Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Машинско инжењерство
Ниво студија: Основне академске студије
Модул: Информатика у инжењерству
Предмет: Архитектура рачунарских система
Број индекса: 729/2014

Срђан Д. Сокић

Архитектура „GNUBLIN“ развојне платформе

завршни рад

Комисија за преглед и одбрану:

1. **Др Александар Пеулић - ментор**
2. **_____** Датум одбране:_____
3. **_____** Оцена:_____

У оквиру овог завршног рада кандидат треба да

Проучи архитектуру *“GNUBLIN”* развојне платформе која ради под Linux оперативним системом. системом. Пре покретања платформе, потребно је да се кандидат упозна са инструкцијама и радом Linux оперативног система. Након теоријске основе, потребно је реализовати практичне примере којима се демонстрира поступак пројектовања и програмирања*“GNUBLIN”* развојне платформе.

Препоручена литература:

[1] Karim Yaghmour, ***Building Embedded Linux Systems***, O'Reilly & Associates, Inc., United States of America, 2003.

[2] Hubert Hoegl, ***Teaching Embedded Linux with Gnublin***, Hochschule Augsburg, Fakultat fur Informatik, January 13, 2013.

[3] Милан Мијалковић, ***Предавање о неким стандардима комуникације***, Висока школа електротехнике и рачунарства струковних студија, Београд, 2008.

[4] Борислав Ђорђевић, Драган Плескоњић, Немања Мачек, ***Оперативни систем Linux,*** Виша електротехничка школа, Београд, 2004.

Крагујевац, датум

ментор:
Александар Пеулић, ванредни професор

АРХИТЕКТУРА “GNUBLIN” РАЗВОЈНЕ ПЛАТФОРМЕ

Резиме:

У овом раду, представљена је архитектура “GNUBLIN” развојне платформе. Реч је о платформи коју покреће оперативни систем Linux.

У уводном делу, приказан је о развој опертивног система Linux кроз време, као и најчешћа питања везана за Embedded Linux системе. У поглављима 2, 3 и 4 описани су неки основни концепти, хардверска подршка као и развојни алати. У следећем делу приказане су неке од познатих Embedded Linux платформи попут Raspberry Pi и BeagleBone. У шестом одељку представљене су карактеристике “GNUBLIN” развојних платформи као што су GNUBLIN-DIP, GNUBLIN-EXTENDED и GNUBLIN-LAN. У поглављу број седам детаљније је описана архитектура GNUBLIN-STANDARD платформе, која је одлична за рад са студентима и почетницима, а коју ћемо касније користити у изради пројектног задатка.

У осмом делу рада, описан је поступак израде пројектног задатка који се састоји из два дела и то : Повезивање LED светла са GNUBLIN-STANDARD плочом и повезивање Bluetooth модула са плочом и његово пријављивање на систем.

Кључне речи: ***Embedded Linux, GNUBLIN***

ARCHITECTURE OF THE "GNUBLIN" DEVELOPMENT PLATFORM

Summary:

In this paper, we present the architecture of the"GNUBLIN" development platform. It is a platform that runs the Linux operating system.

In the introduction, the participants discussed the development of Linux over time, as well as about the most common issues related to embedded Linux systems. In chapters 2, 3 and 4 describes some basic concepts, hardware support and development tools. The following section describes some of the known Embedded Linux platforms such as Raspberry Pi and BeagleBone. Sixth section describes the characteristics of the "GNUBLIN" development platforms such as GNUBLIN DIP, GNUBLIN-EXTENDED and GNUBLIN-LAN. In the chapter number seven is described in more detail architecture of the GNUBLIN-STANDARD platform, which is great to work with students and beginners, and which will later be used in the preparation of project task.

Eighth part of the paper describes the procedure of project design, which consists of two parts: Connecting LEDs with GNUBLIN-STANDARD panel and connect a Bluetooth module with the board and its registration on the system.

Key words: ***Embedded Linux, GNUBLIN***

САДРЖАЈ

1. УВОД

Од свог првог јавног објављивања 1991. године, **Linux** постао оперативни ситем који се масовно користи. У почетку ограничен, повезао је групу програмера и ентузијаста на Интернету, резултујући на крају чврстим **Unix-like** оперативним системом за радне станице, сервере и кластере. Његов раст и популарност убрзо покреће **Free Software Foundation (FSF)** и дериват који ће касније бити познат као **OpenSource** покрет. Присуство медијске и пословне пажње, доприноси успостављању **Linux**-а као легитимног и одрживог избора за оперативни систем. Ипак, то се често занемарује у сегменту рачунарских уређаја код којих **Linux** полако постаје пожељан оперативни систем. Тај сегмент су **Embedded**-системи , рачунари који налазе примену у свим сегметнима живота савременог човека. Од мобилних телефона до медицинске опреме, укључујући клима-навигационе системе, говорне аутомате банака, MP3 уређаје , штампаче, аутомобиле и друге уређаје којих често нисмо ни свесни.

1.1. Дефиниције

Речи **Linux**, **Embedded Linux** и **Real-Time Linux**, често се користе са малим осврtом у односу на оно за шта су одређене. Понекад ознаке могу да значе нешто веома прецизно. Некада, широк опсег и категорија апликација се могу узети у обзир. Сазнаћемо шта ове речи значе у различитим ситуацијама.

1.1.2. Шта је „Linux“?

Реч "**Linux**" се односи само на кернел **Linux**-а али се често (и погрешно) користи као назив за за цео оперативни систем (такође ГНУ / Линук) базиран око тог језгра и броја библиотека и алата из **GNU** пројекта. Више стотина Линук дистрибуција садрже разни софтвер заједно са **GNU / Linux** кернелом. У почетку, Линук су развијали и користили ентузијастични студенти. Од тада, Линук је добио подршку од стране великих компанија као што су IBM и Novell за кориснике на серверима и почиње да улази у употребу и на личним (PC) компјутерима. Промотери и аналитичари тврде да за овај успех треба захвалити његовој независности од било каквог централног произвођача, ниском трошку, тј. бесплатности, безбедности и поузданости. **Linux** је оригинално развијан за **Intel 386** микропроцесоре, данас подржава низ микропроцесора и рачунарских платформи. Употребљава се у распону од личних рачунара до суперкомпјутера и интегрисаних система као што су мобилни телефони и лични видео рекордери.

1.1.3. Шта је „Embedded Linux“?

Када мање упућеном кориснику рачунара поменете **Linux**, шта је прва асоцијација? Вероватно ништа. Они нешто упућенији би вероватно могли да опишу неке карактеристике **Linux**-а, уз незаобилазне полуинформације и дезинформације покупљене са **IT** сајтова и од познаника који се издају за компјутерске познаваоце. А шта се дешава када напреднијег корисника упитате шта је **Linux** и где се све среће? Вероватно ћете добити одговор да је реч о стабилном клону **Unix**-а, често коришћеном како на серверима тако и радним станицама за развој мултиплатформског софтвера.

Мали број познавалаца рачунара зна да **Linux** ужива приличну популарност на наменским (такозваним "embedded") рачунарима.

Према томе, Embedded Linux-системи су наменски рачунари попут данашњих мобилних телефона или таблет рачунара који користе оперативне системе базиране на Linux-језгру (кернелу).

1.1.3. Шта је „Real-Time Linux“?

RTLinux је пројекат који је 1996. објавио Мајкл Бараbanов под надзором Виктора Јодакена. Циљ пројекта је био да се обезбеди детерминистичко време одзива под **Linux** окружењем. Ипак, данас има много више пројеката који обезбеђују једном или другом облику реалног времена одзив под **Linux**-ом. **RTAI, Kurt, and Linux/RK,** сви пружају перформансе у реалном времену под **Linux**-ом. Побољшања неких пројеката се добијају уметањем секундарног кернела у **Linux** кернел. Остали унапређују-побољшавају одзив кернела помоћу фластера.

Придев "у реалном времену" се користи у комбинацији са **Linux**-ом да опише неколико различитих ствари. Углавном се користи да се каже да систем или један од његових компоненти треба да има фиксни одговор у времену, али ако користите строгу дефиницију "реалног времена", можете наћи да оно што се нуди није нужно "у реалном времену".

2. ОСНОВНИ КОНЦЕПТИ

Као што смо видели у претходном поглављу, постоји много варијанти **Embedded Linux** система. Ипак, има неколико кључних карактеристика које се равномерно односе на **Embedded Linux** системе. Сврха овог поглавља је да нам представи основне појмове и проблеме на које ћемо наићи приликом пројектовања оваквих система.

Поглавље почиње разматрањем врсте хостова који се најчешће користе за развој **Embedded Linux** система, врсте хост(домаћин)/мета развојних окружења, као и врсте хост(домаћин)/ мета **debug** подешавања. Ове секције имају за циљ да нам помогну да одаберемо најбоље окружење за развој **Embedded Linux** система. Поглавље затим приказује детаље структура које се обично могу наћи у овим системима.

2.1 Типови хостова (домаћина)

Сваки могући систем-мета може бити развијен од стране широког спектра хостова(домаћина). Разматраћемо типове хостова(домаћина) који се често користе, њихове податке, и како је лако развити **Embedded Linux** систем користећи их.

2.1.1 „Linux“ радне станице

Ово је најчешћи тип развојног хоста(домаћина**)** за **Embedded Linux** системе који се препоручује зато што развој ових система подразумева да је корисник детаљно упознат са **Linux**-ом, јер не постоји бољи начин да се то уради него да га користимо за наш свакодневни рад.

Linux радне станице најчешће могу бити персонални рачунари (Desktop рачунари или Lap-Top рачунари). Не смемо заборавити да **Linux** ради у скоро свим хардверским окружењима и да нисмо ограничени на коришћење персоналног рачунара. Наравно, најновији и најбржи хардвер је сан сваког инжењера. Поседовање брзе машине, сигурно ће нам помоћи у нашем раду, међутим и даље можемо користити слабије машине са адекватном **RAM** меморијом за овај тип развоја. Морамо имати у виду да је **Linux** врло добар у извлачењу најбољих особина из нама доступног хардвера. Шта нам је заправо потребно, потребан нам је простор и то на чврстом диску и **RAM** меморији. Поред простора који користи изабрана дистрибуција, потребно је још од 2 до 3 GB простора на чврстом диску, за развојно окружење и радни простор. Што се тиче **RAM** меморије, неки од **GNU** компајлерских корака, захтевају велике количине исте, поготово током изградње **„C“** библиотека. Препоручује се 128 MB и 128 MB **swap** простора.

2.1.2 „Unix“ радне станице

У зависности од околности, можда ћемо морати да користимо традиционалну **Unix** радну станицу. **Solaris** радне станице, су на пример, веома честе међу телекомуникационим решењима програмера.

Зато што је сам по себи **Linux** веома сличан **Unix**-у, већина онога што се односи на **Linux** важи и за **Unix**. Ово је нарочито тачно када је у питању развој **GNU** ланаца алата, јер су основе **GNU** алата као што је компајлер, **„C“** библиотека и бинарне „услуге“ (обично познате под називом **„binutils“**) развијене и коришћене на традиционалним **Unix** системима, чак пре него што је **Linux** постојао.

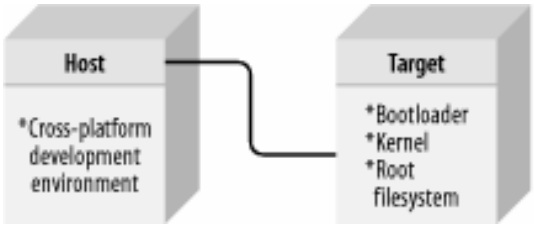
Када су меморијски простори у питању важе сличне карактеристике, као код **Linux**-а.

2.2 Типови хост(домаћин)/мета развојних подешавања

Три различите хост(домаћин)/мета архитектуре су доступне за развој **Embedded Linux** система: повезана подешавања, подешавања преносивог простора, самостална подешавања. Наша тренутна подешавања могу спадати у више од једне категорије или можда чак и променити категорије током времена, у зависности од захтева методологије развоја.

2.2.1 Повезана подешавања

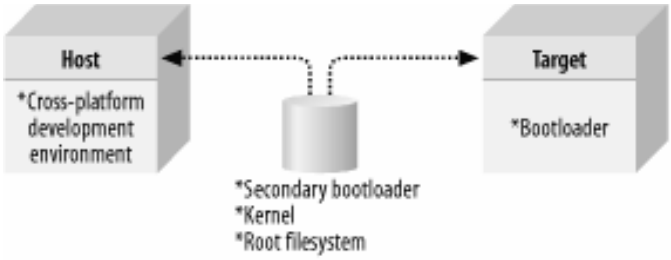
У овој поставци мета и домаћин су трајно повезани помоћу физичке везе – кабла. Ова веза је обично серијска или **Ethernet** веза. Најважнија особина овог подешавања је да нема преноса физичког хардвера за складиштење између мете и хоста(домаћина).



Слика 2.2.1 Хост(домаћин)/мета повезано подешавање

2.2.2 Подешавање преносивог простора

У овој поставци, не постоје директне физичке везе између хоста(домаћина) и мете. Уместо тога, складишни простор је написан од стране хоста(домаћина), а затим пребачен у мету(target), где је искоришћен за покретање уређаја.

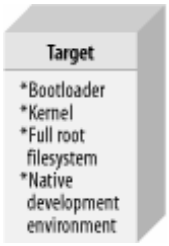


Слика 2.2.2 Хост(домаћин)/мета подешавање преносивог простора

2.2.3 Самостално подешавање

У овој поставци мета је самостални развојни систем, који укључује сав потребан софтвер да се покрене, ради и развија додатни софтвер. У суштини, ово подешавање је слично стварној радној станици, осим што основни хардвер није конвенционална радна

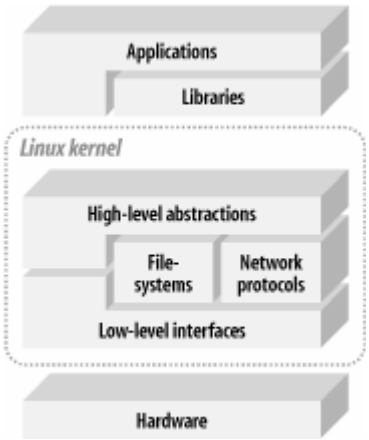
станица већ сам уграђен(embedded) систем.



Слика 2.2.3 Хост(домаћин)/мета самостално подешавање

2.3 Општа архитектура „Embedded Linux“ система

Од **Linux** система који су направљени са мноштвом компонената, погледајмо укупну архитектуру општег **Linux** система. То ће нам омогућити да подесимо сваку компоненту у контексту да ће нам помоћи да разумемо интеракцију између њих и како најбоље искористити њихове саставе. (Слика 2.3) представља архитектуру **Linux** система са свим компонентама.



*Слика 2.3 Општа архитектура **Linux** система*

Постоје дефинисане карактеристике које се очекују од хардвера за покретање **Linux** система. Прво, **Linux** захтева 32-битни процесор који садржи **Memory Managment Unit (MMU)**. Друго, довољна количина **RAM**-а, треће, минималне **I/O** могућности ако постоји развој који треба да се врши на мети са debugging објектима. Ово је такође веома важно за било које касније решавање проблема на терену. Коначно, језгро мора бити у стању да учита и / или приступи **root** фајл систему кроз неки облик трајног или мрежног складиштења.

3. ХАРДВЕРСКА ПОДРШКА

Пошто смо у претходном поглављу дефинисали основе **Embedded Linux** система, сада ћемо нешто рећи о хардверу који је подржан од стране **Linux** система. Прво ћемо обрадити архитектуре процесора које су подржане **Linux**-ом, а затим обрадити хардверске компоненте као што су магистрале и **I/O** системи.

3.1 Архитектуре процесора

Linux ради на великом броју архитектура, међутим, не користе се све за уграђене системе.

3.1.1 x86 архитектура

У 1980. и раним 1990. годинама када су 8088 и 80286 и даље били у општој употреби, термин x86 обично је представљао било који 8086 компатибилни CPU. Данас, међутим, x86 обично подразумева бинарну компатибилност такође са 32-битним скупом инструкција од 80386. То је због чињенице да је овај скуп инструкција постао нешто као најмањи заједнички именилац за многе оперативне системе, а вероватно и због тога што је постао уобичајени термин након увођења 86386 у 1985. години. Термин x86 понекад може се истаћи као x86-32 да би се разликовао од или оригиналног 16-битног "x86-16" или од 64-битног x86-64. Иако се већина x86 процесора, који се користе у новим персоналним рачунарима и серверима, има 64-битну компатибилност, да би се избегли проблеми са компатибилношћу са старијим рачунарима или системима, термин x86-64 (или x64) се често користи да означи 64-битни софтвер, са термином x86 који тренутно имплицира само 32-бита.

Иако је 8086 првенствено развијен за уграђене системе и мале једно-корисничке рачунаре, у великој мери као одговор на успешну 8080-компатибилност Zilog-Z80, линији x86 убрзо су порасле карактеристике и процесорска снага. Данас, x86 је свеприсутан у стационарним и преносивим персоналним рачунарима и заменио је опсег средњих рачунара и рачунара са смањеним скупом инструкција **(RISC)** заснованих рачунара већином од сервера и радних станица. Велика количина софтвера, који укључују оперативне системе (OS) као што су DOS, Windows, Linux, BSD, Solaris и Mac OS X функционишу са x86 заснованим хардвером. Модеран x86 је релативно редак у уграшеним системима, међутим, и мале апликације са ниском потрошњом енергије (које користе мале батерије) као и јефтино миксопроцесорско тржиште, као што су кућни апарати и играчке, немају никакво значајно x86 присуство. Једноставна 8-битна и 16-битна архитектура су заједничке овде, иако x86-компатибилан VIA-C7, VIA Nano, AMD Geode, Athlon Neo и Intel Atom су примери 32- и 64-битног дизајна који се користе са релативно малом снагом и јефтиним сегментима. Било је неколико покушаја, укључујући и сам Intel, да прекине тржишну доминацију "нелегалне" x86 архитектуре дизајниране директно од првих једноставних 8-битних микропроцесора. Примери за то су iAPX 432 (друго име Intel 8800), Intel 960, Intel 860 и Intel/Hewlett-Packard Itanium архитектура. Међутим, континуирано пречишћавање x86 микроахитектуре, кола и производње полупроводника направили су потешкоће у замени x86 у многим сегментима. AMD-ово 64-битно проширење x86 (којим је Intel коначно одговорио да је са компатибилним дизајном) и скалабилност x86 чипова као што су осмојезгарни Intel

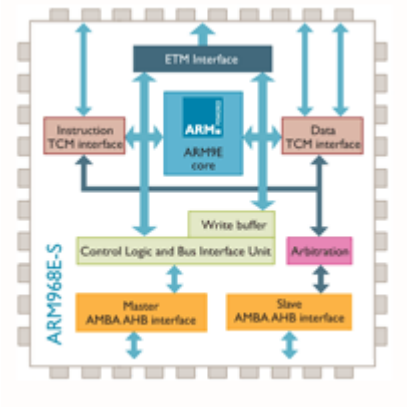
Xeon и 12-језгарни AMD Opteron су подвукли x86 као пример како континуирано пречишћавање утврђених индустријских стандарда може одолети конкуренцији од потпуно нових архитектура

Слика 3.1.1 Intel 8086

3.1.2 ARM

ARM архитектура представља скуп инструкција за процесоре базиране на **RISC**-архитектури развијена од стране британске компаније ARM Holdings.
RISC-базирани приступ рачунарског дизајна значи да ARM процесори захтевају знатно мање транзистора од типичних процесора код просечних рачунара. Овај приступ смањује цену, загревање и потрошњу енергије. Ово су жељене особине код лаких, преносних, уређаја који користе батеријско напајање укључујући паметне телефоне, лап-топ рачунаре, таблет рачунаре и друге уграђене системе. Једноставнији дизајн омогућава ефикасније вишејезгарне процесоре и већи број језгара по мањој цени, што даје већу рачунарску моћ и вишу енергетску ефикасност за сервере и супер-рачунаре. ARM Holdings развија скуп инсрукција и архитектуру за ARM-базиране производе, али не производи производе. Компанија периодично избацује ажуриране верзије својих језгара. Актуелна језгра из ARM Holdings-а подржавају 32-битни адресни простор и 32-битну аритметику; скоро представљена ARMv8-A архитектура уводи подршку за 64-битни адресни простор и 64-битну аритметику. Инструкције за ARM Holdings-ова језгра имају 32-бита широке инструкције фиксиране дужине, али новије верзије архитектуре такође подржавају и скуп инструкције променљиве дужине који омогућава и 32-бита и 16-бита широке инструкције ради боље густине кода. Нека језгра омогућавају и хардверско извршавање Јава бајткода. ARM Holdings лиценцира дизајн чипова и архитектуре ARM скупа инструкција трећим странкама, које дизајнирају своје производе који имплементирају једну од тих архитектура—укључујући системе на чипу (SoC) који интегришу меморију, интерфејсе, радио пријемнике (GSM), и тако даље. Тренутно, варијанте широко коришћених Cortex језгара, старијих „класичних“ језгара, и специјализованих SecurCore језгара су доступне сваком од њих са дозволом да додају или избаце одређене могућности. Компаније које производе ARM производе укључују Apple, NVIDIA, Qualcomm, Samsung, и Texas Instruments. Apple је први имплементирао ARMv8-A архитектуру у свом Apple A7 чипу у iPhone 5S. У 2005, око 98% свих продатих мобилних телефона су користили бар један ARM процесор. Мала потрошња струје је учинила ARM процесоре јако популарним: 37 милијарди ARM процесора је произведено до 2013-е, за разлику од 10 милијарди из 2008. ARM архитектура (32-битна) је најраспрострањенија архитектура код мобилних уређаја, и најпопуларнија 32-битна код уграђених система. Према ARM Holdings-у, само у 2010, произвођачи чипова базираних на ARM архитектури су пријавили продају 6.1 милијарди ARM-базираних процесора, што чине 95% телефони, 35% телевизори и сет-топ боксови и 10% мобилни рачунари. То је најраспрострањенија архитектура 32-

битног скупа инструкција по количини производа.



Слика 3.1.2 ARM архитектура

3.2 Магистрале и интерфејси

Магистрале и интерфејси су нити које повезују централну процесорску јединицу са периферијама које су део система. Свака магистрала и интерфејс имају своје сложености, и ниво подршке **Linux**-а различитим магистралама и интерфејсима варира у складу са тим. У наставку је дат резиме магистрала и интерфејса који се налазе у уграђеним системима и подршци **Linux**-а. **Linux** подржава многе друге магистрале као што су **Sbus**, **NuBus**, **TurboChannel**, и **MCA**.

3.2.1 Индустријски стандардна архитектура (ISA)

Када се појавила на првом PC рачунару, 8-битна **ISA** магистрала је радила на учестаности од скромних 4,77MHz - истом брзином као и процесор. Она је годинама побољшавана да би на крају постала магистрала индустријски стандардне архитектуре (**ISA**) у 1982. години, доласком рачунара **IBM PC/AT** који је користио процесор Intel 80286 и 16-битну магистралу за податке. На том степену развоја, она је успешно држала корак са системском магистралом, прво на 6MHz, а касније и на 8MHz. **ISA** магистрала дефинише 16-битну везу коју покреће генератор такта од 8MHz, то изгледа примитивно у поређењу са брзином данашњих процесора. Она има теоретску брзину преноса података до 16MB у секунди. Функционално, ова брзина би била преполовљена на 8MB у секунди, зато што је један циклус магистрале потребан за адресирање, а још један циклус магистрале за пренос 16 битова података. У стварности, магистрала преноси око 5MB у секунди - још увек довољно за многе периферијске уређаје - и велики број **ISA** картица за проширење је осигурао њено присуство у касним 90-им годинама.

Како су процесори постали бржи и добили шире стазе за податке, основна **ISA** конструкција није била у стању да се промени да би држала корак. Чак и данас, већина **ISA** картица остаје на 8-битној технологији. Мањи број типова са стазама података од 16-бита - контролери чврстих дискова, графички адаптери и поједини мрежни адаптери - ограничени су ниским нивоима пропусне моћи **ISA** магистрале, па ови процеси могу боље да се опслужују користећи картице за проширење у прикључцима бржих магистрала.

3.2.2 PCI магистрала

Интелов оригинални рад на стандарду PCI-био је објављен као ревизија 1.0 и препуштен посебној организацији, специјалној интересној групи за PCI (Специјална интересна група – PCI SIG). Организација PCI је произвела, PCI локалну магистралу ревизија 2.0 у мају 1993. године: Она је узела у обзир инжењерске захтеве од чланова и дала потпуну дефиницију саставних делова и конектора за проширење, дакле нешто што би могло бити употребљено за израду система спремних за производњу, заснованих на технологији 5V. Ван потреба за перформансом PCI је трагао да направи лакше имплементирање проширивања, понудивши "прикључи и ради" ("plug and play" - PNP) хардвер - систем који омогућава персоналном рачунару да се аутоматски подеси према новим картицама када се један прикључи, избегавајући потребу да се проверавају постављања краткоспојника и нивоа прекида. Оперативни систем: Windows 95-, уведен те године, обезбедио је софтвер оперативног система за подршку PNP хардвера, и све тадашње матичне плоче су имале уграђен BIOS посебно пројектован да ради са PNP могућностима таквих система. До 1994. године, PCI је био заснован као доминантан стандард за локалну магистралу. Док је VL магистрала била у суштини проширење магистрале, или стазе, коју је централна процесорска јединица користила да приступи главној меморији PCI је посебна магистрала, изолована од централне процесорске јединице, али има приступ главној меморији. Као таква, PCI-магистрала је отпорнија и више перформансе од VL магистрале је, за разлику од ове последње која је била пројектована да ради брзином системске магистрале, PCI-магистрала се повезује са магистралом путем електронских системских кола посебног "моста" и ради сталном брзином, независно од генератора процесорског такта. PCI-стандардно је ограничен на пет конектора, мада сваки од њих може бити замењен са два уређаја уграђених у матичну плочу. Могуће је, такође, да процесор подржава више од једног чипа за већину. Строжије је дефинисана од VL магистрале и нуди известан број додатних особина. Посебно, може да подржава картице које се напајају из обе врсте извора од 5V до 3,3V, користећи различите "прикључке - кључеве" да би се спречило да се картица стави у погрешан слот.

3.3 I/O(Input/Output) системи

I/O(Input/Output) системи играју централну улогу било ког рачунарског уређаја.

3.3.1 Серијски порт (Serial Port)

Серијски порт је вероватно најбољи пријатељ програмера уграђених система. Многи угрђени системи су развијени и дебаговани користећи **RS232** серијску везу између хоста(домаћина) и таргета(мете). Понекад, **PCB** су постављени за смештај серијског порта, али само развојне верзије плоча укључују прави конектор, а производни системи се испоручују без њега. Једноставност **RS232** интерфејса је охрабрио своју широку употребу и усвајање, иако је његов капацитет прилично ограничен у поређењу са другим средствима за пренос. Постоје и други серијски интерфејси осим **RS232**, од којих су неки мање осетљиви и стога више прилагођени индустријским срединама. Хардверски серијски протокол, међутим није толико важан као стварни програмски интерфејс омогућен од стране хардвера серијског уређаја. Пошто је **RS232** хардверски интерфејс, језгро не мора да подржи **RS232**. Уместо тога,

језгро укључује драјвере који чиповима доносе **RS232** комуникацију, **Universal Asynchronous Receiver-Transmitters (UARTs)**. **UART-и,** варирају од једне до друге, архитектуре мада неки **UART-и**, као што је 16550, се користе на више од једне архитектуре. Главни серијски **UART** драјвер у језгру је *drivers/char/serial.c*. Неке архитектуре, као што је **SH**, имају друге серијске драјвере, да прихвате свој хардвер. Неки архитектуре независно од периферне картице такође пружају серијске интерфејсе. Као и код других **Unix** система, серијским уређајима у **Linux** систему се равномерно приступа, без обзира на основни хардвер и сродни драјвер. Одговарајући уноси уређаја почињу са /dev/ttyS0, и могу ићи све до /dev/ttyS191. У већини случајева, међутим, постоје само неколико серијских уноса уређаја у системском /dev директоријуму.

3.3.2 Паралелни порт(Parallel Port)

У односу на серијски порт, паралелни порт је ретко важан део уграђеног(embedded) система. У неким случајевима, паралелни порт се користи јер уграђен систем мора да подржи штампач или неки други екстерни уређај, међутим са појавом USB и IEEE1394 та употреба се смањује. Област развоја уграђених система где паралелни порт одговара сасвим лепо, је једноставни мултибит I/O. Када вршимо дебаговање неког кода, на пример можемо лако повезати скуп LED диода, и те LED диоде користити за индикацију позиције у коду. Трик је у убацивању скупа излазних команди паралелног порта у различите делове кода и коришћењу LED диода за идентификацију последње позиције. То је могуће јер хардвер паралелног порта задржава последњу вредност излаза и према томе остаје непромењен без обзира на стање остатка система.

Linux подржава паралелни порт I/O кроз три слоја. На средњем нивоу је *parport* драјвер који је независан од архитектуре. Овај драјвер омогућава централни управни објекат за ресурсе паралелног порта. Драјвер средњег нивоа није видљив са стране корисничког простора и није доступан као упис у */dev* директоријуму. Драјвери нижег нивоа који контролишу стварни хардвер региструју своје присуство тако да своје услуге учине доступним за драјвере вишег нивоа. Оба драјвера, нижег и средњег нивоа се налазе у директоријуму језгра *drivers/parport*. Најчешћи драјвер високог нивоа је драјвер штампача, који омогућава кориснику апликације да користи штампач који је директно повезан са системским паралелним портом. Први уређај линије штампача је видљив као уређај */dev/lp0*, а други као */dev/lp1* и тако даље. Неки други драјвери високог нивоа, користе паралелни порт као продужетак магистрале за приступ екстерним уређајима који су повезани на систем. Сви они користе драјвер паралелног порта средњег нивоа, на овај или онај начин, и налазе се у */dev* директоријуму. И на крају, паралелни порт је доступан из корисничког простора, преко драјвера корисничког простора, који се види као */dev/parportX*, где је *X* број порта.

4. РАЗВОЈНИ АЛАТИ

Слично као програмерима мејнстримова, и програмерима уграђених уређаја су потребни компајлери, линкери, интегрисана развојна окружења и слични развојни алати. Алати које користе програмери уграђених уређаја су различити и обично раде на једној платформи док развијају апликације за другу. То је разлог зашто се ови алати често називају крос-платформски развојни алати или краће крос-развојни алати.

Ово поглавље говори о подешавању, конфигурацији и коришћењу крос-платформских развојних алата. Дискутоваћемо о практичном коришћењу пројектног радног простора и GNU крос-платформским развојним ланцима алата.

4.1 Коришћење практичног пројектног радног простора

У току развоја и прилагођавања софтвера таргету(мети), мораћемо да организујемо различите софтверске пакете и компоненте пројекта на свеобухватан и једноставан начин за коришћење структура директоријума. Табела 4.1 показује предложену структуру директоријума. Међутим, сваки програмер може ту структуру прилагодити својим потребама и захтевима. Приликом одлучивања, где поставити компоненте, увек треба покушати са проналаском најинтуитивнијег распореда. Такође, потребно је покушати са задржавањем кода у директоријуму потпуно одвојеног од пакета који ће се преузети са Интернета. Ово ће смањити дилеме у вези власништва и лиценцирања статуса извора информација.

Табела 4.1 Предложени распоред директоријума

Директоријуми	**Садржај**
bootldr	Програм за покретање таргета(мете).
build-tools	Пакети и директоријуми потребни за изградњу вишеплатформског ланца алата.
debug	Отклањање грешака алата и повезаних пакета.
doc	Потребна документација за пројекат.
images	Бинарне слике покретача, језгра, и *root* фајл система, спремне да се користе на таргету(мети).
kernel	Различите верзије језгра које процењујемо за мету.
project	Изворни код потребан за пројекат.
rootfs	*Root* фајл систем који језгро мете види у току рада.
sysapps	Системске апликације потребне за мету.
tmp	Привремени директоријум за складиштење и експериментисање пролазних фајлова.
tools	Комплетан вишеплатформски-развојни ланац алата и „C“ библиотека

Наравно, свако од ових директоријума садржи мноштво поддиректоријума.

4.2 GNU крос-платформски развојни ланци алата

Ланац алата морамо да саставимо заједно са развојем апликација за било коју мету, што укључује бинарне додатке као што су *ld*, *gas*, и *ar*, „C“ компајлер(GCC), „C“ библиотеку(glibc).

4.2.1 Основе GNU ланаца алата

Конфигурација и изградња одговарајућег GNU ланца алата је сложена и деликатна операција која захтева добро разумевање зависности између различитих софтверских пакета и њихових улога, што је веома значајно јер су се компоненте GNU ланаца алата развиле независно једна од друге.

4.2.1.1 Верзије компонената

Први корак у изградњи ланаца алата је одабир верзије компонената коју ћемо користити. То укључује одабир верзије бинарних додатака, верзије *GCC* и верзије *glibc*. Пошто су ови пакети одржавани и пуштени независни један од другог, неће се све верзије пакета правилно градити у комбинацији са различитим верзијама других пакета. Може се покушати са коришћењем најновијих верзија сваког пакета, али ова комбинација не гарантује да ће радити.

Да бисмо изабрали одговарајуће верзије, морамо да тестирамо комбинације пролагођене хосту(домаћину) и таргету(мети).

Табела 4.2 Познате комбинације функционалних верзија пакета

Host	Target	Kernel	binutils	gcc	glibc	Patches
i386	PPC		2.10.1	2.95.3	2.2.1	No
PPC	i386		2.10.1	2.95.3	2.2.3	No
PPC	i386		2.13.2.1	3.2.1	2.3.1	No
i386	ARM	2.4.1-rmk1	2.10.1	2.95.3	2.1.3	Yes
PPC	ARM		2.10.1	2.95.3	2.2.3	Yes
i386	MIPS		2.8.1	egcs-1.1.2	2.0.6	Yes
i386	SuperH		2.11.2	3.0.1	2.2.4	Yes
Sparc (Solaris)	PPC	2.4.0	2.10.1	2.95.2	2.1.3	No

4.2.1.2 Захтеви израде

Да би се израдио вишеплатформски-развојни ланац алата, потребан нам је функционалан ланац алата. Већина мејнстрим дистрибуција обезбеђује овај ланац алата као део њиховог пакета. Ако није инсталиран на нашој радној станици или није изабран због уштеде простора, мораћемо да га инсталирамо користећи поступак који одговара нашој дистрибуцији. Са *Red Hat* дистрибуцијом, мораћемо да инсталирамо одговарајуће *RPM* пакете. Такође ће нам бити потребан скуп заглавља језгра на нашем

хосту(домаћину). Та заглавља се обично налазе у */usr/include/linux*, */usr/include/asm*, и */usr/include/asm-generic* директоријумима, и то би требало да буду заглавља коришћена за компајлирање *glibc* библиотека инсталираних на наш систем. У старијим дистрибуцијама, и неким инсталацијама, ови директоријуми су повезани унутар */usr/src/linux* директоријума. Овај директоријум је сам по себи симболички линк језгра, инсталираног од стране актуелне дистрибуције.

5. EMBEDDED LINUX РАЗВОЈНЕ ПЛАТФОРМЕ

У складу са описаним динамичким развојем, данас постоји много развојних платформи за **Embedded Linux** системе. У овом поглављу ћемо поменути неке од познатијих платформи и укратко описати њихове архитектуре и могућности.

Међу познатим **Embedded Linux** развојним платформама су:

- *Raspberry Pi*
- *BeagleBone*
- *Gnublin (биће приказан у следећем поглављу)*

5.1 Raspberry Pi

Око *Raspberry Pi* рачунара, се подигло пуно прашине откад је први пут најављен. Овај рачунар величине кредитне картице може да изврши многе задатке да које обавља наш десктоп рачунар, као сто су прављење табела, обрада текста, играње видео игара. Такође има могућност да репродукује видео записе високе резолуције. *Raspberry Pi* може да да покрене неколико издања **Linux**-а и широм света се користи за подучавање деце програмирању.

Тајни састојак који омогућава да овај рачунар буде тако мали и моћан је *Broadcom BCM2835*, систем на чипу (*SoC*) који садржи *ARM1176JZFS* процесорско језгро са покретним зарезом, које ради на 700MHz и *Videocore 4* графичку процесорску јединицу (*GPU*). Графичка процесорска јединица омогућава коришћење *Open GL ES 2.0* и хардверски убрзаног *OpenVG*, а обезбеђује и декодирање *1080p30 H.264* сигнала као и израчунавања опште намене при брзинама од *1Gpixel/s*, *1.5Gtexel/s*, или *24 GFLOPs*. Шта све ово значи? Значи да ако *Raspberry Pi* прикључимо на *HDTV*, можемо гледати високо квалитетан *BluRay* видео користећи *H.264* на *40MBits/s*.

Слика 5.1 Raspberry Pi-Model B

Модел *B* такође има 10/100 *Ethernet* порт па можемо сурфовати вебом (или

преузети веб улогу сервера). Системски фајлови се налазе на *SD* картици, па можемо лако да припремимо, покренемо и дебагујемо неколико различитих оперативих система на истом хардверу. Већина **Linux** оперативних система за *Raspberry Pi* се може сместити на *SD* картицу капацитета 2GB, али су подржане и картице већег капацитета.

Два интегрисана *USB* порта на моделу *B* су довољна за повезивање миша и тастатуре, међутим ако желимо да додамо још неки уређај, у том случају препоручује се коришћење *USB* чворишта (*USB HUB*). Препоручљиво је коришћење чворишта са напајањем како се не би преоптеретио регулатор напона на основној плочи. Снабдевање *Raspberry Pi* енергијом је једноставно, само се укључи било које *USB* напајање у микро *USB* порт. Прекидач за напајање не постоји, већ се *Raspberry Pi* покреће чим се повеже на напајање, а искључује се једноставним уклањањем напајања. Поред свега, нисконапонске периферије чине га погодним за модификације хардвера. *GPIO* конектор са проредом *2.54mm (0.1 inch)* на *Raspberry Pi* омогућује приступ за *8xGPIO, UART, I2C, SPI,* као и за *3.3V* и *5V*.

Карактеристике:

- *Broadcom BCM2835 SoC*
- *700 MHz ARM1176JZF-S централно процесорско језгро (CPU core)*
- *Broadcom VideoCore IV графичка процесорска јединица (GPU)*
- *512 MB RAM меморија*
- *2 x USB2.0 порта*
- *Видео излаз за композитни сигнал (PAL и NTSC), HDMI или изворни LCD (DSI)*
- *Аудио излаз преко 3.5mm конектора или аудио преко HDMI интерфејса*
- *Складиштење: SD/MMC/SDIO*
- *10/100 Ethernet (RJ45)*

Нисконапонске периферије:

- *8 x GPIO*
- *UART*
- *I2C магистрала*
- *SPI магистрала са могућношћу избора два чипа (two chip selects)*
- *+3.3V*
- *+5V*
- *Маса*
- *Zahtevi za napajanje: 5V, 700 mA преко MicroUSB ili GPIO пинске лестивце*

- *Подржава Debian GNU/Linux, Fedora, Arch Linux, RISC OS и друге*

5.2 BeagleBone

BeagleBone је јефтин рачунар величине кредитне картице на бази **BeagleBoard** плоче са **Linux** оперативним системом, који се конектује на Интернет и покреће оперативне системе као сто су Андроид 4.0, и Убунту. Мањи је и лакши од других плоча попут **BeagleBoard-xM**-а без обзира што нема моћ процесирања као већи модели, **BeagleBone** је савршен за "физичко рачунарство" *(physical computing)* и мање интегрисане апликације. Ethernet на чипу *(On-chip Ethernet)*, и приступ нисконапонским периферијама као што су аналогно-дигитални конвертори дају предност **BeagleBone**-у приликом проширења.

Зато што поседује мноштво улаза и излаза и процесорску снагу за анализе у реалном времену коју му омогућава **TI Sitara™ AM335x ARM® Cortex™-A8** процесор **BeagleBone** се може допунити са "плашт"*(cape)* додатним *(plug-in)* плочама које му повећавају функционалност. Са преко 1,5 милиона *Dhrystone* операција по секунди и векторским аритметичким операцијама са покретним зарезом *(vector floating point)*. **BeagleBone** је у стању да се повеже са свим драјверима мотора робота *(robotics motor drivers)*, сензорима за локацију и притисак, као и 2D или 3D камерама. Поред тога може да покрене **OpenCV**, **OpenNI** и друге софтвере за прикупљање и анализу фотографија који на роботу омогућавају прикупљање и анализу фотографија уз помоћ којих можемо да га контролишемо. Захваљујући *HDMI, VGA* или *LCD* плочама за проширење може да декодује и приказује вишеструке видео формате тако што користи софтверски стек отвореног кода *(open source software stack)* и синхронизује плејбек преко **Ethernet**-а ili **USB** кабла са другим **BeagleBoard** плочама да би креирао масивне видео записе *(massive video walls)*. Ако бисмо правили 3D штампач, **BeagleBone** би нам пружио опсежне **PWM** *(Pulse-Width-Modulation)* могућности, **Ethernet** на чипу *(On-chip Ethernet)*, 3D рендеровање и могућности за манипулацију.

Све нам то може помоћи да заменимо недовољно снажну плочу на бази микроконтролера и наш стари PC.

Слика 5.2 BeagleBone плоча

Карактеристике:

- 720MHz super-scalar ARM Cortex-A8 процесор
- USB клијент и хост конектори
- Ethernet конекција
- 2 x46-пинских периферних лествица
- 2x I2C
- 5xUART
- I2S
- SPI
- CAN
- 66x3.3V GPIO
- 7xADC
- Програмабилан помоћу Cloud9 IDE развојног окружења на Node.JS softverskom систему са Bonescript библиотеком
- Испоручује се спреман за покретање уз microSD картицу од 4GB са Angstrom дистрибуцијом на Node.js софтверском систему и Cloud9 IDE развојном окружењу
- Једноставан је за клонирање захваљујући већем размаку (pitch) куглица на BGA уређају (0.8mm уместо 0.4mm), уклањању паковање-на-паковању (package-on-package) меморије, стандардној DDR2 уместо LPDDR-a, интегрисаном PHY чипу на USB-у и тако даље.

6. GNUBLIN-РАЗВОЈНА ПЛАТФОРМА

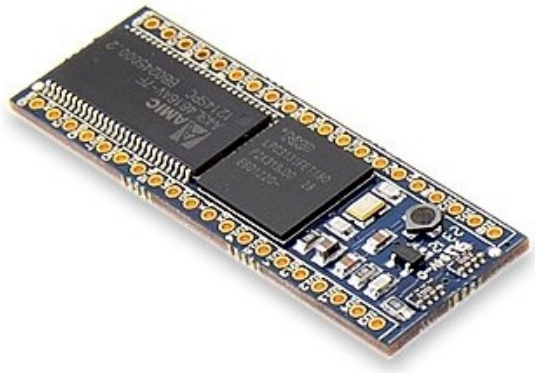
Gnublin је Embedded Linux платформа за развој контрола или типичних микроконтролерских апликација. **Gnublin** нуди све оно што нам је потребно на лак и једноставан начин. Помоћу **Gnublin** инсталера можемо сместити оперативни систем на SD меморијску картицу. Као оперативни систем користи се *Debian*, који се касније позива са **Gnublin** плоче. Међутим, ако је неопходно може се креирати сопствена дистрибуција. За контролисање, мерење и регулисање понуђени су многи додатни модули, као што су: контролери мотора, релеји, улази, излази, AD и DA конвертори, температурни сензори, и тако даље, за креирање веома брзих прототипова. Заједно са веома енергетски штедљивом **Gnublin** плочом и **Gnublin API**-који нам дозвољава да пишемо апликације у програмским језицима као што су: C++, Python, bin/bash и слично, можемо лако креирати сопствене апликације.

6.1 Развојне плоче

- GNUBLIN DIP
- GNUBLIN Extended
- GNUBLIN LAN
- GNUBLIN Standard *(биће детаљно приказана у 7. поглављу)*

За нас је најважнија *GNUBLIN Standard* плоча јер је одлична за почетнике и студенте. У даљем тексту, највише пажње ће се посветити управо овом типу развојне платформе.

6.1.1 GNUBLIN DIP



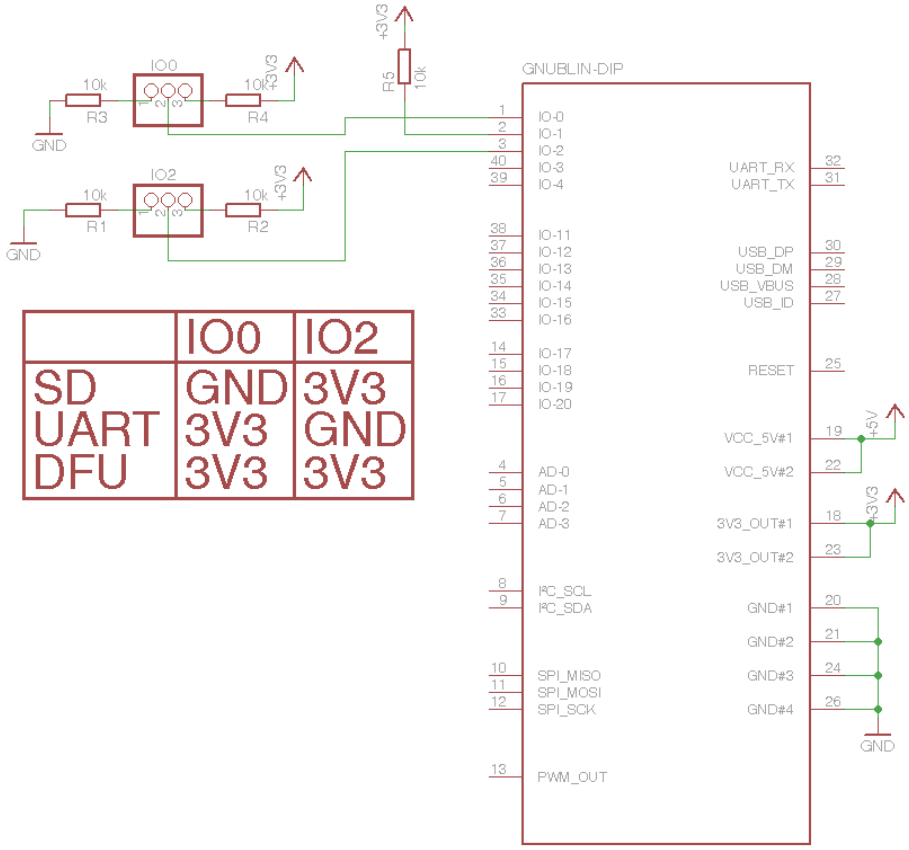
Слика 6.1.1a GNUBLIN DIP плоча

Карактеристике:

- 180MHz процесор
- 8 или 32 MB RAM-а
- SD-картица (до 16 GB)
- USB OTG (хост или уређај)
- 15 X дигиталних улаза и излаза
- 4 x аналогних улаза
- I2C
- SPI
- PWM
- Напајање 5V
- Потрошња енергије је око 100mA (5V) → 0,5Watts

Минимално повезивање:

- Да би покренули **GNUBLIN DIP,** морамо да повежемо основно коло. Покретачки уређај је подесив са *IO0* и *IO2*. *IO1* мора бити повезан са *VCC = 3.3V*.

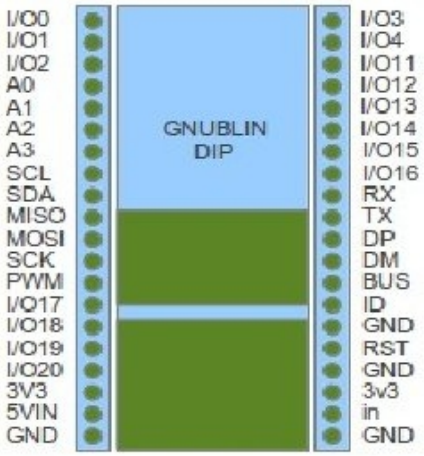


	IO0	IO2
SD	GND	3V3
UART	3V3	GND
DFU	3V3	3V3

Слика 6.1.1б Шематски приказ GNUBLIN DIP плоче

Локације пинова:

- Следећа слика показује распоред пинова и интерфејсе.



Слика 6.1.1в Распоред пинова

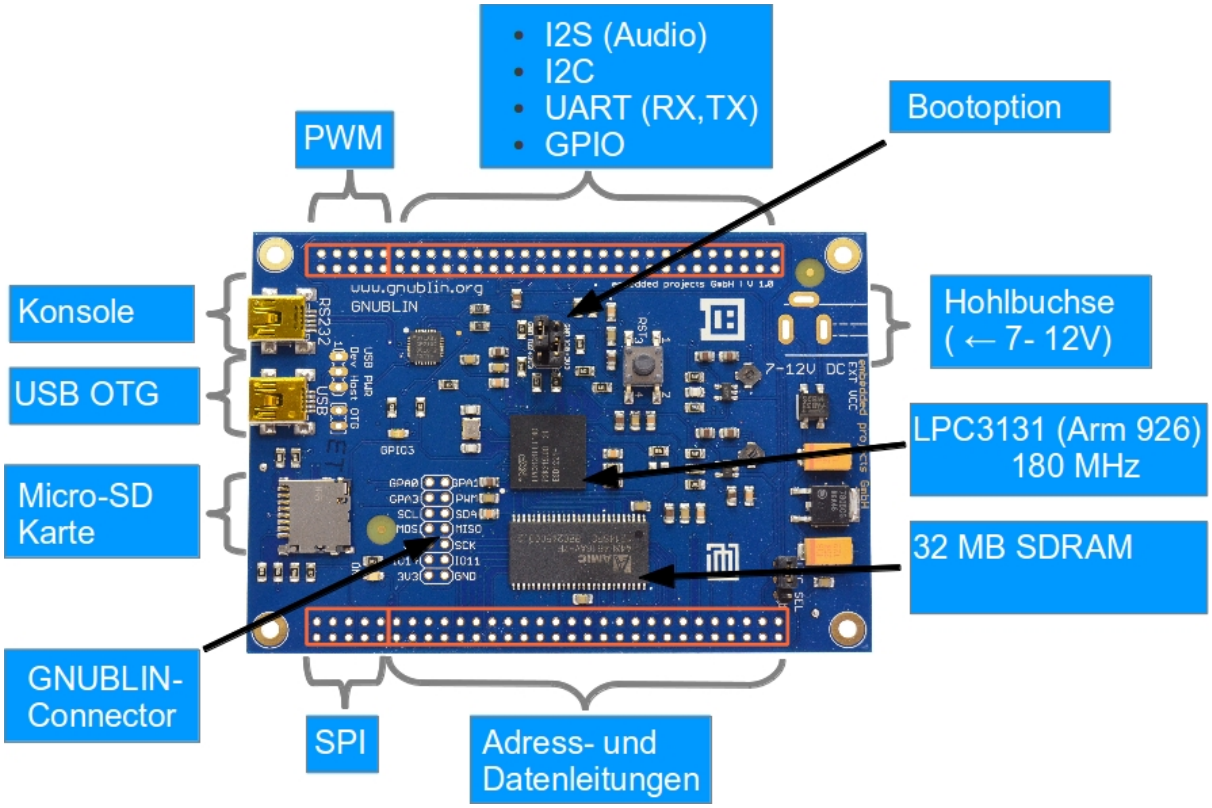
6.1.2 GNUBLIN Extended

Карактеристике:

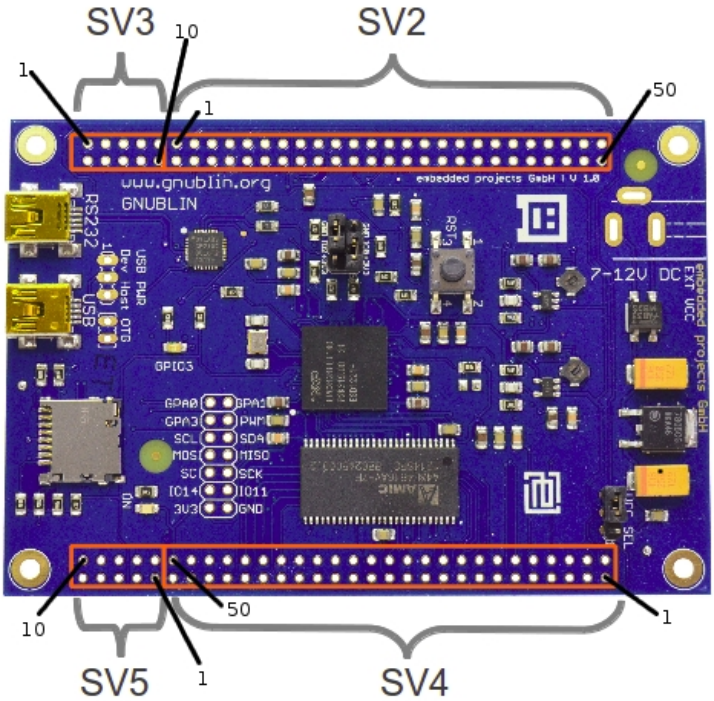
- 180MHz процесор ARM 9
- 32 MB SDRAM
- microSD картица (до 32 GB) оперативни систем, језгро, кориснички подаци
- 15 x GPIO, 4 x ADC, UART, SPI, I2C, i2S (Audio), MCI (за SD и MMC-картице), подаци и адресе доступне на 2x60 шини
- 1 x PWM
- SPI и I2C доступни на 2x7 конектору
- JTAG (SMD)
- 1 x црвена LED
- Индикатор напајања-LED
- Различите опције покретања-могућност избора коришћењем краткоспојача (џампера)-SD-картица, USB или RS232
- Екстерно могуће напајање од 7- 12 V
- Тастер за ресетовање
- USB-OTG (хост или уређај)

Интерфејси и компоненте:

Следећа слика показује **GNUBLIN Extended** плочу са свим компонентама и најважнијим интерфејсима.



Слика 6.1.2а GNUBLIN Extended ūлоча



Слика 6.1.2б Приказ бројa ūинова

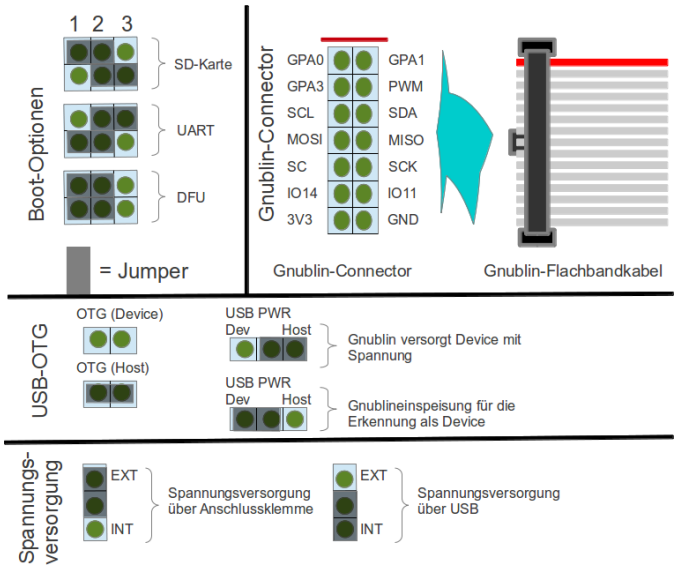
Табела 6.1.2 Функције пинова на GNUBLIN Extended плочи

Pin	Funktion	Pin	Funktion	Pin	Funktion	Pin	Funktion
SV2 - 50	GND	SV4 - 01	3V3	SV2 - 20	GPIO15	SV4 - 31	LPC_D14
SV2 - 49	3V3	SV4 - 02	GND	SV2 - 19	GPIO16	SV4 - 32	LPC_D15
SV2 - 48	I2SRX_D1	SV4 - 03	5V	SV2 - 18	GPIO13	SV4 - 33	LPC_A0
SV2 - 47	I2SRX_D0	SV4 - 04	NCS0	SV2 - 17	GPIO14	SV4 - 34	LPC_A1
SV2 - 46	I2SRX_WS1	SV4 - 05	NCS1	SV2 - 16	GPIO11	SV4 - 35	LPC_A2
SV2 - 45	I2SRX_WS0	SV4 - 06	NCS2	SV2 - 15	GPIO12	SV4 - 36	LPC_A3
SV2 - 44	I2SRX_BCK1	SV4 - 07	NCS3	SV2 - 14	LPC_MCI_D2	SV4 - 37	LPC_A4
SV2 - 43	I2SRX_BCK0	SV4 - 08	RYBN0	SV2 - 13	LPC_MCI_D3	SV4 - 38	LPC_A5
SV2 - 42	I2STX_D1	SV4 - 09	RYBN1	SV2 - 12	LPC_MCI_D0	SV4 - 39	LPC_A6
SV2 - 41	I2STX_D0	SV4 - 10	LPC_MCI_CD	SV2 - 11	LPC_MCI_D1	SV4 - 40	LPC_A7
SV2 - 40	I2STX_WS1	SV4 - 11	RYBN3	SV2 - 10	LPC_MCI_CLK	SV4 - 41	LPC_A8
SV2 - 39	I2STX_WS0	SV4 - 12	LPC_DQM1	SV2 - 09	LPC_MCI_CM	SV4 - 42	LPC_A9
SV2 - 38	I2STX_BCK1	SV4 - 13	LPC_CKE	SV2 - 08	GPIO3	SV4 - 43	LPC_A10
SV2 - 37	I2STX_BCK0	SV4 - 14	LPC_CS	SV2 - 07	GPIO4	SV4 - 44	LPC_A11
SV2 - 36	GND	SV4 - 15	LPC_WE	SV2 - 06	GPIO1	SV4 - 45	LPC_A12
SV2 - 35	I2STX_CLK0	SV4 - 16	STCS0	SV2 - 05	GPIO2	SV4 - 46	LPC_A13
SV2 - 34	I2C_SDA0	SV4 - 17	LPC_D0	SV2 - 04	5V	SV4 - 47	LPC_A14
SV2 - 33	I2C_SDA1	SV4 - 18	LPC_D1	SV2 - 03	GPIO0	SV4 - 48	LPC_A15
SV2 - 32	I2C_SCL0	SV4 - 19	LPC_D2	SV2 - 02	3V3	SV4 - 49	GND
SV2 - 31	I2C_SCL1	SV4 - 20	LPC_D3	SV2 - 01	GND	SV4 - 50	3V3
SV2 - 30	RXD	SV4 - 21	LPC_D4	SV3 - 10	GPA2	SV5 - 01	CLK256
SV2 - 29	GND	SV4 - 22	LPC_D5	SV3 - 09	GPA3	SV5 - 02	GND
SV2 - 28	RTS	SV4 - 23	LPC_D6	SV3 - 08	GPA0	SV5 - 03	SYSCLK_O
SV2 - 27	TXD	SV4 - 24	LPC_D7	SV3 - 07	GPA1	SV5 - 04	CLK_OUT
SV2 - 26	GND	SV4 - 25	LPC_D8	SV3 - 06	n.c.	SV5 - 05	CS_IN
SV2 - 25	CTS	SV4 - 26	LPC_D9	SV3 - 05	PWM_DATA	SV5 - 06	CS_OUT
SV2 - 24	GPIO19	SV4 - 27	LPC_D10	SV3 - 04	n.c.	SV5 - 07	SPI_MISO
SV2 - 23	GPIO20	SV4 - 28	LPC_D11	SV3 - 03	n.c.	SV5 - 08	SPI_MOSI
SV2 - 22	GPIO17	SV4 - 29	LPC_D12	SV3 - 02	n.c.	SV5 - 09	SCAN_TDO
SV2 - 21	GPIO18	SV4 - 30	LPC_D13	SV3 - 01	n.c.	SV5 - 10	SPI_SC

Подешавања опција покретања:

Можемо изабрати различите опције покретања користећи два краткоспајача.

Следећа слика показује положај краткоспајача и изабране опције, зависно од тога како се поставља.



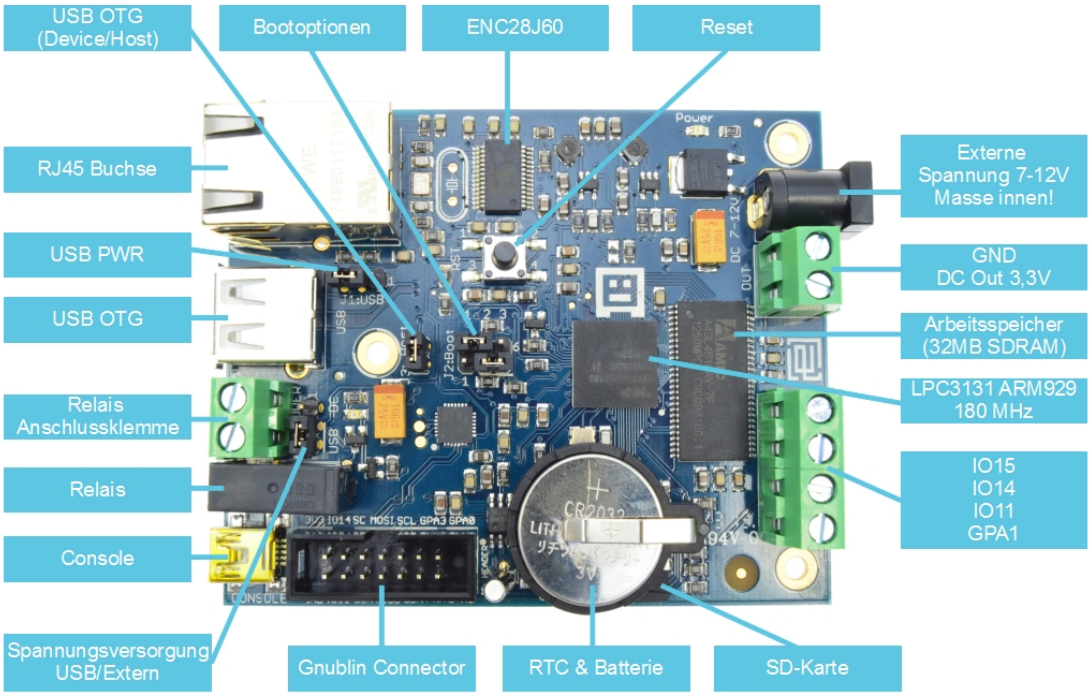
Слика 6.1.2в Положај краткоспајача (џампера)

6.1.3 GNUBLIN LAN

Карактеристике:

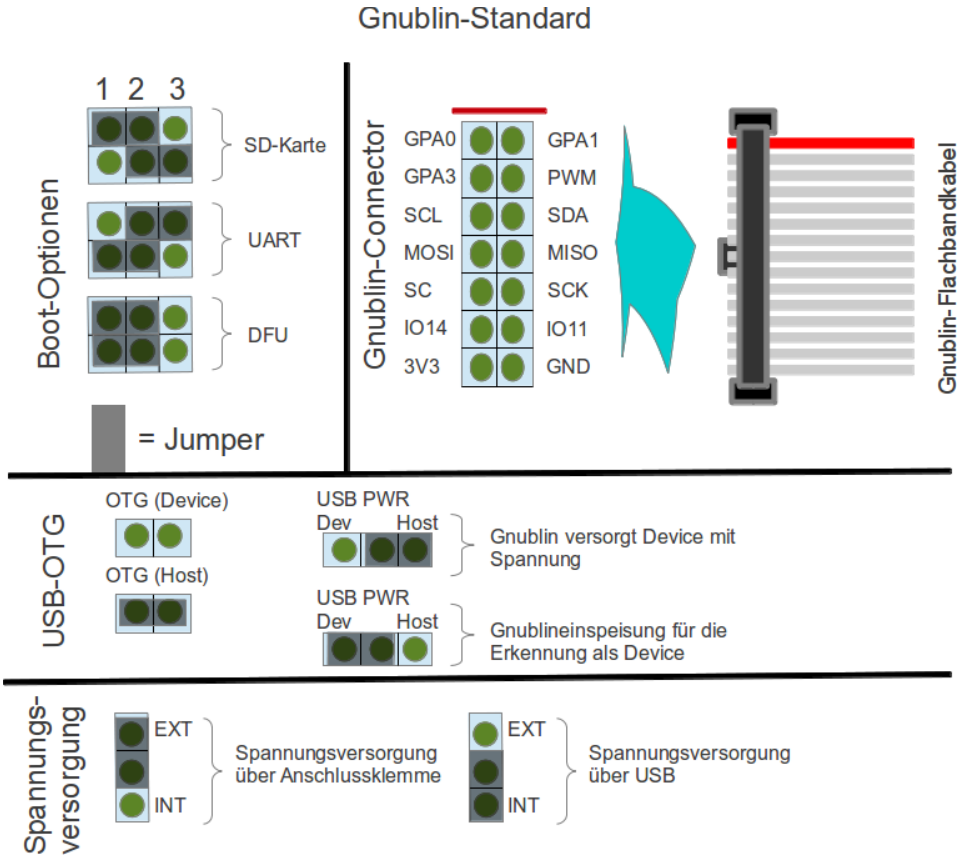
- ARM 9 180 MHz процесор
- 32 MB SDRAM
- MicroSD картица (до 32GB) Оперативни систем/ језгро/ кориснички подаци
- 1 x RJ45 мрежни порт (ENC28J60)
- 1 x интегрисана USB конзола (USB на RS232 конвертер)
- 1 x релеј на плочи
- 1 x RTC (сат са батеријом)
- GNUBLIN конектор
- 1 x LED на GPIO3
- Индикатор напајања-LED
- Опције покретања (SD картица, DFU или RS232)
- Екстерно 7-12 V напајање
- USB OTG (хост или уређај)

Да бисмо добили увид шта се од компоненти и интерфејса налази на GNUBLIN LAN плочи, анализирајмо следећу слику.



Слика 6.1.3а GNUBLIN LAN плоча

Подешавања краткоспајача:



Слика 6.1.3б Поставка краткоспајача

Конфигурисање LAN-а је приказано на страници:

- *http://en.gnublin.org/index.php/GNUBLIN-LAN*

7. GNUBLIN-STANDARD РАЗВОЈНА ПЛОЧА

Као што је већ речено, овај тип плоче је одличан за почетнике и студенте. Писање апликација за овај тип и уопште за **GNUBLIN** платформу је флексибилно, јер подржава више програмских језика, како процедуралних тако и објектно-оријентисаних (C, C++, Python, bin/bash, lua).

7.1 Архитектура GNUBLIN-Standard развојне плоче

Да бисмо се упустили у рад са овом плочом, морамо познавати неке основне карактеристике, такође морамо знати шта нам је све потребно за писање апликација.

7.1.1 Техничке карактеристике

Техничке карактеристике се односе на хардверски део односно шта све од компонената поседује ова плоча. У даљем тексту ћемо видети које су то компоненте и описати протоколе за комуникацију (*RS232*).

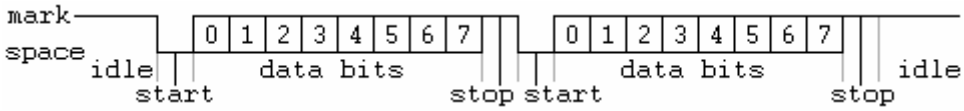
- Централно место на плочи заузима процесор LPC3131 ARM9 радног такта 180 MHz, који је базиран на моћној *RISC* архитектури.
- 32 MB SDRAM-меморија.
- Слот за *microSD* картицу – плоча не поседује интерну меморију, већ се за складиштење система и апликација користи *microSD* картица (обично је довољан капацитет од 2GB).
- USB-уређај или USB-хост/домаћин конектор (подесив са краткоспојачима).
- 3 x GPIO- ***G**eneral **P**urpose **I**nput **O**utput* пинови помоћу којих се **GNUBLIN** повезује са додатним хардвером.
- 3 x ADC-излази за аналогно-дигитални конвертор.
- 1 x PWM-***P**ulse **W**ide **M**odulation*- Импулсно ширинска модулација је врста управљања која представља начин да се од дигиталног сигнала направи сигнал аналогне вредности.
- 1 x SPI-интерфејс *(биће посебно објашњен).*
- 1 x I2C-интерфејс *(биће посебно објашњен).*
- 1 x USB-RS232 конвертор за серијски терминал.
- 1 x Црвена LED диода-налази се на пину чији је редни број три, можемо програмирати њен рад по различитим режимима.
- 1 x Индикатор напајања-LED диода-зелене боје.
- Могуће екстерно напајање (7 - 12V DC).

7.1.1.1 RS-232 стандард

RS-232C је стандард за повезивање DTE *(Data Terminal Equipment)* уређаја са DCE *(Data Communication Equipment)* уређајима применом асинхроне серијске везе релативно малих брзина. Рачунар спада у DTE уређаје, док је типичан пример DCE уређаја модем. C-верзија је усвојена 1969. године. Касније је стандард мало модификован и преименован у EIA-232, а од 1988. у TIA-232, да би 1997. био усвојен данас важећи стандард TIA-232-F. Разлике у односу на RS-232C су углавном у спецификацијама временских дијаграма и промени стандардом дефинисаних имена неких сигнала али суштинских разлика нема.

EIA је скраћеница за друштво под називом *Electronic Industries Association*, које се 1988. трансформисало у *Telecommunications Industry Association* (TIA).

- Асинхрона серијска веза битске брзине до 115,2 Kbits/s (оригинално, стандард дозвољава битске брзине до 20 Kbits/s). Битске брзине које се данас користе су 9,6 Kb/s 19,2Kb/s, 33,6Kb/s, 56Kb/s, 115,2Kb/s али и веће брзине, у везама ван стандарда.
- Стандард за везу од тачке до тачке
- Максимална дужина кабла доста зависи од брзине. По стандарду, за брзину 19,2Kb/s, дозвољена је дужина кабла 15m (50ft).
- Несиметрична веза са спољним сигналима.
- Напонски сигнали максималних нивоа −25V (логичка јединица-*mark*) и +25V (логичка нула-*space*) иако стандард дозвољава нивое ±25V, данас је уобичајено користити ±12V и мање.
- Пријемник мора да буде у стању да све сигнале на улазу између −3V и −25V прими као логичку јединицу, а све сигнале између +3V и +25V као логичку нулу.
- Стандард дефинише излазну отпорност предајника мању од 50Ω и улазну отпорност пријемника 3kΩ и 7kΩ уз улазну капацитивност мању од 2,5nF.
- Подржана дуплекс веза.
- Стандард дефинише облик рама у који се пакује податак који се преноси. Рам се састоји од једног старт-бита и једног стоп-бита. У рам се поред података (5-8 бита) може убацити још и бит парности.

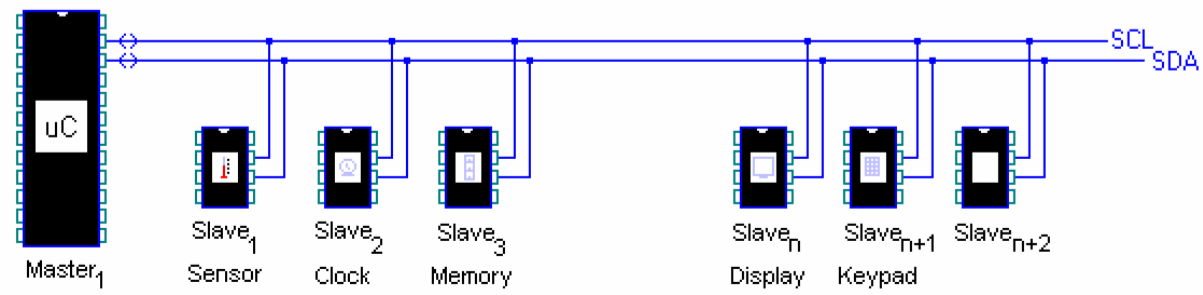


Слика 7.1.1.1 Изглед сигнала на линијама за пренос података

Слика изнад показује изглед сигнала на линијама за пренос података са два логичка нивоа. Логичка јединица се у литератури означава са *mark*, а логичка нула *space*. На слици су два бајта која се преносе непосредно један за другим. Стање када се преко линије не преноси ништа се обично назива *"idle"* и тада је на воду логичка јединица. Старт-бит је логичка нула, а стоп-бит је логичка јединица.

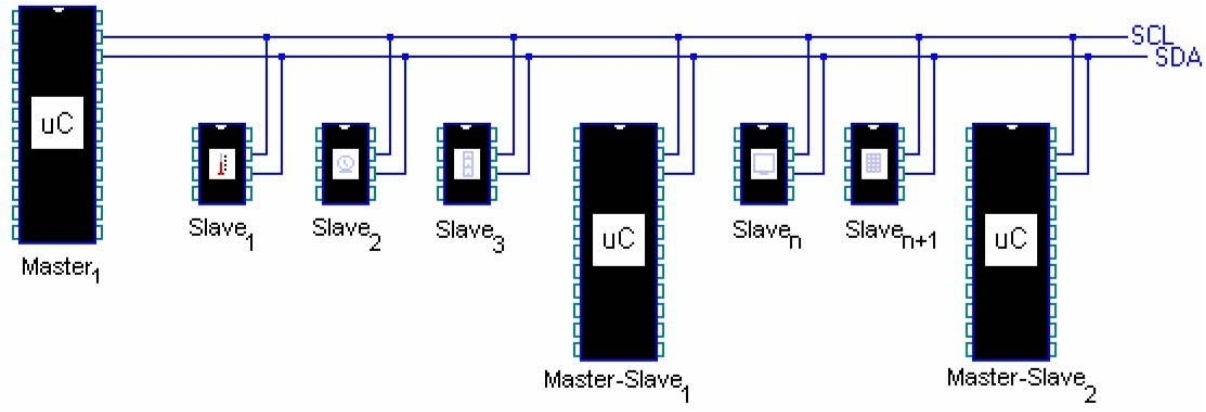
7.1.1.2 I2C (или I IC *Inter Integrated circuit Communication*)

Стандард је увео Филипс као јефтину и ефикасну замену за паралелно повезивање микроконтролера са периферијама. I2C је магистрала при чему на магистрали мора да постоји бар један главни (*master*) уређај, обично микроконтролер, и више споредних уређаја (*slaves*). Могуће је постојење више главних уређаја а евентуални конфликт је решен такозваним процесом арбитраже, где један од главних одустаје од комуникације и чека.



Слика 7.1.1.2а I2C магистрала са једним главним уређајем

Главни уређај генерише такт и адресира споредне. Сваки споредни мора да има јединствену адресу. Протоколом је дефинисано поље адресе величине 7 бита у пакету који се преносе. Шеснаест адреса је резервисано, што значи да је могуће да постоји укупно 112 учесника на магистрали. Новија проширења стандарда омогућавају 10-битну адресу и преко 1000 учесника на магистрали.



Слика 7.1.1.2б I2C магистрала са више главних уређаја

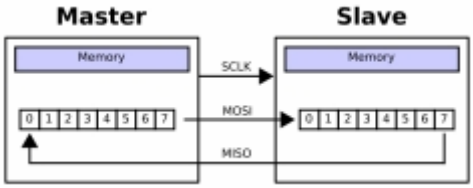
Магистралу чине само два вода: вод такта означен са SCL и вод података SDA. Оба вода захтевају излазе са отвореним колектором.

Основни стандард је дефинисао три брзине преноса 10 Kbit/s-LM (*low-speed mode*), 100 Kbit/s-SM (*standard mode*) і 400 Kbit/s-FM (*fast mode*). Данашњи стандард верзија I2C 2.1 из 2001. године дефинише и 1 Mbit/s-Fm+ и 3.4 Mbit/s-HS (*High Speed mode*). Преко I2C се повезују AD, DA конвертори, меморије, сензори, дисплеји и многе

друге периферије. Огроман број чипова подржава ову комуникацију.

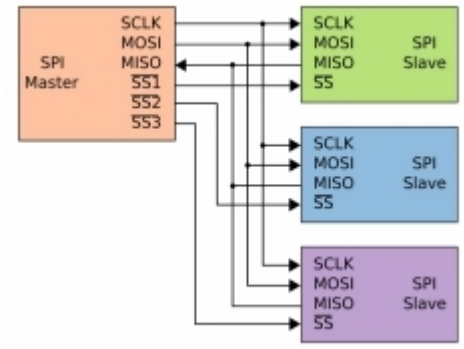
7.1.1.3 SPI (Serial Peripheral Interface-серијски периферни интерфејс)

SPI је једноставан начин серијског синхроног преноса података настао у компанији Моторола, првенствено за спајање микроконтролера са периферијама (улога је слична I2C стандарду и једножичној вези). **SPI** се још и назива четворожична веза због четири сигнала који учествују у комуникацији. **SPI** остварује двосмерну, дуплекс везу између једног главног уређаја и једног или више споредних уређаја.



Слика 7.1.1.3а Веза „1 на 1“

Главни, који је по правилу микроконтролер, генерише такт (*SCLK*) који је између 1MHz и 70MHz. И главни и споредни уређаји имају у себи осмобитни (мада су дозвољене и друге величине) померачки регистар. Комуникација почиње када главни рачунар активира сигнал *SS*(*slave select*), то јест постави логичку нулу на ту линију (сигнал *SS* назначен на слици). Комуникација је увек двосмерна, један бајт иде од главног ка споредном, други бајт истовремено се преноси од споредног ка главном уређају. Оба преноса се обављају истовремено, чак и када неки од њих нема смисла. На пример, ако микроконтролер шаље већи број бајтова ка периферији, а периферија нема података које би слала микроконтролеру, смер од периферије ка микроконтролеру преноси податке који немају смисла, али се ипак преносе.



Слика 7.1.1.3б Веза „1 на више“

Сигнали који учествују у комуникацији су:

- SCLK – Сигнал такта који генерише главни уређај. Смер је од главног ка споредном.

- MOSI (*Master Output Slave Input*) – Пренос од главног ка споредним уређајима.
- MISO (*Master Input Slave Output*) – Пренос од споредног ка главном уређају.
- SS (*slave select*) – Сигнал који генерише главни уређај. Нула на овом воду отпочиње комуникацију.

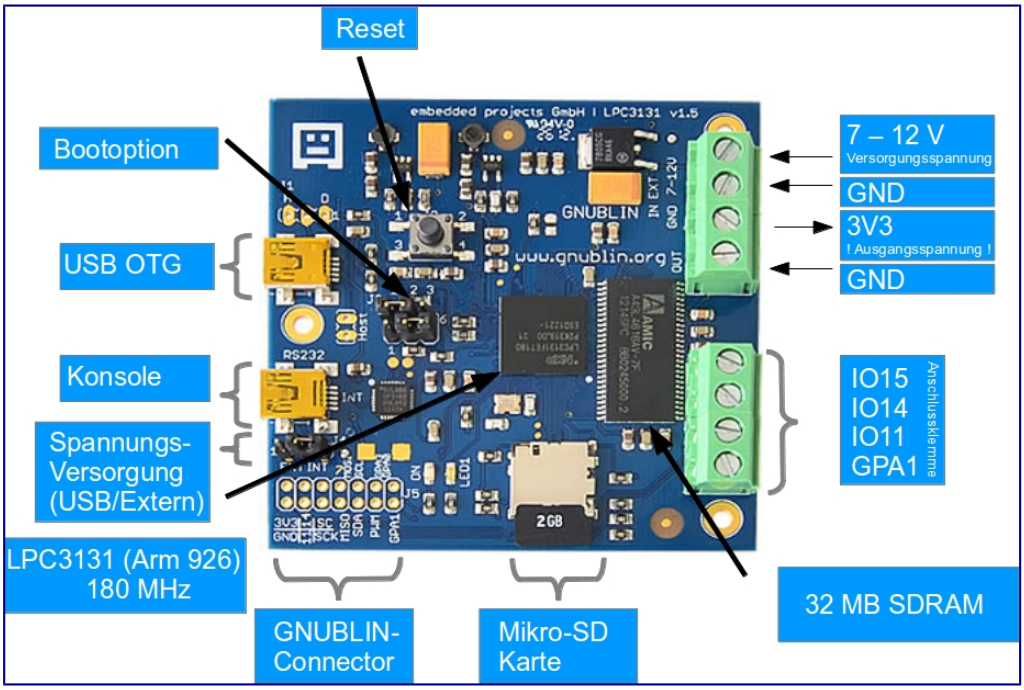
Веза са више споредних уређаја захтева да главни (*master*) обезбеди посебне SS (*slave select*) сигнале за сваки споредни (*slave*). Оваква веза захтева и да споредни уређаји на свом излазу *MISO* имају коло са стањем високе импедансе како не би дошло до сукоба више излаза који су везани за исту тачку.

Постоји могућност повезивања више споредних уређаја у ланац (*dasy chain*) када је излаз главног везан на улаз првог споредног, његов излаз за улаз другог споредног и тако даље. Излаз последњег споредног у ланцу везан је за улаз главног уређаја.

7.1.2 Компоненте

Плоча је дизајнирана тако да се лако могу разумети све опције и могућности које модеран интегрисани систем може да понуди.

Плоча покреће кернел и фајл систем комплетно са SD картице. Након процеса покретања, можемо се улоговати преко серијског терминала и почети са неким примерима писаним у програмским језицима као што су: Python, Perl, Bash, Lua, и тако даље. Можемо такође користити едиторе VIM и NANO за испис неких текстуалних фајлова.



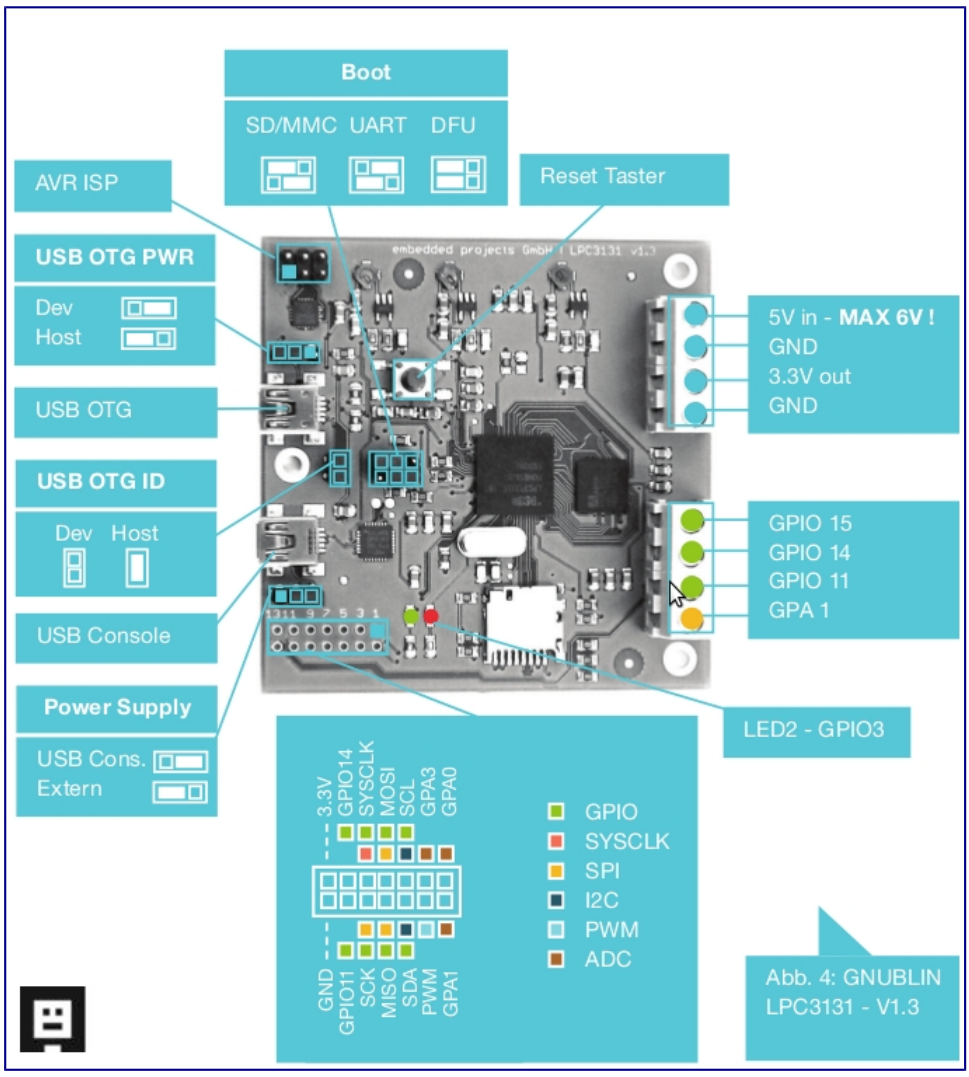
Слика 7.1.2а Компоненте GNUBLIN-STANDARD плоче

Плоча се састоји од два слоја, тако да лако можемо пратити водове и пронаћи их сондом осцилоскопа. Уместо интерне меморије коју је компликовано програмирати, користи се SD картица. Уместо програматора, ту је *PC* софтвер помоћу кога можемо лако програмирати SD картицу. Сви важни интерфејси су повезани са 2x7 заглављем.

Ако желимо да повежемо *GNUBLIN* са локалном мрежом, можемо то учинити са USB-LAN адаптером или са USB-WLAN адаптером. USB интерфејс *GNUBLIN* плоче може радити као уређај или хост/домаћин, па према томе можемо повезати USB хардвер који желимо, као на пример LAN, WLAN, Bluetooth, веб-камеру, звучну картицу, и тако даље.

7.1.3 Поставке

Као што је већ поменуто различита подешавања се могу извршити на плочи. Дакле, могуће је користити краткоспајаче (џампере) за конфигурисање напајања (USB или екстерно) и USB опцију (изабрати да *GNUBLIN* буде хост/домаћин или уређај). Следећи дијаграм показује конфигурационе опције.



Слика 7.1.2б Дијаīрам конфиīурационих оīција

7.1.4 Интерфејси

На плочи можемо пронаћи све уобичајене интерфејсе типичног микроконтролера. Осим тога, можемо пронаћи све драјвере интерфејса у језгру и почети да их контролишемо из корисничког простора. Могуће је укључити црвену LED диоду сетом команди:

```
  echo 3 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio3/direction
  echo 1 > /sys/class/gpio/gpio3/value
```

Такође је могуће користити интерфејсе са *fopen*, *fwrite*, *fread* и тако даље из различитих програмских језика C++, Lua, Squirrel, Bash и тако даље.

8. ПРАКТИЧНА РЕАЛИЗАЦИЈА ПРОЈЕКТНОГ ЗАДАТКА

У оквиру практичне реализације обрадићемо контролу LED светла (укључивање / искључивање и промена режима рада) и повезивање Bluetooth модула са GNUBLIN плочом.

8.1 Контрола LED светла

За овај задатак потребни су нам:

- GNUBLIN-STANDARD плоча
- PC рачунар (десктоп или лап-топ) са инсталираним оперативним системом Linux (*Debian, Ubuntu или Mint*)
- Системски програм – PICOCOM који омогућава учитавање и покретање система са плоче
- Комуникациони медијум - USB кабл

Израда овог задатка се одвија у неколико корака:

- Инсталација системског програма - PICOCOM
- Повезивање плоче са рачунаром
- Покретање системског програма – PICOCOM
- Учитавање и покретање система са плоче (инсталиран на SD картици)
- Повезивање LED светла са плочом
- Писање апликација за контролу и њихово позивање

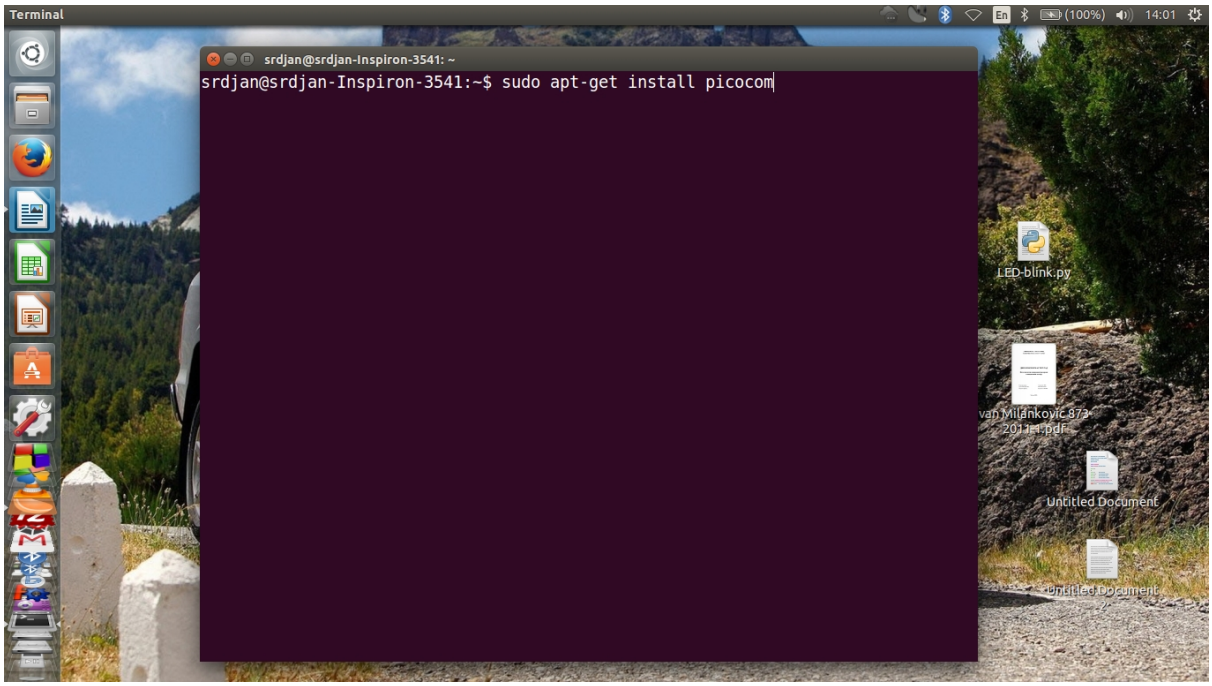
8.1.1 Инсталација системског програма – PICOCOM

PICOCOM представља системску апликацију која нам омогућава комуникацију са плочом (учитавање и покретање система са плоче).

Инсталација овог програма је једноставна, позивањем терминала на нашем оперативном систему и уписивањем следеће команде:

sudo apt-get install picocom

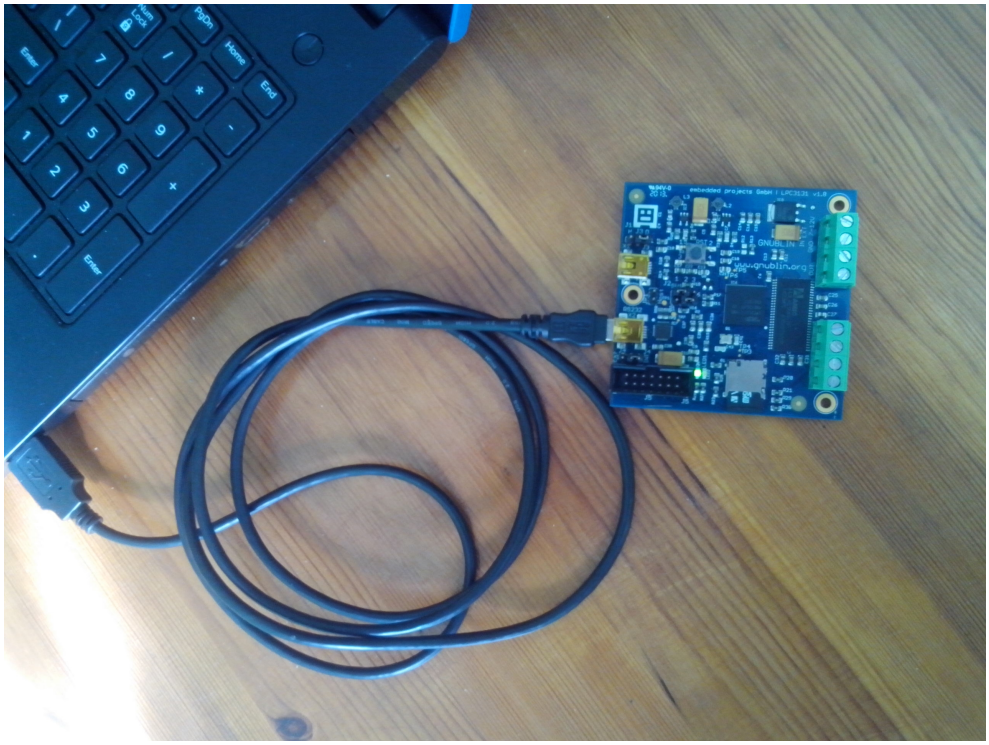
Притиском на тастер *ENTER*, отпочеће инсталација програма.



Слика 8.1.1 Терминал са исūисаном командом за инсūалацију ūроīрама

8.1.2 Повезивање GNUBLIN-STANDARD плоче са рачунаром

Након инсталације системског програма, потребно је правилно повезати плочу са рачунаром. Следећа слика показује правилно повезану плочу са рачунаром.

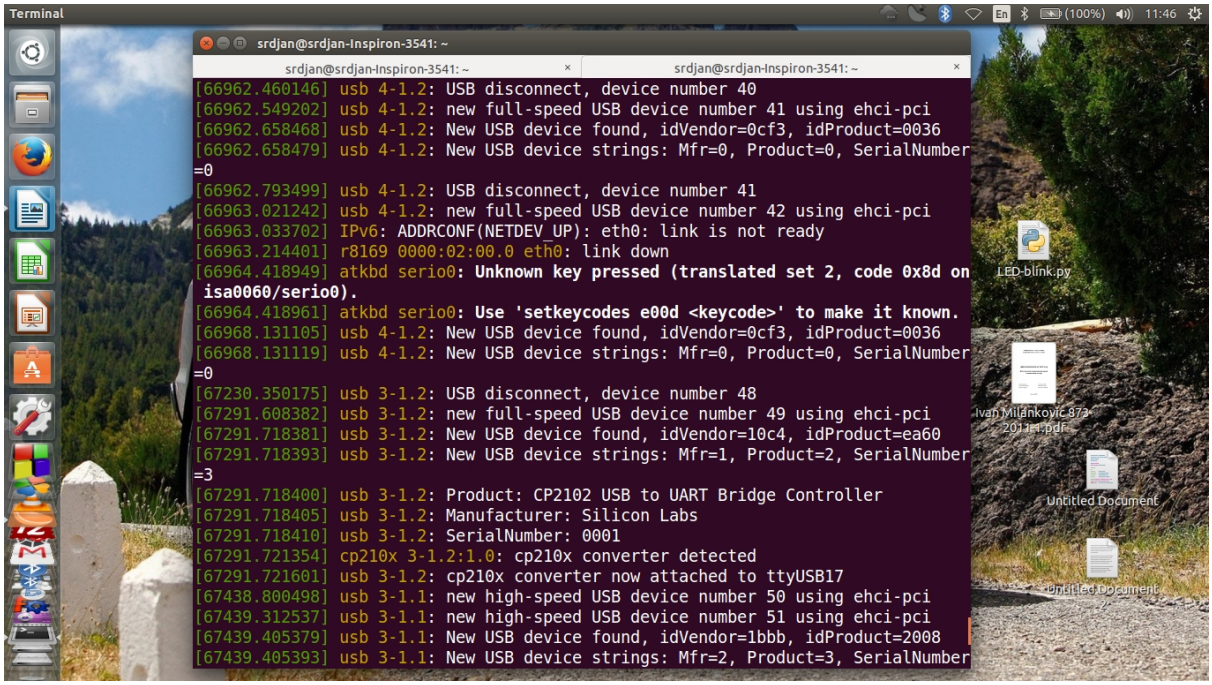


Слика 8.1.2 Правилно ūовезана ūлоча са рачунаром

Плоча се повезује са рачунаром путем USB кабла. Показатељ правилног повезивања је индикатор LED зелене боје.

8.1.3 Покретање системског програма – PICOCOM

Као што је већ речено, PICOCOM представља системску апликацију која нам омогућава комуницирање са плочом, учитавање и покретање система са плоче. Пре самог позива програма, потребно је повезати плочу са рачунаром и проверити који је уређај у питању у нашем случају је *ttyUSB17* а некад може да буде и *ttyUSB0*. Провера се може извршити уносом команде *dmesg* као на слици.



Слика 8.1.3а Преīлед уређаја

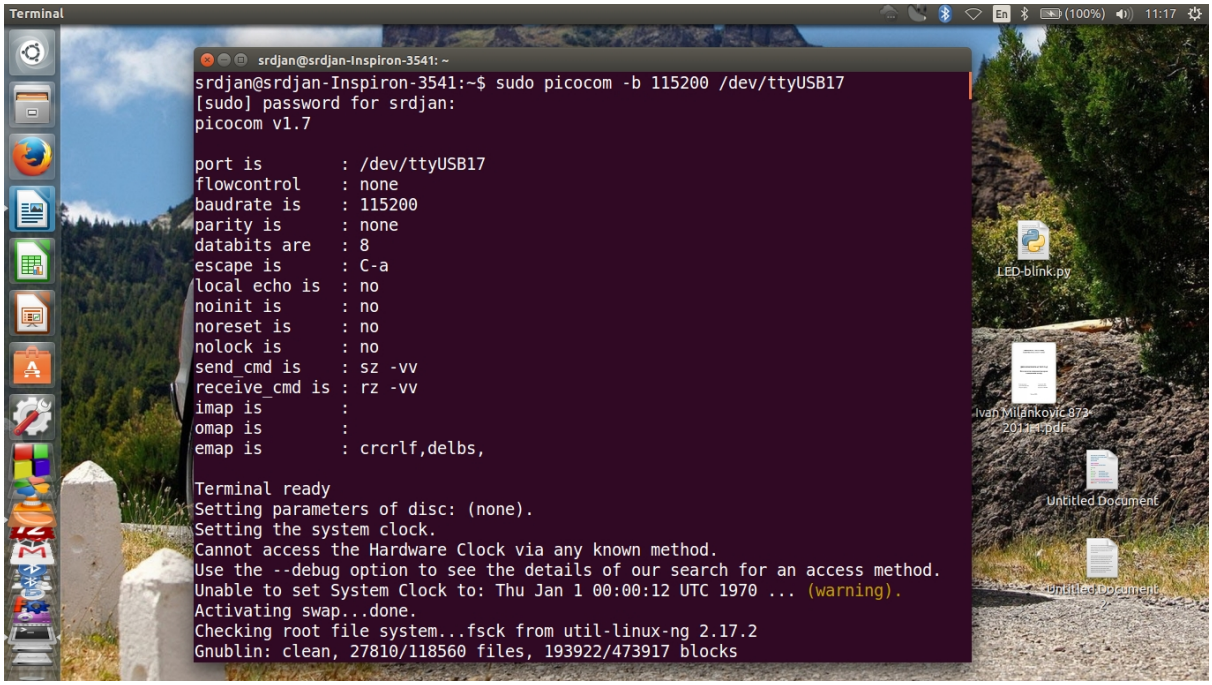
Пошто је у питању уређај *ttyUSB17*, он се налази у директоријуму */dev/ttyUSB17*.

Покретање програма извршава се уносом следеће команде у терминал:

sudo picocom -b 115200 /dev/ttyUSB17

Где је:

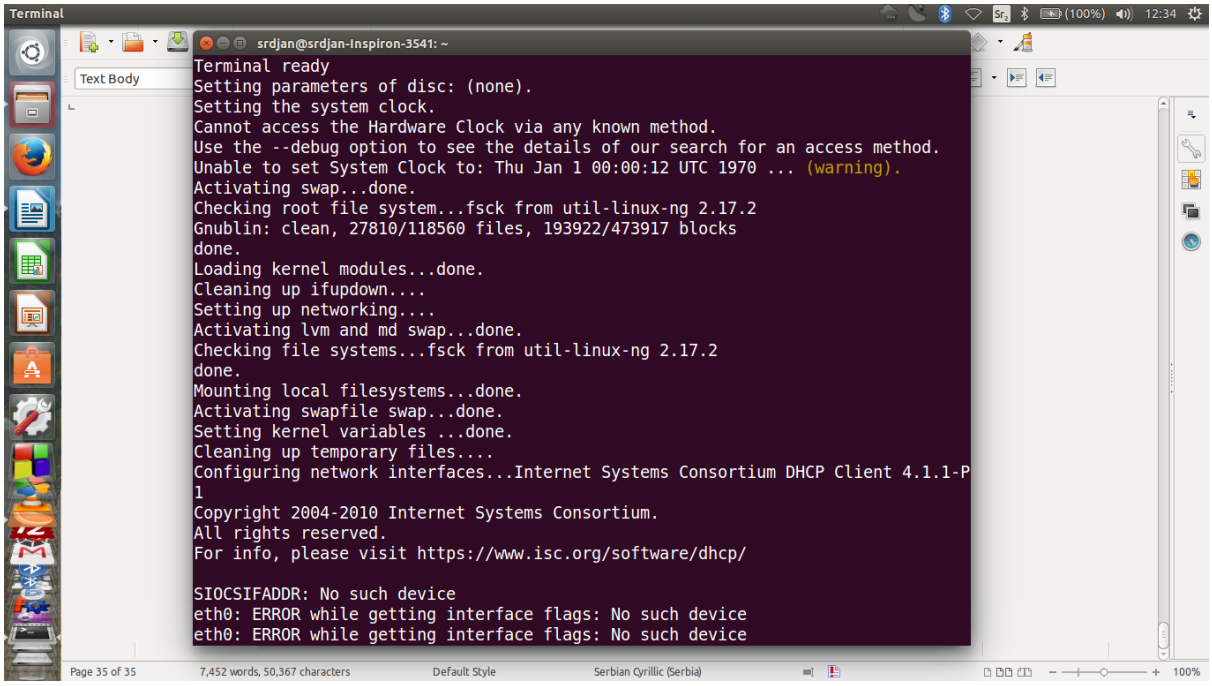
- **sudo picocom** – позив програма
- **-b 115200** – baud rate – брзина преноса изражена у битовима по секунди
- **/dev/ttyUSB17 –** локација уређаја



Слика 8.1.3б Покретање PICOCOM-а

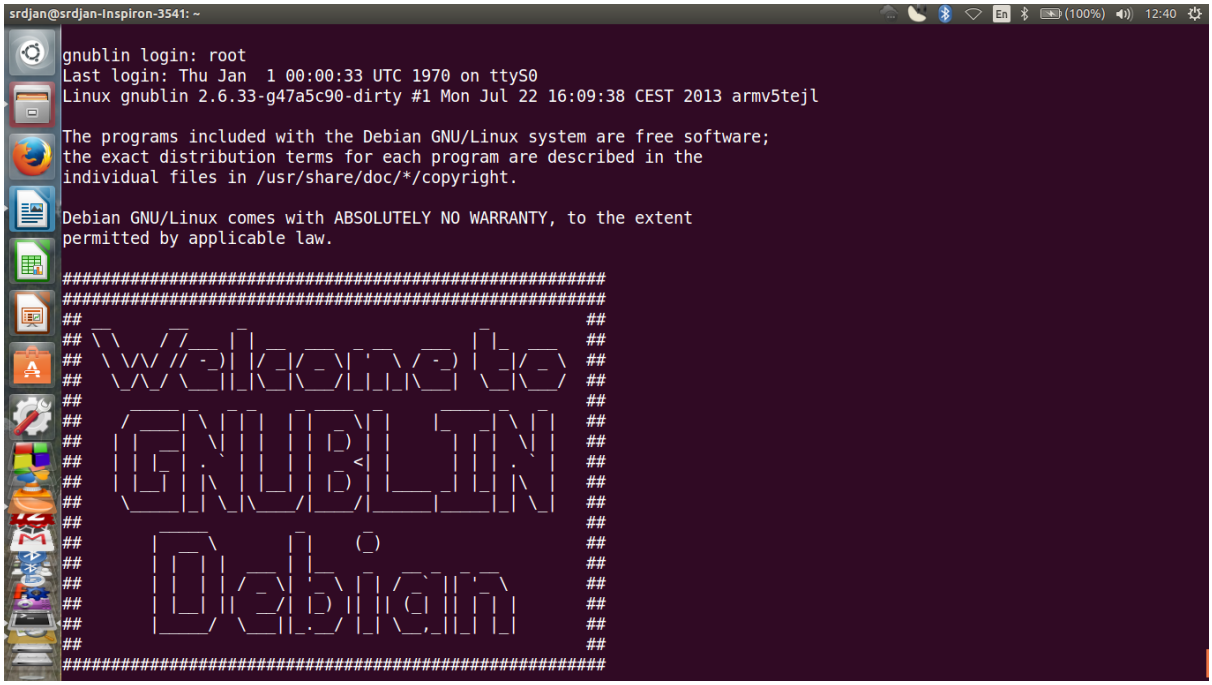
8.1.4 Учитавање и покретање система са плоче

Након позивања PICOCOM-а следи учитавање и покретање система са плоче. Систем је инсталиран на SD картици, што омогућава чување великог броја апликација писаних у C, C++, Python, Bash и Lua програмским језицима. GNUBLIN користи оперативни систем GNU Linux Debian 6.0. Следећа слика показује учитавање и покретање система.



Слика 8.1.4а Учитавање система

Након учитавања приступа се покретању, морамо се улоговати као *root* корисник да бисмо имали све привилегије.



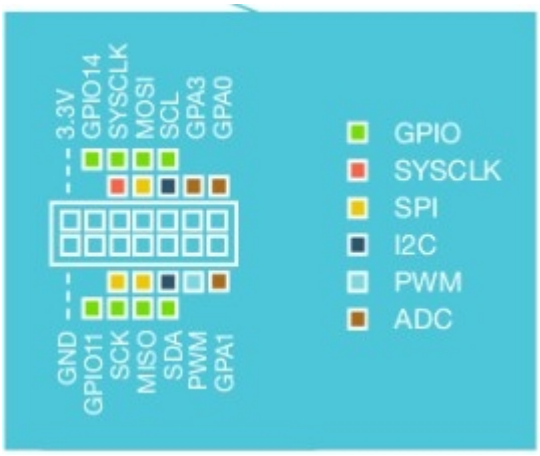
Слика 8.1.4б Покретање система

8.1.5 Повезивање LED светла са плочом

За овај део задатка, потребно нам је неко LED светло. Пример је дат на следећој слици.

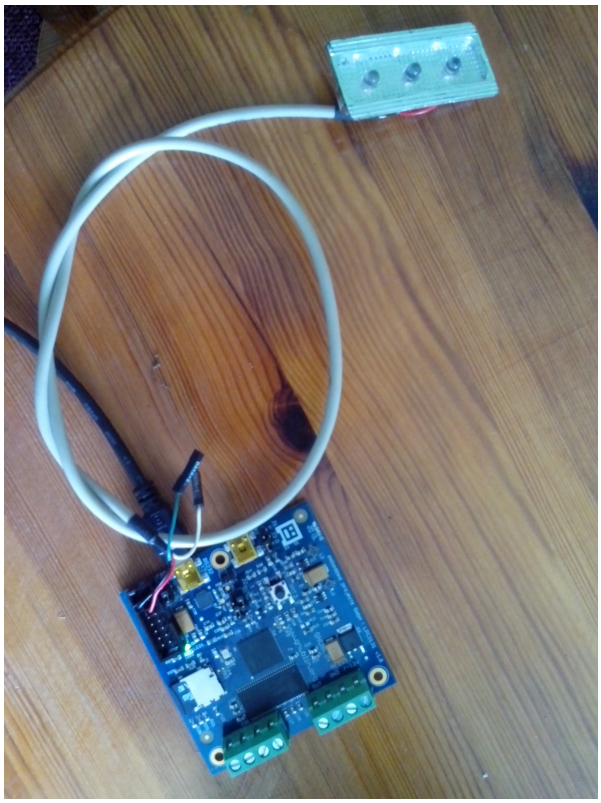
Слика 8.1.5а LED светло

За повезивање, користићемо два вода светла и то вод GND (ground (-)) и вод VCC (+). Светло се повезује на GNUBLIN конектор, на полеђини плоче дате су ознаке пинова.



Слика 8.1.5б Ознаке пинова на GNUBLIN конектору

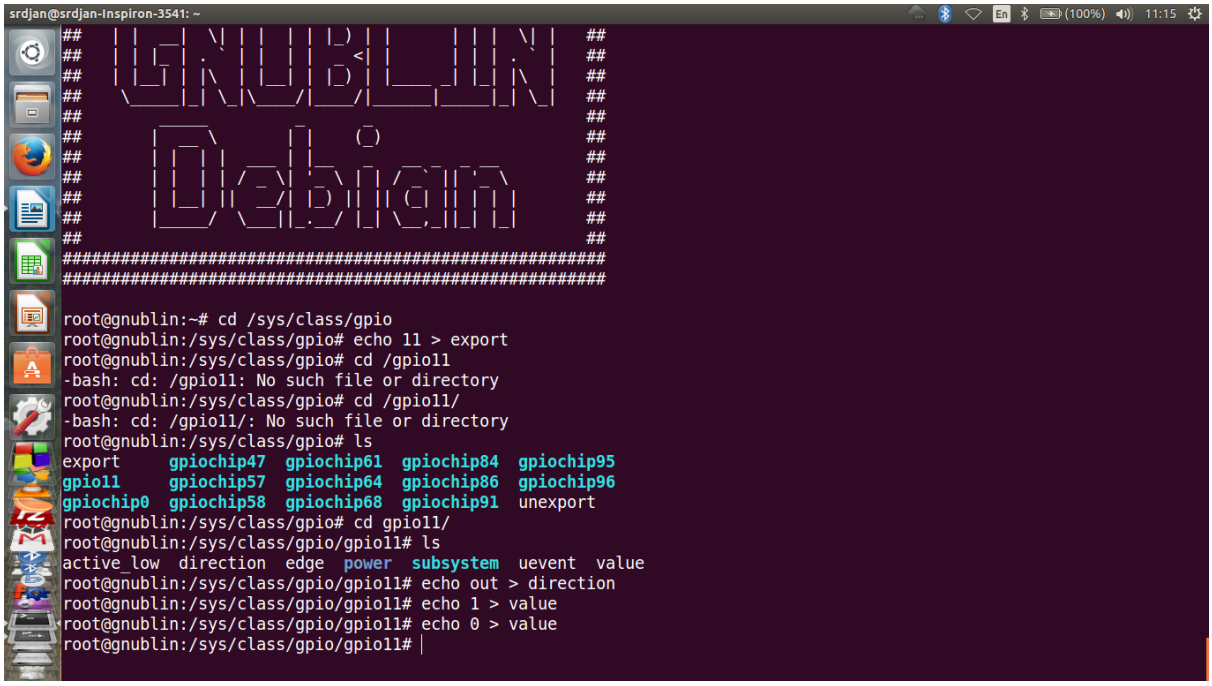
Правилна веза се остварује тако што вод светла GND, се везује на пин GND конектора и вод светла VCC се везује на неке од слободних GPIO пинова (GPIO14 или GPIO11). Уколико би везали VCC светла на пин са ознаком 3.3V, у том случају би имали беспрекидан рад светла.



Слика 8.1.5в Правилно повезано светло са GNUBLIN плочом

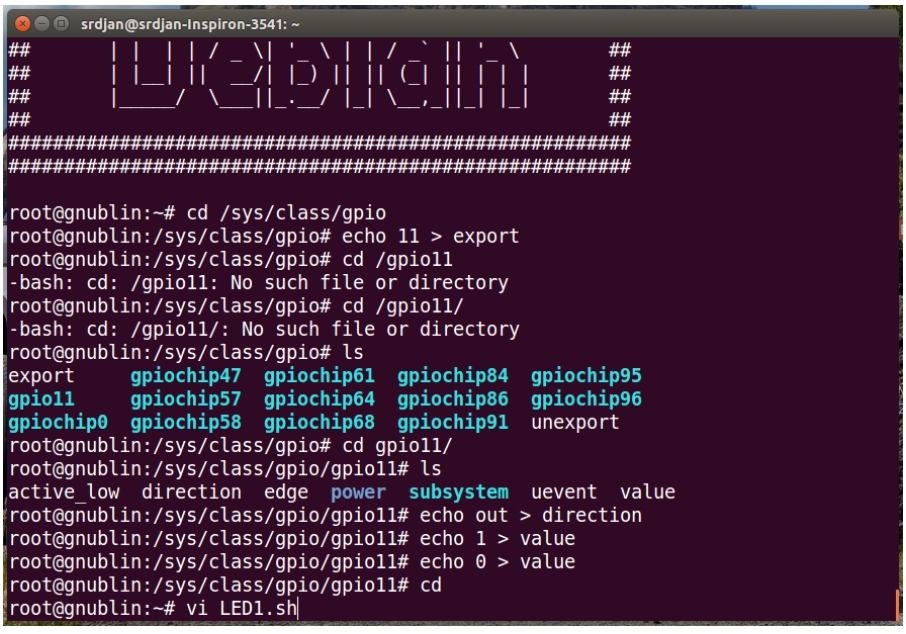
8.1.6 Писање апликација за контролу светла и њихово позивање

Ако смо извршили све претходне радње, улоговали се у систем плоче, можемо почети са креирањем апликација за контролу рада светла и промену режима рада. Као излазни пин користићемо GPIO11. Даље је неопходно откључати пин, а то се постиже сетом инструкција-команди.

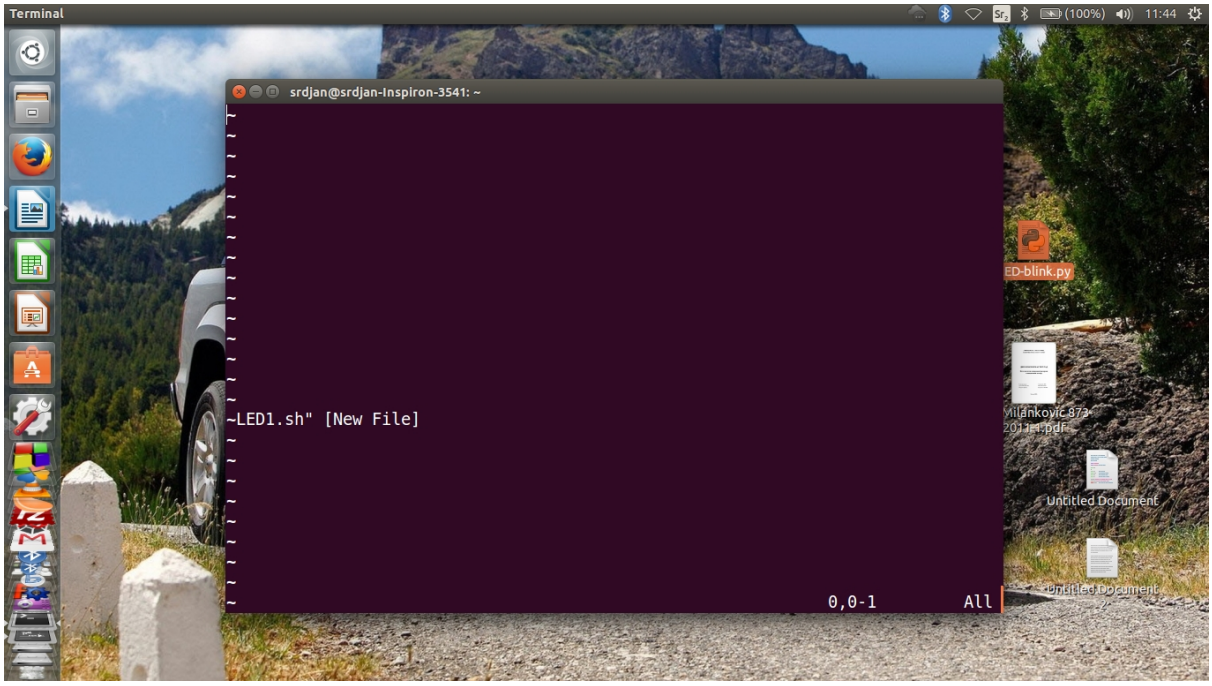


Слика 8.1.6а Команде за отклучавање излазног пина

Пошто смо извршили овај корак, командом *cd* враћамо се на почетак. За писање кода користићемо Shell (#!bin/bash) интерпретирани програмски језик и *vi* едитор за исписивање кода. Приликом креирања фајла, мора се означити екстензија која је за апликације писане у Shell-у „ *.sh* “.

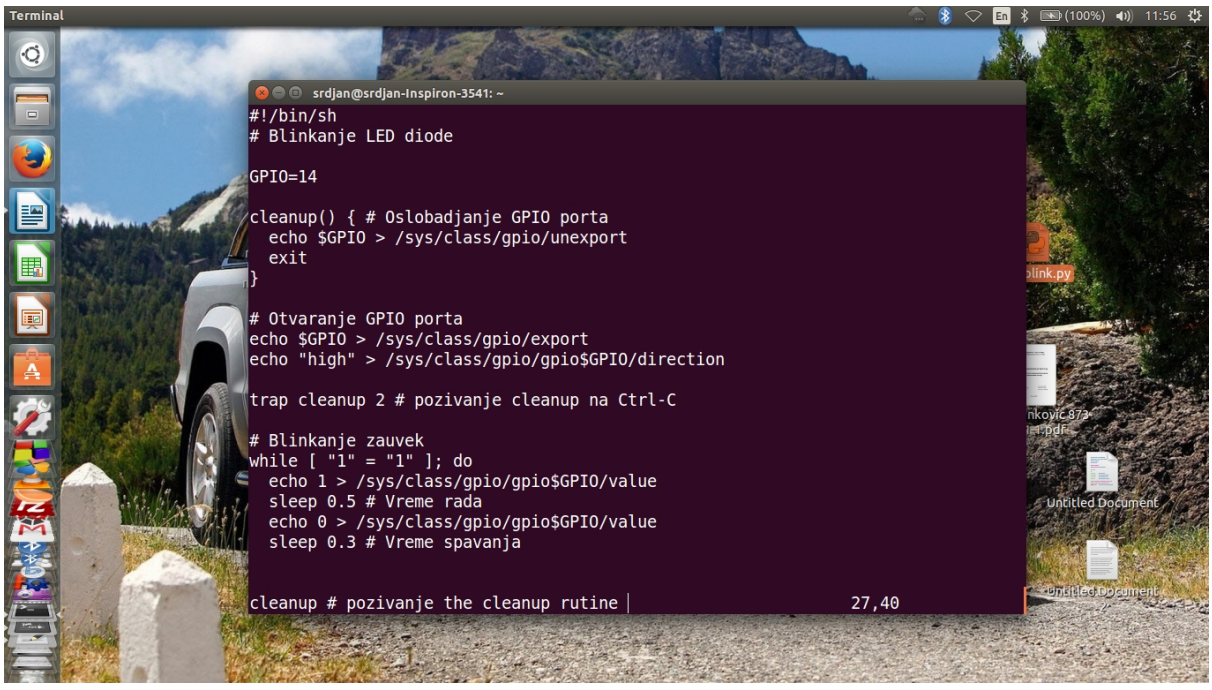


Слика 8.1.6б Позивање „vi“ едитора

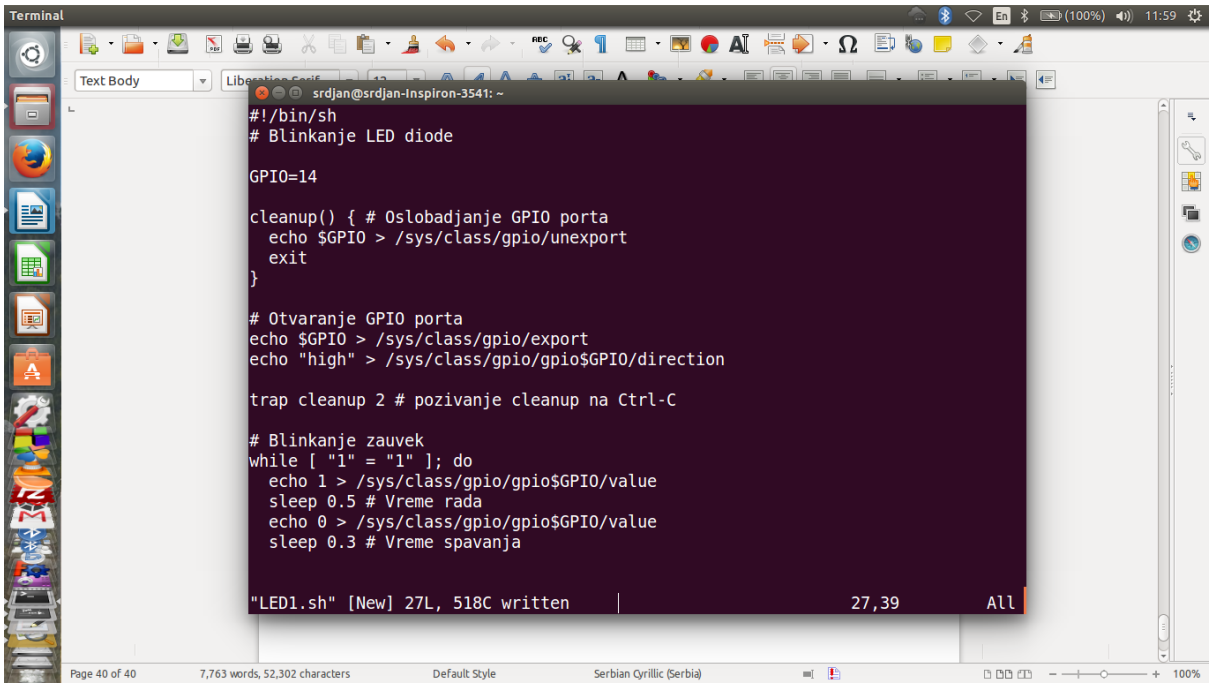


Слика 8.1.6в „vi“ едитор

Затим морамо да омогућимо унос кода у едитор, а то се извршава притиском на тастер „I“ на тастатури (INSERT мод). По завршетку уноса комбинацијом CTRL+C затварамо мод за унос. Чување фајла је једноставно, након затварања мода за унос притиском на „:“ и „w“ (:w) у истом реду омогућавамо да фајл буде сачуван на картици.



Слика 8.1.6г INSERT мод



Слика 8.1.6g Уписивање фајла на картицу

Пример кода:

```
#!/bin/sh
# Блинкање LED диоде

GPIO=14

# Ослобађање GPIO порта
cleanup() {
  echo $GPIO > /sys/class/gpio/unexport
  exit
}
# Отварање GPIO порта
echo $GPIO > /sys/class/gpio/export
echo "high" > /sys/class/gpio/gpio$GPIO/direction

trap cleanup 2 # Позивање cleanup на Ctrl-C

# Блинкање заувек
while [ "1" = "1" ]; do
  echo 1 > /sys/class/gpio/gpio$GPIO/value
  sleep 0.5 # Време рада израженог у секундама
  echo 0 > /sys/class/gpio/gpio$GPIO/value
  sleep 0.3 # Време прекида израженог у секундама
```

```
done
cleanup # Позивање the cleanup рутине
```

Позивање фајла се извршава инструкцијом „*sh LED1.sh*“. И као резултат добићемо укључивање и искључивање светла у зависности од времена дефинисаног у коду (израженог у секундама).

8.2 Повезивање Bluetooth модула са GNUBLIN – ом

У оквиру овог задатка реализоваћемо повезивање модула и његово пријављивање на неке од уређаја (мобилни телефон или лап-топ рачунар).

8.2.1 Bluetooth модул-карактеристике

Технички детаљи:

- BLUEGIGA WT41 Bluetooth® 2.1 са EDR модулом.
- Подржане комуникације са платформама: UART (Rx, Tx), SPI (MOSI, MISO, SCK, CS), SDA, SCL, RST, AN, INT, и PWM линије.
- Модул је дизајниран тако да ради само на 3.3V.
- Зелена LED диода – показује присуство напајања

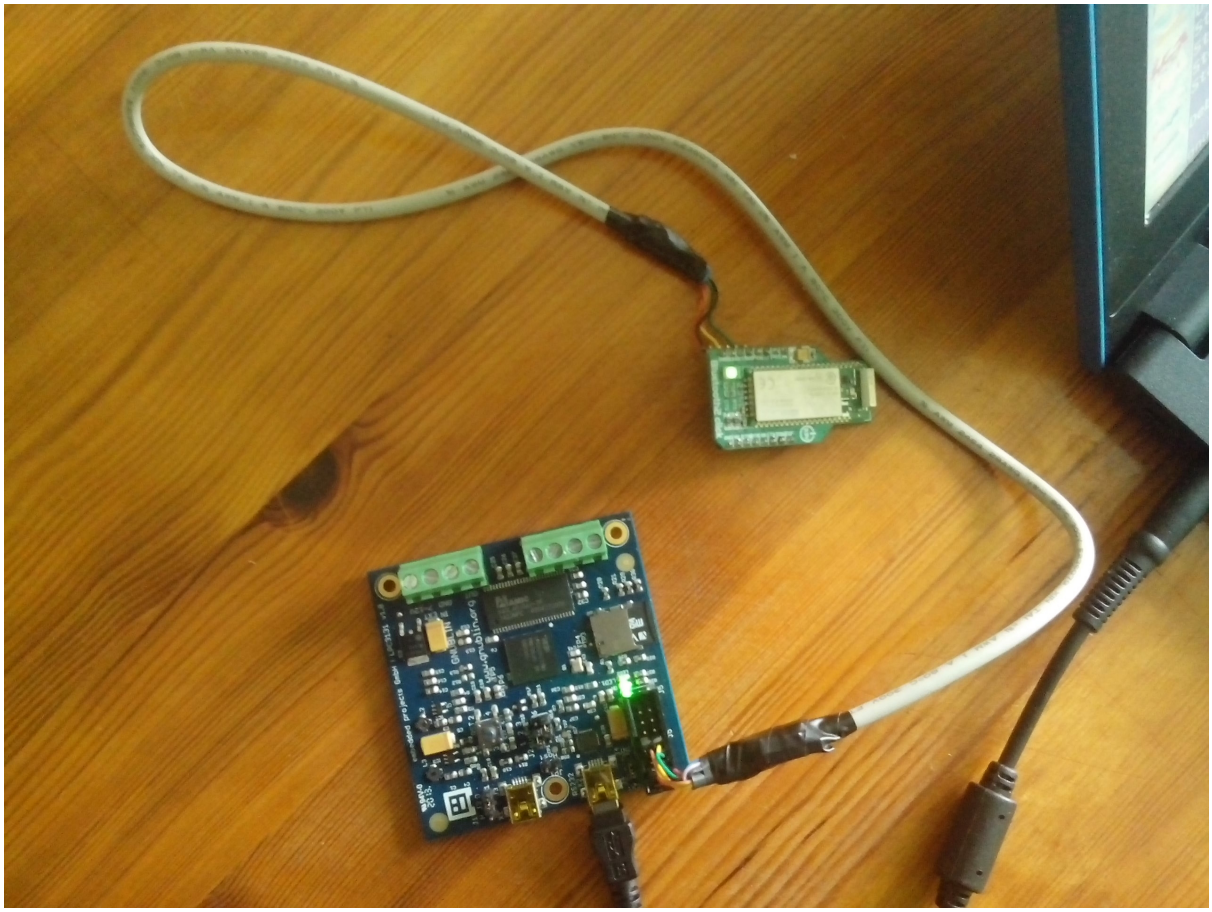
Слика 8.2.1 Bluetooth модул

8.2.2 Начин повезивања

Bluetooth модул се повезује са плочом преко GNUBLIN конектора. За комуникацију се користи **SPI** (**S**erial **P**heripherial **I**nterface) магистрала. Водови који учествују у комуникацији су напојни водови (GND, VCC(3.3V)), SCK, MISO, MOSI.

Распоред водова:

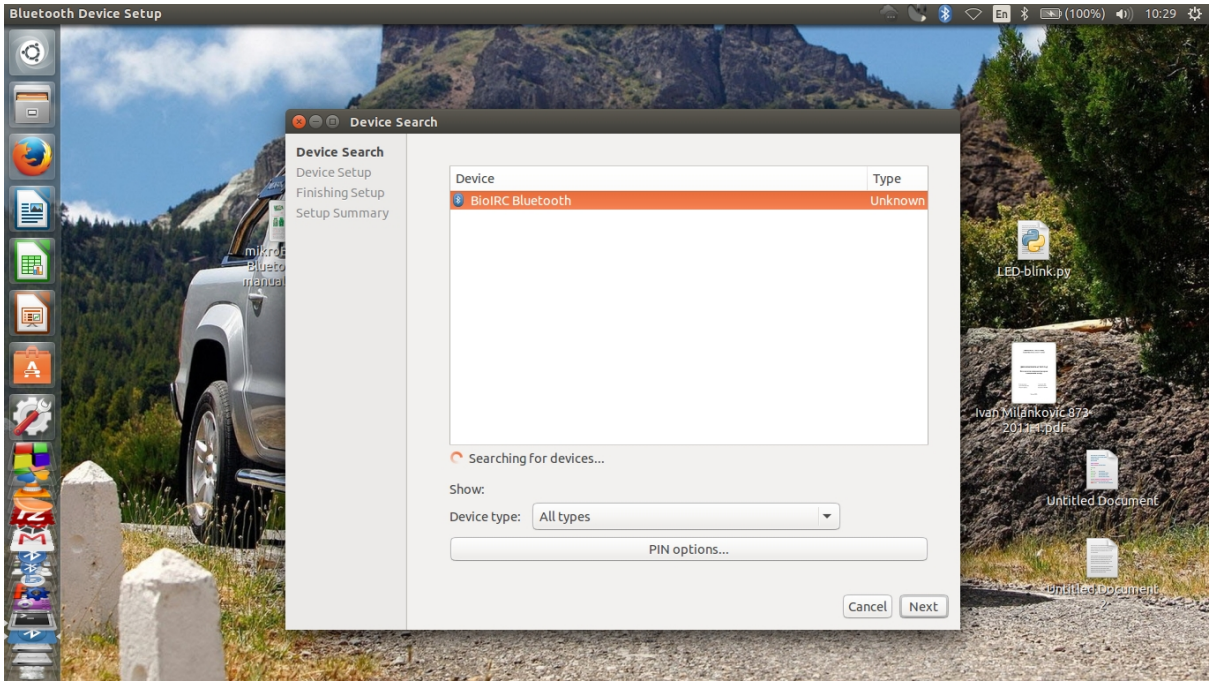
GNUBLIN		**Bluetooth**
GND	→	GND
3.3V	→	3.3V
SCK	→	SCK
MOSI	→	SDI
MISO	→	SDO



Слика 8.2.2 Правилно повезан модул са GNUBLIN плочом

8.2.3 Пријављивање модула

Након завршеног повезивања, потребно је да се уређај пријави на лап-топ рачунар или мобилни телефон (у овом случају уређај ће се пријавити на лап-топ рачунар). Следећа слика показује пријаву модула на систем.



Слика 8.2.3 Пријава Bluetooth модула на систем лаптоп рачунара

9. ЗАКЉУЧАК

Циљ овог рада је приказ основних концепата **Embedded Linux** система и практична реализација пројеката на једном од система. Овакви системи користе дистрибуције оперативних система заснованих на **Linux** језгру (кернелу). Комплетна филозофија се своди на принцип познат као „Open Source“ (отворени код). То значи да програмери или чак почетници који желе да се бave овим системима, могу лако креирати мале, а потом и сложеније апликације и то користећи различите програмске језике (C, C++, Python, bash, Lua, Perl, Ruby).

Затим је приказана архитектура једног таквог система, који се назива **GNUBLIN. GNUBLIN** може да се користи за имплементацију великог броја експеримената, који у најширем смислу имају везе са *Sensing* и *Control* апликацијама. Преко ADC, SPI, I2C интерфејса, може се додати велики број сензора, на пример, за температуру, влажност, притисак, интензитет светлости, убрзања и много других, а могуће је управљање LED, алфанумеричким дисплејем, серво-моторима и и најразличитијим извршним елементима. Принцип који стоји у позадини је „додај периферије по потреби“, тако да сваки експеримент захтева додавање додатних модула, који су заиста потребни.

Озбиљним приступом овој архитектури, могу се пројектовати и израдити комплексни аутономни системи који би извршавали низ сложених задатака.

10. ЛИТЕРАТУРА

[1] Karim Yaghmour, ***Building Embedded Linux Systems***, O'Reilly & Associates, Inc., United States of America, 2003.

[2] Hubert Hoegl, ***Teaching Embedded Linux with Gnublin***, Hochschule Augsburg, Fakultat fur Informatik, January 13, 2013.

[3] Милан Мијалковић, ***Предавање о неким стандардима комуникације***, Висока школа електротехнике и рачунарства струковних студија, Београд, 2008.

[4] Борислав Ђорђевић, Драган Плескоњић, Немања Мачек, ***Оперативни систем Linux,*** Виша електротехничка школа, Београд, 2004.

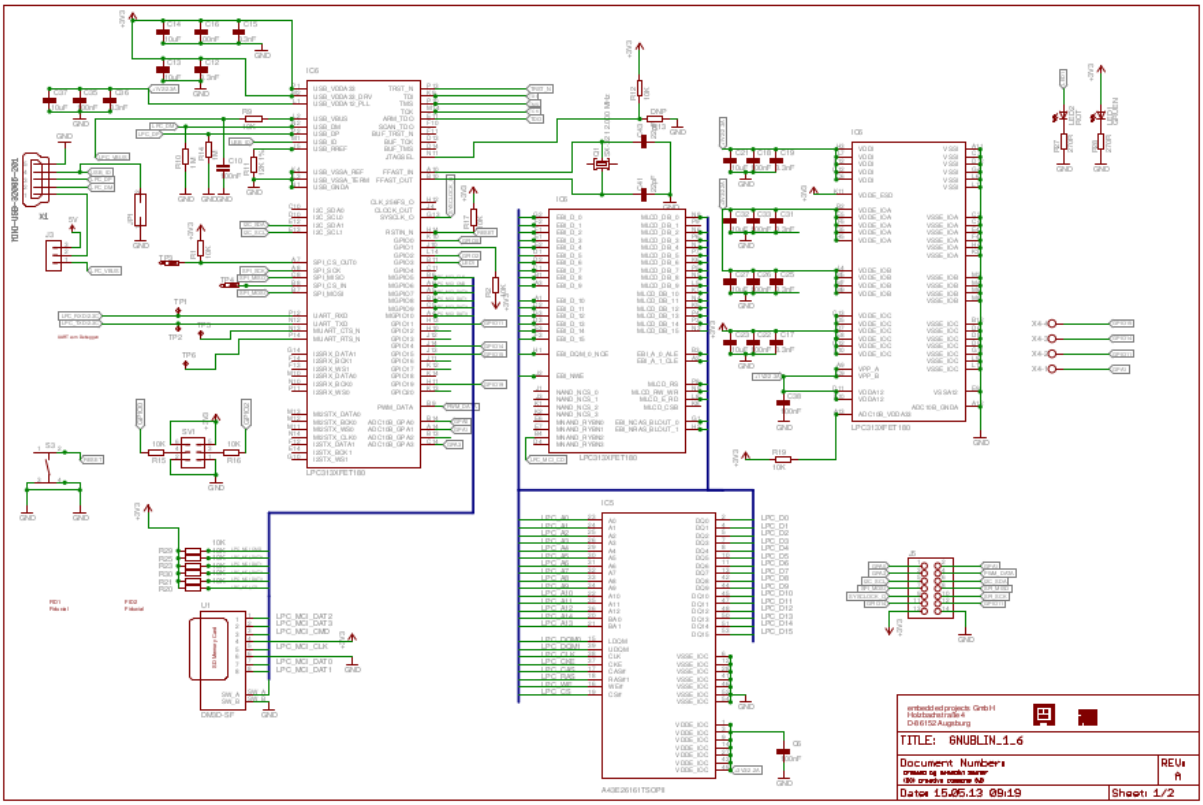
[5] WEB:

http://012lab.com/raspberry-pi-model-b
http://www.cnx-software.com/2012/06/26/list-of-39-low-cost-linux-friendly-boards-and-products/
http://gnublin.embedded-projects.net/all/
http://www.mikroe.com/click/bluetooth2/

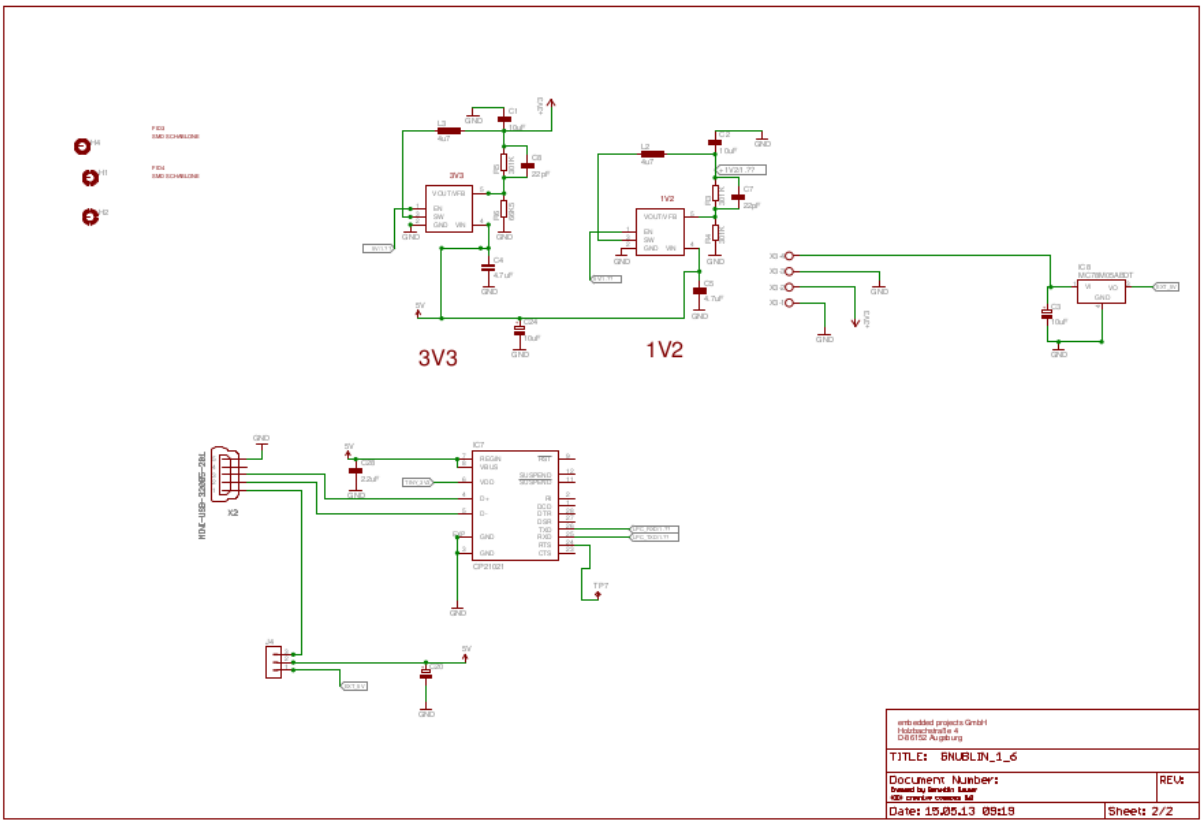
11. ПРИЛОГ

11.1 Прилог А

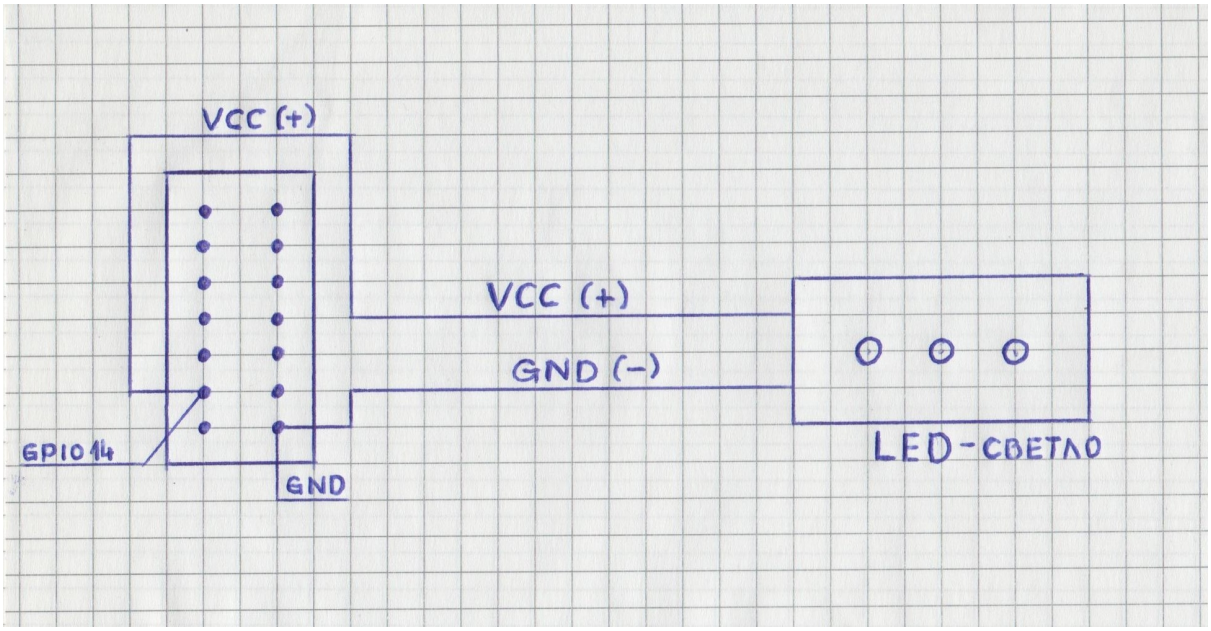
У прилогу **А** дате су шеме GNUBLIN-STANDARD плоче као и шема повезивања LED светла.



Слика 11.1а Први део шеме



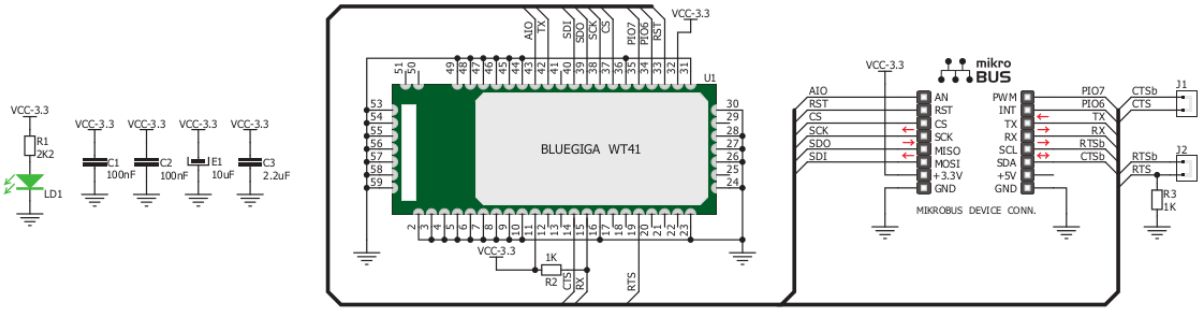
Слика 11.1б Други део шеме



Слика 11.1в Шема повезивања LED светла са GNUBLIN конектором

11.2 Прилог Б

У прилогу **Б** дата је шема *Bluetooth* модула и његове комуникације путем SPI и UART интерфејса.



Слика 11.2 Шема Bluetooth модула и његове комуникације