Smart Watering Automation Network

Berta Máté

HPD5LB

Konzulens:

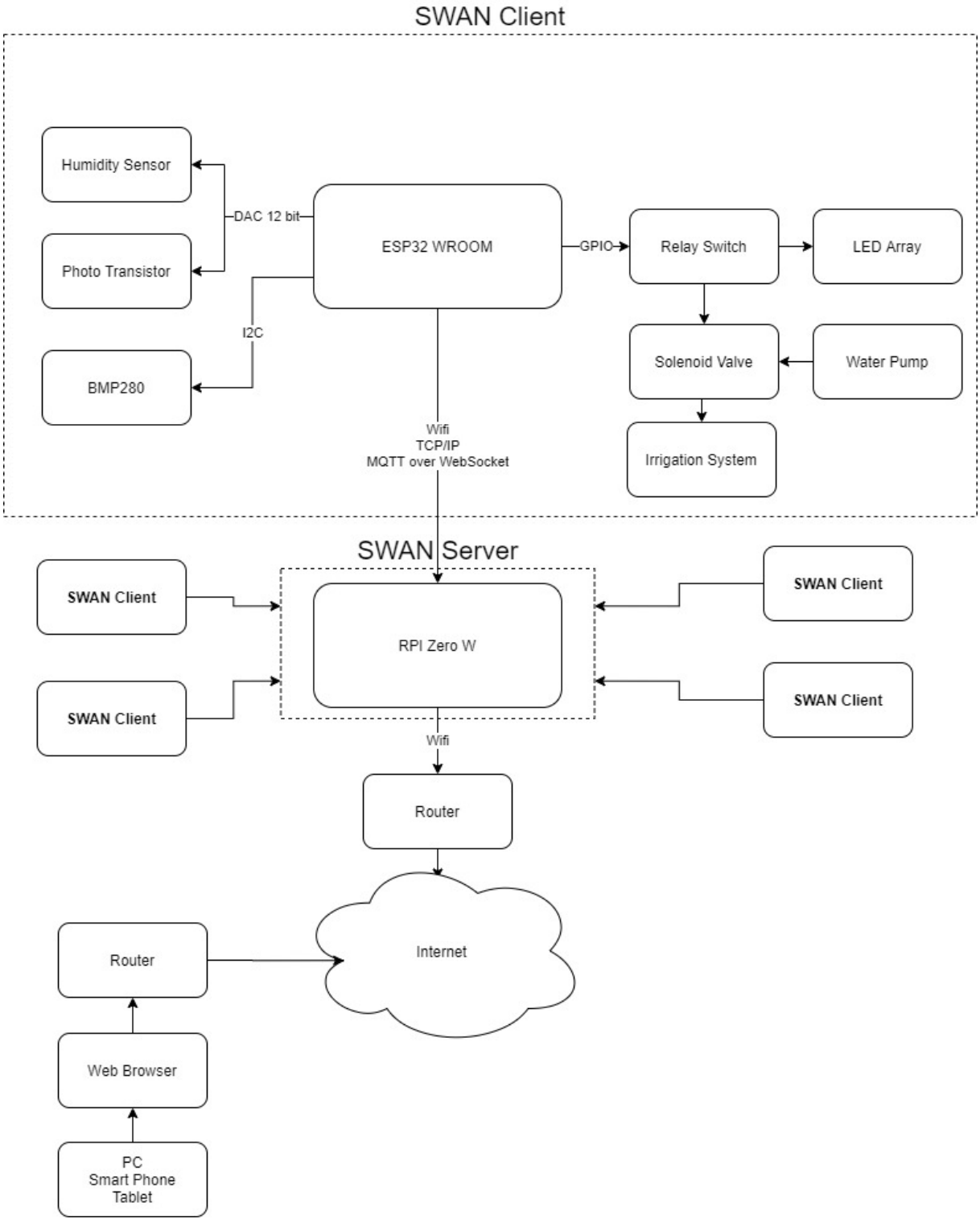
Naszály Gábor

Tartalomjegyzék

Feladat kiírás

Berta Máté
HPD5LB

Rendszerterv



Berta Máté
HPD5LB

Platform választás

Telekommunikációs megoldás

Szükséges a távoli információszerzés és beavatkozás megvalósításához valamilyen távkommunikációs megoldás beépítése a rendszerbe. A megoldás kiválasztása során figyelembe kell venni a felhasználandó alkatrészek bonyolultságát, villamos energia fogyasztását, költségeit és hatótávolságát.

A kiválasztás során három technológia alkalmazása került szóba, a Wifi, a Bluetooth Low Energy és a LoRaWAN.

Alacsony fogyasztás és nagy hatótávolság miatt LoRa technológia egyértelmű választás lenne, viszont magas prototipizálási költséggel rendelkezik (1:10-hez aránylik a LoRa:Wifi/BLE development boardok árához). Emellett a távoli beavatkozás megvalósítása nagyban növelte volna a villamos energia fogyasztást (Class A működés esetén túl sokszor kellene felébreszteni az eszközt, Class C működés esetén pedig nem tekinthető alacsony fogyasztású eszköznek), amely a legnagyobb előnyét mérsékelte volna. [1]

Bluetooth Low Energy alacsony belépési költséggel rendelkezik, viszont a kb. 10 méteres hatótávolsága (ESP32 SoC-vel) nem megfelelő egy átlagos kert - ház távolságának áthidalására, mesh hálózat kiépítése pedig feleslegesen növelné az alkatrészek számát és költségét. [2]

Végül Wifi technológiára esett a választás, alacsony ára miatt, emellett mérsékelhető fogyasztással és megfelelő hatótávval rendelkezik egy kerti öntöző megvalósításához (Extrém esetben eltúlzott nagyságú vevőantennával akár 10 km). Ráadásul az alkalmazandó szolenoid tekercses szelepek miatt a modul hálózati táplálással kell, hogy rendelkezzen emiatt a fogyasztás csökkentése csak a fenntartási költség mérséklése miatt fontos, tehát nem szükségesek az ultra low power megoldások, mint ahogy például egy telepről működő IoT adatgyűjtő esetén lenne. [3]

Wifi Chipset és mikrokontroller választás

Manapság könnyen beszerezhetők olyan Wifi kommunikációra alkalmas SoC-k, melyek sok általános mikrokontrolleres feladatot ellátnak, tartalmazzák a megfelelő hardverelemeket (nem wireless kommunikációs perifériák, AD/DA átalakítók, PWM generátorok stb.). Célszerű egy ilyen kontrollert választani, így csökkentve az alkatrészek számát.

Két könnyen hozzáférhető mikrokontroller jött szóba a projekttel kapcsolatban, ESP8266 és ESP32. Előbbi kisebb teljesítményű, viszont ez kisebb beszerzési költségekkel is jár és a feladat elvégzésére alkalmasnak tűnik.

Mindenképpen mérlegelni kellett a skálázhatóságot, hiszen egy I2C szenzor kiolvasása és egy ADC csatornán való feszültségmérése még bőven megoldható ESP8266 segítségével (bitbanged I2C-vel rendelkezik és egy ADC-vel) viszont, ha több mérési pontot szeretnénk a rendszerbe vinni, akár problémákat okozhat a hardveres korlátoltság.

Emellett a számítási kapacitásra is gondolni kell, hiszen a hálózati kommunikáció gyakran sokáig foglalhatja az erőforrásokat, így a szabályozó ritkább mintavételekkel dolgozhat, amely a rendszer időállandóját növeli, ami akár hibás szabályozást is eredményezhet. [4]

A választás az ESP32 SoC-ra esett, mivel hardveres I2C perifériával rendelkezik 18 csatornás 12 bites SAR ADC-vel (ADC1 – 8 csatorna ADC2 – 10 csatorna), illetve két processzor maggal, amelyek segítségével teljesen elkerülhető a szabályozást végző task kiéheztetése. [5]

Másik fontos érv az ESP32 mellett, az volt, hogy a gyártó támogatja a FreeRTOS operációs rendszert a chipen, némi kiegészítéssel a két magos processzor miatt. [6]

Fejlesztői környezet

ESP32 platformra rengetek programozási segédlet, fejlesztő eszköz és szoftver könyvtár létezik, mivel kedvelt a hobbi IoT „kütyük” készítői között. Ezen eszközök közül legismertebb az Arduino Open Hardware/Software dokumentáció és az Arduino IDE. Gyors és könnyű fejlesztést tesz lehetővé, viszont a hardver egyes elemeihez nehezíti vagy ellehetetleníti a hozzáférést (pl.: energia csökkentő módok ki/be kapcsolása). [7]

A választott megoldás a gyártó által kiadott szoftver könyvtár Espressif IoT Development Framework (ESP-IDF) használata lett. A fejlesztés könnyítésére nem csak egyszerű text editor és a gyártó által kiadott programozói szoftvert (esptool) használtam fel, hanem az Eclipse IDE és ESP-IDF plugin felhasználásával egy felületen írhattam a kódot, programozhattam az eszközt és monitorozhattam a debug portot (UART). [8]

A fejlesztői környezet telepítését egy virtuális gépen végeztem el, így könnyen „mozgatható” munkaállomást kaptam végeredményül. A virtuális gép Ubuntu 19.10 operációs rendszerrel rendelkezik, a fejlesztőkörnyezet működtetéséhez szükséges szoftver komponensek a következők: [9]

- Eclipse CDT (C/C++ plugin, GCC compiler)
- Python 3.7
- Java 8
- Git
- ESP-IDF v4.0

A gyártó által kiadott telepítő állomány folyamatosan változik, az újabb és újabb verziók telepítése során gyakori, hogy hibaüzenetekkel találkozik a felhasználó. A telepítés idejében a legfrissebb verzió Windows és Linux alatt is a virtualenv python package legújabb verziójával nem tudott települni, ezért vissza kellett azt állítani egy régebbi verzióra.

CMake projekt szerkezete

Hőmérséklet és nyomás mérés

I2C

Gyakori problémát jelent több szenzor alkalmazása során az AD átalakítók korlátozott száma, annak érdekében, hogy több azonos vagy más fizikai mennyiséget mérő szenzorral tudjunk mikrokontrolleres környezetben mérni, valamilyen integrált áramkörök közötti busz kommunikációt szükséges használni.

A projektben ritka (pl. percenkénti) adatgyűjtésre van szükség, nem időkritikus egy mért adat kiolvasása, így az IoT alkalmazásokban egyik leggyakoribban alkalmazott busz rendszert az I2C-t (Inter-Integrated Circuit) lehetett használni.

A busz két vezetékes (SDA, SCL), 100 kHz és 5 MHz közötti sebességű kommunikációra képes. Szükség esetén lassú slave eszközök lassíthatnak a kommunikáción az SCL vonal alacsony szinten tartásával.

Minden eszköz egyedi 7 bites címmel rendelkezik, melyeket a gyártók adnak meg. Annak érdekében, hogy azonos mérőeszközök is a buszon lehessenek, a gyártók a címek csak egy részét specifikálják, így a szabadon hagyott bitek hardveres beállításával külön cím adható kettő vagy több azonos típusú chipnek.

Előnye ennek a protokolnak, hogy kis erőforrásokat igényel, illetve akár hardveres támogatás nélkül is használható a mikrokontroller oldaláról. [11]

BMP280

A Bosch által gyártott BMP280 chip hőmérsékelt és nyomás mérésére alkalmas, I2C és SPI buszon keresztül vezérelhető szenzor, melynek alacsony a fogyasztása és ezen felül fogyasztás csökkentő alvó üzemmódja is van. [12]

A kerti öntöző számára értékes információt jelent a levegő hőmérséklete, melyből egyaránt közvetlen és közvetett következtetéseket vonhatunk le és dönthetünk a beavatkozás szükségességéről.

Közvetlenül tudjuk, ha fagy miatt nem lehet öntözni, illetve túl nagy hőség esetén dönthetünk úgy, hogy hűtés céljából öntözünk.

Közvetett információt nyerhetünk a napszakról, évszakról több mérési pont alkalmazásával akár a kert felületét érintő árnyékról is.

Nyomás mérésével közvetetten információt gyűjthetünk a páratartalomról adott hőmérséklet mellett, ezzel akár különböző növényfajták ideális környezetére is szabályozhatunk.

A szenzor használatához a gyártó által biztosított C-ben megírt drivereket használtam fel [13], melyeket csak a konkrét mikrokontrollerhez tartozó I2C/SPI kezelő kóddal kellett kiegészíteni.

Talaj nedvességtartalom mérés

Kapacitív alapon működő mérési elv

Rezisztív alapon működő mérési elv

Hálózati kommunikáció

Wifi

MQTT

Irodalomjegyzék

[1] LoRa technológia összefoglaló dokumentáció:
https://buildmedia.readthedocs.org/media/pdf/lora/latest/lora.pdf

[2] Bluetooth hatótávolság az ESP32 SoC-vel:
https://www.esp32.com/viewtopic.php?t=6959

[3] ESP32 extrém nagy hatótávolságú wifi kommunikáció:
https://www.espressif.com/en/media_overview/news/esp32%E2%80%99s-wi-fi-range-extended-10-km-directional-antenna

[4] ESP8266 dokumentáció:
https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

[5] ESP32 technical reference manual:
https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[6] ESP32 FreeRTOS kiegészítése:
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/freertos-smp.html

[7] Arduino hivatalos dokumentáció és források:
https://www.arduino.cc/

[8] ESP32 IDF programming tools:
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html

[9] ESP IDF plugin for Eclipse:
https://github.com/espressif/idf-eclipse-plugin#Prerequisites

[10] ESP IDF CMake Project szerkezete
https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/build-system.html

[11] I2C dokumentáció:
https://i2c.info/

[12] BMP280 dokumentáció:
https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmp280-ds001.pdf

[13] BMP280 driver github elérése:
https://github.com/BoschSensortec/BMP280_driver

[14] Kapacitív mérés:

[15] Rezisztív mérés:

Mellékletek