

Stavebnice programovatelného robota

ASURO ARX-03

Obj. č.: 19 14 51



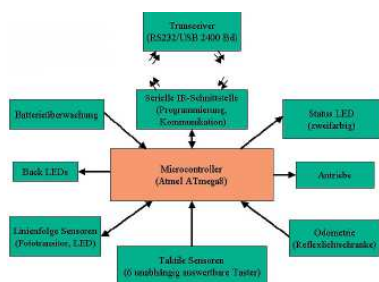
Vážení zákazníci,

děkujeme Vám za Vaši důvěru a za nákup stavebnice programovatelného robota ASURO ARX-03. ASURO je mobilní mini robot, kompletně programovatelný v jazyku C, který byl vyvinut ke vzdělávacím účelům v oddělení robotiky a mechatroniky Německého centra pro letectví a kosmonautiku. Sestavování stavebnice je jednoduché pro zkušené elektrotechniky a zvládnutelné pro začátečníky. S výjimkou desky plošných spojů (DPS) se ke skládání používají jen standardní součásti. K programování se používá jen volně dostupný software. Proto se ASURO výborně hodí jako projekt pro zájemce o hobby elektroniku s ovládacím procesorem, pro vzdělávací projekty na školách a univerzitách i pro centra vzdělávání dospělých. Ve všech fázích vývoje elektroniky robota a jeho softwaru byly použity nástroje, které jsou volně dostupné pro soukromé použití, což dokazuje, že roboty lze vytvořit i bez použití drahých nástrojů a mechatik. ASURO je vybaven procesorem RISC a dvěma nezávisle ovládanými motory, optickou jednotkou pro sledování dráhy pohybu, šesti přepínači pro detekci kolize, dvěma odometrickými senzory a sadou pro infračervenou komunikaci, která umožňuje programování a dálkové ovládání přes PC (viz obr. 0.1).

Varovný symbol upozorňuje na paragrafy, které vyžadují důkladné prostudování, aby se zabránilo poškození některých komponentů nebo poranění osob.

Všechny uživatelé musíme upozornit, že ASURO není hračka a neměl by se dostat do rukou dětí mladších 3 let, protože obsahuje množství malých součástí a hrozí nebezpečí jejich spolknutí. Opatřete si prosím nějaké baterie a začněte se skládáním.

A ještě něco: ASURO je jen jiný způsob jak říct: "Další malý a unikátní robot z Oberpfaffenhofenu!"



Obr. 0.1 Blokové schéma ASURO (viz také příloha E)

UPOZORNĚNÍ

- Právo na vrácení výrobku ztrácíte po otevření plastových sáčků obsahujících jednotlivé části a komponenty.
- Pozorně si přečtěte tento návod ještě předtím, než začnete robota skládat.
- Při manipulaci s nástroji zachovávejte opatrnost.
- Výrobek udržujte mimo dosah dětí. Neskládejte robota v přítomnosti malých dětí. Mohly by se poranit nástroji, nebo spolknout malé části a komponenty.
- Zkontrolujte správnou polaritu baterií.
- Dejte pozor, aby byly baterie stále suché. Pokud se ASURO namočí, vyjměte baterie a všechny části co nejlépe vysušte.
- Pokud nebudete robota delší čas používat, vyjměte z něj baterie.

Mechanická část

Potřebné nástroje

Kromě součástí stavebnice budete při sestavování robota ASURO potřebovat následující nástroje:

- Držák pro pájení ("Třetí ruka")
- Nůž s odlamovací čepelí
- Malé kladivo
- Jehlové kleště pro práci s elektronikou
- Štípačky na kabely
- Pájčku, nebo pájecí stanici (přibližně 20 - 30 W)
- Pájecí drát (1 mm), pokud možno bez olova
- Odpájecí knot (2 - 3 mm) - jen pro odstranění přebytečné pájky
- Lepidlo (buď instantní jednosložkové lepidlo, nebo dvousložkové lepidlo, nebo pistole pro tavné lepení)
- Počítač: laptop nebo osobní počítač (s operačním systémem Windows, nebo Linux)
- Volitelně: multimetr

* Pro volitelný vysílač - přijímač USB IR může být potřebný USB kabel A - B



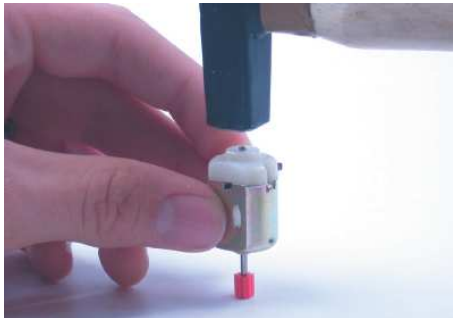
Obr. 1.1 Potřebné nástroje

Příprava mechanické části

Nejdříve je potřebné zkontrolovat úplnost celé stavebnice. Seznam všech dílů je uveden v příloze A. Dříve než přistoupíme k práci s elektronikou, musíme ještě provést určitou mechanickou přípravu. Takže začínáme.

Pastorek motoru

Výkon motoru se převádí na převod pomocí malých ozubených kol (s ozubením 1,9 mm), které se musí nasunout na nápravu motoru. Zaoblenou stranu ozubeného kola položte na rovný povrch a nápravu motoru natlačte jemně do otvoru ozubeného kola, aniž byste přitom použili příliš síly. Pro úplné zasunutí nápravy motoru do ozubeného kola můžete použít malé kladívko. Položte ozubené kolo na nějaký měkký povrch (např. kartonový papír) a kladívkem jemně poklepejte na zadní stranu nápravy motoru (viz obr. 2.1).



Obr. 2.1: Montáž pastorku motoru.

Pingpongový míček

ASURO je navržen tak, aby se klouzal na půlce pingpongového míčku, který se musí připravit. Nejlepší bude, když vezmete celý pingpongový míček a pilkou, nebo odlamovacím nožem jej rozřezáte na dvě půlky. Hrany rozřezaného míčku můžete vyčistit pilníkem nebo smirkovým papírem.

To je vše!



Obr. 2.2 Pingpongový míček



Nepoužívejte elektrické nástroje, jako např. elektrickou pilku, nebo elektrický nůž. Pingpongový míček se může lehce vznítit.

Nápravy

Nápravy jsou vyrobeny z mosazné tyče. Potřebujeme dva páry náprav, v délce 24,5 mm, resp. 42,0 mm. Nápravy v patřičné délce jsou součástí dodávky.

Senzory kol

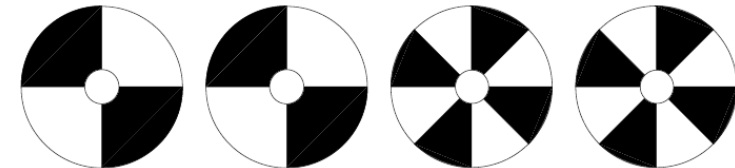
Odometr, který se skládá z diody emitující světlo (LED) a fototranzistoru, směřuje dopředu na černé a bílé značky na ozubeném kole 50/10.

Kompletní sada nalepovacích terčů odometru je součástí stavebnice. Nalepte je na ozubené kolo 50/10, jak ukazuje obrázek 2. 3. Kvůli kompatibilitě s jinými roboty ASURO (demo programy) doporučujeme použít terčíky senzoru se 6 černými a bílými prvky.



Obr. 2. 3 Doporučované terčíky na ozubených kolech

Čím více prvků na terčik namalujete, tím přesněji se bude registrovat rychlost ozubeného kola a tím lépe bude fungovat ovládání rychlosti. Velký počet černých a bílých prvků však snižuje rozdíly mezi světlem a tmou (viz obr. 2. 4).



Obr. 3. 4: Příklad označení ozubených kol

Tím jste dokončili mechanickou přípravu.

Elektronická část

Pokyny k pájení

ASURO má větší množství součástí určených k připojení (na rozdíl od menšího počtu součástek, jejichž vývody se pájí přímo na povrch DPS. Obr. 3. 1 ukazuje nejmenší dostupné pouzdro procesoru IC v ASURO a pouzdro, které se připojuje. Čip uvnitř obou pouzder je však stejný! I když je pájení připojených součástí snadné a pohodlné, obzvláště nezkušené osoby musí zvážit některá preventivní opatření.



Deska plošných spojů musí být samozřejmě během pájení odpojena od napájení. Nestačí jen vypnutí!! Odstraňte baterie a odpojte zdroje proudu.

Pájecí hrot, pájka a teplota

Obrázek 3. 2 ukazuje základy pájení. Horké místo pájeného zařízení dosahuje při pájení olověnou pájkou teplotu kolem 360 °C a při použití bezolovnatých pájek kolem 390 °C, přičemž pájení náprav lze provádět i při vyšších teplotách (420°C). Elektronika by se měla pájet s miniaturním hrotem, zatímco pro pájení náprav se musí použít hrotem s větší pájecí plochou.

Použijte lehce navlhčenou pájecí houbu a naneste trochu pájky na horký pájecí hrot. Krátce před prvním použitím pájecího zařízení nebo po přestávce v pájení se musí z pájecího hrotu odstranit stará pájka. Použijte pájku s průměrem 0,8 nebo 1 mm pro práci s elektronikou.



Obr. 3. 1: Porovnání největšího a nejmenšího pouzdra pro čip ATmega8L



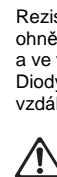
OBR. 3. 2: Základ pájení



Při pájení se vytváří kouř, který může poškodit zdraví. Nedýchejte vznikající plyny, resp. pracujte pod odsavačem par. Používáním kalafuny, resp. pájky, která není určena pro elektroniku, můžete zničit pájené komponenty.

Příprava částí

Pájení malých částí vyžaduje určitou zkušenost a fortel dostat se k místům pájení. Pokud jste už někdy letovali, možná jste zjistili, že byste přitom potřebovali ještě jednu ruku (tento problém může částečně vyřešit používání držáku, kterému se říká třetí ruka). Některé součásti (tranzistory, LED, integrované obvody, přepínače, kondenzátory a propojky) jsou již připraveny k nasazení na DPS, zatímco diody a rezistory musíte nejdříve připravit pro upevnění na DPS.



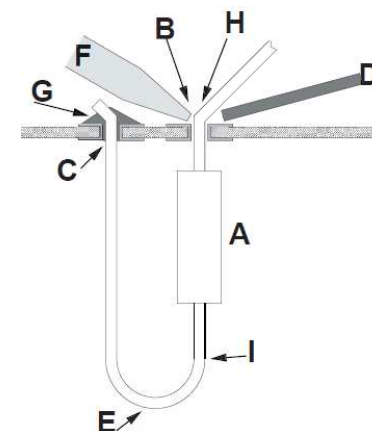
Rezistory se do robota vkládají svisle. Jeden vývod rezistoru přitom nechte v původní poloze a druhý ohněte o 180°. Aby se rezistor nepoškodil, musí se ohýbání provést v zakřivení s průměrem 2,5 mm a ve vzdálenosti několika milimetrů od těla rezistoru. Diody se vkládají ve vodorovné poloze. Oba vývody se přitom musí ohnout (např. kleštěmi) v takové vzdálenosti, aby vedly do otvorů na DPS.

Prvky jako procesor IC1 ATmega, IC3 CD4081 a IR-přijímač IC2 SFH5110-36 jsou citlivé na elektrostatický výboj. Pokud jste nabití elektřinou (např. chozením po koberci), můžete se tyto části poškodit pouhým dotykem, nebo dokonce když se k nim co i jen přiblížíte rukou. Před manipulací s těmito komponenty byste se měli nejdříve vybit pomocí antistatického náramku.



Obr. 3. 3: Části se zahnutými vývody

- A. Součástka
- B. Vývod, pájecí podklad a ohnuté místo se musí zahřívát současně
- C. Pájka se musí dostat do otvoru
- D. Pájka
- E. Zahnutá část bez hran
- F. Pájecí hrot
- G. Dokonalý pájecí kužel
- H. Zahnuté vývody, které zabrání, aby součástka vypadla z DPS
- I. Začátek ohybu v určité vzdálenosti od součástky



Obr. 3. 4: Vytvoření čistého pájecího spoje

Pájení součástek

Když jsou připraveny drátové vývody, je třeba připevnit součástky k metalizovaným otvorům na desce plošných spojů (DPS) a součástky, které mají jen dva nebo tři vývody roztáhnout. Zahnutí vývodů na zadní straně DPS zabrání, aby součástka vypadla. V případě součástek s více vývody, např. patič pro integrované obvody, bude úplně postačovat, když se zahnou dva diagonálně proti sobě položené vývody.

Po připevnění součástky zahřejte pájecím hrotem současně vývod a pájecí podklad. Zároveň musíte přidat malé množství pájky. Roztavená pájka poteče do metalizovaného otvoru. Přidejte trochu pájky, dokud nebude otvor zcela zaplněn (viz obr. 3. 4). Nyní odstraňte přebytečnou pájku a páječku. Se součástkou ani s DPS nehýbejte, dokud spoj nevychladne a nebude pevný. Jestliže se během tuhnutí spoje se součástkou pohne, bude to mít za následek, že spoj bude nespolehlivý a bude docházet k občasnému selháním.

Špatně letované spoje poznáte podle kulatých kapek pájky na podkladě nebo podle matného povrchu (v případě bezolovnaté pájky bude povrch dokonce extrémně matný) a musí se letovat znovu. Při vkládání patič a jiných součástek, které se vkládají na DPS vodorovně, můžete použít následující trik: Nejdříve přileťte jen jeden vývod součástky. Poté zatlačte součástku lehce dolů, zatímco zahříváte stejný podklad (**Pozor: součástka může být velmi horká**). Součástka si nyní sedne

na povrch desky a zůstane v pevné poloze. Nyní naletujte všechny ostatní kontakty a nakonec znovu nějakou extra pájkou ještě první kontakt.

Po přiletování součástky byste měli přebývajících částí vývodů useknout štípačkami. Vývody usekněte těsně u DPS a dávejte pozor, abyste vývody nevytrhli nebo na ně netlačili.



Při stříhání drátů dávejte pozor na odlétávající kousky ostrých okrajů drátů. Svisle montované součástky se samozřejmě nesmějí navzájem dotýkat, a pokud jste je umístili příliš blízko sebe, musí se ohnout od sebe.

Odpájení

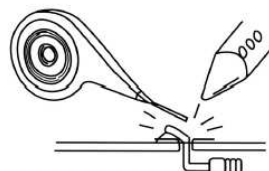
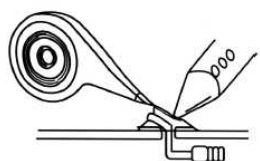
Pokud se náhodou stane, že se součástka dostane na místo, kam nepatří, tak ji musíte odstranit. Jak jste mohli očekávat, ASURO má oboustrannou DPS s metalizovanými otvory, díky čemuž je odstraňování součástek jednodušší.

Může být pro vás užitečné:

Do pájky součástky, která se musí odstranit, přidejte trochu nové pájky. Když jsou všechny pájené spoje součástky zahřáté, pokuste se kleštěmi odstranit součástku z DPS. Nakonec můžete pájku snadno odstranit pomocí odsávacího knotu.

Odpájecí knot položte na pájku (viz krok 1 a 2, níže). Můžete tak udělat, dokud je ještě součástka na DPS! Zahřejte zároveň knot i pájku. Po určitém zahřátí knot nasaje pájku do měděného pleťence. V tomto momentu odstraňte rychle páječku i knot.

Pokud zbyla ještě pájka na druhé straně otvoru, můžete ji odstranit stejným způsobem.



Krok 1

Přiložte měděný pleťenec na pájku komponentu, který se musí odstranit. Zahřejte pleťenec i pájku a pleťenec nasaje pájku.

Krok 2

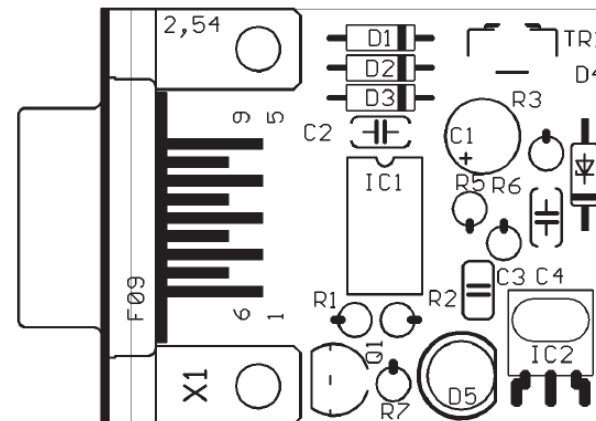
Odstraňte současně páječku i měděný pleťenec.

Sestavování elektroniky

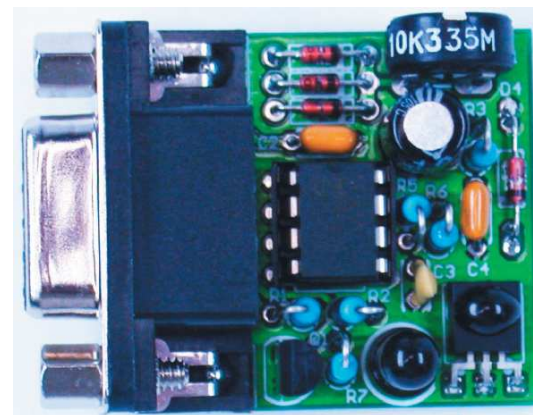
Sestavení infračerveného vysílače RS232

- IC1: Nejdříve vložte 8 pólovou patičku. Označení polaritů na patičce (lehce asymetrické) musí odpovídat označení příslušných symbolů na DPS.
- D1, D2, D3: 1N4148, věnujte pozornost polaritě! Přečtěte si popisky na součástkách a dávejte pozor, abyste je nezaměnili za ZPD5.1 nebo BZX55-C5V1!
- D4: ZPD5.1 nebo BZX55-C5V1, věnujte pozornost polaritě! Přečtěte si popisky na součástkách a dejte pozor, abyste je nezaměnili za 1N4148!
- D5: SFH-415-U IR LED (černá dioda) - věnujte pozornost polaritě a zatlačte ji dolů na DPS.
- C1: 100 μ F, alespoň 16 V, věnujte pozornost polaritě!
- C2, C4: keramický kondenzátor, 100 nF, potisk: 104
- C3: keramický kondenzátor, 680 pF, potisk: 681
- Q1: BC547 (A, B nebo C) nebo BC548 (A, B nebo C)
- R1, R5: 20 k Ω (červený, černý, oranžový, zlatý)
- R2: 4,7 k Ω (žlutý, fialový, červený, zlatý)
- R3: 470 Ω (žlutý, fialový, černý, zlatý)
- R6: 10 k Ω (hnědý, černý, oranžový, zlatý)
- R7: 220 Ω (červený, červený, hnědý, zlatý)
- TR1: 10 k Ω rezistor s proměnnou hodnotou

- IC2: Infračervený přijímač IC SFH5110-36, vhodnými kleštěmi ohněte vývody! Věnujte pozornost polaritě (ohyb se musí udělat směrem ven)! Pozor: Součástka se může poškodit elektrostatickým výbojem (ESD) nebo teplem při pájení!
- X1: 9 pol. konektor SUB-D, pouzdro se musí osadit do blízkosti DPS. Přiletovat se musí také připevňovací pásy.
- IC1: vložte NE555P, věnujte pozornost polaritě!



Nakonec zkontrolujte, jestli na DPS nedošlo ke zkratům a k chybám v polaritě. Zkontrolujte kvalitu pájených spojů a špatné spoje znovu přiletujte.



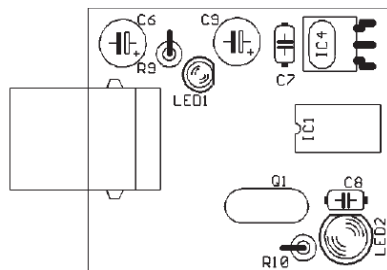
Obr. 4. 4: Sestavený IR vysílač

Pokud je vysílač RS-232-IR připraven k propojení s ostatními součástkami, bude vypadat jako výše uvedený obrázek.

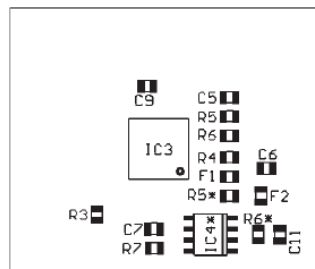
Sestavení IR USB vysílače



Obr. 4. 2: Infračervený USB vysílač



Obr. 4. 3: Strana s komponenty IR USB vysílače



Obr. 4. 4: Spodní strana IR USB vysílače

Vkládání součástek na desku plošných spojů robota ASURO

Dvě delší nápravy, které budou potřebné k druhé části převodovky, se přilepují, nebo přilepí na spodní stranu DPS. Nejlepším řešením bude pájení, protože spoj ztvrdne mnohem rychleji než při lepení a pokud to bude potřebné, můžete jej později vždy snadno opravit.

Dvě kratší nápravy se musí připevnit na horní stranu DPS blíže ke středu desky. Před připojením se nápravy musí vyčistit na pájené, nebo lepené straně (ale ne na straně kde jsou kola) jemným sirkovým papírem (číslo zrnitosti 240 nebo vyšší), aby se zlepšila přilnavost pájky, resp. lepidla.

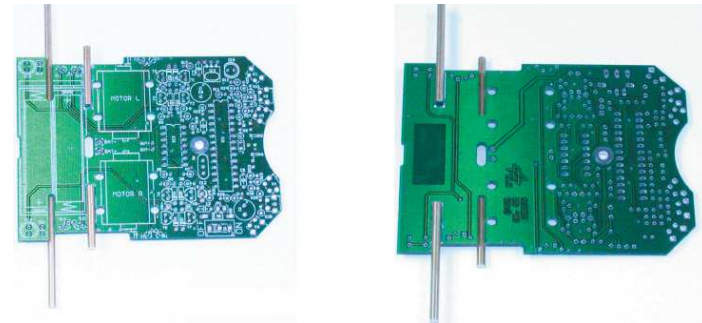
Pokud se rozhodnete pro pájení těchto částí, postupujte podle následujících kroků:

Nejdříve se připojí delší nápravy. Položte DPS na stůl, spodní stranou nahoru a delší nápravu vložte do žlábků. Zatlačte ji celkem až na konec žlábků. Náprava by měla ležet rovně v délce žlábků.

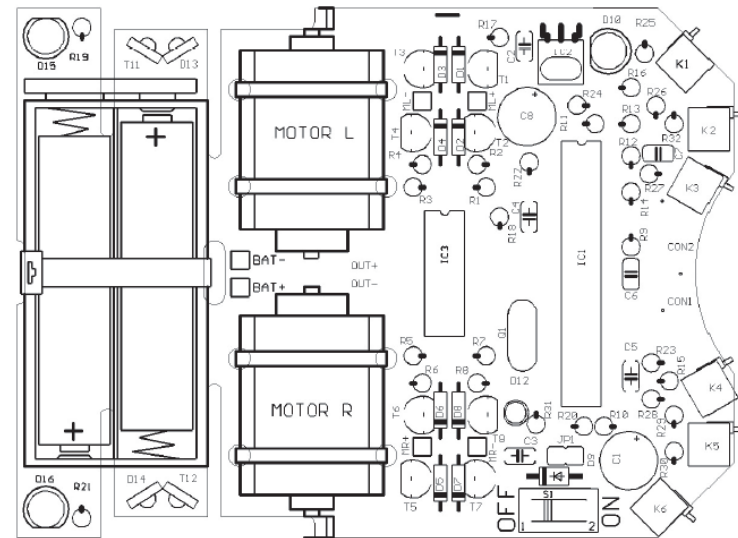
Namočte pájecí hrot do nějaké pájky a zatlačte jej na nápravu. Když se náprava zahřeje, musíte přidat pájku na pájený podklad pod nápravou. Po dokončení pájení můžete dát páječku pryč, zatímco pomocí šroubováku tlačíte nápravu směrem dolů, dokud pájený spoj zcela nevychladne.

Opakujte stejný postup pro připojení druhé dlouhé nápravy. Otočte DPS a zopakujte celý proces i s oběma krátkými nápravami. Na obr. 4. 3 je zobrazena DPS se 4 připojenými nápravami.

Po ochlazení se k nápravám můžou připojit kola. Ozubena kola musí přesně sedět a kola se musí snadno otáčet. Jestliže se kola neotáčejí lehce, byly zřejmě nápravy připojeny chybně a celou operaci budete muset opakovat. V případě, že na povrchu nápravy zůstaly v části pro připevnění kol připevněny nějaké zbytky pájky, musí se odstranit smirkovým papírem, nebo jemným pilníkem. Pokud se kola otáčejí lehce, můžete je odstranit a dát na stranu, aby se na DPS mohly vkládat elektronické součástky.



Obr. 4. 3: DPS ASURO s připevněnými nápravami



Obr. 4. 2: Pohled na vkládání součástek na horní stranu DPS-ASURO

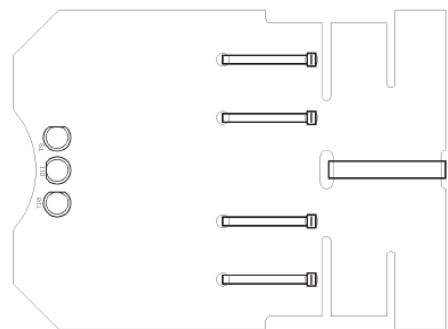
- IC1: Nejdříve vložte jenom patici (buď jednu patici DIP s 28 konektory, nebo dvě patice DIP14). Věnujte pozornost polaritě! Označení polarity na patici a symboly polarity na DPS jsou trochu asymetrické!
- IC3: Opět vložte jenom patici (DIP14). Věnujte pozornost polaritě! Označení polarity na patici a symboly polarity na DPS jsou trochu asymetrické!
- K1, K2, K3, K4, K5, K6: Senzorové spínače, které se musí namontovat plochou na povrch DPS!
- Q1: Rezonátor 8 MHz
- D1, D2, D3, D4, D5, D6, D7, D8: Diody 1N4148; věnujte pozornost polaritě!
- D9: N4001; věnujte pozornost polaritě!
- JP1: Propojka; Krátké kolíky, které se budou pájet. Zatím nepoužijte spojovací prvek propojky.
- D12: Dvoubarevná LED, průměr 3 mm, tři vývody; věnujte pozornost polaritě! (polarita se může lišit součástka od součástky, ale nejkratší vývod se musí vložit čtvercového pájeného podkladu)!
- C2, C3, C4, C5: 100 nF, keramika; Potisk: 104
- C6, C7: 4,7 nF, keramika; Potisk: 472
- T1, T3, T5, T7: BC327-40 nebo BC328-40
- T2, T4, T6, T8: BC337-40 nebo BC338-40

- R1, R2, R3, R4, R5, R6, R7, R8, R19, R21, R24: 1 k Ω (hnědý, černý, červený, zlatý)
- R9, R16: 220 Ω (červený, červený, hnědý, zlatý)
- R10, R17, R22, R31: 470 Ω (žlutý, fialový, hnědý, zlatý)
- R11: 100 Ω (hnědý, černý, hnědý, zlatý)
- R12: 12 k Ω (hnědý, červený, oranžový, zlatý)
- R13: 10 k Ω (hnědý, černý, oranžový, zlatý)
- R14, R15: 20 k Ω (červený, černý, oranžový, zlatý)
- R18, R20: 4,7 k Ω (žlutý, fialový, červený, zlatý)
- R23: 1 M Ω (hnědý, černý, zelený, zlatý)
- R25, R26, R32: 2 k Ω (červený, černý, černý, hnědý, hnědý)
- R27: 8,2 k Ω (šedý, červený, černý, hnědý, hnědý)
- R28: 16 k Ω (hnědý, modrý, černý, červený, hnědý)
- R29: 33 k Ω (oranžový, oranžový, černý, červený, hnědý)
- R30: 68 k Ω (modrý, šedý, černý, červený, hnědý)
- C1, C8: Elco 220_F 10 V nebo vyšší hodnoty; věnujte pozornost polaritě!
- IC2: Infrachervený přijímač-IC SFH5110-36, Vývody ohněte jehlovými kleštěmi! Věnujte pozornost polaritě (strana se zakřivením ve tvaru kupole musí být směřovat směrem ven)! Pozor: Součástka se může poškodit elektrostatickým výbojem (ESD) nebo teplem při pájení!
- D10: SFH 415-U IR-LED 5 mm; černé pouzdro; věnujte pozornost polaritě! Pouzdro se musí usadit vedle DPS.
- T11, T12: LPT80A, fototranzistor, bezbarvé pouzdro; věnujte pozornost polaritě! Pouzdro se musí usadit vedle DPS.
- D13, D14: IRL80A, IR-LED, růžové pouzdro; pouzdro se musí usadit vedle DPS. Věnujte pozornost polaritě!
- D15, D16: LED 5 mm, červené, růžové, resp. červené pouzdro. Věnujte pozornost polaritě! (krátký vývod se musí vložit na značku)!
- S1: Přepínač On/Off

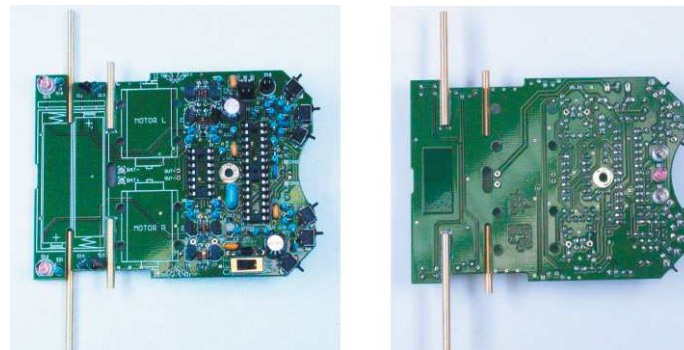
Budou potřebné ještě tři další součástky (které slouží pro sledování dráhy), ale budou se vkládat na spodní stranu DPS a musí se pájet na horní straně (viz obr. 4. 5).

- T9, T10: Fototranzistor SFH300 5 mm. Věnujte pozornost polaritě! Tyto komponenty se musí ukládat ve stejné vzdálenosti od DPS.
- D11: 5 mm LED, červená, červené nebo červenohnědé pouzdro; Věnujte pozornost polaritě! (krátký vývod se musí vložit na značku)!

Na obr. 4. 6 je přehled DPS se všemi do této chvíle vloženými součástkami při pohledu shora a zespodu.



Obr. 4. 5: Vkládání součástek na spodní stranu DPS-ASURO



Obr. 4. 6: Kompletně osazena horní a spodní strana DPS

Připojení motorů

Po dokončení vkládání součástek na DPS musíme připojit kabely k motorům a motory dočasně připojit, abychom provedli několik zkoušek.

Pro napájení motorů elektrickou energií budeme pro každou svorku potřebovat 70 mm červený a černý drát. Čtyři dráty určené pro dva motory se musí na obou koncích odizolovat asi v délce 4 mm. konce drátů musíte zkroutit a předběžně přiletovat nějakou pájkou. Přebytečné kousky pájky, které zůstanou na konci drátů, můžete uříznout štípacími kleštěmi. Pájením se připojí červený drát ke svorce motoru označené tečkou, nebo znakem plus. Černý drát se připojí k druhé svorce. Dráty motorů se mohou zakroutit (není to nezbytné, ale snižuje se tím citlivost na elektromagnetické rušení a vypadá to lépe...).

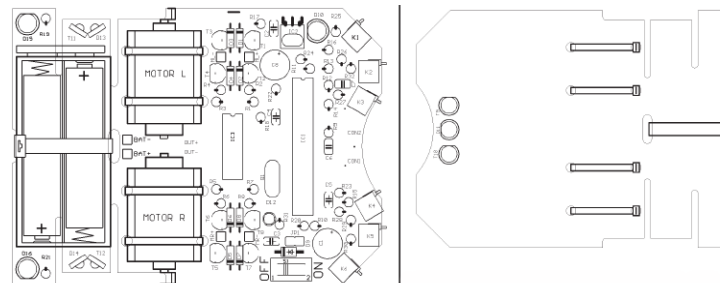
Červený drát motoru na levé straně se musí přiletovat k portu označenému jako "ML+", černý drát k portu "ML-", červený drát na pravé straně se musí přiletovat k portu "MR+" a černý drát k portu "MR-". Dále musíme motory dočasně připojit k DPS vazací páskou kabelů. Vazací pásky, které jsou součástí dodávky, prostrčte přes dva otvory. V této fázi bude postačovat, když upevníte jenom 2 vnitřní vazací pásky.

Napájení



Pokud se má ASURO vybavit a provozovat s bateriemi, musí se v každém případě otevřít propojka JP1! Jestliže se rozhodnete používat nabíjecí akumulátory, propojka se musí zavřít. Opacným uložením článků při zavřené propojce se zničí elektronické součástky!

Červený drát držáku baterií přileťte ke svorce "BAT+" a černý drát ke svorce "BAT-". Dávejte pozor, aby byl přepínač ON/OFF v poloze OFF. Čtyři napájecí články vložte při dodržení správné polaridy do schránky pro baterie. Nyní, nebo po dokončení zkoušek se schránka s bateriemi musí pomoci větší vazací pásky připevnit k otvoru v DPS.



Obr. 4. 7: Pohled na horní a spodní stranu DPS

Příprava k uvedení do provozu

Po dokončení montáže uvedeme robota do pohybu. Nejdříve však musíme najít a odstranit chyby, ke kterým mohlo dojít v předchozí fázi, aby se žádná součástka nepoškodila.

Infračervený vysílač RS232

Následující provozní zkouška se týká jenom infračerveného vysílače RS232 (obr. 4. 8). Nejdříve ze všeho se musí IR vysílač RS232 zkontrolovat, protože jej budeme potřebovat k dalšímu kroku, který představuje samočinný test systému. K této zkoušce připojte IR vysílač pomocí přiloženého 9 - pólového prodlužovacího kabelu k volnému sériovému portu PC a ve Windows otevřete program "Hyperterminal" (nebo "Minicom" v Linuxu). Za normálních okolností byste měli tento program najít v kategorii Addons --> Communication --> Hyperterminal. Není-li program dostupný, můžete jej nainstalovat z CD s Windows.

Po otevření programu Hyperterminal budete vyzváni k určení názvu připojení. Můžete si zvolit ASURO, nebo nějaký jiný název. V dalším okně vyberte "connect by" (způsob připojení) a COM rozhraní, ke kterému byl vysílač v předchozím kroku připojen.

Poté stiskněte "OK" a zvolte následující nastavení:

- Bits pro Second: 2400
- Databits: 8
- Parity: none
- Stopbits: 1
- Flowcontrol: none

Znovu stiskněte "OK" pro potvrzení nastavení.



Obr. 4. 8: IR vysílač RS232

Podržte IR vysílač ve vzdálenosti 10 cm nad listem bílého papíru. Strana s komponenty musí směřovat k papíru.

Stiskněte několik kláves na počítačovém terminálu.

Program terminálu by měl normálně zobrazovat symboly kláves. IR vysílač přenáší symboly kláves pomocí IR diody (D5), přenášený signál se odráží od povrchu papíru a směřuje zpět na přijímač integrovaného obvodu (IC2), z kterého se vrací zpět do počítače. Pokud se symboly nezobrazují, nebo se zobrazují špatné symboly, opatrně otočte trimrem v rozmezí jeho krajních poloh. Miniaturním šroubovákem poklepejte v každé poloze trimru na několik kláves, dokud se nezobrazí správné symboly.

Pokud se vám operace nedaří, máme problém s obvodem, který se musí vyřešit (viz níže "Závada na IR vysílači RS232").

Po této zkoušce by se měl IR vysílač odstranit a znovu poklepejte na několik kláves. Teď by se na displeji už neměly zobrazovat žádné symboly.

Infračervený USB vysílač

Tato operace se týká jenom USB IR vysílače.

POZOR! USB IR vysílač nemá žádné pouzdro a proto je velmi citlivý na elektrostatické výboje.

Předtím než jej začnete používat, vybijte se dotykem kovové schránky počítače, nebo jiného zemnicího bodu. Další možností je zabudovat IR vysílač do průhledného pouzdra, aby byl chráněn.

Windows

Následující provozní zkouška se vztahuje jen na USB IR vysílač.

Nejdříve se musí IR vysílač zkontrolovat, protože jej budeme potřebovat k dalšímu kroku, který představuje samočinný test systému. K této zkoušce připojte IR vysílač pomocí USB kabelu k volnému USB portu na PC.

Objeví se zpráva, že byl nalezen nový hardware: "NEW HARDWARE WAS FOUND": **AREXX ASURO USB-IR-TRANSCIEVER**.

Nyní můžete z ASURO CD nainstalovat USB ovladač. Pokud nebude ovladač nalezen automaticky, vyberte jej na CD manuálně z CD\windows\USB Driver, (k této operaci budete potřebovat oprávnění správce systému). Po instalaci ovladače můžete přistoupit k USB vysílači jako k běžnému sériovému portu.

Po otevření programu Hyperterminal budete vyzváni k určení názvu připojení. Můžete si zvolit ASURO USB, nebo nějaký jiný název. V dalším okně vyberte "connect by" (způsob připojení) a COM rozhraní, ke kterému byl vysílač v předchozím kroku připojen.

Poté stiskněte "OK" a zvolte následující nastavení:

- Bits pro Second: 2400
- Databits: 8
- Parity: none
- Stopbits: 1
- Flowcontrol: none

Znovu stiskněte "OK" pro potvrzení nastavení.



Podržte IR vysílač ve vzdálenosti 10 cm nad listem bílého papíru. Strana s komponenty musí směřovat k papíru.

Stiskněte několik kláves na počítačovém terminálu.

Program terminálu by měl normálně zobrazovat symboly kláves. IR vysílač přenáší symboly kláves pomocí IR diody (D5), přenášený signál se odráží od povrchu papíru a směřuje zpět na přijímač integrovaného obvodu (IC2), z kterého se vrací zpět do počítače. Pokud se symboly nezobrazují, nebo se zobrazují špatné symboly, opatrně otočte trimrem v rozmezí jeho krajních poloh. Miniaturním šroubovákem udeřte v každé poloze trimru na několik kláves, dokud se nezobrazí správné symboly. Pokud se vám operace nedaří, máme problém s obvodem, který se musí vyřešit (viz níže "Závada na IR vysílači RS232").

Linux

Následující provozní zkouška se vztahuje jen na USB IR vysílač v prostředí Linux.

Nejdříve se musí IR vysílač zkontrolovat, protože jej budeme potřebovat k dalšímu kroku, který představuje samočinný test systému. K této zkoušce připojte IR vysílač pomocí USB kabelu k volnému USB portu na PC. Krátké pípnutí potvrdí, že Linux detekoval IR vysílač. Pro jistotu zkontrolujte, jestli se v proc-declaration zobrazuje následující zpráva:

```
foo@bar:~>cat /proc/tty/driver/usb-serial
```

Je tam také následující zadání ("0" v našem příkladě může být nahrazena "1" nebo "2"):

```
usbserinfo:1.0 driver:v1.4  
0: module:ftdi_sio name:"FTDI 8U232AM Compatible" vendor:0403 product:6001  
num_ports:1 port:1 path:usb-00:11.2-1
```

Ke zkušebnímu účelům musíte nakonfigurovat rozhraní Minicom /dev/ttyUSB0 (nebo 1, 2 usw...)

podle následujícího nastavení:

- Bits pro Second: 2400
- Databits: 8
- Parity: none
- Stopbits: 1
- Flowcontrol: none

Znovu stiskněte "OK" pro potvrzení nastavení.

Je možné, že budete potřebovat kořenová práva.

Možná že k novému zařízení budete muset deklarovat práva pro zápis a čtení pro uživatele, nebo skupinu dev/ttyUSB?. Můžete tak učinit pomocí chmod u+rw /dev/ttyUSB0 (nebo 1, 2...), nebo chmod g+rw /dev/ttyUSB0 (opět budete samozřejmě potřebovat právo správce.

Podržte IR vysílač ve vzdálenosti 10 cm nad listem bílého papíru. Strana s komponenty musí směřovat k papíru.

Stiskněte několik kláves na počítačovém terminálu.

Program terminálu by měl normálně zobrazovat symboly kláves. IR vysílač přenáší symboly kláves pomocí IR LED (D5), přenášený signál se odráží od povrchu papíru a směřuje zpět na přijímač integrovaného obvodu (IC2), z kterého se vrací zpět do počítače. Pokud se symboly nezobrazují, nebo se zobrazují špatné symboly, opatrně otočte trimrem v rozmezí jeho krajních poloh. Miniaturním šroubovákem udeřte v každé poloze trimru na několik kláves, dokud se nezobrazí správné symboly.

Provoz robota ASURO



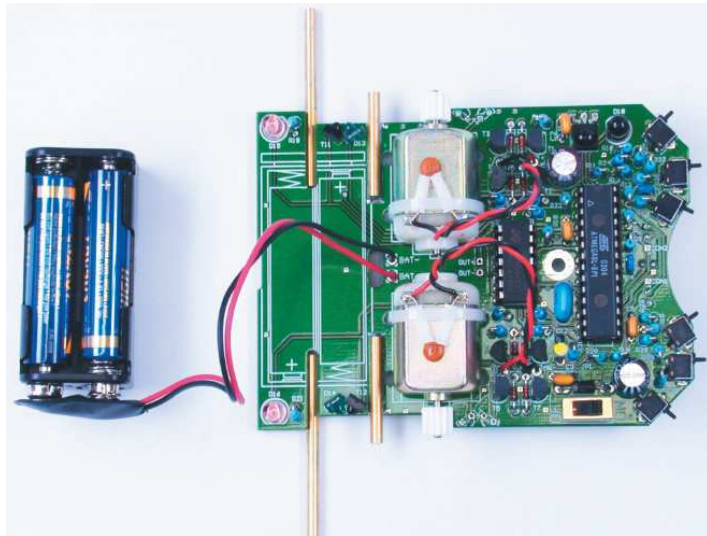
Nevkládejte zatím procesor (IC1)!

Nyní budete opatrní a přepínač ON/OFF dejte do polohy ON. Obě zadní diody (D15, D16) by se měly slabě rozsvítit. V opačném případě systém okamžitě vypněte a pokračujte ve čtení níže uvedené části "Zadní diody se při spuštění nerozsvítí". Pokud se diody rozsvítily, vypněte systém a vložte IC1 (procesor) a IC3 (AND Gate), viz obr. 5. 1.

Někdy je potřebné vývody IC opatrně ohnout, aby je bylo možné vložit do kontaktních otvorů na patiči. To lze snadno provést, pokud vezmete IC a jednu stranu s vývody zatlačíte vodorovně na rovný povrch stolu.



Jak Procesor IC1 ATmega8, tak Gate IC3 CD4081 jsou elektrostaticky citlivé součástky. Mohou se zničit, pokud se jich dotknete, aniž jste se zbavili elektrostatické energie, kterou lehce získáte např. chozením po nevodivé podlaze. Předtím než začnete s těmito součástkami pracovat, není na škodu natáhnout si elektrostatický náramek, nebo se alespoň dotknout kovové skříň uzemněného zařízení, nebo radiátoru.



Obr. 5. 1: ASURO po instalaci IC

Nyní vložte propojku (J1) pro nabíjecí akumulátory. Značky polarity na IC se musí zarovnat se značkami na patičích a na DPS. Procesor při spuštění provede samočinný test a zkontroluje všechny komponenty. Abyste se vyhnuli těžkostem, doporučujeme přečíst si nejdříve následující část návodu a až poté systém zapnout a vrátit se k této části. V této chvíli začneme s testováním. Zapněte robota a odeď z robota nespouštějte oči.



Pokud je aktivní program Hyperterminal (Windows), nebo Minicom (Linux) a mezi ASURO a IR vysílačem je viditelný kontakt, můžete samočinný test sledovat na obrazovce počítače.

Světelné Indikátory

LED kontrolka stavu (D12) se krátce rozsvítí oranžově a budou také svítit zadní LED (D15, D16), i když jen matně. Pokud se některá z LED kontrolky nerozsvítí, vypněte robota a začněte hledat příčinu chyby (viz níže "LED kontrolka stavu se po startu nerozsvítí"). Právě jste viděli bootovací sekvenci robota. V další fázi se postupně po 3 sekundách zkontrolují všechny ostatní zobrazovací prvky v následujícím pořadí:

- LED kontrolka stavu (D12) zelená
- LED kontrolka stavu (D12) červená
- Přední LED (D11) ve spodní části robota
- Zadní LED (D15) vlevo
- Zadní LED (D16) vpravo
- Všechny indikátory současně

Objeví-li se chyba, robota okamžitě vypněte a začněte hledat příčinu chyby (viz níže "Světelné indikátory nepracují"), protože všechny světelné indikátory hrají důležitou roli v následujících testech.

Fototranzistory (T9, T10)

Po dokončení testů by měla LED kontrolka stavu (D12) svítit zeleně a jasně indikovat, že se nyní asi 10 sekund testují fototranzistory na spodní straně robota, které jsou zodpovědné za sledování dráhy. Zatímco jsou fototranzistory (T9, T10) osvětleny, zapnou se příslušné zadní LED (D15, D16) a ztlumí se, jakmile se světlo vypne (pravý fototranzistor (T10) > pravá zadní LED (D16); levý fototranzistor (T9) > levá zadní LED (D15)). Všechno je v pořádku, pokud se zadní LED po odstranění světla nevypínají úplně, ale stále tlumeně svítí. Samočinný test může pokračovat, i když se objeví chyba a její odstranění se může odložit.

Spínače

ASURO je nehybný a světelné indikátory jsou tmavé. To je dobré znamení! Nyní jsme připraveni asi 15 sekund kontrolovat tlačítka. Jednoduše tlačítka stisknete a sledujte, jestli se něco děje.

Výsledek by měl vypadat následovně:

- K1 > Stavová LED (D12) se rozsvítí zeleně
- K2 > Stavová LED (D12) se rozsvítí červeně
- K3 > Zapne se přední LED na spodní straně
- K4 > Zadní LED vlevo (D15)
- K5 > Zadní LED vpravo (D16)
- K6 > Začne běžet levý motor (v opačném případě může pokračovat samočinný test).

Motory se zkontrolují později a chyby v těchto prvcích lze odstranit pomocí testů, které jsou popsány níže v části "Jeden motor se nepohybuje".

Stisknutím několika tlačítek se aktivuje příslušná kombinace signálů. V případě chyby může samočinný test pokračovat a odstranění chyby lze odložit.

Odometer

Nyní začne svítit LED sledování trasy (D11). Signál indikuje začátek dalšího testu (15 sekund), během kterého se zkontrolují oba odometry (každý z nich se skládá z LED a z fototranzistoru). Přidržení listu bílého papíru před senzory způsobí rozsvícení stavové LED. Nejdříve podržte list papíru před levým senzorem (T11) > rozsvítí se zelená LED (D12). Poté podržte list papíru před pravým senzorem (T12) > rozsvítí se červená LED (D12). Odstranění papíru bude mít za následek vypnutí stavové LED. Odometry jsou v pořádku, pokud jsou aktivní meze on/off na levé a na pravé straně. V případě chyby může samočinný test pokračovat a odstranění chyby lze odložit.

Motor

Obě zadní LED (D15, D16) jsou zapnuty a signalizují poslední test, který trvá asi 15 sekund. Kontrola motoru je intenzivní. Levý motor se nastartuje směrem dopředu z nulové rychlosti na maximální rychlost a přejde zase zpět na nulovou rychlost. Poté se směr otočí a rychlost bude akcelarovat z nuly na maximum a znovu zpět na nulu. Motor na pravé straně se zkontroluje stejným způsobem. Po kontrole jednotlivých motorů se provede kontrola obou motorů současně. Ještě jednou opakujeme naši poznámku: V případě chyby může samočinný test pokračovat a odstranění chyby lze odložit.

IR rozhraní

Pokud stavová LED svítí přerušovaným žlutým světlem, začal poslední test (asi 15 sekund). V průběhu tohoto testu IR vysílač odesílá a přijímá data. Aby mohla být tato data přijata, musí se dříve nainstalovaný IR vysílač propojit s PC a k danému účelu se může použít terminálový program, např. "Hyperterminal" ve Windows. Konfigurace bude stejná, jako při testu IR vysílače. Při příjmu symbolů ASURO odpoví symbolem, který následuje v abecedě. Pokud v rámci pevně daného časového limitu nedojde k příjmu symbolů, ASURO vyšle "T". Jakýmkoliv odeslaným symbolem se zapne červená stavová LED souběžně se zelenou stavovou LED, což má za následek blikající oranžové světlo.

Pokud je mezi robotem a IR vysílačem optické propojení (k čemuž je potřebné, aby byly navzájem ve vzdálenosti max. 50 cm), v terminálovém programu se ukáže symbol "T" (podle tlačítka, které jste aktivovali na klávesnici PC a který byl přenesen vysílačem), po němž následuje symbol, který je další v abecedě, např.

Je aktivována klávesa "e" → terminálový program hlásí "ef"

Je aktivována klávesa "j" → terminálový program hlásí "jk"

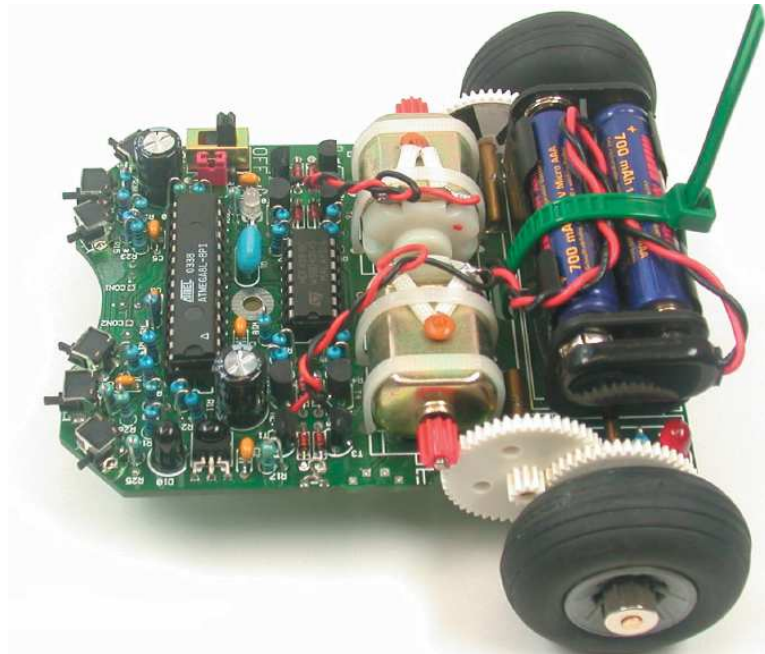
Je aktivována klávesa "3" → terminálový program hlásí "34"

V případě chyby viz níže část "IR rozhraní".

Hotovo?

Pokud se objeví chyba, vypněte systém a odstraňte baterie. Najděte příčinu a problém vyřešte podle níže uvedeného návodu k řešení problémů. Zopakujte automatický test, abyste se přesvědčili, že nedošlo k poškození žádné součástky. Pouze kompletní test bez chyb může zaručit, že hardware robota je v pořádku a že se v softwaru nevyskytnou žádné další chyby.

Když musíte zkontrolovat hardware poté, co jste instalovali do robota další programy, musíte z CD znovu načíst program "SelfTest.hex".



Kompletně složený ASURO

Řešení problémů

Závada na IR vysílači RS232

Aktivovaná klávesa a zobrazený symbol si navzájem neodpovídají

Kalibrujte trimr TR1, dokud nebude zobrazený symbol odpovídat aktivované klávese.

V terminálovém programu se nezobrazuje žádný symbol

Vložili jste IC1 a byl vložen se správnou polaritou (značka musí směřovat k třem diodám)? Vezměte si infračervený dálkový ovladač nějakého hifi, nebo video zařízení (videorekordér, TV, atd.), nasměrujte jej na IR vysílač a stiskněte nějaká tlačítka. Pokud se v terminálovém programu ukáží nestandardní symboly, přijímač ((IC2, R3, C4, D4, T1) pracuje. Musí se zkontrolovat všechny ostatní komponenty.

Robot stále nepracuje

Zkontrolujte správnou polaritu všech součástek a správné nastavení hodnot (viz výše "Sestavování elektroniky"). Zkontrolujte, jestli nedošlo ke zkratování pájených spojů a jestli jsou spoje v pořádku. Nedošlo k narušení pájených spojů? Pokud kontrola nepřinesla žádná zjištění, musí se chyba zjistit pomocí schématu (viz příloha B) a patřičného měřicího zařízení (multimetr, nebo osciloskop). Ve většině případů mohlo dojít k poškození jednoho z komponentů IC1, IC2, Q1, nebo D4.

Nepracuje USB IR vysílač

Windows

Zkontrolujte, jestli jsou správně nainstalovány ovladače a jestli byl vybrán správný com port.

Linux

Odpojte USB vysílač, několik minut počkejte a znova ho připojte. Problém se tím může vyřešit. Další možností je nainstalovat nové jádro.

Zadní LED (D15, D16) se při startu nerozsvítí!

Žádná ze zadních LED se nerozsvěcuje

Podívejte se na světla velmi pozorně v tmavé místnosti. Jestliže nevidíte žádné světlo, zkontrolujte následující kroky:

- Jsou vloženy 4 baterie? Nejsou baterie vybité, resp. byly akumulátory znovu nabitě?
- Připojili jste správně kabely baterií? (červený k Bat+ a černý k Bat-)
- Byla dioda D9 (1N4001) vložena se správnou polaritou?
- Používáte správné hodnoty u R22? 470 Ω (žlutý, fialový, hnědý, zlatý)
- Zkontrolujte také hodnoty R18, R19, R20, R21
4,7 kΩ (žlutý, fialový, červený, zlatý)
1 kΩ (hnědý, černý, červený, zlatý)

Svítil jenom jedna ze zadních LED

Vložili jste diody (růžovou a červenou) D13 (vlevo) a D14 (vpravo) a fototranzistory (průhledné, nebo černé pouzdro) T11 (vlevo), T12 (vpravo) na správné místo a se správnou polarizací?

Zkontrolujte hodnoty rezistorů R18, R19 (vlevo) a R20, R21 (vpravo):

4,7 kΩ (žlutý, fialový, červený, zlatý)

1 kΩ (hnědý, černý, červený, zlatý)

Byly tyto součástky vloženy na správné místo? (zkontrolujte potisk na DPS!).

Stavová LED (D12) se po startu nerozsvítí

Stavová LED se vůbec neaktivuje > viz níže!

Stavová LED bliká. Baterie jsou slabé; vyměňte je.

Pokud jsou baterie v pořádku, zkontrolujte rezistory R12 a R13.

12 kΩ (hnědý, červený, černý, červený, hnědý)

10 kΩ (hnědý, černý, černý, červený, hnědý)

Nefunguje světelná indikace

Byl správně vložen procesor? (Polarita!)

Nefunguje stavová LED D12

Zkontrolujte polarizaci LED D12.
Zkontrolujte rezistory R10, R31.
470 Ω (žlutý, fialový, hnědý, zlatý)

Jednoduchou kontrolu můžete provést následujícím způsobem: odstraňte procesor (IC1) a propojte drátem Pin7 (VCC) a Pin14 (aktivuje se zelená stavová LED), respektive Pin4 (stavová LED bude svítit červeně). Když je tato zkouška úspěšná, 1) buď může být vadný procesor, nebo 2) vodivá plocha na DPS může být rozbita / přerušena.

Nepracuje přední LED D11

Zkontrolujte polarizaci LED D11.
Zkontrolujte rezistor R9.
220 Ω (červený, červený, hnědý, zlatý)

Jednoduchou kontrolu můžete provést následujícím způsobem: odstraňte procesor (IC1) a propojte drátem Pin7 (VCC) a Pin12 (aktivuje se červená přední LED). Když je tato zkouška úspěšná, 1) buď může být vadný procesor, nebo 2) vodivá plocha na DPS může být rozbita / přerušena.

Nepracuje levá zadní LED D15

Zkontrolujte polarizaci LED D15.
Zkontrolujte rezistory R19, R18
1 KΩ (hnědý, černý, červený, zlatý)
4,7 KΩ (žlutý, fialový, červený, zlatý)

Jednoduchou kontrolu můžete provést následujícím způsobem: odstraňte procesor (IC1) a propojte drátem Pin7 (VCC) a Pin24 (aktivuje se červená levá zadní LED). Když je tato zkouška úspěšná, 1) buď může být vadný procesor, nebo 2) vodivá plocha na DPS může být rozbita / přerušena.

Nepracuje pravá zadní LED D16

Zkontrolujte polarizaci LED D16.
Zkontrolujte rezistory R21, R20.
1 KΩ (hnědý, černý, červený, zlatý)
4,7 KΩ (žlutý, fialový, červený, zlatý)

Jednoduchou kontrolu můžete provést následujícím způsobem: odstraňte procesor (IC1) a propojte drátem Pin7 (VCC) a Pin23 (aktivuje se červená pravá zadní LED). Když je tato zkouška úspěšná, 1) buď může být vadný procesor, nebo 2) vodivá plocha na DPS může být rozbita / přerušena.

Nepracuje senzor sledování dráhy (T9, T10)

Zkontrolujte polarizaci T9, T10.
Zkontrolujte rezistory R14, R15.
20 KΩ (červený, černý, oranžový, zlatý)

Zkontrolujte, jestli jsou rezistory R15, R23 a R28 vloženy na správné místo!
Odstraňte procesor a zkontrolujte signál senzoru na Pin25, respektive na Pin26 (tmavý ≈ 0 V, světlý ≈ VCC).

Nefunguje přepínač

Pokud se aktivuje kombinace tlačítek

Zkontrolujte R12 a R13
12 kΩ (hnědý, červený, černý, červený, hnědý)
10 kΩ (hnědý, černý, černý, červený, hnědý)

Zkontrolujte také R24, R25, R26, R27, R28, R29, R30, R32!

1 KΩ (hnědý, černý, černý, hnědý, hnědý)
2 KΩ (červený černý, černý, hnědý, hnědý)
8,2 KΩ (šedý, červený, černý, hnědý, hnědý)
16 KΩ (hnědý, modrý, černý, červený, hnědý)
33 KΩ (oranžový, oranžový, černý, červený, hnědý)
68 KΩ (modrý, šedý, černý, červený, hnědý)
2 KΩ (hnědý, černý, černý, hnědý, hnědý)

Světelná indikace reaguje, jako kdyby byly přepínače zaměněny

Byly zaměněny rezistory přepínačů.
Zkontrolujte R24, R25, R26, R27, R28, R29, R30, R32!
1 KΩ (hnědý, černý, černý, hnědý, hnědý)
2 KΩ (červený černý, černý, hnědý, hnědý)
8,2 KΩ (šedý, červený, černý, hnědý, hnědý)
16 KΩ (hnědý, modrý, černý, červený, hnědý)
33 KΩ (oranžový, oranžový, černý, červený, hnědý)
68 KΩ (modrý, šedý, černý, červený, hnědý)

Věci stále nefungují, jak mají

Zkontrolujte R23, R24 a také R12, R13 a C7!
1 MΩ (hnědý, černý, zelený, zlatý)
1 KΩ (hnědý, černý, černý, hnědý, hnědý)
12 kΩ (hnědý, červený, černý, červený, hnědý)
10 kΩ (hnědý, černý, černý, červený, hnědý)
220 μF/10V

Nepracuje odometer (senzor odráženého světla)

Nefunguje ani jeden o odometrů

Zkontrolujte rezistor R22!
470 Ω (žlutý, fialový, hnědý, oranžový)
Zkontrolujte rotaci D13 a D14.
D13 a D14 jsou růžové, nebo šedě zbarvení bipolární komponenty s malou tečkou na jedné straně. Tečka musí směřovat ven z DPS.

Nepracuje levý odometer

Zkontrolujte rezistor R18!
4,7 KΩ (žlutý, fialový, červený, zlatý)
Zkontrolujte rotaci T12, což je průsvitný, nebo černý bipolární komponent s malou tečkou na jedné straně. Tečka musí směřovat ven z DPS.

Nepracuje pravý odometer

Zkontrolujte rezistor R20!
4,7 KΩ (žlutý, fialový, červený, zlatý)
Zkontrolujte rotaci T12, což je průsvitný, nebo černý bipolární komponent s malou tečkou na jedné straně. Tečka musí směřovat ven z DPS. Odstraňte procesor a zkontrolujte signál senzoru na Pin24, respektive na Pin23 (tmavý ≈ VCC, světlý ≈ 0 V).

Jeden motor se nepohybuje

Ani jeden motor se nepohybuje

Zkontrolujte polaritu a polohu IC3.

Levý motor se nehýbe, nebo se pohybuje jen jedním směrem

Zkontrolujte kompletní přemostění motoru, které se skládá z tranzistorů T1, T2, T3, T4 (vložili jste správné tranzistory na správná místa?), diod D1, D2, D3, D4 (polarita!) a rezistorů R1, R2, R3, R4. BC327-40 nebo BC328-40, BC337-40 nebo BC338-40, BC327-40 nebo BC328-40, BC337-40 nebo BC338-40

1 K Ω (hnědý, černý, červený, zlatý)

Jeden motor se točí opačným směrem

Zkontrolujte kabely, které připojují motor k systému. Tyto spoje by se měly zaměnit.

IR rozhraní

ASURO neodesílá symboly

Zkontrolujte polaritu IR diody D10.

Zkontrolujte rezistor R16 220 Ω (červený, červený, hnědý, zlatý)

100 nF (potisk: 104)

Pokud jste zatím nenašli žádnou chybu, podívejte se na pájený spoj IC2. Tato součástka je citlivá na přehřátí a mohla se během pájení poškodit. V takovém případě komponent vyměňte za nový IC (SFH 5110-36). Jestliže dochází opakovaně k selháním v přenosu dat mezi PC a ASURO, upravte trimr TR1 ve vysílači.

Věci stále nefungují, jak mají

Zkontrolujte polaritu C8.

220 μ F/ min. 10V

Pokud dochází opakovaně k selháním v přenosu dat mezi PC a ASURO, upravte trimr TR1 ve vysílači.



Instantní a dvousložková lepidla mohou při styku s pokožkou způsobit podráždění, nebo vyvolat alergické reakce. Zabraňte proto kontaktu pokožky s lepidlem. Chraňte si ruce vinylovými rukavicemi. Pokud dojde ke kontaktu s lepidlem, zasažené místo okamžitě pořádně omyjte mýdlem a vodou! Instantní lepidla byla původně vyvinuta pro chirurgii, což si můžete všimnout, když se vám dva prsty během několika sekund přilepí k sobě. Když se vám skutečně přilepí dva prsty k sobě instantním lepidlem, můžete je uvolnit pomocí teplé vody mýdla a trochou trpělivosti. Dávejte velký pozor, aby se vám lepidlo nedostalo na rty nebo do očí. Při práci s lepidlem potlačte všechny reflexní pohyby a nevytírejte si rukou oči ani tvář!



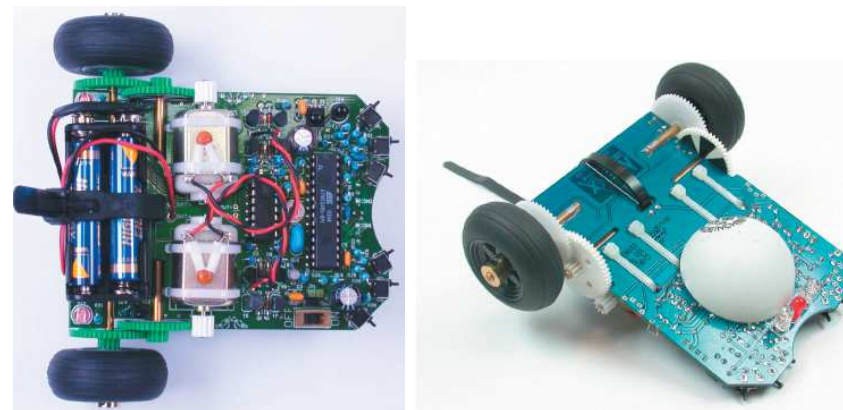
Finální úpravy

Nápravy lehce namažte. Ozubená kola s černými a bílými značkami umístěte na krátké nápravy. Pneumatiky s ozubenými koly se nyní umístí na dlouhé nápravy a zajistí se zajišťovacím kroužkem. Opatrně pohybujte motorem, dokud si navzájem správně nesednou ozubená kola a převodový systém nebude hladce běžet.

(Proveďte kompletní automatický test, abyste se ujistili, že motory a převody jsou v pořádku.)

Když jste zkontrolovali polohu motoru, držte celý systém pevně pohromadě a mezi motor a DPS nakapejte malé množství instantního lepidla. Vezměte do úvahy, že instantní lepidlo vyžaduje několik sekund na zaschnutí. Po upevnění instantním lepidlem použijte na vnější straně DPS vázací pásky na kabely.

Rozpúlený pingpongový míček se rovněž přilepí instantním lepidlem na dvou protilehlých místech na spodní stranu DPS, hned za komponenty sledování dráhy pohybu (viz obr. 7. 1). Opět myslíte na to, že instantní lepidlo bude vyžadovat několik sekund na zaschnutí.



Obr. 7. 1: Kompletně sestavený robot ASURO

Instalace softwaru a první kroky

Vložte do počítače ASURO CD. Pokud je všechno v pořádku, CD se spustí automaticky.

V opačném případě otevřete průzkumníka Windows. Po výběru jazyka najdete všechny potřebné programy v menu softwaru. Než s nimi budete moci pracovat, musíte si nejdříve nainstalovat program na svůj pevný disk. Budete k tomu potřebovat práva správce.

Jestliže nejste přihlášen jako správce, odhlaste se a znovu se přihlaste do systému jako správce.

V průběhu instalace softwaru uděláme následující:

1. Nainstalujeme program Flash-Tool pro přenos vašeho vlastního programu na ASURO.
2. Nainstalujeme editor programu (Programmers Notepad 2, PN2) a kompilátor (WinAVR).
3. Zkopírujeme ukázkové programy z CD na pevný disk.
4. V editoru PN2 vytvoříme menu pro vložení souborů Make a Clean.

Windows


FLASH Tool

Zkopírujte program do složky na pevný disk, např. C:\programs\flash. Můžete jej však spustit i přímo z CD. V každém případě je však rozumné, vytvořit si zástupce programu flash na ploše.



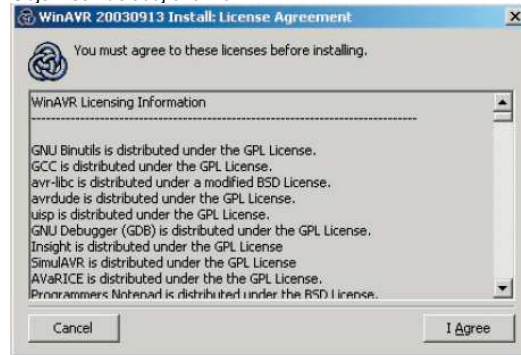
Instalace editoru programu a kompilátoru

K instalaci kompilátoru musíte mít oprávnění správce systému na svém počítači (během instalace dojde k změně položek v registru). Pokud nejste přihlášen jako správce, odhlaste se a znovu se přihlaste do systému jako správce!

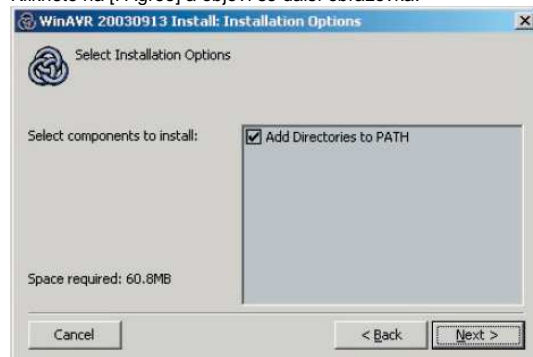
Kliknutím na symbol instalace spustíte instalaci.  **COMPILER WinAVR (20030913).**

Když se po kliknutí nic nestane, otevřete CD pomocí průzkumníka Windows.

Objeví se následující okno:

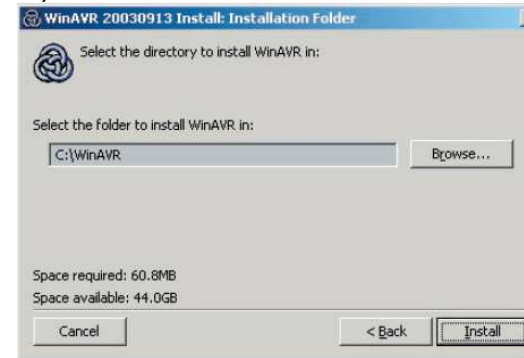


Klikněte na [I Agree] a objeví se další obrazovka:

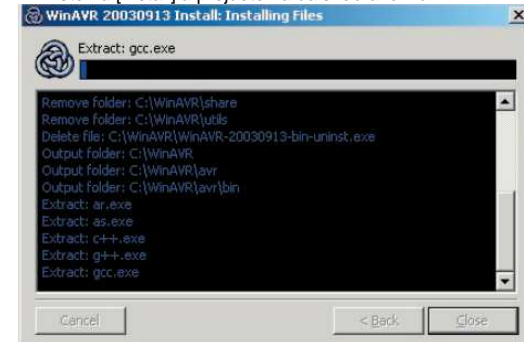


Klikněte na [Next].

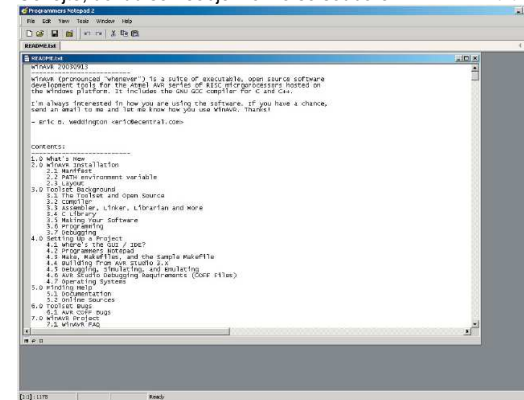
Objeví se další obrazovka:



Klikněte na [Install] a přejdete na další obrazovku:



Čekajte, dokud se neobjeví okno se souborem README.txt editorem Notepad 2 (PN2).



Zavřete obrazovku "programmers notepad 2".



Na ploše se objeví ikona "programmers notepad 2". Editor programu a kompilátor jsou nyní nainstalovány.

Kopírování ukázkových programů z CD na pevný disk

Zkopírujte složku "ASURO_src" z CD na pevný disk (vlozte ji do nějaké složky, jako např. "C:\ASURO_src").

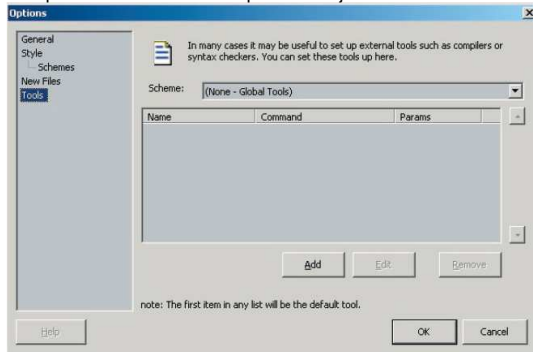
Pokud je datový zdroj zabezpečen, klikněte pravým tlačítkem myši na soubor, přejděte na vlastnosti a zabezpečení deaktivujte.

V editoru programu vytvořte vstupní menu pro kompilátor.

Otevřete "Programmers Notepad 2" (dvakrát klikněte na ikonu programu, která je na ploše).

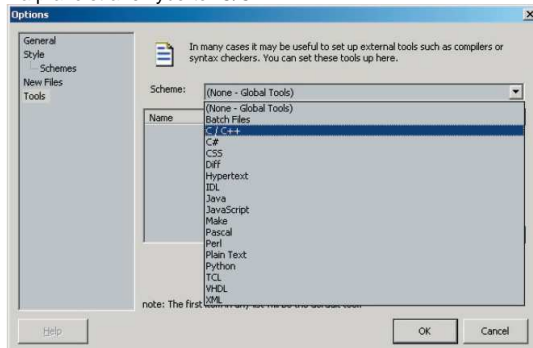


Pod položkou Tools zvolte Options a objeví se obrazovka možností:

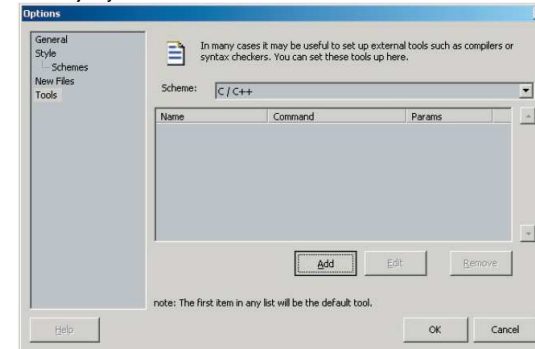


Vyberte Tools.

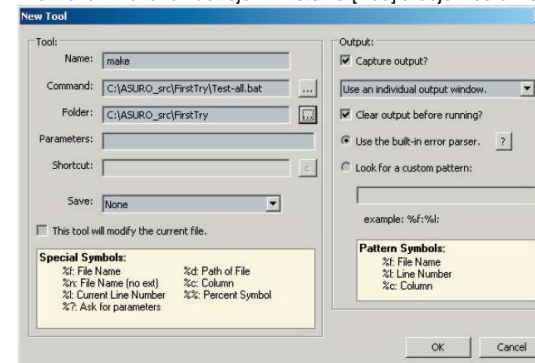
Na pravé straně vyberte "C/C++".



C/C++ je vybráno.



Pro vložení nového nástroje klikněte na [Add] a objeví se okno "New Tool".



Zadejte nebo pomocí tlačítka procházení vyberte následující text:

Name: make

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

Folder: C:\ASURO_src\FirstTry

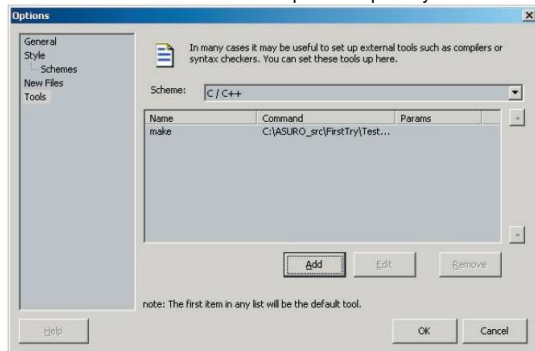
Klikněte na [OK].

V hlavním menu nástrojů (tools) je nyní dostupný nový nástroj PN2 s názvem "make".

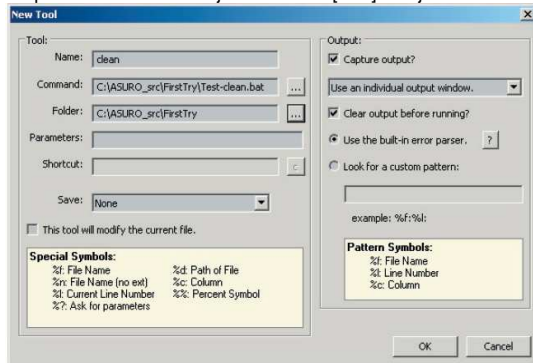
(když tento nástroj aktivujeme, spustí se soubor batch s názvem Test-all.bat a kompilovaná data se budou ukládat do složky C:\ASURO_src\FirstTry).

V menu editoru programu vytvoříte soubor "Clean".

V hlavní nabídce "Tools" zvolíte "Options" a poté vyberte znovu "C/C++":



Pro přidání nového nástroje klikněte na [Add] a objeví se okno nového nástroje "New Tool":



Zadejte nebo pomocí tlačítka procházení ... vyberte následující text:

Name: clean

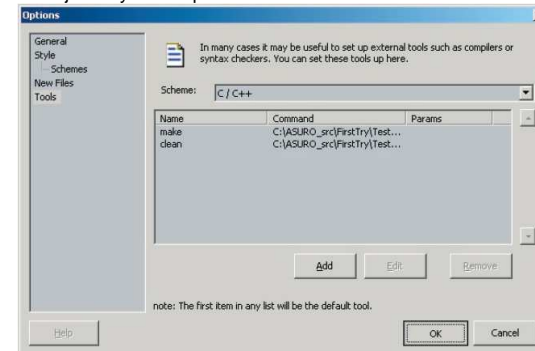
Command: C:\ASURO_src\FirstTry\Test-clean.bat

Folder: C:\ASURO_src\FirstTry

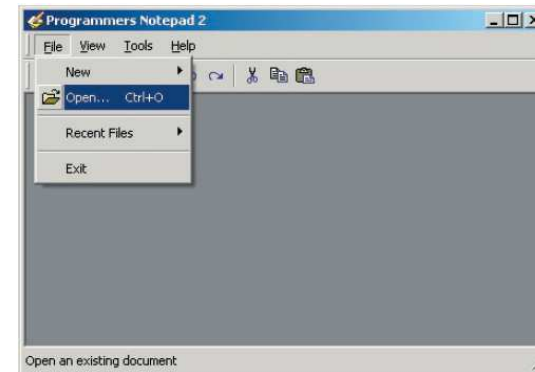
Klikněte na [OK]

V hlavní nabídce nástrojů je nyní dostupný nový nástroj PN2 s názvem "clean". (když tento nástroj aktivujeme, spustí se soubor batch s názvem Test-clean.bat a kompilovaná data, která byla ve složce C:\ASURO_src\FirstTry) se nyní odstraní.

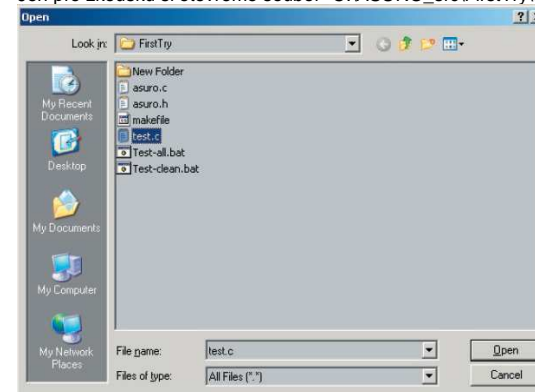
Pod položkou nástrojů najdete na obrazovce možnosti soubory make a clean, které jsme vytvořili v předchozích krocích.



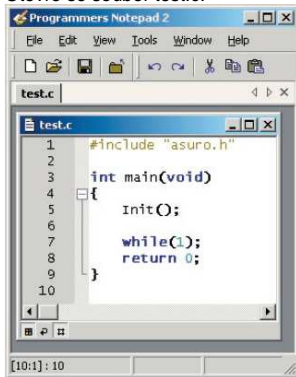
Klikněte na [OK].



Jen pro zkoušku si otevřeme soubor "C:\ASURO_src\FirstTry\test.c":

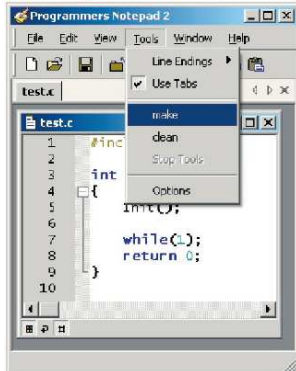


Klikněte na [Open].
Otevře se soubor test.c.

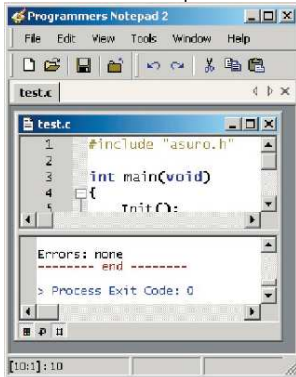


```
1 #include "asuro.h"
2
3 int main(void)
4 {
5     Init();
6
7     while(1);
8     return 0;
9 }
10
```

Když kliknete na Tools, najdete v liště nabídky nové nástroje make a clean.



Klikněte na "make" a provede se kompilace test.c (společně s asuro.c).



Pokud program neobsahuje žádné chyby (co jiného můžeme očekávat, když jsme načeti ukázkový program), objeví se ve spodní části zpráva: Errors: none.
K čemu došlo?

Ze souboru test.c (a asuro.c) se vygeneroval nový soubor test.hex, který obsahuje konvertovaný program ve strojovém kódu. Program ve strojovém kódu lze načíst do paměti robota ASURO. Tento program nemá žádnou funkci, ale později jej z cvičných důvodů načteme pomocí nástroje Flash do paměti robota.

Jak to funguje?

Soubor vstupního menu aktivuje dávkový batch soubor Test-all.bat (tento batch soubor obsahuje seznam příkazových řádků, které se provádějí řádek po řádku).

V Test-all.bat se provede příkaz "make all". "make" vytvoří soubor make, který se umístí (když programujeme robota) do stejného souboru jako Test-all.bat.

Soubor make je textový soubor, který určuje, jak kompilovat jeden, nebo více programů.

V průběhu programování, pokud se programuje jen jeden program, si můžete v souboru udržovat celkem dobrý přehled. Později, když se píše komplexnější systém a programová data obsahují víc souborů, které se musí správným způsobem krok za krokem konvertovat a také navzájem správně propojit, potom bude soubor make velmi složitý.

Příkaz "all" pro všechny vstupy v souboru make znamená, že se bude konvertovat celý projekt a ne jenom jednotlivé vstupy.

Soubor make v našem příkladu je napsán takovým způsobem, že soubor s názvem test.c se zkompiluje s asuro.c (který obsahuje některé předdefinované funkce) a vytvoří se nový soubor s koncovkou .hex. Tento soubor se pak načte (přenes) do paměti robota.

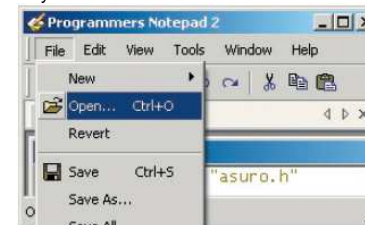
To znamená, že pokud nezměníte soubor make, ale jej pouze kopírujete, měli byste svůj vlastní program pojmenovat test.c.

Pokud chcete vědět více o souborech make (k obsluze robota ASURO to vůbec nepotřebujete), můžete najít podrobnější informace na adrese: <http://www.gnu.org/directory/make.html>

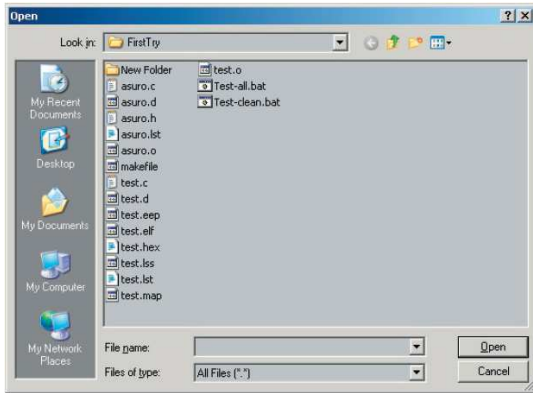
Základy programování robota vysvětlíme níže v části "Programování robota ASURO v jazyku C".

Při kompilaci programu se vygeneruje mnoho dočasných dat, která jsou potřebná pouze během konverze a poté jsou zbytečná. Soubory s těmito daty můžete odstranit pomocí nástroje clean, který jsme právě vytvořili.

Když soubor otevřete...



... uvidíte všechny soubory s vygenerovanými daty...

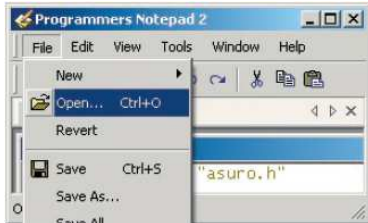


(Klikněte na [Cancel]).

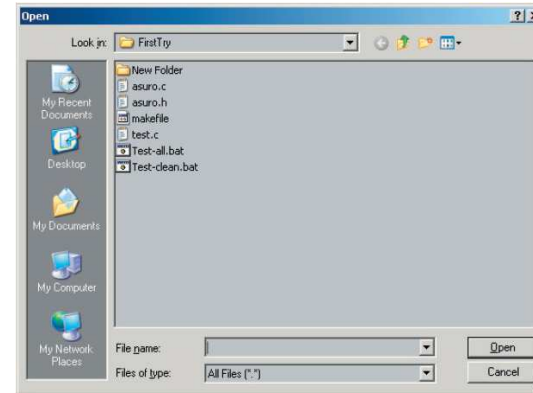
...A poté zadejte příkaz "clean" ...



...uvidíte ...



... že mnoho souborů s vygenerovanými daty bylo odstraněno.



K čemu došlo?

Vstup menu "clean" spustí batch soubor Test-clean.bat. Inicialoval to make s parametrem "clean". Nyní se jménem clean provede vstup v souboru make a všechna data, která už nebudou potřebná, se vymažou.

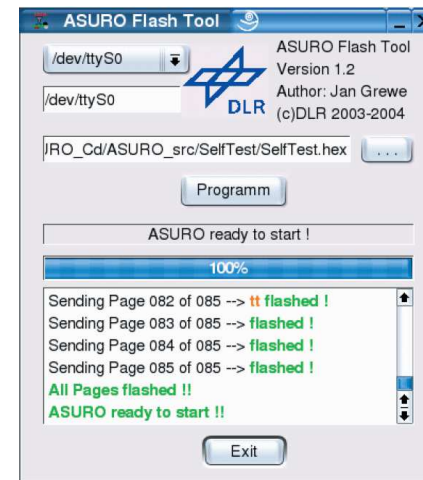
Linux

K instalaci budete potřebovat kořenová práva. Jestliže nejste přihlášen jako správce, odhlaste se a znovu se přihlaste do systému jako správce, nebo si kořenová práva vyžádejte pomocí "su".

Flash - nástroj k programování robota



Otevřete program z menu na CD a zkopírujte dva flash nástroje "asuroflash" a "asurocon" ze složky "/linux/tools" do složky "usr/local/bin". Poté musíte povolit spuštění programu pomocí "chmod a+X/usr/local/bin asurocon a asuroflash".



Obr. 8. 1: Flash-Tool

Kompilátor

Při instalaci kompilátorů Gnu pro procesory AVR vložte do čtecí mechaniky CDROM ASURO a ve složce "/Linux/Compiler/" vyberte následující:

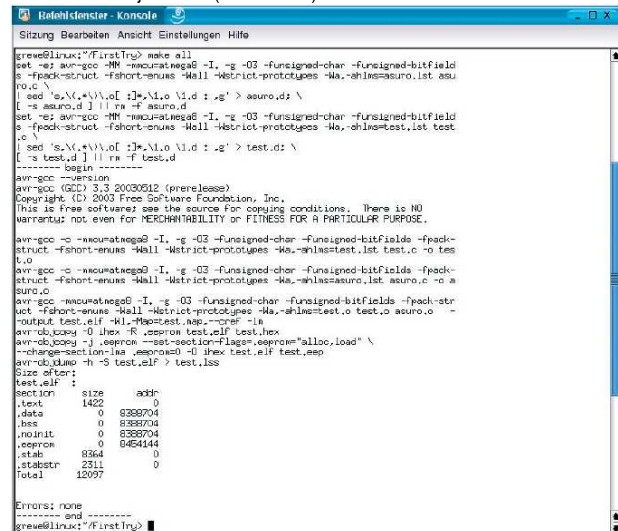
1. **avr-binutils-... .rpm**
2. **avr-gcc-... .rpm**
3. **avr-libc-... .rpm**

Instalace je celkem jednoduchá!

V kořenovém adresáři zadejte jen příkaz: rpm -i <paket>.rpm

Hotovo!

Z editorů můžete použít Emacs, Kate nebo Kedit. Na zkušku si můžete z CD zkopírovat demo programy. Najdete je ve složce "/ASURO_src/FirstTry/". Poté otevřete Shell, změňte složku a zadejte "make". Když je všechno v pořádku, uvidíte následující okno (viz obr. 8.2):



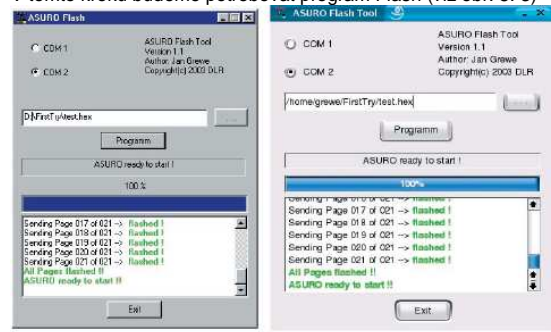
```
grewe@linux:~/FirstTry$ make all
g++ -c avr-gcc -MM -mcpu=atmega8 -I. -g -O3 -funsigned-char -funsigned-bitfield -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Werror -Wno-asano -list asuro.o
ls -s asuro.o | | ra -f asuro.o
ls -s asuro.o | | ra -f asuro.o
g++ -c avr-gcc -MM -mcpu=atmega8 -I. -g -O3 -funsigned-char -funsigned-bitfield -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Werror -Wno-asano -list test.o
ls -s test.o | | ra -f test.o
ls -s test.o | | ra -f test.o
----- begin -----
avr-gcc --version
avr-gcc (GCC) 3.3 20030512 (pre-release)
Copyright (C) 2003 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

avr-gcc -o -mcpu=atmega8 -I. -g -O3 -funsigned-char -funsigned-bitfields -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Werror -Wno-asano -c test.o
avr-gcc -o -mcpu=atmega8 -I. -g -O3 -funsigned-char -funsigned-bitfields -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Werror -Wno-asano -c asuro.o
avr-gcc -mcpu=atmega8 -I. -g -O3 -funsigned-char -funsigned-bitfields -fpack-struct -fshort-enums -Wall -Wstrict-prototypes -Werror -Wno-asano -list asuro.o -o asuro.o
avr-objcopy -O ihex -R .eeprom test.o test.hex
avr-objcopy -j eeprom --set-section-flags=eeprom="alloc,load" \
--change-section-lma .eeprom=0 -i ihex test.o test.ssp
avr-objdump -h -S test.o > test.lst
Size of file:
test.o:
section      size      addr
.text       1422      0
.data        0      838704
.bss         0      838704
.noinit      0      838704
.eeprom      0      845444
.stab        8364      0
.stabstr     2311      0
Total       12097

Errors: none
----- end -----
grewe@linux:~/FirstTry$
```

Obr. 8. 2: Make all

V tomto kroku budeme potřebovat program Flash (viz obr. 8. 3)



Obr. 8. 3: Flash-Tools pro Windows a Linux

Otevřete program a vyberte rozhraní, ke kterému je připojen IR vysílač. V adresáři C:\Own files\ASURO_src\FirstTry vyberte Test.hex.

Kompletně sestaveného robota ASURO položte vedle IR vysílače ve vzdálenosti max. 50 cm. Strany s komponenty obou DPS musí směřovat k sobě a nesmí být mezi nimi překážky (musí se navzájem "vidět"). Ve Flash-Tool klikněte na tlačítko Programm. Nyní přepněte S1 do polohy ON, dříve než indikátor stavu dosáhne pravý konec stavové oblasti. Pokud jste nedokázali reagovat dostatečně rychle, nebo byla komunikace přerušena, robota jednoduše vypnete, stisknete Programm a přepínač S1 dejte znovu do polohy ON.

Jakmile bude komunikace úspěšná, můžete na displeji a na indikátoru stavu pozorovat, jak se soubor Test.hex přenáší na robota. Soubor s programem se uloží ve Flash paměti uvnitř procesoru, kde zůstane program dostupný i po vypnutí napájení.

Po načtení programu se ASURO musí vypnout a znovu zapnout, aby se program spustil. Sekvence spustí načtený program a zelená LED se jasně rozsvítí.

K čemu došlo?

Pokud se spustí Flash program, PC se pokusí o komunikaci robotem. Zapnutí robota se systémem bootuje, což bude signalizováno rozsvícením dvoubarevné stavové LED na 1 sekundu. ASURO kontroluje, jestli je připraven nějaký nový program pro načtení. Pokud najde nový program, načte jej. Po načtení se program spustí, když robota vypnete a znovu zapnete.

Selhání Flash

V průběhu přenosu může dojít k následujícím chybám:

- "c" Checksum Error. ASURO obdržel nějaké chybné signály. Signály mohly být narušeny optickými zdroji, jako např. fluorescenčním světlem, nebo byly krátce přerušeny pohyby.
- "t" Timeout. Přerušila se přímá viditelnost mezi IR vysílačem a robotem.
- "V" Verify Error. ASURO zapsal do své Flash paměti neplatná data. Jedná se o tu nejméně obvyklou situaci, kdy je energeticky nezávislá programovací paměť (Flash - EPROM) na konci své životnosti. Podle technické dokumentace k tomu dochází po 10 000 programovacích cyklech.

Oprava chyb se může opakovat desetkrát. V případě, že se oprava nepovede, proces přenosu se ukončí.



Pokud během přenosu dochází pravidelně k ohlášení chyby "Checksum Error", zkuste vypnout nebo ztlumit světla v místnosti, zvláště pak fluorescenční světla.



Dříve než zapnete robota, vždy stiskněte tlačítko Programm. V opačném případě proces načtení nezačne.

Váš první program

V editoru Programmers Notepad dochází k modifikaci načteného zdrojového souboru. Pozor: držte se přesně textu v příkladech, včetně psaní malých a velkých písmen.

```
#include "asuro.h"
int main(void) {
  Init();
  StatusLED(RED);
  while(1);
  return 0;
}
```

Soubor si uložte a vyberte Tools > make, sledujte zprávy v zobrazovacím okně. Několik sekund počkejte, dokud program nedokončí hlášení. Zkontrolujte, jestli se ve spodním řádku objeví informace o bezchybném zpracování zdrojového souboru: Process Code: 0.

Pokud kompilátor zobrazí jinou zprávu, budete muset analýzou zdrojového kódu zjistit, kde se stala chyba. Stojí za to, podívat se na první řádek zdroje, ve kterém se píše o chybách. V levém spodním rohu zobrazovaného okna zobrazuje editor číslo řádku zdroje.

Po úspěšném převodu můžete záměrně změnit slovo v zdrojovém programu. Restartujte proces make a zkontrolujte, pod kterým číslem řádku byla ohlášena chyba.

Po úspěšné, tj. bezchybné kompilaci se nový program může přenést na robota. Připojte IR vysílač, otevřete Flash-Tool, vyberte nově vytvořený program test.hex a správné COM rozhraní. Položte robota vedle IR vysílače, aktivujte tlačítko Programm a počkejte, dokud se nedokončí přenos souboru. Po úspěšném dokončení přenosu vypněte robota a znovu ho zapněte. Sekundu počkejte a zkontrolujte, jestli se stavová LED změnila barvu na červenou. Dříve než změníte zdrojový kód, byste se měli dozvědět více o programování, a proto doporučujeme přečíst si nejdříve následující část "Programování ASURO v jazyce C".

Pozor!

Je také možné načíst demo program test.c ze složky ASURO_src\FirstTry;

Tento program, jenž obsahuje následující řádky, lze snadno zaměnit za výše uvedený program.

```
#include "asuro.h"
int main(void) {
  Init();
  while(1);
  return 0;
}
```

Programování ASURO v jazyce C

Tato část je věnována programovacímu jazyku C, ale zabývat se budeme jenom prvky, které jsou potřebné pro ovládání systému ASURO. Nejedná se samozřejmě o kompletní návod k C, který si můžete opatřit běžně na trhu ¹.

Jazyk C byl zvolen kvůli tomu, že jeho standard je hodně rozšířený a kompilátor C je dostupný pro téměř každý procesor. Pro robota ASURO jsme zvolili kompilátor volně dostupný Gnu-C-Kompilátor, který generuje dobře optimalizovaný kód pro procesor robota ATmega8.

Každý, kdo se už seznámil s jazykem C, se v této části návodu nedoví nic nového a může přejít dál na "Přehled funkcí robota ASURA". Popíšeme si vybrané prvky, které jsou nezbytně nutné pro ovládání robota ASURO.

Žádné obavy: Pokud si dáte pozor na používání závorek a středníků, nebude jazyk C vůbec složitý. A samozřejmě, ASURO není hračka!

Základy programování v jazyce C

Úvod

Procesor v zásadě vykonává příkazy v programu C krok za krokem od začátku do konce programu ². V standardním procesoru robota nelze zpracovávat paralelní procesy, takže se musíme zaměřit na sekvencní pracovní příkazy.

Prázdná místa na začátku řádků v příkladech slouží jen jako strukturální pomůcka pro čtení a lze je vynechat. Vložené mezery však umožňují vytvořit skvělou programovací strukturu a jsou mimořádně užitečné, protože pomáhají udržovat si přehled v delších programech.

Každý příkaz v jazyce C se musí ukončit středníkem ";", aby mohl kompilátor přiřadit všechny prvky k správnému příkazu. Bude-li potřebné spojit příkazy ve funkcích, smyčkách, nebo v podmíněných příkazech, seskupí se tato sada příkazů vložením mezi speciální závorky ("(", ")").

Příklad:

```
#include "asuro.h"
int main(void) {
  /* All elements between these brackets belong to one set of commands */
}
```

¹Například: Brian W. Kernighan, Dennis M. Ritchie: "Programming in C",

²Metody odchylek od sekvencního řízení v programech se nazývají "řízení toku" a popisují se níže. Když chceme mezi dva programovací řádky umístit komentář, uvádí se sada slov komentáře na začátku "/*" a končí znakem "*/". Jako komentář lze definovat i jeden řádek, který uvedeme znakem "/*" ³. Transformace příkazového řádku na komentář se často používá k vyjmutí určitých částí ze zpracování. Všechny komentáře bude kompilátor ignorovat. Správně vložené komentáře nezpůsobí nikdy problémy v provádění programu.

Proměnné a datové typy

Proměnné slouží v programování jako úložiště pro data a v průběhu programu se mohou definovat, vyplnit, číst, nebo změnit. Aby se proměnná mohla použít, musí se nejprve deklarovat. V deklaracích se proměnná přiřadí k určitému typu a může se jí zadat i počáteční hodnota. Typem se definuje, jaký druh dat se uvnitř proměnné ukládá (celá čísla, kladná čísla, reálná čísla, atd.).

Proměnné se označují názvy, které musí začínat písmenem (podtržítka _ se také považuje za písmeno) a mohou obsahovat číslice, ale nesmí obsahovat žádný ze zvláštních znaků. Rozeznávají se přitom velká a malá písmena. Například: x a X odkazují na různé proměnné. Velká písmena se běžně používají pro konstanty (které nelze v toku programu měnit, např. číslo $\pi = 3,1415$) a malá písmena se používají pro proměnné.

Následující názvy jsou vyhrazena slova, která nelze použít pro pojmenování proměnných.

<i>auto</i>	<i>default</i>	<i>float</i>	<i>long</i>	<i>sizeof</i>	<i>union</i>
<i>break</i>	<i>do</i>	<i>for</i>	<i>register</i>	<i>static</i>	<i>unsigned</i>
<i>case</i>	<i>double</i>	<i>goto</i>	<i>return</i>	<i>struct</i>	<i>void</i>
<i>char</i>	<i>else</i>	<i>if</i>	<i>short</i>	<i>switch</i>	<i>volatile</i>
<i>const</i>	<i>enum</i>	<i>int</i>	<i>signed</i>	<i>typedef</i>	<i>while</i>
<i>continue</i>	<i>extern</i>				

K programování robota se použijí následující datové typy:

Typ	Rozsah hodnot	Poznámky
char	-128 ... +127	Bajt (v délce 8 bitů) ukládá znak abecedy
unsigned char	0 ... 255	Ukládá jen kladné hodnoty
int	-32768 ... +32767	Dvoubajtové hodnoty v rozsahu -32768 ... +32767
unsigned int	0 ... 65535	Celá čísla v rozsahu 0 ... 65535
float		Reálná čísla

³ Podle standardů C++ je "/" znakem komentáře. Ve skutečnosti kompilátor pro robota je kompilátorem C++ a znak "/" bude chápat jako znak komentáře, ale v jiných kompilátorech s tím mohou nastat problémy.

Proměnné lze deklarovat jako globální mimo hlavní () funkci, nebo jako lokální uvnitř hlavní () funkce. Globální proměnné jsou platné v rámci celého programu, zatímco lokální proměnné budou platné jen uvnitř hlavní (), nebo nějaké jiné funkce. Lokální proměnné budou platné pro programovací kód uvnitř funkce a mimo funkce budou neplatné.

Proměnné jsou nám k ničemu, pokud je nedokážeme naplnit daty. Přiřazení dat je však poměrně jednoduché:

```
a=17; // proměnná, která obsahuje hodnotu 17
```

nebo pomocí výpočtu:

```
a=17+23; // proměnná, která obsahuje hodnotu 40
Speed=a+3; // proměnná Speed obsahuje hodnotu 43
Speed=Speed*2; // proměnná Speed obsahuje hodnotu 86
```

Je dobrou programovací praxí používat jasná a srozumitelná pojmenování. Název proměnné "speed" (rychlost) v uvedeném příkladě se rozumí sám o sobě. Používání samostatných písmen jako názvu proměnných by se mělo zamezit, protože to má často za následek špatně čitelný programovací kód.

Na následujícím příkladu ukážeme jednoduchý kompletní programovací kód:

```
#include "asuro.h"
int main(void) {
int i; // i může obsahovat čísla v rozsahu mezi -32768 and 32767
char any_token; // proměnná any_token může obsahovat symboly ASCII nebo
// čísla mezi -128 and 127
i=3;
any_token =17+; // proměnné any_token se nyní přiřadí 20
i=i/2; // dělení 2, vždy se zaokrouhlí směrem dolů
// proto i bude mít nyní hodnotu 1!
return 0;
}
```

V programovacím jazyku C existuje několik zajímavých zkratk, jako např.:

```
i=i+1;
lze rovněž zapsat jako
i++;
a stejně tak:
i=i-1;
lze rovněž zapsat jako:
i--;
```

Direktivy kompilátoru (překladače)

Možno vás zaujal první řádek programu #include "asuro.h". Tato direktiva #include definuje zařazení externího zdrojového kódu do vašeho programu a přikazuje překladači zařadit do své sekvence textfile. Náš soubor "include"obsahuje některé funkce, které budeme potřebovat pro provoz robota. Další důležitou direktivou (která je mimo jiné mimo rozsah našeho úvodu do jazyka C), je tzv. nahrazení textu. Tato direktiva se definuje jako:

```
#define NAME replacement_text
```

A budeme ji používat k definici konstant v našich programech.

Kdykoli když se ve zdrojovém textu objeví symbol NAME, dojde k jeho automatickému nahrazení replacements_text. Pro symbol NAME, který následuje za #define, se uplatňují stejná pravidla jako pro proměnné. Programátoři v jazyce C obvykle píší symboly (např. NAME) v #define velkými písmeny.

Příklad:

```
#include "asuro.h"
#define ne STARTINGVALUE 33
int main(void) {
int i;
i= STARTINGVALUE; // hodnota pro i je nyní 33
return 0;
}
```

Poznámka: direktivy překladače nejsou ukončeny středníkem.

Podmínky

Někdy potřebujeme provést příkaz za určitých podmínek. Těmto podmíněným sekvencím říkáme řídicí struktury. Nejjednodušší z řídicích struktur je sekvence "if-else".

V jazyce C bude správná syntax vypadat následovně:

```
if (Podmínka)
Command_block_1
else
```

Command_block_2

Program zkontroluje hodnotu podmínky mezi závorkami. Je-li podmínka pravdivá (což se aplikuje na každou hodnotu kromě nuly), program provede Command_block_1 a v opačném případě Command_block_2.

Pokud by program dokázal vybrat jednu možnost z několika alternativ, tak můžete použít několik konstrukcí "else-if".

```
if (Condition1)
Command_block_1
else if (Condition2)
Command_block_2
else if (Condition3)
Command_block_3
else if (Condition4)
Command_block_4
else
Command_block_5
```

Mohou se použít následující podmíněné struktury:

Operátor	Vysvětlení
==	Porovnání rovnosti
!=	Porovnání nerovnosti
<	Porovnání pro méně
>	Porovnání pro více
<=	Porovnání pro méně nebo rovnající se
>=	Porovnání pro více nebo rovnající se

Příklad:

```
#include "asuro.h"
int main(void) {
while (1) {
if (PollSwitch(>0) {StatusLED(RED);}
else {StatusLED(GREEN);}
}
}
```

Pokud je aktivován jeden z kolizních spínačů, stavová LED bude svítit červeně, jinak svítí zeleně. Druhý programovací kód vysvětlíme později.

V jazyce C představuje "1" true (pravda) a "0" false (nepravda)

Podmíněný příkaz

```
if (0) {StatusLED(RED);}
```

samozřejmě příkaz StatusLED(RED) nebude nikdy vykonán.

Cykly

Cykly slouží k opakovanému vykonání příkazu.

V cyklu s podmínkou "while" se podmínka vyhodnotí po každém cyklu. Pokud je podmínka pravda, provede se blok příkazu. Podmínka se zkontroluje, dokud se nezmění na nepravdu (false). Za stavu false bude program pokračovat prvním příkazem, který následuje po bloku podmínky.

While (podmínka)
Příkazový blok

Příklad:

```
#include "asuro.h"
int main(void) {
MotorDir(FWD,FWD); // Oba motory běží dopředu
MotorSpeed(120,120); // Oba motor běží přibližně na poloviční rychlost
StatusLED(GREEN); // Stav zapnuto - LED svítí zeleně
while (PollSwitch()==0) { // As long as there is no collision
SerWrite("All OK!\n",10); // ... Feeling groovy ....
}
MotorSpeed(0,0); // Collision! Stop immediately!
StatusLED(RED); // Turn on Status-LED red
while (1) {
SerWrite("Ouch!\n",5); // start crying!
}
}
A "for (expr1, epr2, expr3)"- sequence is equivalent to:
expr1;
while( expr2) {
Command block
expr3;
}
```

Sekvence "for"- se často používá jako početní sekvence.

```
for (i = 0; i < n; i++)
```

Příklad:

```
#include "asuro.h"
int main(void) {
int counter; // defi ne a variable for counting
for (counter =0;counter <10;counter ++) { // repeat ten times:
SerWrite("Go ahead!\n",10); // "Go ahead" message
}
MotorDir(FWD,FWD); // Both engines forward
MotorSpeed(120,120); // Both engines running at around half speed
while (1) { // No more action!
}
}
```

"while(1)" je ekvivalentem "for(;;)" a jde o nekonečné cykly, protože podmínka pro přerušení (v tomto případě 0) nemůže být nikdy dosažena.

Jinou cyklickou sekvencí je "do" cyklus

```
do
příkazový blok
while (podmínka)
```

Oproti podmínce "while" se tato podmínka kontroluje po příkazovém bloku. Tato programová sekvence vynutí provedení příkazového bloku vždy alespoň jednou.

Funkce

Blok definice funkce vypadá vždy následovně:

Typ funkce *Název funkce* (*Typ parametru 1* *Název parametru 1*, *Typ parametru 2*, *Název parametru 2*,...)

V programech někdy potřebujeme použít stejnou programovací sekvenci na různých místech. Můžeme samozřejmě celou sekvenci napsat znovu, nebo k tomuto čelu použít metodu ctrl/C - ctrl V (jde o otravný proces, kterým si uvedeme zmatek do programu), nebo si prostě definovat funkci. Nyní potřebujeme dát do funkce několik proměnných, např. říct naší funkci GoAhead (), aby pracovala přesně v dané rychlosti, v určitém cyklu, nebo v definované vzdálenosti. K těmto detailům se použijí parametry.

Funkce může vrátit i hodnotu. Dobrým příkladem je funkce *HowManySwitchesHaveBeenActivated ()* - (kolik přepínačů se aktivovalo), vracející hodnotu, která se nějak a někde uvnitř funkčního bloku nadefinuje. Hodnota se vrací příkazem *return* na konci těla funkce.

To je důvod proč funkce končí výrazem *return*; nebo *return number*;

Speciální funkcí je funkce *main ()*, která definuje hlavní tělo programu. V ASURO se funkce *main ()* provádí při zapnutí. Funkci *main ()* musí samozřejmě obsahovat každý program.

Po troše teorie týkající se datových typů a funkcím bychom chtěli vytvořit jednoduchou funkci, která vynásobí dvě 8 bitové čísla a vrátí výsledek.

```
int Mult (char a, char b)
/* Funkce vrací celé číslo, má název Mult a pro vstup používá hodnoty tvořené dvěma znaky */
{ // Begin function
int c; // Declaring variable c as an int
c = a * b; // calculate c
return c; // returning integer c
} // End of the function "Mult"
```

Nyní příklad programu, který používá výše definovanou funkci Mult:

```
int main (void) // Function main always returns an int and will not
// input any parameter
{ // Begin Function „main“
char mult1,mult2; // Defining two „char“-variables
int result; // Defining an int-variable for the result of multiplying
// variables mult1 and mult2
mult1 = 2; // assignment
mult2 = 10; // assignment
result = Mult(mult1,mult2); // calling the previously defined function "Mult"
return 0;
} // End of function „main“
```

Ukazatele a vektory

Ukazatele a vektory vysvětlíme do té míry, jak je potřebné pro provoz robota ASURO.

Vektory budeme potřebovat, když chceme sledovat data ze senzorů sledujících dráhu pohybu, nebo z odometrických senzorů.

Deklarace je celkem jednoduchá:

```
int lineData[2];
int odometrieData[2];
```

Jak uvidíte, vytvoříme dva vektory (lData, oData) s 2 prvky pro sledování dráhy a odometrii. Voláním funkce (LineData(), OdometrieData ()), prvek [0] dostává hodnotu levého senzoru a prvek [1] dostává hodnotu pravého senzoru.

Ukážeme si tuto metodu na příkladu:

Když dostává pravý senzor více světla než levý senzor, provede se příkaz 1, v opačném případě se provede příkaz 2.

```
int lData[2]; // Přidělení paměti pro výsledky měření
LineData(lData); // Čtení naměřených dat
if (lData[1] > lData[0])
command1;
else
command2;
```

K použití funkcí sériových rozhraní (*SerWrite()*, *SerRead()*) budeme potřebovat řetězec znaků, který deklarujeme jako:

```
char message[] = "This is a text string"
```

Abychom odeslali textový řetězec do robota, voláme funkci *SerWrite()* s příslušnými parametry. První parametr obsahuje textový řetězec, nebo proměnnou, která obsahuje textový řetězec. Druhý proměnná popisuje počet znaků, který se má odeslat, např.:

```
SerWrite(message,20);
```

respektive

```
SerWrite("This is a text",14);
```

odešle na IR vysílač zprávu "This is a text".

Pro příjem znaků používá ASURO funkci *SerRead()*. První parametr obsahuje proměnnou, ve které se ukládají přijaté znaky. Druhý parametr definuje, kolik znaků se přijme a třetí parametr definuje čas platnosti funkce: Pokud se do určité doby (počet hodinových cyklů procesoru) nepřijmou žádná data *SerRead()*, funkce se zruší. Při použití čísla "0" však funkce počká, až se přijmou všechny znaky. Funkci ukážeme na příkladu.

ASURO má přijmout z IR vysílače zprávu "Hi, here I am". Pomocí definice řetězce

```
char message [] = "01234567890123456789"
```

nejdříve přiřadíme očekávanému textu prostor v paměti. Je samozřejmé, že místo v paměti musí být pro očekávanou zprávu dostatečně velké.

```
SerRead (message,13,0)
```

Přečti 13 znaků a počkej, dokud se nepřijme 13 znaků. Nyní máme za to, že byl odeslán textový řetězec "Hi, here I am". Funkce přepíše prvních 13 znaků předdefinovaného řetězce zprávy větou "Hi, here I am" a výsledkem bude řetězec:

```
Hi, here I am 3456789
```

Přehled funkcí robota ASURO

K programování robota ASURO bylo vytvořeno několik funkcí. Tyto funkce nejsou v žádném případě optimálním řešením pro všechny účely a k některým účelům bude vhodné si napsat speciální funkce. Všechny funkce byly vytvořeny, jak je definováno v deklaracích a lze je pochopit kontrolou příkladů. Aby se zabránilo špatnému pochopení: ovládací funkce jako řízení motoru nebo funkce indikátorů mění stav, který zůstane platný až do další změny. Zelený LED indikátor zůstane zelený až do další změny na jinou barvu, nebo dokud se nevyvypne.

void Init (void)

Tato funkce resetuje mikroprocesor do původního stavu a musí se provést vždy na začátku programu. Když funkce chybí, procesor nebude vědět, ani co má dělat s terminály. Jednoduchý program pro ASURO by měl vypadat aspoň takto:

```
#include "asuro.h"
int main(void) { // zde deklarujeme některé proměnné
Init(); // zde vložíme naše vlastní funkční bloky
while(1); // nekonečný cyklus
return 0; // nikdy se neprovede
}
```

Proč jsme na konec funkce *main()* vložili nekonečný cyklus? Za normálních okolností bude funkce *main()* ukončena příkazem *return 0*, který ukončuje program. V případě robota ASURO však mohou v jeho paměti zůstat a spouštět se části dříve nahraných programů, což může mít vyvolávat zvláštní chování. Abychom se vyhnuli provádění částí starých programů, "zachytíme" program po jeho provedení do nekonečné smyčky. Tímto způsobem si zabezpečíme, že program skončí v nadefinovaném stavu.

void StatusLED (unsigned char color)

Stavová LED (D12) se vypne, nebo zapne. Platné hodnoty parametru jsou OFF, GREEN, RED nebo YELLOW

Příklad:

Stavová LED se zapne a bude svítit červeně:

```
StatusLED (RED);
```

Kompletní ukázkový program bude vypadat následovně:

```
#include „asuro.h“
int main(void) {
Init();
StatusLED (YELLOW);
while(1); // eternal loop
return 0;
}
```

void FrontLED (unsigned char status)

Přední LED (D11) se zapne nebo vypne. Platné hodnoty parametru jsou ON a OFF.

Příklad:

Přední LED se zapne:

```
FrontLED(ON);
```

void BackLED (unsigned char left, unsigned char right)

Zadní LED indikátory (D15 a D16) se zapnou nebo vypnou. První parametr popisuje stav levé zadní LED (D15), zatímco druhý parametr popisuje stav zadní LED na pravé straně (D16). Platné hodnoty parametru jsou ON a OFF.

Příklad:

Zadní LED na pravé straně (D16) se zapne a zadní LED na levé straně (D15) se vypne:

```
BackLED(OFF,ON);
```

void Sleep (unsigned char time72kHz)

Touto funkcí se přikáže procesoru, aby určitou dobu čekal. Doba čekání se může definovat parametrem (unsigned char), který bude obsahovat číslo s maximem 255 a výpočetní cykly, které dodává 72 kHz-timer.

Příklad:

Procesor bude asi 3 ms. v režimu spánku ==> $\frac{0,003 \text{ s}}{\frac{1}{72 \text{ KHz}}} = 216$.

Funkce sleep (216); přinutí procesor 3 milisekundy čekat.

Sleep (216);

void MotorDir (unsigned char left_dir, unsigned char right_dir)

Tato funkce kontroluje směr obou motorů a měla by se volat ještě před voláním ovladačů rychlosti. Platné parametry jsou FWD (vpřed), RWD (vzad), BREAK (brzdění, nebo náhlé zastavení zkratováním motorů v přemostění tranzistorů) a FREE (volnoběh).

Příklad:

Levý motor se bude pohybovat vpřed a pravý motor je zastaven:

MotorDir(FWD,BREAK);

void MotorSpeed (unsigned char left_speed, unsigned char right_speed)

Tato funkce kontroluje rychlost obou motorů. Maximální rychlost je 255 (unsigned char). V závislosti na mechanických podmínkách se motor začne točit při hodnotě kolem 60. Hodnota parametru kontroluje ve skutečnosti výkon motoru a rychlost otáčení závisí také na jiných faktorech, jako je tření nebo sklon svahu.



Jakmile se tato funkce provede, ASURO se začne pohybovat. Někdy se však stává, že program má za následek neočekávané pohyby a musíme dávat pozor, aby ASURO nezpůsobil žádnou škodu, nebo se sám nepoškodil.

Příklad:

Levý motor se bude pohybovat při maximální rychlosti, zatímco pravý motor se nebude pohybovat. Směr pohybu byl už dříve definován jako *MotorDir* ().

MotorSpeed (255,0);

void SerWrite (unsigned char *data, unsigned char length)

Funkce kontroluje výstup dat z robota přes sériové IR rozhraní (2400 Bit/s, No-parity, 1 StopBit, NoFlowControl). První parametr obsahuje odkaz na data, která se mají odeslat, zatímco druhý parametr popisuje počet znaků, které se mají odeslat.

Příklad:

Přes IR rozhraní se odešle řetězec „Hello how are you?“

SerWrite (“Hello how are you?”,18);

void SerRead(unsigned char *data, unsigned char length, unsigned int timeout)

Když už jste schopni odesílat data přes IR rozhraní, pravděpodobně budete chtít i nějaká data přijímat. Následující funkce vám to umožní. První parametr ukazuje na adresu paměti, ve které chcete zprávu uložit. Druhý parametr popisuje, kolik znaků se očekává a třetí parametr udává dobu pro vypršení platnosti (timeout). Timeout se používá, aby se zabránilo nekonečnému čekání, pokud se přijme menší, než očekávané množství dat. Pokud po uplynutí zadané doby pro vypršení platnosti nepřijdou další data, funkce se přeruší a první znak v přijatém řetězci se nahradí znakem "T" (=Timeout). Pokud však definujete třetí parametr jako "0", funkce se nepřeruší a bude čekat, dokud se nepřijme poslední z očekávaného počtu znaků.

Příklad:

Měl by se přijmout řetězec "Go Ahead" a předtím než bude robot pokračovat v činnosti, chceme se ubezpečit, že ASURO skutečně přijal všechny znaky.

```
#include "asuro.h"
int main(void) {
char data[8]; // allocate storage
Init();
SerRead (data,8,0); // Read data
MotorDir(FWD,FWD);
MotorSpeed(120,120);
while(1); // Eternal loop
return 0;
}
.
.
.
```

void LineData(unsigned int *data)

Tato funkce byla vytvořena pro čtení intenzity světla obou fototranzistorů na spodní straně robota. Budete muset definovat ukazatel adresy místa duálního celého čísla v paměti. Funkce přenesené hodnoty naměřených dat obou fototranzistorů do převedeny v AD převodníku. Hodnota prvního celého čísla představuje převedenou hodnotu levého fototranzistoru (T9), hodnota druhého celého čísla představuje převedenou hodnotu pravého fototranzistoru (T10). Maximální intenzita (jas) má hodnotu "1023", zatímco úplná tma se označuje jako "0". Za normálních okolností se tyto dvě krajní meze nevyskytují a v praxi se setkáváme s hodnotami měření, které jsou někde uprostřed mezi oběma krajními hodnotami.

Příklad:

Čtení intenzity světla obou fototranzistorů (T9, T10)

```
unsigned int data[2]; //Allocate storage
.
.
LineData(data);
```

data[0] obsahuje hodnotu naměřenou levým fototranzistorem (T9)
data[1] obsahuje hodnotu naměřenou pravým fototranzistorem (T10)

Příklad celého programu:

```
#include "asuro.h" // Line tracing the easiest way
int main(void) {
unsigned int data[2]; // Allocate storage
Init();
FrontLED(ON); // Switch ON line trace illumination
MotorDir(FWD,FWD); // Both engines go ahead
while(1){ // Eternal loop, ASURO should follow
// a line eternally
LineData(data); // Read brightness data from Phototransistors
if (data[0]>data[1]) // left brighter than right
{MotorSpeed(200,150);} // ... speed up left motor
else
{MotorSpeed(150,200);} // ... speed up right motor
}
return 0;
}
```

void OdometrieData(unsigned int *data)

Funkce snímá senzor odráženého světla: Obě LED (D13, D14) jsou aktivní a funkce vrací AD konvertované hodnoty fototranzistorů (T11, T12). Podobně jako ve funkci LineData() se musí 2 celými čísly zadat oblast v paměti, která bude funkcí vyplněna. První hodnota celého čísla obsahuje AD hodnotu z levého fototranzistoru (T11), druhá hodnota celého čísla obsahuje AD hodnotu z pravého fototranzistoru (T12). Maximální jas označuje hodnota "0" a absolutní tmu představuje hodnota "1023"⁵. Za normálních okolností se tyto dvě krajní meze nevyskytují a v praxi se setkáváme s hodnotami měření, které jsou někde uprostřed mezi oběma krajními hodnotami.

Příklad:

Snímání senzorů odráženého světla.

```
unsigned int data[2]; //Allocate memory
```

```
.
```

```
OdometrieData(data);
```

```
data[0] obsahuje hodnotu z levého fototranzistoru (T11)
```

```
data[1] obsahuje hodnotu z pravého fototranzistoru (T10)
```

Aby se zabránilo špatnému pochopení: OdometrieData() neudávají počet otáček, ale skutečné osvětlení na senzorech odráženého světla. Náročné světlé a tmavé úrovně, počítání přechodů světlo - tma a počítání počtu otáček se ponechává na programátoru.

⁵ Pro zjednodušení hardwarového obvodu tyto hodnoty nekorrespondují s hodnotami v modulu pro sledování dráhy.

unsigned char PollSwitch (void)

Funkce snímá polohu přepínačů (K1 - K16) a vrací jeden bajt s informací, které přepínače jsou aktivní.

Přepínač 1 nastaví první bitové číslo 5, přepínač 2 nastaví druhý bit ... přepínač 6 nastaví bitové číslo 5,

Bit0 (1) -> K6

Bit1 (2) -> K5

Bit2 (4) -> K4

Bit3 (8) -> K3

Bit4 (16) -> K2

Bit5 (32) -> K1

Aktivace přepínačů 1, 3 a 5 způsobí, že funkce vrátí 42 (32+8+2 = 42).

Pro jistotu se funkce může volat několikrát po sobě, aby poskytla "správnou" odpověď. Kondenzátor C7 se musí nejdříve vybit, což může nějaký čas trvat. Pokud nasnímate předčasné AD převodník, mohou být data, která získáte, nespolehlivá.

Příklad:

```
unsigned char taste;
```

```
.
```

```
.
```

```
switch = PollSwitch();
```

```
if (switch>0) {MotorSpeed(0,0);}
```

To je vše. Ostatní je už jen na vaší kreativitě.

Přílohy

Příloha A - seznam dílů

Kromě pingpongového míčku budete k sestrojení robota potřebovat následující části.

1 x deska plošných spojů ASURO

2 x motory typu Igarashi 2025

1 x dioda 1N4001

8 x diody 1N4148

4 x tranzistory BC 327/40 nebo BC 328/40

4 x transistory BC 337/40 nebo BC 338/40

1 x přeprogramovaný procesor ATmega 8L-8PC

1 x IR vysílač SFH 5110-36

2 x fototranzistory SFH300

3 x červené, jasné difuzní LED, 5 mm, nebo asymetrické širokoúhlé

1 x dual LED, 3 mm

2 x postranní fototranzistory LPT80A

2 x postranní LED IRL80A

1 x krystal 8 MHz

2 x Elco 220_F alespoň 10V RM 3,5/10

4 x keramické kondenzátory 100 nF RM 5,08

2 x keramické kondenzátory 4,7 nF RM 2,54

1 x 100 Ω 1/4 W 5%

2 x 220 Ω 1/4 W 5%

4 x 470 Ω 1/4 W 5%

10 x 1 kΩ 1/4 W 5%

1 x 1 kΩ 1/4 W 1%

3 x 2 k 1/4 W 1%

2 x 4,7 k 1/4 W 5%

1 x 8,2 k 1/4 W 1%

1 x 10 k 1/4 W 1%

1 x 12 k 1/4 W 1%

1 x 16 k 1/4 W 1%

1 x 20 k 1/4 W 5%

1 x 33 k 1/4 W 1%

1 x 68 k 1/4 W 1%

1 x 1M 1/4 W 5%

3 x patice 14 pol

6 x kolizních přepínačů

1 x přepínač napájení (on/off)

1 x držák baterií

1 x uchycení baterií

1 x propojka

1 x 2 pólové propojovací kolíky RM 2,5

2 x ozubené kolo 10/50 ozubení 3,1 mm, modul 0,5

2 x ozubené kolo 12/50 ozubení; 3,1 mm, module 0,5

2 x pastorek 10 (nebo 12), ozubení: 1,9, modul 0,5

2 x manžeta na 3 mm nápravu

4 x vázací páska kabelů

1 x vázací páska kabelů nastavitelná

2 x gumové pneumatiky 38 mm

2 x náprava délka 42 mm, průměr 3 mm

2 x náprava délka 24,5 mm, průměr 3 mm

Kolem 15 cm drátu, červený 0,14 mm

Kolem 15 cm drátu černý 0,14 mm

2 x Encoder nálepka

Volitelný vysílač RS-232-IR bude obsahovat následující části:

- 1 x Deska plošných spojů IR-RS232-transceiver
- 3 x diody 1N4148
- 1 x Zenerova dioda ZPD5.1
- 1 x tranzistor BC 547 A, B nebo C nebo BC 548 A, B nebo C
- 1 x integrovaný obvod NE555N
- 1 x IR přijímač SFH 5110-36
- 1 x IR-LED SFH415-U
- 1 x Elco 100_ F alespoň 16V RM 2,5/6
- 2 x keramické kondenzátory 100 nF RM 5,08
- 1 x keramický kondenzátor 680 pF RM 2,54
- 1 x 220Ω 1/4 W 5% nebo lepší
- 1 x 470Ω 1/4 W 5% nebo lepší
- 1 x 4,7 KΩ 1/4 W 5% nebo lepší
- 1 x 10 KΩ 1/4 W 1%
- 2 x 20 KΩ 1/4 W 5% nebo lepší
- 1 x trimr 10 k RM 2,5/5, svisle umístěný
- 1 x 8-pólový socket
- 1 x 9-pólový SUB-D

Recyklace



Elektronické a elektrické produkty nesmějí být vyhazovány do domovních odpadů. Likviduje odpad na konci doby životnosti výrobku přiměřeně podle platných zákonných ustanovení.

Šetřete životní prostředí! Přispějte k jeho ochraně!

Manipulace s bateriemi a akumulátory



Nenechávejte baterie (akumulátory) volně ležet. Hrozí nebezpečí, že by je mohly spolknout děti nebo domácí zvířata! V případě spolknutí baterii vyhledejte okamžitě lékaře! Baterie (akumulátory) nepatří do rukou malých dětí! Vyteklé nebo jinak poškozené baterie mohou způsobit poleptání pokožky. V takovém případě použijte vhodné ochranné rukavice! Dejte pozor nato, že baterie nesmějí být zkratovány, odhazovány do ohně nebo nabíjeny! V takovýchto případech hrozí nebezpečí exploze! Nabíjet můžete pouze akumulátory.



Vybité baterie (již nepoužitelné akumulátory) jsou zvláštním odpadem a nepatří do domovního odpadu a musí být s nimi zacházeno tak, aby nedocházelo k poškození životního prostředí!

K těmto účelům (k jejich likvidaci) slouží speciální sběrné nádoby v prodejnách s elektrospotřebiči nebo ve sběrných surovinách!

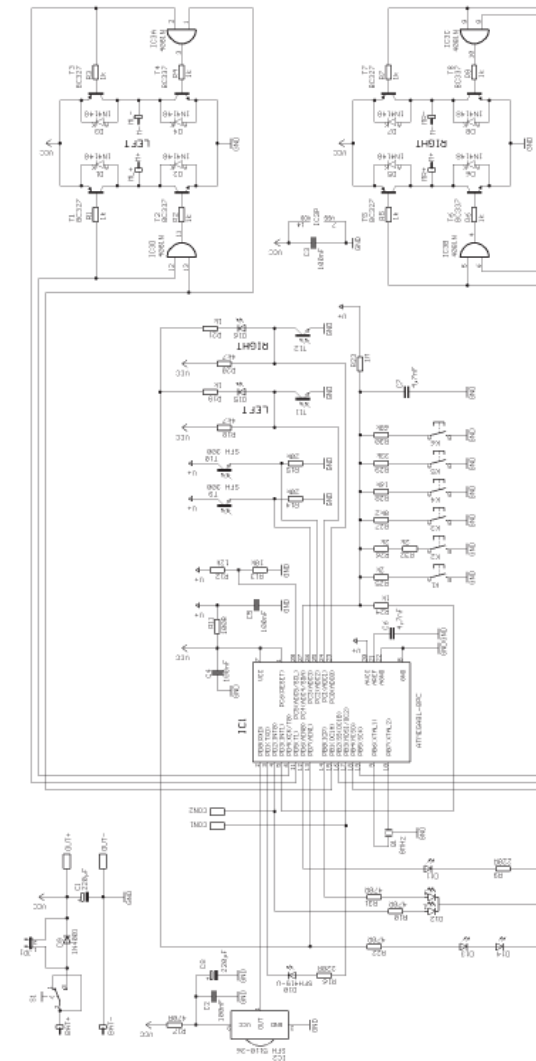


Šetřete životní prostředí!

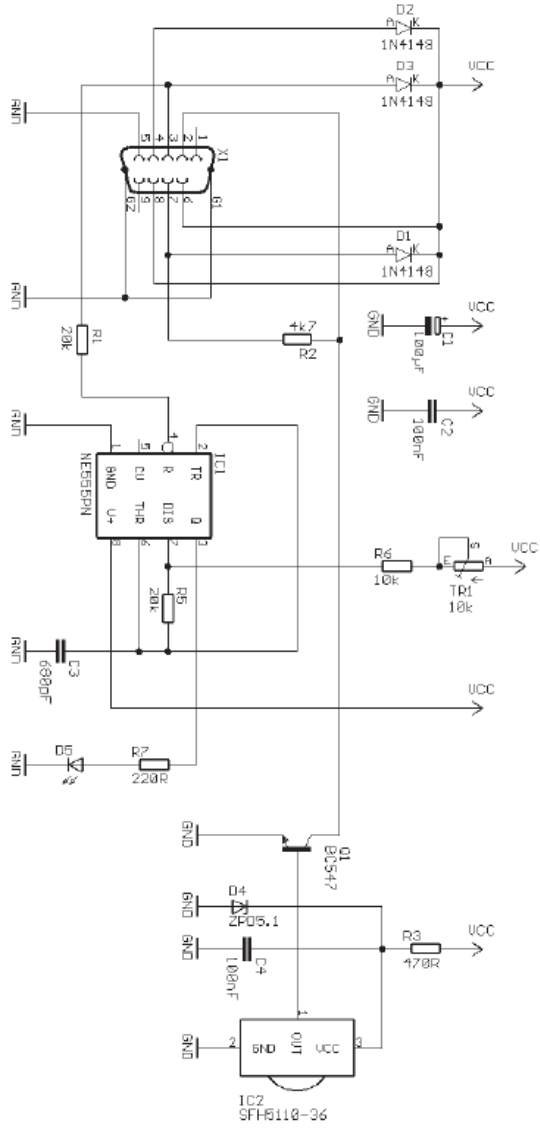
Záruka

Na stavebnici programovatelného robota ASURO ARX-03 poskytujeme **záruku 24 měsíců**. Záruka se nevztahuje na škody, které vyplývají z neodborného zacházení, nehody, opotřebení, nedodržení návodu k obsluze nebo změn na výrobku, provedených třetí osobou.

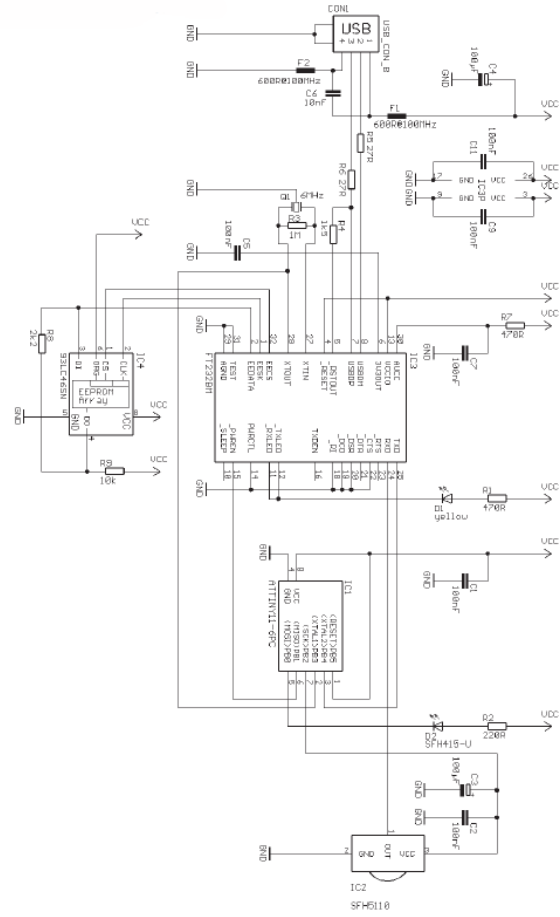
B. Schéma ASURO



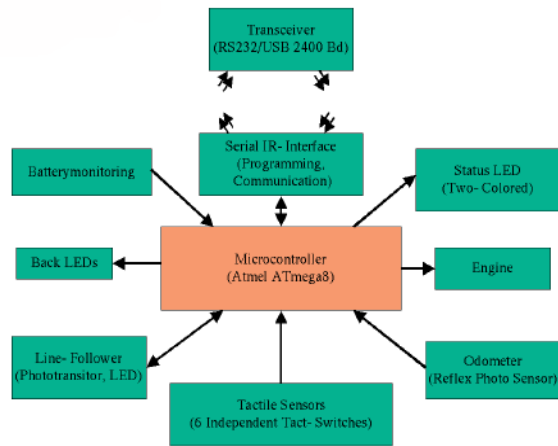
C. Vysílač RS-232



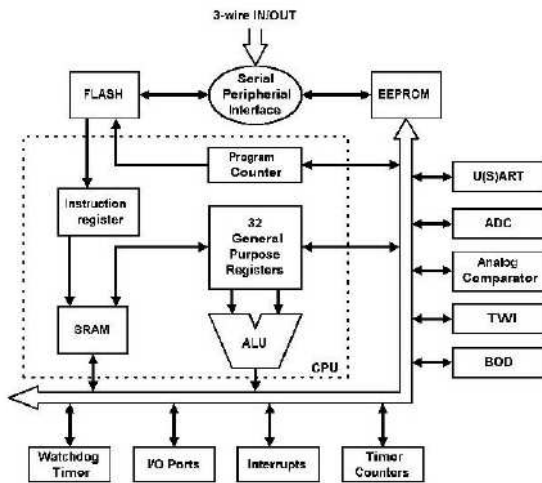
D. USB vysílač



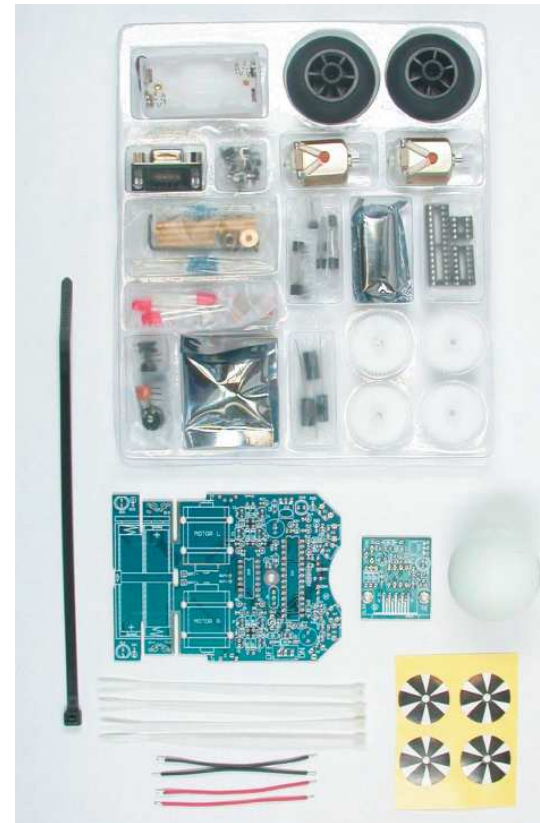
E. Blokové schéma ASURO



F. Blokové schéma PIC procesor



G. Obsah sady ASURO



Překlad tohoto návodu zajistila společnost Conrad Electronic Česká republika, s. r. o.

Všechna práva vyhrazena. Jakékoliv druhy kopií tohoto návodu, jako např. fotokopie, jsou předmětem souhlasu společnosti Conrad Electronic Česká republika, s. r. o. Návod k použití odpovídá technickému stavu při tisku! **Změny vyhrazeny!**

© Copyright Conrad Electronic Česká republika, s. r. o.

VAL/05/2014