Brezžični senzorji IV

Sistemi Daljinskega Vodenja – Laboratorijske Vaje

9. december 2011

Povzetek

V tej nalogi se bomo spoznali s sprejemanjem in pošiljanjem paketov preko vtičnic na vozlišču, kar nam bo omogočalo komunikacijo z oddaljeno aplikacijo. Oddaljeno aplikacijo bomo simulirali s programom, ki bo lokalno tekel na vašem računalniku. Vzpostavili bomo komunikacijo med tem programom, programom nRouted, in brezžičnim vozliščem. Implementirali bomo odčitavanje senzorjev temperature in svetlosti. Inteligentni aspekt bomo dodali vozlišču tako, da bo naša aplikacija lahko sestavila zahtevek po vrednosti z določenega senzorja, vozlišče pa nam bo odgovorilo s primerno meritvijo. V ta namen si bomo izmislili protokol zapisa poizvedovalnih in meritvenih paketov – protokol bomo implementirali na vozlišču in v lokalni aplikaciji. S to nalogo bomo zaokrožili grob oris potrebnih konceptov, ki smo jih spoznavali na prejšnjih vajah, za daljinsko vodeni inteligentni brezžični senzor. Napišite poročilo, v katerem opišete kaj ste počeli in probleme s katerimi ste se srečevali. Poročilo morate oddate do naslednjega petka po elektronski pošti asistentu – v "`zadeva`" vpišete "`sdv_ime_priimek`", poročilo pa tudi poimenujte z imenom in priimkom (npr., `sdv_vaja4_ime_priimek.doc`). V Kolikor ne utegnete oddati v roku (t.j., prejmem vaš mail z datumom po petku 24:00h), oziroma ime datoteke/zadeve ne ustreza navodilom, se bo izhodiščna ocena tega poročila znižala na 8.

1 Preden začnemo

S spletne strani predmeta z naslova `http://vision.fe.uni-lj.si/Classes/SDV-vaje/vaje/2012/vaja4_izhodisce.zip` si prenesite v vašo delovno mapo `/myProjects/` datoteko
`vaja4_izhodisce.zip`. Odpakirajte vsebino v to mapo tako, da se postavite nanjo v Dolphinu (raziskovalcu v Linuxu), kliknete nanjo z desnim knofom miške in izberete "`Extract archive here`". V paketu se nahajata dve mapi. Prva mapa je `/client/`, druga pa `/ex4/`. V mapi `/ex4/` se nahaja izhodiščna koda, ki jo boste nadgradili v tej vaji. Po funkcionalnosti je skoraj enaka, kot tista, ki ste jo sami napisali v prejšnji vaji, le da poleg tega implementira še komunikacijo prek vtičnic in branje vrednosti senzorja temperature in svetlosti.

V mapi `/client/` se nahaja aplikacija "`client.c`" za komunikacijo prek vtičnic. Če imate verzijo programa že na vašem računalniku, jo prepišite s tisto v `vaja4_izhodisce.zip`. V nalogi bomo vzpostavili komunikacijo med vozliščem in ločeno aplikacijo. Aplikacija mora na začetku odpreti vtičnico, se priklopiti na `nRouted` ter ga skonfigurirati kot posrednika med vozliščem in aplikacijo. Preden začnemo z delom na vozlišču, moramo prevesti aplikacijo v izvršno kodo in nastaviti parametre v `nRouted`. Izhodiščna koda za to aplikacijo, ki implementira vtičnice na strani osebnega računalnika, ter že vsebuje pravilno konfiguracijo za `nRouted`, najdete v `client/client.c`. Polega datoteke se nahaja še datoteka `Makefile`, ki vsebuje pravila za prevajanje:

```
client: client.o
    g++ -o client.o client.c
```

V `Terminalu` prevedete program `client.c` z ukazom `make client`. Ko program prevedete, ga lahko zaženite brez parametrov, da vam izpiše sintakso za pravilni zagon. Trenutna funkcionalnost je konfiguracija `nRouted`, pošiljanje paketa vsem vozliščem z določeno vsebino, in čakanje na odgovor.[1] Zaenkrat programa ne zaganjajte. Sedaj se bomo posvetili razvoju programa za vozlišče, nato pa se bomo vrnili k aplikacijam `client` in `nRouted`.

2 Vtičnice: inicializacija, branje in pisanje

Komunikacija med vozlišči in vstopnimi točkami poteka preko nanoStackovih vtičnic, ki se konceptualno ne razlikujejo od klasičnih UNIXovih vtičnic tipa Berkley. Za izhodišče vam je asistent pripravil program, ki je po funkcionalnosti podoben tistemu, ki ste ga sprogramirali na prejšnjih vajah, poleg tega pa že implementira inicializacijo ter branje in pisanje na vtičnice. S programom Kate odprite `main.c` in si oglejte funkcijo
`static void mojaGlavnaFunkcija( int8_t *pvParameters )`
Prve vrstice v neskončni zanki inicializirajo in odprejo vtičnice:

```
 /* Socket initialization code */
    // Open socket for stac status messages
    stack_event = stack_service_init(NULL) ;
    // connect socket
    com_socket = socket(MODULE_CUDP, 0) ;
    // start listening to socket
    socket_bind(com_socket, &naslov_gw) ;
/* Socket initilized */
```

Nato v primeru, da smo vstopno točko že odkrili in ne preiskujemo komunikacijskih kanalov (`skeniraj == 0`), poslušamo 300ms na vtičnici:

```
    moj_buffer = socket_read(com_socket, 300) ;
```

Če prejmemo paket z vtičnice, preberemo dolžino podatkov v `tmp_uint16` in postavimo kazalec `pData` na podatke v paketu:

```
    tmp_uint16 = buffer_data_length( moj_buffer ) ;
    pData = buffer_data_pointer( moj_buffer ) ;
```

Sedaj lahko vsebino podatkov izpišemo preko UART na terminal ali pa jih analiziramo (to boste storili v zadnji vaji). Po uporabi moramo sprostiti rezervirano polje podatkov:

```
    // sprosti buffer
    socket_buffer_free(moj_buffer) ;
    moj_buffer = 0 ;
```

Če želimo poslati paket iz vozlišča nazaj na vstopno točko, se moramo najprej polastiti vtičnice in dobiti kazalec na (novo) rezervirano polje podatkov, v katerega bomo pisali:

```
    moj_buffer = socket_buffer_get( com_socket ) ;
    pData = buffer_data_pointer( moj_buffer ) ;
```

Sedaj lahko napišemo v podatke sporočilo bajt za bajtom. V kodi imate primer, ki vpiše tri znake `Hi!`, in pove, da se v polju *nahajajo trije uporabni podatki* (`moj_buffer->buf_end=3`).

```
    pData[0] = 'H' ; pData[1] = 'i' ; pData[2] = '!' ;
    moj_buffer->buf_end = 3 ;
```

[1] Če boste želeli testirati komunikacijo med vašim vozliščem in klientom `client.c`, boste morali najprej v ločenem terminalu zagnati `nRouted`, saj je slednji posrednik med klientom in vozliščem. Zaradi lažjega dela spremenite vsebino konfiguracijske datoteke programa `nRouted` tako, da spremenite številko IP v "loopback" (127.0.0.1) – to številko IP boste kasneje uporabili tudi pri zagonu programa `client`.

Sledi koda za pošiljanje paketa, ki najprej določi port na katerem posluša končna aplikacija, nato pa poskusi poslati paket. V kolikor je pošiljanje uspešno, se rezervirano polje sprosti samo, sicer pa moramo to opraviti sami ročno.

Preden nadaljujete: Na vrhu funkcije `main.c` morate vpisati naslov MAC vaše vstopne točke, ki jo boste uporabili (tako kot pri prejšnjih vajah – prepišite jo iz vaših zapiskov).

3 Branje senzorjev

Poleg skeleta za branje in pisanje v vtičnice, vam je asistent pripravil tudi kodo za branje dveh senzorjev, ki sta že integrirana na brezžičnem vozlišču: senzor svetlosti in senzor temperature. Branje senzorjev izvedete s klicem funkcije `readSensors()`. Ta funkcija izvede branje obeh senzorjev in shrani odčitane vrednosti v *globalni spremenljivki* `uint16_t val_temperatura` in `uint16_t val_svetlost`, do katerih imate direkten dostop. Opazite lahko, da sta obe spremenljivki dolžine dveh bajtov.

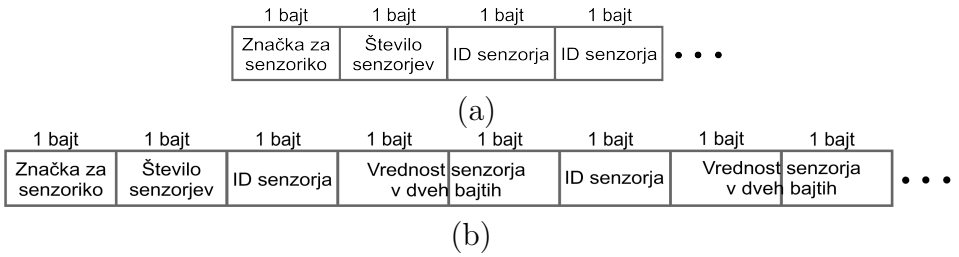
4 Moj prvi brezžični inteligentni senzor (40%)

Zahtevana funkcionalnost 1: V `main.c` dodajte kodo, ki bo ob prejemu paketa izpisala preko UART na CuteComov terminal prvi bajt v prejetih podatkih (`*pData`). Za izpis enega bajta na UART uporabite `debug_put( uint8_t znak )`. Hkrati, ko prejmete paket, preberite še vrednosti senzorjev temperature in svetlosti, ter ju izpišite na UART. Za izpis dvo-bajtne spremenljivke uporabite funkcijo `debug_int( uint16_t vrednost )`.

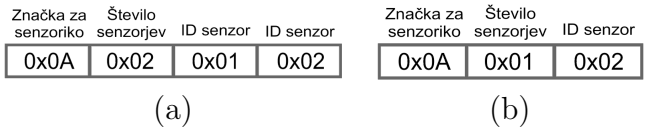
Zahtevana funkcionalnost 2: V `main.c` dodajte kodo, ki bo ob prejemu paketa vrednost za svetlost zapisala v izhodni paket in ga poslala na vstopno točko (spremeniti morate tisti del kode, ki pošlje sporočilo `Hi!`). Bodite pozorni, da je svetlost zapisana z dvemi bajti, vi pa dostopate do podatkov bajt po bajt – vpisanje rešite z uporabo pomikanja bitov dvo-bajtne spremenljivke svetlosti za 8 bitov v desno. Pazite, da vpišete pravilno dolžino polja podatkov.

Zagon: Kodo prevedite in sprogramirajte vozlišče (pred tem morate nastaviti v `app.rules` spremenljivko `BSL_PORT` na port, kjer je priklopljen programator – to preberete z `dmesg | tail`). Medtem, ko se vozlišče programira, si lahko nastavite testno okolje. Zagnati boste morali `./nRouted` in prebrati IP ter Port s konfiguracijske datoteke `nRouted.conf`. Nato boste zagnali `./client` (s temi parametri), ki bo zgeneriral paket, se priklopil na `nRouted` in preko njega poslal paket na vozlišče. Potem bo počakal na odgovor in vsebino paketa izpisal na ekran. Najprej pa vzemite s kovčka nano usmerjevalnik (tisti, ki ste ga označili), ga vstavite v USB in preverite katero številko mu je dodelil Linux. Sledite spodnjim točkam:

- **nRouted:** Odprite nov Terminal in se postavite na `Work/Sensinode/NanoStack/NanoStack-v1.1.0/Binaries/Tools/linux/`, odprite `nRouted.conf` in si prepišite naslov IP ter številko porta. Če je potrebno, nastavite port `dev/tty/USBx` na pravilno številko. Zaženite `nRouted` v "monitor"modu.
- **CuteCom:** Zaženite CuteComov terminal `cutecom&` in nastavite številko vrat na kateri je priklopljen programator `/dev/ttyUSBx`. Če je vozlišče končalo s programiranjem, vklopite komunikacijo med CuteCom in vozliščem. Na vozlišče pošljite ukaz `s`, kar pomeni, da bi moral ponovno začeti z iskanjem vstopne točke.
- **Client:** Odprite nov Terminal in se postavite na `client/`. Ko ste prepričani, da je vozlišče že odkrilo vstopno točko, zaženite client in mu podajte IP ter port na katerem posluša nRouted: `./client ŠtevilkaIP ŠtevikaPorta`



Slika 1: Protokol vsebine poizvedovalnega paketa (a) in meritvenega paketa (b).



Slika 2: Primer vsebine poizvedovalnega paketa za dva senzorja (a) in en senzor (b).

> Paket bi moral prispeti na vozlišče, to bi moralo javiti paket na CuteComov terminal in poslati odgovor, ki bi se moral izpisati v `client`ovem Terminalu, s čimer bi `client` zaključil izvajanje. Če `client` ne zaključi izvajanja, pač pa še vedno čaka na paket, potem nekaj niste pravilno povezali ali sprogramirali.

Ko preverite, da vaš program deluje po predpisih, pokličite asistenta, da zabeleži in lahko nadaljujete z zadnjim delom naloge.

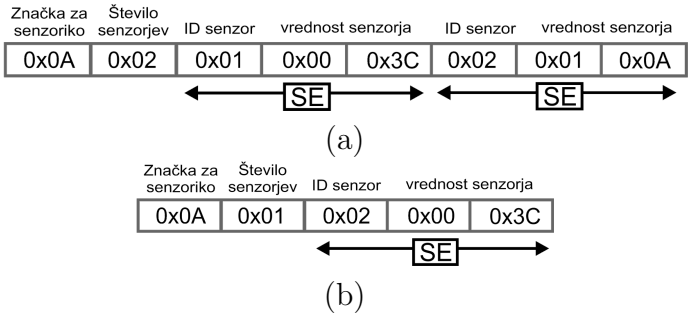
5 Implementacija meritvenega protokola (40%)

Z vajami, ki smo jih opravili do te točke, smo implementirali bistvene funkcionalnosti, ki jih potrebuje brezžični inteligentni senzor: (i) odkrivanje in sledenje vstopni točki, (ii) preverjanje identitete vstopne točke, (iii) prejemanje in oddajanje paketov, ter (iv) branje in vpisovanje vsebine senzorjev. V tem zadnjem delu naloge bomo dodali še funkcionalnost aktivnega merjenja na brezžičnem senzorju. To bo vključevalo spreminjanje programa, ki teče na vozlišču, kakor tudi programa v aplikaciji `client.c`. V obeh programih bomo implementirali preprost protokol, ki bo omogočal aplikaciji upravljanje z vozliščem. Aplikacija bo lahko na vozlišče poslala poizvedbo po določenem senzorju. Vozlišče bo na zahtevek/poizvedbo odgovorilo z odčitano vrednostjo senzorja (ali večih senzorjev). Ukaze in odgovore bomo pošiljali kot zaporedje bajtov v paketu. Torej moramo najprej določiti princip zapisa te sekvence bajtov – izmisliti si moramo protokol paketov.

Protokol za sestavljanje poizvedovalnih paketov: Poizvedovalni paket (Slika 1a) sestavi klient in ga pošlje na vozlišče. Paket je lahko poljubno dolg (do omejitve UDP protokola, na katerem teče). Začne se z bajtom, ki vsebuje značko za "senzorični protokol´´ – to značko si lahko zmislimo sami (npr., naj bo `0x0A`). Naslednji bajt naj vsebuje število zahtevanih senzorjev (`n`). Naslednjih `n` bajtov pa naj vsebuje značke katere senzorje želimo odčitati. Ker imamo trenutno na voljo le dva senzorja, recimo, da naj bo značka za senzor svetlosti `0x01`, za senzor temperature pa naj bo `0x02`. V Sliki 2a vidimo primer poizvedovalnega paketa, ki zahteva dve meritvi, prvo meritev s svetlobnega senzorja (`0x01`) in drugo s toplotnega (`0x02`). V Sliki 2b pa vidimo preprostejši primer poizvedovalnega paketa, kjer zahtevamo meritev le z enega senzorja, in sicer senzorja toplote (`0x02`).

Protokol za sestavljanje meritvenih paketov: Meritveni paket (Slika 1b) sestavi vozlišče in ga kot odgovor na poizvedbo pošlje klientu. Paket je lahko poljubno dolg (do omejitve UDP protokola, na katerem teče). Začne se z bajtom, ki vsebuje značko za "senzorični protokol" – to značko si lahko zmislimo sami mora pa biti enaka kot pri poi-

zvedovalnih paketih (npr., `0x0A`). Naslednji bajt naj vsebuje število senzoričnih vrednosti v paketu (`n`). Sledi `n` senzoričnih elementov (SE), vsak senzorični element pa je dolg po tri bajte. Prvi bajt v SE vsebuje značko za kateri senzor gre, svetlobni ali toplotni (`0x01` ali `0x02`). V naslednjih dveh bajtih pa je vpisana 16 bitna vrednost s senzorja. Tipični primer meritvenega paketa s svetlobno in toplotno meritvijo prikazuje Slika 3a. Preprostejši paket, samo z meritvijo topolote pa prikazuje Slika 3b. Če vam uspe implementirati bolj



Slika 3: Primer vsebine meritvenega paketa za dva senzorja (a) in en senzor (b).

kompleksnejšo obliko meritvenega/poizvedovalnega paketa, t.j., tako, kjer lahko poizvedujemo po poljubnem številu meritev znotraj enega paketa, dobite dodatnih 10% (bodite pozorni na dejstvo, da to ni tako preprosto, kot se sliši).

Naloga: Spremenite program v `client.c` tako, da sestavi poizvedovalni paket za poljubni senzor, ga pošlje na vozlišče in čaka na odgovor. Ko prejme paket z vozlišča, pregleda njegovo vsebino, ugotovi za kateri senzor se poizveduje, vrednost senzorja spravi v lokalno spremenljivko (npr., tipa `int`) in jo izpiše na ekran. V primeru svetlobnega senzorja naj, recimo, izpiše "`Senzor svetlosti: 400 lux`"[2]. Spremenili boste tudi kodo na vozlišču: Ko vozlišče prejme paket, preveri ali gre za poizvedovalni paket, prebere značko za senzor in ugotovi kateri senzor želimo odčitati. Potem tvori meritveni paket in ga pošlje klientu[3]. Protokol naj omogoča poizvedovanje/odgovarjanje po vrednosti samo enega senzorja naenkrat.

Sistem mora biti skonfiguriran kot v prejšnji nalogi. Ko končate nalogo, in preverite da deluje po predpisih, pokličite asistenta, da pregleda in zabeleži vaš prvi uspešno implemetiran brezžični senzor.

5.1 Pregled paketov v Wiresharku (20%)

Zaženite Wireshark in prestrežite paket, ki vsebuje poizvedbo po vrednosti senzorjev na vašem vozlišču. Vsebino paketa prikažite in naredite print-screen. V sliki poiščite in označite bajte (heksadecimalne vrednosti), ki ustrezajo vašemu paketu.

[2]Sedaj boste imeli situacijo, ko imate vrednost senzorja vpisano v dveh bajtih, zapisati pa jo morate v spremenljivko, ki je dolga 4 bajte. To boste najlažje dosegli z vpisovanjem po osem bitov v spremenljivko in premikanjem njene vsebine v levo za osem bitov.

[3]Protokol zapisa meritvenega in poizvedovalnega paketa si lahko po želji izmislite tudi sami