Edukativni pristup enkriptovanom prenosu podataka u embedded i frontend razvojnim okruženjima

Ivan Gutai, Member, IEEE, Prof. dr Platon Sovilj, Member, IEEE, Marina Subotin, Member, IEEE, Marjan Urekar, Member, IEEE, Jelena Milojević, Member, IEEE, Milica Mitrović, Member, IEEE

***Apstrakt*—Pre IIoT-a (The Industrial Internet of Things), embedded programiranje i frontend programiranje nisu mogli da se nađu ni u istoj rečenici. Hardver koji je omogućio neprimetnu integraciju ove dve kompleksne oblasti je Espressif-ov ESP32 MCU (MicroController Unit). Tek kada je broj uređaja povezanih na internet dostigao značajnu cifru, u fokus je došla bezbednost podataka. ESP-NOW je Espressif-ova tehnologija za bežični prenos podataka. Prenos može biti enkriptovan i obezbeđuje bezbednu kumunikaciju između više ESP32. HTTPS (Hypertext Transfer Protocol Secure) je protokol koji povećava bezbednost na internetu. Ovaj rad predstavlja uputstvo za konfigurisanje razvojnog okruženja za ESP32, ESP-NOW primer, primer HTTPS servera i prikaz programerske prakse za upravljanje greškama. Kao dodatak prikazan je i prilagodljivi grafički korisnički interfejs IIoT uređaja. U ovom trenutku postoji mnogo putanja u embedded i frontend programiranju. U ovom radu je izabrana putanja: ESP32 za hardver i C++ za firmver. JavaScript, HTML5 i CSS3 su neizbežan deo modernih industrijskih uređaja, pa je dat primer korišćenja JavaScript Highcharts biblioteke. Korišćena kombinacija hardvera i softvera košta manje od 10$, što čini konfiguraciju pogodnom za zemlje u razvoju. Highcharts biblioteka je vlasnički softver, ali u edukativne svrhe se može koristiti u okviru Creative Commons (CC) Attribution-Non-Commercial licence.**

***Ključne reči*— IIoT; embedded programiranje; frontend programiranje; ESP32; ESP-NOW; HTTPS; prilagodljivi dizajn; SPIFFS; C++; JavaScript; HTML5; CSS3; Highcharts; Web Bazirani Merno-Akvizicioni Sistemi; JSON.**

I. Uvod

Pametni uređaji su postali deo naše svakodnevice, a podjednako ih koristimo i kao alat i kao nešto što se može nazvati hobi projektom. Broj takvih uređaja i njihovih funkcionalnosti se svakodnevno uvećava. U takvim okruženjima moramo obratiti pažnju na bezbednost informacija i ne smemo zaboraviti dobre programerske i inženjerske prakse. Danas svi imamo jednaku mogućnost da razvijemo prototip IIoT uređaja, koji će biti deo naše kućne Wi-Fi mreže, a uređaj možemo kontrolisati preko web pretraživača. Ovaj rad daje niz smernica, sa namerom da čitaocima ubrza ulazak u svet embedded programiranja i/ili web programiranja. Dato je praktično uputstvo kako se kreira jedan IIoT uređaj. Najteži deo na početku je izbor pravog hardvera i odgovarajućeg skupa tehnologija. Autori su izabrali: ESP32[1], C++, JavaScript, HTML5 i CSS3. ESP-NOW [2] tehnologija omogućava povezivanje velikog broja ESP32 uređaja, koji komuniciraju međusobno preko Wi-Fi-ja. Čist tekst je podložan izmenama u web aplikacijama u toku prenošenja preko Wi-Fi-ja. Ukoliko dobijemo priliku da nešto enkriptujemo, to treba odmah da uradimo. ESP-NOW podaci mogu biti enkriptovani preko LMK (Local Master Key), koji mora da se slaže i na prijemnicima i na predajnicima. Dodavanje senzora i releja je zasebna oblast i čitaoci mogu da biraju između raznih open-hardware rešenja i vlasničkih alternativa kao što su proizvodi Mikroelektronike. Nakon završetka sa hardverskim delom, koristi se SPIFFS (Serial Peripheral Interface Flash File System) memorija, za postavljanje (eng. deploy) web aplikacije. SPIFFS sadrži web aplikaciju koja je kreirana sa modernim i besplatnim alatima. Jedina razlika je u hosting-u i aplikacija se ne nalazi na tipičnom web serveru, već se nalazi na ESP32.

Ivan Gutai – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: gutai@uns.ac.rs).
Platon Sovilj – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: platon@uns.ac.rs).
Marina Subotin – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: marina.bulat@uns.ac.rs).
Marjan Urekar – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: urekarm@uns.ac.rs).
Jelena Milojević – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: jmilojevic@uns.ac.rs).
Milica Mitrović – Fakultet tehničkih nauka, Novi Sad, Srbija (e-mail: m.mitrovic@uns.ac.rs).

II. Konfigurisanje integrisanog razvojnog okruženja

Ceo proces se započinje instalacijom Arduino IDE (Integrated Development Environment) [3].

ESP32 treba dodati na listu postojećih razvojnih sistema: "Arduino IDE, File, Preferences, Additional Boards Manager URLs [4]."

Proširivanje liste postojećih razvojnih sistema sa ESP32: "Tools, Board, Boards Manager, esp32 i Install". Nakon uspešne instalacije, "ESP32 Arduino" će se pojaviti na listi dostupnih razvojnih sistema. Biće dostupno mnoštvo ESP32 razvojnih ESP32 razvojnih sistema, uključujući i "ESP32 Dev Module".

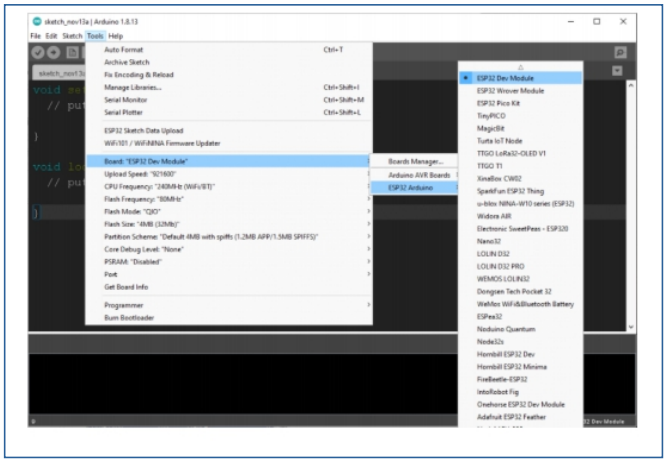
Dodavanje "ESP32 Sketch Data Upload" opcije omogućava postavljanje fajlova na SPIFFS. Možemo to zamisliti kao postavljanje frontend koda na ESP32 sistem. Dodatak (eng. plugin) [5] za postavljanje koda je potrebno kopirati unutar Arduino IDE foldera. Dodatak je potrebno iskopirati u sledeći direktorijum: "C:\Program Files

(x86)\Arduino\tools\ESP32FS\tool\esp32fs.jar".

Tamna tema za Arduino IDE je dostupna na [6]. Fajlovi od nove teme treba da zamene postojeće, u folderu: "C:\Program Files (x86)\Arduino\lib\theme". Poželjno je da se fajlovi od originalne teme takođe sačuvaju.

Fajlovi koji su potrebni, da bi uopšte bio moguć rad sa ESP32 su [7]. Navedene fajlove je potrebno otpakovati u lokalni Arduino folder: "C:\Users\Ivan Gutai (odgovarajuće korisničko ime)\Documents\Arduino\hardware\espressif\esp32". Treba napomenuti da je "esp32" folder zapravo preimenovani "arduinoesp32-master" folder.

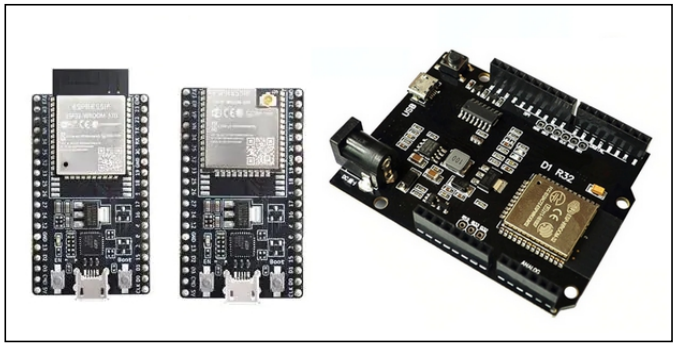
Ukoliko ESP32 razvojni sistem nije vidljiv u Device Manager-u, potrebno je instalirati drajvere za programer. Najpopularniji su CH340 [8] i CP210x [9]. Nakon uspešne konfiguracije, razvojno okruženje izgleda kao na slici 1.



Sl. 1. Arduino integrisano razvojno okruženje, konfigurisano za ESP32.

III. Izbor odgovarajućeg hardvera

ESP32 ima više varijacija. U ovom radu je korišćen ESP32-DevKitC-32D [10] sa integrisanom bakarnom antenom. Da je bilo potrebe da pojačavamo Wi-Fi signal, upotrebili bi ESP32-DevKitC-32U [11], sa zasebnom antenom. Ukoliko je neko navikao da radi sa Arduino Uno, postoji ESP32 koji fizički veoma sličan, a reč je o verziji Wemos D1 R32 [12]. Slika 2 prikazuje sva tri spomenuta tipa ESP32 razvojnih sistema.



Sl. 2. ESP32 razvojni sistemi: ESP32-DevKitC-32D, ESP32-DevKitC- 32U i ESP32 Wemos D1 R32.

IV. Istraživanje ESP-NOW tehnologije

Postoji kompletno uputstvo za konfigurisanje ESP-NOW, sa više predajnika i sa više prijemnika [13]. ESP-NOW se koristi ukoliko imamo 2 ili više ESP32 uređaja, između kojih želimo da ostvarimo komunikaciju preko Wi-Fi-ja. Potrebno je istaći da je struktura podataka na predajniku i na prijemniku mora biti identična. Ukoliko se odlučimo da koristimo enkriptovanu komunikaciju, LMK mora biti identičan i na prijemnicima i na predajnicima. Takođe, za Wi-Fi komunikaciju možemo izabrati kanale od 1 do 14. Paketi podataka od 250 bajta se šalju i primaju desetinama puta u sekundi, što ih čini pogodnim za podatke koji se brzo menjaju. Navedeni podaci se dobijaju sa uređaja koji vrše akviziciju podataka, kao što su merenje pozicije u prostoru i ugaonog ubrzanja.

V. Dodeljivanje lokalne fiksne IP adrese uređaju

Svaki put kada se uređaj poveže u kućnu ili industrijsku Wi-Fi mrežu, dobija različitu lokalnu IP adresu. Za to je odgovoran DHCP (Dynamic Host Configuration Protocol), što je u potpunosti u redu, ukoliko programer koristi uređaj, ali krajnji korisnik to ne želi. Iz navedenog razloga uređaju dodeljujemo fiksnu lokalnu IP adresu, koja se uklapa u parametre naše kućne Wi-Fi mreže. Na slici 3 je prikazana konfiguracija koju treba da primenimo.

```
IPAddress local_IP(192, 168, 1, 99);
IPAddress gateway(192, 168, 1, 1);
IPAddress subnet(255, 255, 0, 0);
IPAddress primaryDNS(8, 8, 8, 8);
IPAddress secondaryDNS(8, 8, 4, 4);
```

Sl. 3. Postavljanje lokalne fiksne IP adrese i Google DNS (Domain Network Server) servera.

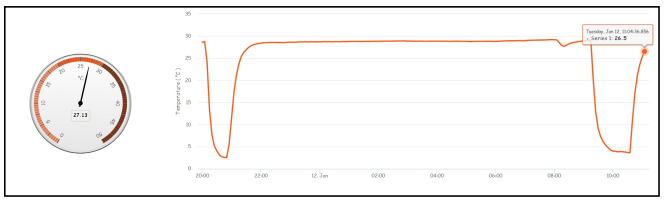
VI. Istraživanje SPIFFS tehnologije

Biblioteka koja omogućava kreiranje kućnog REST (Representational State Transfer) servera je dostupna na Github-u [14].

U Arduino IDE, sve dodatne biblioteke se preuzimaju na sledeći način: "Tools, Manage Libraries, search i install". Ključne reči su: "esp32 HTTPS", a zatim je potrebno instalirati "ESP32_HTTPS_SERVER" biblioteku. Iz navedene biblioteke, koristimo primer "REST-API". Pomoću ovog primera se generiše i Self Signed sertifikat, koji je koristan prilikom razvoja i omogućava upotrebu HTTPS-a. Fajlove sa ekstenzijama: .html, .js, .css i ostale, je potrebno smestiti u folder: "REST-API\data\public". Sve može biti postavljeno na SPIFFS, koristeći opciju iz Tools sekcije: "ESP32 Sketch Data Upload".

VII. Kreiranje web interfejsa IIoT uređaja

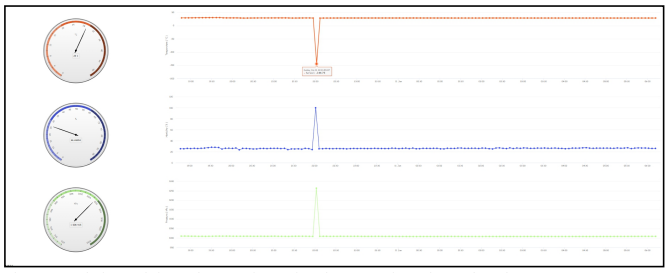
Slika 4 prikazuje deo prilagodljivog web interfejsa sa Highcharts bibliotekom [15], koja je zasnovana na JavaScript-u.



Sl. 4. Deo prilagodljivog interfejsa koji se sastoji od analogne skale i grafika.

VIII. Upravljanje greškama i prevazilaženje problema

Prilikom kreiranja kompleksnih sistema, konstantno vodimo računa o desetinama komponenata. Pored sve pažnje, ponekad se desi da se zaborave osnovni principi, koji su stari koliko je staro i programiranje. Jedan od tih principa je žargonski rečeno, upravljanje "greškama" ili u preciznoj programerskoj terminologiji, upravljanje "izuzecima". Koji god termin koristili, svakako ne smemo da dozvolimo da do korisnika stigne pogrešna ili nepotpuna informacija. Na slici 5 je prikazan web interfejs u kom je zabeležen "loš" signal, koji je direktno plasiran sa hardvera.



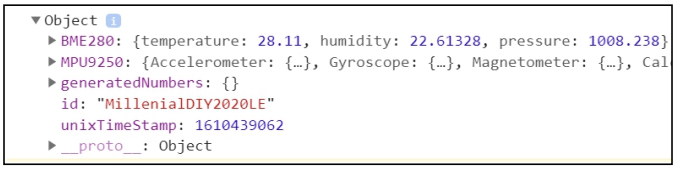
Sl. 5. Web interfejs nakon primanja "lošeg" signala sa hardvera.

Tehnički gledano, u ovom slučaju nije reč o grešci, ali je očigledno da je reč o brojevima koji su izašli van opsega. Korisnik uređaja i/ili aplikacije, to ne želi da vidi. Ukoliko postoji verovatnoća da će se to dogoditi, to mora biti na neki način iskontrolisano, a korisniku treba da bude omogućen kontinualan ispravan rad uređaja, tj. sitema za akviziciju. Navedeni primer predstavlja podsetnik, da moramo biti svesni opsega brojeva koje očekujemo u svakoj komponenti kompleksnog sistema. Treba uzeti u obzir da prilikom očitavanja vrednosti sa senzora Bosch BME280 [16], koji omogućava merenje parametara okruženja, uključujući i atmosferski pritisak, vrednosti budu u opsegu od 300 hPa do 1100 hPa. Sve što je van navedenog opsega je rezultat nekog vida greške, npr. nepotrebnog preopterećenja sa hardverske strane, koje može da prouzrokuje povremena (eng. intermittent) očitavanja "loših" podataka sa senzora. Najveći problem kod povremenih grešaka je činjenica, da ne može da se utvrdi kada će se dogoditi i kako će se one izraziti. Takav tip podataka ne sme da dođe do korisnika, pošto je pogrešan, a takođe se ne sme ni sakriti. Korisniku moraju biti pružene precizne instrukcije šta da radi, bez mnogo tehničkih detalja. Tehnički detalji se zapisuju na takav način da programer može jasno da vidi šta se i kada se dogodilo. Frontend ima mnoštvo naprednih opcija za prikazivanje informacija korisniku, kao što su "toast" notifikacije i popunjavanje polja sa specifičnim porukama. Programeri imaju kreativnu slobodu da izaberu if..else if..else ili try...catch...finally blokove. Dobra analiza sprečava prikaz pogrešnih informacija, kao što je atmosferski pritisak u dnevnoj sobi od 1264 hPa. Da bi bili u potpunosti sigurni da raspolažemo sa ažurnim informacijama, preporučljivo je da se koristi vremenska oznaka (eng. timestamp), Unix tipa, ili bilo koja druga. Vreme može biti očitano sa hardverske komponente kao što je DS3231 [17], preko NTP (Network Time Protocol) servera [18], ili na način po izboru čitaoca. Treba napomenuti da je vremenski žig vrlo bitno parče informacije i ako se pravilno koristi, omogućava da se grafici ne ažuriraju ukoliko ne postoje sveži podaci. Navedeni pristup sprečava širenje dezinformacija.

IX. Test development

Odnos programiranja i testa mora biti bar 1:5. Ukoliko će uređaj imati industrijsku primenu, potrebno je još više testirati. Ekstremne vrednosti mogu biti korisne prilikom testiranja, zato što ako postoji i 1 % šansa da se nešto dogodi, to će se jednom i dogoditi. Kao programeri, moramo biti spremni da izađemo sa tim na kraj i da držimo stvar pod kontrolom. Jedna od prednosti sistema zasnovanom na ESP32, je mogućnost da web interfejs koristimo kao tzv. test bench. Moguće je povezati mnoštvo senzora, dodeliti im ID-je i pratiti njihova očitavanja, npr. na svakih 5 minuta u naredna 24 sata. Mikrokontroler ima dovoljno memorije da prati navedene parametre i bez web interfejsa. Možemo se odlučiti za web interfejs ukoliko želimo da pratimo očitavanja sa mnoštva senzora i da prikazujemo rezultate na uređajima sa velikim i preglednim ekranima, kao što su UHD (Ultra High Definition) televizori rezolucije 3840 px sa 2160 px. Pisanje medija upita (eng. media queries) nije ništa kompleksnije od pisanja medija upita za mobilne telefone i tablete i kreće od : "@media only screen and (max-width: 3840px) {}".

U sred pisanja firmware-a i kreiranja web aplikacije, korisno je pratiti "sirove" podatke (eng. raw data). Na slici 6 su prikazani podaci u JSON (JavaScript Object Notation) formatu, koji se ispisuju u konzoli internet pretraživača, koji su primljeni sa hardvera.



Sl. 6. Primer prikaza podataka u JSON formatu, u konzoli web pretraživača.

Još jedna programerska praksa koja je primenjena, je davanje smislenih naziva promenljivim i njihovo adekvatno grupisanje. Vremenska oznaka 1610439062 prikazuje koliko je sekundi prošlo od 1. januara 1970. godine i predstavlja 12. januar 2021. godine u 08:11:02.

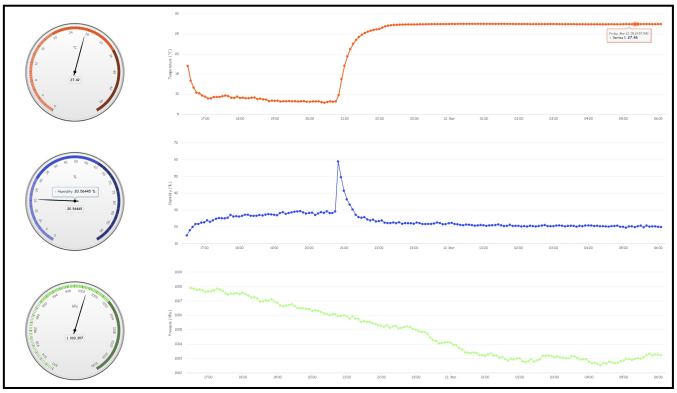
X. Biznis analiza

Mladi inženjeri umeju često da pomešaju biznis analizu sa biznis planom. Biznis analiza nije biznis plan. Biznis analiza je oblast koja predstavlja integralni deo kreiranja bilo kakvog uređaja ili proizvoda, koji će krajnji korisnik koristiti.

Podjednako je kompleksno kao programiranje i projektovanje elektronskih kola, a biznis analitičari najčešće predstavljaju vezu (eng. link) između inženjera i krajnjih korisnika. Na neki način predstavljaju prevodioce, koji zajedno sa korisnicima kreiraju nacrte sistema, a zatim uz maksimalan napor programerima objašnjavaju šta je to što korisnik želi da dobije. Često su biznis analitičari eksperti u njihovim profesijama, koje ne moraju da budu inženjerske.

XI. Regularan web interfejs

Na slici 7 je prikazan web interfejs na kom se vidi znatna promena temperature, a po očitavanju vlažnosti vazduha, na srednjem delu slike se može primetiti da je uređaj premešten iz hladnije prostorije u topliju.



Sl. 7. Web interfejs na kom se prikazuje znatna promena temperature.

Tri vrednosti koje se prikazuju se dobijaju sa istog senzora, Bosch BME280 i nema smisla slati 3 zahteva kada se proverava trenutna vrednost, već je dovoljno to uraditi jednom.

Zahvalnica

Rad su podržali Centar za metrologiju, Fakulteta tehničkih nauka i projekti KALCEA "Knowledge Triangle for a Low Carbon Economy" 618109-EPP-1-2020-1-EL-EPPKA2-CBHE-JP i "Razvoj naučno-stručnih metoda u oblasti metrologije, industrijsko-tehničkih merenja u digitalnom konceptu Industrije 4.0, biomedicinskih mernih sistema i kognitivnih neuronauka primenom napredne metodologije i digitalne tehnologije". Svako pitanje koje student postavi, otvara niz novih mogućnosti i dobar deo ovog rada je nastao upravo kao odgovor na najčešće postavljana pitanja.

Zaključak

Ovaj rad je namenjen svim ambicioznim ljudima, koji ulaze u svet kreiranja uređaja, kao i u programiranje. Takođe, namenjen je i iskusnim programerima i hobistima. Ovaj rad je dokaz koncepta, da je uz entuzijazam moguće krenuti sa izradom industrijskog uređaja, po ceni nižoj od 10$. Ove činjenice omogućavaju korišćenje ovog hardvera i u zemljama u razvoju. Po mišljenju autora, Highcharts se vremenom nametnuo kao vredna zamena Chart.js-u [19], a da je Apache ECharts [20] nešto što bi moglo u skorijoj budućnosti da postane zanimljivo.

Literatura

[1] https://docs.espressif.com/projects/esp-idf/en/latest/esp32/
[2] https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_now.html
[3] https://www.arduino.cc/en/software
[4] https://dl.espressif.com/dl/package_esp32_index.json
[5] https://github.com/me-no-dev/arduino-esp32fs-plugin/releases/
[6] https://github.com/jeffThompson/DarkArduinoTheme
[7] https://github.com/espressif/arduino-esp32
[8] https://learn.sparkfun.com/tutorials/how-to-install-ch340-drivers/all
[9] https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers
[10] https://eu.mouser.com/ProductDetail/Espressif-Systems/ESP32-DevKitC-32D?qs=%252BEew9%252B0nqrDsObWEpDx6YQ==
[11] https://eu.mouser.com/ProductDetail/Espressif-Systems/ESP32-DevKitC-32U/?qs=%252BEew9%252B0nqrCEVvpkdH%2FG5Q%3D%3D
[12] https://www.aliexpress.com/item/4001136108709.html?spm=a2g0s.9042311.0.0.7a9b4c4dMRyFWh
[13] https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/
[14] https://github.com/fhessel/esp32_https_server
[15] https://www.highcharts.com/
[16] https://www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/
[17] https://www.maximintegrated.com/en/products/analog/real-time-clocks/DS3231.html
[18] https://www.pool.ntp.org/zone/rs
[19] https://www.chartjs.org/
[20] https://echarts.apache.org/en/index.html

Abstract

Abstract: Prior to IIoT (The Industrial Internet of Things), embedded programming and frontend development didn't even found themselves in the same sentence. Hardware that enables seamless integration of these two vast areas is the Espressif ESP32 MCU (MicroController Unit). Once numerous devices were connected to the internet, the security of data became the focal point. ESP-NOW technology is Espressif's proprietary solution for wireless data transfer. It can be encrypted and it enables secure communication between multiple ESP32's. HTTPS (Hypertext Transfer Protocol Secure) is a protocol that increases security on the internet. This article provides a guide for configuring a development environment for ESP32, ESP-NOW example, HTTPS server example, and programming practice intended for error handling. In addition, it offers a responsive example of a graphical user interface of the IIoT device. At the moment, there are numerous possible paths in embedded and frontend programming. This paper follows the path using ESP32 as the hardware, and C++ for the firmware. JavaScript, HTML5, and CSS3 are unavoidable parts of modern industrial devices. Hence, the article provides an example of using JavaScript Highcharts library. This combination of hardware and software is the solution that costs less than 10$, which makes it acceptable in countries under development. Highcharts library is proprietary, and for educational purposes, it is used under Creative Commons (CC) Attribution-Non-Commercial license.

An educational approach to an encrypted data transfer in an embedded and frontend development environment

Ivan Gutai, Platon Sovilj, Marina Subotin, Marjan Urekar, Jelena Milojević, Milica Mitrović