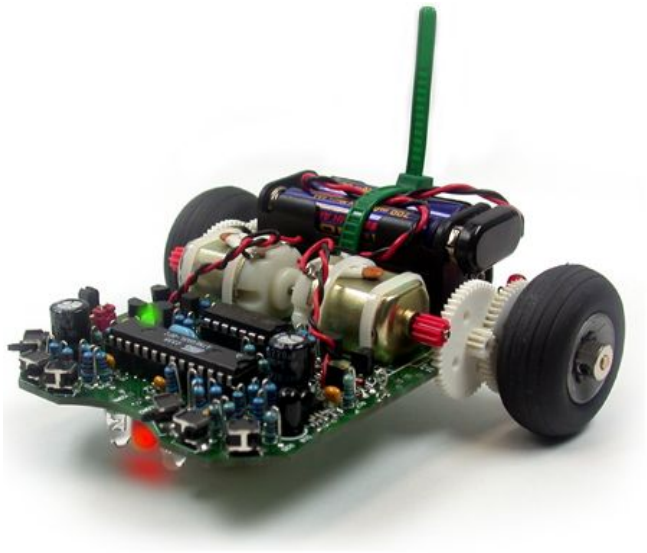
MAŠINSKI FAKULTET U KRAGUJEVCU

СЕМИНАРСКИ РАД

ПРОГРАМИРАЊЕ И УПОТРЕБА ASURO РОБОТА



ПРЕДМЕТ:

РАЧУНАРСКИ ПОДРЖАНО МЕРЕЊЕ И УПРАВЉАЊЕ

СТУДЕНТИ:
Стефић Милош 47/2006
Пушкарић Хрвоје 56/2006
Голубовић Урош 104/2007

PROFESOR:
Пр. Др. Милан Матијевић

KRAGUJEVAC 2010.

1.0 УВОД

Овај рад је настао на основама семинарских радова о ASURO мобилном едукативном роботу који је изучаван на предмету *Рачунарски подржано мерење и управљање.*

Тема овог рада тиче се аутоматског управљања над школско-едукативним роботом ASURO о коме ће бити више речи касније.

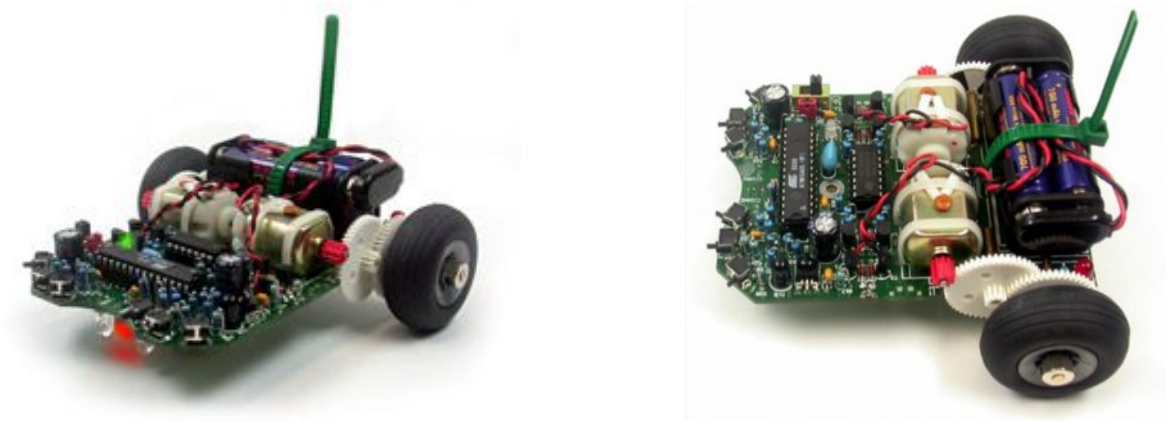
Како су ASURO роботи пристигли у кабинет за мехатронику са документацијом на немачком, овај рад представља компилацију превода делова упутства које долази са ASURO роботом, превода неке од литература коју смо успели да пронађемо на интернету, искустава других истраживача на ову тему и на крају, нашег резултата самосталног експерименталног рада на програмирању и репрограмирању програмског кода за ASURO, те и личних запажања.

Сматрамо да као такав може представљати корисну базу за нова истраживања на ову тему и са овом врстом робота за неке нове и млађе колегинице и колеге.

2.0 ASURO ХАРДВЕР

2.1 ASURO – едукативни мобилни робот

ASURO је мали мобилни робот развијен у едукативне сврхе од стране **DLR** (**D**eutsches Zentrum für **L**uft- und **R**aumfahrt), тј. немачког аеро-наутичког центра, својеврсног еквивалента америчкој NASA-и који се налази у немачком граду Oberpfaffenhofen.



Слика 1. **-** ASURO едукативни мобилни робот

ASURO је заправо скраћеница од:

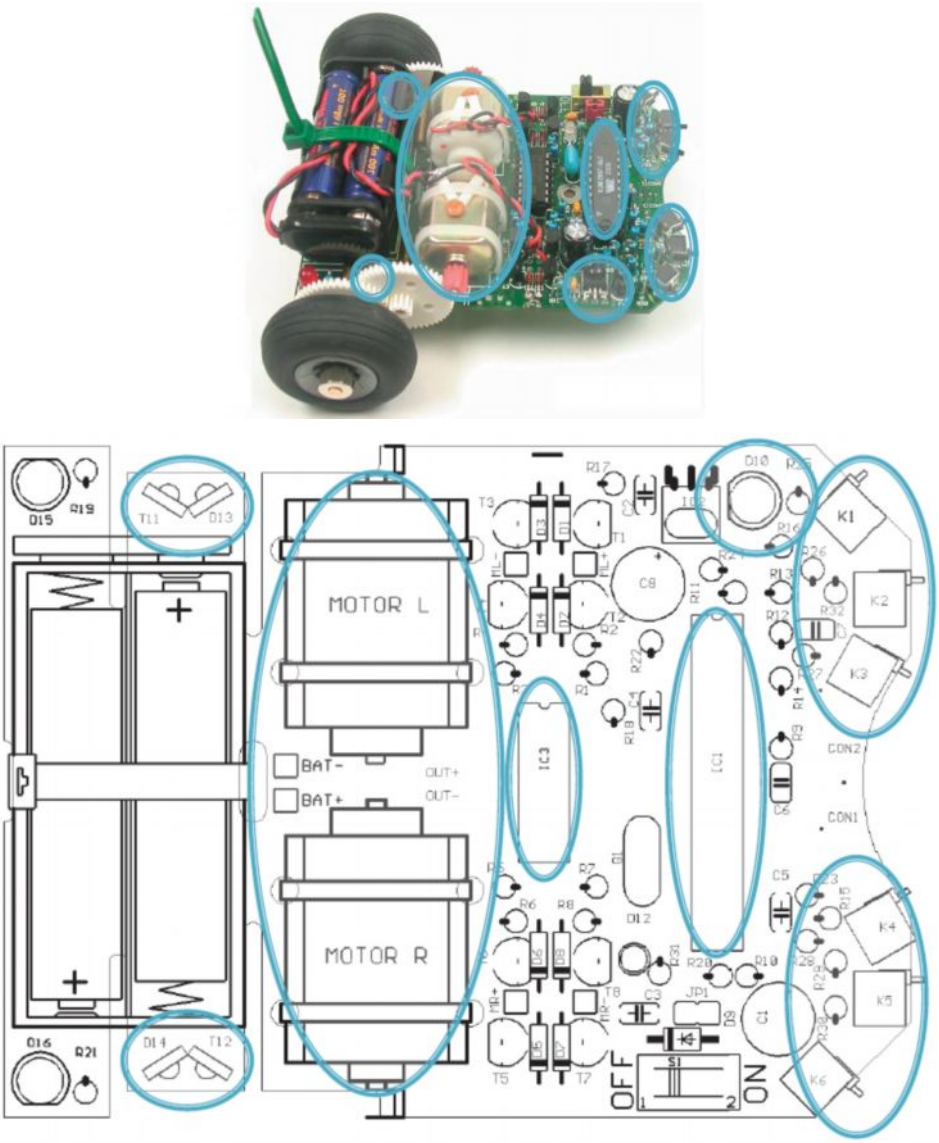
*'**A**nother **S**mall and **U**nique **R**obot from **O**berpfaffenhofen'.*

Флексибилан је и комплетно програмабилан у програмском језику C. Монтажа је лака за искусне електронце и флексибилна за нове кориснике.Осим штампане електронске плоче (PCB) коришћени су стандардни делови, док је од софтвера коришћен бесплатно доступан програмски компајлер за C.

2.2 ASURO архитектура система

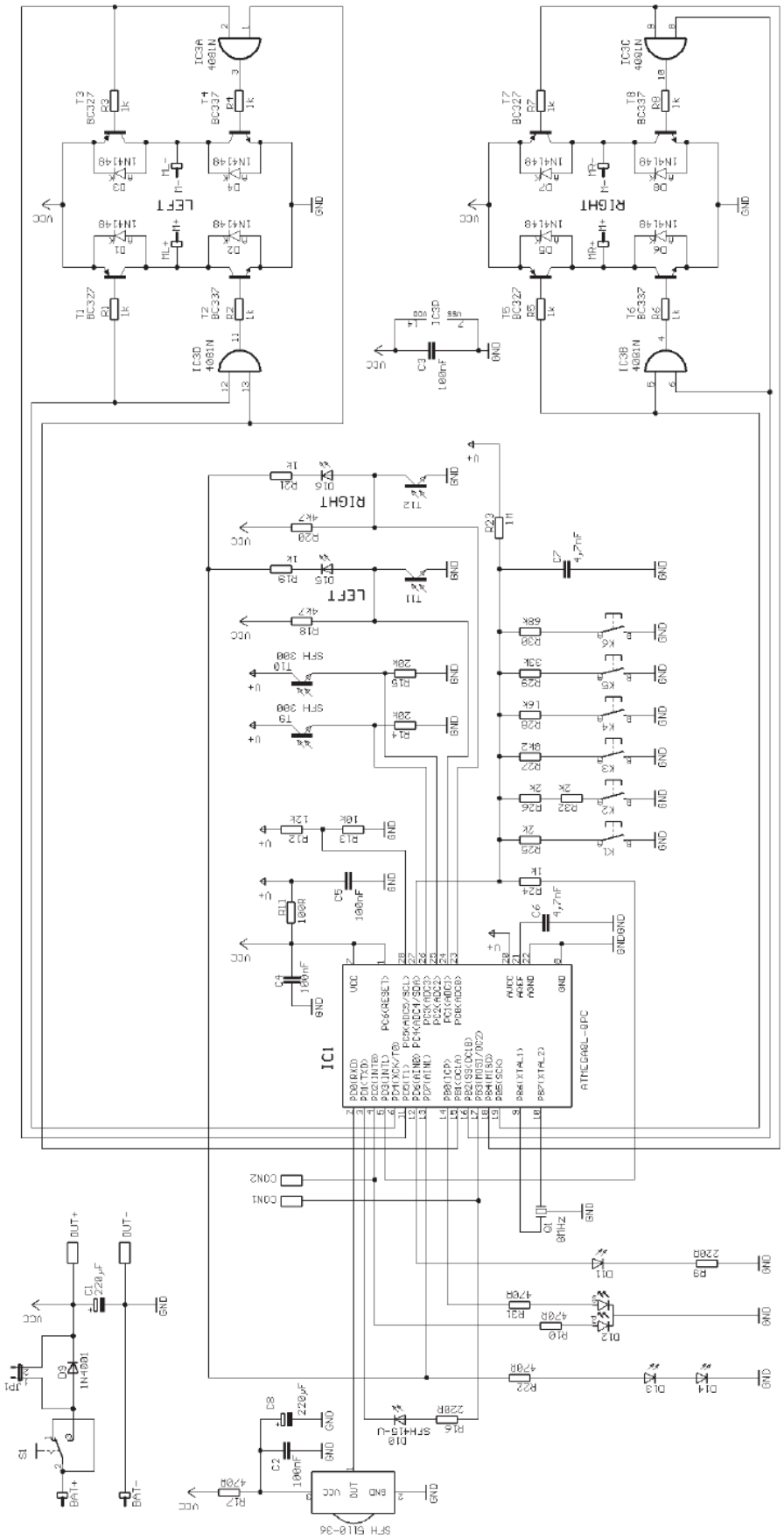
Сам поступак склапања робота уз праћење добијеног упуства је једноставан и биће укратко описан касније, након упознавања са архитектуром система. Поред штамапане плоче и већ поменутог софтвера, саставни делови ASURO мини-робота су :

- IC1 : Atmel AVR RISC – процесор,
- MOTOP L, MOTOP R – два независно контролисана мотора,
- Ooptical linetracer (сензор D12) – Оптички трагач линија,
- K1,K2….K6 – Шест независних сензора на притисак, у виду прекидача,
- T11, T12; T13, T14 – Одометарски сензори,
- D15, D16 – LED индикатори.
- IR – Интерфејс за програмирање и контролисање од стране PC-а.



Слика 2 – Основни делови ASURO робота

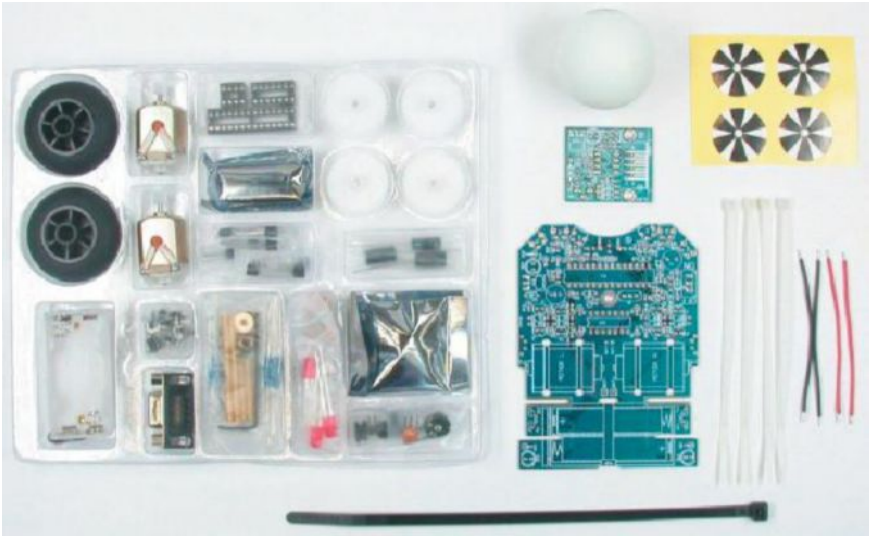
На слици 2 је приказан релни - физички и технички призак ASURO мини-робота.



Слика 3 – Електронска шема ASURO мини-робота

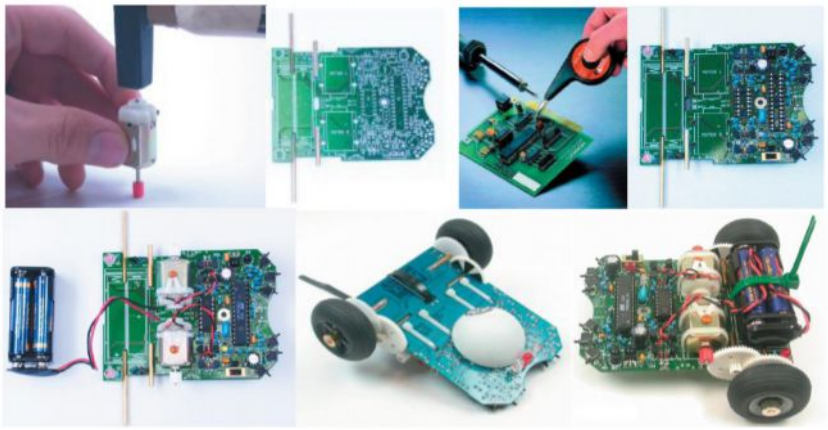
2.3 Склапање ASURO модела

Након упознавања са архитектуром система, време је да укратко објаснимо како се те инфромације требају употребити у пракси - склапањем објекта управљања нашег система!



Слика 4 **-** Саставни делови ASURO

Слику 4 можемо схватити као декомпозицију слике 2. Овде су такође приказани сви делови ASURO робота. Све што треба урадити јесте саставити их према шеми са слике 3. За то је коришћен стандардни алат: лемилица са пратећим прибором, шравцигер, лепак ... итд. Најпре се подешавају зупчаници на осовинама. Потом се леме елементи дигиталних кола на штампаној плочи према шеми из упуства. Након тога се постављају делови за напајање - кудиште за батерије, оно се повезује са електромоторима и штампаном плочом. На крају је потребно расећи пинг-понг лоптицу на пола и залепити је за штампану плочу. На тај начин смо обезбедили трећу тачку ослонца за ASURO и спреман је за тестирање.

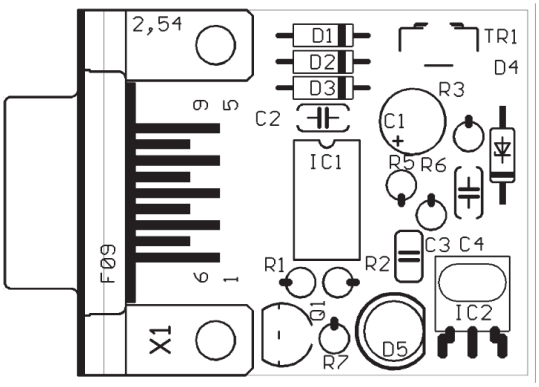


Слика 5 – Поступак склапања ASURO робота

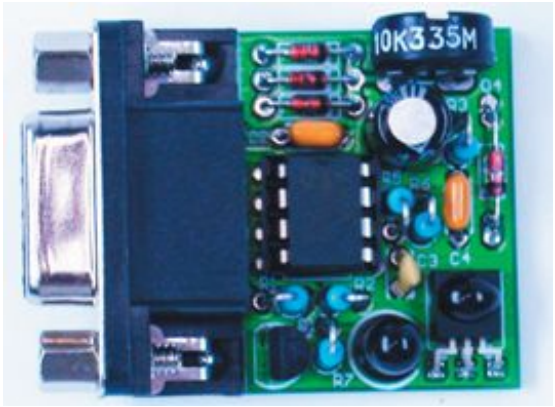
2.4 Комуникација АСУРО робота са рачунаром

2.4.1 IR примопредајник

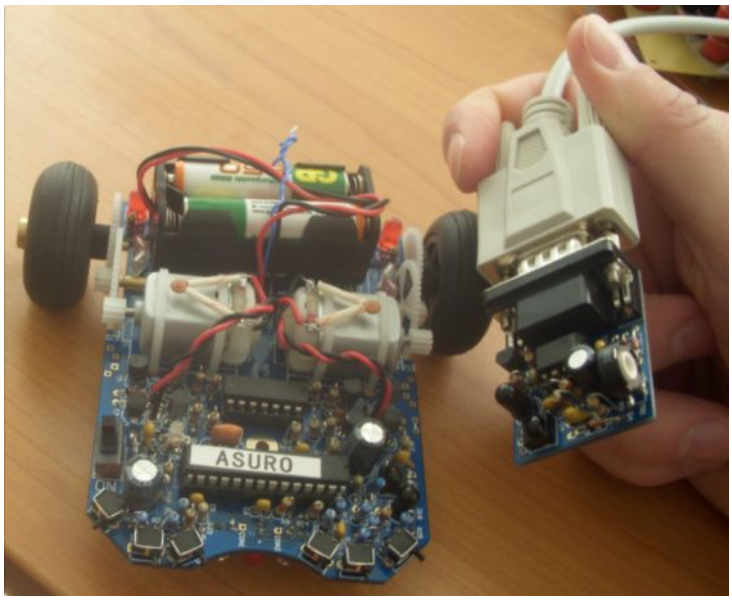
На слици 6 и 7 је приказан IR трансивер (примопредајник) помоћу кога се врши комуникација између рачунара и ASURO робота. Подаци се могу слати са рачунара на ASURO (нпр. слање програма на ASURO помоћу апликације (*flash-tool*) или се подаци могу слати у обрнутом смеру при чему се користи посреднички програм (нпр. *hyperterminal*).)



Слика 6 - Шема IR-примопердајника

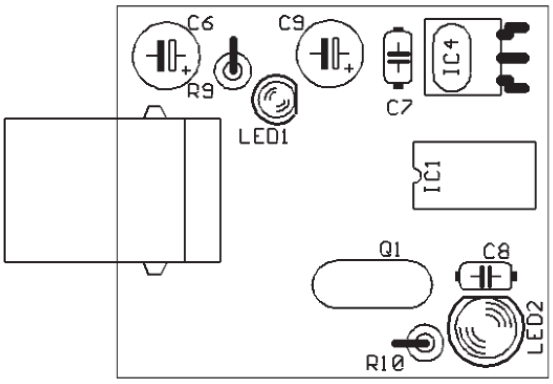


Слика 7 - IR-примопердајник

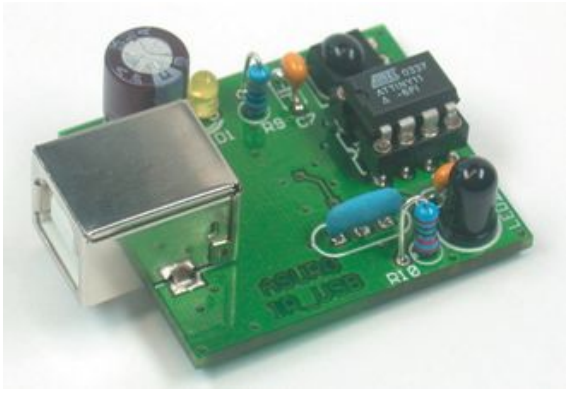


Слика 8 – Комуникација са рачунаром помоћу IR трансивера

2.4.2 USB IR-трансивер



Слика 9 - Шема USB IR-Примопердајника



Слика 10 - USB IR-Примопердајник

Треба напоменути да приликом пребацивања програма на ASURO може доћи до неких неправилности. Те неправилности ће бити пријављене следећим симболима:

“c” ➢ ASURO је примио неке неправилне сигнале. Сигнали могу бити ометени неким другим оптичким изворима или могу бити краткотрајно прекинути померањем IR трансивера.

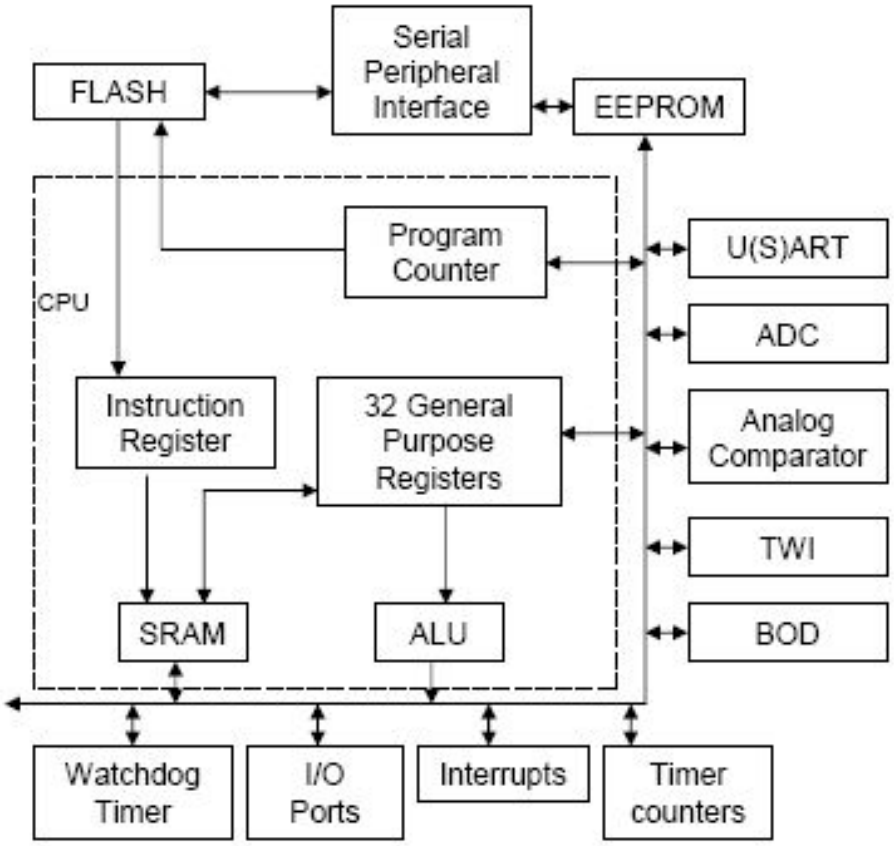
“t” ➢ Линија сигнала између ASURO робота и IR трансивера је комплетрно прекинута.

“v” ➢ ASURO је заменорисао неке неправилне податке у својој меморији. То значи да је програмска меморија дотрајала и то ће се десити након отприлике 10.000 циклуса пребацивања података.

3.0 СТРУКТУРА ASURO СИСТЕМА

3.1 Микроконтролер – ATMEGA8

Микроконтролер је тзв. „рачунар-у-чипу“ који се користи за контролу интегрисаних кола, склопова, уређаја итд. Типичан микроконтолрер садржи сву потребну меморију и I/O интерфејсе (A/D конверторе, серијске и паралелне комуникационе модуле, тајмере итд).



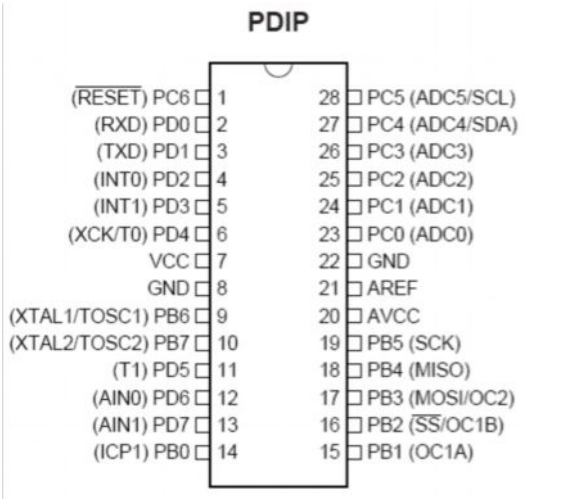
Слика 11 - Општа блок шема микроконтролера

ATmega8 је 8-битни CMOS микроконтролер базиран на AVR RISC архитектури. Брзина рада је 1 MIPS/MHz. AVR језгро извршава инструкције уз подршку 32 радна регистра. Свих 32 ргистра су директно повезани на ALU-аритметичку логицку јединицу - део CPU-а изведен у виду логичких кола која извршавају аритметичку или логичку операцију задату преко контролера CPU-а и уписује резултате у посебан регista-који се назива акумулатор.

Flash меморија се може репрограмирати преко серијског или USB порта, или преко on-chip програма поркренутог на AVR језгру док се подаци са пинова уписују у EEPROM и одатле се њима манипулише. Иначе, Atmega8 је опремљен са системским развојним алатима који укључују C компајлере, макро асемблере, програм debugger/simulator-е.

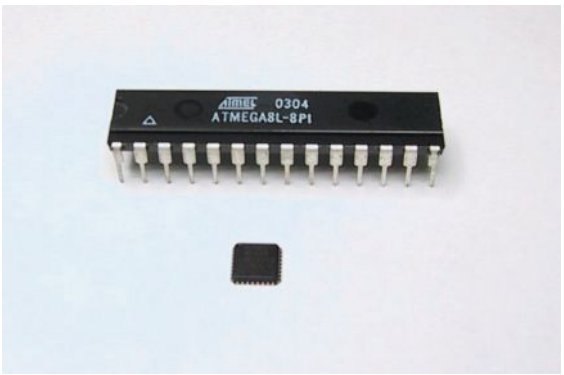
Особине ATMEGA8 :

- Поседује 8Kb In-System RW Programmable Flash меморије,
- 512b EEPROM,
- 1Kb SRAM,
- 23 I/O линија,
- 32 радна регистра,
- 3 флексибилна Timer/Counter бројача.



Слика 12 - Распоред портова на ATMEGA8

Микроконтролери се данас користе у embedded системима за управљање разним процесима. Они се веома често користе у ситуацијама када треба да се веома брзо реагује на спољне сигнале - посебно ако се такви системи користе за рад у реалном времену.



Слика 13 - Највећи и најмањи ATMEGA8 чип.

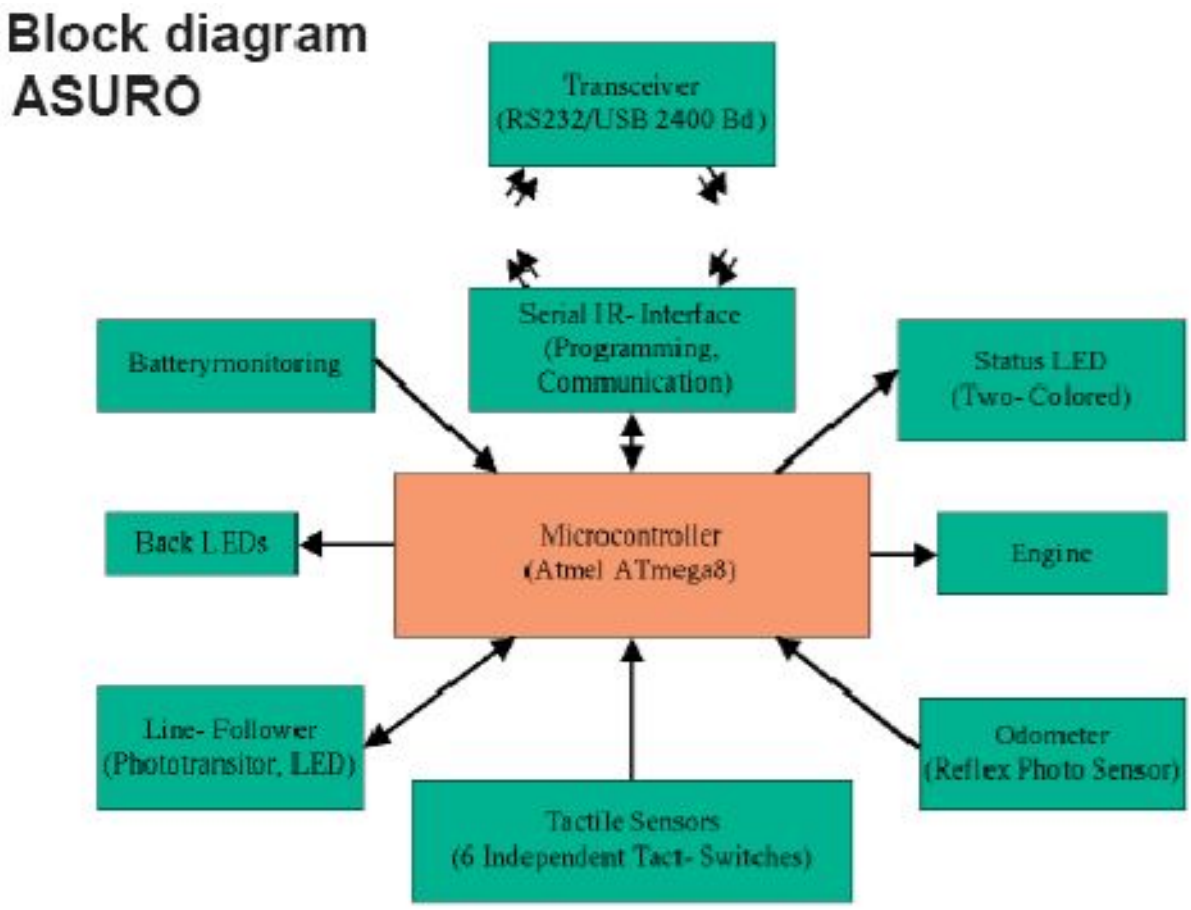
3.1.1 Улога микроконтролера у ASURO мини-роботу

Главна управљачка јединица

Најједноставније речено, микроконтролер ATMEGA8 има задатак да одржава комуникацију са свим уређајима који су повезани на њега, да им задаје задатке, прати њихов рад и регује тако да се систем налази у предвиђеном радном стању-опсегу.

На улазне портове микроконтролера повезани су сензор за топлоту, тастери на притисак и одометарски сензори (њих у аутоматском управљању називамо улазним величинама система управљања), на излазне портове микроконтролера повезани су електромотори (излазне - управљане величине).

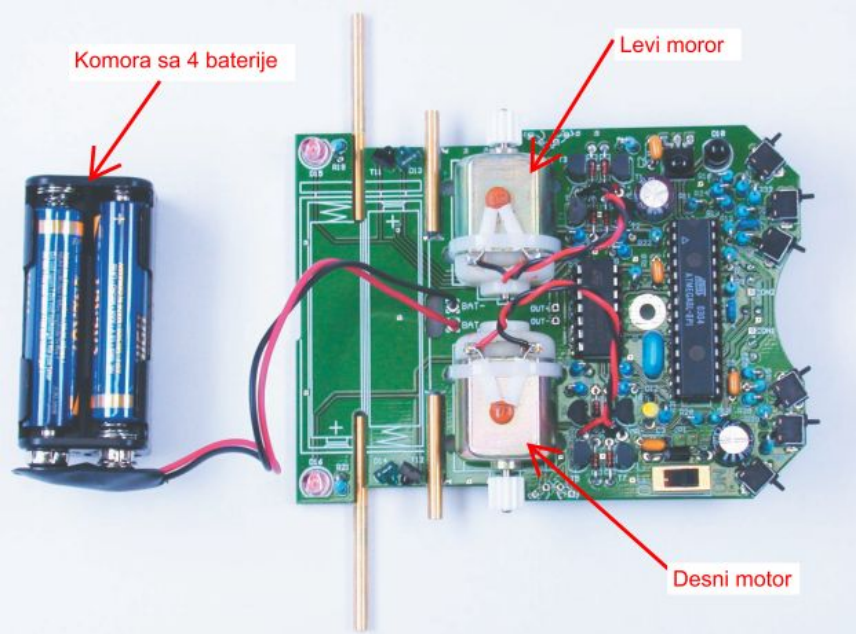
Микроконтролер са корисником комуницира преко RS232 или USB порта преко којих се може репрограмирати или слати резултате рада на корисников PC.



Слика 14 **-** Блок шема ASURO-а

3.2 Покретачка јединица

Покретачку јединицу ASURO робота чине 2 мотора којима се независно може управљати. Мотори се електричном енергијом могу снабдевати помоћу 4 батерије које се смештају у комору за батерије.



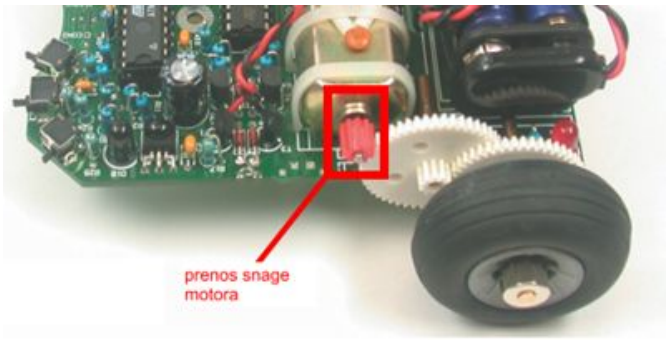
Слика 15 - Покретачка јединица ASURO робота

Као што се види са слике 15 електрична енергија се из коморе за батерије преноси помоћу црне и црвене жице, при чему је црна жица залемљена за прикључак BAT- , а црвена за прикључак BAT + на табли.

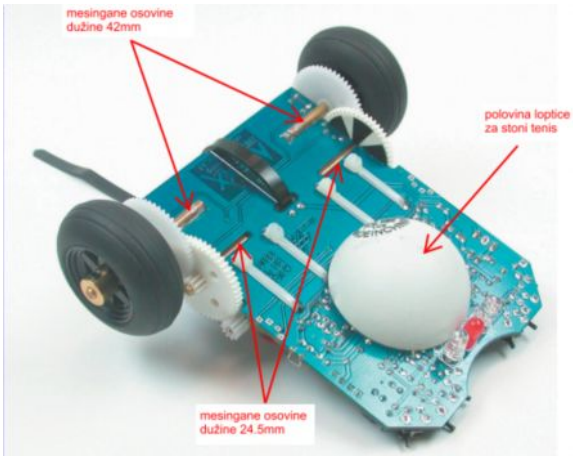
Црвена жица је везана за прикључак на мотору са знаком +, а црна за прикључак на мотору са знаком -. Црвена жица левог мотора је залемљена за прикључак ML +, а црна жица левог мотар за прикључак ML -. Црвена жица десног мотора је залемљена за прикључак MR+, а црна жица десног мотар за прикључак MR-.

Снага мотора се преноси на зупчанике преко малог зупчаника који је постављен на осовину мотора као што је то приказано на слици 16.

ASURO робот је дизајниран да клизи помоћу половине лоптице за стони тенис. Модел садржи 2 пара осовина које су израђене од месинга и које су залемљене или залепљене за плочу.



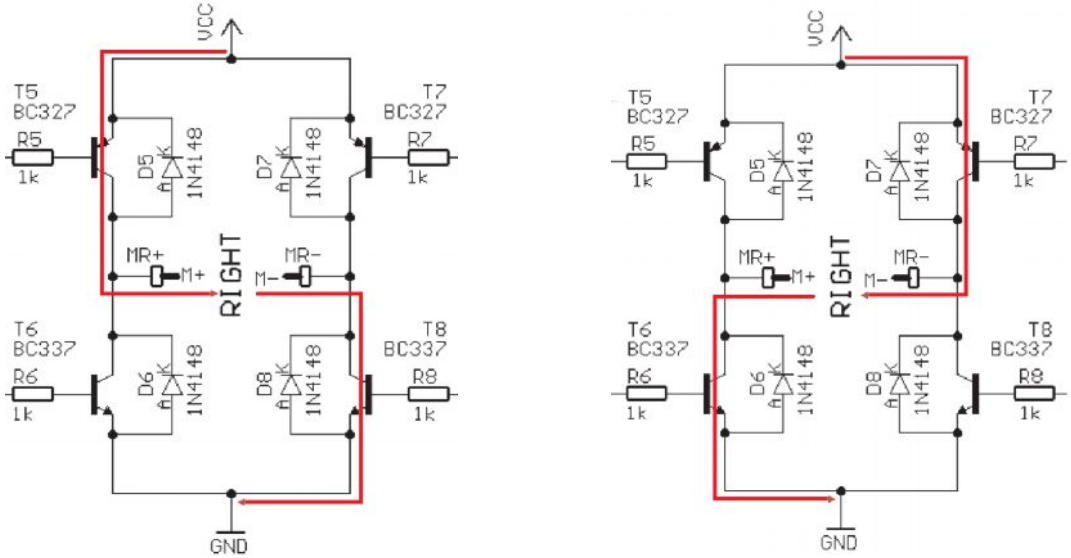
Слика 16. - Пренос снаге мотора



Слика 17. - ASURO робот са доње стране

Као што смо објаснили, снага електромотора се преноси са зупчаника који су причвршћених на осовину електромотора на зупчанике који се налазе на заједничкој осовини (пречника 1.9 mm) и потом на зупчанике који се налазе на истој осовини на којој и точкови.

Пошто се ради са моторима које покреће једносмерна струја, промена смера обртања мотора (односно окретања точкова и кретања ASURO робота) остварује се помоћу H-моста. H-мост је повезан на излазне портове микроконтролера преко система логичких кола одакле добија одговарајуде логичке вредности (нула - низак напон, јединица - висок напон). Логичка кола дају такве напоне да ће за логичку јединицу на одговарајућем излазном порту спроводити пар транзистора T5-T8, а за логичку нулу T7-T6 (за десни електромотор).

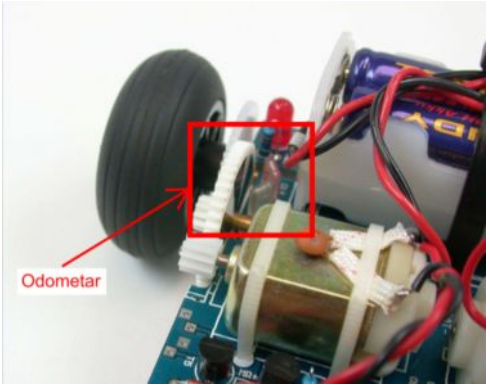


Слика 18 - H-мост, протицање ел. струје кроз H-мост.

3.3 Систем за регулисање брзине

Као што је већ напоменуто ASURO робот садржи 2 сензора пређеног пута (*odomer-sensors*). Одометар, који се састоји од диоде која емитује светлосни сигнал и фототранзистора, усмерен је према црним и белим маркерима на зупчаник који је приказан на слици 20. Бели маркери рефлектују више светлости него црни и на тај начин узрокују виши сигнал на сензору (фототранзистору).

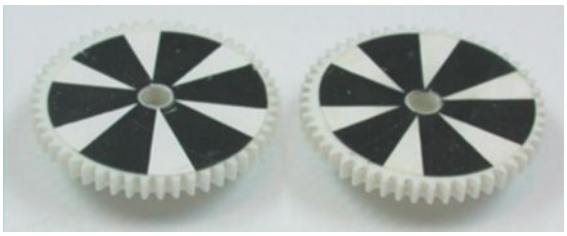
Слика 19 - Електромотор



Слика 20 - Одометар

Дакле на слици 21 је приказан зупчаник са 6 црних и 6 белих маркера од којих се рефлектује светлост коју фото диода емитује и потом фототранзистор региструје. Шта више сегмената постоји то ће бити боља регистрација брзине зупчаника, а самим тим и боља регулација брзине. Али са друге стране велики број сегмената смањује разлику између тамног и светлог сегмента што може да доведе до повећања грешке.

Фототранзистор је транзистор чија се колекторска струја мења под утицајем светлости која пада на њега. Фототранзистори реагују на интензитет светла, генеришући пропорционалну струју. Због те особине налазе примену као електронски сензори. На ASURO роботу котишћен је фототранзистор LPT80A и фото диода IRL80A који су приказани на слици 22.



Слика 21 - Маркери на зупчаницима

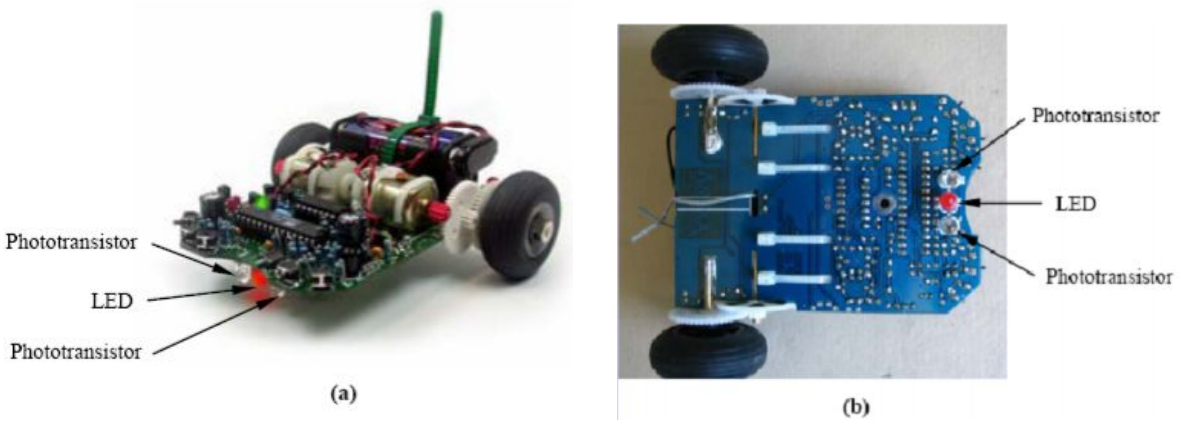
Слика 22 - Фотосензор LPT80A

3.4 Систем за праћење линије

Систем за праћење линије састоји се од LED диоде и два фототранзистора, који су приказани на слици 23. Систем ради тако што се црвена светлост LED диоде рефлектује са површине испод робота и бива примљена од стране фототранзистора.

Фототранзистор је уређај који спроводи одређену количину струје која је пропрционална примљеном интезитету светлост: већи интезитет примљене светлости значи да ће већа количина струје бити спроведена кроз транзистор.

Од проведене струје у транзисторима добија се даље напон. Напон фототранзистора се може мерити помоћу микроконтролера који су под надзором одговарајућег програма.



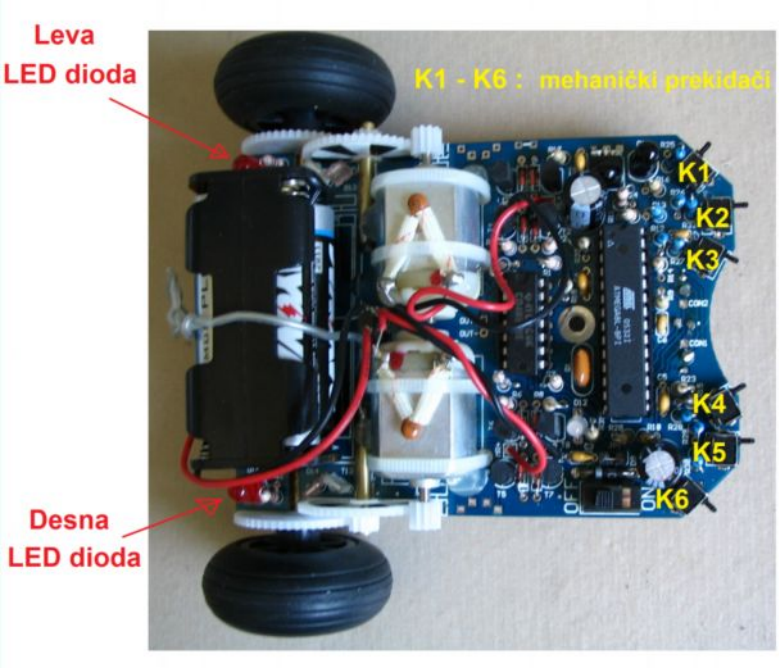
Слика 23 - ЛЕД диода и два фототранзистора

Овај систем за праћење линије искористили смо код симулације ASURO чистача и код ASURO сумо борбе робота, да ASURO робот не изађе из беле површине која је ограничена црном линијом. (нпр. сумо ринг).

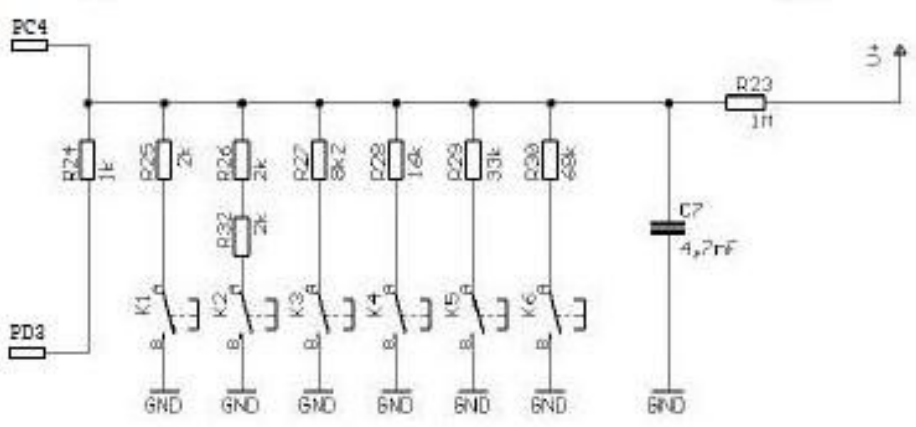
3.5 Механички прекидачи - тастери K1-K6

Механички прекидачи K1-K6 су приказани на слици. Ови прекидачи враћају одговарајућу вредност бита која је додељена сваком сензору.

Bit0 (1) -> K6
Bit1 (2) -> K5
Bit2 (4) -> K4
Bit3 (8) -> K3
Bit4 (16) -> K2
Bit5 (32) -> K1

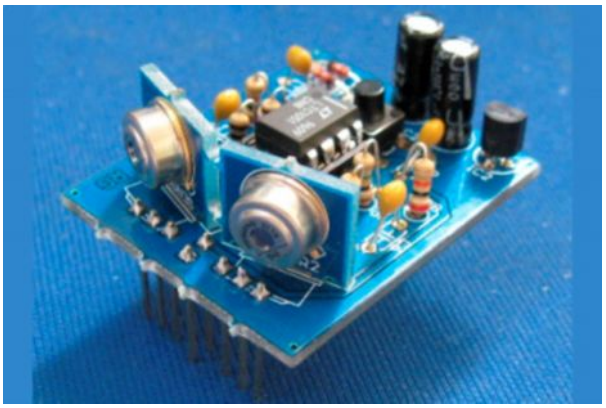


Слика 24 – Механички прекидачи K1 – K6

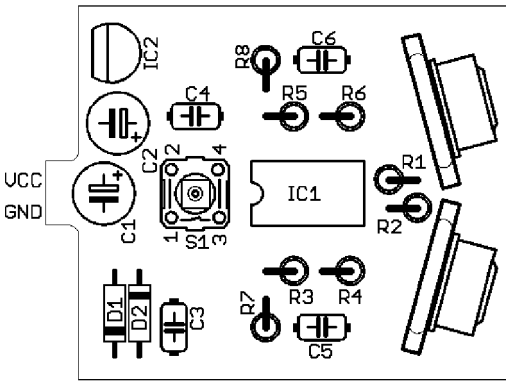


Слика 25 – шема тастера

3.5 Сензор за топлоту – Snike vision



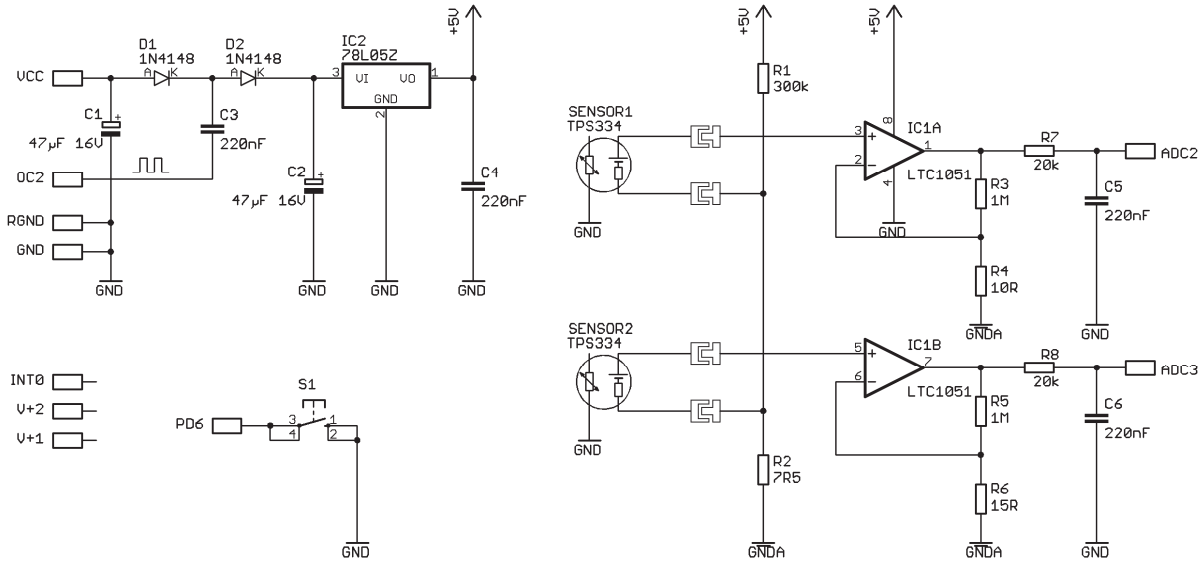
Слика 26 - Snake vision – сензор за топлпту



Слика 27 - Шема Snake vision-a

Тестирајући сензор, дошли смо до информација да је видно поље сензора око 60°, с'тим да највећа удаљеност на којој региструје топлоту износи око 20 cm у зависности од интензитет извора топлоте. Вредност интензитета топлоте коју сензор враћа у повратној спрези микроконтролеру је у K (келвинима).

$$C° = K - 273.15$$



Слика 28 – Електронска шема сензора за топлоту

Делови сензора за топлоту

- 220 nF керамички кондензатор 4 к 5,08 мм
- 2 к 16В ЕЛКО 47uFстоји 2,54 мм
- 1 к 11 ohm метал филм отпорник
- 2 х100 ohm метал филм отпорник
- 2 х метал филм отпорника од 20 KOhm
- 3 х метал филм отпорници 1 MOhm
- 1 х 78 L 2005 регулатор напона
- 1 х операциони појачавач LTC 1051 ЦН 8
- 2 х334 ТПС Пирометар
- 2 х 1 диоде Н 4148
- 1 х Воки-дугмад за PCB монтажу
- 2 х2 пин конектор за прикључак
- 2 х 3-пински конектор за прикључак
- 2 х 2 конектор
- 2 х 3-пински конектор

4.0 ИНСТАЛАЦИЈА СОФТВЕРА

Убаците ASURO CD. Када је све уреду стартоваће се аутоматски у супротном отворите га помоћу windows explorera. После избора језика наћићете све потрбне програме за ASURO у software менију. Пре него ли почнете рад са њима морате их инсталирате на ваш hard диск.

За време инсталације софтвера бице урадјено следеце:

1. Flash-Tool, програм за трансмисију вашег програма на ASURO, биће инсталиран.
2. Програмски едитор (Programmers Notepad 2, PN2) и компајлер (WinAVR) ће бити инсталирани.
3. Примери програма ча бити копирани са CD-а на ваш HD.
4. Биће креиран, у програмским едитору (PN2), мени за унос Make (срб. Направити) и Clean (срб. Очисти) фајлова.

4.1 Flash алати за WINDOWS

Копирајте flash програм у фолдер на вашем хард диску, на пример:
C:\programs\flash.

Такође је могуће покренути фласх програм директно са ЦД-а.
У сваком слуцају мудро је направити иконицу за flash програм на десктопу.

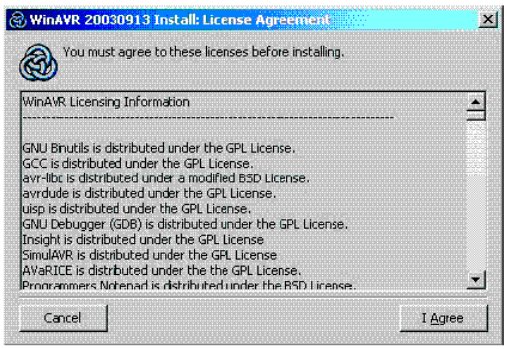
4.2 Инсталација програмског едитора и компајлера

За инсталацију кликните на инсталациони симобол

COMPILER WinAVR (20030913)

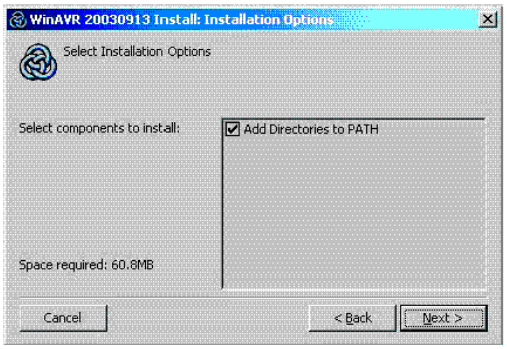
Ако се ништа не догоди отворите CD помоћу windows explorera.

Појавиће се овакав екран



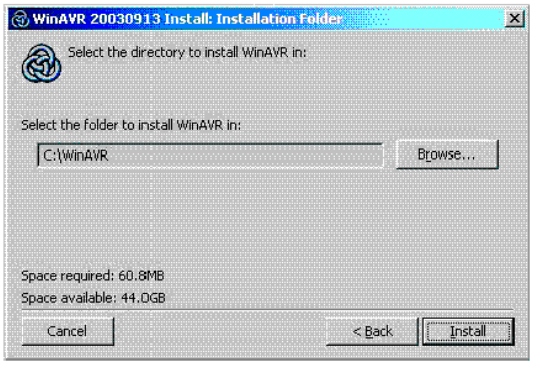
Кликните на [I Agree]

Појавиће се следећи екран :



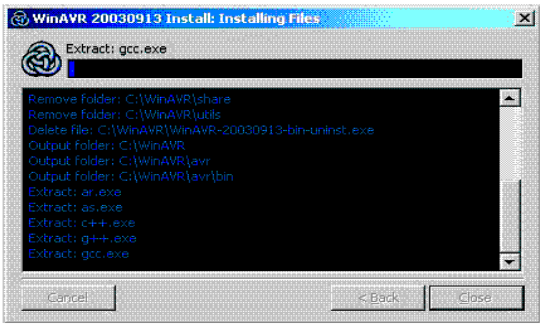
Кликните на [Next]

Појавиће се следећи екран

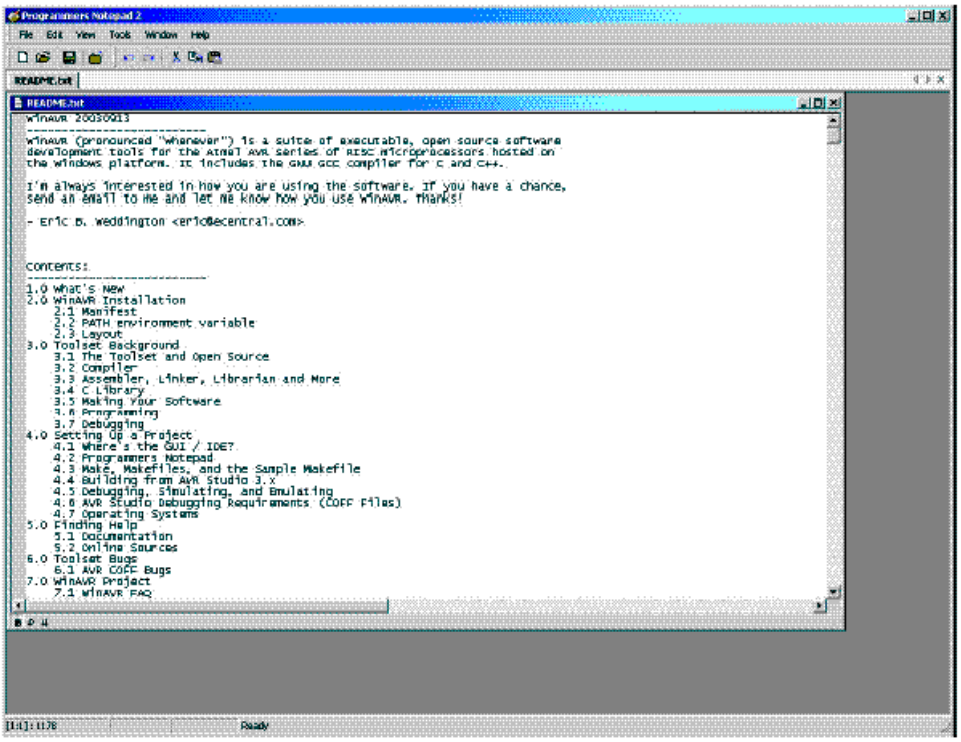


Кликните на [Install]

и појавиће се сл. екран:



Сачекајте док се програмски Notepad 2 (PN2) едитор не појави са README.txt фајлом.



Затворите прозор „програмског notepad2“

На DESKTOP-у ће се појавити иконица „програмског notepad2“

Програмски едитор и компајлер су сада инсталирани.

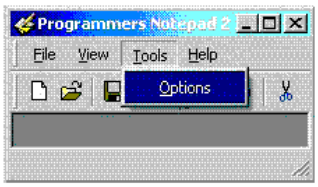
4.3 Копирање програмских примера са CD-а на Hard Disk (HD)

Треба ископирати фолдер ‘ASURO_src’ са ASURO CD-а на HD (сачувати га у фолдеру на пример: ‘C:\ASURO_scr’).

Када је изворни податак сигуран, кликните десним кликом миша на фајл, идите у property и деактивирајте сигурностну заштиту.

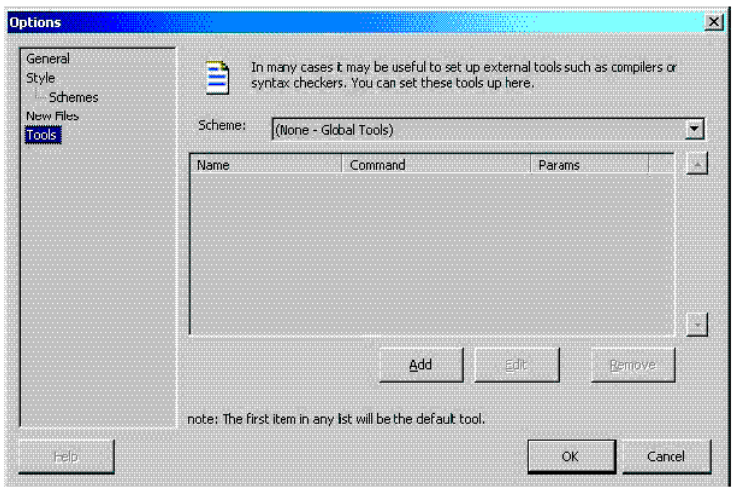
Треба подесити мени за унос у програмском едитору.

Отворити Notepad2:



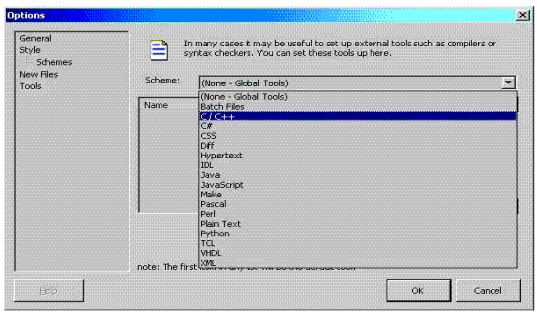
У менију Tools | изаберите options

Појавиће се прозор са опцијама

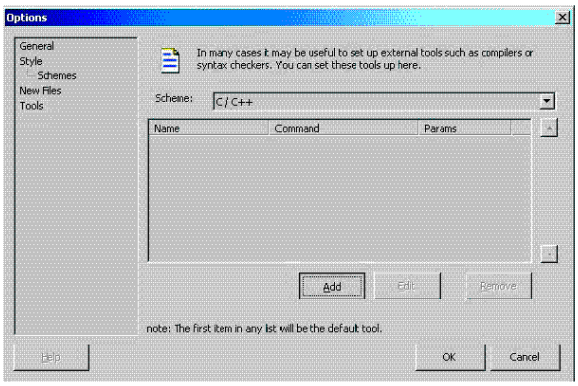


Изаберите Tools.

Изаберите са десне стране ‘C/C++’:



Изабран је ‘C/C++’.



Кликните на [Add] да би додали нови алат

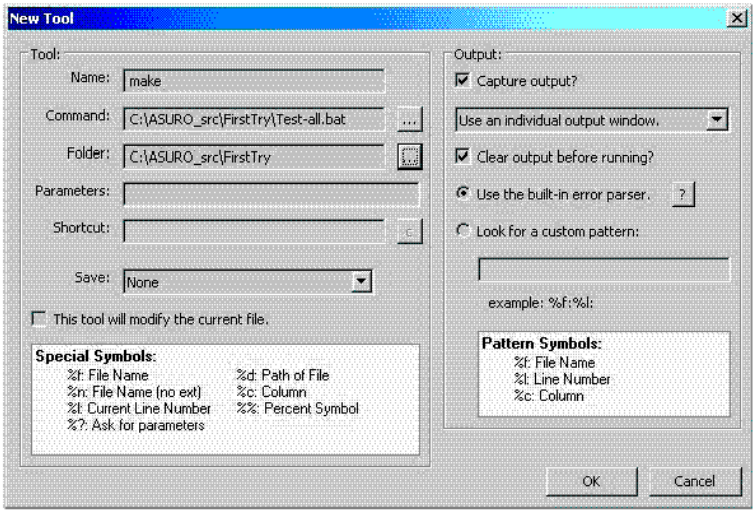
Прозор ‘New Tool’ ће се појавити:

Укуцајте следеће или кликните на Browse дугме  :

Name: make

Command:
C:\ASURO_src\FirstTry\Test-all.bat

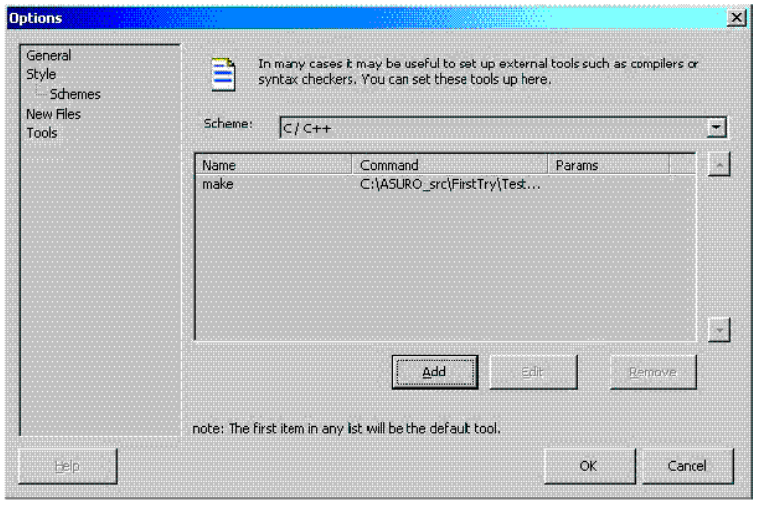
Folder :
C:\ASURO_src\FirstTry
Кликните на [OK]



Нови PN2 алат са именом “make” је сада доступан у tools главном менију.

Када активирамо овај алат, он ће покренути batch фајл са именом Test-all.bat, и онда ће компајлирани подаци бити смештени у фолдер: C:\ASURO_src\FirstTry.

У главном менијз “Tools” поново изаберите “Options” и поново селектујте “C/C++”



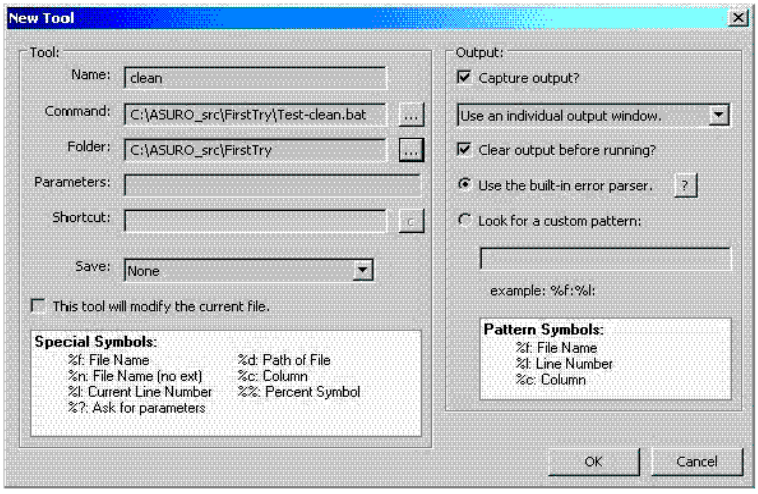
И кликните на [Add] да би додали нови алат,

Прозор ‘New Tool’ ће се појавити:
Укуцајте следеће или кликните на Browse- дугме  :

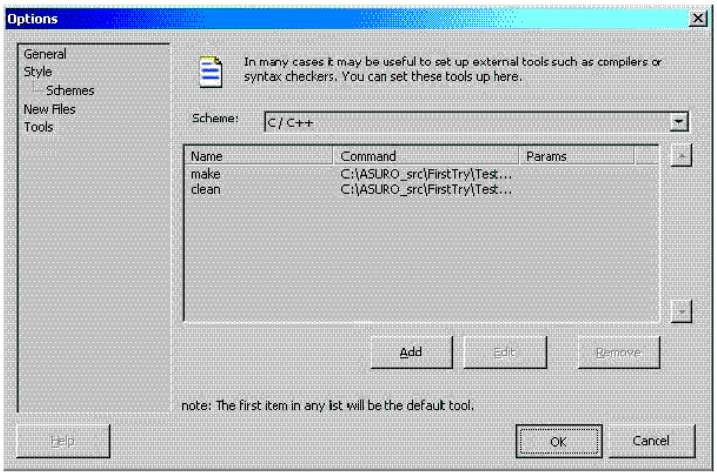
Name: clean

Command:
C:\ASURO_src\FirstTry\Test-clean.bat

Folder :
C:\ASURO_src\FirstTry
Kliknite na [OK]



Нови PN2 алат са именом “make” је сада доступан у tools главном менију. Када активирамо овај алат, он ће покренути batch фајл са именом Test-all.bat, и онда ће компајлирани подаци бити смештени у фолдер: C:\ASURO_src\FirstTry бити избрисани. У опционом екрану, у делу за алате (tools), сада се могу пронаци clean и make фајлови које смо претходно направили.

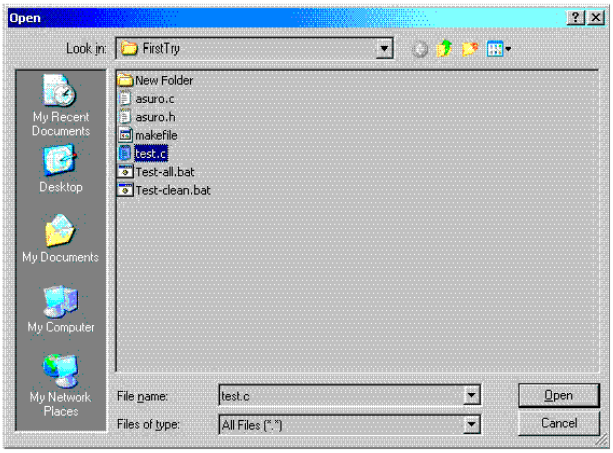


Кликните на [OK]

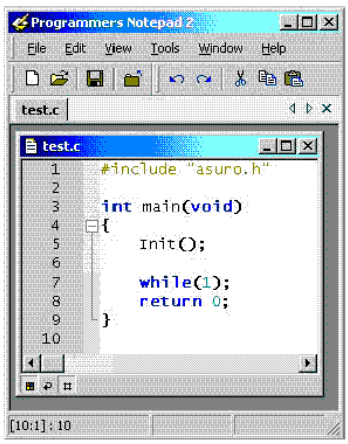
Пробе ради отворићемо фајл:

C:\ASURO_src\FirstTry\test.c’:

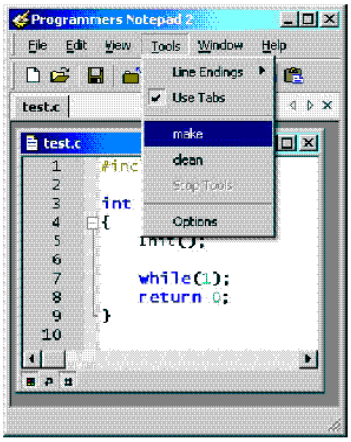
Кликните на [Open].



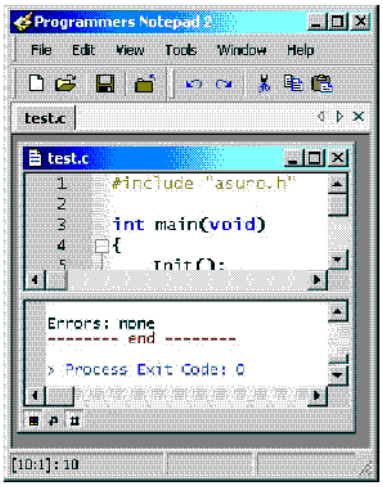
Отвориће се test.c



Кад се отвори одаберите tools... Отвара се падајући мени са два нова алата make и clean.



Потребно је кликнути на make.
Test.c фајл ће бити компајлиран



И кад програм не садржи грешке (што се може очекивати) на дну ће се појавити следећа порука: Errors: none.

Шта се догодило?

Из фајла test.c генерисан је фајл test.hex. Овај фајл садржи машински код конвертованог програма. Овај машински код може бити ископиран у меморију ASURO робота. Овај програм нема функцију. За аплоадовање користимо Flash алат.

Како ради?

Унешени фајл позива позадински фајл Test-all.bat (позадински фајл садржи листу са линијским командама које се извршавају линија за линијом).

У Test-all.bat команда ’make all’ ће бити извршена. ’make’ ће креирати извршни фајл који ће бити лоциран у истом фолдеру као и Test-all.bat.

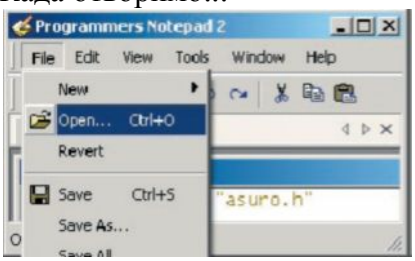
Итвршни фајл је текстуални фајл, који говори о томе како компајлирати један или више програма. За време програмирања, када имамо само један фајл, имамо добар преглед. Касније, када испишемо комплексан систем и када програмски подаци садрже доста фајлова, који морају бити конвертовани корак по корак и такође медјусобно повезани на одговарајући начин, тада ће и извршни фајл такође бити врло сложен. Позив ’all’ за све уносе у извршном фајлу значи, да ће комплетан пројекат а не само неки његов део бити конвертован.

Извршни фајл у нашем примеру програма је написан тако да ће се фајл са именом test.c компајлирати заједно са asuro.c (који садржи неке унапред дефинисане функције) фајлом и креирати нови .hex-fajl. Овај фајл може бити пребачен у меморију ASURO робота.

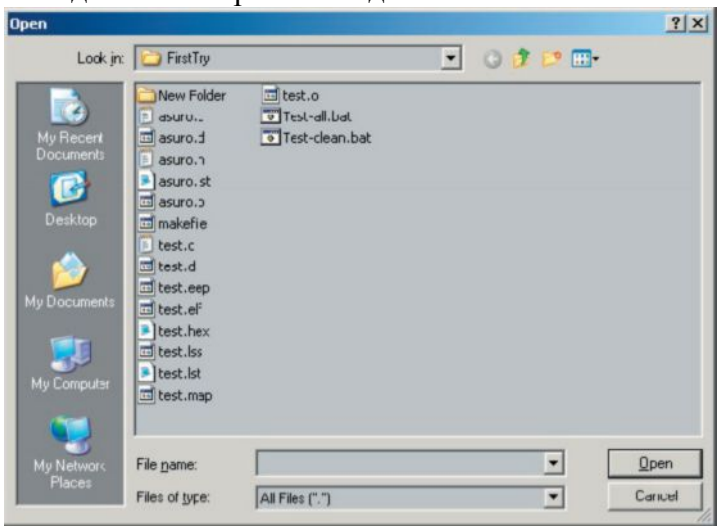
Пажња! Ово значи да - све док не мењате make фајл и само га ископирате - требало би увек вашем програму да дате име test.c.

Када се програм искомпајлира, долази до генерисања још неких података. Ови подаци су неопходни само за време превођења, после се могу избрисати. Ови сувишни фајлови се могу избрисати помоћу clean алата који смо управо креирали.

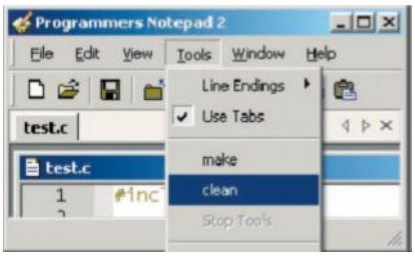
Када отворимо...



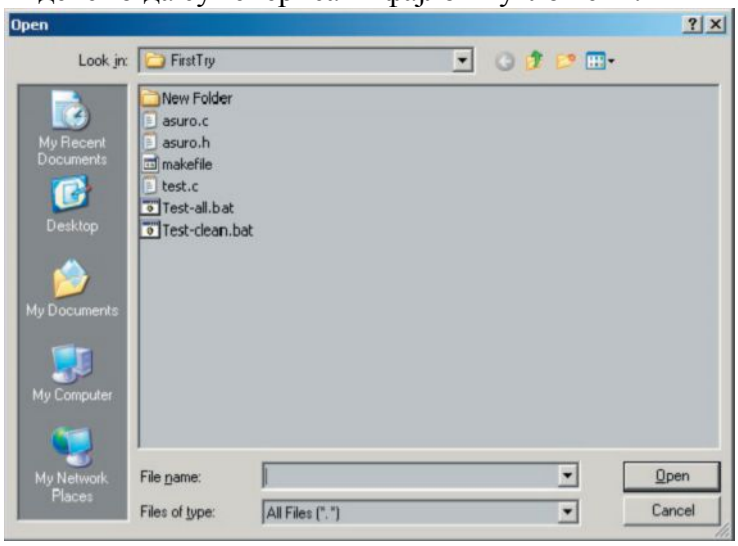
Видећемо генерисане податке...



И када покренемо команду clean...



Видећемо да су генерисани фајлови уклоњени.

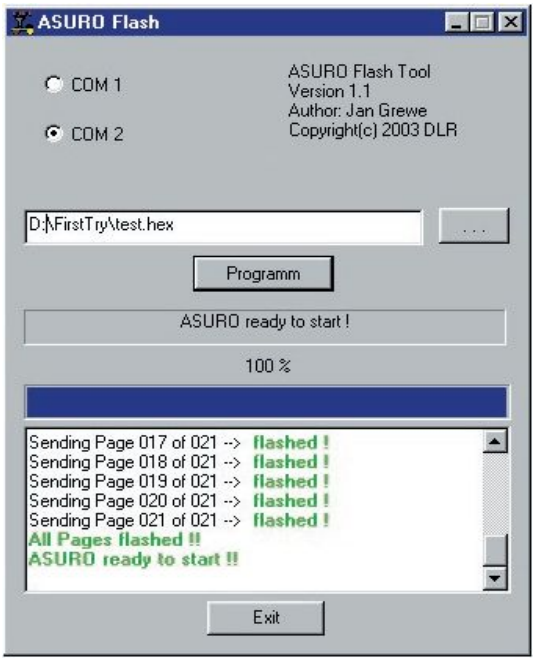


Шта се догодило?

Позивна функција ”clean” позива из позадине фајл Test-clean.bat. Извршење почиње стартовањем параметра “clean’.

4.4 Flash ASURO програмски алат

Flash алат се користи за пренос података са рачунара ка ASURO роботу.



После стартовања програма и избора интерфејса на који је прикључен IR-Transrisiver, имамо могућност да одаберемо у ком фолдеру се налази наш .hex фајл који треба пренети у меморију робота.

Потребно је поставити робота близу IR-Transrisiver-а на максималној удаљености од 50cm. Пријемник робота и одасиљац IR-Transrisiver-а морају бити директно усмерени један наспрам другог. Кликом на дугме Programm процес преноса података одпочиње. Прекидач S1 треба пребацити на ON пре него што индикатор статуса инсталације стигне до самог краја статусне линије.

Када је комуникација извршена успешно пребацени фајл Test.hex биће сместен у Flesh-меморију процесора, где остаје доступан и након самог искључивања напајања робота.

Да би програм радио како треба потребно је робота искључити и поново укључити, како би се прогроргам стартовао.

5.0 ИГРЕ

На основу расположивих ресурса (сензора и робота), осмислили игре у којима смо искористили њихове потенцијале.

- У игри у којој робот чисти затворену област коришћени су снезори за детекцију рефлектоване светлости.
- У другој игри која има за циљ да се роботи међусобно избаце из ринга, коришћени су снезори за мерење интензитета светлости и прекидачки сензори за детекцију удара
- У трећој игри чија је сврха да се детектује и прати топлотни извор, коришћен је сензор за детекцију извора топлоте (тзв. SnakeVison).
- У последњој игри робот прати присволјну црну линију, уз помоћ фотосензора, а са електромоторима одржава правац.

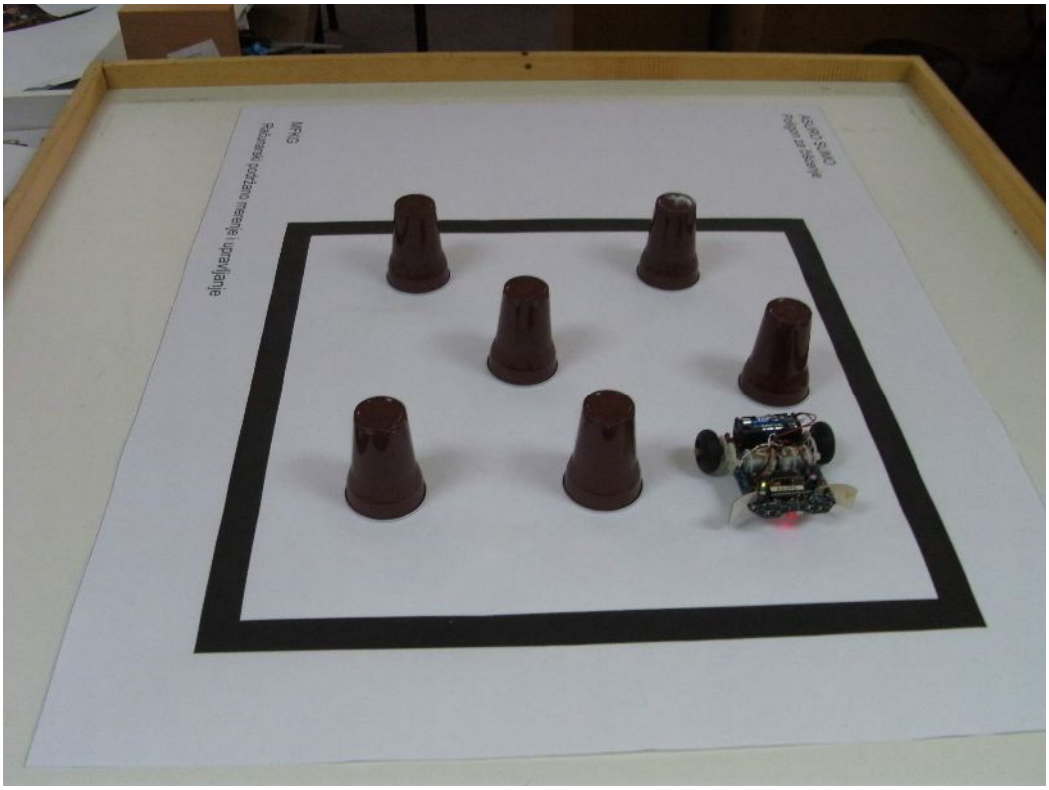
5.1 ASURO – чистач обележеног простора

Ова игра представља оспособљавање робота да било који затворени простор одржавају чистим. Програмирањем асуро робота представили смо симулацију наведеног проблема. У овом случају робот гура пластичне чаше изван поменутог простора и тиме испуњава испрограмирану функцију.

Применом оваквог кода или његовом модификацијом може се на пример програмирати неки робот који ће самостално да усисава радни простор или да скупља одпатке који су бачени на под.

Робот је испрограмиран тако да у простору који је означен црном линијом, у зависности од вредности на фото сензору, бира пут након што наиђе на црну линију.

Слика 29 – Чишћење чаша у сумо рингу



Слика 30 – чишћење чаша у обележеном простору

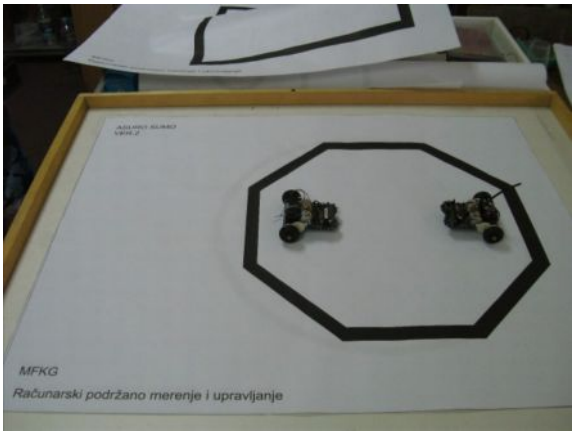
5.1.1 Програмски код за чишћење чаша

```
#include "asuro.h"                    // standardna biblioteka

int main(void){                       // pocetak main funkcije
unsigned int lineData[2];             // promenljiva za prikupljanje podataka sa senzora
unsigned int i;                       // brojac

Init();                               // pokretanje funkcija robota

while(1){                             // beskonacna petlja

   LineData(lineData);                // funkcija za prikupljanje podataka sa senzora

   FrontLED(ON);                      // ukljucena prednja dioda

   MotorDir(FWD,FWD);                 // pravac motora podesen na napred
   MotorSpeed(160,160);               // podesavanje brzine motora

if((lineData[0] < 40) || (lineData[1] < 40)){       // proveravaju se podaci sa senzora

   if(lineData[0]<lineData[1]){       // provera vrednosti na levom i desnom senzoru

      MotorSpeed(0,0);                // podesavanje brzine motora
         for(i=0;i<150;i++){          // petlja za cekanje 150 milisekundi
            Sleep(72);
         }

      MotorDir(BREAK,RWD);            // pravac motora podesen za skretanje u levo
      MotorSpeed(0,180);              // podesavanje brzine motora za skretanje
         for(i=0;i<400;i++){          // petlja za cekanje od 400 milisekundi
            Sleep(72);
         }

         MotorDir(FWD,FWD);           // pravac motora podesen na napred
         MotorSpeed(160,160);         // podesavanje brzine motora
   }
   else if(lineData[0]>lineData[1]){  // provera vrednosti na levom i desnom senzoru
      MotorSpeed(0,0);                // podesavanje brzine motora
         for(i=0;i<150;i++){          // petlja za cekanje 150 milisekundi
            Sleep(72);
         }

      MotorDir(RWD,BREAK);            // pravac motora podesen za skretanje udesno
      MotorSpeed(180,0);              // podesavanje brzine za skretanje
         for(i=0;i<400;i++){          // petlja za cekanje od 400 milisekundi
            Sleep(72);
         }

      MotorDir(FWD,FWD);              // pravac motora podesen na napred
      MotorSpeed(160,160);            // podesavanje brzine motora

      }
   }
}

return 0;                             // vracanje nule main funkciji
}
```

5.2 ASURO – сумо

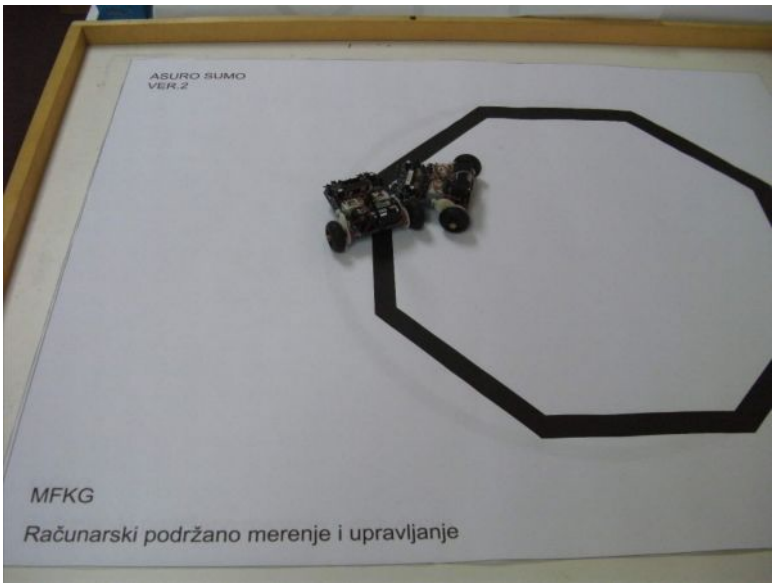
Ова игра представља симулацију популарне јапанске игре. Циљ игре је да се роботи међусобно изгурају ван означеног ринга који може бити кружног или октагоналног облика. Границе ринга су означене црном линијом.

Логика којом је робот програмиран своди се на употребу фото сензора и сензора за притисак. На почетку игре два робота постављају се у произвољном правцу како би се избегло сударање одмах на почетку игре. Робот путује праволинијски с тим што се стално врши провера на фото сензорима како робот не би изашао ван ринга и тиме изгубио меч. Када робот на свом путу наиђе на црну линију у зависности од осветљења на левом и десном сензору, робот сам одлучује на коју страну ће да се окрене и настави потрагу за другим роботом. У случају да налети на другог робота, тј. када се активирају сензори за притисак, у зависности од тога који је прекидач притиснут робот ће се кретати у одређеном правцу. Уколико су активирани прекидачи на левој или десној страни микроконтролер треба да повећа брзу на једном од точкова како би исправио робота и тиме активирао друга два предња прекидача помоћу којих микроконтролер зна да се налази у правцу са другим роботом и говори моторима да гурају максималном брзином док не избаце другог робота ван ринга. Победник је онај робот који остане сам у рингу.



Слика 31 – Почетак сумо игре

Слика 32 – тражење противичког робота



Слика 33 – Избацивање робота ван ринга.

5.2.1 Програмски код за сумо борбу

```
#include "asuro.h"                  // standardna biblioteka

void gurajLevo(void){               // funkcija koja sluzi da ispravi robota
MotorDir(FWD,FWD);                    // podesavanje pravca motora
MotorSpeed(220,100);                // podesavanje brzine motora
}

void gurajNapred(void){               // funkcija koja sluzi da gura robota van ringa
MotorDir(FWD,FWD);                    // podesavanje pravca motora
MotorSpeed(255,255);                // podesavanje brzine motora
}

void gurajDesno(void){                // funkcija koja sluzi da ispravi robota
MotorDir(FWD,FWD);                    // podesavanje pravca motora
MotorSpeed(100,220);                // podesavanje brzine motora
}

int main(void){                     // pocetak main funkcije
unsigned int lineData[2];             // promenljiva za prikupljanje podataka sa foto senzora
unsigned int i;                     // brojac
unsigned char prekidac;               // promenljiva za prikupljanje podataka sa senzora za pritisak

Init();                           // pokretanje funkcija robota

while(1){                         // beskonacna petlja
   LineData(lineData);                // funkcija za prikupljanje podataka sa foto senzora
   prekidac = PollSwitch();         // dodeljivanje vrednosti promenljivoj sa senzora za pritisak
```

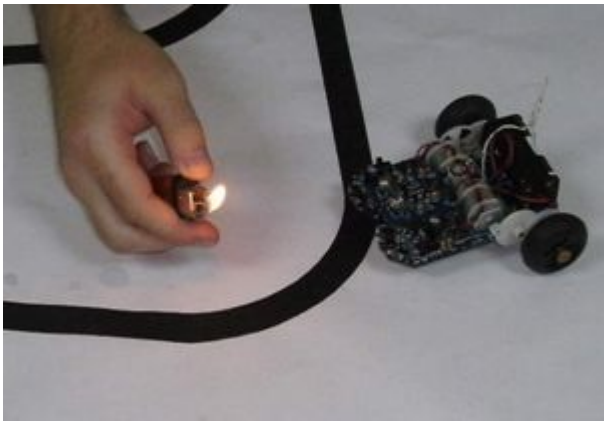
```
    FrontLED(ON);                   // podesavanje prednje diode

    MotorDir(FWD,FWD);              // podesavanje pravca motora
    MotorSpeed(140,140);            // podesavanje brzine motora

if((lineData[0] < 40) || (lineData[1] < 40)){       // provera da li je robot naisao na granicu ringa

    if(lineData[0]<lineData[1]){       // medjusobna provera levog i desnog foto senzora

        MotorSpeed(0,0);            // podesavanje brzine motora
           for(i=0;i<150;i++){      // petlja za cekanje od 150 milisekundi
              Sleep(72);            // funkcija za cekanje od 1 milisekunde
           }

        MotorDir(FWD,RWD);          // podesavanje pravca motora
        MotorSpeed(130,150);        // podesavanje brzine motora
           for(i=0;i<300;i++){      // petlja za cekanje od 300 milisekundi
              Sleep(72);            // funkcija za cekanje od 1 milisekunde
           }

           MotorDir(FWD,FWD);       // podesavanje pravca motora
           MotorSpeed(140,140);     // podesavanje brzine motora
    }
    else if(lineData[0]>lineData[1]){    // medjusobna provera levog i desnog foto senzora
        MotorSpeed(0,0);            // podesavanje brzine motora
           for(i=0;i<300;i++){      // petlja za cekanje od 300 milisekundi
              Sleep(72);            //  funkcija za cekanje od 1 milisekunde
           }

        MotorDir(RWD,FWD);          // podesavanje pravca motora
        MotorSpeed(130,150);        // podesavanje brzine motora
           for(i=0;i<300;i++){      // petlja za cekanje od 300 milisekundi
              Sleep(72);            // funkcija za cekanje od 1 milisekunde
           }

        MotorDir(FWD,FWD);          // podesavanje pravca motora
        MotorSpeed(140,140);        // podesavanje brzine motora
        }
    }
    else if( prekidac > 0 ){            // provera senzora za pritisak
        switch(prekidac){               // switch funkcija
           case 1: gurajLevo();break;       // guraj robota ulevo za slucaj 1
           case 2: gurajNapred();break;     // guraj robota napred punom brzinom
           case 4: gurajNapred();break;     // guraj robota napred punom brzinom
           case 8: gurajNapred();break;     // guraj robota napred punom brzinom
           case 16: gurajNapred();break;    // guraj robota napred punom brzinom
           case 32: gurajDesno();break;     // guraj robota udesno za slucaj 32
        }
    }

}
return 0;                           // vrati nulu main funkciji
}
```

5.3 ASURO- термални сензор, праћење извора топлоте.

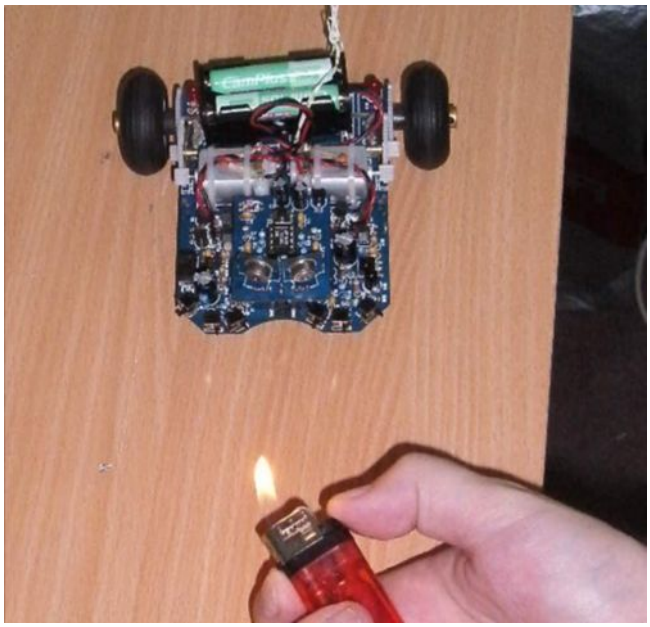
Коришћењем сензора snake vision осмислили смо игру помоћу које робот прати изворе топлоте. Сензори који су уграђени у асура способни за детекцију различитих интензитета топлоте на малим растојањима.

Робот се налзи у стању приправности док се не детектује пожар (извор светлости са свеће) и на основу вредности температуре на левом и десном термалном сензору прати путању извора топлоте.

Применом ове симулације можемо да осмислимо систем за аутоматско детектовање и гашење пожара у ситуацијама када није могуће да човек реагује већ се та улога доделјује роботима.



Слика 34 – Праћење извора светлости



Слика 35 – Праћење извора светлости

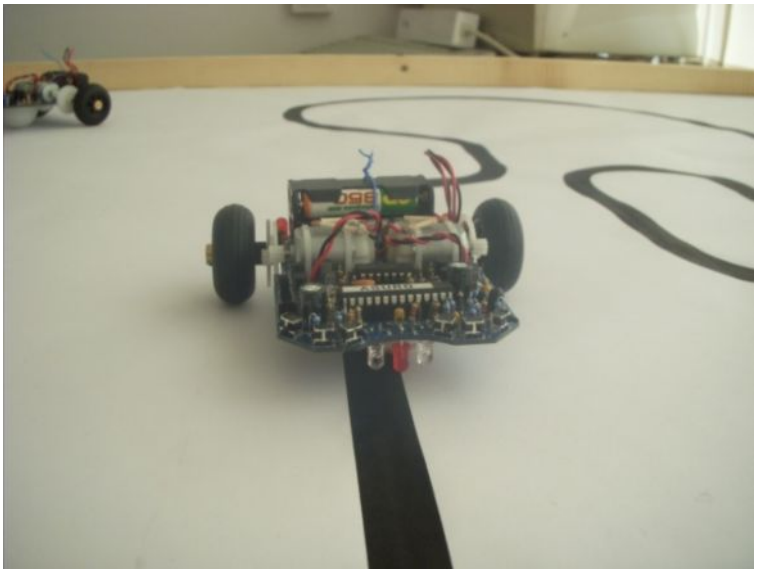
5.3.1 Програмски код за термални сензор

```
#include"asuro.h"                          // standardna biblioteka
#define granicnaTemperatura 300        // definisanje granicne temperature u kelvinima



void ThermalData(unsigned int *data) {         // funkcija za prikupljanje vrednosti sa termalnog senzora

ADMUX = (1 << REFS0) | (1 << REFS1) | IR_LEFT;
ADCSRA |= (1 << ADSC);
while (! (ADCSRA & (1 << ADIF)));
ADCSRA |= (1 << ADIF);
data[0] = ADCL + (ADCH << 8);


ADMUX = (1 << REFS0) | (1 << REFS1) | IR_RIGHT;
ADCSRA |= (1 << ADSC);
while (!(ADCSRA & (1 << ADIF)));
ADCSRA |= (1 << ADIF);
data[1] = ADCL + ( ADCH << 8);
}

void voziRobotaNapred(){        // funkcija za kratanje napred

BackLED(ON,ON);                  // podesavanje zadnjih dioda
MotorSpeed(180,180);           // podesavanje brzine motora

}
void voziRobotaUlevo(int a, int b){        // funkcija za skretanje u levo

BackLED(OFF,ON);                     // podesavanje zadnjih dioda
MotorSpeed(a,b);                    // podesavanje brzine motora

}
void voziRobotaUdesno(int a, int b){    // funkcija za skretanje u desno

BackLED(ON,OFF);                     // podesavanje zadnjih dioda
MotorSpeed(a,b);                    // podesavanje brzine motora

}
void stani(void){                    // funkcija za zaustavljanje robota

StatusLED(OFF);                      // podesavajne statusne diode
BackLED(0,0);                       // podesavanje zadnjih dioda
MotorSpeed(0,0);                    // podesavanje brzine motora

}

int main (void) {                    // pocetak main funkcije

   unsigned int temperatura[2];       // promenljiva za prikupljanje podataka sa senzora
   unsigned int prvaBrzina = 180;       // podesavanje vrednosti jedne brzine
   unsigned int drugaBrzina = 90;       // podesavanje vrednosti druge brzine
   signed int razlika;                 // definisanje razlike u toploti na senzorima
   signed int suma;                   // definisanje sume toplote na senzorima

   Init();                          // funkcija za pokretanje robota
```

```
    voziRobotaNapred();                 // pozivanje funkcije za kretanje

    while(1){                           // beskonacna petlja

       ThermalData(temperatura);        // funkcija za prikupljanje podataka sa senzora

       suma=temperatura[0]+temperatura[1];    // definisanje sume na senzorima


    if(suma>granicnaTemperatura){       // provera senzora
          StatusLED(GREEN);             // podesavanje statusne diode


       // definisanje razlike u temperaturama
       razlika = ((signed)temperatura[0] - (signed)temperatura[1])*32/suma;

       if(razlika >3){       // razlika na senzorima je veca od tri stepena

       voziRobotaUlevo(drugaBrzina,prvaBrzina);    // pozivanje funkcije za kretanje
       }

       else if(razlika < -3){       // razlika na senzorima je manja od tri stepena

       voziRobotaUdesno(prvaBrzina,drugaBrzina);    // pozivanje funkcije za kretanje
      }

       else{

         voziRobotaNapred();        // pozivanje funkcije za kretanje
       }
    }

    else{
       stani();        // pozivanje funkcije za zaustavljanje
    }
}

return 0;        // vracanje nule main funkciji
}
```

5.4 Праћење црне линије

Претпоставимо да два фототранзистора опкораче црну линију на белој површини, и да је ширина црне линије нешто мања од простора између транзистора. Идеално, ако су фототранзистори удаљени једнако од линије примаће једнаку количину светлости која се рефлектује са површине, па ће и произведена волтажа бити иста. Међутим ако робот делимично скрене, тако да фототранзистори не буду једнако удаљени од црне линије, један од транзистора ће примати више светлости од другог (зато што ће црна линија смањити интезитет светлости коју један од фототранзистора прима) па произведена волтажа неће бити иста. Може се написати програм који одговара на разлике произведених волтажа фототранзистора. Приказ поменутих делова дат је на слици 3.испод.



Slika 3. Trenutak kada levi fototranzistor izađe nizvan linije

5.4.1 Програмски код за термални сензор

```
#include "asuro.h"              // standardna biblioteka

void voziNapred(void){          // funkcija za kretanje napred
MotorDir(FWD,FWD);                      // podesavanje pravca motora
MotorSpeed(80,80);              // podesavanje brzine motora
}

void skreniDesno(void){         // funkcija za skretanje u desno
MotorSpeed(150,60);             // podesavanje brzine motora
}
void skreniLevo(void){          // funnkcija za skretanje u levo
MotorSpeed(60,150);             // podesavanje brzine motora
}

int main(void)                  // pocetak main funkcije
{
unsigned int data[2];           // promenljiva za prikupljanje podataka sa senzora
Init();                                 // pokretanje funkcija robota
FrontLED(ON);                           // podesavanje prednje diode
voziNapred();                   // pozivanje funkcije za kretanje
while(1){                               // beskonacna petlja
        voziNapred();           // pozivanje funkcije za kretanje

LineData(data);                         // funkcija za prikupljanje podataka sa foto
senzora
if(data[0]>data[1]){            // medjusobna provera na foto senzorima
        skreniDesno();          // pozivanje funkcije za kretanje
}
else{
        skreniLevo();           // pozivanje funkcije za skretanje
}

}
return 0;                               // vracanje nule main funkciji
}
```

ЛИТЕРАТУРА

Књиге:

Mehr SpaB mit ASURO- Band I, Robin Gruber, Martin Hofmann.

Mehr SpaB mit ASURO- Band II, Robin Gruber, Martin Hofmann.

Рачунарски подржано мерење и управљање, др. Милан Матијевић.

Семинарски радови:

Праћење линије са ASURO роботом, Милош Витошевић.

Праћење црне линије и трка ASURO робота, Радовић Милош, Живковић Марко, Дачовић Драган.

ASURO робот у лавиринту, Вукићевић Арсо

Интернет:

http://www.arexx.com/

http://asurowiki.de/

Садржај