IOT KERETRENDSZEREK ÉS IPARI ALKALMAZÁSAIK

SmartComLab

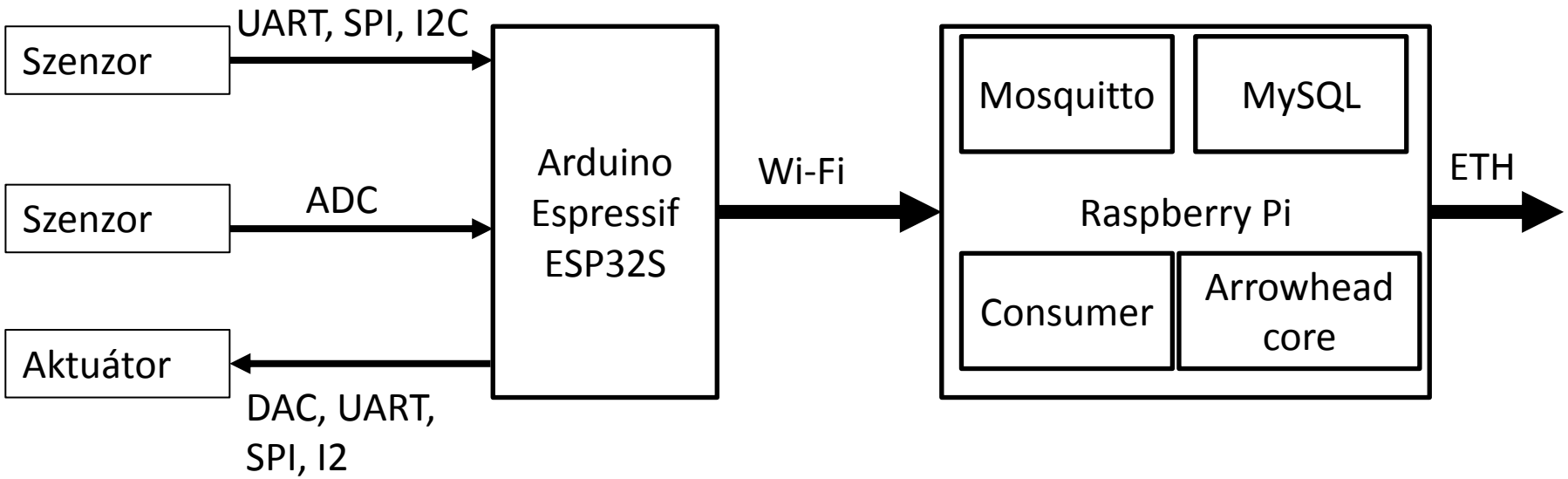
Gyors prototipizálási eszközök, homebrew – DYI I.

Tartalomjegyzék

Cél: elérni addig, hogy a következő feladat már a szerverre való adatbeküldés legyen

- Mikrokontrollerek
- Perifériák
- Perifériaprotokollok
- Arduino IDE
- Espressif 32S uC

Házi feladat



Szenzorok:

- Hőmérséklet és páratartalom
- Gyorsulásmérő, giroszkóp, magnetométer
- Talajnedvesség mérő
- Víz átfolyási sebességmérő
- Ultrahangos távolságmérő

Aktuátorok:

- LCD kijelző
- 9 grammos RC szervó
- Relé
- RGB LED szalag

Raspberry Pi 3B image

The image is for a 4GB microSD card (hopefully class 10 or above).

Preconfigured:

- Arrowhead core framework pre-configured
- Git, Maven, JDK 1.8, MySQL (MariaDB)
- PhpMyAdmin: *http://ip/phpmyadmin*
- Devtools (MCEdit, iptables…)
- Mosquitto MQTT broker
- The Raspberry hosts a Wi-Fi network and tunnels Internet if connected to eth0

Wi-Fi access point details:

- WiFi SSID: Arrowhead-Raspi-IoT#, password: arrowhead
- Raspi is DHCP master on the 192.168.42.10-50 range
- Raspi is the 192.168.42.1

https://learn.adafruit.com/setting-up-a-raspberry-pi-as-a-wifi-access-point/install-software

Mikrokontrollerek (nagyon röviden)

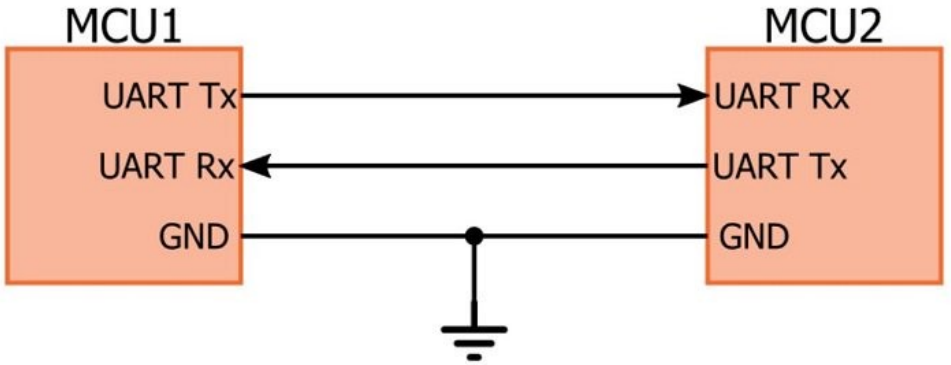
- Mikrokontroller: ~Intel 8085 (digit2)
- Általában C-ben programozzuk őket.
- Mai rendszerek általában:
  - 1-2 processzormag, ARM utasításkészlet
  - 8-32 bites adatábrázolás
  - MHz-es órajel: 1-500MHz
  - Perzisztens tárhely: flash alapú
  - SRAM: pár száz KB
- Perifériák: GPIO, ADC, DAC, UART, SPI, I2C, CAN, …
  - Memory mapped: a processzor a kezelő logikát belső regisztereken keresztül éri el

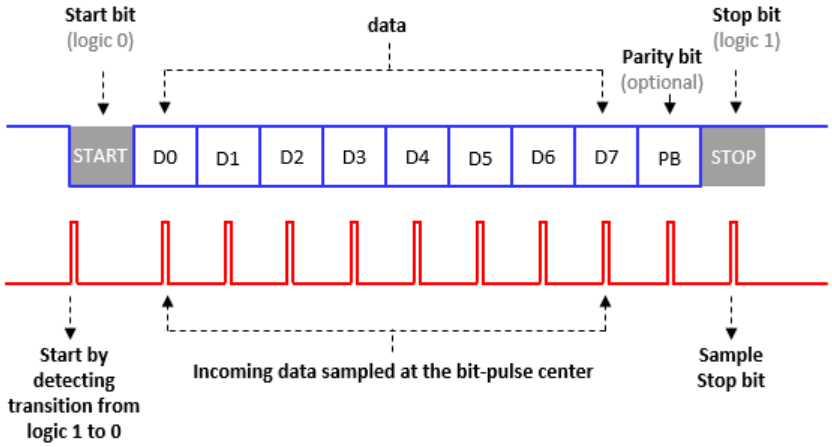
Perifériák I.

- GPIO: general purpose Input – Output
  - A kimenetek egyesével vezérelhetők
  - Kimenet: földre vagy tápra felhúzatás
  - Bemenet: beolvasni, hogy földön vagy tápon van-e (0/1)
- Analóg-digitál átalakítás: ADC
  - Általában 0-VCC közti feszültség átalakítás valamilyen pontossággal (lásd Vill/Labor2) bináris számmá (pl. 12 bit esetén 0-4095 közé)
  - Nagyon nagy ADC sebességek esetén mintavételezési jelenségek, törvények (Vill/Infokomm)
- Digitál-analóg átalakítás: DAC
  - Adott „szám” átalakítása feszültséggé

Perifériák II.

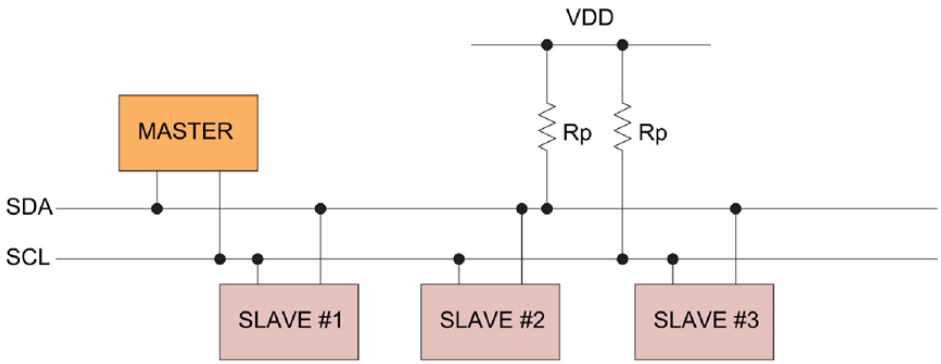
- Digitális periféria protokollok: UART
  - Ipari keretek közt: RS232, RS485
  - Két egyenrangú fél közti kommunikáció (TX/RX)
  - Fogalmak: baud rate, paritás és stop bitek száma



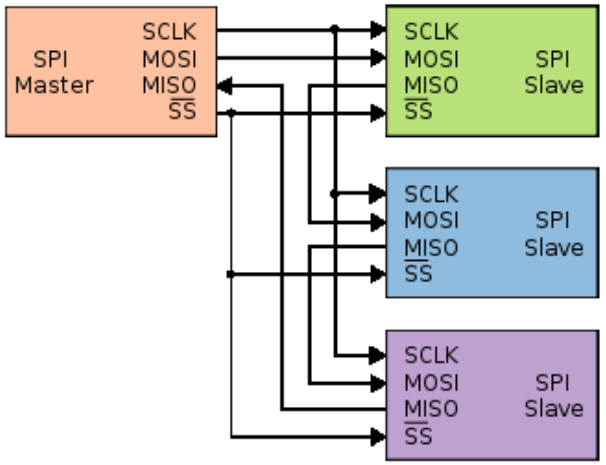


Perifériák III.

- Digit. Perif: I2C és SPI
  - 1 master, több slave modul kommunikál
  - A komm. vonal fel van húzva tápra: Pull-down módszerrel kommunikálnak
  - I2C: slave addr + register addr
  - SPI: adott SlaveSelect pin és regiszterek értékcseréje



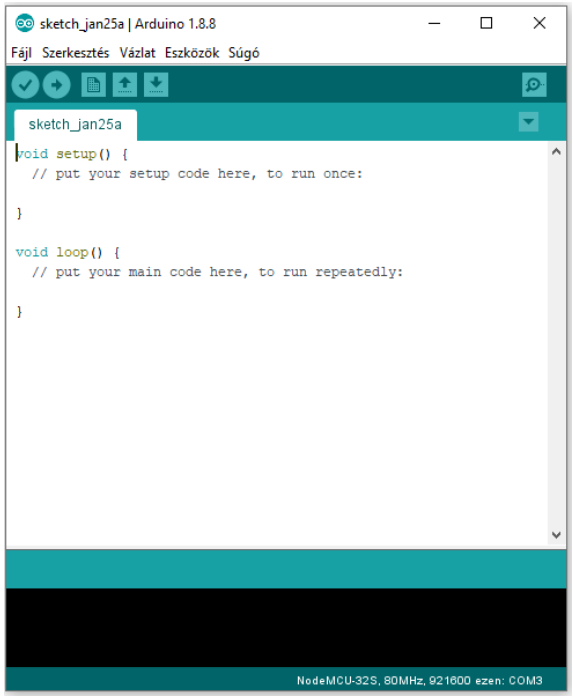
I2C vezetékelés és bekötés

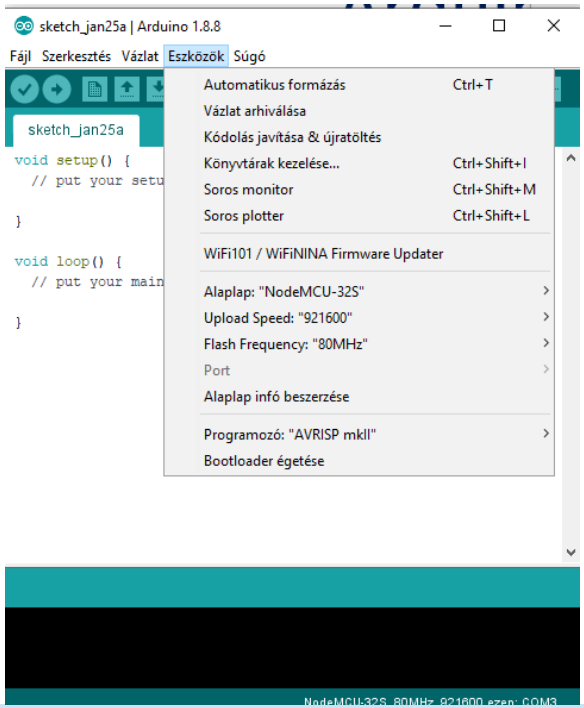


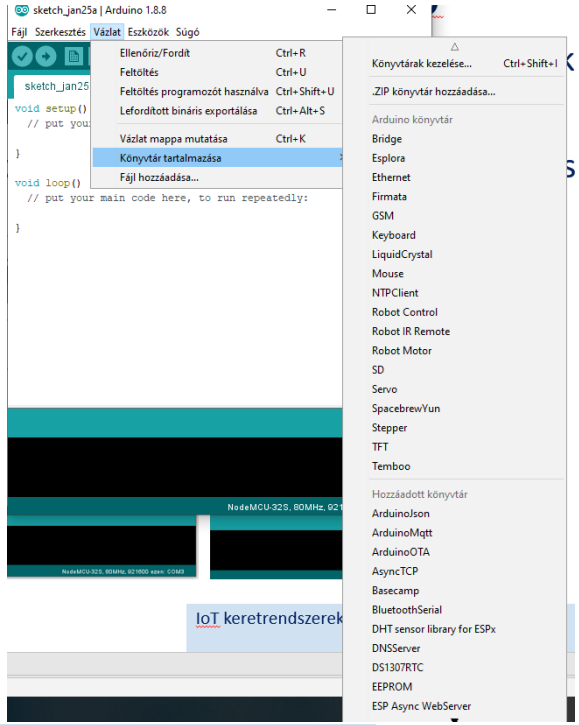
SPI vezetékelés és bekötés

Arduino

- Minimalista, de egységesített fejlesztői környezet
- Telepítési útmutatók:
- https://www.arduino.cc/
- Rengeteg felhasználható könyvtár, támogatott platform







Kódszerkezet

- N+1 library elérhető, kismillió „tutorial”

DHT_Test

```
#include "DHTesp.h"

DHTesp dht;

void setup()
{
  Serial.begin(115200);
  Serial.println();
  Serial.println("Status\tHumidity (%)\tTemperature (C)\t(F)\tHeatIndex (C)\t(F)");

  dht.setup(2); // data pin 2
}

void loop()
{
  delay(dht.getMinimumSamplingPeriod());

  float humidity = dht.getHumidity();
  float temperature = dht.getTemperature();

  Serial.print(dht.getStatusString());
  Serial.print("\t");
  Serial.print(humidity, 1);
  Serial.print("\t\t");
  Serial.print(temperature, 1);
  Serial.print("\t\t");
  Serial.print(dht.toFahrenheit(temperature), 1);
  Serial.print("\t\t");
  Serial.print(dht.computeHeatIndex(temperature, humidity, false), 1);
  Serial.print("\t\t");
  Serial.println(dht.computeHeatIndex(dht.toFahrenheit(temperature), humidity, true), 1);
}
```

Könyvtár include-ok

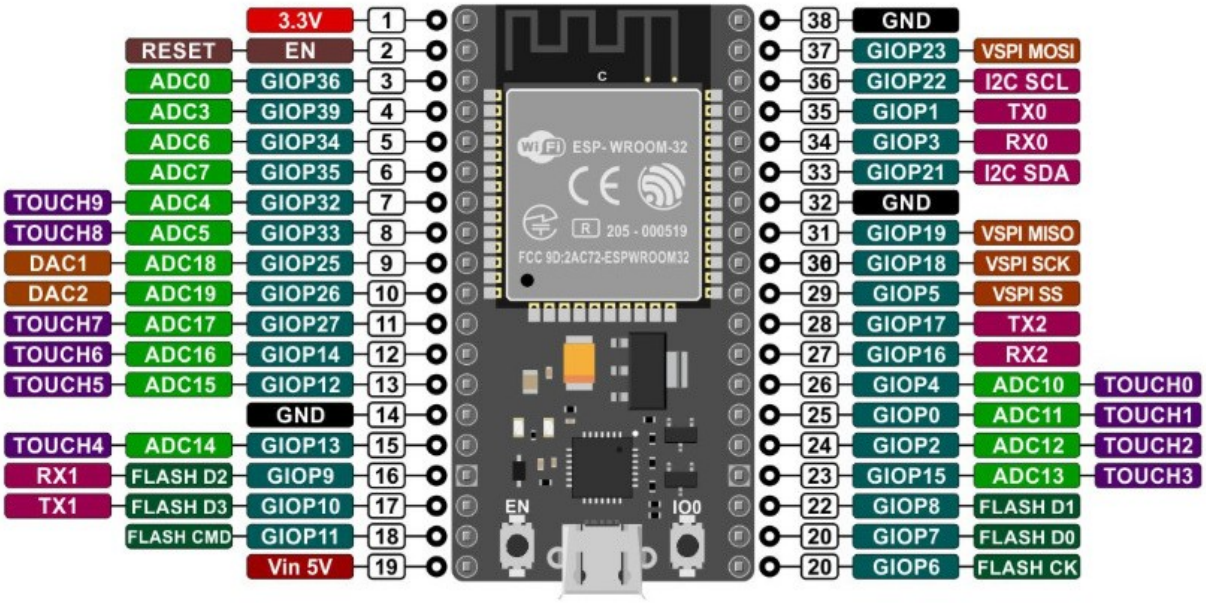
Globális változók

Setup: egyszer lefut induláskor

Loop: folyamatosan hivogatja az SDK

Espressif ESP-32S

- 2 processzormag, 240MHz
- Wi-Fi és Bluetooth 4.0 (BLE is) képes
- ESP-IDF: FreeRTOS opre alapú SDK
- Rengeteg fajta perifériát tud kezelni
- 3.3V VCC, de a devkit USB-ről tápolható



Espressif ESP-32S

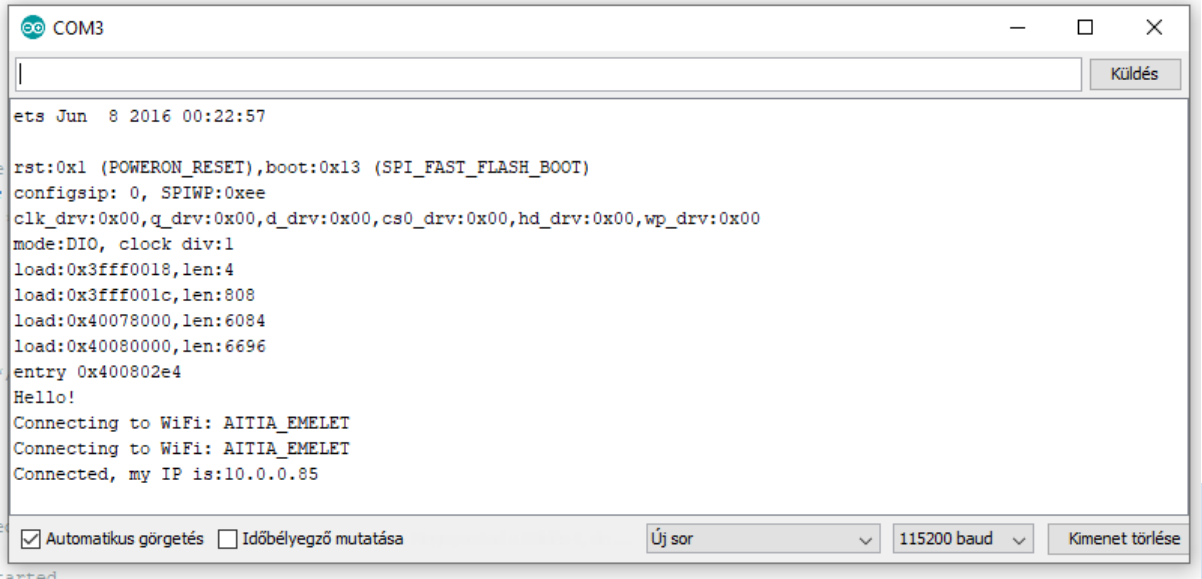
- Kínai mikrokontroller Arduino támogatással
- Board support package telepítése:
- https://github.com/espressif/arduino-esp32
- Ezek után elérhetővé válik a „NodeMCU 32S” az „Alaplapok” menüben
- A „feltöltési sebesség” 928kb/s-ra felvehető, a flash órajel pedig 80MHz-re állítható

SOROS debuggolása

DHT_Test

```
#include "DHTesp.h"

DHTesp dht;

void setup()
{
  Serial.begin(115200);
  Serial.println();
  Serial.println("Status\tHumidity (%)\tTemperature (C)\t(F)\tHeatIndex (C)\t(F)");

  dht.setup(2); // data pin 2
}

void loop()
{
  delay(dht.getMinimumSamplingPeriod());

  float humidity = dht.getHumidity();
  float temperature = dht.getTemperature();

  Serial.print(dht.getStatusString());
  Serial.print("\t");
  Serial.print(humidity, 1);
  Serial.print("\t\t");
  Serial.print(temperature, 1);
  Serial.print("\t\t");
  Serial.print(dht.toFahrenheit(temperature), 1);
  Serial.print("\t\t");
  Serial.print(dht.computeHeatIndex(temperature, humidity, false), 1);
  Serial.print("\t\t");
  Serial.println(dht.computeHeatIndex(dht.toFahrenheit(temperature), humidity, true), 1);
}
```

Soros portra lehet kiíratni vele, amit utána az Arduino IDE-ben a „Soros Monitor” menüponttal lehet figyelni



SmartComLab

Getting started

- Async TCP + HTTP szerver: https://github.com/me-no-dev/ESPAsyncWebServer
- HTTP kliens: https://github.com/espressif/arduino-esp32/tree/master/libraries/HTTPClient
- MQTT kliens: https://www.14core.com/pushing-topic-to-mqtt-with-esp32-esp8266/
- MPU 6050 driver (default: SDA - GPIO21 és SCL -GPIO22): https://create.arduino.cc/projecthub/Aritro/getting-started-with-imu-6-dof-motion-sensor-96e066
- DHT11 hőm. Szenzor driver: http://www.iotsharing.com/2017/05/how-to-arduino-esp32-dht11-dht22-temperature-humidity-sensor.html

- Google keresés: „<bármi> esp32 arduino” szintaxis alapján

Megjegyzés

- Érdemes megváltoztatni az alap flash particionálást, mert különben 1-2 include-től betelik a sketch max méret.

  Ehhez Hegedűs Csaba írt egy tutorialt (két fájl lecserélése a board support package-ben):

  – https://github.com/arrowhead-f/client-arduino/blob/master/ArrowheadProvider-NodeMCU-32S-NTP-v4/Readme.txt