

Univerzitet u Tuzli
Fakultet elektrotehnike
Automatika i robotika

ZAVRŠNI RAD
PRVOG CIKLUSA STUDIJA

**AUTOMATIZACIJA I MONITORING PAMETNIH KUĆA
KORIŠTENJEM FPGA KONTROLERA**

Mentor:

Dr.sc. Lejla Banjanović – Mehmedović, vanredni profesor

Student:

Nurdin Ibrišimović

TUZLA, JULI 2020. GODINE

JU UNIVERZITET U TUZLI
Fakultet elektrotehnike
Broj:05/3-4-37/20

Tuzla, 23.06.2020. god.

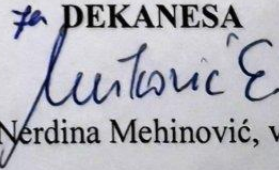
Ibrišimović Nurdin

Na osnovu Vašeg zahtjeva i izdate teme za predmet "Projektovanje sistema na čipu", izdaje Vam se slijedeći pismeni Završni zadatak:

**"Automatizacija i monitoring pametnih kuća korištenjem
FPGA kontrolera"**

Na osnovu Statuta Univerziteta u Tuzli, Završni rad možete braniti kada ispunite sve uslove predviđene Nastavnim planom i programom Prvog ciklusa studija, Statutom i drugim opštim aktima Univerziteta u Tuzli.

Završni rad potrebno je predati u sekretarijat Fakulteta u 5 (pet) primjeraka kucanih i uvezanih.


Dr.sci.Nurdina Mehinović, vanr.prof.

ZAHVALA

Veliku zahvalnost dugujem svojoj mentorici dr. sc. Lejli Banjanović – Mehmedović na prenesenom znanju i savjetima prilikom izrade završnog rada.

Zahvaljujem se mom djedu Ahmetu i neni Sabiri koji su mi pružili neizmjernu podršku tokom školovanja.

I na kraju, najveće zasluge za sve ono što sam postigao dugujem svojoj mami Almi, koja je bila izvor moje snage, moja najveća podrška i motivacija.

Sažetak

Zadatak ovog završnog rada jeste da predstavi proces automatizacije i monitoringa pametne kuće, uz korištenje savremenih tehnologija i dostupnih alata današnjice. Upotrebom mikrokontrolera, kreiranjem web servera kompatibilnog sa mobilnim uređajima i osobnim računarima, kao i realizacijom baze podataka izvršeno je upravljanje različitim elementima kuće, uz neprestanu mogućnost nadziranja uslova unutar i izvan kuće, uz korištenje grafičkog prikaza navedenih radnji.

U prvom poglavlju akcenat se stavlja na sistem rada i značaj pametne kuće, te njeni počeci i razvoj kroz historiju.

U drugom poglavlju su objašnjeni ugradbeni sistemi na kojima se bazira pametna kuća, ali i različiti tipovi sistema koji zajedno sa aktuatorima i senzorima čine jednu kompaktnu jedinicu sposobnu za pružanje maksimalne pogodnosti, efikasnosti i jednostavnosti za korisnika ovakve kuće.

Treće poglavlje objašnjava strukturu pametne kuće, te način na koji je izgrađena. U ovom poglavlju potenciram značaj senzorike i aktuatora koji igraju važnu ulogu u kreiranju sistema pametne kuće, a kojem je osnovna svrha olakšavanje svakodnevnih aktivnosti, te stvaranje ugodnog ambijenta za život, zbog čega će ovakav projekat gradnje, s obzirom na brzi razvoj tehnologija, biti sve više populariziran.

U četvrtom poglavlju prikazana je praktična realizacija pametne kuće uz pomoć makete. Praktična realizacija predstavlja finalni korak u kojem metodom demonstracije objašnjavam rad pametne kuće i njenu kompleksnost. Princip na kojem funkcioniše pametna kuća sastoji se od različitih tehnologija i sistema koji formiraju jednu kompleksnu i povezanu cjelinu a koja ima za svrhu jednostavnije upravljanje, kontrolu i nadgledanje okruženja u kojem se korisnik nalazi unutar i izvan pametne kuće.

Peto poglavlje prikazuje fuzzy koncept upravljanja, realizacija fuzzy kontrolera na praktičnom primjeru te upotreba fuzzy logike u okviru projekata pametne kuće.

U šestom poglavlju je opisana softverska implementacija korištenjem programskih jezika poput HTML, CSS, C++, Javascript, Verilog, PHP i MySQL. Ovaj dio opisuje kreiranje web servera, komunikaciju između Arduino i FPGA kontrolera kao i optimizaciju web stranice za upotrebu mobilnim i personalnim računarima.

Implementacija hardvera i softvera je izvršena kombinacijom nekoliko tehnologija i alata, od Arduina do FPGA razvojnih ploča, preko html, css, javascript, php i ostalih programskih jezika.

Rezultati praktičnog pristupa problemu su detaljno analizirani i objašnjeni.

Ključne riječi: FPGA, Arduino, Pametna kuća, monitoring, automatizacija, fuzzy upravljanje.

Abstract

The problem of this bachelor thesis is presenting the process of automation and control by using modern and impressive tools of today. By implementing microcontroller, creating web server and using the database, controlling different elements of smart house by either desktop or mobile device was made easy by allowing constant monitoring of temperature and humidity in and out the house with visual representation.

In the first chapter, the history of smart house was explained, how the market for it rose as the demand for new technologies grew.

Second chapter explains the importance of embedded systems in implementing different elements while creating a complex system such as smart house itself by using various sensors and actuators in order to achieve unique sense of comfort, efficiency and convenience.

Third chapter explains the structure of the smart house, what tools, technologies and equipment are required to implement a vision of a home that works in your advantage and improves your life for the better.

Fourth chapter showcases a practical realisation of a smart home while implementing previously learned knowledge about different embedded systems, technologies, sensors and actuators and balancing it all in order to create a functional house that has an ability to control and monitor your environment.

Fifth chapter is a representation of a fuzzy concept, fuzzy controller and its usage on the practical example.

Sixth chapter contains software implementation using various programming languages such as HTML, CSS, C++, Javascript, Verilog, PHP and MySQL. This part describes how web server was created, the communication between Arduino and FPGA as well as optimisation of the web page for both mobile and desktop usage.

An implementation of both hardware and software was possible by using different tools such as arduino and fpga development boards while also focusing on various programming languages such as html, css, javascript, php and many others.

Results of this practical approach are analyzed and explained in the chapters to come.

Keywords: FPGA, Arduino, Smart House, Monitoring, Automation, Fuzzy control.

Sadržaj

| | | |
|-------|--|----|
| 1 | Pametne kuće – historija, sadašnjost i budućnost | 12 |
| 1.1 | Uvod..... | 12 |
| 1.2 | Pametne kuće - kako je sve počelo?..... | 12 |
| 1.3 | Pametne kuće u 21. stoljeću | 14 |
| 1.4 | Budućnost pametne kuće..... | 15 |
| 2 | Ugradbeni sistemi..... | 17 |
| 2.1 | Tipovi ugradbenih sistema | 17 |
| 2.2 | Karakteristike ugradbenih sistema | 19 |
| 2.3 | Trend razvoja digitalnih sistema..... | 20 |
| 2.4 | Sistemi na čipu..... | 21 |
| 2.4.1 | Komponente sistema na čipu..... | 21 |
| 2.4.2 | Grativni blokovi sistema na čipu | 22 |
| 2.5 | Sistemi na čipu sa ARM Cortex-M procesorima..... | 23 |
| 2.6 | Različiti tipovi ARM procesora..... | 24 |
| 2.7 | Cortex-M i FPGA | 25 |
| 2.8 | FPGA – Field Programmable Logic Array | 26 |
| 2.8.1 | Interna arhitektura FPGA | 27 |
| 2.9 | Razlika između FPGA i mikrokontrolera..... | 28 |
| 3 | Struktura Smart Home sistema | 30 |
| 3.1 | Uvod – opis strukture pametne kuće..... | 30 |
| 3.2 | Arhitektura pametne kuće | 30 |
| 3.2.1 | Usluge pametne kuće | 30 |
| 3.2.2 | Osnovne komponente pametne kuće | 31 |
| 3.2.3 | Pametna kuća i Internet of Things | 32 |
| 3.3 | Napredna pametna kuća | 33 |
| 4 | Hardverska implementacija pametne kuće | 35 |
| 4.1.1 | FPGA Cyclone IV EP4CE6 | 38 |
| 4.1.2 | ESP32 DevKit..... | 39 |
| 4.1.3 | DHT 11 i DHT22 senzori..... | 40 |
| 4.1.4 | PIR senzori | 41 |
| 4.1.5 | LED diode | 41 |
| 4.1.6 | L9110 ventilatori..... | 42 |
| 4.1.7 | Fen – grijač | 42 |

| | | |
|-------|---|----|
| 5 | Fuzzy koncept upravljanja | 43 |
| 5.1 | Fuzzy sistem zaključivanja – fuzzy regulator..... | 43 |
| 5.1.1 | Fuzzy kontroleri..... | 45 |
| 6 | Softverska implementacija..... | 52 |
| 6.1.1 | Upravljanje Arduino ESP32 uređajem..... | 52 |
| 6.1.2 | Upravljanje FPGA Cyclone IV uređajem | 55 |
| 6.1.3 | Arduino i FPGA komunikacija..... | 55 |
| 6.2 | Navigacija web stranicom pametne kuće | 57 |
| 6.3 | Monitoring pametne kuće..... | 60 |
| 7 | Zaključak | 64 |
| 8 | Literatura..... | 65 |

Popis slika

| | |
|---|----|
| Slika 1: Smart Home aplikacija | 12 |
| Slika 2: 'The Kitchen' računar - H316..... | 13 |
| Slika 3: X10 pametna kuća..... | 13 |
| Slika 4: Pametna kuća - sistem | 15 |
| Slika 5: Elementi pametne kuće..... | 16 |
| Slika 6: Ugradbeni sistemi..... | 17 |
| Slika 7: Tipovi ugradbenih sistema..... | 18 |
| Slika 8: Trend razvoja digitalnih sistema | 20 |
| Slika 9: Sistem na čipu u mobilnim uređajima | 22 |
| Slika 10: Struktura sistema na čipu | 22 |
| Slika 11: Prije i poslije ugradnje SoC na FPGA uređaje..... | 23 |
| Slika 12: Inovativnost ARM procesora i FPGA uređaja..... | 26 |
| Slika 13: Struktura FPGA razvojne ploče | 27 |
| Slika 14: Arhitektura FPGA razvojne ploče..... | 28 |
| Slika 15: Pametna kuća i implementacija konekcije različitih elemenata..... | 31 |
| Slika 16: Pametna kuća i IoT..... | 32 |
| Slika 17: Napredna pametna kuća..... | 33 |
| Slika 18: Sistem napredne pametne kuće | 34 |
| Slika 19: Praktična realizacija sistema pametne kuće..... | 35 |
| Slika 20: Maketa pametne kuće | 36 |
| Slika 21: Maketa pametne kuće | 36 |
| Slika 22: Postavka elemenata makete pametne kuće | 37 |
| Slika 23: FPGA Cyclone IV | 38 |
| Slika 24: ESP32 DevKit | 39 |
| Slika 25: DHT11 i DHT22 | 40 |
| Slika 26: PIR senzor..... | 41 |
| Slika 27: LED diode..... | 41 |
| Slika 28: L9110 ventilatori..... | 42 |
| Slika 29: Fen - grijač | 42 |
| Slika 30: Fuzzy logika..... | 43 |
| Slika 31: Arhitektura fuzzy logike | 44 |
| Slika 32: Osnovna konfiguracija Fuzzy sistema..... | 45 |
| Slika 33: Funkcija pripadnosti fuzzy kontrolera..... | 45 |
| Slika 34: Generalni primjer funkcija pripadnosti | 46 |
| Slika 35: Funkcije pripadnosti za izlaze sistema | 49 |
| Slika 36: Funkcije pripadnosti za sistem pametne kuće..... | 50 |
| Slika 37: Primjer1 - Odziv sistema na promjenu optimalne vrijednosti | 51 |
| Slika 38: Primjer2 - Odziv sistema na promjenu optimalne vrijednosti | 51 |
| Slika 39: ESP32 kao Web server | 53 |
| Slika 40: ESP32 i SPIFFS | 53 |
| Slika 41: Interakcija servera i klijenta..... | 54 |
| Slika 42: UART komunikacija | 55 |
| Slika 43: Smart Home web stranica na računaru | 57 |
| Slika 44: Mobilna verzija aplikacije | 57 |

| | |
|---|----|
| Slika 45: Pametna kuća - navigacioni dio | 58 |
| Slika 46: Bočna traka aplikacije pametne kuće na mobilnom uređaju | 59 |
| Slika 47: Centralni dio aplikacije..... | 60 |
| Slika 48: Smart Home – Data stranica | 61 |
| Slika 49: Interakcija servera sa bazom podataka | 62 |
| Slika 50: Pohrana podataka u tabelu | 63 |
| Slika 51: Smart Home - Monitoring stranica..... | 63 |

Popis tablica

| | |
|---|----|
| Tabela 1: Karakteristike ARM procesora..... | 25 |
| Tabela 2: Razlika između DHT11 i DHT22 senzora..... | 40 |
| Tabela 3: Lingvističke varijable sa lingvističkim vrijednostima za ulaze i izlaze sistema | 47 |
| Tabela 4: Temperatura – ulazna lingvistička varijabla | 47 |

Popis skraćenica

FPGA – field programmable gate array

LTE – long term evolution

IoT – internet of things

RFID – radio-frequency identification

WAP – wireless application protocol

RAM – read only memory

PROM – programmable read-only memory

EEPROM – electrically erasable programmable read-only memory

DSP – digital signal processing

RTOS – real time operative system

ASIC – application-specific integrated circuit

UART – universal asynchronous reciever - transmitter

PIR – passive infrared sensor

SPIFFS - serial peripheral interface flash file system

URL – uniform resource locator

HTTP – hypertext transfer protocol

CSS – cascading style sheets

1 Pametne kuće – historija, sadašnjost i budućnost

1.1 Uvod

Trenutno svjedočimo informatičkoj eri koja sve više uzima maha i aktuelizira zastupljenost elektronskih i informatičkih tehnologija, te automatizacije i robotizacije, a koje predstavljaju budućnost čovječanstva.

Mijenja se životna paradigma i način života zahvaljujući naglom razvoju elektronske i informatičke tehnologije, te automatizacije, kako na poslu, tako i privatnom životu. Zastupljenost informatičke i elektroničke tehnologije sve više se očituje u raznim segmentima, od automobila do tzv. pametnih kuća koje predstavljaju kompleksan spoj informatičkih tehnologija i elektronskih sistema kroz forsiranje automatizacije. Razvoj ovakvih sistema u osnovici treba da bude skalabilan, robustan, isplativ, energetski efikasan, te ekološki prihvatljiv. Pametne kuće predstavljaju značajan dio istraživačkog projekta u 21. stoljeću, zbog njihovog nacionalnog značaja po zdravlje, ekonomiju, infrastrukturu, sigurnost, efikasnost, ugodnost i udobnost života koju nude, te mnoge druge faktore [1].

Ključne tehnologije koje su odigrale ulogu u implementaciji i razvoju pametnih kuća u svakodnevnom životu čovjeka uključuju: internet, bežične mreže, sistemi poput Wi-Fi, Bluetooth, Zigbee, mreže senzora, pametni uređaji uključujući LTE, 3G, 4G, 5G mreže, obnovljiva energija, Internet of Things (IoTs), bežične senzorske mreže, pametni televizori, radio-frekventna identifikacija (RFID), baze podataka i analitika samo su neki od alata koji su učestvovali u implementaciji savremenih pametnih kuća [1].

1.2 Pametne kuće - kako je sve počelo?

Upravljanje svjetlima, korištenjem glasovnih efekata ili podešavanje temperature putem interneta je danas dio svakodnevnice no, međutim, prvi koraci pametnih kuća započinje mnogo prije [2].



Slika 1: Smart Home aplikacija

Zahvaljujući stalnom razvoju tehnologije pronalaze se novi načini da se tehnološka rješenja i otkrića implementiraju u svakodnevnom životu. Danas smo svjedoci da tehnologija naše svakodnevne zadatke i poslove čini jednostavnijim i lakšim. Upravo zbog toga, ideja o pametnim kućama nije ništa novo. Već desetljećima, nauka istražuje i proučava mogućnosti automatizacije kuća. Čak i 50-ih godina, autori poput Ray Bradburya su opisivali budućnost interaktivnih domova koji naizgled funkcionišu samostalno [2].

Ideja izgradnje pametnih kuća postoji već duži period, ali tek nedavno su ovakvi sistemi bili javno raspoloživi za upotrebu, kroz implementaciju ideja i tehnologija koje su bile potrebne za ovakav projekt. Nekoliko bitnih prekretnica koje su obilježile početke i razvoj pametnih sistema su:

- 1901-1920: Izum kućanskih aparata

Iako se izum kućanskih aparata ne može direktno povezati sa današnjim „pametnim uređajima“, u tom periodu su bili nevjerovatna senzacija. Sve je započelo sa motorizovanim usisivačem u 1901. godini da bi se zatim pojavili frižderi, pegle, sušilice, mikروvalne i slično [2].

- 1966: – 'the kitchen' računar



Slika 2: 'The Kitchen' računar - H316

'The Kitchen Computer' ili tzv. kuhinjski računar nije ugledao svjetlo dana, iako je bio prvi računar pametne kuće. Mogao je sastavljati liste za kupovinu, kontrolisati temperaturu te upravljati kućanskim aparatima iako je bio veličine kao i sama kuća [2].



Slika 3: X10 pametna kuća

- 1975: X10 Home Automation projekat

Pico Electronics su otkrili najbolji način iskorištavanja trenutne raspoložive snage i upravljanja svjetlima i manjim aparatima. Nekoliko dana kasnije, projekat je stupio na tržište pod X10 nazivom [2].

Pametne kuće nisu bile izrazito popularne 60-ih godina zbog visokih troškova, niskog nivoa umrežavanja te generalne želje za promovisanjem tehnologije umjesto pružanja pažnje korisniku i pristupačnosti korištenja pametne kuće. Zanimarivanje zadnje stavke je povezan sa 'efektom pepeljuge'. Osnovni razlozi zanemarivanja pristupačnosti kao ključnog faktora pametne kuće su:

- Nedostatak motivacije da se poveća produktivnost u kućanskim poslovima
- Manjak sudjelovanja korisnika tehnoloških alata u dizajnerskom procesu
- Stavovi dizajnera proizvoda koji ističu da su kućanske tehnologije neinteresantne
- Kontinuiran fokus na zasebne alate pri dizajniranju novih tehnologija

Iako djeluje da se situacija mijenja, problemi koji obuhvataju kreiranje interaktivnih tehnologija za kuću su specifičnog karaktera. Pametnu kuću nije nemoguće dizajnirati, ali predstavlja izrazito težak zadatak. Uprkos tome, velika je potražnja za pametnim kućama u današnjem vremenu što je i pokazatelj projekta 'Orange at Home' kao i postepeno povećanje komercijalnih i akademskih istraživanja u svijetu na ovu temu [3].

1.3 Pametne kuće u 21. stoljeću

Šta čini kuću „pametnom“ u 21. stoljeću? Ukoliko posmatramo Orange at Home projekat, 50 godina stara kuća je povezana na internet, te bazirana na serveru koji upravlja svim funkcijama iste. Svjetlost, grijanje, sigurnost, audio-vizuelni sistemi, zavjese, tuševi i mnoge druge aplikacije su kontrolisane putem WAP, SMS ili telefona korištenjem 'bežične' tehnologije. Tu su također i standardni računari koji su povezani na širokopojasnu mrežu kao i raznovrsne tehnologije koje omogućavaju na primjer praćenje zdravstvenog stanja [3].

Tehnologije koje su idealne za radnu atmosferu pokazale su se kao neuspješne u ovakvim kućama. Orange at Home je podržavala komandnu ploču ugrađenu na zidu za kontrolisanje svjetla i drugih funkcionalnosti, što je jedna od standardnih funkcionalnosti ureda. Mnogi korisnici ove kuće su ovakvu implementaciju smatrali komplikovanim pristupom ka izvršavanju relativno jednostavnih aktivnosti. Veliki broj korisnika izrazili su želju za upotrebom glasovnih komandi, pomoću kojih bi se upravljalo tehnologijama i sistemom unutar ove kuće. Orange kompanija je ovo aktivno istraživala te uprkos porastu popularnosti za ovakvim aplikacijama izrazito rasla, kreatori su se suočili sa problemima procesiranja velikog broja komandi kako bi dobili željenu reakciju [3].

Iako tehnologija pametnih kuća omogućava procesuiranje informacija, nedostaci se javljaju u mehaničkom aspektu kućnog života [3].

Inicijalni uzorci iz prijašnjih projekata i eksperimenata pokazuju da, ukoliko ovi uslovi nisu zadovoljeni, usvajanje pametnih tehnologija u kućama može biti znatno usporeno [3].

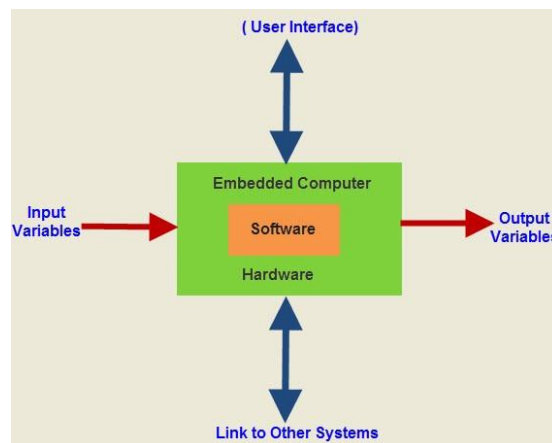


Slika 5: Elementi pametne kuće

2 Ugradbeni sistemi

Ugradbeni sistemi su „prikriveni računari“ sastavljeni od računara i procesorskog sistema koji nije programabilan ili u većini slučajeva nema vidljive sličnosti sa računarom. Softver je tipično razvijen od strane kreatora sistema, pripremljen ili kodiran u sistem tokom izrade, te nedostupan krajnjem korisniku. Ovakvi sistemi mogu biti pronađeni u kućama (mašina za pranje, DVD reproduktor, igraća konzola), u uredu (printer, automatizacija zgrade), automobilima (kontrola motora, ABS kočnice, alarmni sistem), avionima (navigacija, autopilot), u našim džepovima (mp3 reproduktor, mobilni uređaj) i čak i u našim novčanicima (elektronske kartice za autobus, kreditne kartice). Ekskluzivni ugradbeni sistemi poput igračih konzola i mobilnih uređaja imaju u nekim slučajevima vrhovnu procesorsku moć u odnosu na osobne računare. Također, raznovrsni mobilni dodaci i usluge akcenat su stavile na male cijene, veličine i potrošnje energije u ugradbenim komponentama [4].

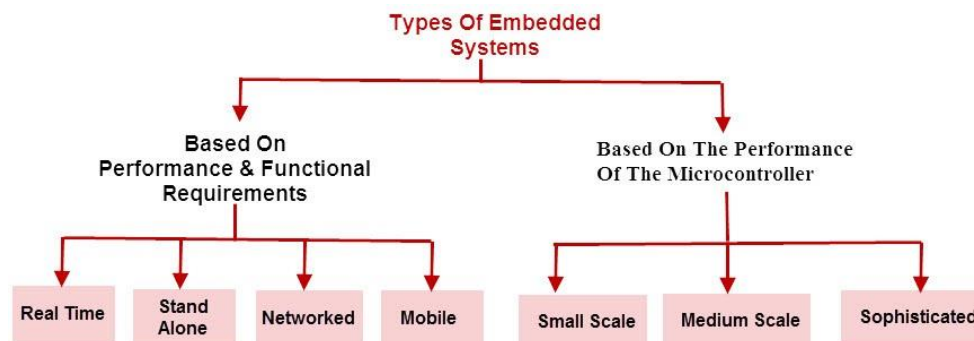
Ugradbeni sistemi koji su kombinacija hardvera, koji je izgrađen na bazi mikroprocesora, ili mikrokontrolera i softvera, uključuju elemente poput korisničkog sučelja, ulazno-izlaznih jedinica, ekrana i memorije. Generalno ugradbeni sistemi prilagođavaju napojnu jedinicu, procesor, memoriju, tajmere(programatore), portove za serijsku komunikaciju i aplikaciono specifične sklopove. Softver je kreiran korištenjem odgovarajućeg programskog jezika te kompajliran u cilju izvršavanja specifične funkcije unutar memorije u hardveru. Ovakvi sistemi su kreirani uz održavanje 2 limita: dostupnost systemske memorije i brzina procesora [5].



Slika 6: Ugradbeni sistemi

2.1 Tipovi ugradbenih sistema

Ugradbeni sistemi mogu biti klasificirani na osnovu performansi, funkcionalnih zahtjeva kao i performansi mikrokontrolera.



Slika 7: Tipovi ugradbenih sistema

Ugradbeni sistemi se klasificiraju u četiri kategorije na osnovu performanse i funkcionalnih zahtjeva:

- Nezavisni ugradbeni sistemi
- Real time ugradbeni sistemi
- Umreženi ugradbeni sistemi
- Mobilni ugradbeni sistemi

Na osnovu performansi mikrokontrolera, ugradbene sisteme možemo podijeliti na:

- Mali ugradbeni sistemi
- Srednji ugradbeni sistemi
- Sofisticirani ugradbeni sistemi

Nezavisni ugradbeni sistemi ne zahtijevaju glavni sistem poput računala. Koriste ulaz iz ulaznih priključaka digitalnog ili analognog procesa, izračunava i konvertuje signal te šalje rezultat preko povezanog uređaja koji kontroliše i prezentuje iznos na priključenom ekranu. Primjeri ovakvih sistema su MP3 reproduktori, digitalne kamere, mikrovalne, igraće konzole...

Real time ugradbeni sistemi su definisani kao sistemi koji daju željeni odnos ulaz/izlaz vrijednosti u odgovarajućem vremenu. Ovakvi sistemi prate specifične rokove za izvršavanje zadataka.

Umreženi ugradbeni sistemi su povezani sa mrežom u cilju preuzimanja resursa. Povezana mreža može biti LAN, WAN ili Internet. Konekcija može biti ožičena ili bežična (Wi-Fi). Ovakvi tipovi sistema su najbrže rastući sistemi u današnjim aplikacijama baziranim na ugradbenim sistemima. Ugradbeni web server je primjer sistema gdje su svi ugradbeni uređaji povezani na taj server koji je kontrolisan putem web pretraživača.

Mobilni ugradbeni sistemi su korišteni u portabilnim uređajima poput mobiteli, kamere, mp3 reproduktori i slično. Osnovna limitacija ovih uređaja je memorija.

Manji ugradbeni sistemi su dizajnirani sa 8-bitnim ili 16-bitnim mikrokontrolerom koji može biti aktiviran putem baterije. Za razvoj ovakvog sistema, osnovni alati su editor, assembler i integrisano razvojno okruženje.

Srednji ugradbeni sistemi su dizajnirani sa 16-bitnim ili 32-bitnim mikrokontrolerom, RICS ili DSP. Ovakvi sistemi nailaze na hardverske i softverske kompleksnosti. Za razvoj softvera, glavni programski alati su C, C++, JAVA, Real Time operativni sistem, simulator te integrisano razvojno okruženje.

Sofisticirani ugradbeni sistemi sadrže nevjerovatne nivoe kompleksnosti na hardverskom i softverskom nivou. Korišteni su u ekskluzivnim aplikacijama koje zahtijevaju uporedno dizajniranje hardvera i softvera u cilju produciranja konačnog rezultata [5].

2.2 Karakteristike ugradbenih sistema

Ugradbeni sistemi i aplikacije zahtijevaju veliki broj različitih procesora (mali, jeftini, sa malim napajanjem uz visoke performanse). Ključne karakteristike procesora su:

- Fleksibilnost – softverska izvedba definiše ponašanje ugradbenog sistema
- Efikasnost – energetska efikasnost, runtime efikasnost uz striktna vremenska ograničenja)

Karakteristike dizajna ugradbenih sistema:

1. Ugradbeni sistemi su namijenjeni da obavljaju specifične zadatke
2. Rad ugradbenih sistema podržan je od strane širokog dijapazona procesora i procesorskih struktura
3. Ugradbeni sistemi treba da su jeftini
4. Ugradbeni sistemi imaju ograničenja koja se odnose na rad u realnom vremenu:
 - a. Vremensko osjetljiva ograničenja
 - b. Vremensko kritična ograničenja
5. Upotreba RTOS operativnog sistema (eng. Real Time Operating System)
6. Implikacije softverskih grešaka su značajno ozbiljnije kod ugradbenih sistema u odnosu na desktop sisteme
7. Ugradbeni sistemi se u velikom broju slučajeva izrađuju kao baterijsko napajani uređaji
8. Ugradbeni sistemi podržavaju rad u ekstremnim ambijentnim uslovima
9. Ugradbeni sistemi imaju daleko manji broj ugrađenih sistemskih resursa u poređenju sa desktop sistemima
10. Ugradbeni sistemi čuvaju sav svoj objektni kod u ROM-u (eng. Read Only Memory)
11. Ugradbeni sistemi zahtijevaju korištenje specijalnih sredstava i metoda za projektovanje [6]

Ograničenja povezana sa embedded sistemima se odnose na:

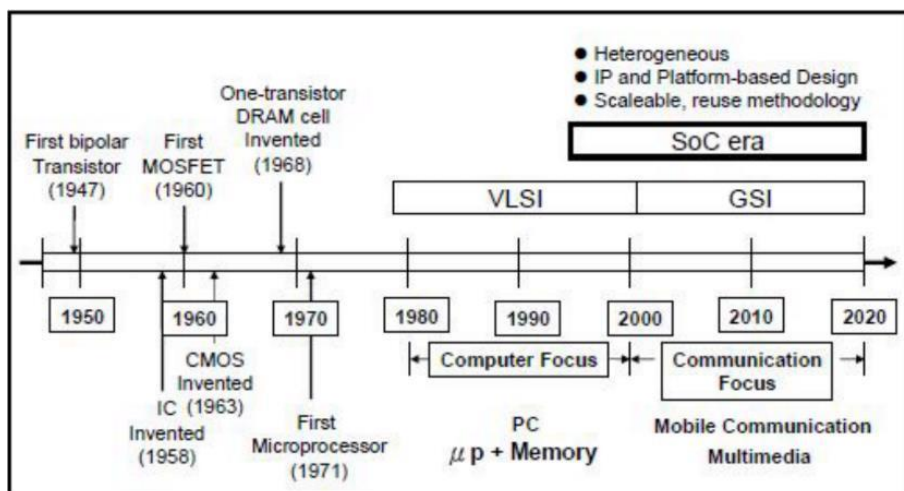
- Niske cijene
- Mala potrošnja energije
- Real time response (odgovor u realnom vremenu)
- Jednostavnost u programiranju
- Prenosivost koda
- Biblioteke koda
- Programerske alate
- Podrška za hardware/software koegzistenciju [6]

2.3 Trend razvoja digitalnih sistema

Digitalni sistemi su doživjeli nekoliko promjena u zadnje četiri dekade:

- Digitalni računari – sastavljeni od digitalnih kola obrađujući podatke u digitalnom obliku
- Mikroprocesor, mikrokontroler, mikroračunar
- Mikroračunari opšte namjene – ugradbeni ili ugrađeni sistemi
- Sistemi na čipu

Evolucija digitalnih sistema je pratila sljedeći trend:



Slika 8: Trend razvoja digitalnih sistema

Mikroračunar je definisan kao manji, jeftini računar limitirane namjene koji sadrži istu strukturu kao i današnji računari. Mikroračunari današnjice su veličine notebook-a i dolaze u znatno manjim dimenzijama nego u prošlosti. Inicijalno, ovakvi uređaji su okarakterisani sa manjom snagom te ograničenim internim operacijama i instrukcijama. Danas, ovakvi uređaji ne samo da imaju mogućnost množenja / dijeljenja sa pozitivnim i negativnim brojevima, već također i izvode aritmetičke operacije sa decimalnim brojevima.

Mikroprocesor je procesor na jednom čipu koji može izvršavati mikro instrukcije. Ove instrukcije su u vidu 0 i 1 vrijednosti. Mikroprocesor je centralna procesorska jedinica mikroračunara. Kao glavne komponente, ovaj element sadrži kontrolnu jedinicu (eng Control Unit), aritmetičku logičku jedinicu. Jedan od primjera mikroprocesora je Intel 8085 koji sadrži RAM/PROM/EPROM/EEPROM za spremanje programa, RAM za spremanje podataka, rezultata, ulazno/izlazne uređaje za komunikaciju sa vanjskim svijetom te ulazno/izlazne portove za komunikaciju sa ulazno/izlaznim uređajima.

Mikrokontroler je manji, jeftin mikroračunar dizajniran da izvršava specifične zadatke ugradbenog sistema kao što su prikazivanje informacija na ekranu, primanje signala. Mikrokontroleri se generalno sastoje od procesora, memorije, serijskih portova i periferala.

2.4 Sistemi na čipu

Sistemi na čipu (SoC) predstavljaju integrisana kola koji koriste jednu platformu za integrisanje cjeloukupne elektronike ili kompjuterskog sistema na istu. Kako ime sugerise, SoC predstavlja jedan čitav sistem na jednom čipu. Komponente koje uključuje SoC su centralna procesorska jedinica, ulazi i izlazi, interna memorija kao i analogni ulazni i izlazni blokovi između ostalog. U ovisnosti od sistema koji je reduciran na jedan čip, mogu se izvršavati raznovrsne funkcije kao što su procesiranje signala, bežična komunikacija, umjetna inteligencija i slično [6].

Sistemi na čipu su prvobitno kreirani s ciljem smanjivanja gubitaka energije, uštedi novca kao i reduciranju prostora. Sa SoC, ovi ciljevi su postignuti smanjivanjem veličine elemenata na jedan procesor koji koristi znatno manje energije. Ovi čipovi su razlog zbog kojeg smo u mogućnosti kreirati raznovrsne uređaje i nositi ih bez kompromisa po pitanju funkcionalnosti i sposobnosti ovih dodataka. Kao takvi, često nalaze mjesto u sistemima baziranih na IoT, embedded sistemima kao i mobilnim telefonima, automobilima i slično.

Ovakva tehnologija se također koristi u manjim osobnim računarima i laptopima s ciljem reduciranja potrošnje snage i poboljšanja performansi uz pomoć jedinstvenog čipa koji upravlja različitim aspektima sistema [6].

2.4.1 Komponente sistema na čipu

Sistemi na čipu dolaze sa komponentama koje mogu vršiti različite funkcije. Lista komponenata koje ulaze u sastav sistema na čipu su:

- CPU – centralna procesorska jedinica je 'mozak' operacije koji obavlja sve važne kalkulacije i upravlja ostalim komponentama ovog sistema. Bazirana je na FEDEX sistemu – preuzmi, dekodiraj, izvrši. CPU može sadržavati od 2 do 8 jezgara u zavisnosti od potreba i performansi koje aplikacija ispoljava.
- GPU – grafička procesorska jedinica je grafička kartica koja osigurava vizuelne tranzicije u korisničkom sučelju kao što su animacije i 3D igre.
- ROM – memorija namijenjena samo za čitanje (eng. Read Only Memory) je nepristupačni dio računara ili pametnog mobilnog uređaja. Ovdje se nalaze sve sistemske informacije kao i operativni sistem.
- RAM – kompjuterska memorija u koju se učitavaju aplikacije i procesi neposredno prije njihovog korištenja. Više RAM memorije podrazumijeva mogućnost simultanog pokretanja većeg broja aplikacija bez primjetnog kašnjenja između aplikacija.
- Modem – element koji osigurava konekciju i vidu bežične radio mreže. Jedan od načina spajanja na internet je u vidu WiFi, GPS, Bluetooth, 2G/3G/4G LTE i drugih [7].

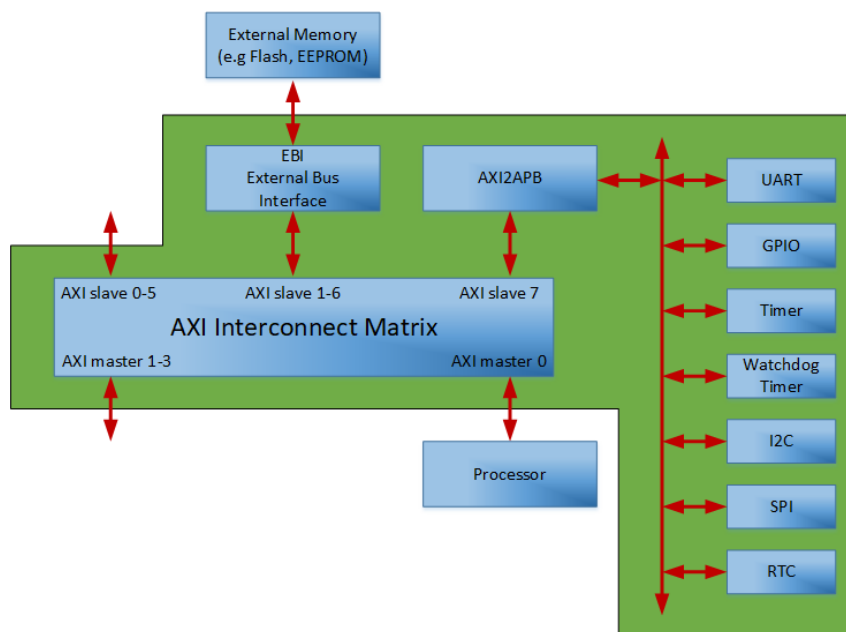
Najpopularniji SoC današnjice su Qualcomm Snapdragon, MediaTek, Samsung Exynos, HiSilicon Kirin te NvidiaTegra [7].



Slika 9: Sistem na čipu u mobilnim uređajima

2.4.2 Gradivni blokovi sistema na čipu

Sistem na čipu mora sadržavati procesor kao osnovni dio koji definiše funkcije ovakvog sistema. Uobičajno SoC ima više procesorskih jezgara. Može biti mikrokontroler, mikroprocesor, digitalni signalni procesor ili aplikaciono specifični instrukcijski set procesor. Čip također mora da sadrži memoriju koja mu omogućava izvršavanje operacija. Može sadržavati RAM, ROM, EEPROM ili čak i flash memoriju. Pored memorije i procesora, SoC mora posjedovati eksterno sučelje koje podržava industrijske standardne komunikacione protokole poput USB, Ethernet, HDMI, DPI i slično. Potrebno je da sadrži GPU ili grafičku procesorsku jedinicu za vizualizaciju kao i elemente poput regulatore napona, oscilatore, tajmere, satove, analogno-digitalne i digitalno-analogne konvertore itd [8].



Slika 10: Struktura sistema na čipu

Prednosti sistema na čipu su:

- Očuvanje energije, prostora i cijene
- Visoko-efikasni sistemi koji maksimiziraju performansu po vat
- Minimiziranje latencije različitih elemenata pomoću adekvatnog strateškog postavljanja matične ploče u cilju smanjenja interferencije i kašnjenja kao i ubrzavanje transmisije podataka

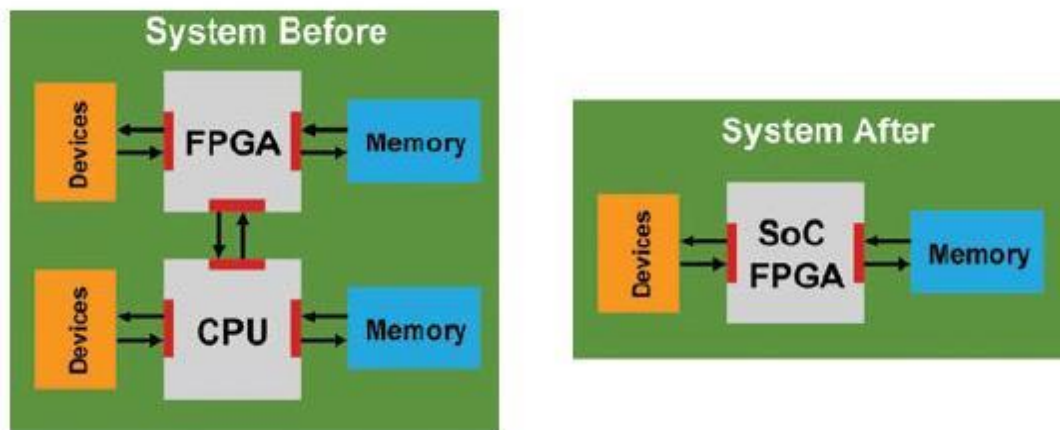
Mane sistema na čipu su:

- Zahtjevi vezani za vrijeme izlaska na tržište
- Cijena fabrikacije
- Povećana kompleksnost sistema
- Povećani zahtjevi za verifikaciju

Procesori današnjice, iako nekad primarni i najznačajniji dijelovi računarskog sistema predstavljaju dio jednačine čiji rezultat je upravo sistem na čipu koji kombinuje moć procesora sa ostalim komponentama postižući finalnu funkciju [8].

2.5 Sistemi na čipu sa ARM Cortex-M procesorima

Arm Cortex-M procesori predstavljaju jednu od najpopularnijih arhitektura koja se koristi za Internet of Things (IoT) i ugradbene aplikacije. Za mnoge dizajnere digitalnih sistema, digitalni blokovi međusobno povezuju procesore u mnogim operacijama poput upravljanjem kontrole toka. Također, upotrebom manjeg Cortex-M procesora jednostavnog za upotrebu pri dizajnu sistema javlja se jednostavna solucija za mnoge probleme [9].



Slika 11: Prije i poslije ugradnje SoC na FPGA uređaje

Integracija tehnologija na jedinstven dio silicijskog materijala eliminiše upotrebu plastičnog pakovanja, uz znatno očuvanje prostora. Kako se signali između procesora i FPGA nalaze na jednoj površini, komunikacija između dva dijela zahtijeva znatno manju potrošnju energije u poređenju sa upotrebom dva odvojena čipa. Pored toga, ovakva povezanost osigurava znatno manje kašnjenje u poređenju sa prijašnjom implementacijom [9].

Upotreba mašine konačnog stanja implementirane u Verilogu ili VHDL-u bi u jednostavnijim digitalnim aplikacijama mogla zadovoljiti sve kontrolne funkcije, te u tom slučaju procesor nije potreban. U slučaju kompleksnijih aplikacija gdje broj konačnih stanja eksponencijalno raste, fleksibilnost sistema je ugrožena te upotreba procesora postaje neizbježna. Za bolju fleksibilnost, kompleksnija kontrola toka programa je pokrivena procesorom upravljanim od strane kontrolnog softvera koji se jednostavno može podesiti i modificirati. Kao rezultat ovakve implementacije, ugradbeni procesori u sve više popularni u FPGA dizajnu. Iako je moguće koristiti zasebne mikrokontrolere za kontrolu FPGA baziranih digitalnih sistema, rezultat je povećan broj komponenti sistema kao i potencijalni problemi u povezivanju signalnih veza između procesora kao i jedan od glavnih nedostataka koji se ogleda u pouzdanosti takvog koncepta. Generalne prednosti upotrebe procesora u FPGA su:

- Mogućnost kontrole kompleksnih zadataka poput grafičkog sučelja kao i pohrana podataka
- Aplikacioni programi su razvijeni u ažurirani odvojeno od hardverskog dizajna što osigurava fleksibilnost u razvoju proizvoda
- Smanjen broj komponenti potrebnih za formiranje sistema jer se eliminiše zahtjev za odvojenim procesorom
- Preusmjeravanje signala između procesora i funkcionalne logike je odrađeno automatski od strane FPGA razvojnih alata
- Mala limitacija sučelja između procesora i korisnički – definisanih blokova.
- Pohranjivanje programskog koda na memoriji FPGA uređaja omogućujući istovremeno ažuriranje softvera i izvršavanje programskog koda na hardveru istovremeno [9].

Dizajn Cortex-M procesora osigurava:

- Dobru performansu na malom području
- Jednostavan razvoj softvera
- Dokazano pouzdana i efikasna tehnologija

2.6 Različiti tipovi ARM procesora

Arm procesori su rasprostranjeni u mnogim aplikacijama i imaju različite funkcije. Za podršku širokog spektra mogućnosti, definisani su različiti tipovi ARM procesora kako bi dizajneri odabrali najbolju opciju za njihov uređaj. Aplikacioni zahtjevi mobilnih uređaja su drastično drugačiji od zahtjeva kontrolera motora. Kako bi se obuhvatila raznovrsnost aplikacionih zahtjeva, ARM pruža širok raspon različitih procesora koji spadaju u familiju Cortex procesora:

- Cortex-A: aplikacioni procesori za kompleksne sisteme. Primjer procesora u ovoj klasi je Cortex-A53 razvijen za podržavanje mobilnih uređaja koji zahtjeva procesiranje visoke performanse kao i operativni sistem poput Linux, Android, Windows...
- Cortex-R: procesor namijenjen za real-time sisteme čiji primjer predstavlja procesor Cortex-R52 kreiran da pruži visoku performansu, malo kašnjenje i robusne karakteristike.
- Cortex-M: procesori za mikrokontrolere. Primjer ovakvih uređaja je Cortex-M3 procesor namijenjen za ugradbene aplikacije koje zahtijevaju niske cijene uz pružanje visokih performansi i rapidnog odgovora sistema na prekid. Tipični primjeri upotrebe ovih

procesora u aplikacijama uključuju industrijske kontrole, korisnički proizvodi poput prenosivih audio uređaja, digitalnih kamera i slično [9].

Tabela 1: Karakteristike ARM procesora

| Karakteristike | Cortex-A | Cortex-R | Cortex-M |
|-------------------------------|---|---|--|
| Tip arhitekture | Podržava 64 i 32-bit od Armv8-A, starija arhitektura podržava striktno 32-bit | Podržava 64 i 32-bit od Armv8-R, starija arhitektura podržava striktno 32-bit | Samo 32-bit |
| Taktna frekvencija | Duži kanal optimiziran za visok frekventni raspon | Srednji kanal | Manji kanal optimiziran za sistem niže potrošnje |
| Podrška za virtualnu memoriju | Da | Ne | Ne |
| Podrška za virtualizaciju | Da | Ne | Da od ArmV8-M verzije procesora |
| Korištenje prekida | Bazirano na generičnom kontroleru prekida sa podrškom za virtualizaciju. | Bazirano na generičnom kontroleru prekida. | Bazirano na 'Nested Vectored Interrupt Controller' internom za procesor. |

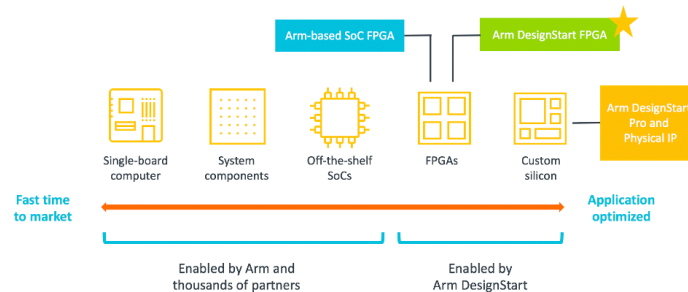
2.7 Cortex-M i FPGA

Za generalno procesiranje podataka i kontrolne aplikacije, Armv6-M procesori su više nego dovoljni da zadovolje različite uvjete i potrebe:

- Cortex-M0 procesor je najmanji ARM procesor (samo 12,000 logičkih sklopova u minimalnoj konfiguraciji), jednostavna struktura kanala na 3 nivoa bazirana na Von-Neumannovoj arhitekturi sabirnice.
- Cortex-M1 procesor je sličan M0 procesoru ali je optimiziran za FPGA aplikacije.
- Cortex-M0+ procesor je baziran na Armv6-M arhitekturi sa izbornom zaštitnom jedinicom za memoriju.
- Cortex-M23 procesor je pogodan za ugradbene sisteme koji zahtijevaju dodatni nivo sigurnosti uz upotrebu Arm TrustZone ekstenzije za sigurnost. Pored dodatne ekstenzije, ostale pogodnosti ovog procesora su:
 - Dodatne instrukcije (poređenja, dijeljenja)
 - Podrška za veći broj prekida (do 240)
 - Real-time instrukcije
 - Više konfigurabilnih mogućnosti
- Cortex-M3 procesor se koristi za aplikacije koje igraju bitnu ulogu u procesiranju podataka. Instrukcijski set pruža podršku za različite adresne modove, kondiciono izvršavanje, procesiranje, množenje i akumuliranje. Sa relativno malim Cortex-M3 procesorom se može postići izrazito visoka preformansa.

- Cortex-M4 procesor se koristi u digitalnoj obradi signala gdje je potrebno precizno izračunavanje uzimajući u obzir da ovaj procesor podržava 32-bitne operacije kao i izbornu jedinicu za decimalne operacije (eng. Floating-point unit).
- Cortex-M7 procesor bilježi najbolje performanse u Cortex-M familiji sa kanalom od 6 nivoa kao i spektakularnim dizajnom omogućavajući izvršavanje i do 2 instrukcije po ciklusu. Kao i Cortex M4, podržava 32-bitne operacije kao i izbornu jedinicu za decimalne operacije. Ovaj procesor je kreiran za aplikacije visokih performansi kao i kompleksnih memorijskih sistema podržavajući različite instrukcije kao i pohranu podataka [9].

Supporting innovation across the design spectrum



Slika 12: Inovativnost ARM procesora i FPGA uređaja

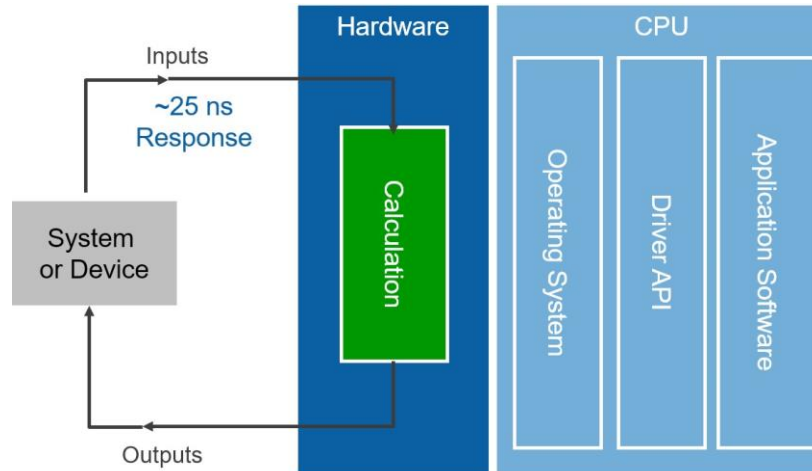
2.8 FPGA – Field Programmable Logic Array

FPGA je skraćeno za programabilna logička kola (eng Field-Programmable Gate Array) – naziv rezervisan za tip integriranih kola koja se mogu programirati čak i nakon proizvodnje iz čega potiče naziv „programabilna“ kola, dok logička kola predstavljaju dvodimenzionalni niz logičkih kola koji se nalazi u arhitekturi. Svi moderni osobni računari, mobilni uređaji i tableti su primjeri opšte-namjenskih računara. Ovakvi računari inkorporiraju 'Von Neumann-ov' pristup koji podrazumijeva da instrukcija preuzimanja i obrade podataka ne može biti izvršena istovremeno. S obzirom na to da su ovo sekvencijalne mašine, njihova performansa je ograničena. S druge strane, aplikaciono-specifična integraciona kola (ASICs) su kreirana za specificiran zadatak. Ovakva kola koriste prostorni pristup za implementaciju jedne konkretne aplikacije i pružanja maksimalne performanse. Nije moguće reprogramirati ovakva kola u cilju izvršavanja neke druge funkcije [10].

FPGA predstavljaju kompromis između performansi aplikaciono-specifičnih integracionih kola i fleksibilnosti opšte-namjenskih procesora.

FPGA bilježe manju energetska efikasnost u poređenju sa ASIC kao i manju podršku za masovnu produkciju. S druge strane, reprogramabilnost i niska cijena su prednosti koje karakterišu ovakva kola. FPGA dolaze sa velikim brojem programabilnih logičkih blokova i veza koja mogu biti rekonfigurisana. Na ovaj način upotrebom XOR i AND kola se mogu proizvesti izrazito kompleksne funkcije. Iz tog razloga, programabilna logička kola nalaze primjenu u automobilima, medicinskoj opremi, obradi slike i zvuka i slično [10].

U poređenju sa procesorima, FPGA izvršavaju akcije paralelno, što podrazumijeva dovoljno resursa za istovremene procesorske operacije. Svaki nezavisni zadatak je dodijeljen sekciju čipa te može funkcionirati autonomno bez sudjelovanja ostalih blokova. Na ovaj način se sadržava performanse jednog dijela aplikacije usljed dodavanja novih operacija i funkcija [10].



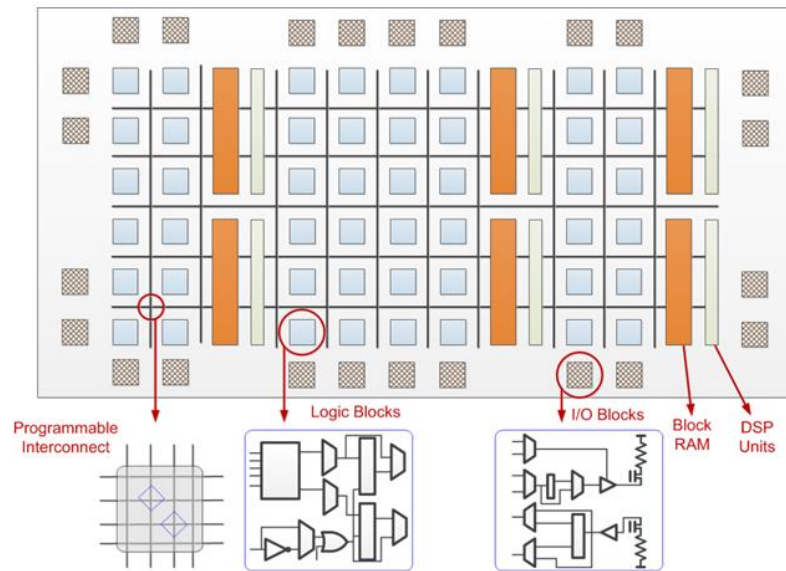
Slika 13: Struktura FPGA razvojne ploče

2.8.1 Interna arhitektura FPGA

Kompanija pod nazivom Xilinx 1985. godine proizvodi prvi komercijalni FPGA – XC2064. Konkurentna kompanija Altera, otkupljena od strane Intela 2015. je također doprinijela napretku i pomjeranjem granica FPGA proizvoda. FPGA su nastali iz jednostavnijih tehnologija poput programabilnih memorija namijenjenih samo za čitanje (eng Programmable Read-Only memory – PROM) ili programabilni logički uređaji (eng Programmable Logic Devices – PLD) [10].

FPGA se sastoji od 3 osnovne komponente:

- Konfigurabilni logički blokovi – implementiraju logičke funkcije
- Programabilne veze – implementiraju povezivanje između blokova
- Programabilni logički blokovi – povezuju eksterne komponente (XOR, AND..)



Slika 14: Arhitektura FPGA razvojne ploče

Logički blokovi implementiraju logičke funkcije sastavljeni su od različitih elemenata poput parova tranzistora, look-up tabela, flip flopova i multiplexera. Ovakvi blokovi se ponašaju kao odvojeni moduli koji funkcioniraju paralelno. Za razliku od lego blokova, ovi blokovi su konfigurabilni čime se osigurava željena funkcija. Hijerarhija programabilnih veza je iskorištena za dodjeljivanje resursa između konfigurabilnih logičkih blokova (eng Configurable logic blocks – CLB) gdje su različiti segmenti povezani putem kanala različitih dužina. Svaki CLB je povezan sa matricom koja osigurava generalnu strukturu putanje. Ova matrica sadrži programabilne multiplexere koji su korišteni u svrhu odabira signala u odgovarajućem kanalu povezivanjem vertikalnih ili horizontalnih linija [10].

Ulazno/Izlazni blokovi se koriste za povezivanje konfigurabilnih logičkih blokova i usmjeravanje eksternih komponenata. U prošlosti, nije postojao procesor da pokreće softver što je podrazumijevalo kreiranje kola od nule u cilju implementiranja aplikacije. Današnja FPGA arhitektura sadrži i specijalizirane programabilne blokove poput aritmetičko-logičkih funkcionalnih blokova, RAM blokova, multiplexera i mikroprocesora [10].

2.9 Razlika između FPGA i mikrokontrolera

FPGA i mikrokontroleri su učestalo korišteni s ciljem prezentiranja izlaznih rezultata na ekranu. Arhitekturu mikrokontrolera je moguće kreirati korištenjem FPGA ali obratno nije moguće [11].

- Skoro svaki računar sadrži ugradbeni mikrokontroler za izvršavanje zadataka i interakcija. FPGA je integrisano kolo koje sadrži milion programabilnih logičkih kola dizajniranih da izvršavaju logičkih kola. FPGA trebaju eksterne periferale poput RAM i ROM za aplikaciju.

- Mikrokontroleri koriste softverski program za izvršavanje komandi kao što su C, C++ dok FPGA koriste alate poput VHDL i Verilog.
- Procesorska moć kod mikrokontrolera je vremenski limitirana, dok je kod FPGA uređaja prostorno limitirana – veći broj logičkih kola povećava šansu obavljanja željene funkcije.
- FPGA su generalno fleksibilniji imajući u vidu mogućnost reprogramiranja i preusmjeravanja logičkih kola bezbroj puta u cilju obavljanja specifičnog zadatka. Mikrokontroleri mogu vršiti limitirane zadatke koje dolaze sa instrukcijama. Prilikom razvijanja vlastitog koda, developer mora pratiti odgovarajuće restrikcije.
- Mikrokontroleri prolaze kroz svaku liniju kola sekvencijalno dok FPGA procesuiru komande simultano te može izvršiti nekoliko linija koda istovremeno.
- U slučaju mikrokontrolera, procesor prelazi sa jednog koda na drugi kako bi se postigao svojevrstan vid paralelizma. Iz tog razloga, jednostavnost u pisanju koda je jedna od prednosti mikrokontrolera u odnosu na FPGA uređaje.
- Paralelizam FPGA omogućava kontrolu programa putem prekida korištenjem mašine konačnog stanja (eng Finite State Machine – FSM). U slučaju mikrokontrolera, potrebno je uzeti u obzir vrijeme potrebno da se riješi prekid.
- FPGA su prikladni za paralelno procesuiranje velikom brzinom uz odgovarajuću dozu prilagodbe. Neophodno je prvo kreirati FPGA funkcije, kompajlirati kod i konvertovati isti u mašinski jezik. Mikrokontroleri raspolažu gotovim paketima/modulima za izvršavanje različitih zadataka shodno vlastitim potrebama.
- Mikrokontroleri su jednostavni za korištenje i konfiguraciju te sekvencijalnim upravljanjem podataka velikim brzinama. Mikrokontroleri su fleksibilniji u odnosu na FPGA po pitanju programiranja [12].

3 Struktura Smart Home sistema

3.1 Uvod – opis strukture pametne kuće

Pametna kuća predstavlja kombinaciju raznovrsnih podsistema putem naprednih tehnologija kao što su fiber optički internet, bluetooth i slično. Ovakvi sistemi omogućavaju razmjenu resursa i informacija unutar kuće, kao i razmjenu informacija sa eksternom mrežom. Glavni cilj je pružanje efikasne, udobne, sigurne i ekološki prihvatljive verzije kuće integrišući sistem, uslugu i menadžment. Glavne osobine koje karakterišu pametnu kuću su:

- Pametna kuća može realizovati interakciju između korisnika i električne mreže, te dobiti informaciju o iskorištavanju električne energije i cijeni. Također može podesiti plan potrošnje elektriciteta, kao i pružiti mogućnost racionalne usluge električne energije uz očuvanje iste predstavljajući ekološki prihvatljive uslove.
- Pametna kuća može poboljšati sigurnost, udobnost, pogodnost i interaktivnost kućnog života
- Pametna kuća može podržavati udaljeno plaćanje (eng remote payment)
- Pametna kuća može pratiti i komunicirati korištenjem mobilnog uređaja, uvezane mreže i slično
- Pametna kuća realizira očitavanja podataka u pravom vremenu kao i mjerenje temperature, gasa i različitih drugih elemenata osiguravajući pogodne okolnosti za bolji život [13].

3.2 Arhitektura pametne kuće

Pametna kuća predstavlja prostor unutar kojeg je uključena kontrola i automatizacija ugrađenih tehnologija, a kojima je cilj ugodniji boravak ljudi u kući. Ovakvo prebivalište uključuje uređaje, svjetla, grijanje, klimu, TV, računare, friždere, alarmne sisteme, kamere koji imaju mogućnost međusobnog komuniciranja pomoću odgovarajućeg vremenskog rasporeda, mobitela ili interneta. Ovakvi sistemi se sastoje od prekidača i senzora povezanih sa centralnim čvorištem koji je kontrolisan od strane korisnika putem terminala postavljen na zid ili mobilnog uređaja povezan na internet servis. Pametne kuće pružaju sigurnost, energetske efikasnost i niske cijene izvedbe, uz određenu pogodnost. Instalacija pametnih produkata pruža pogodnost i očuvanje novca, vremena i energije. Ovakvi sistemi su fleksibilni i prilagodljivi trenutnim izazovima u skladu sa potrebama korisnika. U velikom broju slučajeva, infrastruktura ovakvog doma je dovoljno fleksibilna da integriše različite tipove uređaja bez obzira na standarde i dobavljače [14].

3.2.1 Usluge pametne kuće

Tipična pametna kuća sadrži set senzora za mjerenje kućnih uslova kao što su temperatura, vlažnost, svjetlost. Svaki senzor je definisan za mjerenje jednog ili više elemenata. Temperatura ili vlažnost mogu biti izmjerene korištenjem jednog senzora, drugi senzori se mogu koristiti za

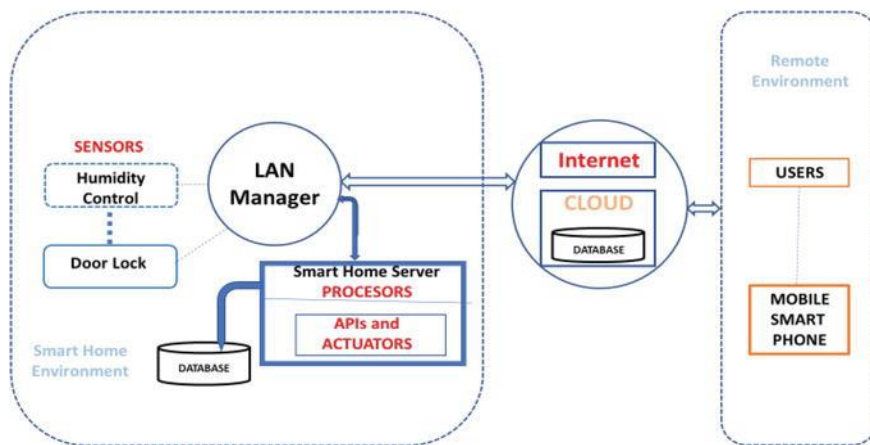
mjerjenje nivoa svjetlosti za specifično područje kao i udaljenost objekta kojim je to svjetlo izloženo. Svi senzori omogućavaju pohranjivanje podataka i vizualizaciju tako da korisnik može izvršiti monitoring istih u bilo kojem trenutku i bilo gdje. Da bi ovo funkcionisalo, potrebno je procesirati signal, upotrijebiti komunikaciono sučelje kao i izgraditi tzv. 'cloud' infrastrukturu [14].

Kreiranje web usluge za upravljanje kućnih uređaja će biti podržano korištenjem web infrastrukture. Ovakvi servisi omogućavaju korisniku kontrolu izlaza pametnih aktuatora povezanih sa kućnim elementima poput lampi i ventilatora. Pametni aktuatori su uređaji, poput ventila i prekidača koji izvršavaju akcije poput paljenja i gašenja te podešavanja operacionog sistema. Aktuatori pružaju raznovrsne funkcionalnosti poput: otvaranja/zatvaranja, podešavanja nivoa, promjena toka kretanja, hitno isključivanje i slično. Aktiviranje aktuatora podrazumijeva slanje komande na isti (eng digital write) [14].

Tehnologije pametne kuće su često iskorištene za pristup javnim ustanovama. Sistem podrazumijeva korištenje baze podataka sa mogućnostima identifikacije i autorizacije individualnih osoba. Kada osoba pristupa takvom kontrolnom sistemu, identifikacioni atributi su prikupljeni te se vrši upoređivanje takvih podataka sa bazom. Ako se podaci poklapaju, pristup je dozvoljen, u suprotnom, pristup je odbijen. Za širu distribuciju, web servis je korišten za prikupljanje i obradu podataka. Čest pristup ovakvim slučajevima je korištenje RFID kartice gdje svaka ovlaštena osoba sadrži ovakvu karticu. Osoba skenira karticu na RFID čitaču koji šalje skenirani ID na tzv. Cloud/web servis. ID je upoređen sa trenutnim ovlaštenim ID-ovima u bazi podataka, na osnovu čega se odlučuje dodjeljivanje pristupa, ili odbijanje istog [14].

3.2.2 Osnovne komponente pametne kuće

Kako bi se osigurao kvalitetan protok i prikaz podataka i aktivnosti korištenih u pametnoj kući, potrebno je kreirati adekvatan sistem kao na sljedećem primjeru:



Slika 15: Pametna kuća i implementacija konekcije različitih elemenata

Aktuatori izvršavaju odredbe od strane servera ili drugih kontrolnih uređaja. Tokom procesuiranja podataka sa senzora, izvršena je provjera zadanih uvjeta. U slučaju zadovoljavanja određenih uslova, aktuatori sprovode definisane mjere i akcije. Baza podataka sprema procesuirane podatke prikupljene od strane senzora te iste koristi za analizu, obradu, prezentaciju i vizualizaciju. Ovakvi podaci su sačuvani u bazi za daljnje upotrebe u budućnosti [14].

Internet of Things (IoT) paradigma definiše uređaje povezane sa internetom. Uređaji su objekti kao što su senzori, aktuatori povezani sa komunikacionim interfejsom, procesorskom jedinicom, limitiranim prostorom za pohranu podataka kao i softverskom aplikacijom. Kao takav, IoT omogućava integraciju objekata na internetu osiguravajući interakciju između ljudi i definisanih elemenata. Glavna tehnologija IoT uključuje radio frekventnu identifikaciju (RFID), tehnologije kojima raspolažu senzori kao i inteligentne tehnologije. Internet of Things omogućava procesiranje i komunikacije vještine zajedno sa unikatnim algoritmima koji integrišu različite komponente u jednu kompaktnu cjelinu. Ovakav način organizacije osigurava fleksibilnost i promjenu željenih komponenata u ovisnosti od potreba korisnika [14].

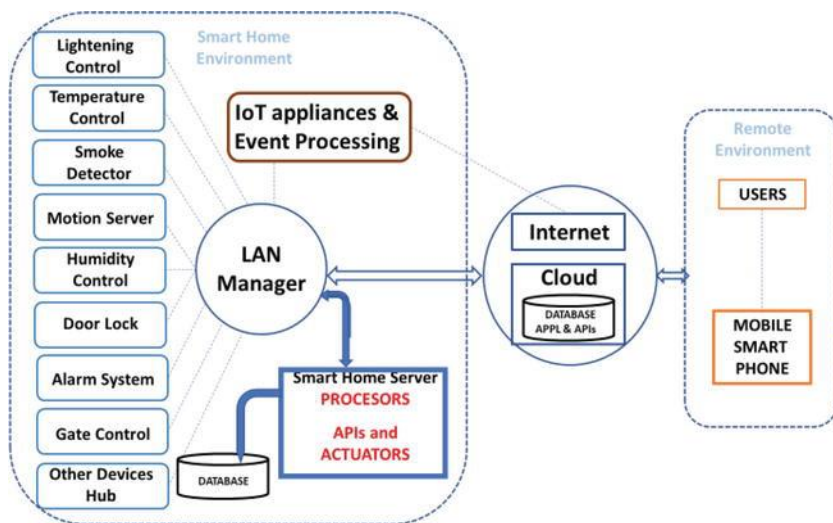


IoT i pametna kuća mogu imati korist od širokog spektra resursa i funkcionalnosti koji web pruža kako bi se nadoknadili nedostaci povezani sa ograničenjima u pohranjivanju, procesiranju, komunikaciji, kopijama i oporavku. Web servis ili ‘cloud’ može podržavati IoT servis te obrađivati i izvršavati aplikacione zadatke na temelju primljenih podataka. Takva pametna kuća može biti fokusirana na osnovne i kritične funkcije minimizirajući upotrebu lokalnih resursa uz oslanjanje na mogućnosti cloud servisa i resursa. U tom slučaju, pametna kuća i IoT uređaji se fokusiraju na prikupljanje podataka, osnovnu obradu istih te transmisiju na web za daljnje procesiranje. Kako bi se adekvatno zaštitio, cloud mora biti privatni, za maksimalnu zaštitu podataka.

IoT, pametna kuća i cloud nisu direktan spoj tehnologija već balans između lokalnog i centralnog obrađivanja, optimizirajući potrošnju raspoloživih resursa. Računarske zadatke je moguće izvršiti lokalno na IoT i uređajima pametne kuće ili direktno na web servisu. Odabir obrade podataka zavisi od različitih kompromisa poput dostupnosti podataka, broj podataka koji se prenosi, zavisnost komunikacije, te uzimanje u obzir level sigurnosti kojim raspolaže sistem.

3.3 Napredna pametna kuća

Savremena pametna kuća sadrži kombinaciju elemenata međusobno povezanih koji osiguravaju napredne funkcije olakšavajući život korisniku iste.



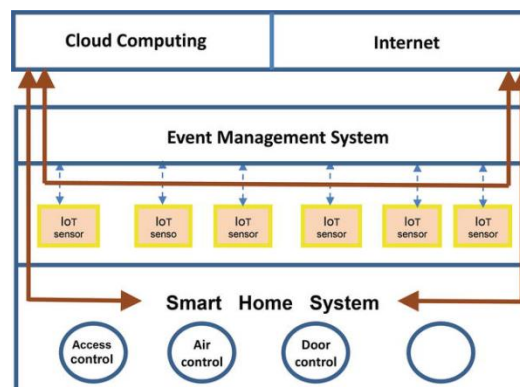
Slika 17: Napredna pametna kuća

Posmatrajući sliku iznad, na lijevoj strani kućnog okruženja se nalaze tipični uređaji koji su povezani sa lokalnom mrežom (eng local area network – LAN). Ova mreža omogućava međusobnu komunikaciju uređaja koji čine ovaj sistem kao i spoljašnjih elemenata. Na mrežu su povezani server i baza podataka. Server kontroliše uređaje, pohranjuje njihove aktivnosti, daje izvještaje, odgovara na upite, te izvršava zadane komande. Za kompleksnije upite i zadatke, server pametne kuće šalje podatke na web te sa udaljene lokacije aktivira zadatke korištenjem aplikacionog sučelja. IoT elementi su povezani na internet i LAN mrežu. Povezivanje elemenata na internet omogućava krajnjem korisniku udaljenu komunikaciju i kontrolu nad pametnom kućom.

Pametna kuća ima 3 komponente:

- Hardver
- Softver
- Komunikacione protokole

Napredne pametne kuće uključuju komponente poput IoT senzora, protokole, ugrađene programe, baze podataka te posredničke softvere.

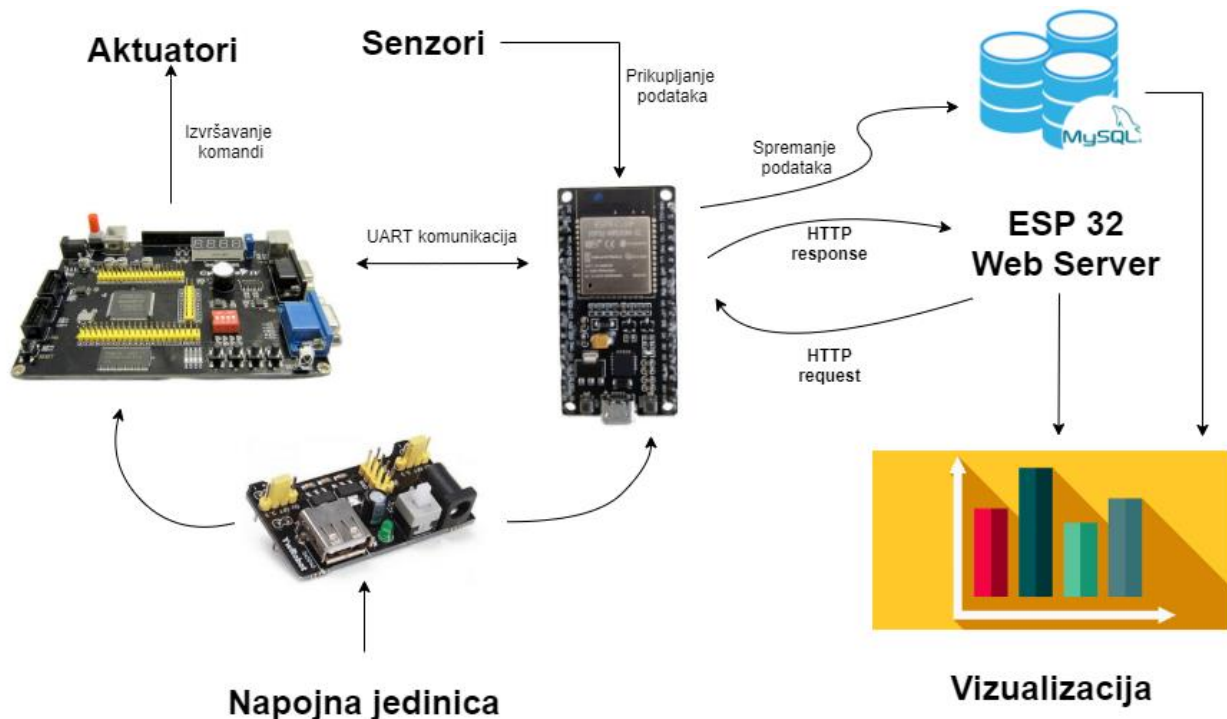


Slika 18: Sistem napredne pametne kuće

Aplikacija namijenjena za pametnu kuću ažurira bazu podataka u webu omogućavajući nesmetanu kontrolu i konstantu provjeru trenutnog stanja kuće. Tipična IoT platforma sadrži: sigurnosne uređaje za odobravanje pristupa, prikupljane poruka i informacija, te pohranu istih, administracija podataka, protokole, prikupljanje podataka za vizualizaciju i analizu, integracija sa ostalim web servisima, skalabilnost, aplikaciona sučelja za protok podataka u stvarnom vremenu, te različite biblioteke/module. IoT senzori su poznati po svojim mogućnostima za mjerenje temperature, svjetlosti, nivoa vode, pritiska, vlage, vibracija i slično. Komunikacioni protokoli koji se koriste su bluetooth, Wi-Fi ili GSM.

4 Hardverska implementacija pametne kuće

Implementacija pametne kuće omogućava hostiranje vlastitog lokalnog servera korištenjem odgovarajućih modula i tehnika, te iskorištavanje brzine i mogućnosti FPGA Cyclone IV razvojne ploče koja osigurava neometano paralelno izvršavanje operacija i zadanih procedura. Arduino preuzima željene upite od strane korisnika korištenjem mobilne aplikacije ili putem web preglednika na osnovu kojih putem UART komunikacije, dobivene komande proslijeđuje na FPGA ploču koja obrađuje zadanu komandu izvršavanjem neke od sljedećih funkcija: paljenje sijalica, aktiviranje alarmnog sistema, ventilatora, grijanja i slično.



Slika 19: Praktična realizacija sistema pametne kuće

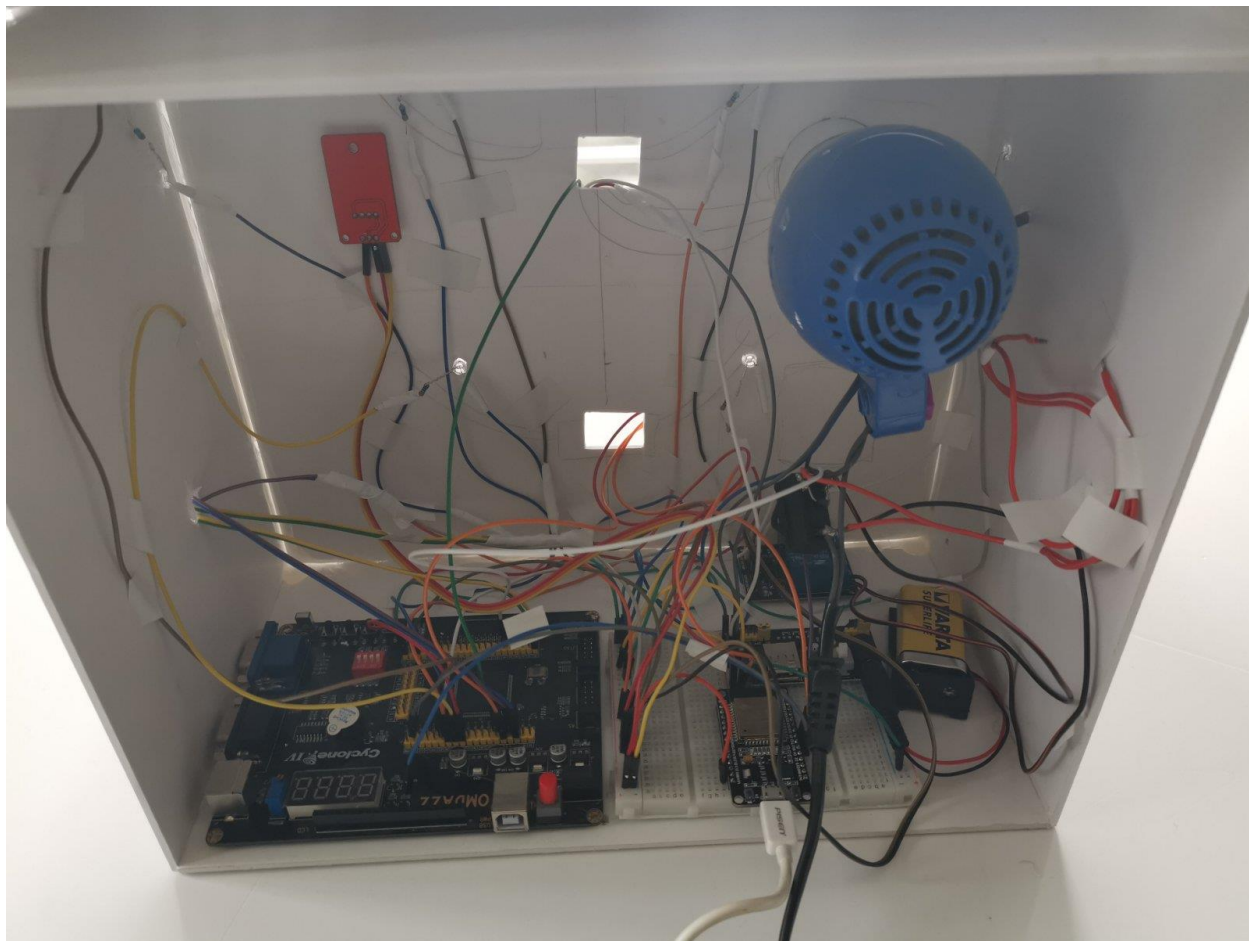
Maketa pametne kuće sadrži integraciju različitog hardvera u cilju simulacije karakteristika i funkcija koji karakterišu jednu ovakvu kuću. U ovom eksperimentalnom dijelu upotrebljena su dva temeljna elementa koji osiguravaju kontrolu i upravljanje manjim uređajima: FPGA Cyclone IV i Arduino ESP32 ploča.



Slika 20: Maketa pametne kuće



Slika 21: Maketa pametne kuće



Slika 22: Postavka elemenata makete pametne kuće

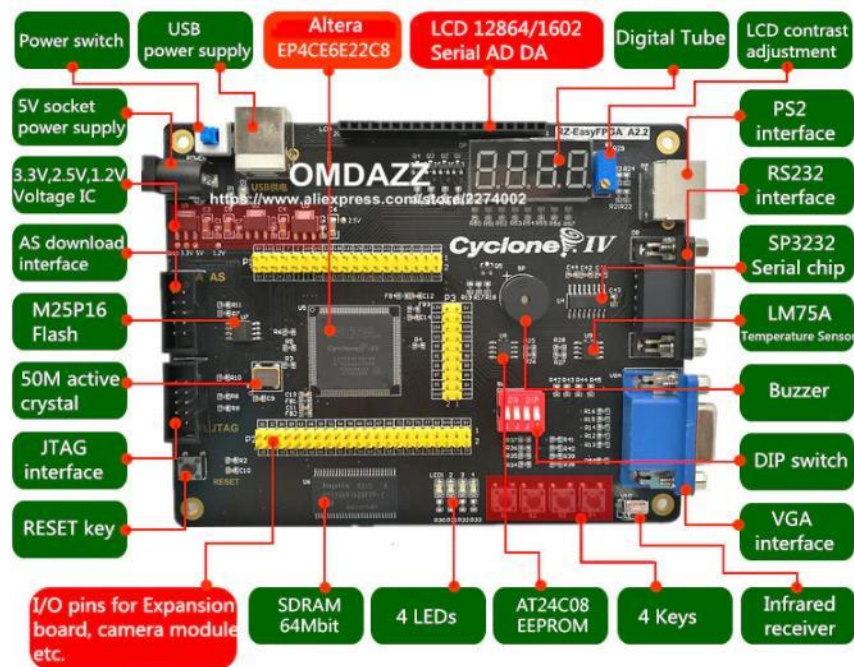
Ovi elementi međusobno razmjenjuju informacije prikupljene od strane senzora na osnovu kojih uvedeni aktuatori izvršavaju specifične akcije.

Princip rada je sljedeći:

- Korisnik na web stranici ili mobilnom uređaju šalje komandu koju želi da izvrši.
- Komanda se procesira na osnovu HTTP zahtjeva. Prilikom odabira određenog dugmeta ili unošenja specifičnih vrijednosti, HTTP zahtjev šalje predefinisano vrijednost putem UART komunikacije na FPGA Cyclone IV ploču.
- Na osnovu željene komande, aktivira se aktuator za pokretanje akcije.
- Senzori kontinualno mjere trenutno stanje kuće (temperaturu i vlažnost unutar i izvan kuće) te shodno tome, usljed aktiviranja automatske regulacije podešavaju temperaturu korištenjem FUZZY regulatora.
- Vrijednosti koje senzori očitavaju se direktno pohranjuju u bazu podataka koja se nalazi na web servisu omogućavajući vizualizaciju i pregled stanja nekoliko dana nazad.

4.1.1 FPGA Cyclone IV EP4CE6

Cyclone IV E familija FPGA razvojnih ploča je okarakterisana izrazito visokim nivoom funkcionalnosti uz minimalnu potrošnju energije i cijenu [15].



Slika 23: FPGA Cyclone IV

Ova razvojna ploča je poslužila za izvršavanje paralelnih operacija poput aktiviranja svjetla, paljena ventilatora, grijanja, aktiviranja alarmnog sistema ili otvaranje vrata/prozora [15]. Specifikacije Cyclone IV EP4CE6 ploče su:

- 136mm * 106mm veličine
- 6272 logička elementa
- 270 Kbits ugrađene memorije
- 15 ugrađenih 18x18 multipleksera
- 10 globalnih umreženih satova
- 8 ulazno/izlaznih elemenata koji dijele zajednički izvor
- 179 maksimalnih ulazno/izlaznih pinova

Parametri koji karakterišu ovu razvojnu ploču su:

- ALTERA Cyclone IV EP4CE6E22C8N glavni čip
- Ploča sa 16Mbit EPCS16N serijski konfigurisanim čipom gdje korisnik može skinuti i pokrenuti program preko JTAG ili AS sučelja.
- Ploča sa 64Mbit SDRAM, podržava SOPC, NIOSII razvoj.
- Upotreba 1117-3.3V čipa za regulator napona koji pruža izlaz od 3.3V
- Upotreba 1117-1.2V čipa za regulator napona koji pruža osnovni napon za FPGA

- Upotreba 1117-2.5V čipa za regulator napona koji pruža PPL izlaz
- Ulazni napon – DC 5V; ploča se također može napajati korištenjem USB konekcije.

Periferali i resursi koje uključuje ova ploča su:

- 1 dugme za paljenje, 1 dugme za reset i 4 korisnička tastera
- 4 led diode
- 8-segmentni led ekran
- 1 zujalicu
- PS2 sklop
- RS232 serijski port
- 1x20pin LCD socket
- Senzor za mjerenje temperature LM754A
- VGA sučelje
- I2C serijski port
- Modul za infrared

4.1.2 ESP32 DevKit

ESP32 DevKit u kombinaciji sa Arduino IDE omogućava kreiranje lokalnog servera, pohranjivanje HTML I CSS podataka u flash memoriju uređaja za pomenuti server. Omogućava i obrađivanje i slanje HTTP zahtjeva na server i odvojenu stranicu namjenjenu za monitoring, komunikaciju sa FPGA uređajem, mjerenje temperature i vlažnosti uz pomoć DHT senzora kao i korištenje fuzzy regulatora za regulaciju uslova unutar kuće [16].



Slika 24: ESP32 DevKit

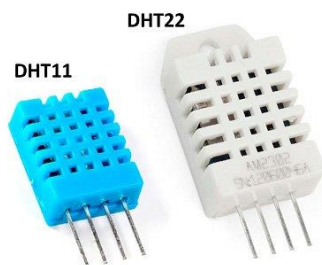
Karakteristike koje ističu ovaj uređaj su:

- CPU: Xtensa dual-core 32-bit LX6 mikroprocesor
- Memorija: 520 KiB SRAM

- Bežična konekcija: Wi-Fi ili Bluetooth v4.2
- 12-bitni SAR ADC do 18 kanala
- 2 x 8-bitni DAC
- 10 x senzor dodira
- 4 x SPI
- 3 x UART
- LED PWM do 16 kanala
- 2 x I²S sučelje
- 2 x I²C sučelje
- Infrared

4.1.3 DHT 11 i DHT22 senzori

DHT11 i DHT22 senzori omogućavaju jednostavno mjerenje temperature i vlažnosti zraka uz odgovarajuće razlike između pomenutih elemenata. Mjerenja temperature i vlažnosti se koriste za monitoring kao i regulaciju temperature na osnovu željene vrijednosti definisane od strane korisnika [17].



Slika 25: DHT11 i DHT22

Tabela 2: Razlika između DHT11 i DHT22 senzora

| Karakteristika | DHT11 | DHT22 |
|---------------------|-----------------------|------------------------|
| Temperaturni raspon | 0 – 50 °C / ± 2°C | -40 -125 °C / ± 0.5 °C |
| Raspon vlažnosti | 20-80% / ± 5% | 0 – 100% / ± 2-5% |
| Brzina uzorkovanja | 1Hz | 0.5Hz |
| Veličina | 15.5mm x 12mm x 5.5mm | 15.1 mm x 25mm x 7.7mm |
| Napon | 3-5V | 3-5V |
| Maksimalna struja | 2.5mA | 2.5mA |

4.1.4 PIR senzori

PIR senzori su uređaji za detekciju pokreta. Kombinacijom PIR senzora sa ugrađenim buzzerom FPGA uređaja dobijamo efikasan alarmni sistem na osnovu kojeg možemo detektovati neželjene pokrete. Ovakvi senzori mjere infracrveno zračenje objekata u svom polju na osnovu kojeg detektuje ‘zračenje’ emitovano ili reflektovano od strane različitih objekata. Svi objekti sa temperaturom iznad apsolutne nule emituju energiju u formi radijacije, na osnovu čega se funkcionalnost ovog senzora zasniva [18].



Slika 26: PIR senzor

Karakteristike ovog senzora su:

- Izlazni napon je visok / nizak (3.3V TTL)
- Velik raspon ulaznog napona od 4V do 12V (+5V preporučljivo)
- Može praviti razliku između pomjeranja objekta i kretanja čovjeka
- Pokriva udaljenost od 120° I 7 metara
- Radna temperatura između -20° do +80°C.

4.1.5 LED diode

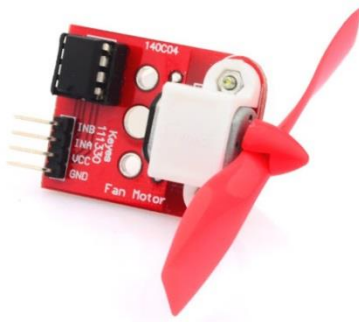
LED diode se koriste za osvijetljavanje prostora u unutrašnjosti i vani. Od ukupno 6 LED dioda koje se napajaju naponom od 5V, tri se koriste za kuhinju, dnevnu i trpezariju dok ostale tri osvijetljavaju prednji ulaz i garažu [19].



Slika 27: LED diode

4.1.6 L9110 ventilatori

Dva ventilatora kao na slici poslužila su za rashlađivanje prostorije i spuštanje temperature na željenu tačku [20].



Slika 28: L9110 ventilatori

Specifikacije ventilatora su:

- Ugrađen driver i N20 mini motor
- Maksimalna struja: 300mA
- Radni napon: 3.3-5V
- Veličina propelera: 75mm
- Veličina modula: 35 x 24 x 13mm

4.1.7 Fen – grijač

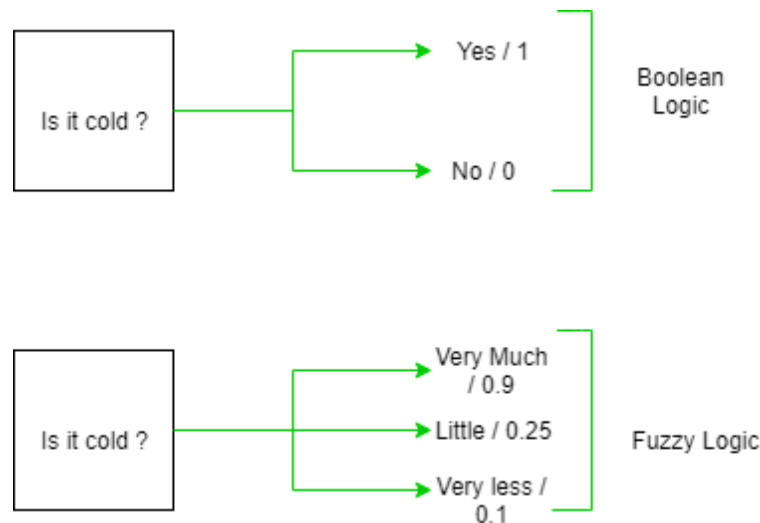
Fen je poslužio kao grijač koji održava temperaturu u slučaju uključivanja automatske regulacije i pruža mogućnosti zagrijavanja.



Slika 29: Fen - grijač

5 Fuzzy koncept upravljanja

Termin fuzzy predstavlja stvari koje nisu precizne ili u potpunosti jasne. U realnom svijetu gdje se susrećemo sa mnogobrojnim situacijama u kojima ne možemo odrediti da li je stanje tačno ili netačno, fuzzy logika pruža nevjerovatnu fleksibilnost sa kojom možemo predstaviti nepreciznost i netačnost u bilo kojoj situaciji. U Booleanovom sistemu, 1 predstavlja apsolutnu tačnu vrijednost dok 0 apsolutnu predstavlja netačnu vrijednost. U fuzzy sistemu ne postoji logika za apsolutnu tačnost i netačnost. U ovakvom sistemu, srednja vrijednost je prisutna i definiše parcijalnu istinu i neistinu [21].



Slika 30: Fuzzy logika

5.1 Fuzzy sistem zaključivanja – fuzzy regulator

Fuzzy logika je bazirana na vrijednostima koji se kreću između 0 i 1. Fuzzy modeli i sistemi su matematički način prezentovanja neodređene i neprecizne informacije iz čega proizilazi termin fuzzy – nejasan. Ovakvi modeli imaju mogućnost prepoznavanja, prezentovanja, manipulisanja interpretacije i iskorištavanja podataka i informacija koji su nejasni i neodređeni. Osnovni elementi fuzzy sistema zaključivanja su:

- Fazifikacija
- Baza znanja (baza podataka i baza pravila)
- Sistem zaključivanja
- Defazifikacija (izoštavanje) [22]

Fazifikacija se koristi za konvertovanje ulaza u fuzzy setove. Ulazi su najčešće mjerene vrijednosti od strane senzora koji se prenose na kontrolni sistem za procesiranje kao što su temperatura, pritisak, rpm [22].

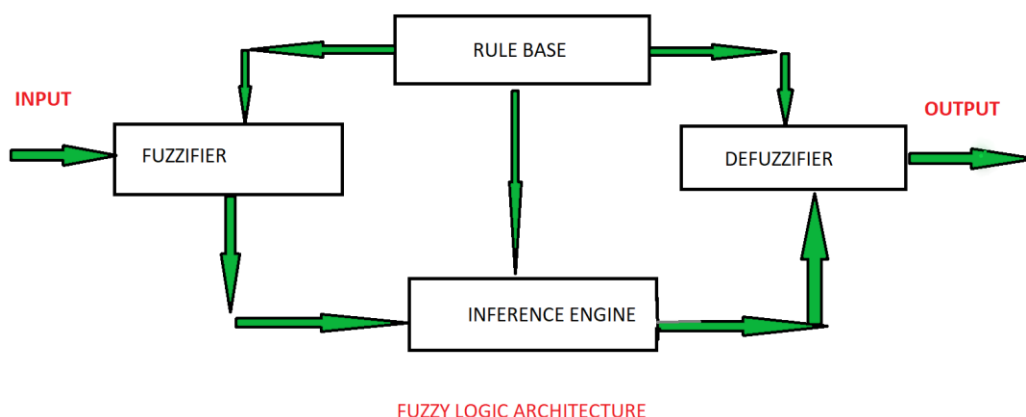
Baza znanja – sadrži set pravila i IF-THEN uslove koji usmjeravaju izvršavanje glavnih odluka sistema baziranih na lingvističkim informacijama [22].

Sistem zaključivanja definiše stepen podudaranja ulaza sistema sa definisanim pravilima na osnovu kojih određuje koja se pravila izvršavaju. Dati set pravila okida specifične radnje i pruža kontrolu nad sistemom [22].

Defazifikacija – konvertovanje fuzzy setova dobivenih sistemom zaključivanja u konkretne vrijednosti [22].

Funkcije pripadnosti – grafovi koji definišu kako je svaki ulazni dio mapiran u vrijednost između 0 i 1. Neki od tipova fazifikatora su:

- Singleton
- Gaussian
- Trapezoidal



Slika 31: Arhitektura fuzzy logike

Prednosti ovakvog sistema je u efikasnom radu sa različitim tipovima ulaza, bez obzira na njihovu preciznost, distorziju ili prisutnost šuma. Pored toga, konstrukcija ovakvih sistema je jednostavna i razumljiva. Fuzzy logika dolazi uz matematičke koncepte pružajući efikasan pristup rješavanja kompleksnih problema u različitim poljima života koji podsjećaju na osnovno ljudsko razmišljanje i donošenje odluka. Algoritmi bazirani na ovim sistemima mogu biti opisani sa malom količinom podataka što dovodi do uštede memorije [22].

Mane ovakvih sistema se ispoljavaju u fuzzy logici koja može dovesti do dvosmislenosti prilikom rješavanja i pristupa određenom problemu. U ovakvim slučajevima ne postoji sistematski pristup rješavanju problema upotrebom fuzzy logike [22].

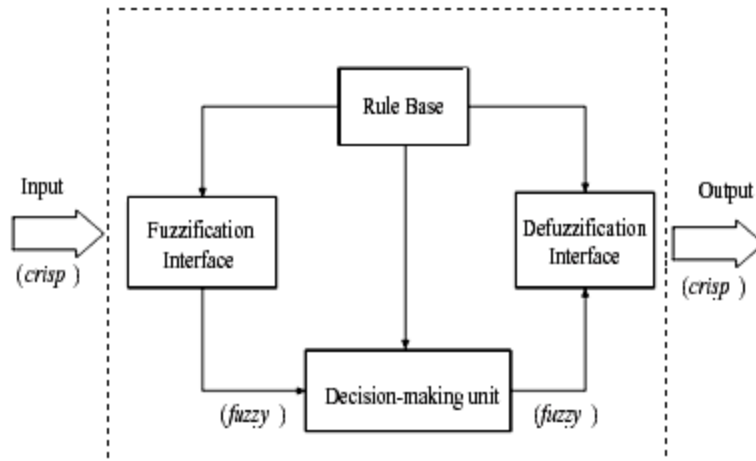
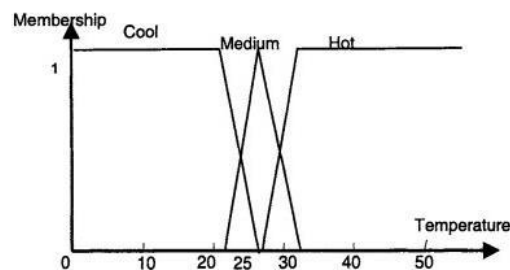


Figure 1: Basic Configuration of a fuzzy logic system.

Slika 32: Osnovna konfiguracija Fuzzy sistema

5.1.1 Fuzzy kontroleri

Fuzzy kontroleri su konceptualno vrlo jednostavni. Sastoje se od ulazne faze, procesiranja i izlazne faze. Prvi stepen kreiranja fuzzy kontrolera je definisanje ulaznih i izlaznih varijabli kontrolera. Ovo je odrađeno shodno željenim funkcijama istog. Ne postoji generalni set pravila prilikom odabira varijabli iako su to tipično stanja kontrolisanog sistema, greška i varijacija greške kao i akumulacija greške. Ulazna faza mapira senzore i druge ulaze poput prekidača, ručica za podešavanje i ostalih odgovarajućim funkcijama pripadnosti. Faza procesiranja stvara pravila koja generišu pojedinačni rezultat koji se u finalnom prikazu kombinuje kao rezultat svih pravila zajedno. Izlazna faza konvertuje kombinovani izlaz u specifičnu kontrolnu/izlaznu vrijednost. Fuzzy kontroleri su formirani tzv. lingvističkim terminima – kao što su toplo, hladno, vruće, nisko, visoko predstavljeni korištenjem funkcijama pripadnosti čiji setovi definišu kako je ulazna varijabla predstavljena unutar fuzzy ulaza [23].



Slika 33: Funkcija pripadnosti fuzzy kontrolera

Prepoznatljiva funkcija pripadnosti je trouglasta zajedno sa trapezoidnim oblikom, iako je oblik manje relevantan u poređenju sa brojem krivulja i njihovim pozicioniranjem. Faza procesiranja pravila je bazirana na kolekciji logičkih pravila u formi IF-THEN izjava. Tipični fuzzy sistemi imaju nekoliko pravila, kao na primjer:

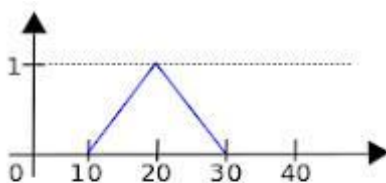
IF (temperatura je niska) THEN uključi grijalicu (grijalica je 'high' – visoka vrijednost).

Ovo pravilo koristi stvarnu vrijednost ulaza(temperature) kako bi se generisao stvarni rezultat u obliku fuzzy seta za grijalicu. Ovaj rezultat je korišten sa rezultatima ostalih pravila kako bi se formirao finalni kompozitni izlaz [23].

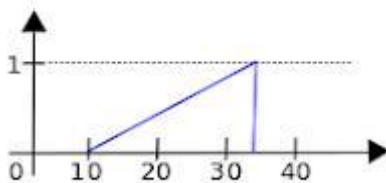
Realizacija fuzzy kontrolera u primjeru pametne kuće izvršena je na Arduino kontroleru korištenjem Fuzzy.h biblioteke. Ova biblioteka sadrži sljedeće objekte:

- Fuzzy objekat – uključuje cjeloukupni Fuzzy sistem preko kojeg manipulišemo fuzzy setovima, lingvističkim pravilima te ulazima i izlazima.
- FuzzyInput objekat - grupiše sve ulaze Fuzzy setova.
- FuzzyOutput objekat – grupiše sve izlaze Fuzzy setova
- FuzzySet objekat – glavni objekat Fuzzy biblioteke koji omogućava modelovanje sistema sa svakim setom. FuzzySet je kreiran sa parametrima baziranim na vrijednostima A,B,C,D koji generišu funkciju pripadnosti. Parametri u konstruktoru se definišu po principu FuzzySet(float a, float b, float c, float d).

Primjer generisanja funkcije pripadnosti korištenjem Arduino okruženja je sljedeći:



```
FuzzySet* fs = FuzzySet(10, 20, 20, 30);
```



```
FuzzySet* fs = FuzzySet(10, 33, 33, 33);
```

Slika 34: Generalni primjer funkcija pripadnosti

U praktičnoj primjeni izrade sistema pametne kuće sa Fuzzy regulatorom, ulaz je unutrašnja temperatura dok izlaze definišu ventilatori i grijači kao svojevrsan sistem regulacije temperature. Uz određivanje fuzzy ulaznih i izlaznih objekata, kreiran je set pravila koji podešava temperaturu unutrašnjosti sistema kuće, a na osnovu unesene željene vrijednosti korisnika. Razlikom između mjerene i željene vrijednosti postizemo optimalni rezultat (adekvatna aktivacija grijača ili ventilatora shodno definisanim pravilima) [23].

Lingvističke varijable sa lingvističkim vrijednostima koje su odabrane za ulaze i izlaze su:

Tabela 3: Lingvističke varijable sa lingvističkim vrijednostima za ulaze i izlaze sistema

| Tip | Naziv lingvističke varijable | Vrijednosti |
|-------|------------------------------|------------------------|
| Ulaz | Temperatura | Niska, srednja, visoka |
| Izlaz | Grijač | Ugašeno, upaljeno |
| Izlaz | Ventilator | Ugašeno, upaljeno |

Prvobitno se kreira jedinstven fuzzy objekat koji će sadržavati ulaze, izlaze i unaprijed određena pravila.

Tabela 4: Temperatura – ulazna lingvistička varijabla

| Naziv LV | Tip LV | Parametri |
|------------------|-------------|---|
| Niska - temp | trapezoidal | [0,0,optimalTemp-3, optimalTemp] |
| Optimalna - temp | trapezoidal | [optimalTemperature-5, optimalTemperature-2,optimalTemp+2, optimalTemp+5] |
| Visoka - temp | trapezoidal | [optimalTemperature, optimalTemperature+3, 50, 50] |

```
1. #include <Fuzzy.h> // Library for FUZZY sets - rules
2. Fuzzy *fuzzy = new Fuzzy(); // Creating new FUZZY object that can contain inputs, outputs and rules
```

Sljedeći korak je instanciranje ulaznih objekata – temperature.

Da bi kreirali adekvatan fuzzy regulator, potrebno je da prvo instanciramo ulazne objekte što je u ovom slučaju temperatura. Nakon definisanja ulaznog objekta, podesimo set podataka za taj objekat - niska, optimalna i visoka temperatura čije vrijednosti zavise od željenog unosa optimalne temperature korisnika. Varijacija na temperaturu je ± 2 stepena kako bi osigurali optimalnu temperaturu za korisnika u specifičnom rangu. Za odgovarajući ulazni objekat, istom dodijelimo definisani set i proslijedimo input cjelokupnom fuzzy sistemu korištenjem addFuzzyInput metode.

```
1. // Instantiating a FuzzyInput object
2. FuzzyInput *temperature = new FuzzyInput(1);
```

Jedinstvenom objektu fuzzy se dodjeljuje novokreirani ulaz – temperatura korištenjem funkcije addFuzzyInput.

Nakon definisanja ulaznog objekta, potrebno je podesiti set pravila – niska, optimalna i visoka temperatura.

```
1. // Instantiating a FuzzySet object
2. FuzzySet *low = new FuzzySet(0, 0, optimalTemperature - 3, optimalTemperature);
3. // Instantiating a FuzzySet object
4. FuzzySet *optimal = new FuzzySet(optimalTemperature -
    3, optimalTemperature, optimalTemperature, optimalTemperature + 3);
5. // Instantiating a FuzzySet object
6. FuzzySet *high = new FuzzySet(optimalTemperature, optimalTemperature+3, 50, 50);
7. temperature->addFuzzySet(low);
8. // Including the FuzzySet into FuzzyInput
9. temperature->addFuzzySet(optimal);
10. // Including the FuzzySet into FuzzyInput
11. temperature->addFuzzySet(high);
12. // Including the FuzzyInput into Fuzzy
```

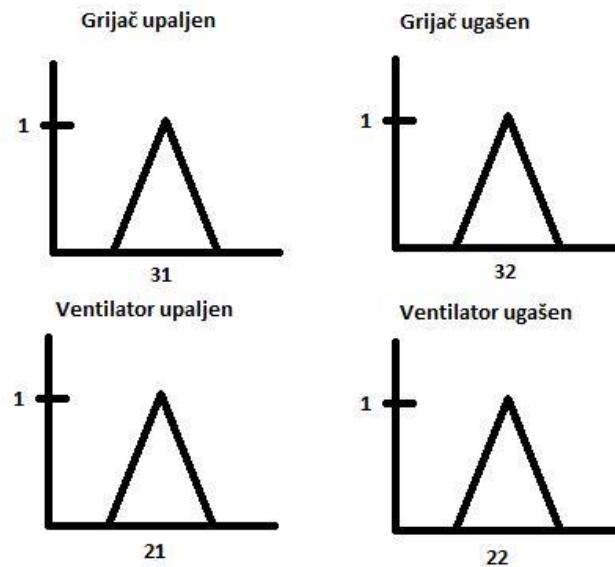
Jedinstvenom fuzzy objektu se dodjeljuje novokreirani ulaz – temperatura korištenjem funkcije addFuzzyInput funkcije.

```
1. fuzzy->addFuzzyInput(temperature);
```

Prethodnu konfiguraciju ponovimo za ventilator i grijač kao izlaze našeg Fuzzy sistema uz odgovarajući set podataka.

```
1. // Instantiating a FuzzyOutput objects
2. FuzzyOutput *vent = new FuzzyOutput(1);
3. FuzzyOutput *heater = new FuzzyOutput(2);
4. // Instantiating a FuzzySet object
5. FuzzySet *ventOn = new FuzzySet(19, 21, 21, 23);
6. FuzzySet *ventOff = new FuzzySet(20, 22, 22, 24);
7. FuzzySet *heaterOn = new FuzzySet(29, 31, 31, 33);
8. FuzzySet *heaterOff = new FuzzySet(30, 32, 32, 34);
9.
10. // Including the FuzzySet into FuzzyOutput
11. vent->addFuzzySet(ventOff);
12. vent->addFuzzySet(ventOn);
13. heater->addFuzzySet(heaterOn);
14. heater->addFuzzySet(heaterOff);
15. fuzzy->addFuzzyOutput(vent);
16. fuzzy->addFuzzyOutput(heater);
```

S obzirom da se komunikacija odvija između ESP i FPGA kontrolera, grijač i ventilator su podešeni tako da imaju specifične vrijednosti koje okidaju odgovarajuće akcije na FPGA kontroleru.



Slika 35: Funkcije pripadnosti za izlaze sistema

Nakon podešavanja ulaza, izlaza, kao i seta podataka za svaki ulaz i izlaz, formiramo pravila po sljedećem principu:

```

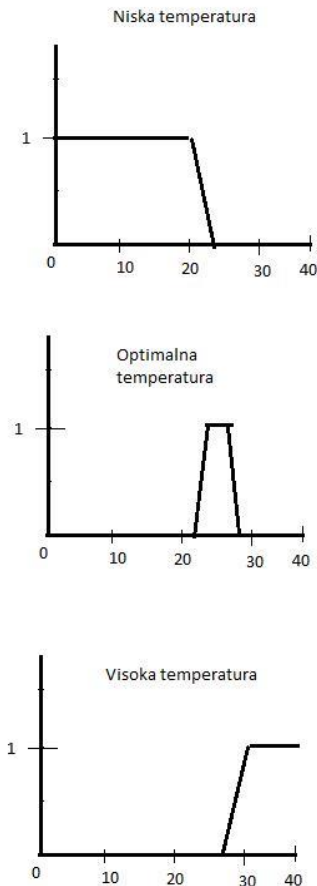
1. // Instantiating a FuzzyRuleAntecedent objects
2. FuzzyRuleAntecedent *ifTemperatureLow = new FuzzyRuleAntecedent();
3. // Creating a FuzzyRuleAntecedent with just a single FuzzySet
4. ifTemperatureLow->joinSingle(low);
5. // Instantiating a FuzzyRuleConsequent objects
6. FuzzyRuleConsequent *thenVentOff = new FuzzyRuleConsequent();
7. // Including a FuzzySet to this FuzzyRuleConsequent
8. thenVentOff->addOutput(ventOff);
9. // Instantiating a FuzzyRule objects
10. FuzzyRule *fuzzyRule01 = new FuzzyRule(1, ifTemperatureLow, thenVentOff);
11. // Including the FuzzyRule into Fuzzy
12. fuzzy->addFuzzyRule(fuzzyRule01);

```

Pravila koja se koriste u slučaju kreiranja fuzzy regulatora za sistem pametne kuće su:

- IF (temperatura je niska) THEN (ventilator je ugašen).
- IF (temperatura je optimalna) THEN (ventilator je ugašen).
- IF (temperatura je visoka) THEN (ventilator je upaljen).
- IF (temperatura je niska) THEN (grijač je upaljen).
- IF (temperatura je optimalna) THEN (grijač je ugašen).
- IF (temperatura je visoka) THEN (grijač je ugašen).

Optimalna temperatura je uobičajno definisana u vrijednosti od 25 °C. Korisnik ima mogućnost pomoću web ili mobilne aplikacije podesiti temperaturu na željenu vrijednost. U slučaju promjene temperature, novi set pravila se kreira shodno željenoj vrijednosti. Ukoliko je željena temperatura 25 °C, funkcije pripadnosti poprimaju sljedeći oblik:



Slika 36: Funkcije pripadnosti za sistem pametne kuće

Nakon dodavanja željenih pravila za ovaj sistem, čitamo vrijednosti temperature senzora, na osnovu čega uz pomoć pravila vršimo fuzifikaciju. Da bi dobili izlaznu vrijednost, izvršena je defuzifikacija sa defuzzify metodom na osnovu čega šaljemo vrijednosti na FPGA razvojnu ploču koja upravlja grijačem i ventilatorima. Temperatura se mijenja i postupak se ponavlja dok se ne postigne optimalna/željena vrijednost temperature. Pri svakoj promjeni izlazne vrijednosti grijača i ventilatora, vrijednost se proslijeđuje na FPGA uređaj korištenjem UART komunikacije.

U slučaju da želimo postaviti optimalnu temperaturu na vrijednost 27 °C, dinamički se kreira instanca fuzzy objekta na osnovu kojeg se formira FuzzySet sa lingvističkim vrijednostima low(nisko), optimal(srednje,optimalno), high(visoko). Optimalna temperatura se tada kreće u rangu od 25-29 °C. Temperatura ispod 25 °C je definisana kao niska dok temperatura iznad 29 °C je definisana kao visoka temperatura. U ovisnosti od našeg ulaza koji dobijamo mjerenjem vrijednosti na senzoru, temperatura će poprimiti jedno od prethodnih vrijednosti na osnovu čega će se definisanim pravilima utvrditi potreba za paljenjem/gašenjem ventilatora ili grijača respektivno.

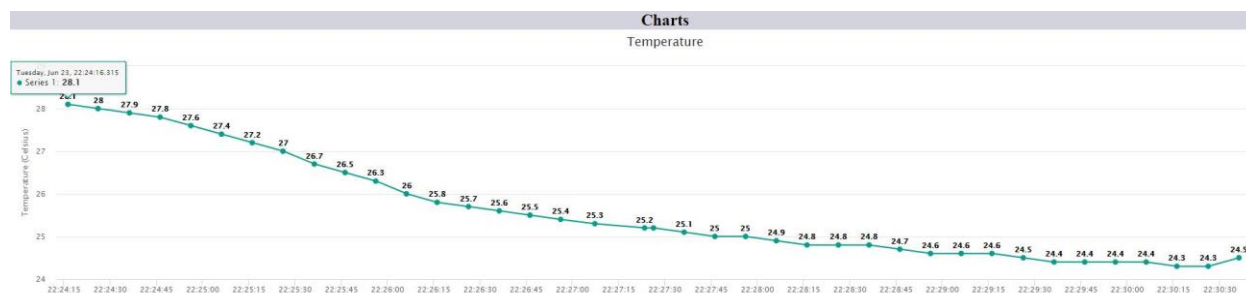
Ukoliko smo zadali optimalnu temperaturu od 27 °C dok je unutrašnja temperatura 23 °C, pravilo će se aktivirati da se grijač uključi kako bi povećali temperaturu na optimalnu vrijednosti.

Ovakav linearni sistem upravljanja svakih 5 sekundi provjerava optimalno stanje. Kako je riječ o maketi manjih dimenzija i fenu koji služi kao grijač, potrebno je nekoliko sekundi da se postigne željena temperatura. Ovakav odziv iako nije idealan, osigurava minimalan preskok pri postizanju zadane temperature.



Slika 37: Primjer1 - Odziv sistema na promjenu optimalne vrijednosti

Nakon postizanja željene vrijednosti, ukoliko želimo da optimalnu temperaturu smanjimo na vrijednost 23 °C, dobijamo sljedeći odziv:



Slika 38: Primjer2 - Odziv sistema na promjenu optimalne vrijednosti

Uočavamo da sistem shodno elementima iskorištenim treba duži period za postizanje željene vrijednosti ali istu adekvatno zadržava sa minimalnom greškom.

6 Softverska implementacija

U nastavku je opisana softverska implementacija pametne kuće korištenjem sljedećih programskih jezika:

- HTML, CSS, Javascript za kreiranje web stranice
- C++ uz Arduino IDE za kontrolu ESP32 uređaja
- Verilog za upravljanje FPGA i povezanim aktuatorima
- PHP i MySQL za kreiranje baze podataka i slanje podataka putem HTTP zahtjeva

HTML, CSS i Javascript za kreiranje web stranice su prilagođeni upotrebi za osobne računare kao i za mobilne uređaje.

6.1.1 Upravljanje Arduino ESP32 uređajem

Arduino ESP32 uređaj je iskorišten za mjerenje temperature, kreiranje Web servera, Fuzzy upravljanje, UART komunikacija sa FPGA kontrolerom kao i slanje HTTP zahtjeva koji sadrže trenutne vrijednosti unutrašnje i vanjske temperature na osnovu kojih se putem PHP i MySQL baze formiraju grafovi te osigurava monitoring/kontrola pametne kuće.

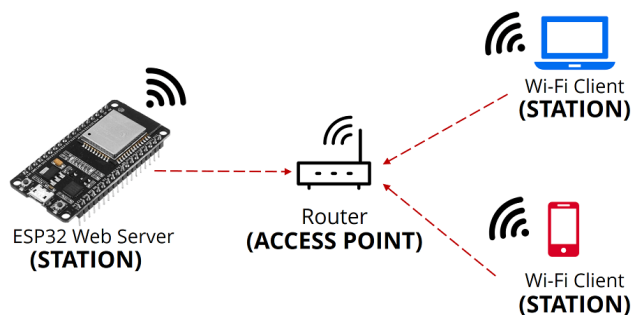
Mjerenje temperature se vrši na ESP32 kontroleru koji omogućava Wi-Fi / bežičnu konekciju sa mobilnim uređajem i desktop računarom. S obzirom da FPGA kontroler ne sadrži analogno-digitalni konvertor, mjerenje temperature korištenjem DHT senzora je izvršeno korištenjem ESP32 uređaja.

Kreiranje Web servera je izvršeno korištenjem ESPAsyncWebServer.h biblioteke koja omogućava formiranje servera na lokalnoj mreži uz dvosmjernu komunikaciju. Lokalni server je otvoren na portu broj 80. ESP32 funkcionise kao stanica, dok je ruter na koji je spojen pomenuti uređaj mrežna pristupna tačka na koji se spajaju desktop računar i mobilni aparat.

Biblioteka omogućava konfiguraciju putanje gdje server očekuje dolazeći HTTP zahtjev, te izvršavanje definisanih funkcija kada je isti primljen na specifičnoj putanji.

```
1. server.on("/temperature", HTTP_GET, [](AsyncWebServerRequest * request) {  
2.     request->send_P(200, "text/plain", readDHTTemp().c_str());  
3. });
```

Kada server dobije zahtjev na URL koji sadrži dio /temperature, šalje zahtjev klijentu da se procesuirala definisana funkcija readDHTTemp koja očitava trenutno stanje temperaturnog senzora. Rezultat se vraća web serveru u cilju vizualne reprezentacije trenutne temperature unutar i izvan pametne kuće. Ovakav asinhroni web server omogućava ažuriranje podataka u stvarnom vremenu (eng. real time).



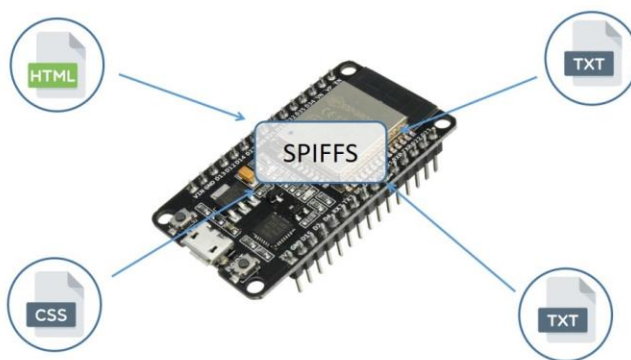
Slika 39: ESP32 kao Web server

Web server je izgrađen korištenjem Javascript, HTML i CSS tehnologija koje osiguravaju vizuelnu reprezentaciju i dvosmjernu interakciju sa ESP32 Arduino uređajem. U cilju osiguravanja portabilnosti, serijsko periferno sučelje flash sistema (eng. Serial Peripheral Interface Flash File System – SPIFFS) omogućava pohranjivanje datoteka na ESP32.

SPIFFS omogućava pristup flash memoriji koja naizgled liči na pohranjivanju datoteka na personalnom računar u odgovarajuće limitacije. Moguće je čitati, pisati, zatvarati i brisati datoteke. Korištenje SPIFFS sa ESP32 pločom je korisno za:

- Kreiranje konfiguracijskih datoteka sa prethodno podešenim postavkama sistema.
- Trajno pohranjivanje podataka
- Spremanje manje količine podataka bez upotrebe microSD kartice.
- Spremanje HTML i CSS datoteka za formiranje Web Servera
- Spremanje slika, figura, ikona i slično

Uobičajno kreiranje servera se vrši spremanjem cjelokupnog HTML koda u string varijablu. Realizacija upotrebom SPIFFS omogućava pisanje HTML, CSS i Javascript koda na zasebne fajlove kao i njihovo spremanje na ESP32 sistemu čime se osigurava skalabilnost i fleksibilnost projekta.

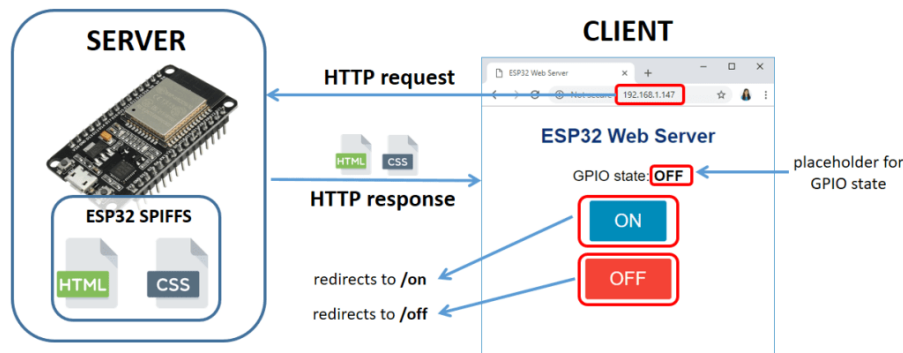


Slika 40: ESP32 i SPIFFS

Način implementacije WebServera zajedno sa SPIFFS je sljedeći:

- ESP32 pokreće kod web servera baziranog na ESPAsyncWebServer biblioteci.

- HTML, CSS i Javascript fajlovi su spremljeni na ESP32 SPIFFS.
- Prilikom zahtjeva za specifični URL korištenjem web preglednika, ESP32 odgovara sa traženim datotekama.
- Prilikom aktiviranja **ON** dugmeta, preusmjereni smo do izvorne putanje (eng. URL) sa sufiksom **/on** i LED dioda je upaljena
- Prilikom aktiviranja **OFF** dugmeta, preusmjereni smo do izvorne putanje (eng. URL) sa sufiksom **/off** i LED dioda je upaljena



Slika 41: Interakcija servera i klijenta

Svako dugme na web stranici je povezano sa odgovarajućim HTTP zahtjevom kojim se izvršava jedna od funkcionalnosti pametne kuće – paljenje diode, ventilatora, regulacije, grijalice i slično. Funkcionalnost pametne može se objasniti na primjeru aktivacije alarmnog sistema pametne kuće predstavljenog na sljedećem primjeru:

- Korisnik pritisne odgovarajuće dugme a koje je namijenjeno za aktivaciju sistema
- Kreira se novi HTTP zahtjev za obradu na osnovu željenog inputa od strane korisnika. HTTP zahtjev napisan u javascript formatu ima sljedeći oblik:

```

function turnAlarmSystem(element) {
  var xhr = new XMLHttpRequest();
  const alarmSystemText = document.getElementById('alarmSystemText');
  if (element.checked) {
    xhr.open("GET", "/alarmssystem?alarm=" + 3, true);
    alarmSystemText.innerHTML = 'Activated';
    alarmSystemText.classList.add('active-text');
  } else if (!element.checked) {
    xhr.open("GET", "/alarmssystem?alarm=" + 4, true);
    alarmSystemText.innerHTML = 'Deactivated';
    alarmSystemText.classList.remove('active-text');
  }
  xhr.send();
}

```

- HTTP GET zahtjev se šalje sa odgovarajućom predefinisanim vrijednošću. ESP32 razvojna ploča i server obrade zahtjev, te putem UART komunikacije prosljede definisan broj za datu aktivnost.
- FPGA razvojna ploča primi zahtjev, na osnovu switch kondicionih uslova odredi koji aktuator se aktivira.
- Aktuator aktiviran, proces završen.

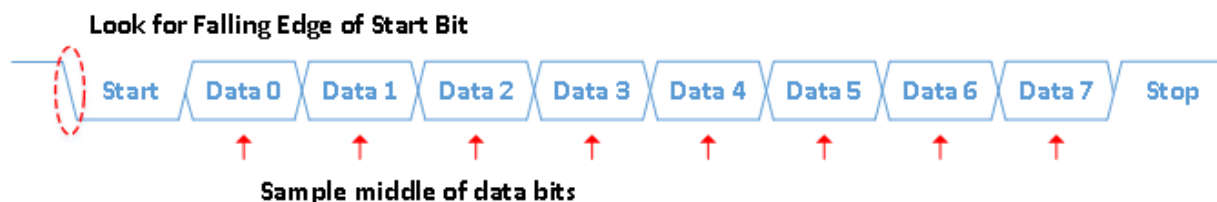
6.1.2 Upravljanje FPGA Cyclone IV uređajem

FPGA Cyclone IV uređaj zbog svojih specifikacija omogućava povezivanje različitih aktuatora uz mogućnost paralelnog izvršavanja različitih funkcija pametne kuće. Brzina, fleksibilnost i skalabilnost ovog uređaja predstavljaju idealnu kombinaciju sa ESP32 pločom za formiranje pametne kuće koja predstavlja uvid u mogućnost realizacije svih njenih elemenata na praktičnom primjeru. U tu svrhu, FPGA Cyclone IV ploča je iskorištena za pokretanje alarmnog sistema, paljenje dioda, aktiviranje ventilatora i grijača. Ovakva realizacija pruža opciju nadogradnje i implementacije raznovrsnih alata za postizanje unikatnog dojma pametne kuće koja zadovoljava sve potrebe krajnjeg korisnika. UART komunikacija omogućava dvosmjernu razmjenu podataka dva uređaja (Arduino ESP32 i FPGA Cyclone IV) čime se na osnovu zahtjeva krajnjeg korisnika pritiskom tastera na grafičkom sučelju prethodno opisane web stranice šalje poruka na FPGA uređaj za izvršavanje date akcije.

Funkcionalnost na ovoj ploči je izvršena korištenjem Verilog programskog jezika. Nakon prvobitnog podešavanja aktuatora na početno stanje, korištenjem if/else naredbi u ovisnosti od informacija dobivenih preko UART komunikacije od ESP32 uređaja izvršavaju se zadane aktivnosti.

6.1.3 Arduino i FPGA komunikacija

Osnova upravljanja pametne kuće zasniva se na UART komunikaciji između ESP32 i FPGA razvojnih ploča. Glavna uloga ovakve komunikacije je slanje i primanje serijskih podataka. Podaci se kreću od Tx pina transmitujućeg uređaja prema RX pinu primajućeg uređaja. S obzirom da UART vrši asinhroni prenos podataka, to podrazumjeva odsustvo taktnog signal za sinhronizaciju. Prilikom ove komunikacije, transmisija zajedno sa paketom podataka šalje tzv. start i stop bite koji definišu kraj i početak transmisije. Kada UART detektuje početni/startni bit, započinje sa čitanjem podataka na odgovarajućoj frekvenciji poznatoj kao baud rate – brzina prijenosa podataka [24].



Slika 42: UART komunikacija

Brzina prijenosa podataka je predstavljena u broju bita po sekundi te je u ovom slučaju odabrana vrijednost 9600. ESP32 uređaj koristi biblioteku `HardwareSerial.h` putem koje otvorimo drugi port kreirajući novi objekat ove biblioteke. Objekat se iskoristi za direktno slanje podataka na FPGA uređaj. Cyclone IV sadrži par modula kojima se definiše brzina prijenosa podataka (eng. baud rate) kao i moduli za primanje i transmisiju podataka [24].

Zbog izrazito velike brzine FPGA ploče, zahtjev je obrađen bez vidljivog gubitka vremena. Na ovaj način funkcionišu ostali elementi koji se nalaze na centralnom dijelu. Za svaki element je unaprijed predefinisano brojevi koji odgovaraju određenoj akciji i aktuatorsu za koji je ta akcija vezana. Verilog kod koji obrađuje neke od zahtjeva poslanih od strane ESP32 razvojne ploče ima sljedeći izgled:

```
1. always @(posedge clk)
2. begin
3.   case (rxdata)
4.     3: begin
5.         alarmSystem = 1'b1;
6.       end
7.     4: begin
8.         alarmSystem = 1'b0;
9.       end
10.    7: begin
11.        kitchenlight = 1'b1; // palimo kitchenlight
12.      end
13.    8: begin
14.        kitchenlight = 1'b0;
15.      end
16.   endcase
17. end
```

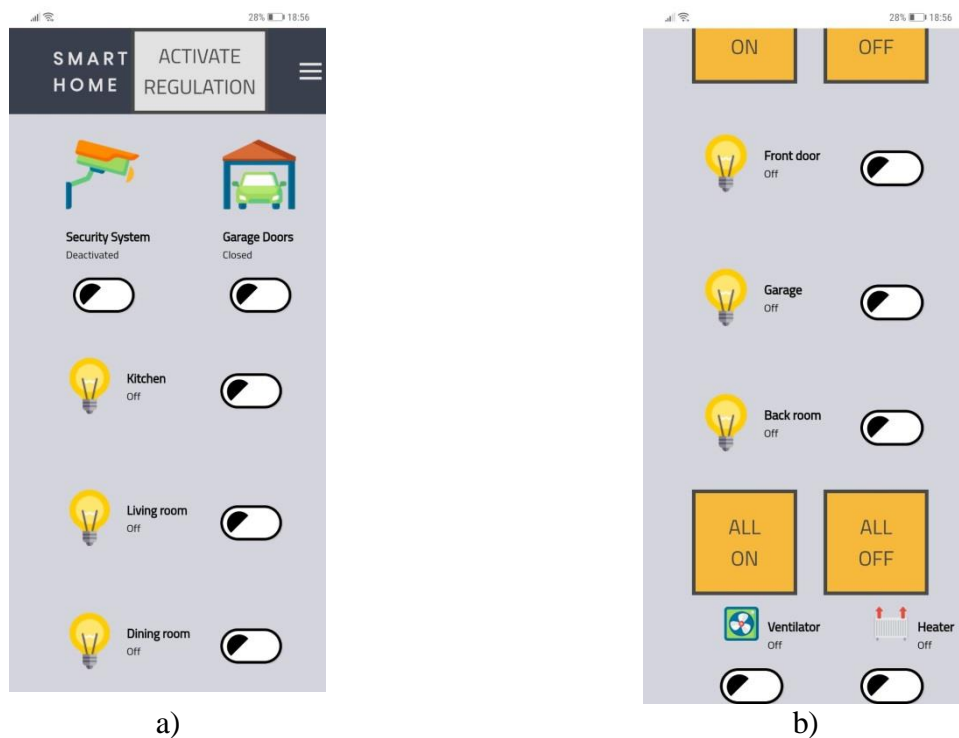

6.2 Navigacija web stranicom pametne kuće

Navigiranje web stranicom pametne kuće je moguće izvršiti na dva načina: korištenjem mobilnog uređaja ili personalnog računara.



Slika 43: Smart Home web stranica na računaru

Mobilna verzija je adaptirana manjem ekranu te prva stranica poprma sljedeći izgled:



Slika 44: Mobilna verzija aplikacije

Da bi osigurali kompatibilnost web stranice na različitim uređajima, potrebno je iskoristiti tehnologije poput HTML, Javascript i CSS. HTML kao prezentacijski jezik za izradu web stranice je osigurao postavljanje elemenata na željene pozicije, kreiranjem zasebnih dijelova kojima možemo upravljati i doradivati. CSS kao stilski jezik je poslužio za uređivanje prethodno spomenutih dijelova, doradivanje, pozicioniranje i oblikovanje HTML elemenata. Na ovaj način, upotrebom HTML i CSS tehnologija smo vizuelno zamišljen izgled web stranice pretvorili u stvarnost podešavajući izgled i raspored stranice potrebama i željama korisnika. Funkcionalnost stranice poput aktiviranja različitih tastera, slanje HTTP zahtjeva kao i real-time prikazivanje temperature i vlage bez potrebe za manuelnim 'osvježavanjem' stranice radi procesiranja informacija očitanih sa temperaturnih senzora.

Web stranica je sastavljena od tri ključna dijela:

- Navigacioni dio na vrhu
- Bočna traka sa lijeve strane
- Centralni dio

Navigacioni dio na vrhu je sastavljen od logotipa pametne kuće – SMART HOME uz dugme za aktivaciju automatske regulacije kao i mogućnosti navigiranja ostalih web stranica na web serveru.



Slika 45: Pametna kuća - navigacioni dio

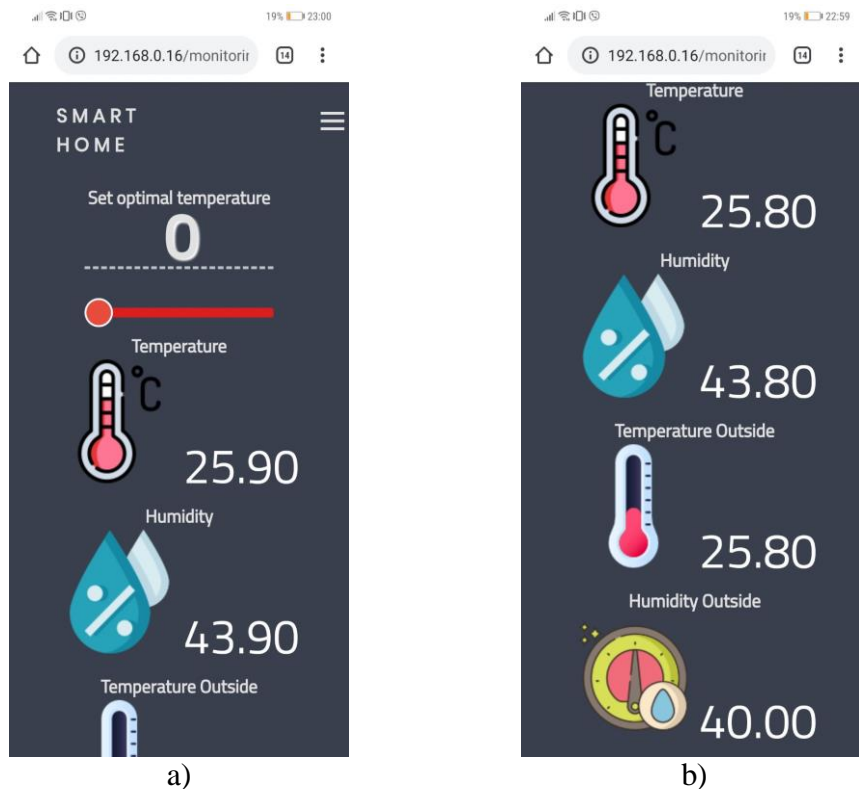
Data i Monitoring u gornjem dijelu je poveznica za web stranicu gdje se vrši nadgledanje pametne kuće čija je funkcionalnost detaljnije opisana u sljedećem poglavlju. Mobile poveznica prikazuje bočnu traku koja sadrži trenutne vrijednosti temperature i vlage u mobilnom obliku.

Bočna traka (eng. side-bar) sa lijeve strane sadrži trenutnu vrijednost temperature koja je praćena real-time promjenama trenutnog stanja temperature i vlažnosti koje unutrašnji senzori DHT očitavaju. Javascript kod zadužen za automatsku sinhronizaciju promjena na web stranici bez potrebe za manuelnog osvježavanja (eng. refresh) stranice je baziran na funkciji setInterval().

```
1. setInterval(function ( ) {
2.     var xhttp = new XMLHttpRequest();
3.     xhttp.onreadystatechange = function() {
4.         if (this.readyState == 4 && this.status == 200) {
5.             var x = (new Date()).getTime(),
6.                 y = parseFloat(this.responseText);
7.             console.log(x);
8.             document.getElementById("temperatureID").classList.add('transition-text-pre-
           animation');
9.             setTimeout(function ( ) {
10.                 document.getElementById("temperatureID").classList.remove('transition-text-
               pre-animation');
11.             }, 1000)
12.             document.getElementById("temperatureID").innerHTML = this.responseText;
13.             console.log(this.responseText);
14.         }
15.     };
16.     xhttp.open("GET", "/temperature", true);
```

```
17. xhttp.send();  
18. }, 40000 ) ;
```

Svakih 40 sekundi, HTTP GET zahtjev sa ključnom riječi 'temperature' je proslijeđen na web server gdje se izračunava temperatura na arduinu te proslijeđuje povratna informacija kojom se na web stranici ažurira trenutno stanje. Specifične tranzicije i animacije teksta koji se ažurira su obrađene dodavanjem prethodno generisanim klasama.



Slika 46: Bočna traka aplikacije pametne kuće na mobilnom uređaju

Centralni dio web stranice se sastoji od nekoliko tastera koje je moguće aktivirati u cilju otvaranja garažnih vrata, paljenja svjetla unutar i izvan kuće, aktiviranja alarmnog sistema kao i ventilatora i grijača.



Slika 47: Centralni dio aplikacije

Svaki taster daje vizuelnu povratnu informaciju koja korisna obavještava da li je željena akcija izvršena. Pritiskom na željeni taster, javascript uz otvaranje HTTP zahtjeva ostvaruje konekciju sa web serverom koji je uspostavljen uz pomoć arduina čime se u ovisnosti od zadane aktivnosti izvršavaju specifične operacije.

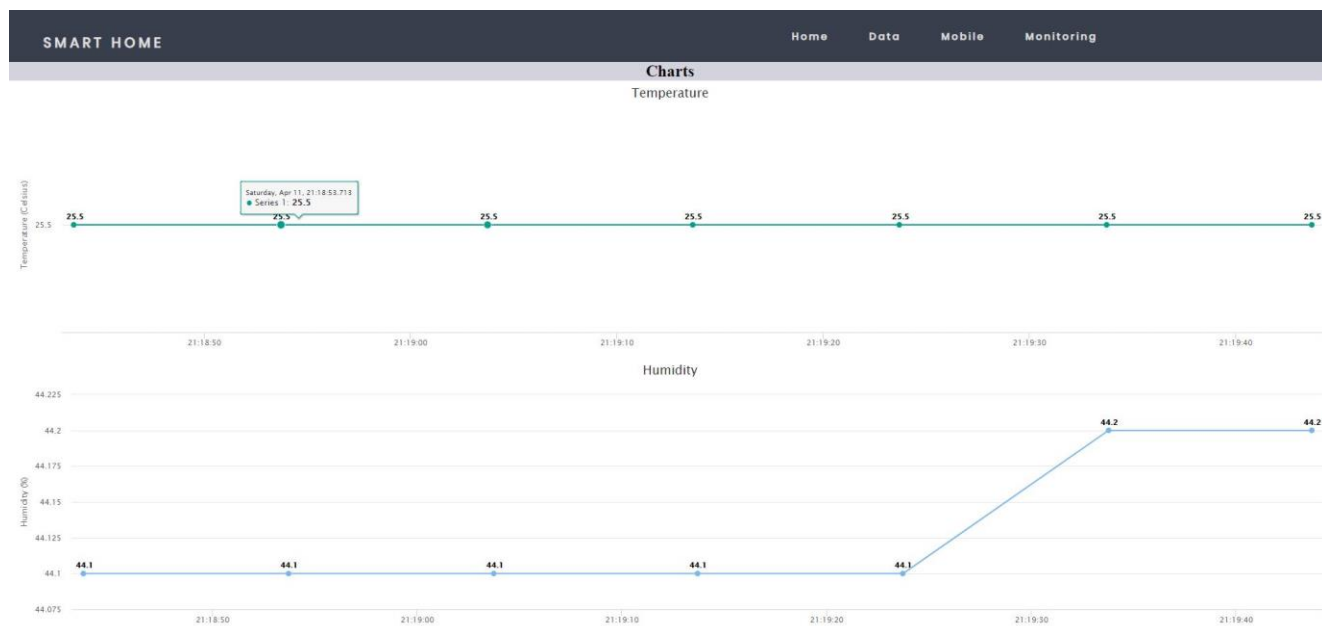
6.3 Monitoring pametne kuće

Monitoring pametne kuće se bazira na mogućnostima vizuelnog praćenja temperature i vlažnosti zraka unutar i izvan kuće. Ovo postizemo upotrebom PHP i MySQL pored već prethodno pomenutih tehnologija na osnovu kojih vizualiziramo trenutno stanje temperature i vlažnosti unutar i izvan kuće.

Pametna kuća nudi par mogućnosti monitoringa:

- Real-time monitoring stanja temperature i vlažnosti na bočnoj traci
- Monitoring na 'Data' web stranici upotrebom Javascript, HTML i CSS
- 'Monitoring' web stranica bazirana na PHP i MySQL

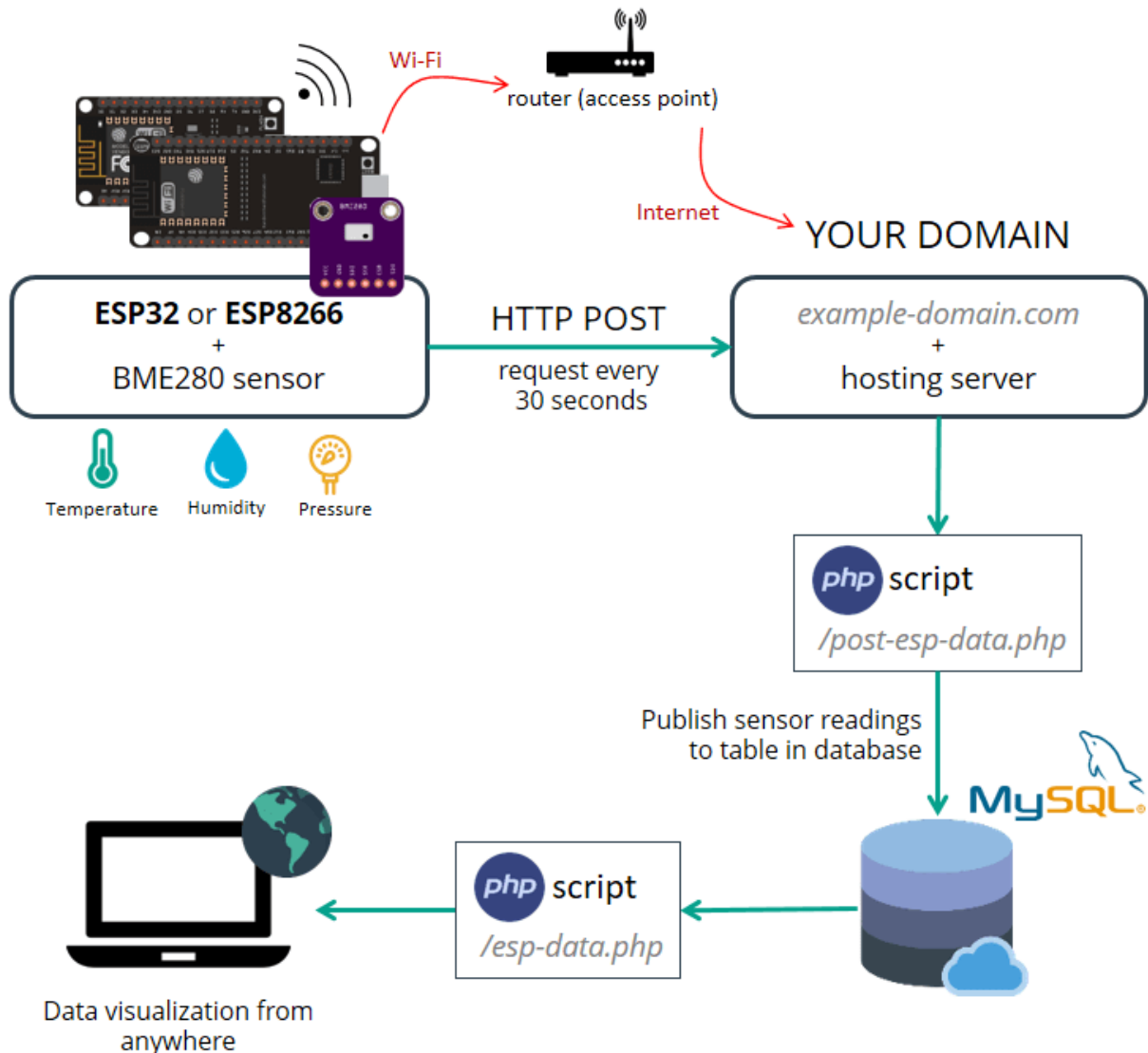
Pored nadgledanja (eng. monitoring) koje možemo da posmatramo na bočnoj traci glavne stranice, trenutne grafike o unutrašnjosti temperature i vlažnosti zraka kao i njihove promjene je moguće pregledati pod 'Data' dijelom web stranice.



Slika 48: Smart Home – Data stranica

Ovakva vizualizacija je postignuta korištenjem HTML-a za podešavanje inicijalnih elemenata grafova, CSS stilskog jezika za vizualnu reprezentaciju, odabir boje i poziciju kao i javascript programskog jezika za formiranje grafikona pomoću highcharts skripte. setInterval() funkcija formira HTTP zahtjeva, provjerava vrijednost temperature i vlažnosti te povratnu informaciju dobivenu od senzora zaduženih za mjerenje istih ispisuje [25].

Ukoliko je potreban pregled stanja temperature i vlažnosti unutar i izvan kuće u prethodnom periodu, 'Monitoring' stranicu nam daje historijski prikaz promjena ova dva elementa.



Slika 49: Interakcija servera sa bazom podataka

Za ovakvu realizaciju, prvobitno je neophodno kreirati bazu podataka u koju ćemo spremati historiju promjena temperaturnog stanja. Nakon generisanja vlastite baze podataka, upotrebom MySQL generišemo tabelu koja sadrži pet proizvoljno definisanih kolona kolona: id, value1, value2, value3, value4, reading_time [25]































Arduino kod zadužen za slanje HTTP POST zahtjeva sadrži zaglavlje (eng. header) sa imenama kolona koje smo prethodno definisali kao i vrijednosti koje prosljeđujemo istim. U ovom slučaju, value1 i value 2 sadrže vrijednosti unutrašnje temperature i vlažnosti respektivno dok value3 i value4 sadrže trenutne vanjske vrijednosti ovih elemenata.

```
1. // Specify content-type header
2. http.addHeader("Content-Type", "application/x-www-form-urlencoded");
3. // Prepare your HTTP POST request data
4. String httpRequestData = "api_key=" + apiKeyValue + "&value1=" + String(InDHT.readTemperature())
```

```

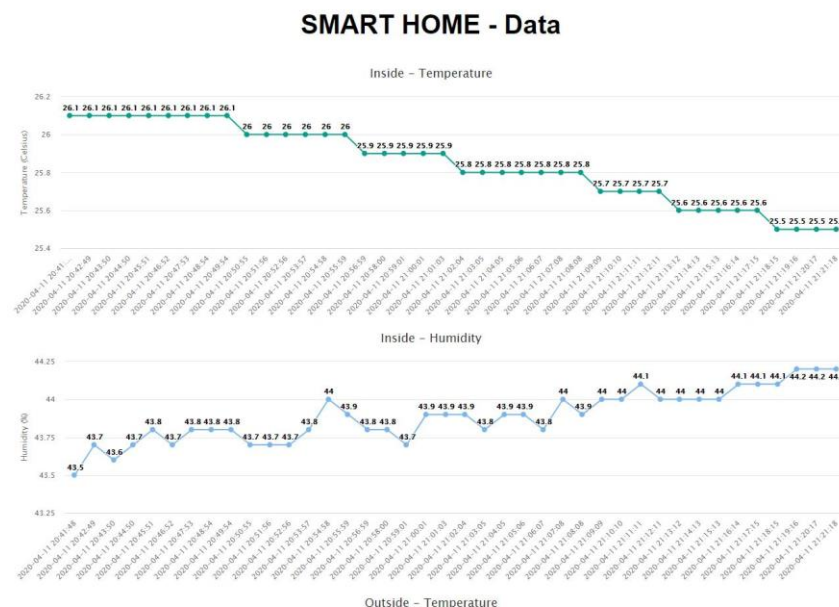
5.         + "&value2=" + String(InDHT.readHumidity()) + "&value3=" + Str
    ing(OutDHT.readTemperature()) + "&value4=" + String(OutDHT.readHumidity()) + "";
6. // Send HTTP POST request
7. int httpResponseCode = http.POST(httpRequestData);

```

| ←T→ | | | | ▼ | id | value1 | value2 | value3 | value4 | reading_time |
|--------------------------|--|--|--|---|------|--------|--------|--------|--------|---------------------|
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3812 | 25.40 | 44.30 | 25.30 | 41.00 | 2020-04-11 21:25:21 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3811 | 25.50 | 44.40 | 25.30 | 41.00 | 2020-04-11 21:24:20 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3810 | 25.50 | 44.20 | 25.40 | 41.00 | 2020-04-11 21:23:19 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3809 | 25.50 | 44.10 | 25.30 | 41.00 | 2020-04-11 21:22:18 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3808 | 25.50 | 44.20 | 25.40 | 41.00 | 2020-04-11 21:21:18 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3807 | 25.50 | 44.20 | 25.40 | 41.00 | 2020-04-11 21:20:17 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3806 | 25.50 | 44.20 | 25.40 | 41.00 | 2020-04-11 21:19:16 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3805 | 25.50 | 44.10 | 25.40 | 41.00 | 2020-04-11 21:18:15 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3804 | 25.60 | 44.10 | 25.50 | 41.00 | 2020-04-11 21:17:15 |
| <input type="checkbox"/> |  Edit |  Copy |  Delete | | 3803 | 25.60 | 44.10 | 25.50 | 41.00 | 2020-04-11 21:16:14 |

Slika 50: Pohrana podataka u tabelu

Nakon prosljeđivanja zahtjeva, tabela koju smo prethodno kreirali se popunjava sa podacima koje sljedeća skripta obrađuje te formira grafove prikazane na sljedećem primjeru:



Slika 51: Smart Home - Monitoring stranica

7 Zaključak

U ovom radu predstavljena je mogućnost korištenja različitih tehnologija, alata i elemenata u cilju formiranja kompatnog sistema koji služi za izgradnju pametne kuće. Projekat „pametna kuća“ generalno gledajući, postat će vrlo primamljiv za mnoge, te u budućnosti možemo očekivati njegovu sve češću implementaciju jer ima za cilj poboljšanje kvalitete života čovjeka.

Ovakvu primjenu mnogih tehnologija na praktičnom primjeru makete moguće je realizovati u vlastitom domu, uz adekvatne aktuatore i senzore, koji bi igrali glavnu ulogu u izvođenju osnovnih operacija poput: preciznog mjerenja temperature, vlažnosti kao i paljenja ventilatora, grijača, klima uređaja i svjetla u okruženju u kojem se korisnik nalazi.

Na ovom praktičnom primjeru je pokazana mogućnost upotrebe savremenih tehnologija i embedded sistema koji su efikasni alati za kreiranje sistema pametne kuće.

Dakle, pametna kuća integriše različite sisteme koji su povezani putem određenog kontrolera a osnovni cilj je automatizacija kućnih sistema, povezivanje, kontrolisanje uvjeta boravka u kući, za vrijeme dok je korisnik u kući, ali i kada je izvan nje. Kontrolisani uvjeti, stvaranje ugodnog ambijenta i sigurnosti čine boravak u pametnoj kući primamljivim i projekti ovakvog tipa automatizaciju čine sve popularnijom i efikasnijom u svakodnevnom životu korisnika.

8 Literatura

- [1] Mohammad S. Obaidat, Petros Nikipolitis: „Smart cities and Homes Key Enabling Technologies“, Elsevier Inc. 2016.
- [2] Historija i budućnost pametne kuće: <https://blog.avira.com/the-smart-home-how-it-all-began/>
- [3] Richard Harper: „Inside the Smart Home“, Springer-Verlag 2003.
- [4] Jari Nurmi: „Processor Design System on Chip Computing for ASICs“, Springer, 2007.
- [5] Klasifikacija ugradbenih sistema: <https://www.watelectronics.com/classification-of-embedded-systems/>
- [6] Sistemi na čipu: <https://anysilicon.com/what-is-a-system-on-chip-soc/>
- [7] Komponente sistema na čipu: <https://www.dignited.com/33687/soc-vs-cpu-what-is-a-system-on-a-chip/>
- [8] Joseph Yiu, „System on Chip Design with ARM Cortex-M processors“, ARM Education Media 2019.
- [9] Arhitektura FPGA razvojne ploče: <https://towardsdatascience.com/introduction-to-fpga-and-its-architecture-20a62c14421c>
- [10] Razlika između FPGA, mikroprocesora, mikrokontrolera: <https://www.tutorialspoint.com/differences-in-microcomputer-microprocessor-and-microcontroller>
- [11] Razlika između FPGA i mikrokontrolera: <https://www.ourpcb.com/fpga-vs-microcontroller.html>
- [12] Min Li, Wenbin Gu, Wei Chen, Yeshen He, Yannian Wu, Yiying Zhang, „Smart Home: Architecture, Technologies, Systems“, 2018.
- [13] Menachem Domb: „Smart Home Systems Based on Internet of Things“, 2019.
- [14] Cyclone IV arhitektura: <https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyclone-iv/cyiv-51001.pdf>
- [15] FPGA Cyclone IV: <https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyclone-iv/cyiv-51001.pdf>
- [16] ESP32DevKit: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/hw-reference/esp32/get-started-devkitc.html>
- [17] DHT Sensors: <https://learn.adafruit.com/dht>

- [18] PIR sensors: https://en.wikipedia.org/wiki/Passive_infrared_sensor
- [19] Light-emitting diode: https://en.wikipedia.org/wiki/Light-emitting_diode
- [20] L9110 fan module: <http://arduinolearning.com/learning/basics/arduino-and-l9110-fan-module-example.php>
- [21] Lejla Banjanović – Mehmedović: „Fuzzy sistemi zaključivanja“ – Inteligentni sistemi
- [22] Fuzzy logika – uvod: <https://www.geeksforgeeks.org/fuzzy-logic-introduction/>
- [23] Fuzzy kontroleri, arduino biblioteka, funkcionalnost i upotreba: <https://blog.zerokol.com/2012/09/arduino-fuzzy-fuzzy-library-for-arduino.html>
- [24] Arduino – FPGA UART komunikacija: <https://circuits4you.com/2018/12/31/esp32-hardware-serial2-example/>
- [25] ESP32 i MySQL baza podataka - <https://randomnerdtutorials.com/esp32-esp8266-mysql-database-php/>