PD.TVZ.HR
POLYTECHNIC & DESIGN Vol. 9, No. 3, 2021.
Mrežna inačica: ISSN 2459-6302
Tiskana inačica: ISSN 1849-1995
DOI: 10.19279/TVZ.PD.2021-9-3-08

"ŽURNA IZRADA MOBILNIH APLIKACIJA ZA NADZOR I UPRAVLJANJE U INDUSTRIJI"

"RAPID PROTOTYPING OF MOBILE APPLICATIONS FOR INDUSTRIAL CONTROL AND MONITORING"

Krunoslav Martinčić[1], Filip Marolt[2], Sebastijan Martinčić[2]

[1]Tehničko veleučilište u Zagrebu
[2]Tehničko veleučilište u Zagrebu, student

SAŽETAK

U ovom radu prikazan je jedan od mogućih pristupa žurnom razvoju platformi za nadzor, upravljanje i parametriziranje uređaja ili sustava pomoću mobilnog telefona. Inicijalni prijedlog arhitekture, razvojnog okruženja, obiju aplikacija i sklopovsko rješenje ciljnog sustava dani su u Završnom radu studenta Elektrotehničkog odjela Tehničkog veleučilišta u Zagrebu. Za realizaciju projekta korišteni su Android uređaj i modul mikroupravljača s mogućnošću komunikacije putem Wi-Fi protokola. Izrađen je prototip i ispitana je njegova funkcionalnost. Predložena su buduća poboljšanja u odnosu na zaštitu od mogućih nepravilnosti ili otkaza u radu pojedinih komponenti sustava u industrijskom okruženju.

Ključne riječi: *Android uređaj, mobilna aplikacija, žurna izrada, industrijsko okruženje*

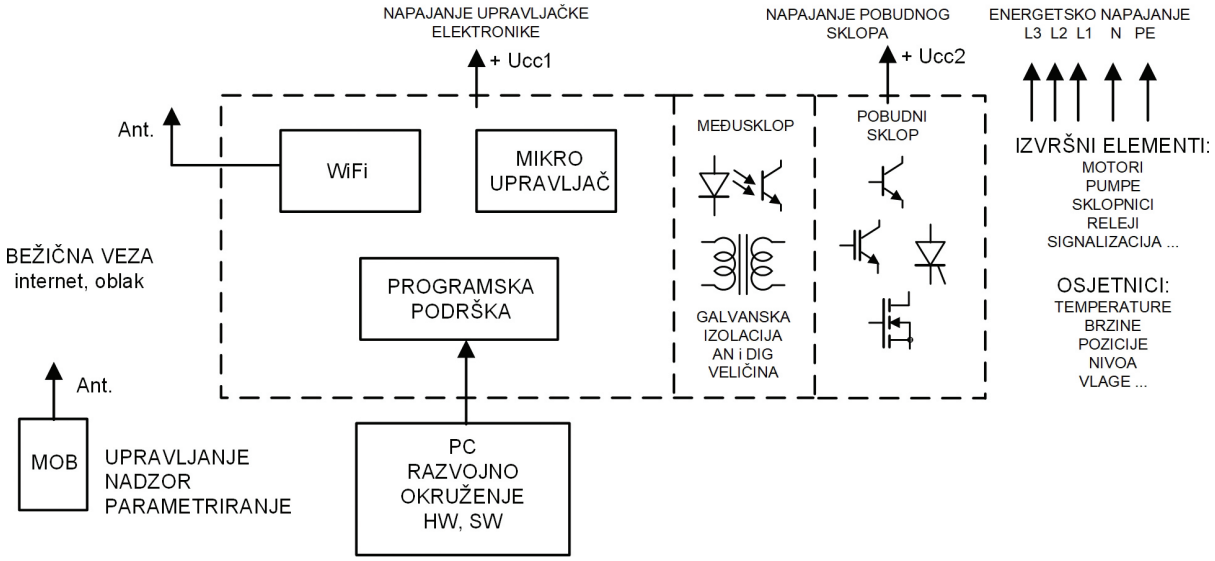
ABSTRACT

This paper presents one of the possible approaches to the rapid development of a platform for monitoring, control and management of devices or systems using mobile phones. The initial proposal of architecture, development environment, both applications and hardware realization of the target system were made in the Undergraduate Thesis at a Electrical Engineering dept., Zagreb University of Applied Sciences. Android phone and microcontroller board with Wi-Fi capabilities were used to realize this project. The prototype has been physically realized and functionality was tested. Further improvements have been proposed regarding protection against possible irregularities or malfunctions in industrial environments.

Keywords: *Android phone, mobile application, rapid prototyping, industrial environment*

1. UVOD
1. INTRODUCTION

Nagli razvoj poluvodičke tehnologije krajem 90-tih godina prošlog stoljeća doveo je do pojave mikroračunala velikih procesnih moći. Paralelno s time razvijaju se alati za izradu programske podrške. Obje komponente postaju dostupne širokoj populaciji te nastaje situacija „tehnološkog supermarketa" [1]. S druge, ekonomske strane, po zakonima tržišta najveći profit ubire onaj proizvod koji se prvi pojavi na istom. Nastaje velika konkurencija među proizvođačima dobara a ključni je cilj pojaviti se prvi ili barem među prvima na tržištu (eng. slogan *"Time to Market"*). Takva situacija uvjetuje i omogućava razvoj moćnih programskih paketa, alata za projektiranje i simulacije procesa u svim područjima ljudske djelatnosti. U iste su ugrađeni sofisticirani algoritmi i znanja stjecana desetljećima što bitno skraćuje proces nastanka i pojave proizvoda na tržištu (eng. slogan *"Rapid Prototyping"*). Daljnji razvoj spomenutih tehnologija doveo je do pojave vizualnih programskih alata, mikroupravljačkih modula velikih procesnih snaga, lako dostupne dokumentacije, primjera i gotovih rješenja s javno dostupnim pripadajućim programskim bibliotekama u izvornom kodu.



Slika 1 *Arhitektura sustava*

Figure 1 *System architecture*

Sve to omogućava i niže kvalificiranim stručnjacima raditi na tehnološki zahtjevnim projektima. Današnji mobilni telefoni posjeduju veće procesne moći, a time i mogućnosti od osobnih računala starih nekoliko godina i masovno su u uporabi. Umjesto namjenskih perifernih jedinica ili osobnog računala te žičnog spajanja na sustave, nadzor, parametriziranje i upravljanje mobilnim telefonom zadnjih generacija uvelike pojednostavljuje spomenute procese. U ovom radu prikazano je kako u nekoliko sati ili u najgorem slučaju nekoliko dana izgraditi sustav daljinskog nadzora i upravljanja fizičkim procesom punjenja ili pražnjenja rezervoara tekućinom pomoću mobilnog telefona.

2. ARHITEKTURA SUSTAVA
2. SYSTEM ARCHITECTURE

Gruba arhitektura svih komponenata potrebnih u procesu izrade spomenutih sustava prikazana je na *Slici 1*.

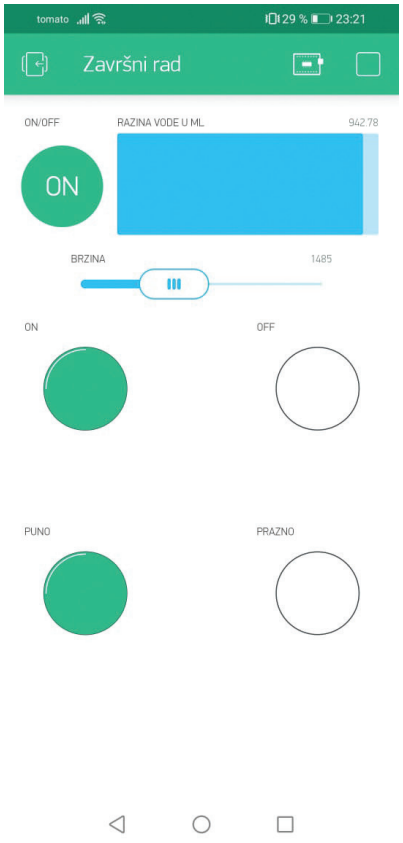
Razvoj programske podrške za mikroupravljač i sklopovlja ciljnog sustava rade se pomoću osobnog računala. Ovisno o pristupu, izrada mobilne aplikacije moguća je ili na osobnom računalu ili direktno pomoću grafičkog okruženja na samom mobilnom telefonu. Potrebno je odabrati prikladno razvojno okruženje (eng. IDE, *"Integrated Development Environment"*) za programsku podršku, elektroničke sklopove i module te po potrebi mehaničke konstrukcije. Razvoj programske podrške za mikroupravljač uvelike je olakšan i ubrzan jer je većina današnjih programabilna na modulu ciljnog sustava (eng. ISP, *„In System Programming"*).

Budući da govorimo o bežičnom upravljanju i žurnom razvoju u obzir dolaze samo standardni bežični komunikacijski sustavi i protokoli masovno u uporabi: Wi-Fi ili Bluetooth. Ovi sustavi rade u ISM (eng. *"Industrial, Scientific and Medical"*) radio opsezima na frekvencijama 2,45 GHz i 5,8 GHz. Uporaba ISM frekvencija regulirana je međunarodnim standardom preko ITU (eng. *"International Telecommunication Union"*) organizacije. ISM standard namijenjen je za bežičnu radio komunikaciju na male udaljenosti, malim snagama zračenja (eng. EIRP, *„Effective Isotropic Radiated Power"*). Ključno je da uporaba ovih frekvencija, uz poštivanje regulative, ne podliježe licenciranju (dobivanju dozvola za korištenje) od strane lokalnih regulatornih agencija (HAKOM u Hrvatskoj). Na tržištu se može po povoljnim cijenama pronaći puno modula s integriranim mikroupravljačem i Wi-Fi podsklopom što bitno smanjuje vrijeme razvoja ciljnog sustava.

Kod upravljanja industrijskim procesima izvršne jedinice (motori, releji, sklopnici, elektroventili i sl.) mogu biti velikih snaga i do stotine kW.

Uključivanje takvih trošila izvodi se pomoću pobudnog međusklopa (eng. *„Power Interface Driver"*) i energetskih elektroničkih komponenata: SSR (eng. *„Solid State Relay"*), IGBT (eng. *„Insulated Gate Bipolar Transistor"*), MOSFET (eng. *„Metal Oxide Field Effect Transistor"*) ili Thyristora. Radi zaštite od ispada iz rada procesno-komunikacijskog dijela elektronike a uslijed jakih elektromagnetskih smetnji i eventualnih naponskih proboja ugrađuje se galvanska izolacija (optička ili transformatorska veza) između procesnog i izvršnog dijela sklopovlja ciljnog sustava, *Slika 1*, [2].

Inicijalni algoritmi i arhitekture kreiraju se pod pretpostavkom da će svi elementi sustava ispravno funkcionirati. Pouzdanost sustava u radu ovisi o svim elementima sustava (programskih, sklopovskih i mehaničkih te vanjskim utjecajima: vlaga, temperatura, potres i sl.). Ispravna pretpostavka je da svaki element sustava može u potpunosti otkazati [2].



Slika 2 *Android aplikacija, Upravljanje procesom zalijevanja biljaka*

Figure 2 *Android Application, Plant watering process management*

Kod programske podrške potrebno je voditi računa o mogućim nepravilnostima izvođenja algoritma te pojave nedefiniranih stanja nakon kojih se sustav treba dovesti u definirano, stabilno ili isključeno stanje nakon kojeg je moguć oporavak (eng. *„Steady State"* ili *„Graceful exit"*). Kod mikroupravljača je udomaćena upotreba „psa čuvara" (eng. WDT, *„Watchdog Timer"*), sklopovskog nadzora mogućeg zaglavljivanja programskog algoritma izvođenja. Na isti je način i kod cjelokupnog sklopovlja ciljnog sustava, neovisno o procesnom dijelu, potrebno voditi računa o kritičnim situacijama i osigurati odlazak sustava u neraspirujuće, nedestruktivno, stabilno stanje. Primjer: Otkaz napajanja procesnog dijela (nestanak napajanja U_{CC1}, *Slika 1*, programski kod se ne izvodi) istovremeno uz proboj jedne od elektroničkih sklopki u vodljivo stanje rezultira nastavkom rada motora pumpe za tekućinu i poplavom kao ishodom. Takve situacije potrebno je predvidjeti, a zaštite od havarija izvesti na više nivoa pouzdanim elektromehaničkim rješenjima.

3. ODABIR RAZVOJNOG OKRUŽENJA I KOMPONENATA SUSTAVA

3. SELECTION OF DEVELOPMENT ENVIRONMENT AND SYSTEM COMPONENTS

3.1. REALNI PROJEKT

3.1. REAL PROJECT

Ideja za ovaj projekt nastala je u okviru izrade Završnog rada na Elektrotehničkom odjelu Tehničkog veleučilišta u Zagrebu [3]. Cilj je napraviti sustav u kojem se pomoću mobilnog uređaja upravlja i nadzire proces zalijevanja biljaka. Zalijeva se iz spremnika konačnog kapaciteta pomoću električne pumpe. Mobilnim uređajem proces se svakog trenutka može pokrenuti ili zaustaviti, upravlja se brzinom protoka vode i prati razina vode u spremniku. Korisničko sučelje Android aplikacije prikazano je na *Slici 2*. Desno je gore prikaz razine vode u spremniku, a ispod su: klizač za upravljanje brzinom protoka vode, tipke za uključivanje i isključivanje pumpe te signalizacija „puno/prazno".

3.2. ANDROID UREĐAJ
3.2. ANDROID PHONE

Sklopovska je jezgra današnjih mobilnih uređaja mikroračunalo velike procesne moći što omogućava izvršavanje zahtjevnih aplikacija s grafičkim korisničkim sučeljem. Za način i dinamiku raspodjele poslova potrebnih pri izvršavanju pojedinih aplikacija na centralnu procesnu jedinicu i periferije brine se operacijski sustav. U upotrebi su: Googleov Android, Appleov iOS i razne distribucije Linuxa za uređaj PinePhone i projekti otvorenog koda (eng. *„Open Source"*). Najrašireniji je operacijski sustav za mobilne platforme je Android. Koristi ga većina proizvođača dok se iOS koristi samo na Appleovim uređajima. Temeljem rečenog odabire se Android uređaj.

3.3. WEB POSLUŽITELJ
3.3. WEB SERVER

Upravljanje i nadzor fizičkih procesa Android uređajem moguće je izvesti putem WEB preglednika tako da se u ciljni sustav ugradi HTML WEB poslužitelj, a komunikacija se ostvaruje standardnim WEB preglednikom. Posebna aplikacija na Android uređaju nije potrebna. Ovaj je pristup programerski zahtjevan, a na strani ciljnog sustava traži dodatne resurse po pitanju procesne moći mikroupravljača i vanjske radne memorije.

3.4. ANDROID STUDIO
3.4. ANDROID STUDIO

Aplikaciju za Android operacijski sustav (OS) moguće je napisati na nekoliko načina. Aplikacija opisana u ovom radu treba komunicirati s perifernom izvršnom jedinicom. Moguće ju je izraditi korištenjem *Android studio* razvojnog alata namijenjenog isključivo za razvoj aplikacija pod Android OS-om. Za njegovo korištenje potrebne su napredne programerske vještine objektnog programiranja i poznavanje programskog jezika Java što ga čini neprikladnim za široku uporabu.

3.5. BLYNK
3.5. BLYNK

Posljednjih godina naglo raste pojava povezivanja i međusobne komunikacije raznih vrsta uređaja putem interneta (IoT, eng. *„Internet of Things"*). Ta je pojava uzrokovala nastanak nove vrste alata za *žurnu* izradu aplikacija gdje mobilni uređaj postaje ključna pristupna točka čovjeka prema IoT napravama. Dva alata tog tipa su komercijalni Blynk [4] i Supla [5] otvorenog koda. Oba alata sastoje se od skupa unaprijed definiranih objekata i procedura (API, eng. *„Application Programming Interface"*). Aplikacija se gradi međusobnim povezivanjem grafičkih prezentacija objekata (ikona) putem grafičkog sučelja na samom mobilnom uređaju postupkom „povuci i stavi" (eng. *„drag and drop"*). Relativno mali zaslon mobilnog uređaja sam po sebi ne omogućava istovremeni prikaz puno objekata u visokoj rezoluciji (ulaznih i izlaznih veličina prema ciljnom IoT sustavu). Proizlazi da je jednostavnu aplikaciju moguće kreirati u nekoliko minuta. Odabiremo ovaj pristup (Blynk) koji, kao što ćemo kasnije vidjeti, ima svojih mana, ali daje brzo rješenje.

3.6. MIKROUPRAVLJAČ
3.6. MICROCONTROLLER

Na tržištu se mogu povoljno dobaviti mikroupravljački moduli s mogućnošću komunikacije putem Wi-Fi-ja raznih dobavljača: Raspberry Pi Zero W (BroadCom SoC, eng. *„System on Chip"*, procesor s ARM-ovom jezgrom), NodeMCU Devkit (otvorena platforma, Espressif Systems SoC, procesor baziran na Tensilica IP jezgri, Cadence Design Systems), WeMos LOLIN-32 (slično kao NodeMCU) i Microchip WFI32 (baziran na PIC mikroupravljačima). Odabran je WeMos LOLIN-32 modul baziran na Tensilica ESP32 SoC-u jer je razvojnu programsku podršku za njega (eng. *„Compiler, Linker, Assembler, Loader"*) moguće integrirati u široko rasprostranjenu IDE platformu Arduino.

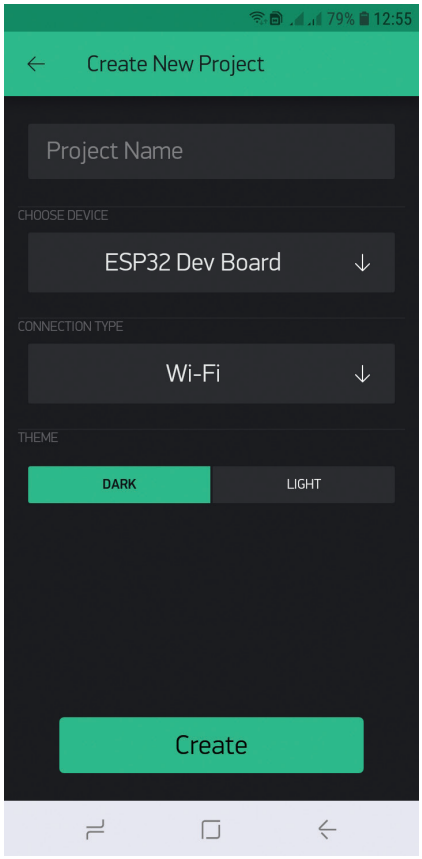
3.7. ARDUINO IDE
3.7. ARDUINO IDE

Za razvoj programske podrške mikroupravljača ciljnog sustava i njegovo punjenje binarnim strojnim kodom potrebno je imati na raspolaganju odgovarajuće sofisticirane alate: jezični prevoditelj (*„kompajler“*), povezivač programskih modula nastalih prevođenjem (*„linker“*), generator binarnog strojnog koda (*„assembler“*), simulator izvođenja programa, optimizator koda, programator i inicijalno sklopovlje (demo ili proto pločicu, eng. *„Board“*). Svaki od spomenutih programskih alata može imati nekoliko ulaznih parametara (opcija) kojima se prilagođavaju konkretnim aplikacijama: optimizaciji po brzini izvođenja, minimalnom zauzeću memorije, minimalnoj potrošnji energije i sl. Rad s tim alatima zahtjeva dobro poznavanje računalnog inženjerstva [6], [7], [8], [9]. Radi pojednostavljenja spomenutih procedura svi proizvođači mikroupravljača imaju u ponudi prikladno razvojno okruženje (IDE na PC-u) koje u jednom prolazu obavi sve prije spomenute radnje na temelju inicijalno postavljenih (eng. „default“) parametara. Korisnik eventualno treba podesiti par ključnih parametara: veličina memorije, radna frekvencija i sl. Već tijekom odabira modula i mikroupravljača treba voditi računa o dobavljivosti razvojnog okruženja za isti. Odabran je Arduino IDE jer je zamišljen tako da je u njega moguće integrirati spomenute alate raznih proizvođača (eng. *„Toolchain“*), a podrška za module na bazi ESP32 procesora javno je dostupna.

4. ANDROID APLIKACIJA
4. ANDROID APPLICATION

Za izradu Android aplikacije korišten je alat *“Blynk”* [4]. Instalacija tog alata na Android uređaj provodi se putem Play Store aplikacije, a prije toga potrebno je otvoriti Googleov račun (eng. *„Account“*). Nakon pokretanja *Blynk* aplikacije prvi je korak registracija nakon čega se može pristupiti izradi vlastite aplikacije odnosno projekta. Sljedeći je korak kreiranje novog projekta uz odabir sklopovske platforme ciljnog sustava i njezinog načina komunikacije.

Blynk podržava preko 60 različitih platformi i većinu masovno korištenih komunikacijskih standarda (Ethernet, USB, WiFi, Bluetooth), *Slika 3*.



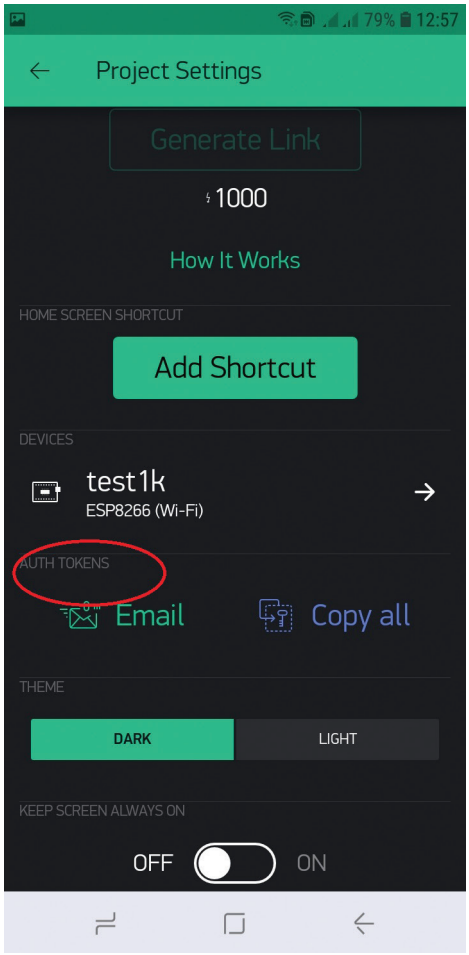
Slika 3 *Konfiguriranje projekta*

Figure 3 *Project configuration*

Nakon uspješne konfiguracije projekta *Blynk* aplikacija generira jedinstveni kod ovlaštenja za korištenje upravo kreirane aplikacije (eng. *„Authentication Code“*). Kod se automatski šalje na *e-mail*, a dostupan je i pod izbornikom *„Project Settings“* na Android uređaju, *Slika 4*. Moguće je besplatno korištenje *Blynka* uz ograničene mogućnosti (dvije platforme i pet korisnika). Za proširene mogućnosti potrebno je plaćati mjesečnu licencu.

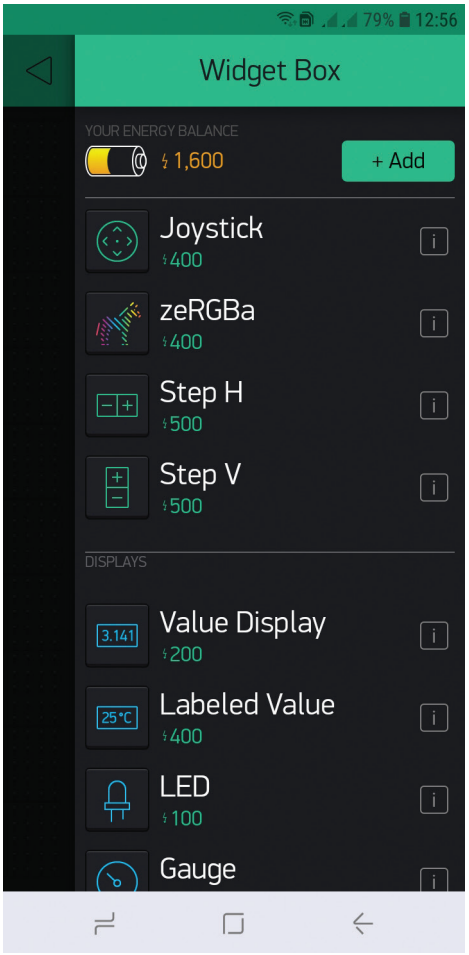
Kod ovlaštenja također je potrebno uklopiti u izvorni programski kod mikroupravljača ciljnog sustava u zaglavlju. Na istom mjestu treba dodati i podatke o lokalnoj WiFi mreži (SSID i PASS):

```
char auth[] = "YOUR_AUTH_CODE";

char ssid[] = "YOUR_SSID";

char pass[] = "YOUR_PWD";
```



Slika 4 *Jedinstveni autentikacijski kod aplikacije*

Figure 4 *Unique Application Authentication Code*



Slika 5 *Izbornik objekata*

Figure 5 *Widget box*

Sada je na Android uređaju moguće pristupiti izradi korisničke aplikacije odnosno projekta. Odabirom opcije „+“ poziva se izbornik s preko 40 unaprijed definiranih objekata za upravljanje fizičkim procesima (tipke, koračni, inkrementalni davači, linearni davači i sl.), prikaz aktualnih veličina i stanja procesa (numerički i grafički) te obavještavanja korisnika u zadanim ili kritičnim situacijama (*e-mail*, Twitter, …), *Slika 5*.

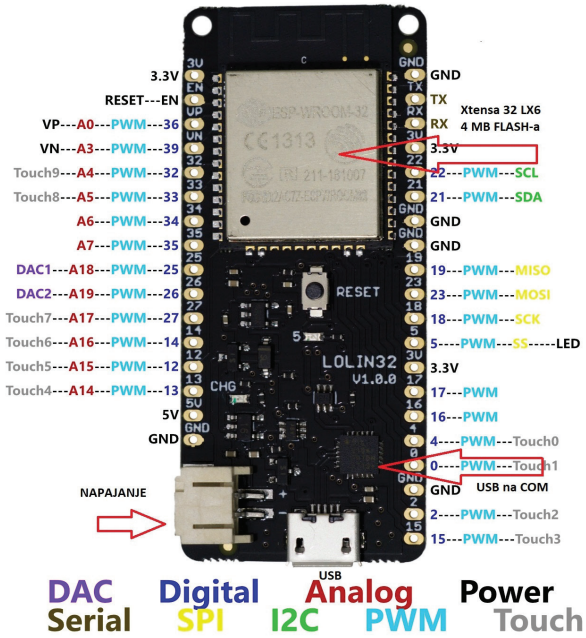
Jednostavnu aplikaciju s nekoliko objekata moguće je kreirati u par minuta postupcima: odaberi, stavi, povuci na poziciju i konfiguriraj. Cijelo vrijeme, od instalacije aplikacije *Blynk* na Android uređaj do kraja izrade korisničke aplikacije, nije potrebna uporaba osobnog računala. Za jednostavnije aplikacije poput ove, *Slika 2*, cijeli proces instalacije *Blynk*-a i izrade korisničke aplikacije s lijepim grafičkim sučeljem traje minimalno nekoliko desetaka minuta, a najviše par sati.

5. CILJNA IZVRŠNA JEDINICA
5. TARGET EXECUTIVE UNIT

Upravljačka jezgra na strani fizičkog sustava generički je modul *LOLIN-32* baziran na Tensilica, Espressif ESP32 mikroupravljaču za kojeg je javno dostupna dokumentacija i mnoštvo konkretnih primjera sklopovskih realizacija s pripadajućim izvornim kodom, *Slika 6*.

Specifikacije:

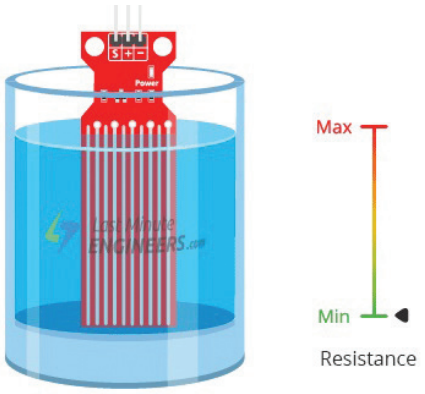
Dupla jezgra: Tensilica Xtensa 32-bit LX6
WiFi
Bluetooth (BLE 4.2 & BR/BDR)
512KB SRAM-a
ESP-WROOM-32 ima 4MB FLASH-a
LOLIN-32: USB na COM međusklop
Radna frekvencija: 80MHZ to 240MHZ
Radni napon: 2.3 V ~ 3.6 V



Slika 6 *LOLIN – 32 modul*

Figure 6 *LOLIN – 32 module*

Kao periferne naprave korišteni su otpornički osjetnik nivoa tekućine u spremniku, *Slika 7* i minijaturna potopna pumpa za vodu, *Slika 8*.



Slika 7 *Otpornički osjetnik nivoa tekućine*

Figure 7 *Resistive Fluid Level Sensor*

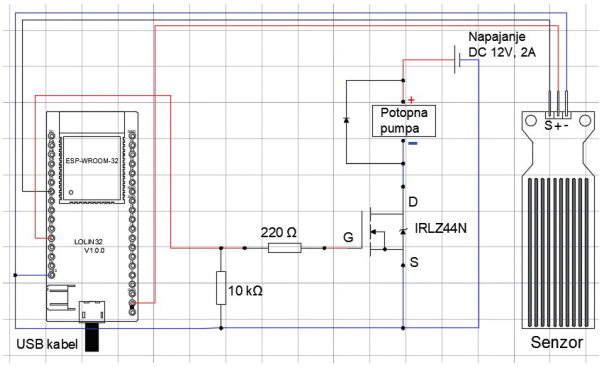
Slika 8 *Pumpa za vodu*

Figure 8 *Water Pump*

Signal s osjetnika nivoa tekućine spojen je na ulaz broj 34 LOLIN – 32 modula (A6) koji je programiran kao analogni ulaz. Internim A/D (eng. *"Analog to Digital"*) pretvaračem naponska veličina proporcionalna razini vode u spremniku pretvara se u digitalni zapis. Tijekom procesa razvoja potrebno je kalibrirati osjetnik tako da se jednoznačno pridijeli bezdimenzionalna numerička vrijednost iz A/D pretvarača stvarnoj količini tekućine u litrama. Važno je napomenuti da vrijednost dobivena ovakvim primitivnim osjetnikom jako ovisi o vrsti tekućine (ionizaciji) i temperaturi tekućine. Također, za lako zapaljive tekućine ovakav način detekcije nije primjeren. Treba koristiti ultrazvučne ili mikrovalne osjetnike na radarskom principu ili mehaničke davače pozicije bez električnog kontakta s istima.

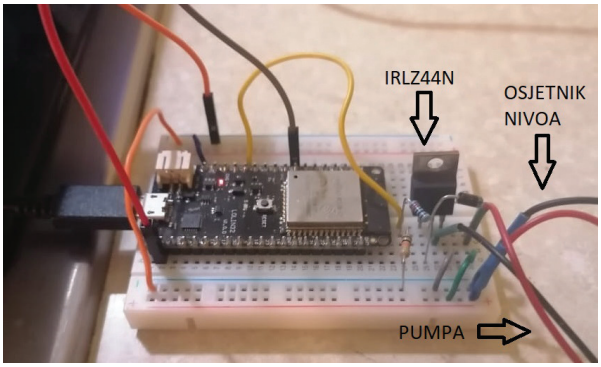
Minijaturna potopna pumpa za vodu, *Slika 8*, uključuje se preko kontakta broj 14 na LOLIN – 32 modulu koji se programira kao PWM (eng. *„Pulse Width Modulation"*, hrv. *„Pulsno širinska modulacija"*) upravljiv izlaz. Time se dobiva mogućnost upravljanja brzinom vrtnje motora pumpe što kao posljedicu ima regulaciju brzine protoka vode. Ova funkcija lako se implementira na bilo koju izvršnu jedinicu gdje je potrebno upravljati određenom veličinom u granicama od 0% do 100%. Najjednostavniji primjer, koji se često koristi kod inicijalnih provjera funkcionalnosti sklopovlja zajedno s programskom podrškom, promjena je intenziteta svjetla svjetleće diode (eng. *„Dimming"*, *„Light Emitting Diode"*, hrv. *„Svjetleća dioda"*) [10].

Električna shema prototipa prikazana je na *Slici 9*. Kao pobudna energetska elektronička komponenta za uključivanje motora pumpe korišten je obogaćeni N-MOSFET IRLZ44N. Otpornik iznosa 10 kΩ između nožice broj 14 LOLIN – 32 modula i referentne točke 0 V (mase) služi da u slučaju ispada procesnog dijela drži MOSFET, a time i pumpu u isključenom stanju. Dioda spojena paralelno s motorom pumpe štiti sklopovlje od pojave velikih napona kod isključenja pumpe, uslijed samoindukcije u namotajima motora. Ožičen prototip spojen na osobno računalo preko USB pristupa spreman za razvoj programske podrške prikazan je na *Slici 10*.



Slika 9 *Električna shema prototipa*

Figure 9 *Wiring diagram of the prototype*



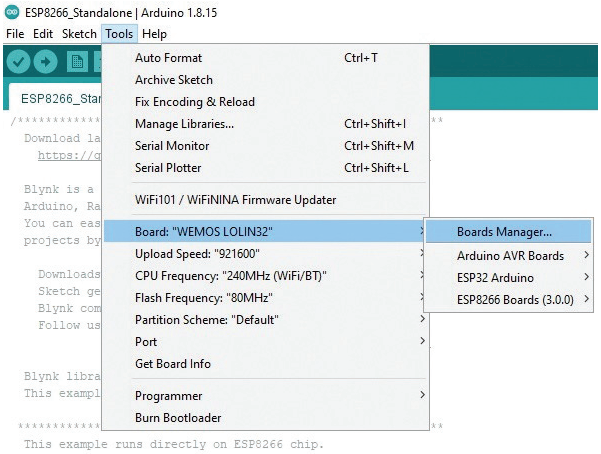
Slika 10 *Ožičen prototip*

Figure 10 *Wired Prototype*

6. ARDUINO IDE

6. ARDUINO IDE

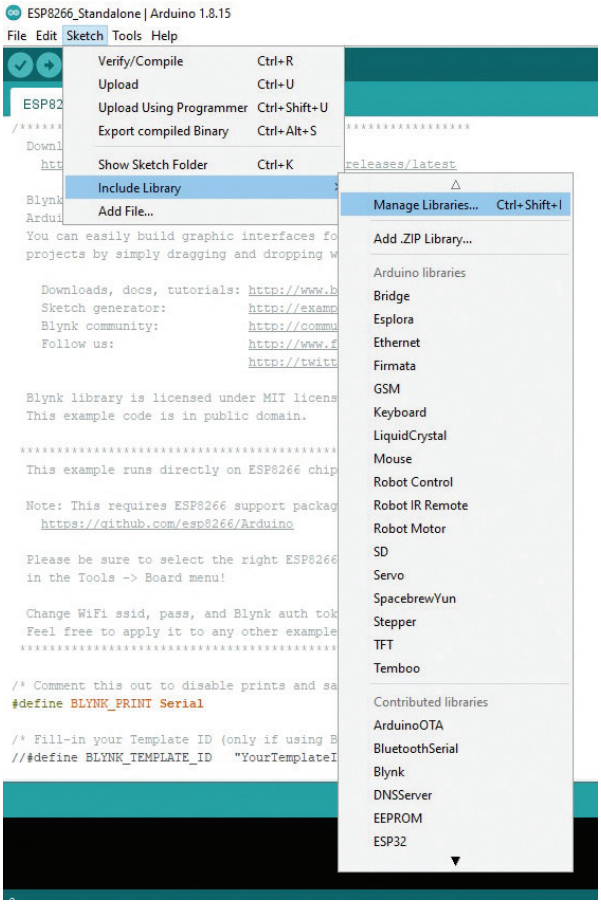
Pod pretpostavkom da je Arduino IDE instaliran, da bi ciljni sustav mogao komunicirati s aplikacijom na Android uređaju, potrebno je u IDE instalirati dodatne biblioteke. ESP32 modul nije originalno podržan u Arduino IDE-u. Instaliranje potpore za ESP32 modul radi se preko izbornika „*File*“ > „*Preferences*“ te se u polje „*Additional Boards Manager URLs*“ dodaje poveznica https://dl.espressif.com/dl/package_esp32_index.json**.** Nakon unesene prethodne poveznice iz izbornika ***„Tools“ > „Board“ > „Boards Manager“,*** *Slika 11*, pronađe se modul s ESP32 procesorom, odabere se verziju, instalira i podesi parametre za konkretni modul.



Slika 11 *Dodavanje podrške za ESP32 modul u Arduino IDE*

Figure 11 *Adding ESP32 module support to the Arduino IDE*

Nakon toga je u IDE potrebno dodati podršku za ***Blynk*** preko izbornika ***„Sketch“ > „Include Library“ > „Manage Libraries“,*** *Slika 12*. U otvorenom prozoru pronađe se ***Blynk*****,** odabere se verzija i doda se u Arduino IDE.



Slika 12 *Dodavanje podrške za Blynk u Arduino IDE*

Figure 12 *Adding Blynk support to the Arduino IDE*

Programski kod algoritma izvođenja relativno je jednostavan, a zaglavlje i inicijalizacija su:

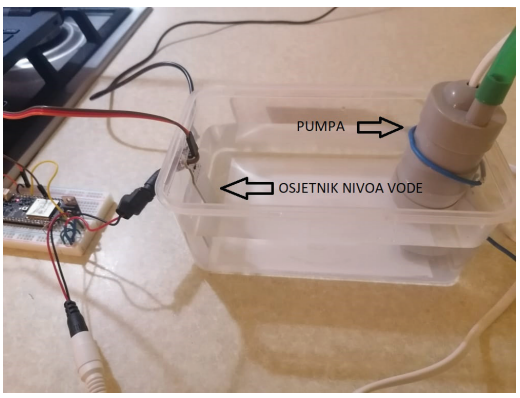
```
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
```

```
char auth[] = "dAF7iYEU--BR4g9-gI2ZTvejKHSqYFNE";
char ssid[] = "Vip WLAN_F9324F";
char pass[] = "BBCFCCFCEB";


WidgetLED prazno(V2); // signalizacija u Android aplikaciji, PRAZNO
WidgetLED puno(V3); // signalizacija u Android aplikaciji, PUNO
WidgetLED On(V6); // signalizacija u Android aplikaciji, UKLJUČENO
WidgetLED Off(V7); // signalizacija u Android aplikaciji, ISKLJUČENO


const int senzor = 34;  //razina vode, priključak br. 34
const int pumpa = 14;  //napajanje pumpe, priključak br. 14
const int napajanje = 2; //napajanje senzora, priključak br. 2
const int freq = 5000;  //pumpa, frekvencija PWM-a 5 kHz
const int rezolucija = 12;  // rezolucija PWM-a, 12 bitova


int razina = 0;  // razina vode, A/D binarna vrijednost
float razinaprava = 0.0;  //kalibrirana razina vode u ml
int brzina = 0;  // klizač u Android aplikaciji, brzina protoka vode
int onoff = 0;  // tipka UKLJ/ISKLJ u Android aplikaciji


BLYNK_WRITE(V5)      // tipka UKLJ/ISKLJ u Android aplikaciji
{
  onoff = param.asInt();
}
BLYNK_WRITE(V4)       // klizač brzine pumpe u Android aplikaciji
{
  brzina = param.asInt();
}
```

```
void setup() {
pinMode(senzor, INPUT);  // osjetnik nivoa, ULAZ u A/D pretvarač
pinMode(pumpa, OUTPUT);  // brzina pumpe, PWM, IZLAZ
pinMode(napajanje, OUTPUT);  // napajanje osjetnika nivoa vode u spremniku, IZLAZ
digitalWrite(napajanje, LOW); //početna vrijednost napajanja senzora, ISKLJUČENO
digitalWrite(pumpa, LOW);  //početna vrijednost brzine pumpe, ISKLJUČENO
ledcSetup(1, freq, rezolucija); // inicijalizacija PWM-a, brzina pumpe, kao „LED Dimmer“
ledcAttachPin(pumpa, 1); // pridjeljivanje PWM kanala
Serial.begin(9600); // inicijalizacija serijske komunikacije
Blynk.begin(auth, ssid, pass); // inicijalizacija Blynk komunikacije
    }
```

Poslije prevođenja, a prije spuštanja binarnog koda mikroupravljača na ciljni sustav potrebno je instalirati potrebne pomoćne programe te konfigurirati parametre COM pristupa osobnog računala zato što se „punjenje“ (eng. ***„Download“***) procesora radi preko serijske komunikacije (COM – USB – COM), ***Slika 11.***

Prototip je sagrađen i pušten u rad, *Slika 13*. Uz prosječne vještine u području elektronike i programiranja ukupno vrijeme potrebno za izradu ovakvog sustava procjenjuje se na par radnih dana što uključuje i proučavanje dostupne dokumentacije u pojedinim fazama izrade projekta.



Slika 13 *Sustav u radu*

Figure 13 *Working System*

7. MOGUĆA POBOLJŠANJA
7. POSSIBLE IMPROVEMENTS

Svaki projekt se može izvesti na više načina. Presudni parametri odlučivanja su potrebno vrijeme, cijena, potrebni kadrovski i materijalni resursi s ciljem optimalnog ishoda. U ovom projektu ključni parametar, po kojem je rađena optimizacija, bilo je potrebno vrijeme izrade. Temeljem rečenog u uvodnom dijelu moguća su poboljšanja galvanska (opto) izolacija pobudnog (MOSFET) i procesnog dijela sklopa, ugradnja visoko pouzdane elektromehaničke zaštite od zaglavljivanja sustava (plovkom i relejem isključiti napajanje pumpe ako je spremnik s vodom prazan) te ugradnja WDT-a u programski kod mikroupravljača. Daljnje poboljšanje bilo bi projektiranje i izrada vlastitog elektroničkog modula što zahtijeva sofisticirane alate i tehnologiju, a imalo bi smisla samo u slučaju serijske proizvodnje prikazanog sustava.

8. ZAKLJUČAK
8. CONCLUSION

U ovom radu prikazan je jedan od mogućih pristupa žurnom projektiranju industrijskih sustava nadziranih i upravljanih pomoću Android uređaja. Prednost ovog pristupa izuzetno je kratko vrijeme izrade Android aplikacije. Ključni nedostaci su licencirani alat za izradu Android aplikacije, potrebna je lokalna mreža i pristup internetu, izvršavanje aplikacije preko servera u oblaku što uzrokuje ovisnost o trećem subjektu pa je upitna profesionalna primjena. Prethodno rezultira sporim odzivima (veliku latenciju sustava) pa takav sustav nije primjenjiv za ultra brze procese.

Nakon krilatica *„Time to Market“* i *„Rapid prototyping“* od prije par desetljeća, kada se ova tehnologija počela naglo razvijati, danas se sve češće pojavljuje druga *„Good enough“*, što znači da u ovisnosti o primjeni proizvod treba biti dovoljno dobar odnosno upotrebiv.

9. REFERENCE
9. REFERENCES

[1.] K. Martinčić: „Elektroničke tehnologije - stanje i trendovi razvoja“, 3. SONT, Opatija/Zagreb, 1991.

[2.] K. Martinčić: "Pristup projektiranju specifičnih elektroničkih sustava"; Magistarski rad, FER, Zagreb, 1998.

[3.] F. Marolt: „Pumpa za vodu upravljana mobilnim uređajem“, Završni rad, Elektrotehnički odjel TVZ-a, Zagreb, 2021.

[4.] https://blynk.io/, 10. mj. 2021. god.

[5.] https://www.supla.org/en/, 10. mj. 2021. god.

[6.] S. Ribarić: „Arhitektura mikroprocesora“, 4-to dopunjeno izdanje, Tehnička knjiga Zagreb, 1990.

[7.] L. Budin: „Mikroračunala i mikroupravljači“, Element, Zagreb, 1997.

[8.] S. Ribarić: „Građa računala, Arhitektura i organizacija računarskih sustava“, Algebra, Zagreb, 2011.

[9.] S. Ribarić: „Arhitektura računala – RISC i CISC“, Školska knjiga, Zagreb, 1997.

[10.] https://community.blynk.cc/t/esp32-pwm-and-blynk/18277, 10. mj. 2021. god.

AUTORI · *AUTHORS*

● Krunoslav Martinčić
Rođen 29. prosinca 1962. god. u Zagrebu. Maturirao 1981. god. pri COIUO “Ruđer Bošković” na smjeru računarske tehnike, na ETF-u u Zagrebu diplomirao 1987. god. na smjeru računarska tehnika, a magistrirao 1998. god. na smjeru jezgra računarskih znanosti. Radio je kao asistent na matičnom fakultetu, a trenutno radi kao viši predavač na Elektrotehničkom odjelu Tehničkog veleučilišta u Zagrebu. Područja interesa su mu ugradbeni, namjenski računalni sustavi profesionalnih primjena te ultra brzi bežični komunikacijski sustavi.

Korespondencija · *Correspondence*
krunoslav.martincic@tvz.hr

● Filip Marolt
Rođen 4. veljače 1999. god. u Zagrebu. Maturirao je 2017. god. u Srednjoj strukovnoj školi-Samobor zvanjem tehničar za računarstvo. Iste godine upisao je studij na Tehničkom veleučilištu u Zagrebu, Prediplomski stručni studij elektrotehnike. Isti studij završio je 2021. god. stekavši zvanje prvostupnika inženjera elektrotehnike. Nastavlja studirati na TVZ-u na Diplomskom specijalističkom studiju elektrotehnike.

Korespondencija · *Correspondence*
filip.marolt@tvz.hr

● Sebastijan Martinčić
Rođen 6. srpnja 1999. god. u Zagrebu. Maturirao je 2018. god. u "V. gimnaziji". Iste godine upisao je studij na Fakultetu elektrotehnike i računarstva u Zagrebu, smjer računarstvo. Na studij Računarstva pri Tehničkom veleučilištu u Zagrebu prešao je 2021. godine.

Korespondencija · *Correspondence*
sebastijan.martincic@tvz.hr