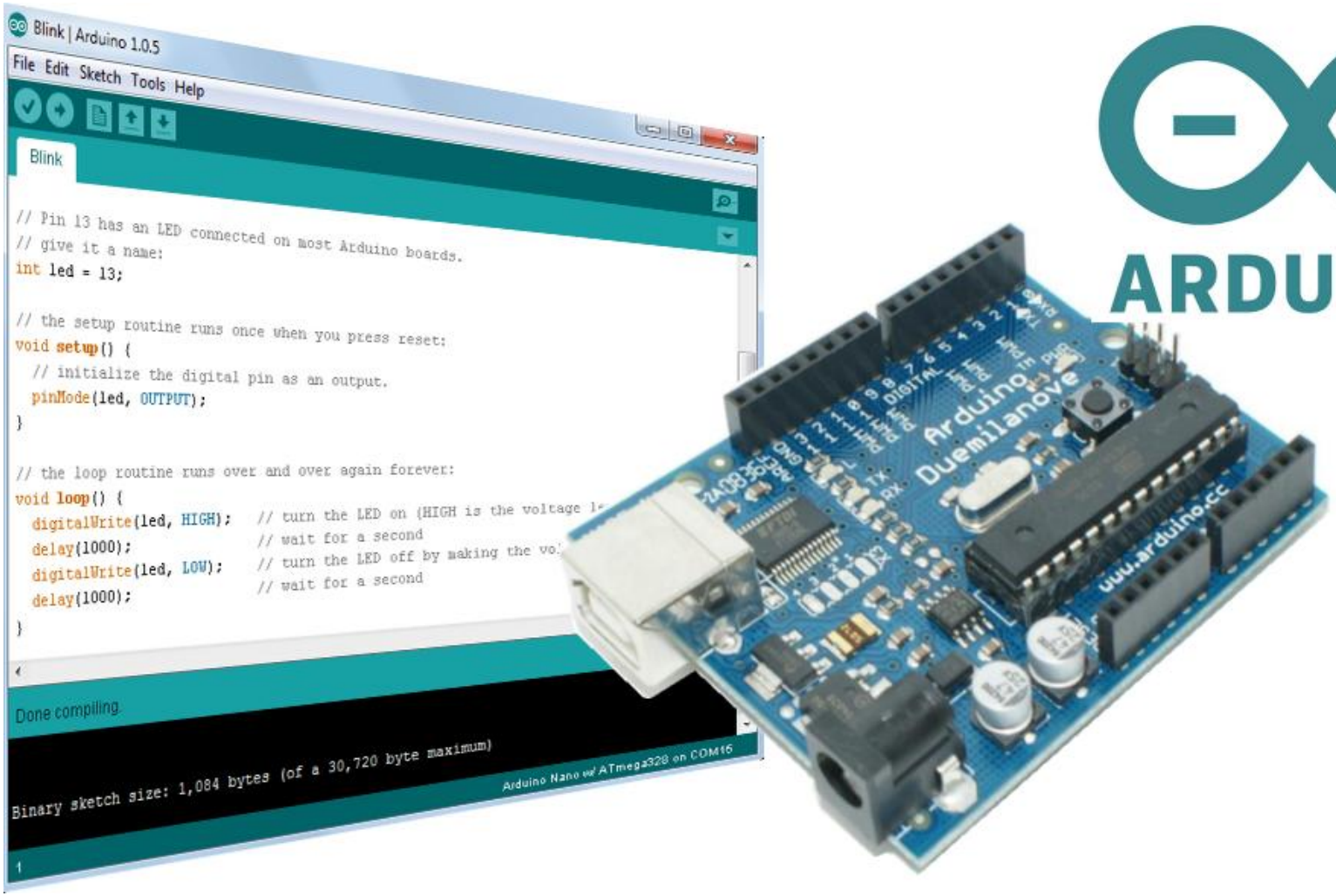


Bevezetés a mikrovezérlők programozásába:

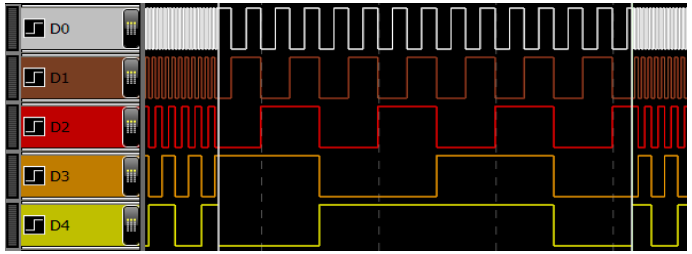
Az Arduino, mint logikai analizátor

Hasznos eszközök hibakereséshez

Logikai áramkörök

- Logikai teszter

- Logikai analizátor



- Speciális protokoll vizsgáló eszközök
  - SPI
  - I2C
  - MODBUS

Analóg áramkörök

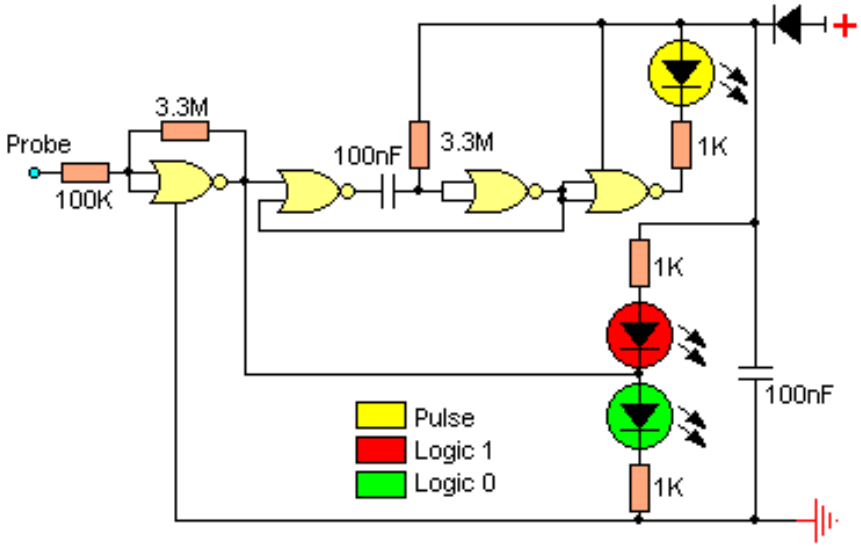
- Voltmérő

- Oszcilloszkóp

- Alkatrészvizsgáló (R, L, C, Dióda, Tranzisztor, …)

Logikai teszter (logic probe)

Általában alacsony/magas és rövid impulzusok jelzésére. Bonyolultabb esetben impulzus injektálás is lehetséges, esetleg számláló vagy frekvenciamérési lehetőség.



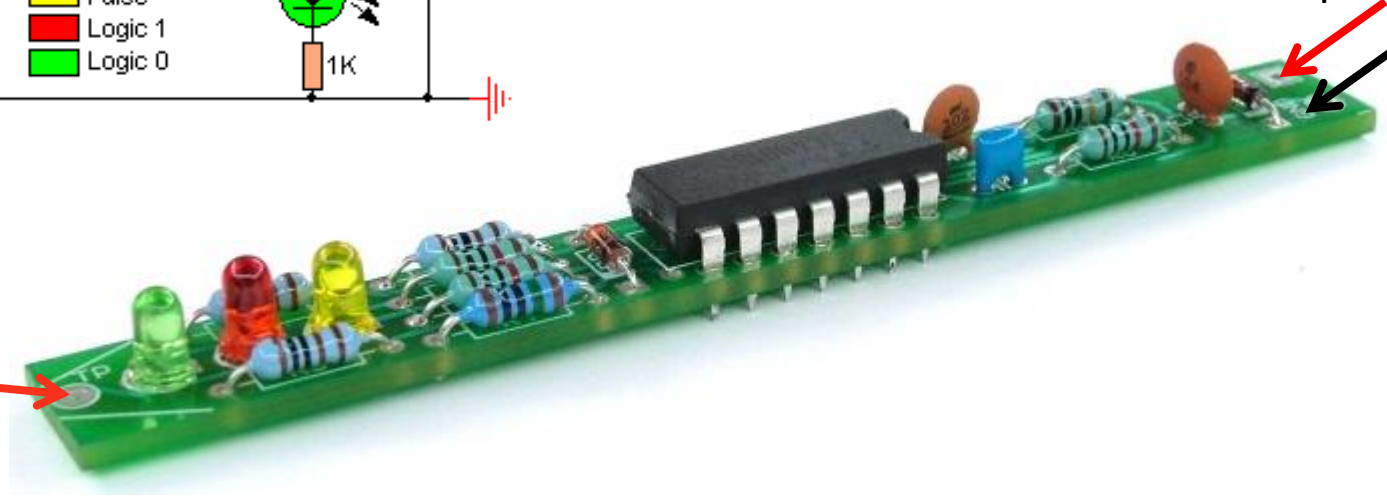
Parts:

MC4001	x1
LED	x3
100nF Capacitor	x2
3.3M Resistor	x2
100K Resistor	x1
1K Resistor	x3
1N914 Diode	x1

Az egyszerű változat kapcsolási rajza és megépítése

Tápellátás

Tapintócsúcs helye

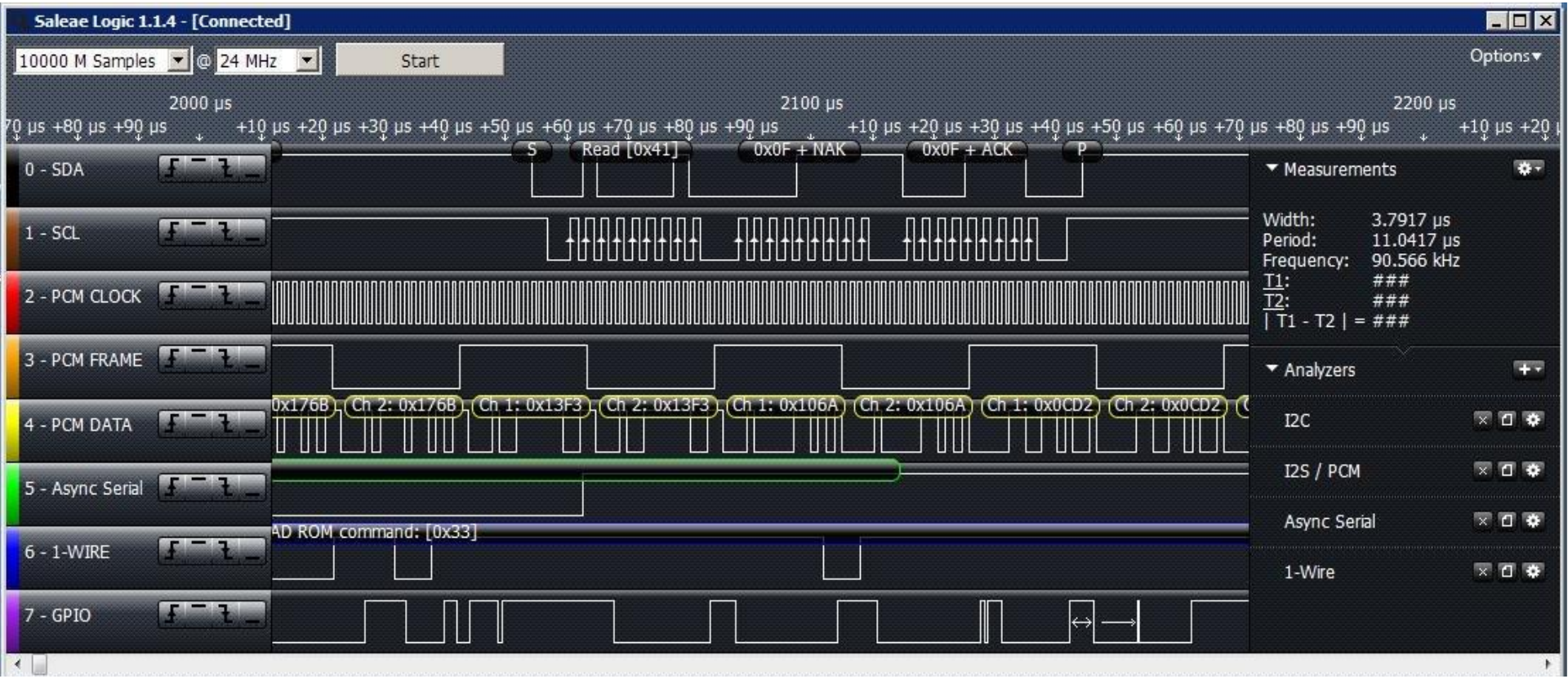


Logikai analizátor

A logikai analizátor szerepe:

- egyidejűleg több logikai kimenet állapotát vizsgálja
- a jelek időbeli lefutását rögzítse és mutassa
- biztosítson különféle triggerelési lehetőséget (azt lássuk, amire kíváncsiak vagyunk)
- Működjön közre a jelalak-minta értelmezésében (1-wire, UART, SPI, I2C, CAN, DMX, JTAG stb. protokoll dekódolása)

Kár, hogy olyan drága!

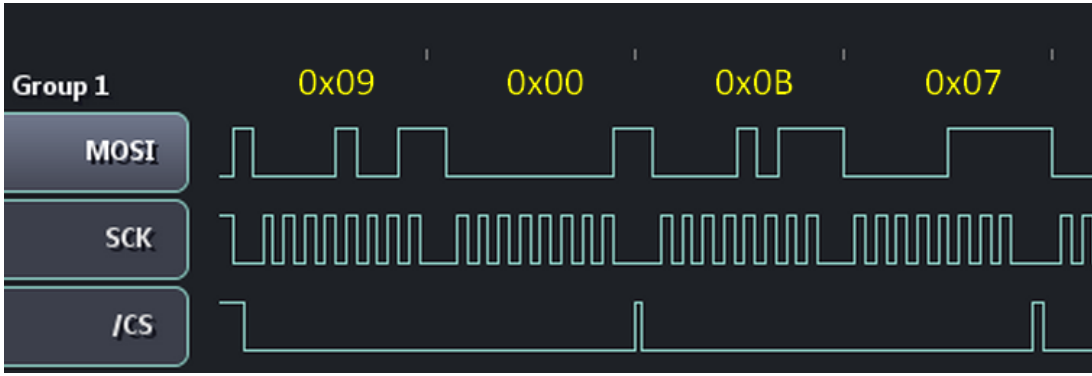


Logikai analizátor – a legolcsóbban

Mi kell egy olcsó logikai analizátorhoz?

- **Arduino nano v3.0** (ha 3,3 V-os rendszert akarunk vizsgálni, akkor jobban járunk egy Meduino nano-val, ami átkapcsolható 3,3 V-ra!)
- **Firmware:** Andrew Gillham „Arduino Generic Logic Analyzer” (AGLA)
  forráskód: https://github.com/gillham/logic_analyzer
  Rövid leírás: https://hackaday.io/project/1633-arduino-generic-logic-analyzer
- **PC szoftver:** J.W. Janssen. *„Open Bench Logic Sniffer Java client”*

  honlap: *https://www.lxtreme.nl/ols/*
  FAQ: *https://github.com/jawi/ols/wiki*
  forráskód: *https://github.com/jawi/ols*



Olcsó húsnak …

Az olcsóság hátránya a szerényebb paraméterekben jelentkezik:

- Korlátozott csatornaszám (5 vagy 6 db)
- Korlátozott mintavételi idő (**AT328:** 1024, **AT2560:** 7168 mintavétel)
- Korlátozott mintavételezési frekvencia (4 MHz-től – 10 Hz-ig)
- Korlátozott triggerelési lehetőség:
  - 4 és 2 MHz-en nincs triggerelés
  - 1 MHz csak „busy wait loop”, párhuzamos triggerjelre várás van
  - 1 – 500 kHz mintavételezésnél párhuzamos és soros triggerelés
- Adatátvitel soros porton (115200 bps)

Fentiek tükrében az *AGLA* eszköz főleg az alábbi jelek vizsgálatára ajánlható:

- UART
- I2C
- kis sebességű SPI kommunikáció
- PWM jelek
- 1-wire (DHT22 is)
- DMX

A firmware fontos részletei

A firmware három részből áll, ugyanis a 4 MHz és a 2 MHz-es mintavételezés nem szervezhető ciklusba, ezért kifejtve, egy-egy külön állományban helyezkednek el.

A bemenetek választhatóan:

- **a D port felső 6 bitje** (az Arduino kártya D2, D3,D4, D5, D6 kivezetései) – ehhez a program elején aktiválni kell a `#define USE_PORTD 1` sort.
- **a B port alsó 6 bitje** (az Arduino kártya D8, D9, D10, D11, D12, D13 kivezetései) – ez az alapértelmezett.
- A LED-hez kötött **D13** kivezetés bemenetként történő használata letiltható (ehhez kommentbe kell zárni a `#define CHAN5 13` sort), ez esetben azonban csak 5 db. csatornánk lesz.

A triggerelés nem működik megfelelően, ha a bemenetek lebegnek. Ezért a szabad bemeneteket külső ellenállással vagy belső felhúzással határozott állapotba kell állítani!
Az **Arduino_Lab27.zip**-ben mi belső felhúzást állítottunk be.

4 Mhz mintavétel

```
logicdata[i] = CHANPIN;
asm("NOP ");
```

2 MHz mintavétel

```
logicdata[i] = CHANPIN;
asm("     NOP
          RJMP 1f
1:        RJMP 2f
2: ");
```

1 MHz mintavétel

```
for (i = 0 ; i < readCount; i++) {
    logicdata[i] = CHANPIN;
    … és még 9 db NOP
}
```

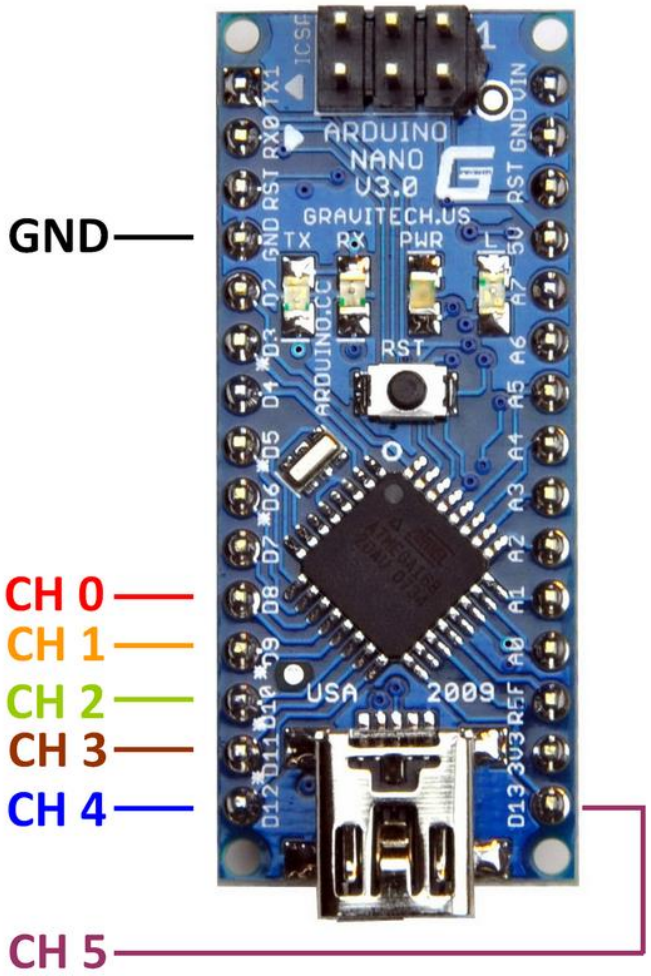
A konfigurációs állomány (részletek)

A konfigurációs állomány az **OLS kliens** program **„plugins”** könyvtárába kell bemásolni, s szükség esetén módosítható.

```
# Configuration for Arduino Generic Logic Analyzer profile
device.type = AGLA
device.interface = SERIAL
device.clockspeed = 16000000                    mintavételi frekvenciák
device.supports_ddr = false
device.samplerates = 10, 20, 50, 100, 200, 500, 1000, 2000, 5000,
10000, 20000, 50000, 100000, 200000, 500000, 1000000, 2000000,
4000000
device.captureclock = INTERNAL
device.capturesizes = 64, 128, 256, 512, 1024   mintavételek száma
device.feature.noisefilter = false
device.feature.rle = true
device.feature.testmode = false
device.feature.triggers = true
device.trigger.stages = 1
device.trigger.complex = false
device.channel.count = 6
device.samples.reverseOrder = true     adatok fordított sorrendben
```

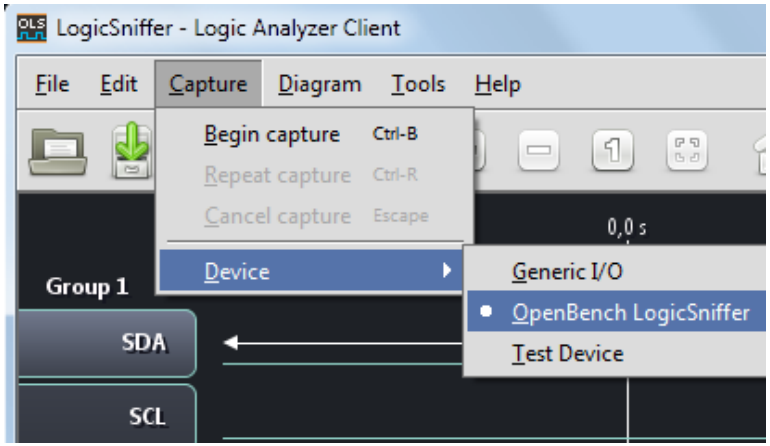
Huzalozási vázlat

- A tápfeszültség közös pontját (**GND**) is össze kell kötni a vizsgált rendszerrel.
- Ha a firmware-t nem módosítottuk, akkor alapértelmezetten a **D8 – D13** kivezetések lesznek a mérőcsatorna bemenetek.
- Ügyeljünk a jelszintek egyezésére! (3,3 V-os jelet nem érzékelnek az **Arduino** Schmitt-triggeres bemenetei)



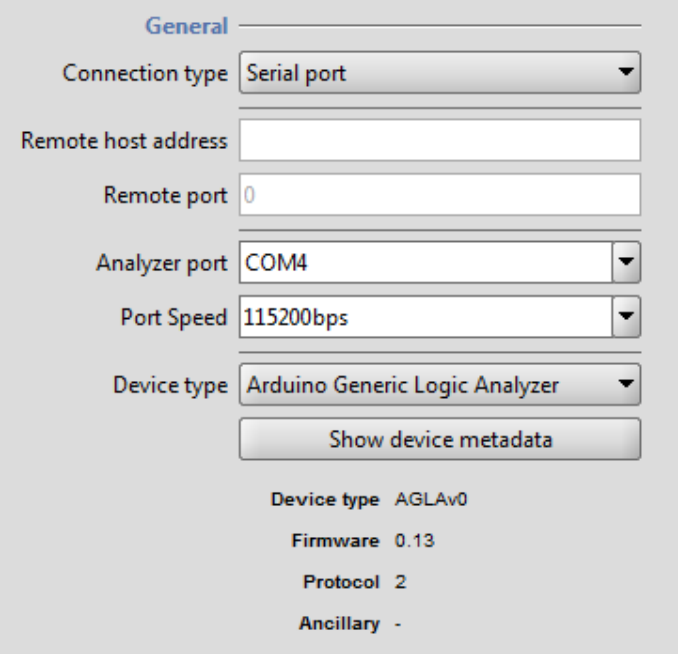
A program használata

1. Indítás a **run.bat** állomány futtatásával.



2. Eszköz kiválasztása a **Capture/Device** menüben (az **OpenBenchLogicSniffer** eszközt válasszuk!)

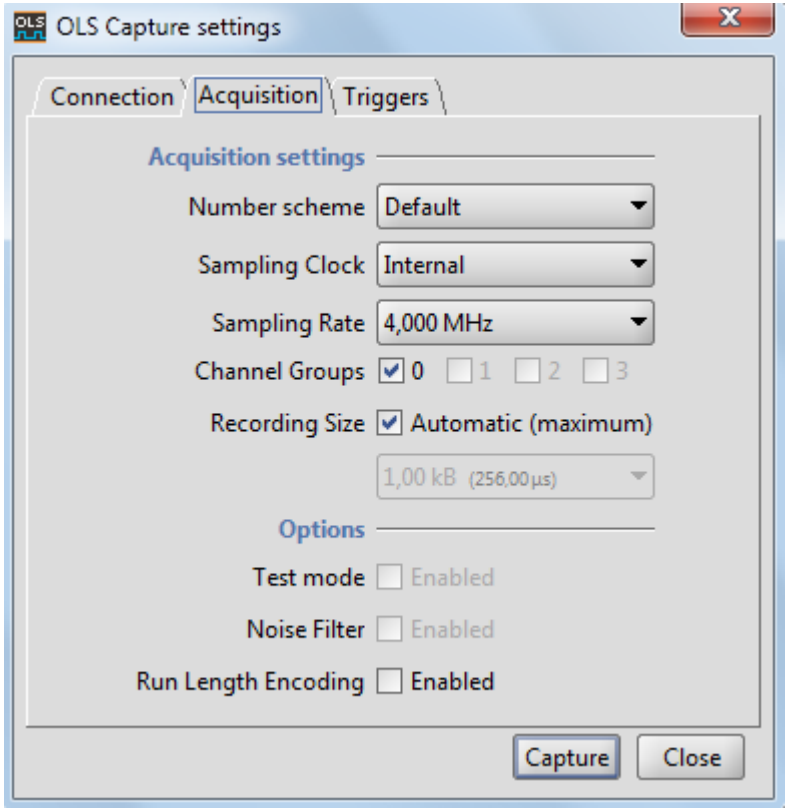
3. A **Capture/Begin** capture menüben állítsuk be:
   - Kapcsolat típusa: **Serial port**
   - Eszköz: az Arduino soros portja, pl. **COM4**
   - Adatsebesség: **115200 bps**
   - Device type: **Arduino Generic Logic Analyzer**



Majd kattintsunk a **Show device metadata** gombra!

Adatgyűjtési beállítások

4. Az **Acquisition** fülre kattintva állítsuk be a kívánt mintavételezési arányt és a buffer méretét!



Megjegyzések:

- Vegyük figyelembe, hogy „normális” triggerelés csak 1 MHz-nél kisebb mintavételi frekvenciák esetén lesz!
- Bár elvileg lehetséges, de különösebben nem érdemes a mintavételezés hosszát csökkenteni.
- A **Run Length Encoding** engedélyezésekor felbukkanó figyelmeztetés bennünket nem érint, Arduino Uno/Nano esetén nem lesz emiatt csatornavesztés!

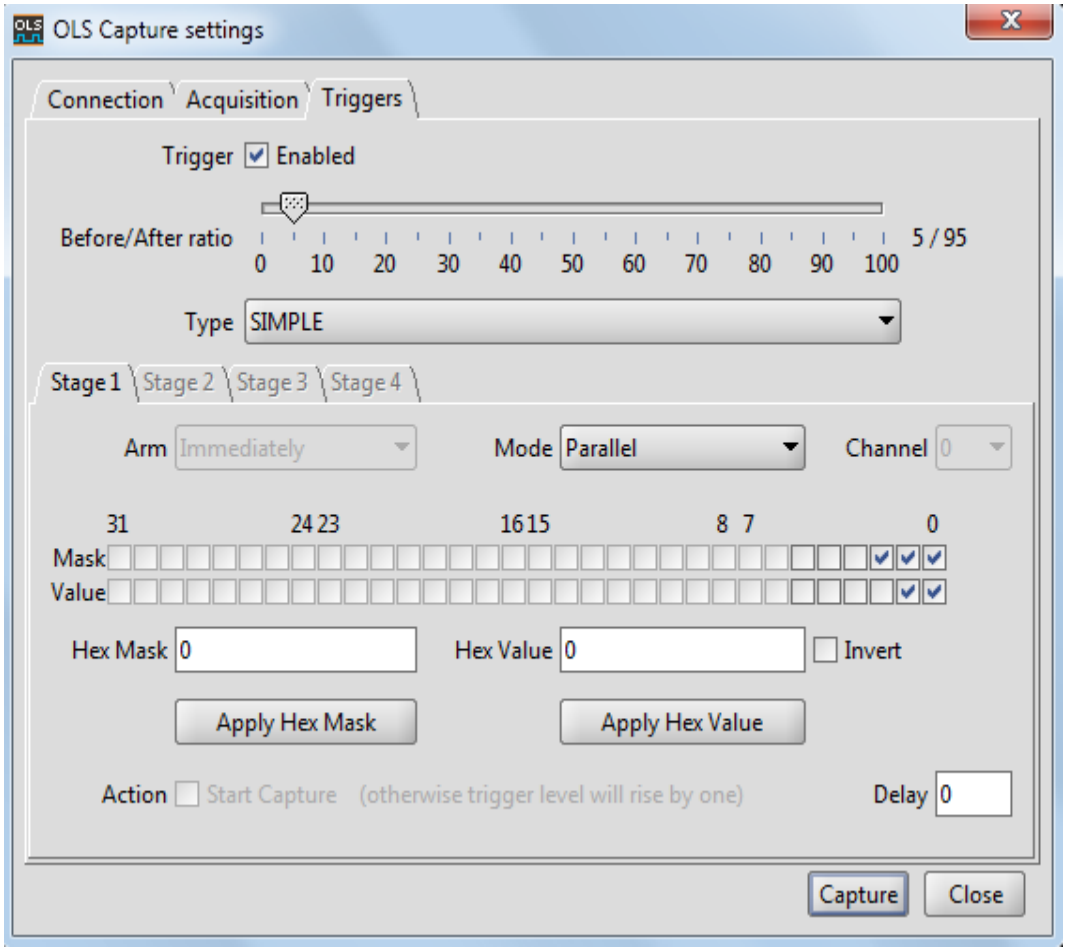
Triggerelési beállítások

5. Kattintsunk a **Triggers** fülre és állítsuk ba a triggerelés feltételeit!

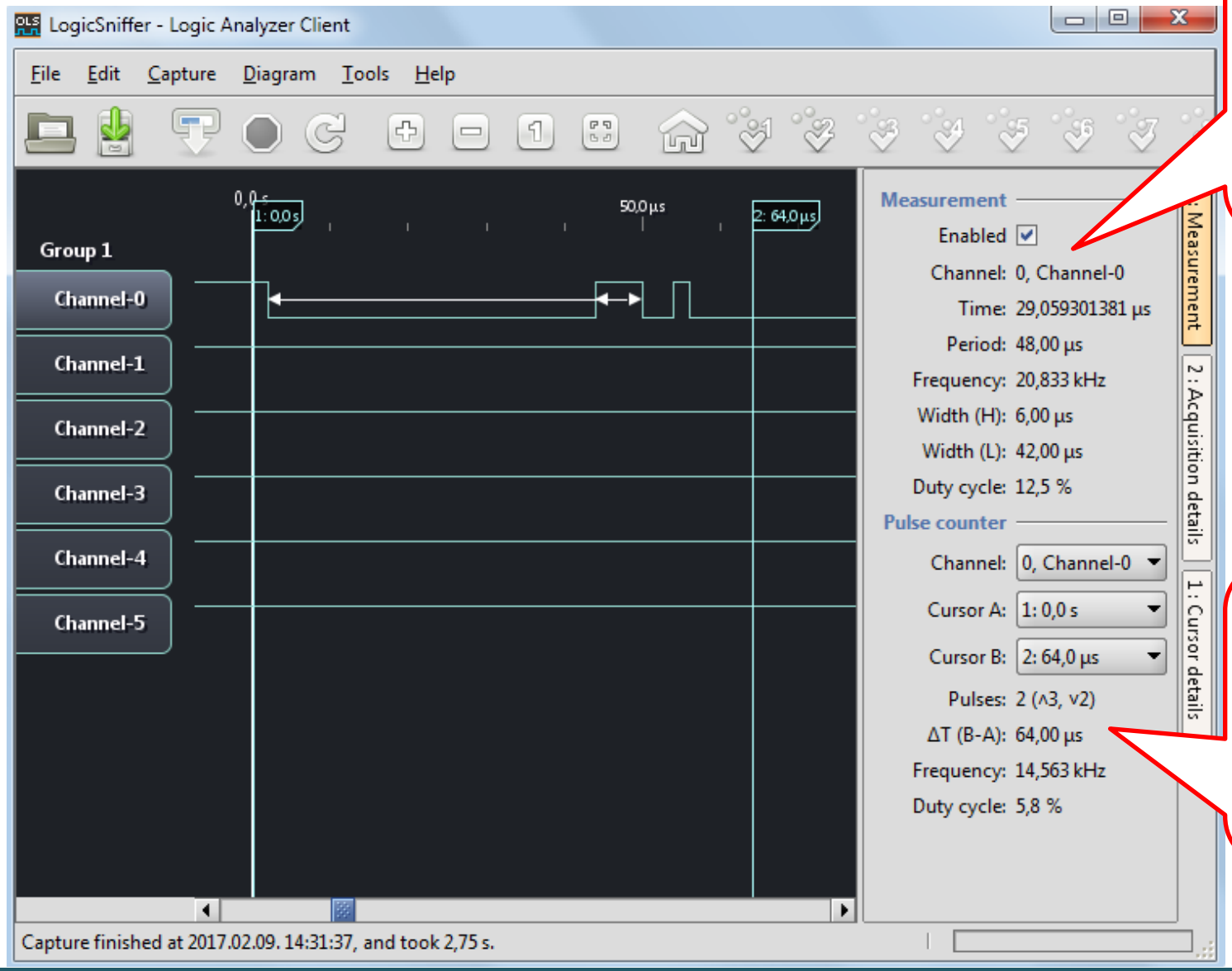
Trigger Enabled: engedélyezés. Enélkül a **Capture** gombra kattintva a mintavételezés azonnal indul.
Before/After ratio: beállítható, hogy a triggerfeltétel teljesülése előtti időszakaszból mennyit mutasson meg. A számok a bufferméret százalékában értendők.

Parallel mód: A bejelölt csatornák megadott értéke (Value) a triggerfeltétel.

Serial mód: a kiválasztott csatorna megadott bitmintája a triggerfeltétel.



Mérési lehetőségek



Impulzus-szélesség, vagy periódus-idő mérése a nyilakkal jelölt tartományban

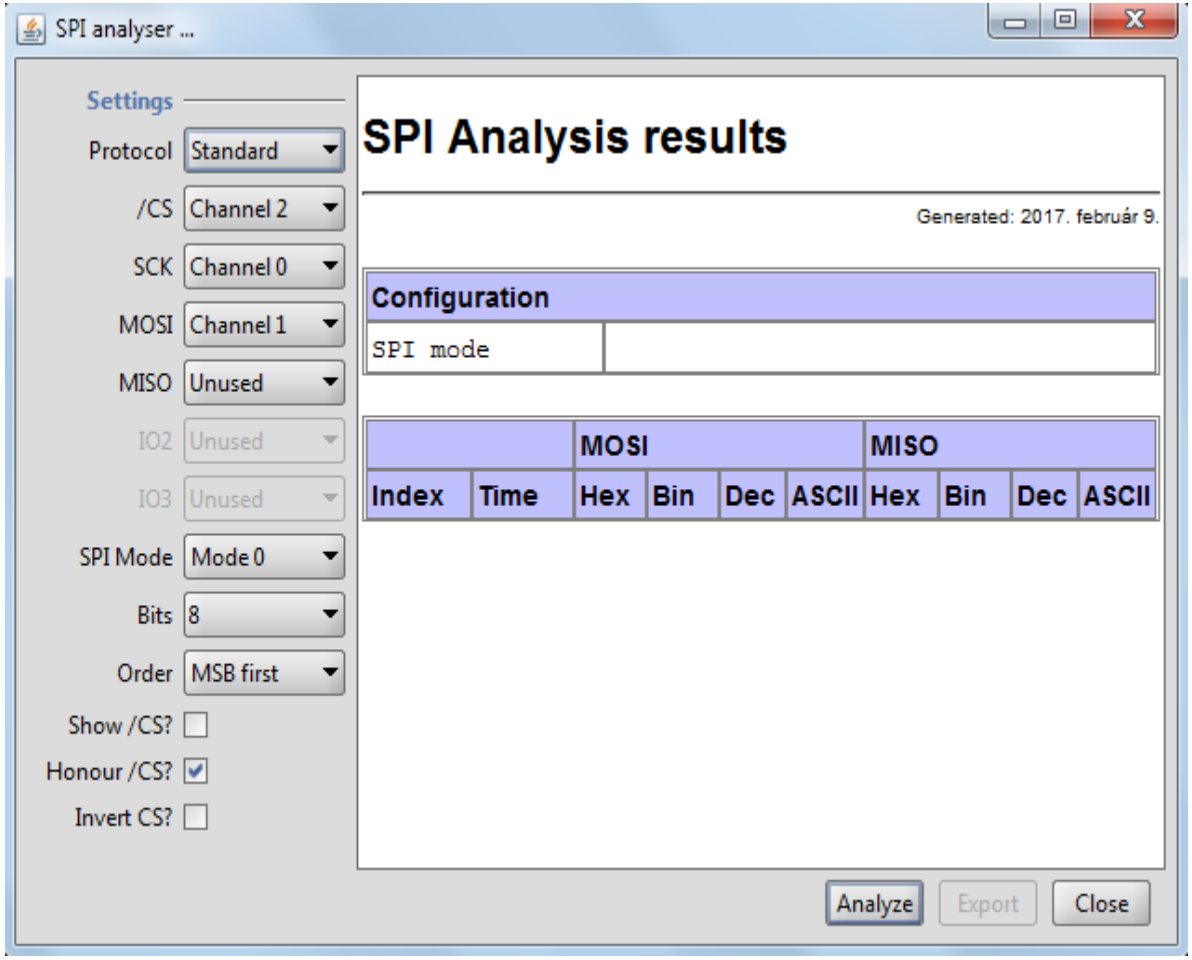
Impulzusok (átmenetek) számlálása, időmérés a kurzorok között

SPI protokoll értelmezése

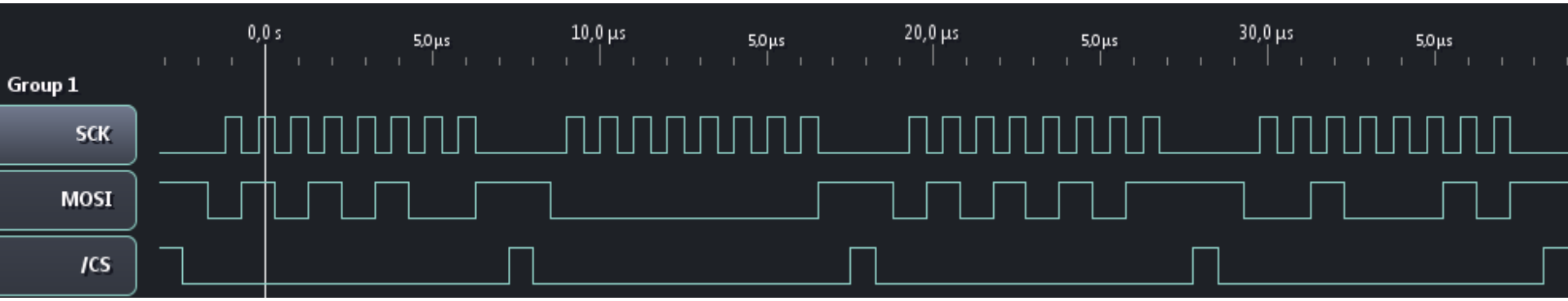
1. A jelek rögzítése után válasszuk ki a **Tools** menüből az **SPI analyzer** menüpontot!

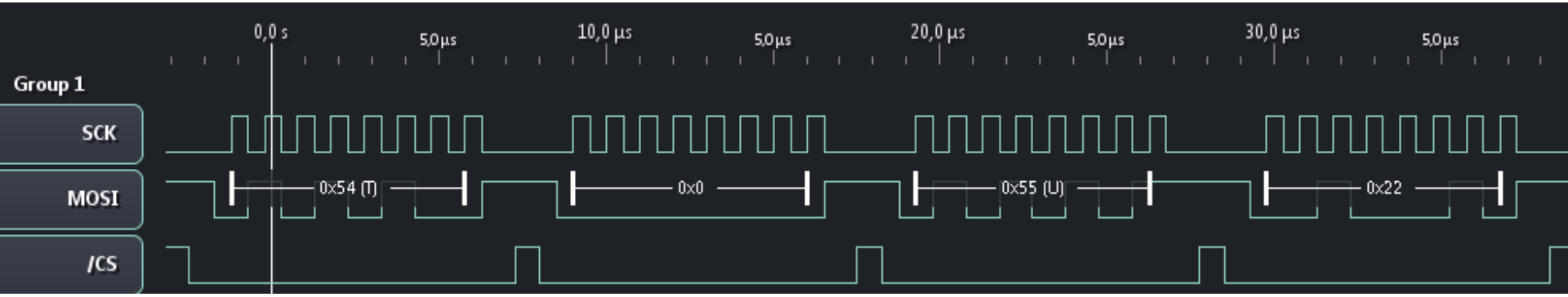
2. Állítsuk be a paramétereket!

- Jelek hozzárendelése a csatornákhoz
- SPI mód (általában 0)
- Bitek száma (8)
- Bitsorrend (MSB first)

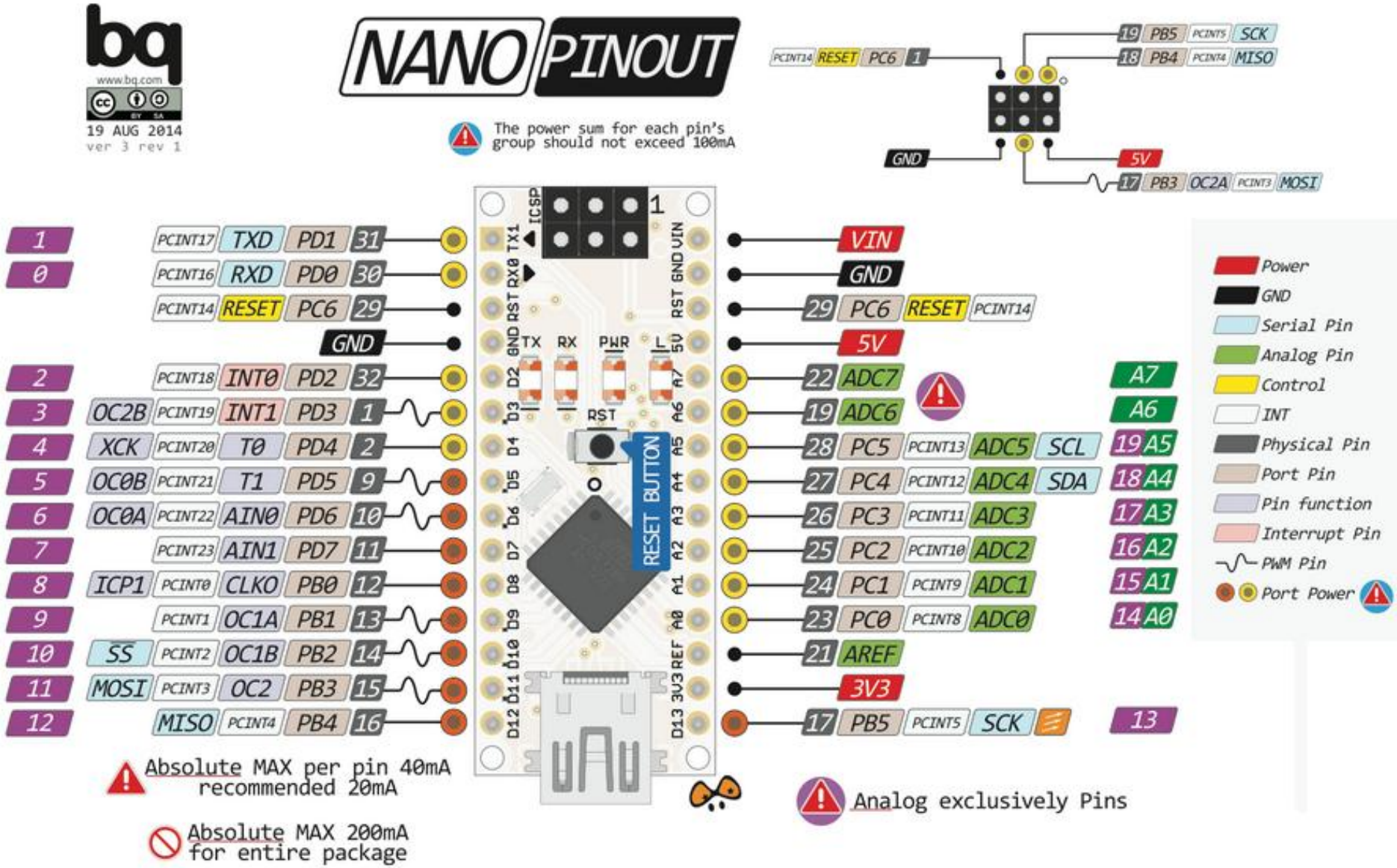


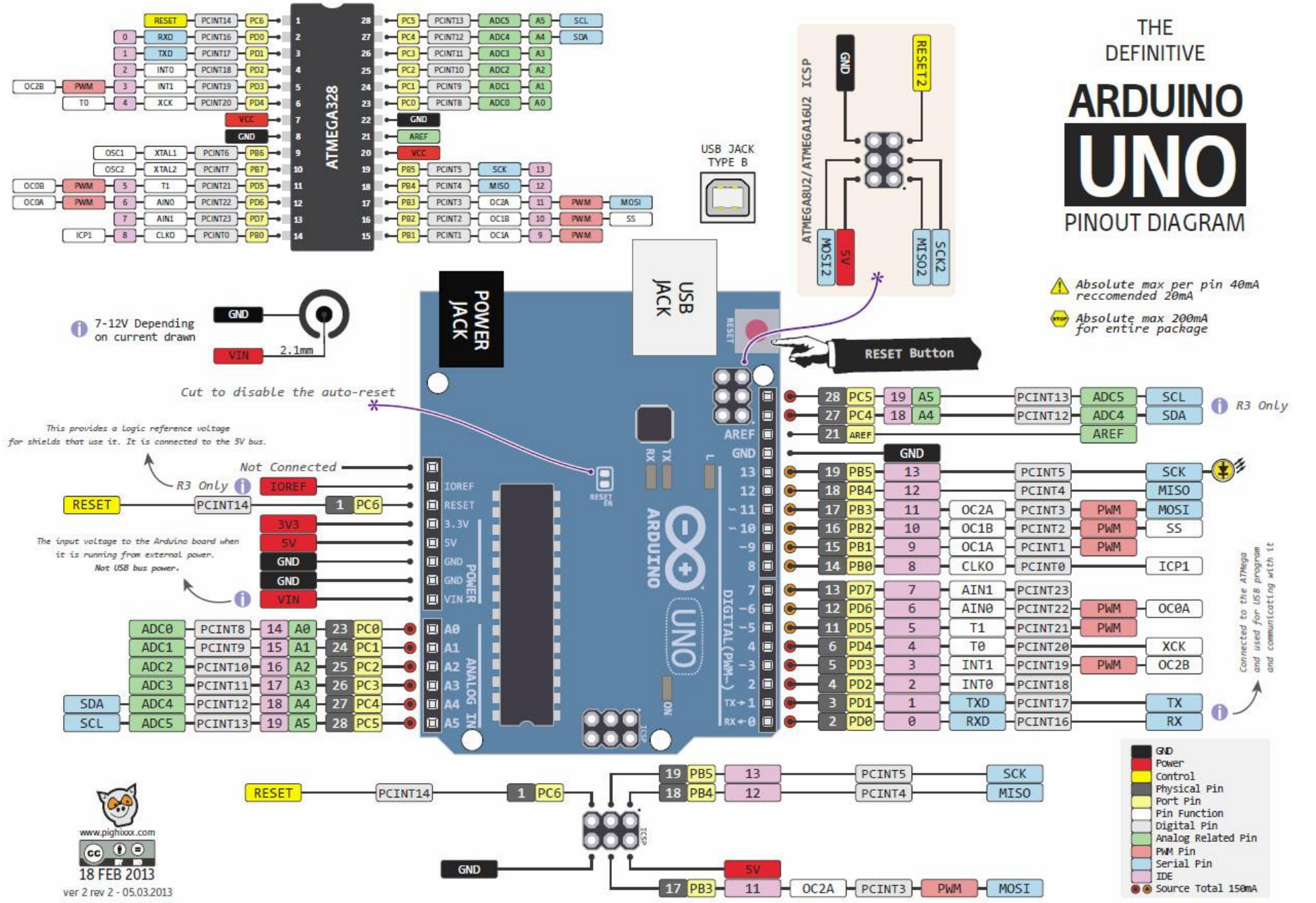
SPI protokoll értelmezése





Emlékeztető: Arduino nano v3.0




THE DEFINITIVE
ARDUINO UNO
PINOUT DIAGRAM
ATMEGA328
ATMEGA8U2/ATMEGA16U2 ICSP
USB JACK TYPE B
POWER JACK
USB JACK
RESET Button
7-12V Depending on current drawn
2.1mm
Cut to disable the auto-reset
This provides a logic reference voltage for shields that use it. It is connected to the 5V bus.
Not Connected
R3 Only
The input voltage to the Arduino board when it is running from external power. Not USB bus power.
Absolute max per pin 40mA reccomended 20mA
Absolute max 200mA for entire package
Connected to the ATMega and used for USB program and communicating with it
ARDUINO
UNO
DIGITAL (PWM~)
ANALOG IN
POWER
ICSP
GND
Power
Control
Physical Pin
Port Pin
Pin Function
Digital Pin
Analog Related Pin
PWM Pin
Serial Pin
IDE
Source Total 150mA
www.pighixxx.com
18 FEB 2013
ver 2 rev 2 - 05.03.2013