

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Jan Šubelj

**Izdelava modula za postavitev IP
parametrov ob vklopu**

SEMINARSKA NALOGA

BREZŽIČNA SENZORSKA OMREŽJA

MENTOR: prof. dr. Nikolaj Zimic

Ljubljana, 2018

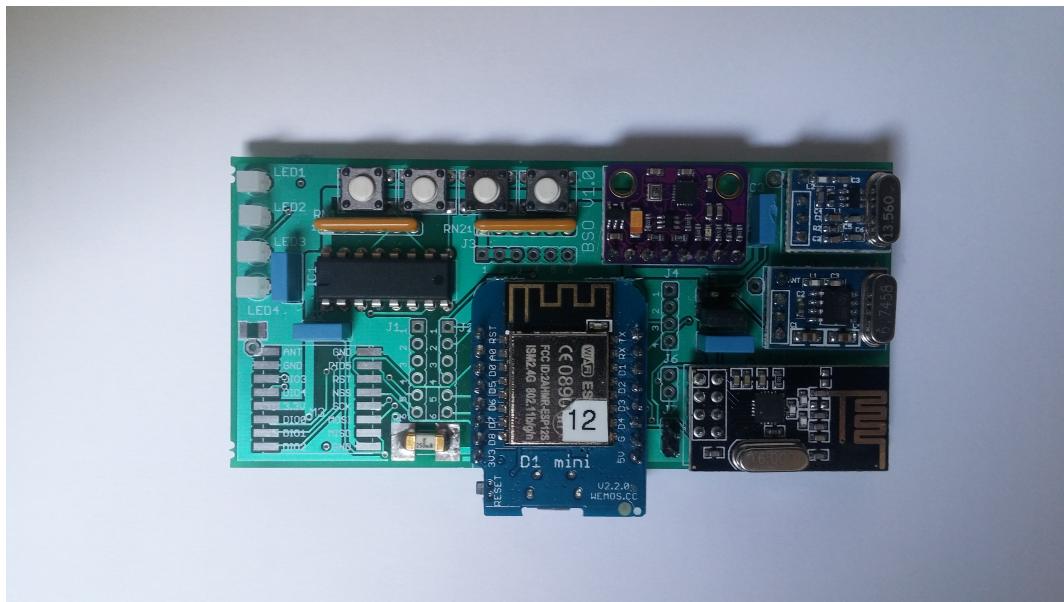
Kazalo

1	Uvod	1
2	Razvojno okolje	3
2.1	Sistem	3
2.2	Programska oprema	3
2.3	Razvojna plošča	4
3	Razvoj programa	5
3.1	Pregled	5
3.2	Sestava projekta	5
3.3	Moduli	6
4	Zaključek	13
	Literatura	13

Poglavlje 1

Uvod

Za seminar pri predmetu Brezžična senzorska omrežja smo si izbrali nalog Izdelava modula za postavitev IP parametrov ob vklopu. Motivacija za to naloge je bila zanimanje za brezžično povezovanje v lokalno mrežo ter konfiguracija WiFi modula na ESP8266. Uporabljali smo razvojno ploščo izdelano na Fakulteti za računalništvo in informatiko, BSO v1.0, ki jo lahko vidimo na sliki 1.1. Celotna koda je dostopna na [5].



Slika 1.1: BSO v1.0 z Wemos D1 mini

Poglavlje 2

Razvojno okolje

2.1 Sistem

Celotno razvojno okolje je postavljeno na virtualnem sistemu ki poganja Ubuntu 14.04 operacijski sistem. Slika sistema nam je bila dana na spletni učilnici [2]. Na njem je že skonfigurirana pot (PATH) in ostale globalne spremenljivke ter programsko ogrodje esp-open-rtos [1]. Tako lahko kodo zapišemo na krmilnik z enostavnim `make flash`.

2.2 Programska oprema

Za razvoj programa smo uporabljali tekstovni urejevalnik Visual Studio Code.

Za lažji in hitrejši razvoj smo dodali še tri aliase v Bashu:

- `alias putty='putty 2> /dev/null &'`
- `alias resetusb='sudo modprobe -r usbhid && sleep 5 && sudo modprobe usbhid'`
- `alias flash='killall putty;cd fsdata; ./makefsdata; cd ..; make && make erase_flash && make flash && putty'`

Alias `putty` je namjenjen zagonu programa Putty, v ozadju, preko katerega komuniciramo s krmilnikom preko serijske povezave. Alias `resetusb` je

namenjen resetiranju usb portov na sistemu ob primeru da sistem ne zazna krmilnika pravilno. Alias `flash` je namenjen pisanju kode na krmilnik. Izkazalo se je, da `make flash` učasih ne prepiše celotnega flash spomina, na krmilniku pa zato lahko ostanejo podatki, ki niso zaželjeni. V tem aliasu je dodano tudi prevajanje HTML kode v C kodo, o katerem bomo govorili v nadaljevanju.

Glavno programsko ogrodje je esp-open-rtos, ki nam je zelo olajšal delo razvoja, saj vsebuje knjižnice za interakcijo z moduli čipa. Prav tako vsebuje knjižnice za komunikacijo preko I2C in serijske povezave.

2.3 Razvojna plošča

Razvijali in preizkušali smo sistem na razvojni plošči izdelani na Fakulteti za računalništvo in informatiko BSO v1.0. Razvojna plošča je vidna na sliki 1.1. Glavna enota plošče je Wemos D1 mini, na katerem je ESP8266 krmilnik, ki omogoča WiFi povezavo. Wemos D1 mini vsebuje tudi gumb za resetiranje krmilnika. Na razvojni plošči so štiri led in štiri gumbi, s katerimi poteka interakcija preko PCF8574AN in I2C povezave. Na razvojni plošči so tudi ostali moduli (za temperaturo, različne brezžične komunikacije itd.), ki pa za to seminarsko nalogo niso pomembni.

Poglavlje 3

Razvoj programa

3.1 Pregled

Glavna naloga programa je, da lahko z gumbi spremojamo omrežne nastavitve krmilnika. Celotno delovanja programa smo si zamislili na naslednji način:

Krmilnik ob vklopu pregleda interni spomin flash, kjer naj bi bila shranjena konfiguracija. Če konfiguracije ni, se postavi v način dostopne točke. Če konfiguracija obstaja se postavi v nastavljen način (priklop na dostopno točko, dinamičen ali statičen način IP). Nato servira spletno stran na vratih 80. Na štirih led se prikaže trenuten omrežni način. Ves čas gleda tudi gumbe, s katerimi lahko preklopimo med načinom dostopne točke ali naprave. Na krmilniku teče tudi telnet strežnik preko katerega lahko nastavimo konfiguracijo WiFi.

V veliko pomoč pri razvoju so nam bili primeri esp-open-rtosa ter dokument z opisanimi vsemi funkcijami, ki jih podpira programsko ogrodje[4].

3.2 Sestava projekta

Celotni projekt je razdeljen modularno, na manjše .c in .h datoteke. Vsaka datoteka je namenjena določeni funkciji. Za tak način smo se odločili ker

je celotni projekt bolj pregleden, razvoj pa zato posledično hitrejši in lažji. Datoteke so sledeče:

- buttons.c/h - interakcija z gumbi
- leds.c/h - interakcija z led
- http_server.c/h - Http strežnik
- i2c.c/h - interakcija preko I2C
- ip_flash_storage.c/h - shranjevanje/branje WiFi konfiguracije
- read_write_flash.c/h - funkcije za pisanje in branje iz flash spomina
- telnet_server.c/h - telnet strežnik
- commands.h - ukazi za telnet strežnik
- utils.c/h - knjižnica z orodji (npr. za čakanje)
- wifi.c/h - Konfiguracija WiFi
- main.c - Vsebuje `user_init` funkcijo

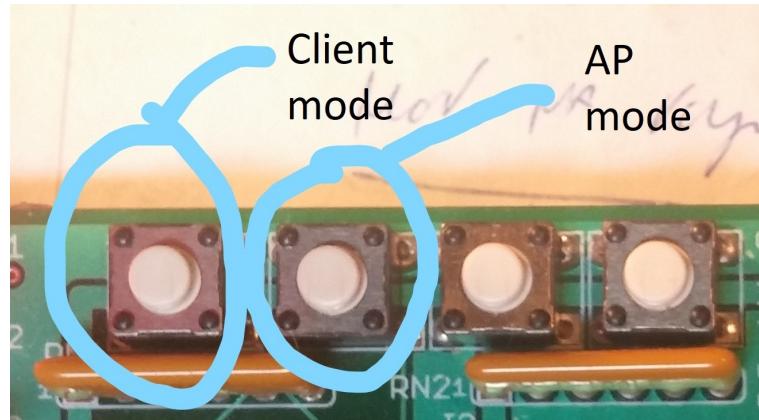
Projekt vsebuje tudi mapo fsdata, ki vsebuje html kodo strežnika ter program, ki jo prevede v C kodo ki je razumljiva krmilniku.

3.3 Moduli

V tem razdelku bomo opisali vse pomembne module, ki sestavljajo končno rešitev.

3.3.1 I2C modul

Ta modul vsebuje naslov PCF8574AN in številke pinov na katerih se nahaja I2C linija. Vsebuje tudi `init_i2c` funkcijo, ki inicializira I2C linijo. Funkcije za pisanje in branje na in iz I2C linije so vključene že v programskem ogrodju.



Slika 3.1: Gumba za nastavljanje WiFi načina

3.3.2 Utils

Ta majhen modul vsebuje zakasnitveni funkciji `os_delay_ms` in `delay_ms`. Funkcija `delay_ms` je namenjena zakasnitvi v RTOS tasku, medtem ko se funkcijo `os_delay_ms` lahko kliče tudi izven taska. Priporočljiva je uporaba `delay_ms`, saj ta ne zaustavlja delovanja procesorja ampak preklopi na drug Task za čas zakasnitve.

3.3.3 Buttons

Buttons modul je namenjen branju gumbov. Branje gumbov je izvedeno na poll način preko I2C povezave in se izvaja v `task_read_buttons` tasku. Ob pritisku na gumb `task_read_buttons` ustvari nov task, v katerem je koda za odziv na pritisk, ko se ta koda izvede se task izbriše.

Če je pritisnjen prvi gumb se kliče funkcija `set_AP_mode`, iz wifi modula, s parametrom (`AP_MODE_OFF_FLAG`), če pa je pritisnjen drugi gumb se kliče funkcija s parametrom (`AP_MODE_ON_FLAG`). Indikator, da se je nastavil željen način delovanja, so led. Za interakcijo z led se kliče funkcija `flash_leds` s parametrom `ALL_LEDS` in `NO_ERROR_LEDS`, če pa je prišlo do napake se uporabi parameter `ERROR_LEDS`. Gumba lahko vidimo na sliki 3.1.

3.3.4 Leds

Modul leds vsebuje funkcije za prižiganje in ugašanje led. Poleg podpornih funkcij (ki skrbijo za pravilno prižiganje, ugašanje, dodajanje itd.) vsebuje funkcijo `void flash_leds(uint8_t leds_state, char err)` s katero, če je `err` enak `NO_ERROR_LEDS` dvakrat počasi utripnejo ledice, ki so nastavljene v `leds_state`, če pa je `err` enak `ERROR_LEDS`, potem trikrat hitro utripnejo.

3.3.5 Read write flash

Modul je namenjen nizko nivojskemu pisanju v flash spomin, kot je opisano v [3]. Piše v sektor 0x01 na flashu, kjer je namenjen prostor za trajno shranjevanje podatkov programa. Ob pisanju se pobriše celoten sektor nato pa se na 124 bajt v sektorju začnejo zapisovati podatki. 124 bajtni zamik je potreben, saj smo opazili da se bajti med 0 in 124 naslovom lahko spreminja ob resetu.

3.3.6 Ip flash storage

Namen modula je shranjevanje in branje WiFi konfiguracije v flash s pomočjo modula opisanega v razdelku 3.3.5. Celotna konfiguracija je shranjena v strukturi, ki vsebuje:

- `ip` - IP naslov krmilnika [4B]
- `netmask` - omrežna maska [4B]
- `gw` - IP naslov omrežnega usmerjevalnika [4B]
- `ssid` - ime dostopne točke [32B]
- `pass` - geslo dostopne točke [64B]
- `dhcp_static` - zastavica ali želimo statičen ali dinamičen ip naslov [1B]
- `ap_mode` - zastavica če želimo način dostopne točke ali naprave [1B]

- `does_config_exist` - zastavica če obstaja konfiguracija [2B]

Celotna konfiguracija je velika 104B. Zastavica `does_config_exist` je velika 2B, ker mora biti velikost bajtnega seznama, ki se shrani v flash deljiv z 4.

Ključne dve funkciji modula sta branje in pisanje v flash. Pisanje je narejeno tako, da se alocira 104B seznam, nato pa se vanj shrani celotna konfiguracija. Nato se kliče funkcija za pisanje v flash, ki celotni seznam shrani. Branje je narejeno ravno obratno, s tem da nam je kazalec do alociranega seznama podan.

Poleg teh dveh funkcij modul vsebuje tudi funkcije ki spreminja samo določene dele strukture. Spisali pa smo tudi funkcijo ki serializira celotno strukturo v niz ki je primeren prikazu.

3.3.7 Telnet server

Telnet je glavni uporabniški vmesnik za spreminjanje WiFi konfiguracije. Dovoljenih je 5 hkratnih povezav. V tem modulu so definirane vsi ukazi, ki omogočajo uporabniku pregledovanje in spreminjanje WiFi konfiguracije. Ukazi so sledeči:

- `ip [ip]` - nastavitev IP naslova
- `gw [ip]` - nastavitev naslova omrežnega usmernika
- `netmask [ip]` - nastavitev omrežne maske
- `ssid [name]` - nastavitev imena dostopne točke
- `pass [pass]` - nastavitev gesla dostopne točke
- `static` - postavljanje zastavice za statičn IP naslov
- `dynamic` - postavljanje zastavice za dinamičn IP naslov
- `save` - shranjevanje trenutnih nastavitev v flash spomin

- `help` - prikaže pomoč
- `display_current` - izpiše trenutno (to ki jo urejamo) konfiguracijo
- `display_save` - izpiše shranjeno (ta ki se trenutno izvaja) konfiguracijo
- `exit` - prekinitev telnet povezave
- `restart` - ugasne in ponovno prižge krmilnik (soft reset)

3.3.8 Wifi modul

Wifi modul je namenjen nastavljanju WiFija glede na shranjeno konfiguracijo. Močno sodeluje z Ip flash storage-om opisanim v [?]. Modul ima glavno funkcijo `wifi_config`, ki pogleda v flash spominu kakšen način je nastavljen. Nato kliče ali funkcijo za dostopno točko ali pa za način naprave. Te dve funkciji poskrbita da se pravilno nastavi in poveže krmilnik na WiFi z nastavitvami zapisanimi v konfiguraciji. Modul vsebuje tudi funkcije, ki vračajo določene informacije povezane z WiFijem (npr. IP naslov v dinamičnem načinu).

Modul vsebuje tudi task, ki gleda ali je krmilnik že prišel do omrežja ali ne in temu primerno prižiga ledice. Vse štiri načine lahko vidimo na sliki 3.2

3.3.9 Http server

Na krmilniku smo postavili tudi http strežnik, ki nam servira stran s podatki o trenutnih WiFi nastavivah in slike ledic. Sama konfiguracija http serverja je enostavna, saj potrebujemo le klicati funkcijo `httpd_init` iz programskega ogrodja v Makefile pa napisati, kje so locirani HTML podatki. V esp-open-rtos primeru za postavitev http strežnika smo tudi našli program, ki vse podatke na uporabnikovi strani prevede v bajtne sezname, ki jih nato uporablja strežnik.

Stran je shranjena v `index.ssi` datoteki v jeziku HTML in vsebuje žetone za vnos podatkov strežnika (Server Side Inputs). V modulu smo zato morali



Slika 3.2: Načini WiFi (od leve proti desni): dostopna točka, dinamičen IP, statičen IP, ni povezave

implementirati pravilno analizo za žetone, ki kličejo funkcije iz Wifi modula opisanega v 3.3.8, ter tako pridobi potrebne podatke.

3.3.10 Main

Main modul je glavni modul, kjer se začne izvajanje kode na krmilniku. Vsebuje `user_init` funkcijo, v kateri so inicializirani vsi komunikacijski moduli. Nato požene taske za preverjanje povezave, branje gumbov, telnet strežnik in http strežnik.

Poglavlje 4

Zaključek

Z implementacijo smo zadovoljni, seveda pa bi se jo dalo razsiriti in zavzakovati (stabilizirati). Opazili smo da včasih mehko ponovno prižiganje ne deluje, še posebej takoj po prenosu programa na krmilnik. Naš program pa bi lahko tudi uporabili kot celostni modul na katerem bi lahko nato naprej gradili rešitve. V tem primeru bi bilo potrebno funkcijo `user_init` v `main.c` preimenovati v nekaj drugega uporabnik pa bi jo nato moral klicati ob koncu svojega `user_init`-a.

Literatura

- [1] Superhouse esp-open-rtos. Dosegljivo: <https://github.com/SuperHouse/esp-open-rtos>. [Dostopano 26.5.2018].
- [2] Virtualno razvojno okolje. Dosegljivo: <https://drive.google.com/open?id=1mKJVTiaYb-0hA186FD5hgC0dRsr38IE1>. [Dostopano 26.5.2018].
- [3] Espressif. Esp8266 flash rw operation. Dosegljivo: https://www.espressif.com/sites/default/files/documentation/99a-sdk-espressif_iot_flash_rw_operation_en_v1.0_0.pdf. [Dostopano 26.5.2018].
- [4] Espressif. Esp8266 non-os sdk api reference. Dosegljivo: https://www.espressif.com/sites/default/files/documentation/2c-esp8266_non_os_sdk_api_reference_en.pdf. [Dostopano 26.5.2018].
- [5] Jan Šubelj. Izvorna koda. Dosegljivo: https://github.com/JSubelj/bso_seminar. [Dostopano 26.5.2018].