztást kiváltó ok a felső két táblázat valamelyikében szintén pirossal kiemelve látható. Ha ez az ok még fennáll a nyugtázáskor, akkor a nyugtázást követően is piros marad, ha már nem áll fenn, akkor a nyugtázással együtt alapszínre (sárgára) vált. Ezekből az következik, hogy egy riasztást kiváltó ok hiába szűnik meg a nyugtázás előtt, addig látható marad (pirossal kiemelt cella), amíg a nagy „gombbal” (cellával) le nem nyugtáztuk.
Normál (nincs riasztás) esetben egyik táblázatban sincs pirossal kiemelt cella, ami így igen gyors ellenőrzést tesz lehetővé.
A mellékletben három tipikus képernyőképet mutatunk a fentiek demonstrálására.
Ugyanígy az egyes géptermi helyiségek, digitális bemenetek megnevezése is a forrásprogramba van fixen „beégetve”, a fenti okok miatt.

[3] Az AJAX egy módszer arra, hogy ha egy adott weboldalon csak kevés adat változik, akkor azt gyorsan, törésmentesen jelenítse meg a böngésző. Bővebben lásd az AJAX ajánlásokban!

Az állítható paraméterek futás közben, – on the fly – fordítás és letöltés nélkül, azonnal az EEPROM-ba tárolódnak. Amit átírunk, megváltoztatunk az automatikusan beíródik az EEPROM megfelelő helyére, s újraindításkor innen, az EEPROM-ból fogja kiolvasni és megjeleníteni a program.
Az Arduino program forráskódja elektronikus formában rendelkezésre áll.

4. Konfigurálás

A paraméterek egyik részét (konkrétan mindhárom táblázat első sorát, a fejlécét és a hálózati parmétereket) csak a forrásfájlban lehet módosítani, ami után újra kell fordítani és letölteni a tárgykódot, hogy a módosítások érvényre jussanak. (A fordításról és letöltésről a „Fejlesztőknek" fejezetben írunk részletesebben.)
A paraméterek másik részét futás közben, – on the fly – fordítás nélkül tudjuk módosítani a böngészőablakban, de ehhez a piros nyomógombnak (JPA jumpernek) megfelelő állásban kell lennie (ezt az állapotot jelzi a nyomógomb világítás bekapcsolása és a képernyő háttérszín megváltozása). Ebbe a csoportba tartozik mindhárom táblázat harmadik sora, valamint a második táblázat negyedik sora.

Most vegyük sorra konkrétan a változtatások lépéseit táblázatonként.
Az „1. Hőmérséklet érzékelők" táblázat harmadik sorát – a riasztási hőmérséklet megadását – úgy lehet átírni, hogy az adott cella fölé pozícionáljuk az egeret, s itt kattintunk egyet az egér bal gombjával. (Legyünk türelemmel a böngésző miatt, vagy nyomjunk a böngészőnek egy „refresh-t".) Ekkor egy kis ablakocska ugrik fel, ennek a fehér mezőjébe kell beírni az új riasztási hőmérsékletet °C-ban, – ennek 10 és 99 közé kell esnie – majd megnyomni a „Küld" gombot. Ezt az új értéket a program beírja az EEPROM megfelelő helyére, s ez az új érték fog megjelenni legközelebb a táblázat adott cellájában (de itt is legyünk türelemmel).
A „2. Üzemállapot bemenetek" táblázat harmadik sorát – riasztás engedélyezve, vagy tiltva – úgy kell megváltoztatni, hogy a kiválasztott cella fölé pozícionáljuk az egeret, és itt kattintunk egyet az egér bal gombjával. Ezzel a mostani érték az ellenkezőjére vált (az eredményt a böngésző lassúsága miatt időkéséssel, vagy

„refresh” gyorsítással látjuk majd), s ez az új érték tárolódik le az EEPROM-ba. Mivel ez egy egybites, bináris adat – vagy engedve van a riasztás, vagy tiltva – ahányszor a cella fölött kattintunk az egér bal gombjával annyiszor fog az érték az ellenkezőjére váltani. E táblázat negyedik sora teljesen hasonlóan működik, csak itt a bemenet invertálását – ponált, vagy negált legyen – tudjuk megadni.
A „3. Riasztás kimenet” táblázatban csak a harmadik sor nagy (összevont) celláját tudjuk módosítani, ezt is csak az egyik irányba, azaz ha egyszer nyugtáztuk a hibát, akkor azt többé már nem tudjuk visszavonni. A nyugtázáshoz – szokásos módon – a nagy cella fölött kattintani az egér bal gombjával, a kimenetek inaktívak lesznek (türelmet kérünk itt is). Ha a kimeneteket nyugtáztuk, de egy új hiba lép fel, akkor a kimenetek ismét aktívvá válnak, s csak egy újabb nyugtázás teszi ezeket inaktívvá. Ha egy riasztás keletkezik, de az még a nyugtázás előtt megszűnik, akkor – az ok megszűnésének ellenére – a cellája piros (riaszt) állapotban marad, jelezve a riasztást kiváltó okot.

5. Műszaki adatok

Tápfeszültség	
Megengedett tápfeszültség tartomány:	24 VDC +/- 10%
Maximális teljesítményfelvétel:	15 W
Bekötés:	sorkapocs
Maximális vezetékkeresztmetszet:	1,5 mm^2

Digitális kimenetek	
Kimeneti relé ohmos terhelhetősége:	10A / 240 VAC, 28 VDC
Kimeneti relé induktív terhelhetősége:	5A / 120 VAC, 28 VDC
Maximális terhelő teljesítmény:	1200 VA / 300 W
Maximális átmeneti ellenállás zárt állapotban:	100 mΩ
Minimális szigetelési ellenállás:	100 MΩ

Bekötés:	sorkapocs
Maximális vezetékkeresztmetszet:	1,5 mm^2
Digitális bemenetek	
Alacsony szint (rövidzár):	< 1 Ω
Magas szint (szakadás):	> 1 MΩ
Bekötés:	sorkapocs
Maximális vezetékkeresztmetszet:	1,5 mm^2
Analóg (4-20 mA) bemenetek	
Bemenet 0 C^o:	4 mA
Bemenet 50 C^o:	20 mA
Bemenő ellenállás:	330 Ω
Maximális kimenő feszültség:	26,4 VDC
Bekötés:	sorkapocs
Maximális vezetékkeresztmetszet:	1,5 mm^2
LAN csatlakozás:	RJ-45, 8P8C
Környezeti feltételek	
Működési hőmérséklet tartomány:	-40 – +85 C^o
Működési relatív páratartalom tartomány:	10–90% (nem lecsapódó)
Tárolási hőmérséklet tartomány:	-65 – +150 C^o
Ipari védettség:	IP 40
Kivitel:	fémdoboz
Méretek	
Hosszúság:	270 mm
Szélesség:	200 mm
Magasság:	85 mm
Tömeg:	1,4 kg

6. Tartalomjegyzék

CSATLAKOZÁSOK

8 db analóg (4-20 mA-es) bemenet sorkapocs jelölések:

AI1+, AI1-, AI2+, AI2-, AI3+, AI3-, AI4+, AI4-,
AI5+, AI5-, AI6+, AI6-, AI7+, AI7-, AI8+, AI8-

16 db digitális (kontaktus) bemenet sorkapocs jelölések:

DI1+, DI1-, DI2+, DI2-, DI3+, DI3-, DI4+, DI4-,
DI5+, DI5-, DI6+, DI6-, DI7+, DI7-, DI8+, DI8-,
DI9+, DI9-, DI10+, DI10-, DI11, DI11-, DI12+, DI12-,
DI13+, DI13-, DI14+, DI14-, DI15+, DI15-, DI16+, DI16

4 db digitális (morse relé) kimenet sorkapocs jelölések:

DO1: NC, COM, NO; DO2: NC, COM, NO;
DO3: NC, COM, NO; DO4: NC, COM, NO

1 db átkötés:

JPA jumper a módosítások érvényre jutásának engedélyezése

1 db Ethernet csatlakozás:

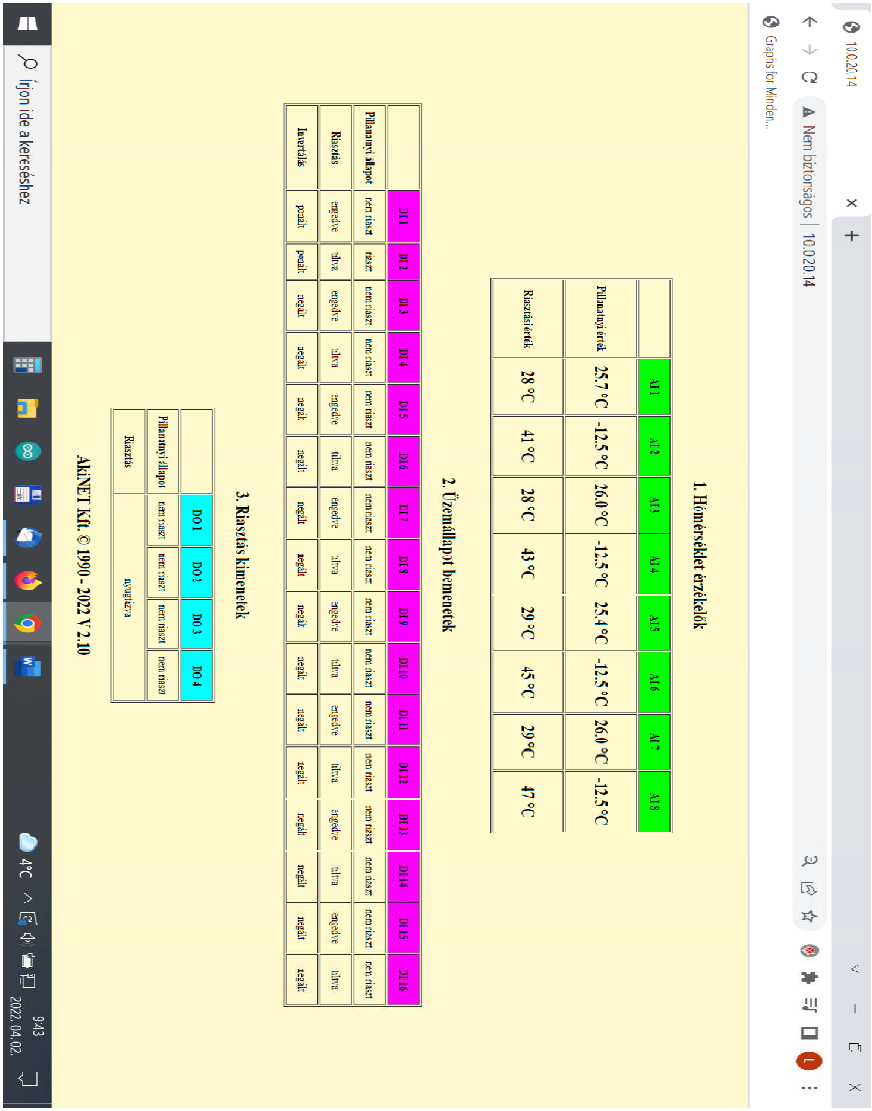
szabványos, egyenes kiosztású RJ-45 8P8C csatlakozó

2 db 24VDC táp csatlakozó sorkapocs jelölések:

24V , GND, 24V, GND

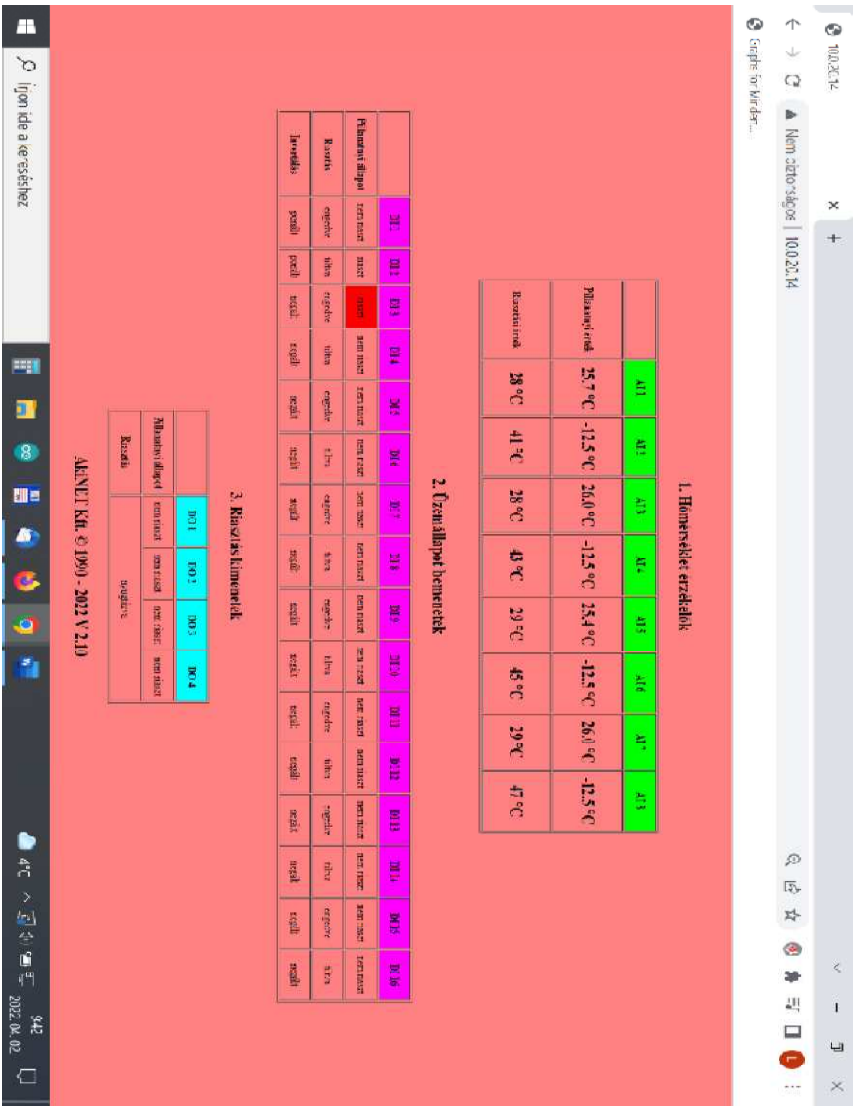
KÉPERNYŐKÉPEK

A következő oldalakon három tipikus képernyőképet láthatunk, valamint a csatlakozók elhelyezkedését.



Normál (hibamentes) képernyőkép

Egy tipikus riasztás képernyőképe



A hibát még mutató, de már nyugtázott képernyőkép
A megváltozott színű háttér a JPA „engedve” állapota miatt van