Středoškolská technika 2014

Setkání a prezentace prací středoškolských studentů na ČVUT

Plotter

Matěj Vodička, Václav Zelený

Střední průmyslová škola sdělovací techniky

Panská 3, Praha 1

ANOTACE:

Cílem naší práce je konstrukce funkčního kreslícího nástroje řízeného počítačem, který lze využít například při tisku výkresů z AutoCADu. Práce obsahuje popis funkce a principu programů pro PC a plotter, návrh datové komunikace a schéma elektrického zapojení. Dále je zde konstrukce ve formě 3D modelu a jednotlivé části ve výkresové dokumentaci.

ANNOTATION:

The aim of our work is the design of a prototype computer-controlled drawingtool which can be used for example for the printing of drawings made in AutoCAD. The work describes the functions of the programm, the principle of use of computer software and plotter-software, and the design of data transmission between the PC and the plotter tool. The work also contains an electrical wiring diagram, the presentation of a 3D-model and the documentation of the individual parts of the drawing.

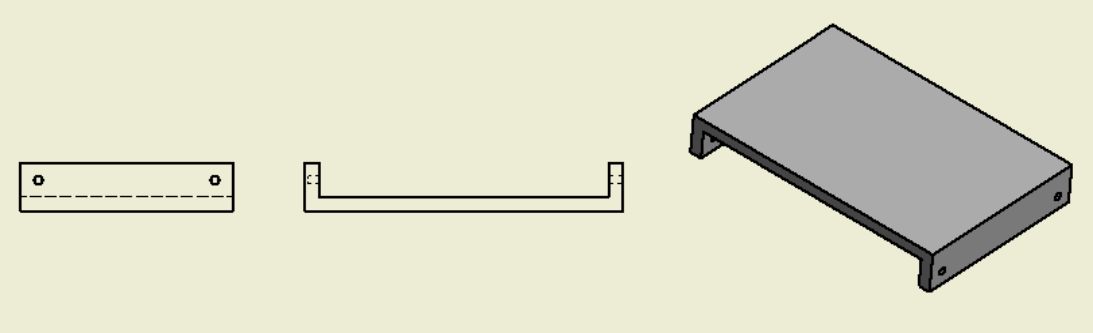
1. ÚVOD

V této práci jsme se zabývali zpracováním návrhu a následně i konstrukcí plotteru. Toto zařízení je vlastně tiskárna, ovšem pracuje na jiném principu než normální tiskové stroje. Principem funkce plotteru je, že kreslící hlava tiskne rovnou celé křivky a čáry a ne pouze pixely, jeden po druhém. Nejčastější použití plotterů je hlavně v technické praxi, ale v dnešní době se využívají pro osobní účely, například jako řezací pomůcka na fólie atd. V hlavách plotterů může být použita celá řada nástrojů. My jsme použili mikrotužku, ale konstrukce nám umožňuje s mírnými změnami užít technické pero, řezné nože nebo barevné fixy.

2. KONSTRUKCE

2.1. Použité materiály

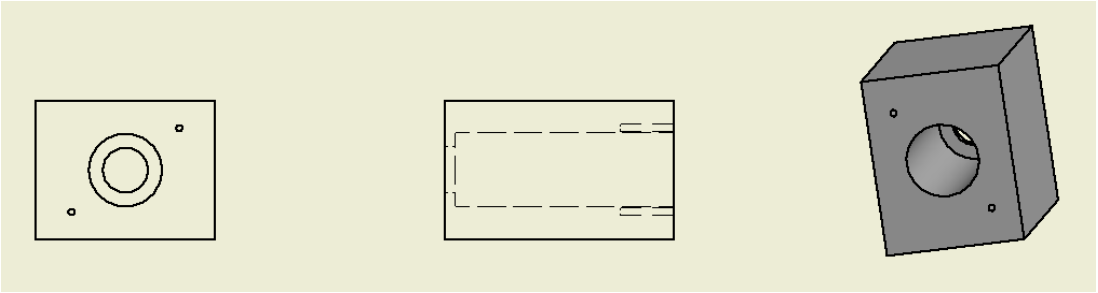
Plotter se skládá z dřevěné kreslicí desky pro uložení papíru, samotné kreslicí hlavy a mechanizmů pro její posuny. Deska o velikosti 4100 x 3000 mm je vyrobena z laťovky, protože jsme k ní měli přístup a zároveň je se svou tloušťkou 20 mm dost masivní, a tím pádem se nebude ani při této velikosti prohýbat. Sice výkresy těchto rozměrů není možné naším plotterem vykreslit, ovšem cílem bylo vyrobit zařízení, které bude schopné vykreslovat výkresy pouze do velikosti A4. Problém s touto deskou bylo ten, že díky struktuře dřeva nebyla dostatečně rovná. Bylo by možné, že by se kreslící nástroj zachytával v jednotlivých létech (drážkách), které dřevo přirozeně na povrchu má, a tím by se přesnosti zvětšovaly. Totiž při kreslení čáry, která není rovnoběžná s léty a svírá s nimi jen malý úhel, by se kreslící nástroj mohl držet v „drážce“. Řešením tohoto problému bylo zatmelení nezbytně nutnou vrstvou tmelu, následné přebroušení a to tolikrát, dokud byla na vrchní straně znatelná léta a jiné nerovnosti. Jinou možností je přelakování dvousložkovým epoxidovým lakem. Deska je připevněna vruty k čelům, které jsou vyrobené ze stejného materiálu. Ostatní části, kreslící hlava a zbytek konstrukce, pohybující se v delším směru, jsou vyrobeny z dubového dřeva.



Obr. 1: Základna plotteru

2.2. Posuv po tyčích

Při vymýšlení realizace samotného posouvání pohyblivé části plotteru jsem nejprve řešil tuto problém tak, že dřevěné kvádry, nesoucí kreslicí nástroj, budou vyvložkovány. Vnitřní průměr vložky by byl vrtán stejným průměrem nástroje, ovšem s přesností například H8 a v něm by byla uložena tyč o stejném průměru s přesností f7, tzn. uložení s malým přesahem. Dalším prvkem, který mněl napomoci vedení bylo to, že kvádry se jednou stranou měly posouvat zespoda po kreslící desce. Toto řešení narazilo na jednu velkou překážku – přesnost zpracování. V podmínkách, které jsme měli, nebylo možné vyrobit všechny součásti tak dokonale, že by toto řešení bylo možné.



Obr. 2: Krychle pro posun po vodících tyčích

Při hledání jiného způsobu vyřešení jsme narazili na součást, která je přímo pro tyto účely vyráběna. Lineární ložisko je zařízení snižující, podobně jako více známá radiální nebo také kuličková ložiska, odpor a v tomto případě zajišťující přesné vedení. Pro pohyb v delší ose plotteru jsou vodící tyče průměru 12 mm.

2.3. Posuv hlavy

Při řešení konstrukce samotného plotteru jsme dospěli k několika změnám oproti návrhu původního řešení. Zatím co v předešlé konstrukci byl pohyb hlavy uskutečňován pomocí závitových tyčí, při reálné konstrukci vyšlo najevo, že krokový motor bude řízen impulzy o frekvenci přibližně 100 Hz (tzn. jedno otočení hřídele motoru za vteřinu). Při stoupání tyče 0,8 mm na otočku, bude kreslení velmi přesné, ovšem naprosto nedostačující po stránce časové. Zvolili jsme proto jinou variantu. Pohyb hlavy je v jednotlivých osách řešen pomocí ozubeného řemene a odpovídající řemenice. Toto řešení je oproti řešení se závitovými tyčemi technicky složitější, protože bylo nutné naproti motoru řemen uchytit pomocí pružiny kvůli nepřesnostem ve vrtání řemenic. Nevýhodou toho řešení je i to, že je zde další pohyblivá část a tím pádem další možnost nepřesnosti. Naopak velikou výhodou je rapidní úspora času. Čára rovnoběžná s jednou z os (je aktivní jen jeden motor) dlouhá 10 cm tímto způsobem trvá nakreslit přibližně 5.5 sekund místo původních 200s.

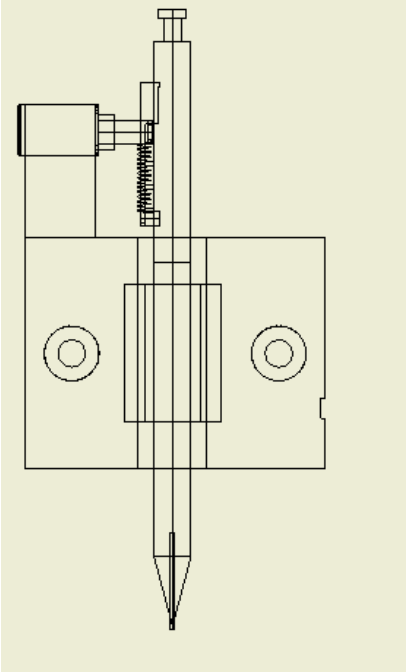
Zakoupil jsem dva řemeny, jeden o délce 1263 mm a druhý o délce 822 mm. Původně jsem chtěl objednat metráž řemene, ovšem poté se ukázalo, že je možné zakoupit vždy nejbližší větší (delší) smyčku. Profil zubů je HTD a jsou vyrobeny z neoprenu.

2.4. Kreslící hlava

Je vyrobena z dřevěného kvádru o rozměrech 51 x 65 x 40 mm, do kterého je zasazeno celkem 5 lineárních ložisek. Tři z těchto ložisek mají vnitřní průměr 6mm a jsou užity pro jeden ze směrů posuvu. Další dvě ložiska mají za úkol držet kreslící nástroj a dovolovat mu pohyb pouze ve třetí ose (kolmé na kreslící plochu). Původní řešení konstrukce bylo myšleno tak, že kreslící nástroj společně s dvěma otočně připevněnými pákami a hlavou bude tvořit čtyřkloubový mechanismus. Kvůli velkému počtu pohyblivých částí by bylo složité udržet požadovanou přesnost, a tak jsme od tohoto návrhu museli upustit a řešit úlohu snadněji.

Kreslení je realizováno tak, že v trubce s upevňovacími šrouby je kreslící nástroj (tužka, rýsovací pero, fixa) a trubka je ovládána servomotorem přes pružinu, aby se zamezilo tomu, že by nástroj díky nerovnosti (i když minimální díky úpravě povrchu kreslicí desky) vynechával nebo se naopak zarýval do papíru nebo dokonce do kreslící desky resp. jejího povrchu.

Součástí kreslící hlavy je také servomotor.



Obr. 3: Kreslící hlava

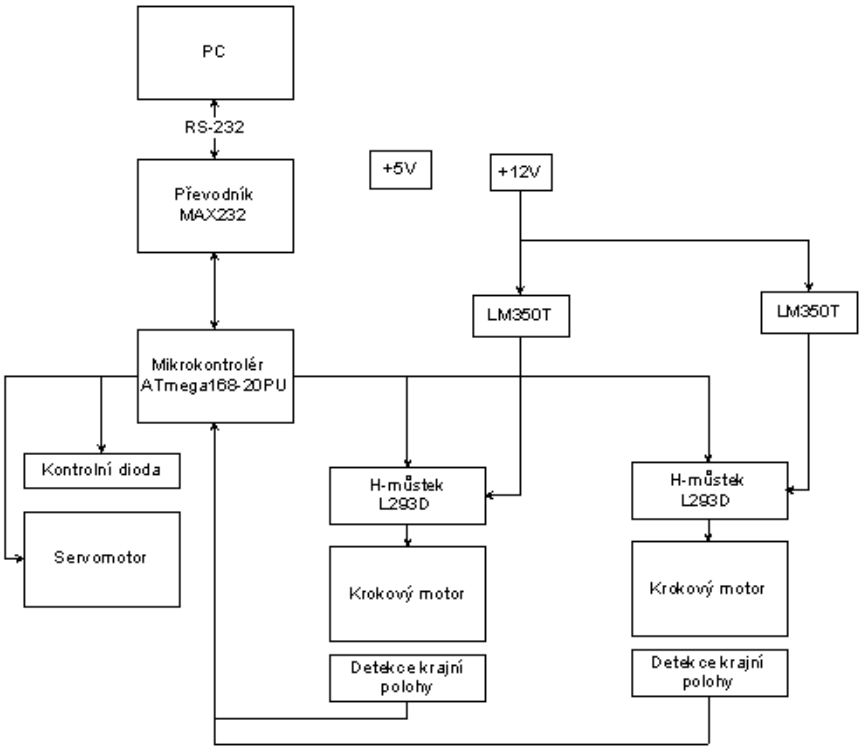
3. Hardware

3.1. Ovládací obvody

Základem ovládacích obvodů je jednočipový mikropočítač ATmega168-20PU od firmy Atmel, dále jen mirokontrolér. Rozhodli jsme se pro něj, kvůli snadnému programování, dostatečnému počtu vstupně výstupních pinů, integrovanému synchronnímu / asynchronnímu sériovému rozhraní USART, které zjednodušuje sériovou komunikaci s počítačem, šestnácti kilobitovou, programovatelnou flash paměť a taktovací frekvenci až 20 MHz.

Protože obvod pracuje s napěťovými úrovněmi nula až pět volů a RS-232 (sériová linka) od mínus patnácti do plus patnácti voltů, je potřeba komunikaci převést pomocí integrovaného obvodu MAX232, který hodnoty napětí mezi těmito úrovněmi převádí a potřebná napětí si sám vytváří z napájení pěti volty.

Mirokontrolér dále ovládá krokové motory pomocí dvou dvojitých h-můstků, přes které pomocí TTL napětí spíná jednotlivé fáze krokového motoru ke zdroji proudu a zemi. A generuje signál, pomocí kterého ovládá servomotor pohybující s uchycením kreslícího nástroje.



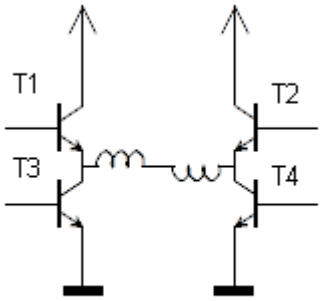
Obr. 4: Blokové schéma ovládacích obvodů

3.2. Mirokontrolér

Jako základní ovládací prvek jsme vybrali mirokontrolér ATmega168-20PU, který je napájen pěti volty a je nataktován vnitřním 8 MHz RC oscilátorem, který umožňuje dostatečný výpočetní výkon a zároveň umožňuje nastavení komunikační rychlosti USART ve velkém rozsahu hodnot od 2400 bps do 1 Mbps. Díky malému datovému toku a potřebě

kvalitního spojení jsme zvolili nižší rychlost 9600 bps. Plošný spoj obsahuje externí krystal s frekvencí 18,432 MHz, který v této verzi využit.

3.3. H-můstek



Obr. 5: Zjednodušené schéma H-můstku

H-můstek zajišťuje, aby proud cívkami motoru mohl téct oběma směry. Při otevření tranzistorů T1 a T4 teče proud cívkami zleva doprava a při otevření tranzistorů T2 a T3 zprava doleva. L293D obsahuje navíc ochranné diody chránící tranzistory proti velkému napětí indukujícímu se na cívkách motorů při odpojení napětí.

3.4. Krokový motor

Zkladní princip krokového motoru je, že proud procházející cívkou statoru vytvoří magnetické pole, které přitáhne opačný pól magnetu rotorů. Postupným přepínáním cívek dosáhneme vytvoření rotujícího magnetického pole, které roztočí rotor.

Výhoda krokového motoru spočívá v tom, že se dá, na rozdíl od klasických motorů, přesně zjistil úhel otočení osy podle počtu přepnutých cívek.

Námi zvolené krokové motory měly možnost zapojení unipolární i bipolární se sériovým nebo paralelním zapojením cívek. Pro větší krouticí moment a požadavek spíše nižších otáček, jsme vybrali bipolární zapojení se sériově zapojenými cívkami. To znamená, že do protějších cívek je vždy pouštěn proud různými směry (na jedné straně je rotor přitahován k cívce a na druhé je odpuzován).

3.5. Ovládání a napájení krokových motorů

Abychom zdvojnásobili počet kroků a tím zvýšili přesnost, budeme krokový motor ovládat s polovičním krokem. Krokový motor je s mirokontrolérem spojen přes čtyřkanálový H-můstek L293D.

Jako napájení jsme zvolili zdroj proudu, který zkrátí čas přechodových jevů a tím se sníží šance ztráty kroku a zvýší rychlost tisku plotteru, vytvořený pomocí L350T. Je nastavený tak, aby dával proud 600 mA, což je maximální proud H-můstkem.

3.6. Servomotor

Servomotor je ovládán pomocí pulzu, generovaného mirokontrolérem, o délce 600 až 2400 mikrosekund a s frekvencí 50 Hz, kde 600 mikrosekund odpovídá natočení vlevo, 1500 natočení na střed a 2400 natočení vpravo.

3.7. Detekce krajní polohy

Detekce krajní polohy je prováděna pomocí mikrospínače připojeného k mikrokontoléru přes rezistor v řádu kiloohmů. A pin mikropočítače je dále uzemněn pomocí rezistoru v řádu desítek kiloohmů, kvůli šumu.

3.8. Světelná signalizace

Dvě světlo emitující diody signalizují funkční napájení. Zelená plus pěti volty a oranžová plus dvanácti volty. Červená dioda blikající jednou za vteřinu signalizuje správnou funkci mikrokonroléruSoftware

3.9. Datová komunikace

K datové komunikaci je použita sériová linka, kvůli jednoduchosti použití na počítači, tak i na čipu. Microsoft Visual Basic, ve kterém je napsán program pro počítač, obsahuje komponentu, která komunikaci usnadňuje a mikrokontrolér obsahuje USART, který přijímá a odesílá data. Stačí nastavit komunikační rychlosti, počet přenášených bitů, parity (doplnění na lichý nebo sudý počet “jedniček”), počet stop bitů a na počítači název portu.

3.10. Komunikační protokol

Komunikace je zcela synchronní, to znamená, že počítač odešle správu a plotter odešle odpověď. Pro navazování spojení se z počítače posílá 0xED dokud plotter nedopoví, to se děje i při ztrátě spojení, každá jiná zpráva začíná start bajtem (0xEF), následuje identifikátor zprávy, popřípadě data zprávy a kontrolní součet.

0xEF	C	ID	D	KS

0xEF	start bajt
C	číslo zprávy
ID	identifikátor zprávy
D	data zprávy
KS	kontrolní součet

Tabulka 1: Strukrura zprávy

Start bitem začíná každá zpráva, s výjimkou navazování spojení. Následuje číslo zprávy, které se zatím používá jen na zajetí do počátku souřadnic, identifikátor, který identifikuje konkrétní zprávu a případně data zprávy. Na konci je přidán kontrolní součet, který se počítá jako exkluzivní disjunkce všech předchozích dat.

ID	Počet bajtů dat	Význam dat	Příkaz
1	0	žádný	zajeď do počátku souřadnic
2	4	první dva bajty souřadnice x, druhé dva bajty souřadnice y	přesuň se na danou polohu
3	4	první dva bajty souřadnice x, druhé dva bajty souřadnice y	nakresli úsečku z aktuální polohy na danou polohu

Tabulka 2: Identifikátory používané počítačem

ID	Počet bajtů dat	Význam dat	Příkaz
10	0	žádný	poslední příkaz přijata
11	0	žádný	poslední příkaz splněn

Tabulka 3: Identifikátory používané plotterem

3.11. Program počítače

Po spuštění se zobrazí okno programu. Po kliknutí na tlačítko Procházet se zobrazí dialogové okno s možností vybrání souboru dxf. Po vybrání vybrání souboru se zpřístupní tlačítko Načíst.

Po kliknutí na tlačítko se začne procházet soubor dxf dokud se nenarazí na řádek s ENTITES, za kterým se nachází informace o objektech. Spustí se cyklus, který čte soubor, dokud nenarazí na řádek ENDSEC a ukládá všechny úsečky, kružnice a oblouky ležící v rovině XY. Po ukončení cyklu se v levém dolním rohu zobrazí popisek s adresou načteného souboru. Když je celkový počet objektů nenulový zpřístupní se tlačítko pokračovat.

Po kliknutí na tlačítko „pokračovat“ se zkryje „výběr“ a načítání souboru a zobrazí se bílý panel, kde se vykreslí tisknutý soubor, tlačítko „začítů a výběrové pole s popiskem, kde bude možno vybrat komunikační sériový port. Pokud ale počítač neobsahuje žádné sériové porty, tak se zobrazí se chybová hláška. Dále se oblouky a posléze kružnice převedou pomocí analytické geometrie na daný počet krátkých úseček. Poté se úsečky přesahující rozměry papíru A4 zkrátí, aby se na ní vešly. Tím pádem mají všechny úsečky mimo papír velikost nula. Dále se souřadnice nenulových úseček převedou z milimetrů na počty kroků krokových motorů.

Po kliknutí na tlačítko „začít“ se otevře vybraný sériový port a nastaví se jeho parametry. V případě chyby se zobrazí chybová hláška. Odešle se zpráva o zajetí do počátku souřadnic a spustí se časovače navazovaniSpojeni, který kontroluje, zda nevypršel časový limit odpovědi na zprávu, a timer1, který každých sto milisekund pouští funkci prijemDat, protože se stávalo, že i když po každém přijetí dat se tato funkce spouští, nepřečetla se poslední data.

3.11.1. Příchod dat

Po každém příchodu dat se tato data uloží do bufferu a spustí se funkce prijemDat.

3.11.2. Zpracování dat – funkce prijemDat

Dále se zjistí, co je první v bufferu a tato hodnota se předá přepínači.

Pokud je hodnota 237, plotter odpověděl na navazování spojení a spustí se funkce odesilaniDat s parametrem True (pravda).

Pokud je hodnota 239, zjistí se, zda jsou v bufferu další čtyři bajty zprávy. Pokud ne funkce skončí. Pak se provede kontrola kontrolního součtu celé zprávy, když vyjde něco jiného, než nula, první čtyři bajty z bufferu se odstraní a funkce skončí. Dále, pokud je druhý bajt zprávy deset, je to odpověď na odeslanou zprávu, že dorazila pořádku, zastaví se časovač kontrolující časový limit odpovědi na zprávu. Pokud je ale druhý bajt zprávy jedenáct znamená to, že příkaz ve zprávě byl splněn a uloží se aktuální pozice a zpustí se funkce odesilaniDat s parametrem False (nepravda).

3.11.3. Odesílání zpráv - funkce odesilaniDat

Tato funkce má jeden parametr určující, zda se neobjevila chyba v komunikaci. Pokud se chyba objevila, poslední zpráva se odešle znovu a funkce se ukončí. Pokud ne, zjistí se, zda je pero na začátku další čáry, pokud tam není, odešle se zpráva s příkazem, aby tam přejelo. Pokud tam již je, odešle se zpráva, aby se nakreslila další úsečka.

Pak se zobrazí, kolik procent tisku je hotovo a překreslí se panel, kde se zobrazuje aktuální stav tisku.

3.11.4. Zobrazování aktuálního stavu tisku

Aktuální stav tisku se vykresluje tak, že se projde celé pole s úsečkami a každá se vykreslí. Vykreslené úsečky se zobrazí černě, úsečka v tiku červeně a nevytisknuté šedě.

3.12. Mikrokontrolér

Program pro mikrokontrolér je napsaný v programovacím jazyce C a jako vývojové prostředí jsme použili AtmelStudio 6.1.

3.12.1. Čítače

Mikrokontrolér obsahuje tři čítače, dva osmibitové, čítač nula a dva, a jeden šestnáctibitový. Čítač nula bliká červenou led diodou a tím signalizuje, že se mikrokontrolér nezasekl a posílá “šum”, protože se stávalo, že se počítač zasekl při zpracovávání dat a tímto “šumem” se problém vyřešil. Čítač jedna se stará o generování řídícího signálu pro servomotor, ovládající pero. Čítač dva pohybuje jedním motorem při kreslení čáry.

3.12.2. Hlavní program mikrokontroléru

Po spuštění napájení se zahájí hlavní program mikrokontroléru. Nejprve se nastaví jednotlivé piny na vstupní nebo výstupní.
Pak se inicializuje USART. Nastaví se u něj rychlost komunikace. Pin PD0 se nastaví na odesílání dat (TXD - Transmit Data) a PD1 na příjem dat (RXD - Receive Data).

V dalším kroku se nastaví časovač nula, povolí se přerušení a nastaví se předdělička na hodnotu 1024. Dále se povolí přerušení od časovače jedna a nastaví se do takzvaného CTC módu (Clear Timer on Compare Match). To znamená, že časovač čítá, dokud jeho hodnota není rovna registru OCR1A a pak vyvolá přerušení a nastaví se na nulu. Předdělička časovače jedna se nastaví na hodnotu 8. Pak se nastaví registr OCR1A. Protože je čítač jedna šestnáctibitový a registry mají velikost osmi bitů, je registr OCR1A rozdělen na OCR1AH vyšších osm bitů, a OCR1AL nižších osm bitů, aby pokryl celý rozsah čítače. Pak se nastaví předdělička čítače dva na hodnotu 64 a rozsvítí se červená dioda. Počká se jednu vteřinu a tím se signalizuje, že je čip připraven. Následně se povolí globální přerušení a spustí se všechny čítače. Zahájí se nekonečný hlavní cyklus.

V něm se nachází přepínač (switch), který zpracovává data z USARTu. Data se načítají pomocí funkce uart_getc a odesílají pomocí uart_putc.

Nejprve, ještě před přepínačem, se do proměnné start načte první bajt z bufferu USARTu a přepínač skočí na návěstí (case) s touto hodnotou. Pokud je hodnota proměnné 237 (navázání spojení) odešle se tato hodnota zpět a pomocí příkazu break se vyskočí z těla přepínače a program pokračuje tímto blokem. Pokud je hodnota 239, načte se proměnná cisloPrikazu a proměnná id a spustí se další vnořený přepínač s argumentem id. Tento přepínač obsahuje tři návěstí.

Návěstí s hodnotou jedna. Načte se proměnná ks. Když se exkluzivní disjunkce proměnných start, cisloPrikazu, id a ks liší od nuly, tak byla data při přenosu poškozena a pomocí příkazu break se vyskočí z přepínače. Pokud vyjde výsledná hodnota nula, pokračuje se výpočtem kontrolního součinu odpovědi počítači, že příkaz byl přijat. Do proměnné ks se dopočítá kontrolní součet následující zprávy. A odešle se zpráva ve formátu 239, cisloPrikazu, 10, ks. Pak se spustí funkce pocatekSouradnic, která zajede kreslící hlavou do počátku souřadnic. Po skončení funkce odešle zprávu (239, cisloPrikaz, 11, ks), že byl příkaz splněn a pomocí break vyskočí z přepínače.

Návěstí s hodnotou dva. Načtou se proměnné x1, x2, y11, y2 a ks. Zkontroluje zprávu stejným způsobem jako v předchozím návěstí a popřípadě odešle odpověď o přijetí příkazu. Dále spustí funkci přejezd, která přesune kreslící hlavu na dané souřadnice. Poté odešle zprávu, že byl příkaz splněn a pomocí break se vyskočí z přepínače.

Návěstí s hodnotou tři. Načte proměnné x1, x2, y11, y2 a kontrolní součet. Provede kontrolu a popřípadě odpoví, že zpráva byla přijata. Spustí funkci tisk. Po dokončení se jako v předchozích případech odešle odpověď počítači a pomocí příkazu break se vyskočí z vnořeného přepínače. Po provedení jednoho z návěstí se pomocí dalšího příkazu break vyskočí z prvního přepínače a cyklus skočí na začátek.

3.12.3. Funkce

V programu se nacházejí tři funkce sloužící pro tisk. I když by nebyly potřeba, protože se každá pouští pouze jednou, jsou zde kvůli zpřehlednění kódu. Dvě funkce pro datovou komunikaci pomocí USARTu s počítačem, dvě funkce pro pohyb s motory a jedna na počítání absolutní hodnoty.

3.12.3.1. Funkce pro tisk

3.12.3.1.1. Funkce pocatekSouradnic

Tato funkce se stará o přejezd kreslící hlavy do počátku souřadnic. Tím se zajišťuje, aby tisk začínal stále ze stejného místa a na konci tisku se zajelo na stranu.

První se zkontroluje, jestli je zvednuté pero. Pokud ne, tak se zvedne a počká se sto milisekund, aby servomotor stihl zareagovat. Potom se spustí cyklus, který běží, dokud nejsou stisknuty oba spínače v krajní poloze. V cyklu jsou dvě podmínky. Pro každý motor jedna. V každé podmínce se kontroluje, zda již není nějaká osa v krajní poloze (není sepnutý spínač), pokud ne, tak se provede jeden krok směrem k počátku souřadnic. Poté se čeká deset milisekund, aby motory a spínače stihly zareagovat. Když sepnou oba spínače, cyklus se přeruší a proměnné x a y se dočasně nastaví na nulu. Poté se spustí druhý cyklus, který běží, dokud je x nebo y menší než nastavená pozice počátku v proměnných pozicePocatkuX a pozicePocatkuY. V cyklu se pomocí dvou podmínek, pro x a y, kontroluje, zda je daná souřadnice menší než uložená v proměnné pozicePocatkuX nebo Y. Pokud je menší, pomocí funkce motorX nebo Y se udělá jeden krok a do proměnné x nebo y se přičte jedna. Po vyhodnocení podmínek se počká deset milisekund, aby mohly motory zareagovat. Když se x a y budou rovnat pozicePocatkuX a Y, tak se cyklus ukončí. Do proměnných x a y, které jsme používali jako pomocné, se uloží nuly.

3.12.3.1.2. Funkce prejezd

Tato funkce se stará o přejezd na dané souřadnice bez kreslení čáry. To znamená, že přejíždí z aktuální polohy na polohu zadanou příkazem z počítače.

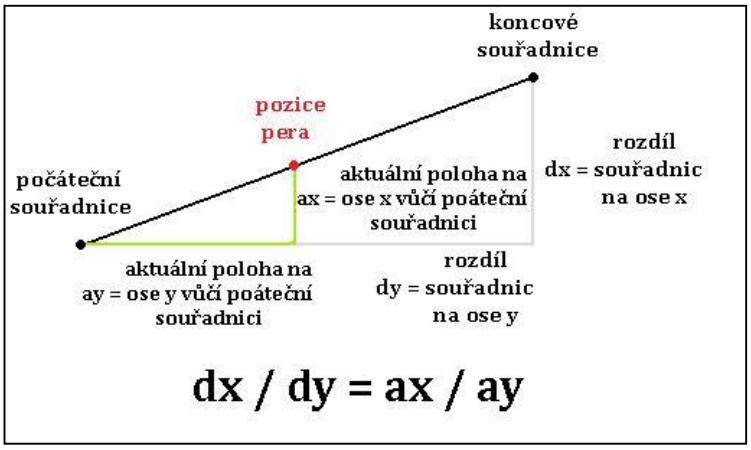
První zkontroluje, zda je zvednuté pero. Pokud není, zvedne ho a počká sto milisekund, aby mohl servomotor zareagovat. Protože rozlišení plotteru je 0,09 milimetru, vychází rozsah souřadnic na papíru A4 nula až tři tisíce tři sta. Proto musí být požadované souřadnice odeslány ve dvou osmi bitových bajtech, které se sloučí do poměných xp a yp z poměných x1, x2, y11 a y2 tak, že se x1 nebo y11 posune o osm bitů doleva a pomocí disjunkce se sloučí s x2 nebo y2. Tím vznikne šestnácti-bitová proměnná, kde prvních osm bitů proměnné je x1 nebo y11 a druhých osm bitů x2 nebo y2. Pak se zjistí, zda souřadnice jsou větší nebo menší než aktuální souřadnice pera. Když je x nebo y menší než xp nebo yp, tak se do zx nebo zy uloží hodnota jedna a když je větší, tak hodnota mínus jedna. Tím jsme zjistili směr pohybu pera v jednotlivých osách. Poté se spustí cyklus, který běží, dokud se požadovaná souřadnice uložená v xp a yp nerovná reálné souřadnici uložení v x a y. Uvnitř cyklu se kontroluje pomocí dvou podmínek, zda již není hodnota souřadnic pro dané osy rovna požadované souřadnici. Pokud ne, tak se pootočí motorem pomocí funkce motorX nebo Y o hodnotu uloženou v proměnných zx nebo zy, což je vlastně směr tisku. Hodnota zx nebo zy se přičte do x nebo y. Následně se počká dvanáct milisekund, aby motory mohly zareagovat.

3.12.3.1.3. Funkce tisk

Funkce tisk slouží k tisku úsečky z aktuálních souřadnic na požadované souřadnice. Protože je tato funkce složitější, nejprve ji popíšeme všeobecně.

Tato funkce využívá podobnosti trojúhelníků. Poměr rozdílu počátečních a koncových souřadnic se rovná poměru rozdílu počátečních a aktuálních souřadnic. Protože to není v reálu možné, kvůli celým krokům motoru, alespoň se snaží co nejvíce přiblížit úsečce zespoda.

Pomocí přerušení se pohybuje v pravidelných intervalech motorem na delší ose a v programu se dopočítává, zda není pero pod úsečkou. Když je, udělá se krok směrem ke koncové souřadnici.



Obr. 6: Kreslení úseček

Podrobný popis programu. Nejprve se deklarují dvě reálné (float) proměnné, konstanta a pomocná a vynulují se proměnné ay a ax. Pokud je zvednuté pero, položí se a počká se sto padesát milisekund.

Pak se mírně přitlačí na dvanáct milisekund, aby se zlepšila kvalita při tisku navazujících čar.

Spočítají se požadované souřadnice a směr tisku jako ve funkci prejezd. Do proměnných dx se uloží absolutní hodnota rozdílu aktuální polohy a požadované polohy, což je délka úsečky v ose x. To samé se spočítá pro osu y. Dále se zjistí, zda je dx větší než dy. Pokud ano, hodnota proměnné XJeVetsiNezY se nastaví na jedna, aby se v přerušení vědělo, jakou osou hýbat, a do proměnné konstanta se uloží poměr dy/dx. Následně se zakáže globální přerušení, kvůli tomu, že proměnné aktuální polohy se mění i v přerušení a protože reálné číslo je uloženo ve více bajtech. Tím pádem výpočet s nimi trvá více kroků a mohlo by se stát, že při probíhajícím výpočtu, by se v přerušení změnily a výsledek by byl napůl spočítán ze staré a nové hodnoty. Poté se povolí přerušení od čítače dva a spustí se smyčka, která běží, dokud se ax nerovná dx nebo dokud se ay nerovná dy. V cyklu se do proměnné pomocna uloží poměr ay a ax a povolí se globální přerušení, protože dále už se nepočítá s vícebajtovými čísly. Pokud je konstanta větší než pomocna, tak se pomocí funkce motorY posune pero o zy a proměnná ay se zvětší o jedna, a k proměnné y se přičte zy. Počká se dvanáct milisekund. Tato hodnota je zvolena tak, aby motory stihly zareagovat, a aby byla menší než čas jednoho kroku druhého motoru v přerušení, a aby se stíhala dopočítávat hodnota proměnné a dokrokovalo se co nejblíže k úsečce. Poté se znova zakáže globální přerušení a cyklus se opakuje. Pokud ale je na začátku dx menší než dy, do proměnné XJeVetsiNezY se uloží nula. Dále je vše stejné, jen poměry se počítají převráceně a hýbe se s druhým motorem. Druhá část funkce se odehrává pomocí přerušení čítače dva. Když nastane přerušení, sníží se hodnota proměnné citacTimeru2 o jedna a pokud se výsledná hodnota liší od nuly, přeruší se. Pokud je ale výsledná hodnota nula, zjistí se pomocí proměnné XJeVetsiNezY, jakým motorem se má hýbat. Pokud je hodnota jedna, spustí se funkce motorX a ta pohne motorem o zx, hodnota proměnné ax se zvětší od jedna a k proměnné x se přičte zx a zkontroluje se, zda ax se nerovná dx. Pokud ano,

přerušení od čítače se vypne, protože jsme na konci úsečky. Pokud ale byla hodnota proměnné XJevesiNezY nula, provede se to samé, ale pro osu y. Nakonec se nastaví hodnota citacTimeru2 na šest, aby jeden krok trval přibližně 12,3 mikrosekund.

3.12.3.2. Funkce pro datovou komunikaci

3.12.3.2.1. Funkce uart_putc

Tato funkce odešle osmi bitovou neznaménkovou proměnou po sériové lince. Nejprve se spustí cyklus, který běží, dokud není datový registr UDR0 prázdný. Pak se do datového registru zapíše proměnná, čímž se odešle.

3.12.3.2.2. Funkce uart_getc

Tato funkce vrací osmibitovou neznaménkovou proměnou z bufferu. Nejprve se spustí cyklus, který běží, dokud není nastaven příznak kompletního příjmu a pak vrátí hodnotu v datovém registru UDR0.

3.12.3.3. Funkce pro pohyb s motory a absolutní hodnota

3.12.3.3.1. Funkce motorX

Tato funkce pohybuje s motorem na ose x. Jako parametr se jí předá, jakým směrem se má pohnout motorem, jedna nebo mínus jedna, a tato hodnota se uloží do lokální proměnné z. Od proměnné poziceMotoru1 se odečte z. Pak se zkontroluje, zda není mimo rozsah pole tak, že se k ní přičte osm. Kdyby byla záporná, vypočítá se zbytek po dělení osmi. Protože na portu C je výstup na ovládání servomotoru, musí se nejdříve nastavit piny ovládající motor na nulové napětí a pak ty, co se mají nastavit na pět voltů podle hodnot uložených v poli a pozici podle proměnné poziceMotoru1, se nastaví pomocí disjunkce. Protože kdyby se jen hodnota pole přiřadila portu C, změnilo i narušilo by se ovládání servomotoru.

3.12.3.3.2. Funkce motorY

Tato funkce pohybuje s motorem na ose y. Jako parametr se jí předá, jakým směrem se má pohnout motorem, jedna nebo mínus jedna, a tato hodnota se uloží do lokální proměnné z.

K proměnné poziceMotoru2 se přičte z. Následně se zkontroluje, zda není mimo rozsah pole tím způsobem, že se k ní přičte osm. Kdyby byla záporná, vypočítá se zbytek po dělení osmi. Protože ale na portu B není žádný jiný výstup, stačí přiřadit hodnotu uloženou v poli na pozici, která se rovná poziceMotoru2.

3.12.3.3.3. Funkce ABS

Tato funkce vrací absolutní hodnotu znaménkové šestnáctibitové proměnné.

3.13. Cena

Dřevo + stavební materiál	79,00 Kč
Lineární ložiska	714,00 Kč
Řemeny HTD-03M-06	404,00 Kč
Řemenice HTD-03M-06	87,00 Kč
Motory SX-17	603,00 Kč
Zdroj 145W	205,00 Kč
Elektronické součástky	488,00 Kč
Programátor	26,00 Kč
Plošný spoj	130,00 Kč
CELKEM	2 736,00 Kč

4. ZÁVĚR

Při výrobě zařízení jako je plotter, které se skládá z relativně velké škály oborů, do kterých zasahuje, se setkáváme s nepřeberným množství problémů. I zdánlivě banální problémy zde mohou mít hned několik možných příčin, od nefunkčního hardwaru (studený spoj) přes chybu programu mikrokontroléru až po chybu komunikace nebo samotného programu v počítači. Náš plotter sice není dost přesný na to, aby byl schopen konkurovat komerčním plotterům, které jsou k dostání na trhu, to ale nebylo naším záměrem. Chtěli jsme si vyzkoušet a v praxi ověřit, jak se plotter konstruuje a programuje jeho ovládání.

5. Seznam použité literatury a zdrojů informací

http://www.kvetakov.net/clanky/avr/59-komunikujeme-uart.html [cit.2014-04-08]

http://www.kulichweb.wz.cz/prog_c.php [cit.2014-04-08]

http://cs.wikibooks.org/wiki/Programujeme_jedno%C4%8Dipy [cit.2014-04-08]

http://forum.mcontrollers.com/viewforum.php?f=8 [cit.2014-04-08]

http://www.sallyx.org/sally/c/c10.php [cit 2014-04-08]

http://robotika.cz/guide/cs [cit.2014-04-08]

http://ok2tej.zbytky.net/rady/skola51.html [cit.2014-04-08]

http://cyber.felk.cvut.cz/research/theses/papers/192.pdf [cit.2014-04-08]

http://robotika.cz/articles/atmega8-board/cs [cit.2014-04-08]

http://lvr.com/serial_ports_dotnet.htm [cit.2014-04-08]

http://noel.feld.cvut.cz/vyu/a2m99mam/index.php/Krokov%C3%BD_motor [cit.2014-04-08]

6. Seznam použitého softwaru

Atmel Studio 6.1 Atmel

AutoCAD 2014 – Angličtina (English) Autodesk

Autodesk Inventor Fusion 2014 Autodesk, Inc.

Diagram Designer

EAGLE 6.4.0 CadSoft Computer GmbH

EAGLE 6.5.0 CadSoft Computer GmbH

Microsoft .NET Framework 4.5.1 Microsoft Corporation

Microsoft Visual Basic 2010 Express

Microsoft Word 2007

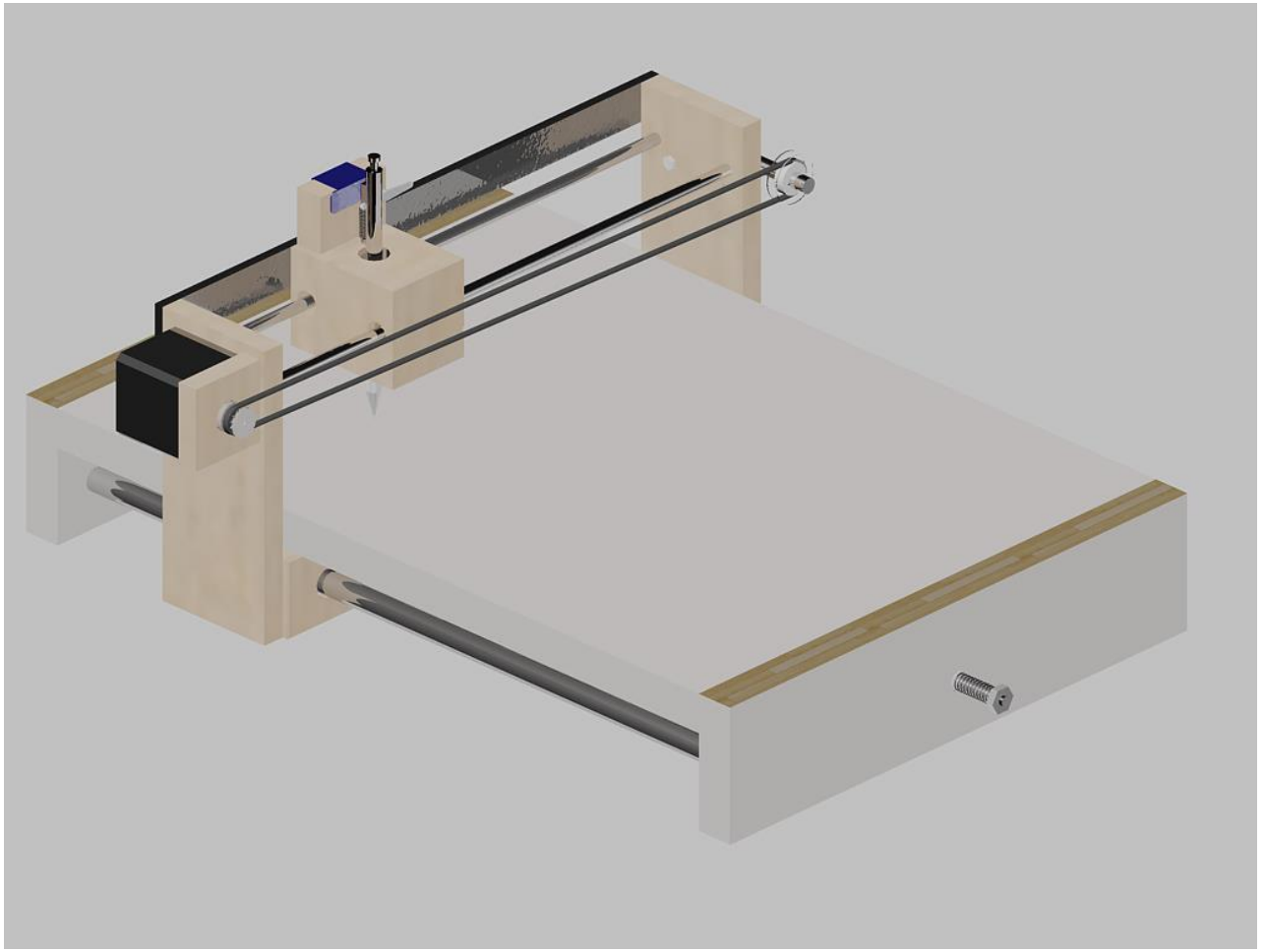
Microsoft Excel 2007

PonyProg 2000

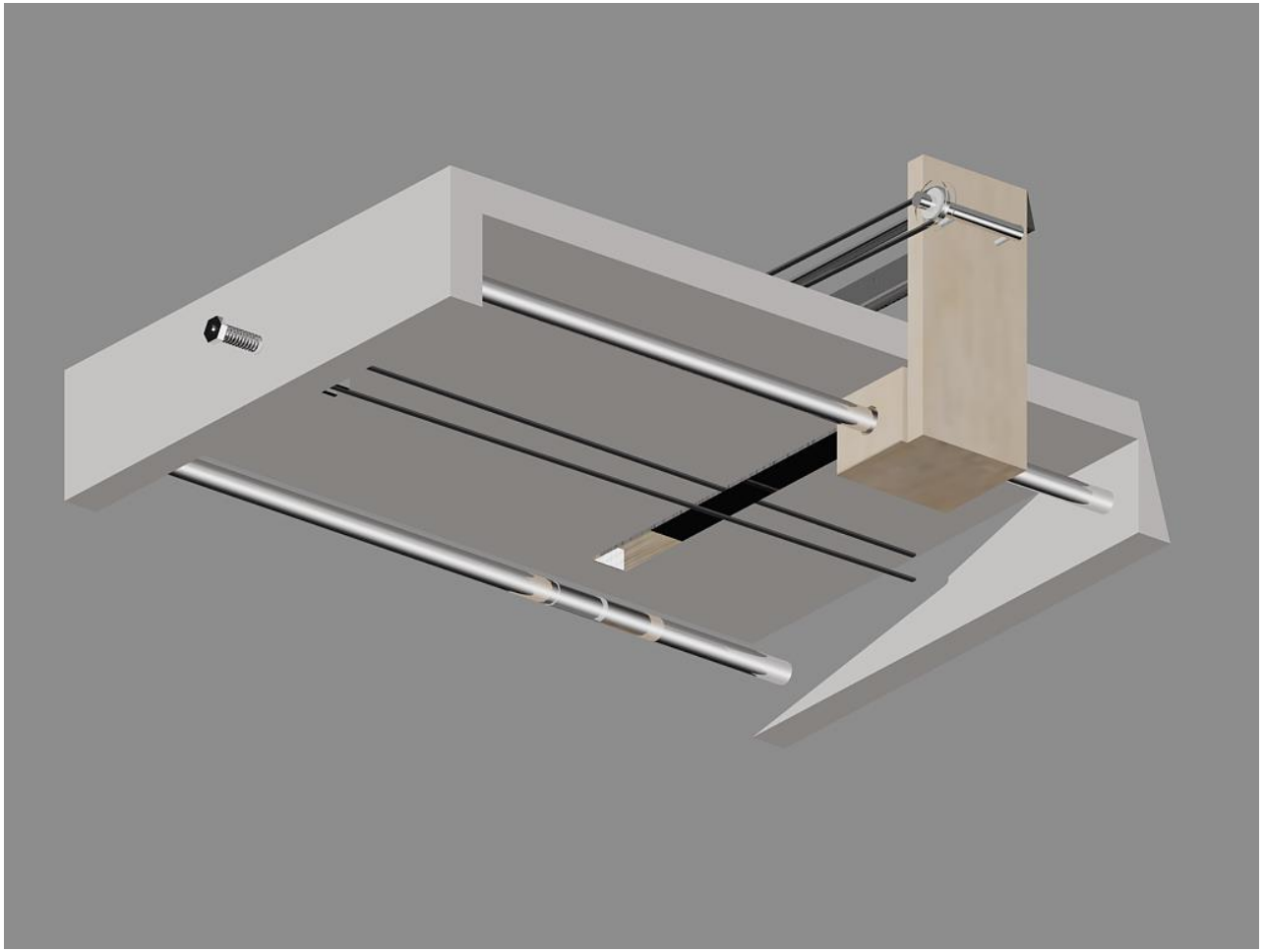
7. Seznam příloh

    1. Plotter – pohled shora
    2. Plotter – pohled zespod
    3. Detail kreslící hlavy
    4. Dřevěná krabice s elektronikou a zdrojem
    5. Fotografie plotteru
    6. Schéma
    7. Plošný spoj
    8. Zdrojový kód počítače
    9. Zdrojový kód plotteru

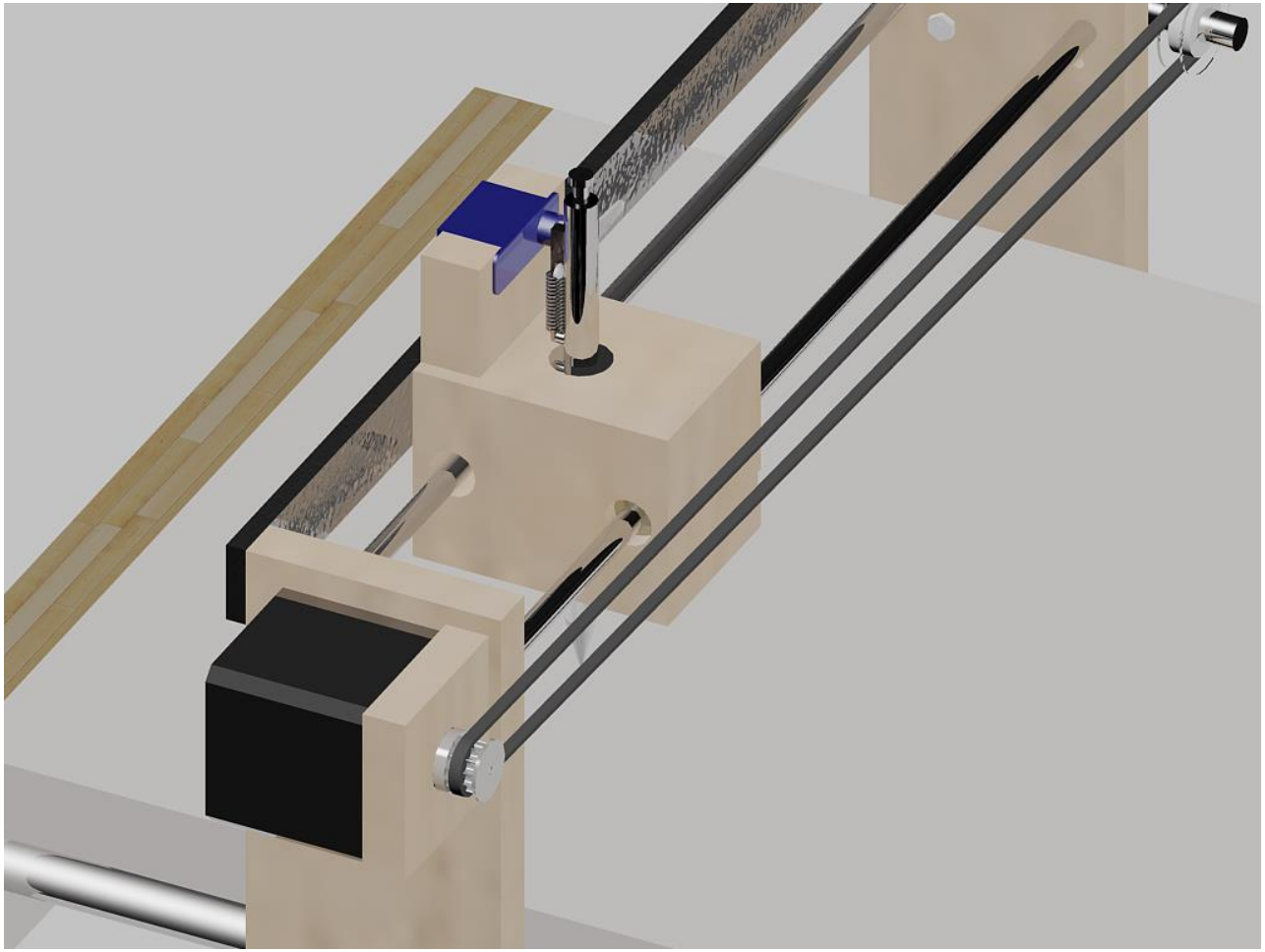
Příloha 1: Plotter – pohled shora



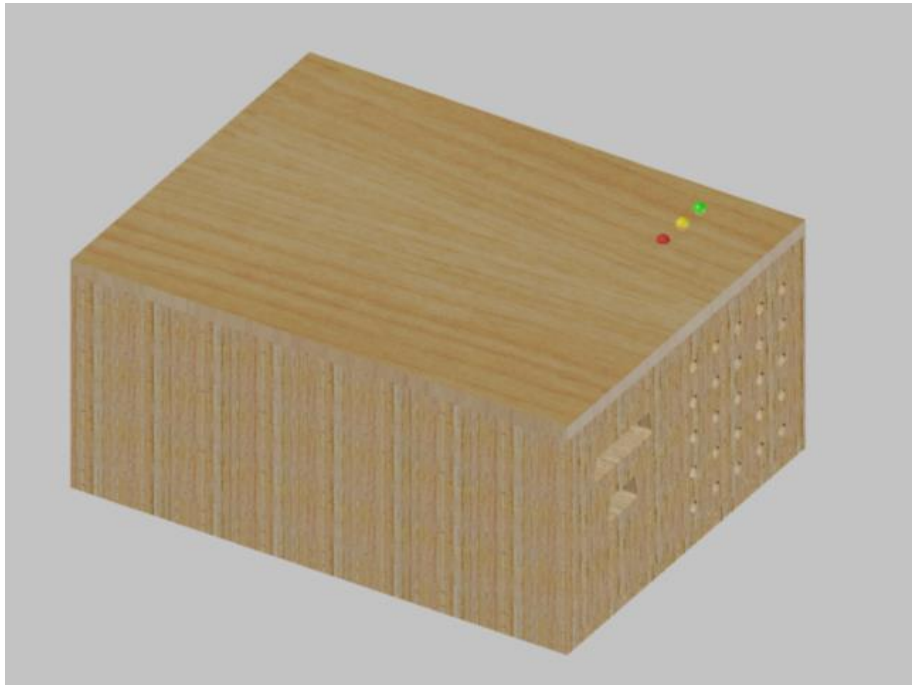
Příloha 2: Plotter – pohled zespodu

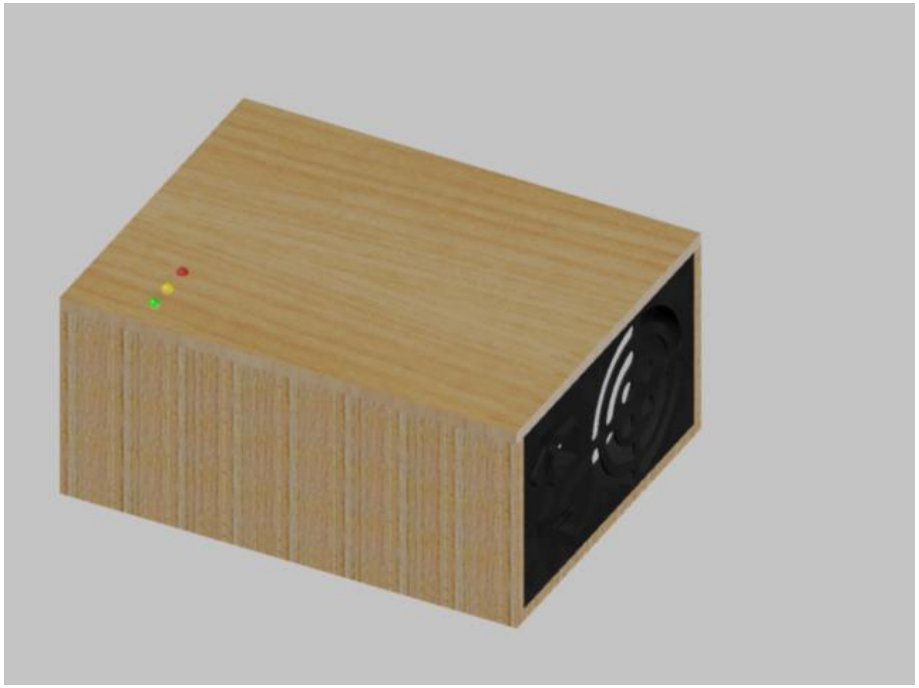


Příloha 3: Detail kreslící hlavy

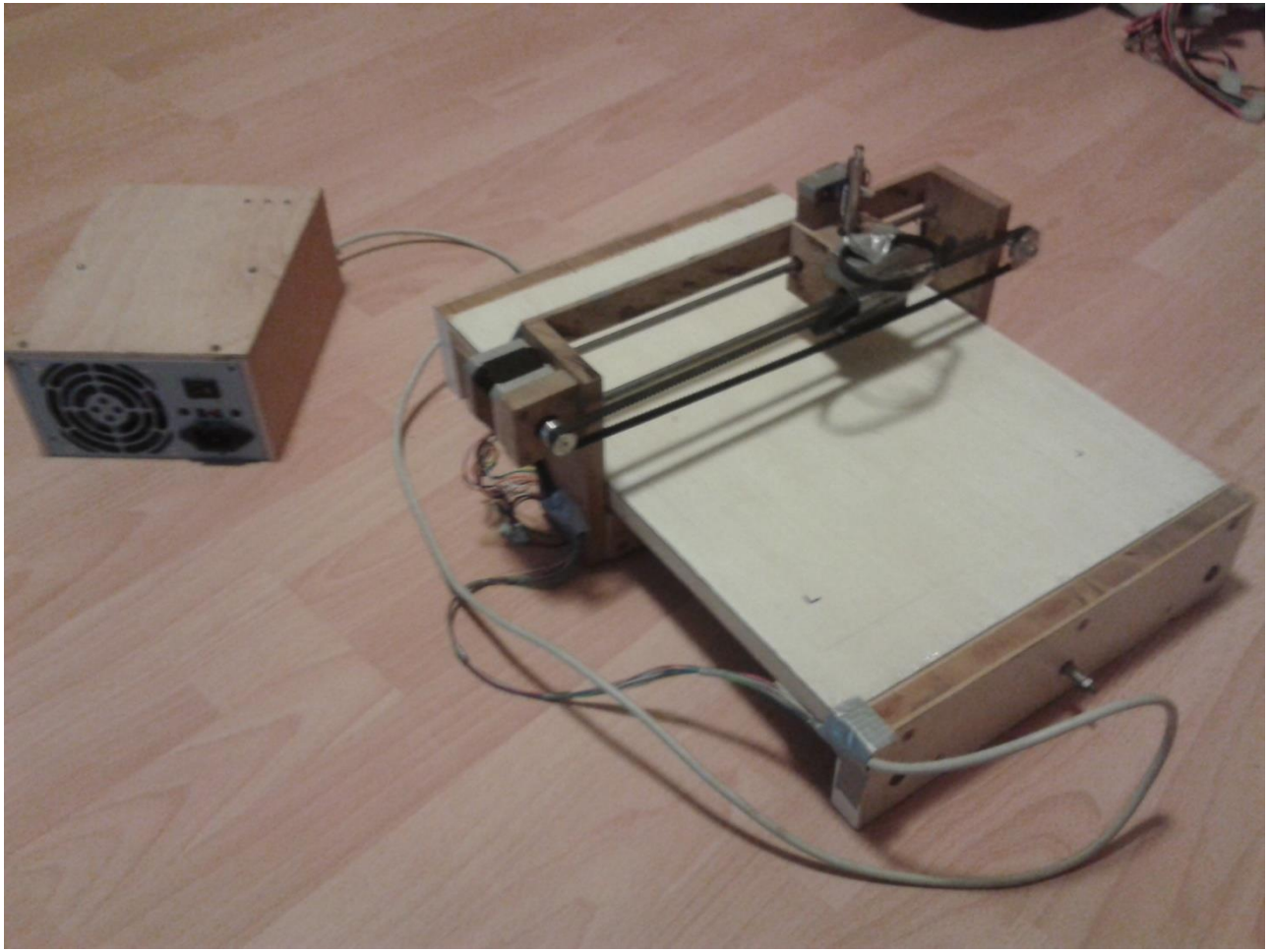


Příloha 4: Dřevěná krabice s elektronikou a zdrojem

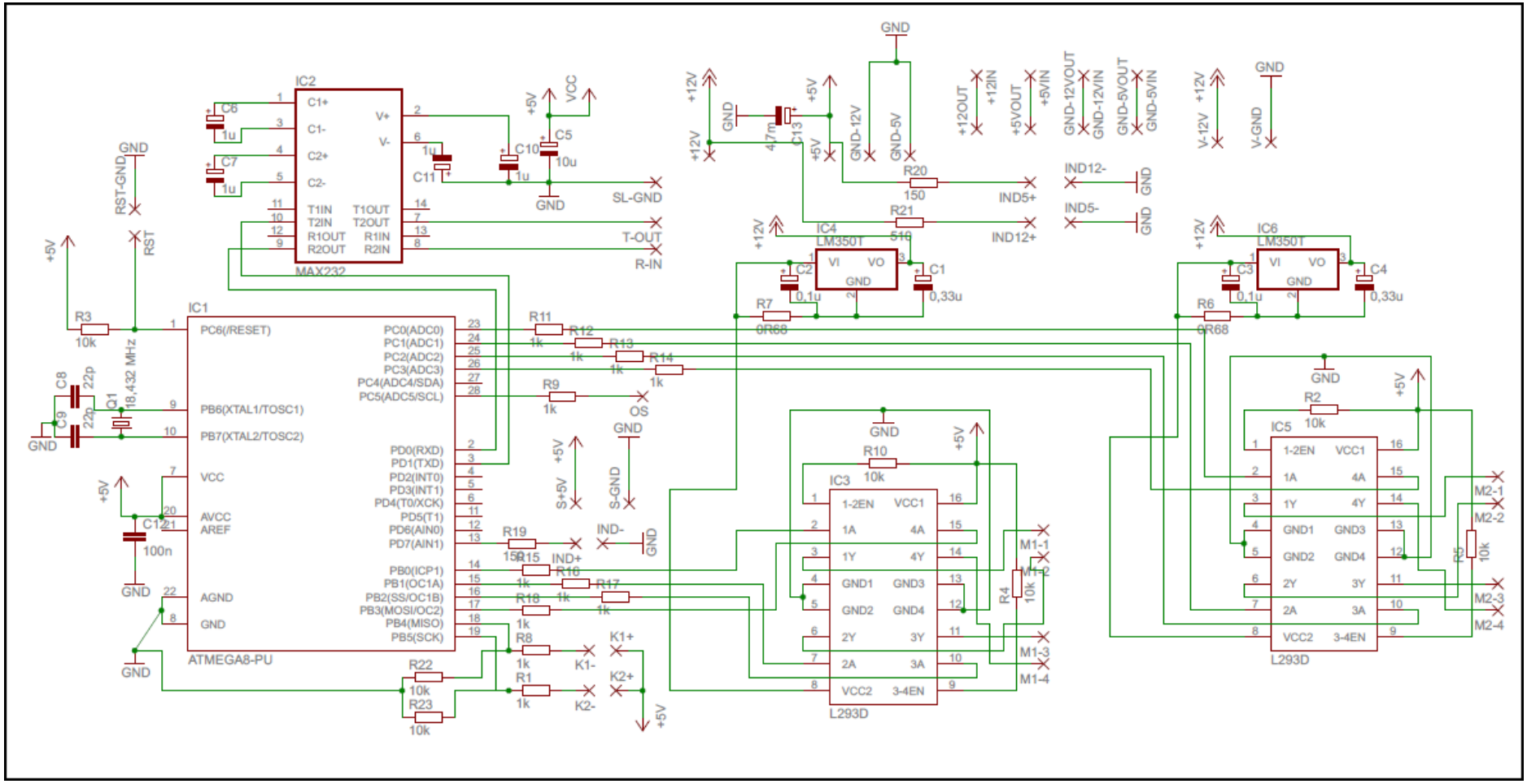




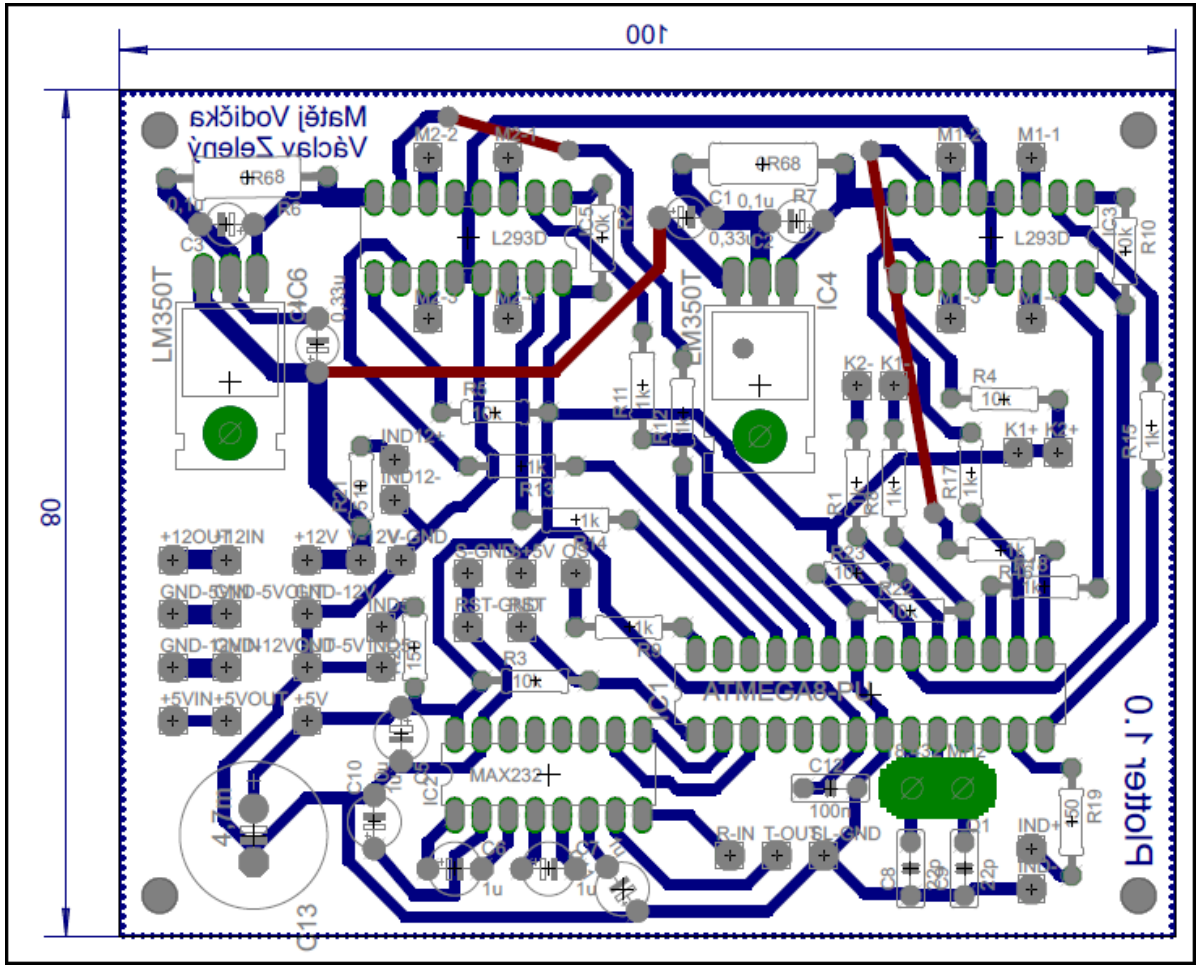
Příloha 5: Fotografie plotteru



Příloha 6: Schéma



Příloha 7: Plošný spoj



Příloha 8: Zdrojový kód počítače

```
Imports System.IO
Imports System.Drawing
Imports System.IO.Ports
Imports System.Threading

Public Class Form1
    Dim cesta As String
    Dim line(4, 0) As Double
    Dim pline(4, 0) As Integer
    Dim circle(3, 0) As Double
    Dim ellipse(7, 0) As Double
    Dim arc(5, 0) As Double
    Dim lwpolyline(2, 0) As Double
    Dim pocetLine As Integer
    Dim pocetLinePuvodni As Integer
    Dim pocetCircle As Integer
    Dim pocetEllipse As Integer
    Dim pocetArc As Integer
    Dim pocetVrcholu As Integer
    Dim pocet As Integer
    Dim radek As String
    Const n As Integer = 145
    Dim k As Integer
    Dim pocetPlatnychLine As Integer
    Dim buffer As List(Of Integer) = New List(Of Integer)
    Dim velikostBufferu As Integer
    Dim cisloPrikazu As Integer = 1
```

```
Dim cisloPrijatyhoprikazu As Integer
Dim ks As Byte
Dim ciloKalibrace As Integer
Dim zprava() As Byte
Dim cisloLine As Integer
Dim x, y As Integer
Dim idPrikazu As Integer
Dim pocitadloCasovace As Integer = 27

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

    Dim cteni As New StreamReader(cesta)

    Do While "ENTITIES" <> radek
        radek = cteni.ReadLine
    Loop

    Do While "ENDSEC" <> radek
        radek = cteni.ReadLine

        Select Case radek
            Case "LINE"
                pocetLine += 1
                Do While True
                    radek = cteni.ReadLine
                    If (line.Length) / 5 <= pocetLine Then
                        ReDim Preserve line(4, pocetLine * 2)
                    End If
                    Select Case radek
```

```
            Case " 10"
               line(1, pocetLine) = cteni.ReadLine.Replace(".", ",")
            Case " 20"
               line(2, pocetLine) = cteni.ReadLine.Replace(".", ",")
            Case " 30"
               If cteni.ReadLine.Replace(".", ",") <> 0 Then
                  pocetLine -= 1
                  Exit Do
               End If
            Case " 11"
               line(3, pocetLine) = cteni.ReadLine.Replace(".", ",")
            Case " 21"
               line(4, pocetLine) = cteni.ReadLine.Replace(".", ",")
            Case " 31"
               If cteni.ReadLine.Replace(".", ",") <> 0 Then
                  pocetLine -= 1
                  Exit Do
               End If
            Case "  0"
               Exit Do
         End Select
      Loop
   Case "CIRCLE"
      pocetCircle += 1
      If (circle.Length) / 4 <= pocetCircle Then
         ReDim Preserve circle(3, pocetCircle * 2)
      End If
      Do While True
         radek = cteni.ReadLine
```

```
        Select Case radek
          Case " 10"
            circle(1, pocetCircle) = cteni.ReadLine.Replace(".", ",")
          Case " 20"
            circle(2, pocetCircle) = cteni.ReadLine.Replace(".", ",")
          Case " 30"
            If cteni.ReadLine.Replace(".", ",") <> 0 Then
              pocetCircle -= 1
              Exit Do
            End If
          Case " 40"
            circle(3, pocetCircle) = cteni.ReadLine.Replace(".", ",")
          Case "210"
            If cteni.ReadLine.Replace(".", ",") <> 0 Then
              pocetCircle -= 1
              Exit Do
            End If
          Case "220"
            If cteni.ReadLine.Replace(".", ",") <> 0 Then
              pocetCircle -= 1
              Exit Do
            End If
          Case "  0"
            Exit Do
        End Select
    Loop

Case "ARC"
    pocetArc += 1
```

```
If (arc.Length) / 6 <= pocetArc Then
   ReDim Preserve arc(5, pocetArc * 2)
End If
Do While True
   radek = cteni.ReadLine
   Select Case radek
      Case " 10"
         arc(1, pocetArc) = cteni.ReadLine.Replace(".", ",")
      Case " 20"
         arc(2, pocetArc) = cteni.ReadLine.Replace(".", ",")
      Case " 30"
         If cteni.ReadLine.Replace(".", ",") <> 0 Then
            pocetArc -= 1
            Exit Do
         End If
      Case " 40"
         arc(3, pocetArc) = cteni.ReadLine.Replace(".", ",")
      Case " 50"
         arc(4, pocetArc) = cteni.ReadLine.Replace(".", ",")
      Case " 51"
         arc(5, pocetArc) = cteni.ReadLine.Replace(".", ",")
      Case "210"
         If cteni.ReadLine.Replace(".", ",") <> 0 Then
            pocetArc -= 1
            Exit Do
         End If
      Case "220"
         If cteni.ReadLine.Replace(".", ",") <> 0 Then
            pocetArc -= 1
```

```
                    Exit Do
                End If
            Case "  0"
                Exit Do
        End Select
    Loop

    End Select
  Loop
  ReDim Preserve circle(3, pocetCircle)
  ReDim Preserve ellipse(7, pocetEllipse)
  ReDim Preserve arc(5, pocetArc)
  Label2.Text = "Načteno: " & OpenFileDialog1.FileName
  If pocetArc + pocetCircle + pocetEllipse + pocetLine > 0 Then
    Button3.Enabled = True
  End If
  Button3.Enabled = True
End Sub

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
  OpenFileDialog1.FileName = ""
  OpenFileDialog1.Filter = "*.dxf|*.dxf"

  If OpenFileDialog1.ShowDialog() = Windows.Forms.DialogResult.OK Then
    cesta = OpenFileDialog1.FileName
    TextBox1.Text = cesta
    Button2.Enabled = True
  End If
```

```
End Sub

Private Sub Button3_Click(sender As System.Object, e As System.EventArgs) Handles Button3.Click
    Label1.Visible = False
    TextBox1.Visible = False
    Button1.Visible = False
    Button2.Visible = False
    Button3.Visible = False
    Panel1.Visible = True
    Button4.Visible = True
    Label4.Visible = True
    ComboBox1.Visible = True
    For Each sp As String In My.Computer.Ports.SerialPortNames
        ComboBox1.Items.Add(sp)
    Next
    If ComboBox1.Items.Count = 0 Then
        ComboBox1.Enabled = False
        MsgBox("Nenalezeny žádné sériové porty", MsgBoxStyle.Information, "Chyba")
    End If
    Dim t, p As Integer
    Dim s1, s2, r, a, b, uo, up, uk, x1, y1, u, du As Double
    pocetLinePuvodni = pocetLine
    t = 1
    p = 1

    Do While p <= pocetArc
        s1 = arc(1, p)
        s2 = arc(2, p)
        r = arc(3, p)
```

```
    up = arc(4, p)
    uk = arc(5, p)
    u = (uk - up + 360) Mod 360
    k = Int(u / 360 * n) + 1
    du = u / k

    ReDim Preserve line(4, pocetLine + k)
    Do While t <= k
        u = (up + (t - 1) * du) / 180 * Math.PI
        line(1, pocetLine + t) = s1 + r * Math.Cos(u) '2 / k * t * Math.PI
        line(2, pocetLine + t) = s2 + r * Math.Sin(u)
        If t <> 1 Then
            line(3, pocetLine + t - 1) = s1 + r * Math.Cos(u)
            line(4, pocetLine + t - 1) = s2 + r * Math.Sin(u)
        End If
        If t = k Then
            line(3, pocetLine + t) = s1 + r * Math.Cos((up + t * du) / 180 * Math.PI)
            line(4, pocetLine + t) = s2 + r * Math.Sin((up + t * du) / 180 * Math.PI)
        End If
        t += 1

    Loop

    pocetLine += k
    t = 1
    p += 1
Loop
p = 1
t = 1
```

```
Do While p <= pocetCircle
    s1 = circle(1, p)
    s2 = circle(2, p)
    r = circle(3, p)
    ReDim Preserve line(4, pocetLine + n)
    Do While t <= n
        line(1, pocetLine + t) = s1 + r * Math.Cos(2 / n * t * Math.PI)
        line(2, pocetLine + t) = s2 + r * Math.Sin(2 / n * t * Math.PI)
        If t <> 1 Then
            line(3, pocetLine + t - 1) = s1 + r * Math.Cos(2 / n * t * Math.PI)
            line(4, pocetLine + t - 1) = s2 + r * Math.Sin(2 / n * t * Math.PI)
        Else
            line(3, pocetLine + n) = s1 + r * Math.Cos(2 / n * t * Math.PI)
            line(4, pocetLine + n) = s2 + r * Math.Sin(2 / n * t * Math.PI)
        End If
        t += 1

    Loop
    pocetLine += n
    t = 1
    p += 1
Loop
p = 1

ReDim pline(4, pocetLine)
pocetPlatnychLine = pocetLine
Dim prevodovyPomer As Double
prevodovyPomer = 1 / 0.09
```

```
    Do While p <= pocetLine
      If line(1, p) > 210 Or line(2, p) > 297 Or line(3, p) > 210 Or line(4, p) > 297 Or line(1, p) < 0 Or line(2, p) < 0 Or line(3, p) < 0 Or line(4,
p) < 0 Then
        b = (line(4, p) * line(1, p) - line(2, p) * line(3, p)) / (line(1, p) - line(3, p))
        If line(1, p) <> 0 Then
          a = (line(2, p) - b) / line(1, p)
        Else
          a = (line(4, p) - b) / line(3, p)
        End If

        If line(1, p) > 210 Then
          line(1, p) = 210
          line(2, p) = a * 210 + b
        End If
        If line(2, p) > 297 Then
          line(2, p) = 297
          line(1, p) = (297 - b) / a
        End If
        If line(3, p) > 210 Then
          line(3, p) = 210
          line(4, p) = a * 210 + b
        End If
        If line(4, p) > 297 Then
          line(4, p) = 297
          line(3, p) = (297 - b) / a
        End If

        If line(1, p) < 0 Then
```

```
            line(1, p) = 0
            line(2, p) = a * 0 + b
        End If
        If line(2, p) < 0 Then
            line(2, p) = 0
            line(1, p) = (0 - b) / a
        End If
        If line(3, p) < 0 Then
            line(3, p) = 0
            line(4, p) = a * 0 + b
        End If
        If line(4, p) < 0 Then
            line(4, p) = 0
            line(3, p) = (0 - b) / a
        End If
    End If
    If line(1, p) - line(3, p) + line(2, p) - line(4, p) <> 0 Then
        pline(1, p - pocetLine + pocetPlatnychLine) = line(1, p) * prevodovyPomer
        pline(2, p - pocetLine + pocetPlatnychLine) = line(2, p) * prevodovyPomer
        pline(3, p - pocetLine + pocetPlatnychLine) = line(3, p) * prevodovyPomer
        pline(4, p - pocetLine + pocetPlatnychLine) = line(4, p) * prevodovyPomer

    Else
        pocetPlatnychLine -= 1
    End If
    p += 1
Loop

ReDim Preserve pline(4, pocetPlatnychLine)
```

```
End Sub


Private Sub Panel1_Paint(sender As System.Object, e As System.Windows.Forms.PaintEventArgs) Handles Panel1.Paint
    e.Graphics.Clear(Color.White)
    e.Graphics.SmoothingMode = Drawing2D.SmoothingMode.AntiAlias
    Dim pero As New Pen(Color.DarkSeaGreen)
    Dim p As Integer = 1

    Dim prevodovyPomer As Double
    prevodovyPomer = 1 / 0.09

    Do While p <= pocetPlatnychLine
        pero = Pens.Red
        If p < cisloLine Then
            pero = Pens.Black
        End If
        If p > cisloLine Then
            pero = Pens.DarkSeaGreen
        End If
        e.Graphics.DrawLine(pero, CInt(pline(1, p) / prevodovyPomer) * 2, CInt(297 - pline(2, p) / prevodovyPomer) * 2, CInt(pline(3, p) /
prevodovyPomer) * 2, CInt(297 - pline(4, p) / prevodovyPomer) * 2)
        p += 1
    Loop

    p = 1
End Sub

Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
```

```
End Sub

Private Sub Button4_Click(sender As System.Object, e As System.EventArgs) Handles Button4.Click
    Try
        SP1.PortName = ComboBox1.SelectedItem.ToString
        SP1.BaudRate = 9600
        SP1.DataBits = 8
        SP1.Parity = Ports.Parity.None
        SP1.StopBits = Ports.StopBits.One
        SP1.Handshake = Ports.Handshake.None

        SP1.Open()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
        Exit Sub
    End Try

    Button4.Enabled = False
    Label3.Visible = True
    ks = 239 Xor cisloPrikazu Xor 1
    ReDim zprava(3)
    ciloKalibrace = cisloPrikazu
    zprava = {239, cisloPrikazu, 1, ks}
    SP1.Write(zprava, 0, zprava.Length)
    pocitadloCasovace = 0
    navazaniSpojeni.Start()
    Timer1.Start()
End Sub
```

```
Private Sub SP1_DataReceived(sender As System.Object, e As System.IO.Ports.SerialDataReceivedEventArgs) Handles SP1.DataReceived
   Try
      If Me.InvokeRequired() Then
         Dim dr As New DataRecievedDelegate(AddressOf DataRecieved)
         Me.Invoke(dr, e, SP1.ReadByte)
      Else
         DataRecieved(e, SP1.ReadByte)
      End If
   Catch
   End Try
End Sub

Delegate Sub DataRecievedDelegate(ByVal e As System.IO.Ports.SerialDataReceivedEventArgs, ByVal linka As Integer)

Function prijemDat() As Integer
   velikostBufferu = buffer.Count()

   Try
      Select Case buffer.First()

         Case 237 'navázání spojení
            navazaniSpojeni.Stop()
            buffer.RemoveAt(0)
            odesilaniDat(True)
         Case 239 'příjem potvrzení
            If velikostBufferu >= 4 Then
               If buffer(0) Xor buffer(1) Xor buffer(2) Xor buffer(3) = 0 Then
                  cisloPrijatyhoprikazu = buffer(1)
```

```
                If buffer(2) = 10 Then
                    pocitadloCasovace = 0
                    navazaniSpojeni.Start()
                    Refresh()
                End If
                If buffer(2) = 11 Then
                    Select Case idPrikazu
                        Case 2
                            x = pline(1, cisloLine + 1)
                            y = pline(2, cisloLine + 1)
                        Case 3
                            x = pline(3, cisloLine)
                            y = pline(4, cisloLine)
                    End Select
                    odesilaniDat(False)
                End If
                buffer.RemoveRange(0, 4)
                Exit Function
            Else
                buffer.RemoveRange(0, 4)

            End If
        End If
    Case Else

        buffer.RemoveAt(0)

    End Select
Catch
```

```
    End Try

    Return buffer.Count()
End Function

Sub DataRecieved(ByVal e As System.IO.Ports.SerialDataReceivedEventArgs, ByVal linka As Integer)
    buffer.Add(linka)
    If linka <> 50 Then
        prijemDat()
    End If
End Sub

Private Sub navazaniSpojeni_Tick(sender As System.Object, e As System.EventArgs) Handles navazaniSpojeni.Tick
    If pocitadloCasovace >= 27 Then
        pocitadloCasovace = 0
        Dim navazoniSpojeni() As Byte = {237}
        SP1.Write(navazoniSpojeni, 0, navazoniSpojeni.Length)
    End If
    pocitadloCasovace += 1
End Sub

Public Sub odesilaniDat(ByRef chyba As Boolean)


    If chyba Then
        SP1.Write(zprava, 0, zprava.Length)
        pocitadloCasovace = 0
        navazaniSpojeni.Start()
        Exit Sub
```

```
End If

If cisloLine = pocetPlatnychLine Then
    cisloPrikazu += 1
    cisloPrikazu = (cisloPrikazu Mod 201) - 1
    ks = 239 Xor cisloPrikazu Xor 1
    ReDim zprava(3)
    ciloKalibrace = cisloPrikazu
    zprava = {239, cisloPrikazu, 1, ks}
    SP1.Write(zprava, 0, zprava.Length)
    SP1.Close()
    buffer.Clear()
    Timer1.Stop()
    navazaniSpojeni.Stop()
    Application.Exit()
End If

cisloLine += 1
Dim xL, xP, yL, yP As Integer
If (pline(1, cisloLine) <> x) Or (pline(2, cisloLine) <> y) Then
    xL = (pline(1, cisloLine) And &HFF00) / 256
    xP = pline(1, cisloLine) And &HFF
    yL = (pline(2, cisloLine) And &HFF00) / 256
    yP = pline(2, cisloLine) And &HFF
    cisloPrikazu += 1
    cisloPrikazu = (cisloPrikazu Mod 201) - 1
    idPrikazu = 2
    ks = 239 Xor cisloPrikazu Xor idPrikazu Xor xL Xor xP Xor yL Xor yP
    zprava = {239, cisloPrikazu, idPrikazu, xL, xP, yL, yP, ks}
```

```
            SP1.Write(zprava, 0, zprava.Length)
            navazaniSpojeni.Enabled = True
            cisloLine -= 1
        Else
            xL = (pline(3, cisloLine) And &HFF00) / 256
            xP = pline(3, cisloLine) And &HFF
            yL = (pline(4, cisloLine) And &HFF00) / 256
            yP = pline(4, cisloLine) And &HFF
            cisloPrikazu += 1
            cisloPrikazu = (cisloPrikazu Mod 201) - 1
            idPrikazu = 3
            ks = 239 Xor cisloPrikazu Xor idPrikazu Xor xL Xor xP Xor yL Xor yP
            zprava = {239, cisloPrikazu, idPrikazu, xL, xP, yL, yP, ks}
            SP1.Write(zprava, 0, zprava.Length)
            navazaniSpojeni.Enabled = True
        End If
        Label3.Text = CInt(cisloLine / (pocetPlatnychLine) * 10000) / 100 & "% " & navazaniSpojeni.Enabled
        Panel1.Refresh()
    End Sub

    Private Sub Timer1_Tick(sender As System.Object, e As System.EventArgs) Handles Timer1.Tick
        prijemDat()
    End Sub
End Class
```

Příloha 9: Zdrojový kód plotteru

```
#define F_CPU 8000000UL
#include <avr/io.h>
#include <inttypes.h>
#include <avr/interrupt.h>
#include <util/delay.h>

unsigned char DOLE = 17;
unsigned char NAHORE = 21;
unsigned char start;
unsigned char cisloPrikazu;
unsigned char id;
unsigned char ks;
unsigned char p = 20;
unsigned char dioda = 1;
unsigned char citacTimeru1;
unsigned char x1;
unsigned char x2;
unsigned char y11;
unsigned char y2;
unsigned char XJeVetsiNezY;
signed char poziceMotoru1;
signed char poziceMotoru2;
signed char zx;
signed char zy;
unsigned short x;
unsigned short y;
unsigned short xp;
```

```
unsigned short yp;
unsigned short citacTimeru2 = 1;
unsigned short ax = 0;
unsigned short ay = 0;
unsigned short pozicePocatkuX = 300;
unsigned short pozicePocatkuY = 150;
signed short dx;
signed short dy;
//volatile unsigned short ax = 0;
//volatile unsigned short ay = 0;
unsigned char pole[] = {0b00001000, 0b00001001, 0b00000001, 0b00000101, 0b00000100, 0b00000110, 0b00000010, 0b00001010};
// send char
void uart_putc(unsigned char data )
{
        while ( !( UCSR0A & (1<<UDRE0)) )
        ;
        UDR0 = data;
}
unsigned char uart_getc( void )
{
        while ( !(UCSR0A & (1<<RXC0)) )
        ;
        return UDR0;
}
ISR (TIMER0_OVF_vect){
         if (!(--dioda))
         {
                PORTD ^= 0x80;
                uart_putc(50);
```

```
                dioda = 20;
            }

}
ISR (TIMER1_COMPA_vect){

        ++citacTimeru1;
        citacTimeru1 %= 200;
        if (citacTimeru1 < p) {
                PORTC |= (1 << 5);
        }
        else
        {
                PORTC &= (~(1 << 5));
        }

}
void motorX(signed char z){
        poziceMotoru1 -= z;
        poziceMotoru1 = (poziceMotoru1 + 8) % 8;
        PORTC &= 0b11110000;
        PORTC |= pole[poziceMotoru1];
}
void motorY(signed char z){
        poziceMotoru2 += z;
        poziceMotoru2 = (poziceMotoru2 + 8) % 8;
        PORTB = pole[poziceMotoru2];
}
ISR (TIMER2_OVF_vect)
```

```
{
        //1/(f*1000000/(2^8*1024[preddelicka]))*1000*citac [ms]
        if (!(--citacTimeru2))
  {
          if (XJeVetsiNezY)
          {
                  motorX(zx);
                  ++ax;
                  x += zx;
                  if (ax == dx)
                  {
                          TIMSK2 = 0; //zakázání přerušení od přetečení timeru 2
                  }
          }
          else
          {
                  motorY(zy);
                  ++ay;
                  y += zy;
                  if (ay == dy)
                  {
                          TIMSK2 = 0; //zakázání přerušení od přetečení timeru 2
                  }
          }

          citacTimeru2 = 6;
  }
}
signed short ABS(signed short x){
```

```
        return (x<0)?-x:x;
}
void pocatekSouradnic(void) {


        if (p != NAHORE)
        {
                p = NAHORE;
                _delay_ms(100);
        }
        while ( (!(PINB & (1<<PINB4))) || (!(PINB & (1<<PINB5))) ) {

                if (!(PINB & (1<<PINB5)))
                {
                        motorX(-1);
                }
                if (!(PINB & (1<<PINB4)))
                {
                        motorY(-1);
                }
                _delay_ms(10);
        }

        x = 0;
        y = 0;

        while (((x < pozicePocatkuX) || (y < pozicePocatkuY))) {

                if (x < pozicePocatkuX)
```

```
                {
                        motorX(1);
                        x += 1;
                }
                if (y < pozicePocatkuY)
                {
                        motorY(1);
                        y += 1;
                }
                _delay_ms(12);
        }
        x = 0;
        y = 0;

}
void prejezd (void) {

        if (p != NAHORE)
        {
                p = NAHORE;
                _delay_ms(100);
        }

        xp = x2 | (x1 << 8);
        yp = y2 | (y11 << 8);

        if (y < yp)
        {
                zy = 1;
```

```
        }
        else
        {
                zy = -1;
        }

        if (x < xp)
        {
                zx = 1;
        }
        else
        {
                zx = -1;
        }

        while (((x != xp) || (y != yp))) {
                if (x != xp)
                {
                        motorX(zx);
                        x += zx;
                }
                if (y != yp)
                {
                        motorY(zy);
                        y += zy;
                }
                _delay_ms(12);
        }
}
```

```
void tisk(void) {
	float konstanta, pomocna; //volatile
	ay = 0;
	ax = 0;

	if (p != DOLE)
	{
		p = DOLE;
		_delay_ms(150);
	}
	p = DOLE - 2;
	_delay_ms(10);
	p = DOLE;

	xp = x2 | (x1 << 8);
	yp = y2 | (y11 << 8);
	// zjištení směru posunu//
	if (y < yp)
	{
		zy = 1;
	}
	else
	{
		zy = -1;
	}
	if (x < xp)
	{
		zx = 1;
	}
```

```
else
{
        zx = -1;
}
// výpočet vzádlenosti//
dx = ABS(xp - x);
dy = ABS(yp - y);
if (dx > dy)
{
        XJeVetsiNezY = 1;
        konstanta = (float)dy/dx;
        cli();
        TIMSK2 = (1<<TOIE2); //povolení přerušení od timeru 2
        while ((ax != dx) || (ay != dy))
        {

                pomocna         = (float)ay / ax;
                sei();
                if (konstanta > pomocna) // dopočítávání jestli se blíží úsečce
                {
                        motorY(zy);
                        ++ay;
                        y += zy;
                }

                _delay_ms(12);
                cli();
        }
}
```

```
        else
        {
                XJeVetsiNezY = 0;
                konstanta = (float)dx/dy;
                cli();
                TIMSK2 = (1<<TOIE2); //povolení přerušení od timeru 2
                while ((ax != dx) || (ay != dy))
                {
                        pomocna = (float)ax / ay;
                        sei();
                        if (konstanta > pomocna)
                        {
                                motorX(zx);
                                ++ax;
                                x += zx;
                        }
                        _delay_ms(12);
                        cli();
                }
        }
        sei();
}



int main(void)
{

        DDRD = 0x80;
```

```
PORTD = 0x00;
DDRB = (1<<PB0)|(1<<PB1)|(1<<PB2)|(1<<PB3);
DDRC = (1<<PC0)|(1<<PC1)|(1<<PC2)|(1<<PC3)|(1<<PC5);


/***** inicializace uart  *****/
UCSR0A = 0x00;
UBRR0L = 51;   // 9600 baud
UCSR0B |= (1<<TXEN0) | (1<<RXEN0);
UCSR0C |= (1<<UCSZ00) | (1<<UCSZ01);  // ramec dat:  8 datovych, 1 stop bit, bez parity


TIMSK0 = (1<<TOIE0); //povolení přerušení od přetečení timeru 0
TCCR0B |= (1<<CS00) | (1<<CS02); //předělička timeru 0 na 1024

TIMSK1 = (1<<OCIE1A); //povolení přerušení od timeru 1 (CTC)
TCCR1B |= (1<<CS11) | (1<<WGM12); //předělicka na 8 a nastavebí CTC módu
OCR1AL = 99; // 460 při 18,xx MHz, 199 při 8 MHz (1. 8 bitů)
OCR1AH = 0; //(zbytek bitů)

TCCR2B = (1<<CS22); //předdělicka timeru 2 na 64

PORTD = 0x80;

_delay_ms(1000);

sei();

while(1)
```

```
{

        start = uart_getc();
        switch (start) {
        case 237:
                uart_putc(237);
                break;
        case 239:
                cisloPrikazu = uart_getc();
                id = uart_getc();
                switch (id) {
                case 1:
                        ks = uart_getc();
                        if (239 ^ cisloPrikazu ^ id ^ ks)
                        {
                                break;
                        }
                        ks = 239 ^ cisloPrikazu ^ 10;
                        uart_putc(239);
                        uart_putc(cisloPrikazu);
                        uart_putc(10);
                        uart_putc(ks);

                        pocatekSouradnic();

                        ks = 239 ^ cisloPrikazu ^ 11;
                        uart_putc(239);
                        uart_putc(cisloPrikazu);
                        uart_putc(11);
```

```
            uart_putc(ks);

            break;
case 2:
            x1 = uart_getc();
            x2 = uart_getc();
            y11 = uart_getc();
            y2 = uart_getc();
            ks = uart_getc();
            if (239 ^ cisloPrikazu ^ id ^ x1 ^ x2 ^ y11 ^ y2 ^ ks)
            {
                  break;
            }
            ks = 239 ^ cisloPrikazu ^ 10;
            uart_putc(239);
            uart_putc(cisloPrikazu);
            uart_putc(10);
            uart_putc(ks);

            prejezd();

            ks = 239 ^ cisloPrikazu ^ 11;
            uart_putc(239);
            uart_putc(cisloPrikazu);
            uart_putc(11);
            uart_putc(ks);

            break;
case 3:
```

```
                x1 = uart_getc();
                x2 = uart_getc();
                y11 = uart_getc();
                y2 = uart_getc();
                ks = uart_getc();
                if (239 ^ cisloPrikazu ^ id ^ x1 ^ x2 ^ y11 ^ y2 ^ ks)
                {

                        break;
                }
                ks = 239 ^ cisloPrikazu ^ 10;
                uart_putc(239);
                uart_putc(cisloPrikazu);
                uart_putc(10);
                uart_putc(ks);

                tisk();

                ks = 239 ^ cisloPrikazu ^ 11;
                uart_putc(239);
                uart_putc(cisloPrikazu);
                uart_putc(11);
                uart_putc(ks);
                break;
            }
            break;
        }

}
```

```
        return 0;
}
```