VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

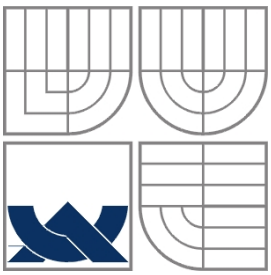
EDITOR A MANAŽER PROGRAMŮ PRO SVAŘOVACÍ AUTOMATY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

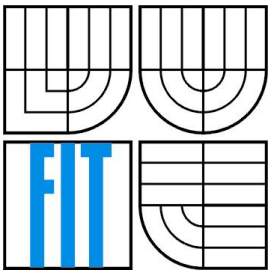
AUTOR PRÁCE
AUTHOR

LENKA ŠEVČÍKOVÁ

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

EDITOR A MANAŽER PROGRAMŮ PRO SVAŘOVACÍ AUTOMATY

EDITOR AND MANAGER OF PROGRAMS FOR WELDING MACHINES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

LENKA ŠEVČÍKOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

doc. Dr. Ing. JAN ČERNOCKÝ

BRNO 2011

Abstrakt

Úkolem této práce je modernizovat a dotvořit grafické uživatelské prostředí pro barevný dotykový panel a naprogramovat funkce usnadňující uživatelům práci. Software je vyvíjen pro plazmové navařovací automaty vyráběné firmou KSK. Modernizace se týká editoru a manažeru svařovacích programů a nových způsobů vytváření svařovacích programů.

Abstract

The task of this work is to update and complete the graphical user interface for color touch-panel and to add functions simplifying the work of its users. The software is being developed for plasma surfacing machines produced by KSK company. The updates and modernization included the editor and manager of welding programs and new ways of generating welding programs.

Klíčová slova

Dotykový panel, B+R, KSK, Automation Studio, Visual Components VC4, PP 400, Power Panel, vizualizace, svařování, svařovací automat, plazmové navařování, grafické rozhraní, editor programů, manažer programů.

Keywords

Touch panel, B+R, KSK, Automation Studio, Visual Components VC4, PP 400, Power Panel, visualization, welding, welding machine, plasma surfacing, graphical interface, editor of programs, manager of programs.

Citace

Lenka Ševčíková: Editor a manažer programů pro svařovací automaty, bakalářská práce, Brno, FIT VUT v Brně, 2011

Editor a manažer programů pro svařovací automaty

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana doc. Dr. Ing. Jana Černockého a konzultanta z firmy KSK pana Stanislava Ševčíka. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Lenka Ševčíková
11. května 2011

Poděkování

Chtěla bych velmi poděkovat vedoucímu práce panu doc. Dr. Ing. Janu Černockému za pomoc a cenné rady, které tuto práci zkvalitnily. Dále děkuji za technickou pomoc a informace mému konzultantovi a otci v jedné osobě, panu Stanislavu Ševčíkovi, mým kolegům v práci za testování systému v provozu a v neposlední řadě mé rodině a příteli za psychickou podporu.

Obsah

1 Úvod

Spolupracuji s českotřebovskou firmou KSK, s.r.o. vyrábějící svařovací automaty, ve kterých používá dotykové obrazovky rakouské firmy B&R automatizace s pobočkou v Brně. Vzhledem k tomu, že ve firmě KSK je pouze jeden zaneprázdněný programátor těchto panelů, na úpravy a doprogramování uživatelského prostředí pro obsluhu stroje už nemá čas. Rozhraní je tudíž již dlouho zastaralé, není úplně ideální a navíc některé funkce úplně chybí. Rozhodla jsem se tedy pro modernizaci a dopracování dvou uživatelských částí, editoru a manažeru programů svařovacích automatů, aby se obsluze stroje ulehčila práce s tímto systémem i se samotným strojem.

Pro představu si manažer programů pro svařovací automaty můžeme představit jako zjednodušený manažer typu Commander, kde můžeme kopírovat, přejmenovávat a mazat soubory. Editor svařovacích programů můžeme graficky přirovnat k velmi zjednodušené verzi programu typu Excel. Je to obrazovka s řádky a sloupci s několika funkcemi. Více se o těchto částech dozvíte v kapitole 3 a 4.

Firma KSK poměrně dlouho používá osvědčený software i hardware firmy B+R a proto je jako programovací nástroj použito nejnovější vývojové prostředí B&R Automation Studio s Visual Components VC4 a barevný dotykový panel řady PP 400 od firmy B+R. Vše si popíšeme v druhé kapitole.

Po seznámení s vývojovým prostředím a panelem začneme se samotnou analýzou celé problematiky (viz kapitola 3). Nejdříve prozkoumáme původní stav rozhraní a využijeme k tomu i zaměstnance firmy KSK pracující se svařovacími automaty, kde je nainstalována starší verze software. Po konzultaci se zadavatelem práce využijeme některé nápady a připomínky zaměstnanců v návrhu nového software a uživatelského rozhraní (viz kapitola 4). Podle možností pak budou připomínky realizovány v implementaci. V páté kapitole si ukážeme nové možnosti generování svařovacích programů.

Další etapa je rozsáhlé testování nového rozhraní a software, porovnání s původním řešením a případné opravy chyb, na které se v průběhu testování přišlo. Testy probíhají prakticky neustále. Již při implementaci se zkouší jednotlivé části software a po důkladném odzkoušení nanečisto může být software uvolněn do provozu na řádné otestování. O testování se dozvíte v kapitole 6.

V poslední, sedmé kapitole, se dočteme o závěrečném zhodnocení celé práce.

2 Seznámení

V této kapitole se seznámíme s již zmíněnými firmami KSK a B&R. Popíšeme si vývojové prostředí využívané při tvorbě programů, použitou dotykovou obrazovku a navařovací automat, na kterém je vytvořený software aplikován.

2.1 KSK

Firma KSK, s.r.o. je malá soukromá českotřebovská firma. Jejich práci můžete vidět na všech kontinentech světa. Své postavení a oblibu v rámci oboru získala kvalitními českými produkty a plazmovým navařováním.

Firma již přes patnáct let nabízí široký sortiment svařovacích a navařovacích materiálů, ale hlavní náplní je výroba svařovacích a navařovacích automatů, polohovadel a jiných specializovaných zařízení. Pro firmy, kterým se automat nevyplatí kupovat a přesto potřebují některé díly navařit, firma KSK nabízí zakázkové svařování a navařování na svých automatech.

Ve většině typů strojů jsou zabudovány obrazovky firmy B&R. Ne všechny automaty však vyžadují přímo dotykové ovládání, proto jsou na nich využity i jiné typy obrazovek. Přestože jsou také velmi zajímavé, my se jimi zajímat nebudeme a zaměříme se pouze na dotykové obrazovky.

2.1.1 Plazmové navařovací automaty řady PPC 250

Starší verze manažeru a editoru programů již běží na plazmových navařovacích automatech řady PPC 250. Řada PPC 250 je základní, nejčastěji používané provedení automatu firmy KSK. Automaty lze snadno upravit podle potřeb zákazníka, protože jsou řešeny stavebnicově. Je tedy možné automaty snadno přizpůsobit pro požadovanou aplikaci.

Součástí automatů jsou polohovadla s různými technickými parametry, též vyráběna firmou KSK. Na polohovadla se připevňují navařované součásti, jejichž hmotnost může být dle typu polohovadla od 20 kg do 10 tun.

Jednou z oblastí využití plazmového navařování jsou aplikace přídavných materiálů (prášek, drát nebo trubička) na těsnicí plochy průmyslových armatur používaných v jaderné i klasické energetice, petrochemii a podobně, jako jsou desky klínů, sedla šoupátek, kuželky regulačních a uzavíracích ventilů. Dále se používá na ventily a vahadla spalovacích motorů, litinové i bronzové formy na sklo, formy na keramiku, šneky a komory dopravníků, zuby rypadel a bagrů, válce dopravníků v hutích, součásti čerpadel a turbín atd.

Důvod, proč se součásti navařují, je zvýšení odolnosti součástí. Po navaření nedochází k mechanickému opotřebení ani korozi. Proto se navařování uplatňuje i při renovacích součástí.

Technologií navařování je několik. Můžeme navařovat elektrickým obloukem, laserem, plamenem nebo právě plazmou. Mezi hlavní výhody plazmového navařování řadíme nízkou spotřebu navařovacího materiálu, malou deformaci navařované součásti vlivem nízkého tepelného ovlivnění a rovnoměrné a přesné navařované vrstvy.

Obrázek 2.1: Plazmový navařovací automat PPC 250 GMR

Obrázek 2.2: Polohovadlo PO 1000 S

2.2 B&R automatizace

Mezinárodní společnost B&R má více jak třicetiletou zkušenost s průmyslovou automatizací. Nabízí kompletní programovací nástroj - Automation Studio s integrovanou vizualizací a ovládáním přes dotykovou obrazovku, servopohony ACOPOS, výkonná průmyslová PC a mnohé další prostředky pro úplné automatizační řešení.

Firma pořádá pravidelná školení i na české pobočce v Brně, kde se dozvíme o všech schopnostech Automation Studia a mnohém dalším. Od instalace software, založení nového projektu, programování v jazyce C nebo Automation Basicu, přes základy komunikace systému, popisu a použití Power Panelů, grafických komponent, hardwarových konfiguracích, až po vytváření vlastních knihoven.

2.3 Automation Studio

Automation Studio s integrovanými Visual Components VC 4 (obr. 2.3) je využito pro tvorbu grafického prostředí a programování nejen tohoto prostředí. Program zahrnuje všechny potřebné nástroje pro každou část vývoje projektu.

Celý projekt je rozdělen do funkčních balíčků, což je u velkých projektů důležité pro snadnější správu a programování aplikace. Datové typy a proměnné jsou také zapouzdřeny do balíčků. Studio umožňuje přidávání hardwarových a softwarových komponent a vytváření hardwarové konfigurace.

Při programování aplikací v Automation Studiu si můžeme vybrat mezi několika programovacími jazyky. Mezi nejpoužívanější patří jazyk Automation Basic, vyvíjený přímo firmou B&R, nebo jazyk C, ve kterém je programován náš software.

Studio běží bezproblémově na systémech Windows 7, Windows XP, Windows 2000 a na jejich rozšířených verzích.

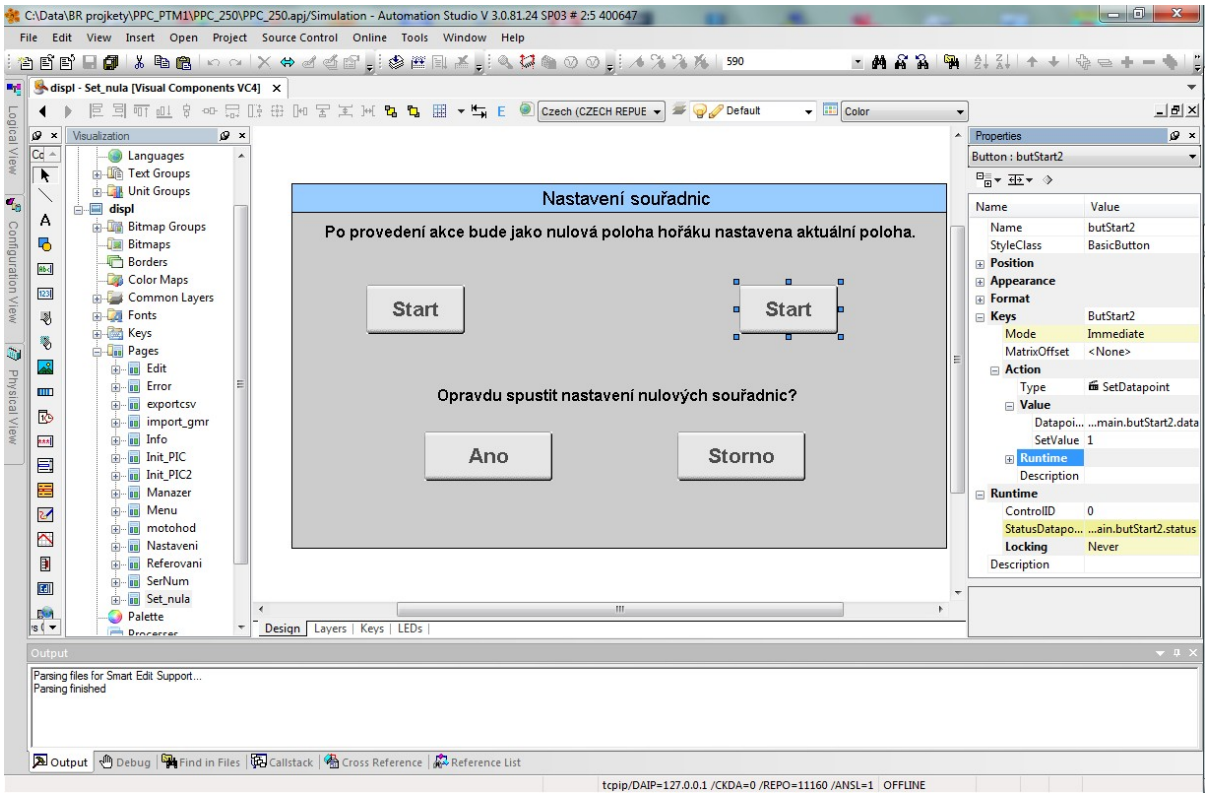
Velkým plusem Automation Studia je obsáhlá integrovaná nápověda v anglickém jazyce a popis dostupných knihoven a funkcí s příklady.

2.3.1 Visual Components

Visual Components obsahuje klasické grafické komponenty, jako jsou textová a číselná vstupní a výstupní pole, tlačítka, linky a jiné geometrické tvary, bitmapy podporující formáty BMP, PNG a GIF, sloupcové a kruhové grafy, prvky pro přehrávání audia a videa a mnoho dalších komponent. Použité grafické komponenty mají v projektu přehledné strukturované uspořádání.

Při použití komponent v projektu se automaticky přiřadí její název, který máme možnost změnit. Na komponentu můžeme navázat několik proměnných, na ty se pak odkazujeme ve zdrojových kódech a smíme je libovolně měnit. Mezi nejběžněji používané proměnné řadíme následující:

- barva komponenty – v případě, že potřebujeme měnit barvu komponenty za běhu programu
- status komponenty – normální, neviditelná nebo zablokovaná komponenta, také se využívá za běhu programu
- hodnota při stisku – například u tlačítek, aktivních bodů a podobně, abychom dokázali určit, zda došlo ke stisknutí komponenty
- provázanost komponent s daty – některé komponenty (např. vstupní, výstupní a textová pole, tlačítka, atd.) mohou zobrazovat data, pokud jsou provázány s proměnnými, ve kterých jsou uloženy hodnoty



Obrázek 2.3: Automation Studio – Visual Components

2.3.2 Automation Runtime

Nezbytným doplňkem studia je Automation Runtime, díky kterému můžeme naše vytvořené aplikace bezproblémově přenášet mezi ostatními B&R systémy a hardware. Toho využíváme například při změně použité obrazovky. Součástí doplňku je rozsáhlá knihovna vyhovující IEC 61131-3 a rozšířená knihovna B&R Automation library. Knihovna B&R je velmi důležitá například pro práci se soubory, protože mezi klasickým programováním v C a programováním v Automation Studiu je nepatrný rozdíl.

2.3.3 Automation NET / PVI

Komunikaci mezi právě právě programovanou aplikací a panelem umožňuje Automation NET. Komunikace probíhá přes sériové rozhraní RS232, sběrnici CAN, nebo přes Ethernet. Umožňuje i bezproblémový přenos již existujících aplikací na nové systémy. Komunikaci dále využíváme například při nahrávání software do panelu, nebo pro ovládání grafických komponent na dálku.

2.4 Power Panel

Obrazovky (obr. 2.4, 2.5) použité ve svařovacích automatech jsou řady Power Panel 400 od firmy B&R. Jedná se o jednobarevné nebo barevné obrazovky, od velikosti 5,7 palců do 15 palců, VGA, QVGA nebo XGA. Vybrat si můžeme z dotykové, klávesové nebo kombinované verze.

Vzhledem k tomu, že na zkušebním automatu ve firmě KSK je nyní použit barevný dotykový 15" Power Panel 420 (obr. 2.5), budeme přizpůsobovat vzhled grafického prostředí pro tento konkrétní typ, který lze ovšem v případě potřeby velmi jednoduše změnit díky vymoženostem Automation Studia a jeho doplňků. Testování prostředí bude také na tomto typu Power Panelu.

2.4.1 Vlastnosti Power Panelu 420

- 15" XGA barevný (512 barev) TFT panel
- rozlišení 1024 x 768 pixelů
- dotykové ovládání
- 128 MB SDRAM, 512 KB SRAM
- CompactFlash slot
- aPCI slot
- Ethernet 10/100 Mbps, RS232, 2x USB
- procesor Geode LX800 500 Mhz (AMD, kompatibilita s architekturou x86)
- provoz při teplotě 0 až 50° C
- průměrná váha 6,7 kg
- rozměry (Š x V x H): 435 x 330 x 86 mm

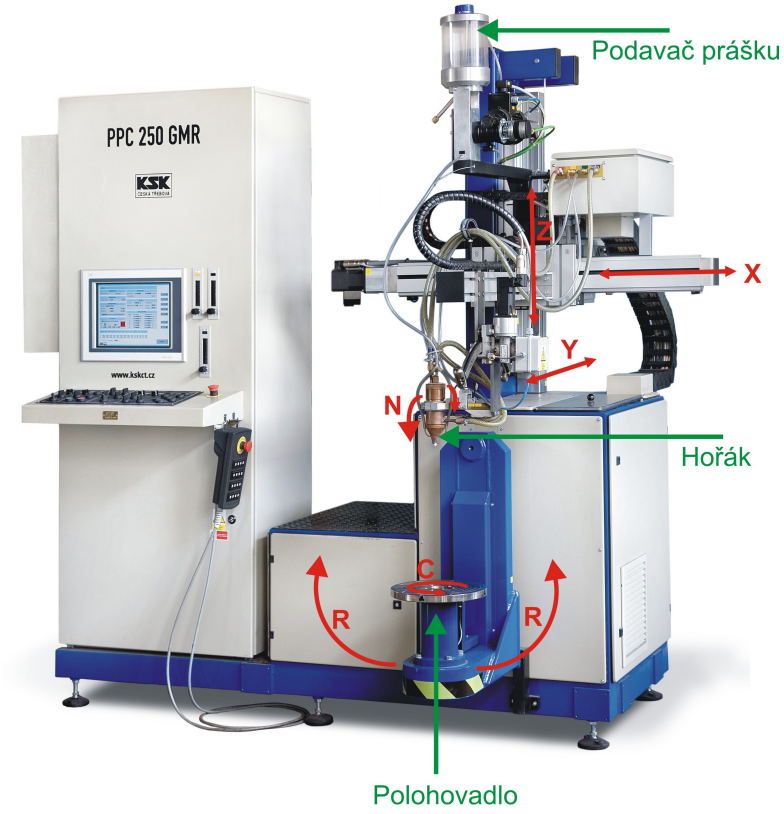
Obrázek 2.4: 15" XGA Power Panel 481

Obrázek 2.5: 15" XGA Power Panel 420

2.5 Základní principy navařovacích automatů

Tato sekce uvádí několik principů, bez kterých se v následujících kapitolách neobejdeme. Nebudeme si vysvětlovat, jak pracuje celý automat, tedy hardwarovou stránku věci, ale shrneme poznatky nutné pro tvorbu software.

Automaty jsou dle typu vybaveny rameny, která se pohybují maximálně ve třech směrech – X, Y a Z. Na rameni osy X je umístěn hořák, kterým se navařují díly. Intenzita navařování je dána několika parametry, které si zanedlouho popíšeme. Hořák může být naklopený z pohledu obsluhy dopředu nebo dozadu. Navařované díly jsou připevněny na stole polohovadla, který je součástí stroje. Polohovadlo může být také naklopené a můžeme rotovat stolem polohovadla. Na obrázku 2.6 jsou všechny osy pro lepší představu znázorněny.



Obrázek 2.6: Znázornění os

Abychom automat dostali do pohybu, musíme mu předávat data. V dotykové obrazovce je paměť, kde uchováváme mimo jiné svařovací programy s příslušnými daty.

Data svařovacího programu jsou uložená ve struktuře, která čítá následujících 19 prvků různých datových typů.

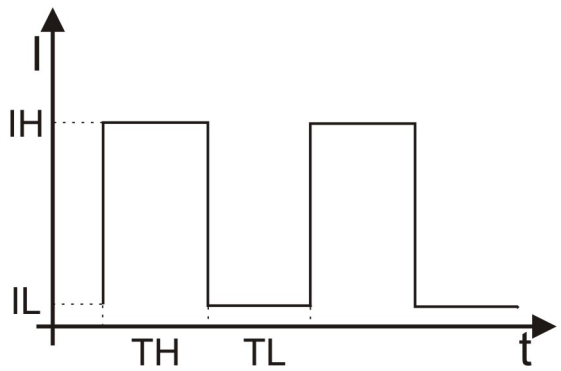
- Osa X – souřadnice osy X, z pozice obsluhy se jedná o pohyb doleva – doprava.
- Osa Y – souřadnice osy Y, z pozice obsluhy jde o pohyb ramene od sebe - k sobě.
- Osa Z – souřadnice osy Z, z pozice obsluhy je to pohyb ramene nahoru a dolů.
- Osa N – souřadnice osy N, což je naklopení hořáku.
- Osa R – souřadnice osy R, naklopení stolu polohovadla.
- Osa C – souřadnice osy C, poloha stolu polohovadla (ve stupních).

- Status – udává, jakým způsobem díl navařujeme. Existují tři statusy:
  - Status 1 – Lineární režim: polohovadlo stojí a v pohybu je pouze hořák. Tzn. osy X, Y a Z se pohybují, osy C a R stojí. (Tento režim se používá při navařování hran sklářských forem, střižných nástrojů, nerotačních ploch – obdélníky, atd.)
  - Status 2 - Přejezdy: nenavařuje se, používá se pouze pro pohyb hořáku nebo polohovadla na nějakou pozici.
  - Status 3 – Rotační režim: polohovadlo je v pohybu a hořák může pendlovat (kývat se ze strany na stranu), ale celý proces je řízen polohou polohovadla. (Režim pro navařování např. kroužků – stůl polohovadla se otáčí a hořák pendluje.)
- Lineární rychlost - rychlost pohybu hořáku po osách.
- Rotační rychlost - rychlost otáčení stolu polohovadla.
- Pendl – pohyb hořáku do stran. Další parametry pendlu jsou:
  - Šířka – vzdálenost mezi krajními body.
  - Rychlost – rychlost pendlování.
  - Pausa pozitivní – časová pomlka v době, kdy se pendl nachází vpravo.
  - Pausa negativní – časová pomlka v době, kdy se pendl nachází vlevo.
- Podávání - zde se myslí podávání prášku, konkrétně jaké množství prášku se spotřebuje za 1 minutu. V nejideálnějším případě se jedná o jednotku g/min. Podávání je seřízeno na určitý typ prášku, ale v případě, že se změní výrobce prášku, může se změnit i podané množství za minutu. S tímto faktem se musí počítat. Do budoucna bude tato funkce plně automatizovaná pomocí speciální váhy a množství prášku bude sypáno přesně.
- Proud – proud má několik složek:
  - IH – hodnota vysokého proudu v ampérech
  - TH – doba trvání vysokého proudu v sekundách
  - IL – hodnota nízkého proudu v ampérech
  - TL – doba trvání nízkého proudu v sekundách
  - IW – proud drátem – běžně navařujeme práškem, v některých případech využíváme drát, který je výrazně levnější

Parametry proudu říkají, jak velkým proudem budeme navařovat a jestli se jedná o tzv. pulzaci (obr. 2.7). Při pulzaci se sníží tepelný příkon a navařovaná součást se nepřehřívá tolik, jako při běžném navařování. S pulzací jsme proto schopni navařit i menší díly, jinak by návar stekl.

Pokud chceme navařovat s pulzací, musí být parametry IH, TH, IL a TL nenulové. Čas vysokého proudu nám udává, jak dlouho se bude navařovat součást vysokým proudem a čas nízkého proudu udává, jak dlouho budeme navařovat součást nízkým proudem. Tím vzniká pulzace.

V případě, že navařujeme větší součást a nepotřebujeme pulzaci, zadáme pouze vysoký proud IH a zbylé parametry ponecháme nulové.



Obrázek 2.7: Pulzace

3 Analýza problematiky

V této kapitole se zaměříme na problematiku celého systému. Podíváme se, jak vypadají zdrojové kódy, jakým způsobem jsou psány a proč. Rozebereme si strukturu grafického rozhraní manažeru, editoru programů a použitých balíků. Ukážeme původní řešení rozhraní, jeho výhody a nevýhody.

Manažer programů je používán zejména k vymazání již nepotřebných programů ve stroji a kopírování mezi CompactFlash diskem a pamětí v Power Panelu. Kopírování se provádí z důvodu bezpečnosti, abychom v případě havárie měli zálohy a nepřišli tak o programy pro svařování. Mazání je pouze za účelem vyčištění disku, aby nepoužívané nebo omylem vytvořené programy zbytečně nezabíraly místo.

Programy pro svařování vznikají buď v editoru programů nebo ručním režimem svařování. Pokud je svařovací automat zapnutý v ručním režimu, svářeč si nastavuje a mění patřičné parametry během svařování sám. Systém automaticky všechny parametry zaznamenává do programu, který se ukládá na disk panelu, odkud je ho možné opětovně spustit. Ruční režim může být používán pouze zkušenými svářeči, protože může velice snadno dojít ke kolizi. Proto se snažíme v možných případech programy generovat automaticky. Nemá však cenu dělat automatické generování na všechny navařované součásti. Některé netypické zakázky se objeví pouze jednou, nebo by bylo naprogramování automatického generování programu natolik složité, že je jednodušší, rychlejší a levnější navařit součástku ručně.

Editor programů slouží tedy pro automatické vygenerování programu pro svařování a jeho případné úpravy.

3.1 Zdrojové kódy a struktura programu

Software automatu má tři základní vrstvy. Logickou, konfigurační a fyzickou vrstvu. V logické části je zahrnut veškerý software, včetně právě nepoužívaného. V konfigurační vrstvě máme několik konfigurací, přičemž jednotlivé konfigurace se od sebe liší podle požadavků zákazníka na automat. V konfiguracích je nahraný software z logické vrstvy podle toho, jak jej v konfiguraci potřebujeme. Možnost takto vytvářet konfigurace je velkým ulehčením při tvorbě dalších programů. Poslední vrstvou je vrstva fyzická a zde je navolený hardware pro každou konfiguraci.

Každý balík z logické vrstvy obsahuje soubor se zdrojovým kódem a soubor s deklaracemi proměnných. Proměnné musí být odděleny od zdrojového kódu, což je vlastnost Automation Studia. Balíky mohou obsahovat mnohé další soubory, které potřebujeme.

Každý zdrojový kód lze rozdělit do několika částí. První inicializační část se spouští pouze při každém zapnutí systému nebo stroje.

Druhou části je cyklická část, která se opakuje podle toho, v jaké úloze (task) je balík zařazen. Základem cyklické části kódu je konstrukce switch. Tato konstrukce se ve studiu běžně používá a je jednou z možností, jak psát cyklické části v real-time systémech. Protože panel má jen jeden procesor, jednotlivé úlohy se spouští postupně v řádu desítek milisekund podle inicializačních tříd, ve kterých jsou zařazeny. Nejdříve se spouští úlohy v nejrychlejších třídách, po nich ty pomalejší. Každé úloze můžeme pro jistotu nastavit i časovou toleranci, aby nedošlo k problému, kdy časově náročný task nestihne proběhnout. V případě, že task přesto nestihne doběhnout včas, vyhlásí se chyba. V dobře naprogramovaných aplikacích však k této kolizi nedochází.

Při každém spuštění úlohy dochází k načítání vstupů z hardware, proběhne kód napsaný před switch a provede se jeden case, ostatní se přeskočí. Poté se provede kód za konstrukcí switch a výstupy se nastaví podle proměnných. Při dalším spuštění úlohy opět proběhne kód před cyklickým switch a přímo ve switch se pokračuje tam, kde skončil kód naposledy. Tímto způsobem probíhá spuštění kódu a jeho opakování.

V následující ukázce zdrojového kódu tgen_mezikruzi.c se pokusím demonstrovat, jak vypadá kód a jak se spouští.

```
// V těchto místech se mohou vyskytovat námi vytvořené funkce, které v programu voláme

void _INIT tgen_mezikruziINIT( void )
{
	/* V této inicializační části nastavíme proměnné dle potřeb. Například zde přednastavujeme
	proměnné, které jsou spojeny s komponentami a chceme tak uživateli nabídnout předvolené
	hodnoty. Tento kód se spouští pouze jednou a to buď při zapnutí stroje nebo resetování
	systému. */

	param.kNabehTransfer = 0.8;
	param.kNabehPodavani = 0.5;
	param.kNabehRychlost = 0.2;

	// atd...
}

void _CYCLIC tgen_mezikruziCYCLIC( void )
{
	// Nyní jsme v cyklické části, která se opakovaně spouští.

	switch(genMezikruzi.step)  //  V real-time systémech používáme konstrukci switch.
	{
		case 0:
			/* Při prvním spuštění proběhne pouze nultý case. Máme zde kontrolu
			na viditelnost obrazovky. Ve zdrojovém kódu tprogram.c se totiž zaznamená
			stisk tlačítka „Generování mezikruží“ v obrazovce menu. Jakmile je toto
			tlačítko stisknuto, status obrazovky „statusLayerGenMezikruzi“ se nastaví
			z neviditelného statusu „STATUS_INVISIBLE“ na viditelný status
			„STATUS_NORMAL“. V tu chvíli už reaguje kód tgen_mezikruzi.c,
			ve kterém máme následující podmínku a mohou tak probíhat i další case
			v tomto kódu při opětovném spuštění úlohy. */

			if(statusLayerGenMezikruzi==STATUS_NORMAL)
			{
				// Obvykle zde nastavujeme další proměnné
				minValue.x = K_X_MIN;
				maxValue.speedLin = K_RYCHLOST_LIN_MAX;
				minValue.proudTransfer.vHi     = K_TRANSFER_MIN;
				minValue.pendl.sirka    = 0;
				// atd ...

				// nastavíme další skok pro další spuštění úlohy
				genMezikruzi.step = 100;
			}
			break;


		case 100:
			/* Úloha se spustila znovu a jsme v case 100. Zde se v tomto případě děje
			kontrola hlavičky svařovacího programu. Každý svařovací program má
			uloženou i hlavičku, ze které se dozvíme o jaký typ programu se jedná a jaké
			parametry obsahuje. Probíhá tu kontrola, zda typ programu odpovídá
			mezikruží. Pokud ano, načteme do komponent pomocí proměnných příslušné
```

```
                uložené parametry. Pokud ne, nastavíme proměnné implicitně. V obou
                případech si načtené hodnoty může uživatel měnit dle potřeb. */

                // Nastavíme další skok.
                genMezikruzi.step = 1000;
        break;

        case 1000:
                /* Znovuspuštěním úlohy jsem doskočili do case 1000. V tomto kroku
                zůstaneme, dokud uživatel nezadá všechny parametry správně.

                // Nastavíme další skok.
                genMezikruzi.step = 1200;
        break;

        case 1200:
                /* Skokem do tohoto case vymažeme pole, do kterého se budou později
                zapisovat vygenerovaná data. */

                // Nastavíme další skok.
                genMezikruzi.step = 1300;
        break;

        // atd ...

        /* Více o generování programu mezikruží se dočtete v kapitole 5.3. */

    } // Konec switch
} // Konec cyklické části
```

3.2 Struktura grafického rozhraní

Základem rozhraní je stránka (page) skládající se z vrstev (layer). Počet vrstev ve stránce je neomezený. Vrstva může být jen jedna, nebo jich může být několik. Vrstvy obsahují grafické komponenty, které byly zmíněny v kapitole 2.3.1. Komponenty můžeme různě překrývat, zviditelňovat a uzamykat, takže je v případě nutnosti smíme na sebe vrstvit.

Na obrázku 3.1 je částečně vyobrazena struktura našeho rozhraní. Po inicializaci stroje naběhne obrazovka menu, z níž se dotykem dostaneme na několik dalších obrazovek. Nás ale zajímá pouze obrazovka edit a manažer. Konkrétně jejich vrstvy.

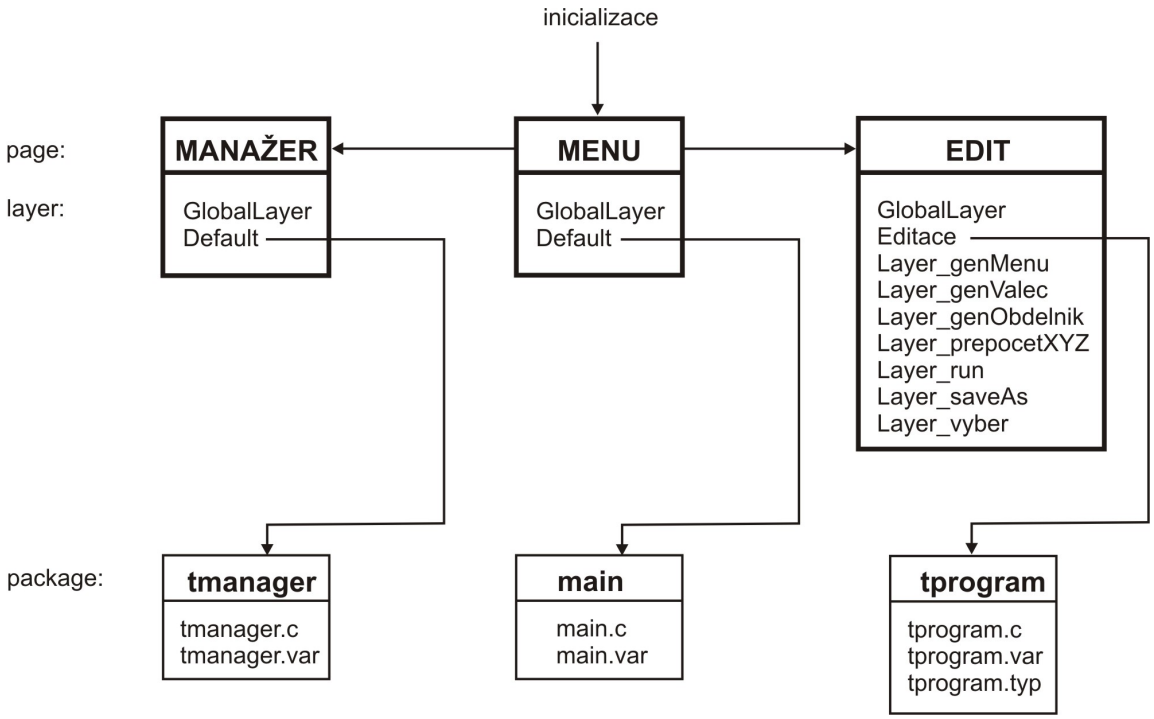
Vrstva s názvem *GlobalLayer* je vrstvou globální a v našem případě je pro všechny stránky stejná. Nicméně není povinností mít globální vrstvu nebo ji mít ve všech stránkách. Případně můžeme mít globálních vrstev i více. Tuto vrstvu používáme jako podklad pro ostatní stránky, protože chceme na každé stránce zobrazovat několik údajů. Mezi klasické údaje nebo funkce patří přepínání jazykových mutací, časové údaje, IP adresa, teplota procesoru a grafický prvek ohraničení obrazovky, jak můžete vidět v modrém rámečku na obrázku 3.3.

Součástí obrazovky *manažeru* (obr. 3.3) je globální vrstva a standardní (default) vrstva s komponentami. Prvky na standardní vrstvě využívají zejména balík (package) *tmanager*, který má v sobě zapouzdřen dva soubory – *tmanager.c* a *tmanager.var*. Jak již název napovídá, *tmanager.c* je

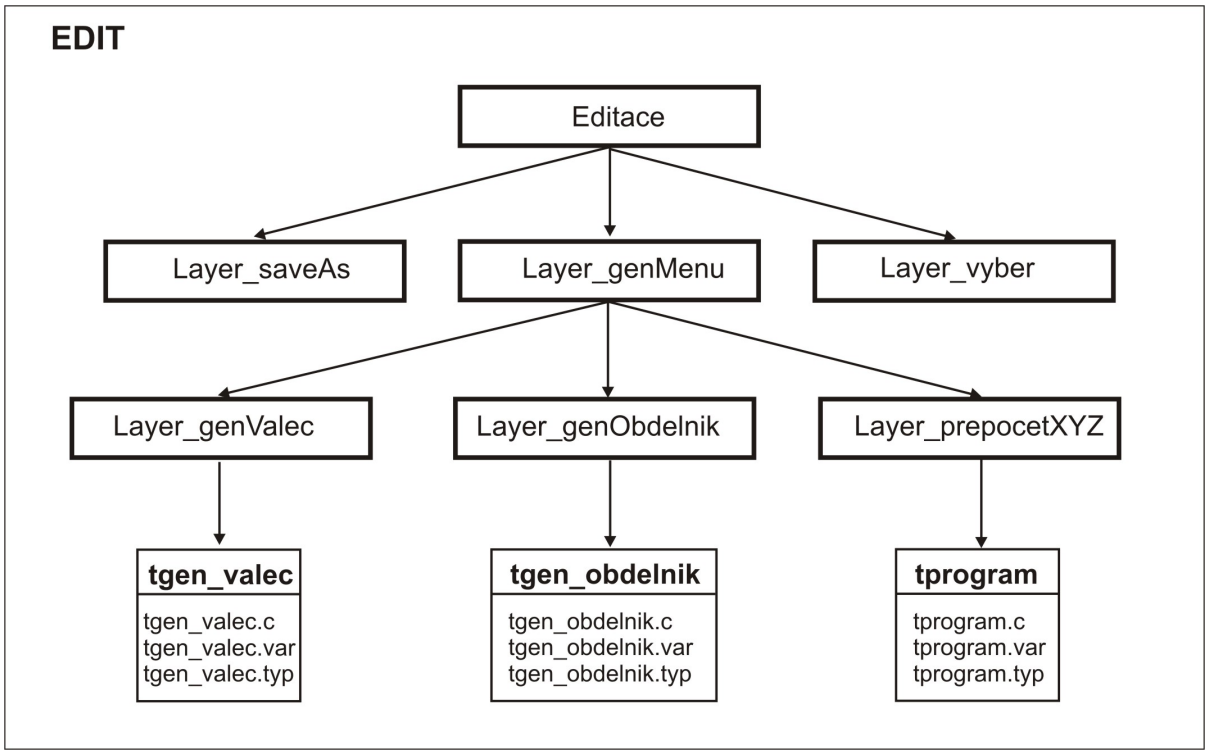
zdrojový kód a *tmanager.var* je soubor s deklaracemi proměnných. To je trochu odlišné od klasického C, kde deklarace proměnných je přímo u zdrojových kódů.

Z menu se lze dostat na stránku *edit* (obr. 3.4), konkrétně na vrstvu *Editace*. Hlavním používaným balíkem vrstvy je *tprogram*. Obdobně jako u manažeru i *tprogram* ukrývá soubor se zdrojovým kódem, soubor s deklaracemi proměnných a navíc i soubor s datovými typy.

Z vrstvy *Editace* jsou dále odkazy na několik dalších vrstev. Ne všechny je zapotřebí zmodernizovat, proto se jimi nebudeme zabývat. Na obrázku 3.2 je naznačena částečná struktura stránky *edit*. Vrstva *Layer_saveAs* se používá pro vytvoření nového programu, vrstva *Layer_vyber* pro výběr programu a vrstva *Layer_genMenu* nás odkazuje na menu generování. Na těchto vrstvách neprobíhaly žádné výrazné změny, proto zde ani neuvádím použité balíky, jsou však potřebné pro pochopení některých funkcí a vztahů v editoru.



Obrázek 3.1: Struktura grafického rozhraní



Obrázek 3.2: Struktura stránky Edit

3.3 Původní stav grafického rozhraní manažeru programů

Manažer programů slouží k vymazání nepotřebných programů ve stroji a kopírování mezi CompactFlash diskem a pamětí v Power Panelu. Na obrázku 3.3 je původní stav manažeru.

Na soubor, který chceme vymazat nebo zkopírovat, najedeme pomocí šipek, přičemž nelze posouvat obě strany manažeru zároveň, což je krajně neefektivní z hlediska rychlosti obsluhy. Navíc, máme-li hodně položek v adresáři a chceme se dostat z horních položek na spodní, zabere nám to spoustu času a pevné nervy. Řádek, na kterém se aktuálně nacházíme, má barevné zvýraznění.

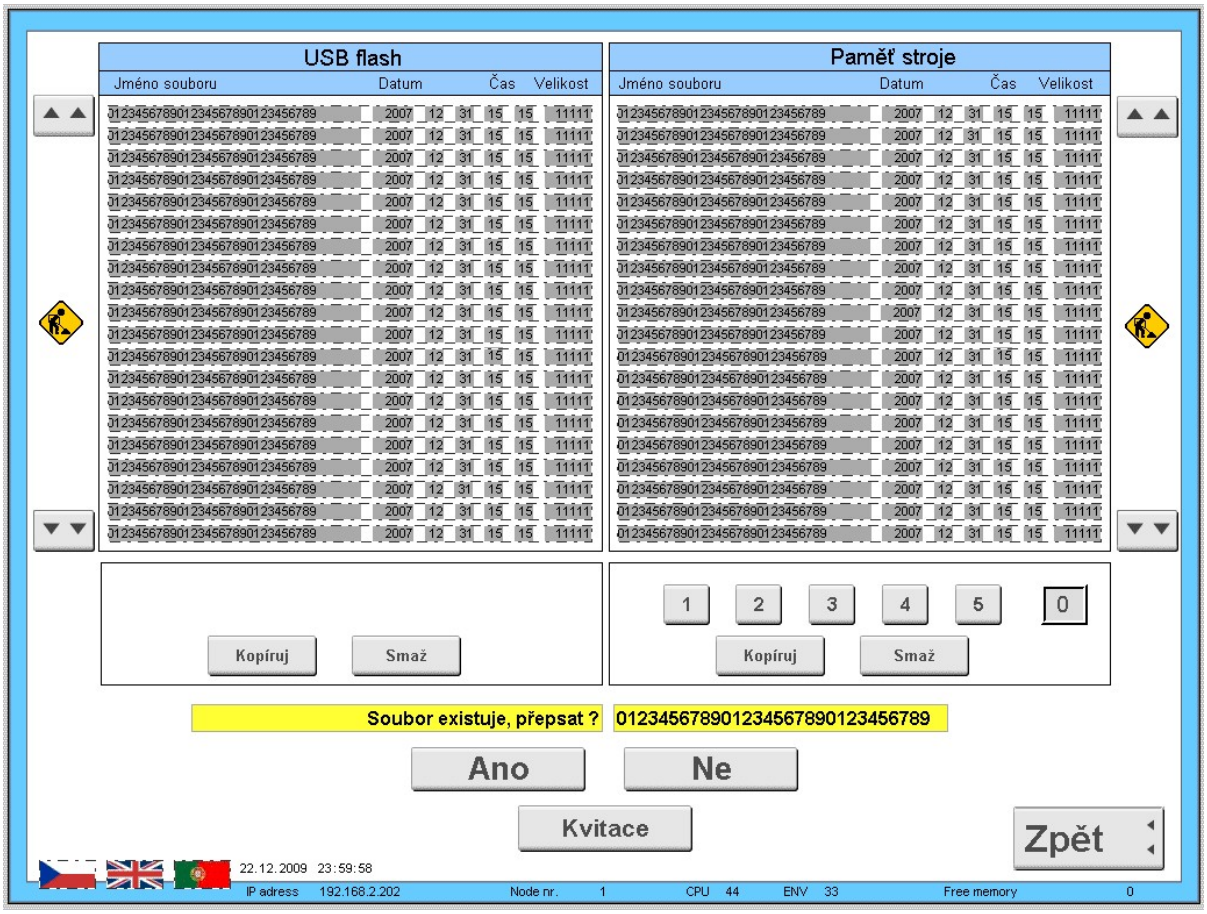
Při kopírování mezi CompactFlash a pevným diskem panelu je obrovská nevýhoda v nemožnosti libovolného kopírování mezi adresáři. Adresářů pro ukládání souborů je celkem pět, pojmenovány jsou pouze číselně od jedné do pěti. V jednom adresáři může být maximálně 500 programů, rozlišených jménem o délce 30 znaků. Délka jednoho programu je omezena na 5000 bloků. Blok programu je jeden řádek.

Tyto adresáře jsou na pevném disku panelu i na CompactFlash disku. Dotykem na některé tlačítko adresáře v pravé části (pevný disk) manažeru se nám automaticky otevře stejný adresář i na levé straně (CompactFlash disk). To znemožňuje zmíněné libovolné kopírování a ztěžuje mazání souborů. Pokud chceme smazat soubor v jiném adresáři, než se právě nacházíme, dotykem na zvolené tlačítko adresáře se aktivuje načtení nejdříve adresáře v pravé části manažeru. Až se zde načtou všechny položky, začnou se načítat položky v levé části a poté je teprve možné soubor smazat. Při načítání velkých adresářů tato operace může trvat i několik minut, během kterých nejsou ostatní

funkce zablokovány. Může tak dojít k nechtěnému zmáčknutí na jiné tlačítko a spuštění funkce, která v daném okamžiku může pouze uškodit.

V manažeru zcela chybí informace o volném místě na disku. V případě zaplnění disku není tato skutečnost uživateli sdělena a systém se při kopírování na plný disk zhroutí. Při zkoušení různých situací byly objeveny další nedostatky. Například, pokud máme již načtený adresář na USB disku, poté USB disk vyjmeme a začneme na něj kopírovat, manažer nevyhlásí chybu, že USB disk byl odpojen. Systém tak opět havaruje. Adresář totiž vizuálně zůstává načtený i po vyndání USB, takže uživatel si situaci nemusí uvědomit.

Podobný případ, jako s kopírováním na plný disk, nastal i při kopírování do plného adresáře. Počet programů v adresáři je omezen na 500, ale při kopírování 501. programu do adresáře opět není vyhlášena chyba.



Obrázek 3.3: Původní stav manažeru programů

3.4 Původní stav grafického rozhraní editoru programů

Editor programů (obr. 3.4) je používán, jak již název napovídá, k editaci programu. Obdobně jako u manažeru programů se pro pohyb mezi řádky používají pouze šipky nahoru a dolů, což je neefektivní. Chceme-li tedy nějaký řádek editovat, mazat či přidávat mezi existující řádky, musíme na něj najet pomocí šipek a pak teprve můžeme pokračovat s úpravami. I zde v editoru programů je aktuální řádek podbarven.

Dříve než můžeme tyto úkony vykonávat, musíme buď založit nový program, nebo program vybrat.

Pokud se rozhodneme pro výběr některého ze stávajících programů, skočíme na vrstvu *Layer_vyber*. Zde vybereme program pro svařování a vrátíme se zpět na vrstvu editace, kde je již otevřený program, jak znázorňuje obrázek 3.4.

Při založení nového programu se ocitneme ve vrstvě *Layer_saveAs*. Založíme nový program a vrátíme se zpět na editaci. Nyní máme tři možnosti, jak program vytvářet.

Tlačítkem „Přidej blok“ můžeme po jednom řádku přidávat data. Tato varianta se spíše hodí pro přidávání řádků do hotového programu, protože vytvořit svařovací program tímto by bylo příliš zdlouhavé a nebezpečné. Každý řádek programu má totiž 18 sloupců, do kterých vkládáme data – souřadnice os, rychlosti, proudy a podobně (kapitola 2.5).

Proto existuje i druhá varianta a to generování programu, na které se dostaneme přes tlačítko „Generování prg.“. V menu generování programů máme dvě varianty generování: generování programu válec a generování programu obdélník.

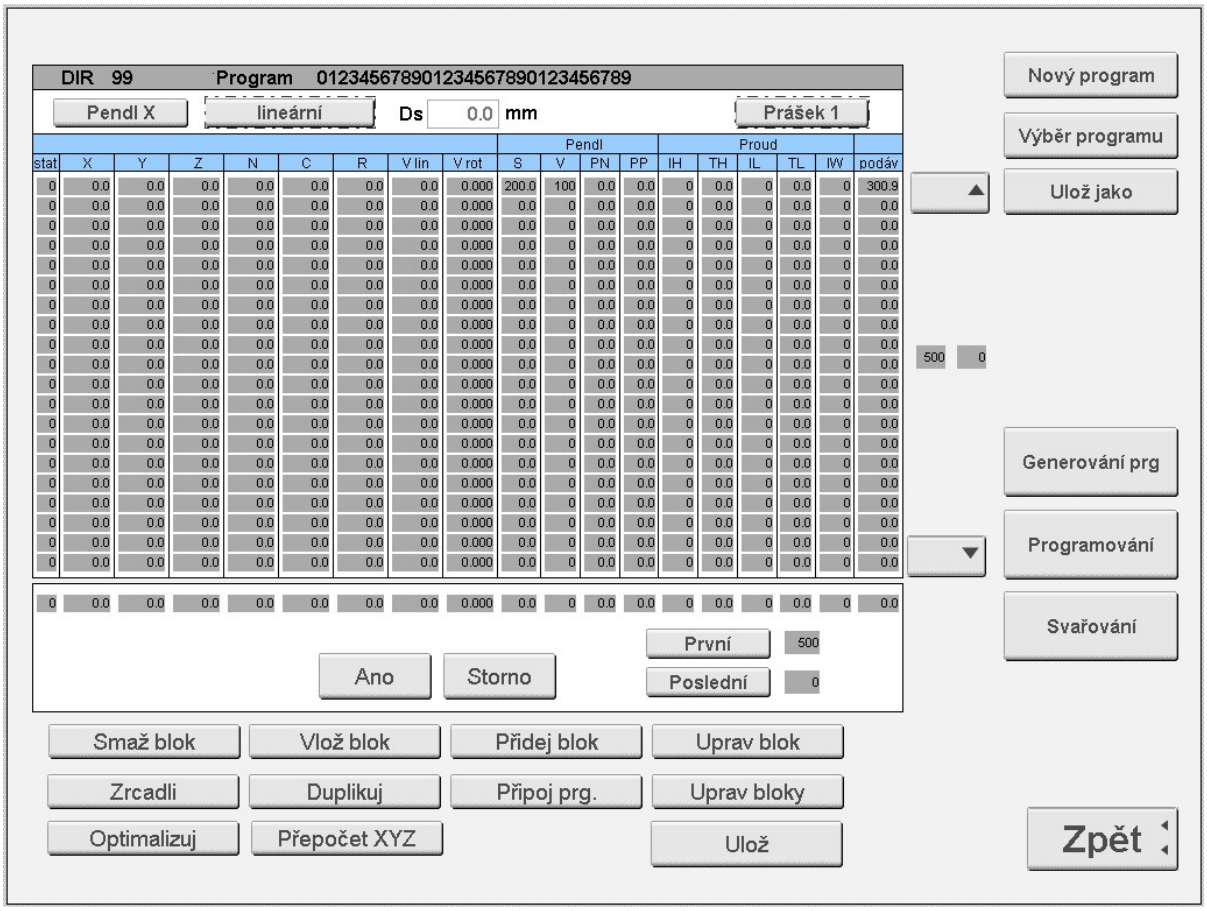
Poslední možnost vytvoření programu je tzv. „Ruční režim“, kdy svářeč navařuje bez připraveného programu a vše si koriguje sám. Program se pak ukládá a svářeč ho může později použít znovu. „Ruční režim“ je však dostupný z hlavního menu, nikoli z editoru.

Pokud jsme tedy vytvořili program v editoru, stiskem tlačítka „Svařování“ se dostaneme do další obrazovky, kde máme již svařovací parametry a můžeme začít navařovat. Stiskem tlačítka se nám také data vytvořená obsluhou stroje nahrají přímo do stroje následujícím způsobem:

1. Jednotlivé bloky (řádky) v editační obrazovce jsou uložené ve struktuře, ze které se vytvoří CNC kód pomocí našeho zdrojového kódu.
2. CNC kódu již rozumí software firmy B+R – ARNC0. Tento software řídí ve stroji pohyby až šesti os.
3. ARNC0 řídí ACOPOS, což je servopohon firmy B+R, který hýbe motorem. Motor dále hýbe převodovkou, která nám již vykonává samotný pohyb stolu polohovadla, rameny stroje atd.

Vytvořený program je potřeba uložit, k čemuž slouží tlačítko „Ulož“. Nicméně toto tlačítko se nám ukazuje stále, i když jsme nic nezměnili. Pokud z obrazovky odcházíme na jinou, například si chceme vytvořit nový program nebo si vybrat jiný program, a nemáme stávající program uložený, nic nás neupozorní, abychom si ho uložili. O vytvořený program můžeme tedy velmi snadno přijít.

Podobně jako u manažeru programů je i zde problém s nezablokovanými funkcemi. Je dobré, aby při vykonávání určitého druhu editace (přidání řádku, editace řádku, mazání řádku, apod.) byly ostatní funkce zablokovány.



Obrázek 3.4: Původní stav editoru programů

4 Analýza požadavků a návrh řešení

Obsahem této kapitoly jsou požadavky na software. Budeme řešit, co je zapotřebí vylepšit z hlediska implementace a grafického rozhraní. Také si ukážeme nové možnosti software.

4.1 Konzultace a všeobecné informace

Po důkladném prozkoumání grafického rozhraní a funkcí, které nabízí, jsem se sešla nejdříve se zadavatelem práce a později i se zaměstnanci firmy pracující se svařovacími automaty.

Ze strany zaměstnanců mi byly sděleny především požadavky na uživatelské prostředí a jednoduchost ovládání. Pro lepší pochopení způsobu práce se svařovacími automaty a používání programu jsem se rozhodla pro osobní přezkoumání problémů přímo v provozu.

Ačkoli již několik let nebyly provedeny významné změny v rozhraní a obsluha stroje je zkušená, zjistila jsem, že počítačová gramotnost u těchto lidí zaostává. Je tedy nutné eliminovat veškeré funkce na minimum úkonů a pokusit se přiblížit k podobnému ovládání jako má operační systém Windows. Lidé se snaží vykonávat a hledat podobné funkce jako právě ve Windows.

Další důvod, proč vytvořit jednoduché ovládání, je v komunikaci a vysvětlování ovládání v zahraničí. Stroje jsou prodávány i do zemí jako je Maďarsko, Rumunsko, Indie, Ukrajina nebo Rusko. Zde je jazyková bariéra v dorozumění. Většina zahraniční obsluhy strojů totiž neovládá jiný jazyk než ten svůj a vysvětlovat, jak zacházet se strojem, bývá obtížné. Navíc je v těchto zemích větší počítačová negramotnost než u nás, v Německu, Rakousku, Francii, USA či jiných zemích, kam jsou stroje prodávány. Pouhé vysvětlení banalit, jako je nemožná existence dvou stejně pojmenovaných souborů v jednom adresáři, je u některých pracovníků úplnou novinkou. Nechci tím nijak snižovat odborné schopnosti pracovníků, ale musíme s tímto faktem počítat.

Velkým problémem, který není viděn na první pohled, je ve velikosti komponent uživatelského prostředí. Obsluha stroje poměrně často při své práci používá ochranné rukavice. Ty znemožňují přesné dotyky na obrazovce. Proto se snažíme o co největší možnou velikost grafických komponent. Ne vždy je to ale možné. Tato skutečnost způsobila první problém, bohužel nepřekonatelný. Návrhem ze strany zadavatele bylo totiž všeobecně přidání více počtu řádků na stránce. Ať už se jedná o manažer či editor programů, není z důvodu velikosti obrazovky možné přidávat řádky, aniž bychom museli jiné komponenty ještě více zmenšovat nebo dokonce přesouvat na jiné obrazovky.

4.2 Rozhraní manažeru programů

V kapitole 3.3 byly popsány problémy s touto částí rozhraní. Na návrh zadavatele práce a samotné obsluhy stroje jsme přidali další varianty posunu mezi řádky. K dosavadním šipkám nahoru a dolů přibyla i tlačítka „Page Up“ a „Page Down“ pro pohyb, na začátek adresáře a na konec adresáře. Abychom využili předností dotykových panelů, na jednotlivé řádky je možné najíždět i pouhým dotykem na samotný řádek, přičemž lze kombinovat dotykové rolování s tlačítkovým. Rolování je možné souběžně v levé i v pravé části manažeru programů. Toto vylepšení přineslo urychlení vyhledávání souborů.

Nejen pro pohodlnější kopírování souborů byla přidána možnost kopírovat z pevného disku na pevný disk nebo CompactFlash disk a naopak. Kopírování navíc již není omezené pouze mezi stejnými adresáři, jak jsme si popisovali v kapitole 3.3. Můžeme kopírovat a mazat soubory, jak potřebujeme. Přidali jsme také dostatečné množství opatření, aby se systém nezhroutil, když

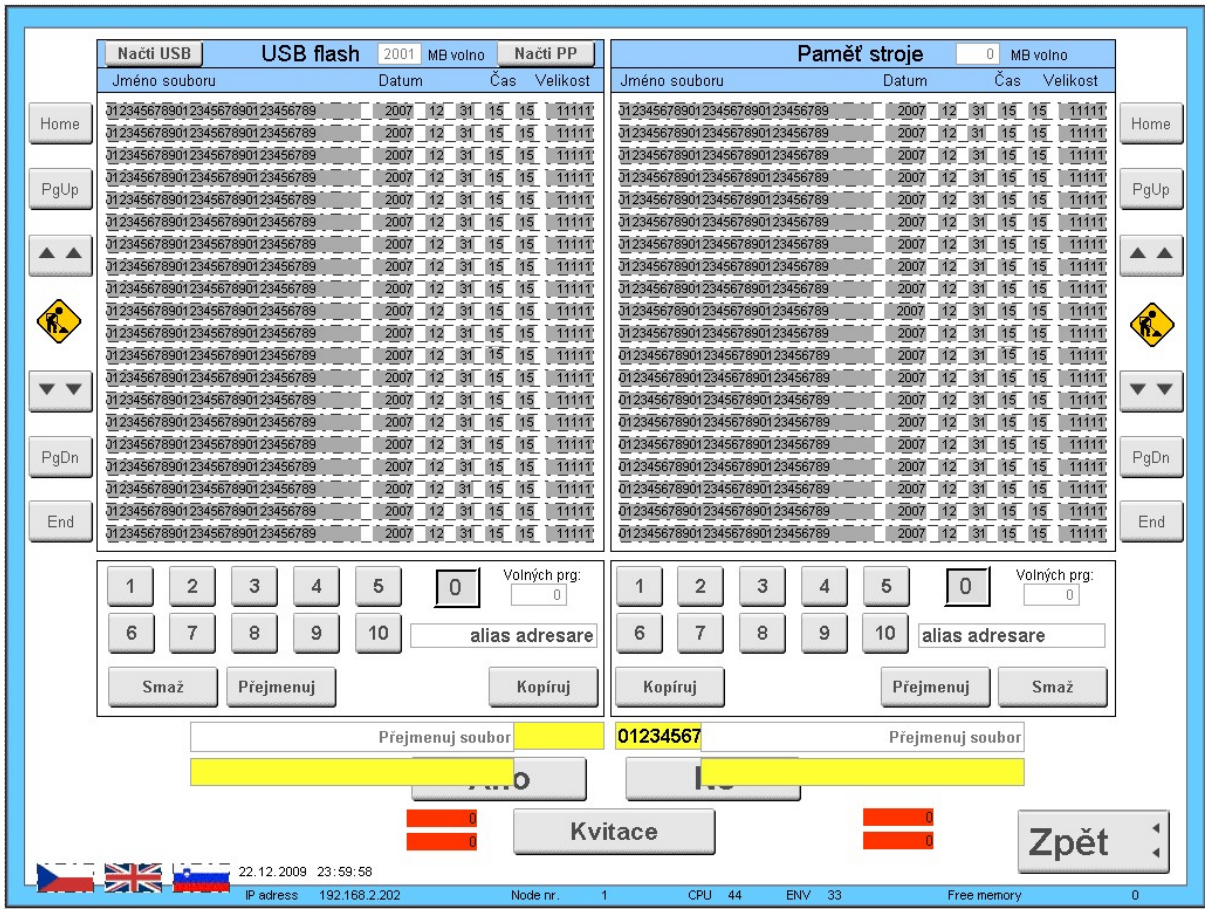
kopírujeme na plný disk, plný adresář nebo na již vyjmutý USB disk. Ve všech případech se objeví chybové hlášení s popisem chyby. Aby si uživatel sám mohl kontrolovat zbývající kapacitu, vložili jsme mu tyto informace přímo na obrazovku.

Na přání zákazníka se počet dostupných adresářů zvětšil z pěti na deset. Do budoucnosti se počítá i s možností, kdy si uživatel sám vytvoří potřebný počet adresářů na pevném disku panelu.

Novou nutnou funkcí je přejmenování souborů a používání aliasů u názvů složek, které jsou zatím pojmenovány pouze od jedné do pěti. Kvůli nedostatku místa na obrazovce manažeru jsme se rozhodli umístit funkci pojmenování souborů (aliasy) do obrazovky nastavení, dostupné z hlavního menu.

Dalšího urychlení jsme dosáhli změnou implementace, kdy jsme umožnili načítat v jedné části například velký adresář a mezitím v druhé části umožnili uživateli hledat či mazat nějaký soubor, přepínat adresáře nebo disky. Zde si opět musíme uvědomit, že jsme v real-time aplikaci a máme k dispozici pouze jeden procesor. Vytvořit jednu funkci a tu pak volat zároveň na dvou místech proto nelze. Zvolili jsem tedy jiný možný postup. Založili jsme dva bloky, jeden pro levou část obrazovky a druhý pro pravou část, a v případě potřeby je voláme.

Na obrázku 4.1 si můžete prohlédnout vylepšenou verzi manažeru programů. Je vidět, že jednotlivé komponenty lze překrývat a za běhu programu je zviditelňujeme a zneviditelňujeme podle potřeby.



Obrázek 4.1: Výsledek BP - stav manažeru programů

4.3 Rozhraní editoru programů

V editoru programů jsme řešili obdobný problém s posunem mezi řádky. I zde byla přidána tlačítka „Page Up“ a „Page Down“ pro pohyb, na začátek adresáře a na konec adresáře.

Pro jakoukoli editaci řádku bylo dosavadní řešení zbytečně složité. Nejdříve se muselo najet na řádek a pak až editovat, mazat, kopírovat a podobně. Nyní je krom této možnosti využito opět vlastnosti dotykového panelu a každá buňka v editoru je samostatným prvkem, který můžeme editovat pouhým dotykem.

Každé vylepšení však odkrývá nový problém a ani zde tomu nebylo jinak. V editoru můžeme viditelně zobrazit maximálně dvacet řádků. Pokud má otevřený svařovací program například deset řádků, zobrazíme pouze prvních deset řádků a zbylých deset zneviditelníme. Při původním řešení jsme měli možnost posouvat pouze po jednom řádku nahoru a po jednom dolů. Tudíž nám stačilo vědět, kolik řádků má program, a kontrolovat, na jakém řádku právě jsme a jestli už je to poslední řádek, případně první, a podle toho jsme se rozhodovali, zda řádek zviditelníme, nebo už jsme na konci a další řádky zneviditelníme.

Při dotyku je jedno, kam uživatel sáhne. Vzhledem k tomu, že při dotyku na komponentu se objeví numerická klávesnice, pomocí které vložíme čísla a poté jedním dotykem uložíme, může dojít k objevení klávesnice i na neviditelné buňce. Neviditelnost buňky je totiž zajištěna pomocí barev, nikoli pomocí opravdového statusu grafické konponenty. Naskytly se dvě možnosti, jak tento problém vyřešit. Buď změnit zobrazení komponenty podle statusu, nikoli barvy, nebo počítat zbylé řádky v editoru a ty dát do statusu neviditelnosti.

Kvůli přehlednosti i efektivitě kódu jsme se rozhodli pro první možnost. Pokud bychom zvolili druhou možnost, museli bychom přidat další cyklus pro zneviditelnění nepoužívaných řádků.

Problémové bylo tlačítko „Ulož“, které bylo na obrazovce viditelné neustále, takže ho uživatel lehce přehlédnul. Nyní se tlačítko objevuje pouze ve chvíli, kdy uživatel změnil nějakou hodnotu ve sloupci, připojil program, změnil typ pendlu, vygeneroval úplně nový program, atd. Pokud si obsluha tlačítka „Ulož“ nevšimne ani nyní a hodlá se přesunout na jinou obrazovku, dostane hlášení o neuloženém programu. Pokud program uloží, název programu se podsvítí žlutě, aby bylo jasné, že program je nějakým způsobem pozměněný. Kdykoli tento program znovu otevře, bude stále podsvícený.

V editační obrazovce si také uživatel volí typ podavače, čímž může být drát, prášek 1, prášek 2, nebo MIG (svařování drátem v ochranné atmosféře). Funguje to na principu tlačítka, kdy se dotykem na tlačítko cyklicky opakují tyto druhy podavače a uživatel si nějaký vybere. Problém je v tom, že ne všechny automaty mají všechny typy podavače, proto jsme upravili výběr tak, aby se zobrazovaly pouze ty podavače, které automat opravdu má.

Také jsme splnili přání zákazníka, který požadoval, aby si svářeč v editoru mohl k otevřenému programu zapsat poznámky. Pro snadnější úpravu svařovacího programu jsme navíc přidali možnost mazat více řádků najednou. Uživatel si zvolí rozsah mazání a řádky vymaže.

Zablokovali jsme veškeré funkce, pokud není otevřený nebo založený nový program.

V editoru jsme měli doposud konstantních 18 sloupců. Nyní jich máme 12 – 19, přičemž v reálném případě jich budeme mít 15 – 18. Pro připomenutí se jedná o následující sloupce:

Status, osy X, Y, Z, N, C, R, rotační a lineární rychlost, parametry proudu, podávání a pendlu. Pole status je pouze výstupní, automaticky generované a nelze jej měnit. Ostatní parametry měnit lze.

Nyní se dostáváme k otázce, proč se počet sloupců mění. Co zákazník, to přání. Ne všichni zákazníci potřebují na svém automatu plný počet os. Například plazmový navařovací automat PPC 250 GMR má všech 6 os, ale automat PPC 250 R má pouze dvě osy týkající se hořáku a jednu osu vztahující se na polohovadlo. Přesto se u PPC 250 R zobrazuje na obrazovce editoru všech 6 os, což je pro uživatele matoucí. Proto jsme v nové verzi software přidali možnost zobrazování os. Podle toho, jaký stroj si zákazník objednal, zobrazíme příslušné osy. Toto nastavení je již z výroby a uživatelé nemají možnost měnit zobrazení z bezpečnostních důvodů. Implementace tohoto vylepšení nebyla nijak složitá. Pouze jsme přidali sloupcům statusy viditelnosti a ty měníme v inicializační části zdrojového kódu.

Všechny parametry mají svůj rozsah hodnot. Kontrola hodnot je v Automation Studiu velice jednoduchá. Vstupní pole mají možnost buď přímo zadat minimální a maximální hodnotu, nebo připojit ke komponentě proměnné a ty naplňovat ve zdrojovém kódu. První možnost je v našem případě velmi nepraktická. Na obrazovce editoru máme celkem 340 buněk, které můžeme měnit. Pokud bychom zadali pevnou hodnotu min/max a později ji potřebovali změnit, museli bychom každé políčko projít a hodnoty ručně upravit. Tato operace by zabrala spoustu času. Proto využijeme druhé možnosti a založíme si pro každý sloupec dvě proměnné (min a max), které ke komponentám připojíme a ve zdrojovém kódu naplníme. Pak lze jednoduše hodnoty přepsat.

Některé parametry v řádku jsou na sobě závislé, proto musíme kontrolovat vložené parametry, aby při spuštění programu nedošlo ke kolizi. Pokud obsluha stroje zadá špatné parametry, buňky změní barevné podsvícení na červenou a data se neuloží. Toto už nelze udělat přes rozhraní Automation Studia, musíme kontrolovat sami. Jedná se o následující kontroly:

- Pokud je proud IH menší, než jeho nastavená minimální hodnota, je to chyba. Možná je matoucí, jak může být proud menší než minimální hodnota. Může se totiž stát, že uživatel má otevřený nějaký starý svařovací program, kdy tyto kontroly ještě nebyly a v dnešním systému by to mohlo způsobit škody.

- V případě, že máme nenulovou lineární i rotační rychlost, je to opět chyba. Může být zadaná právě jedna rychlost, nebo žádná.

- Předpokládáme, že máme zadanou buď lineární, nebo rotační rychlost. V tom případě musíme mít nenulový i proud IH, jinak je to chyba.

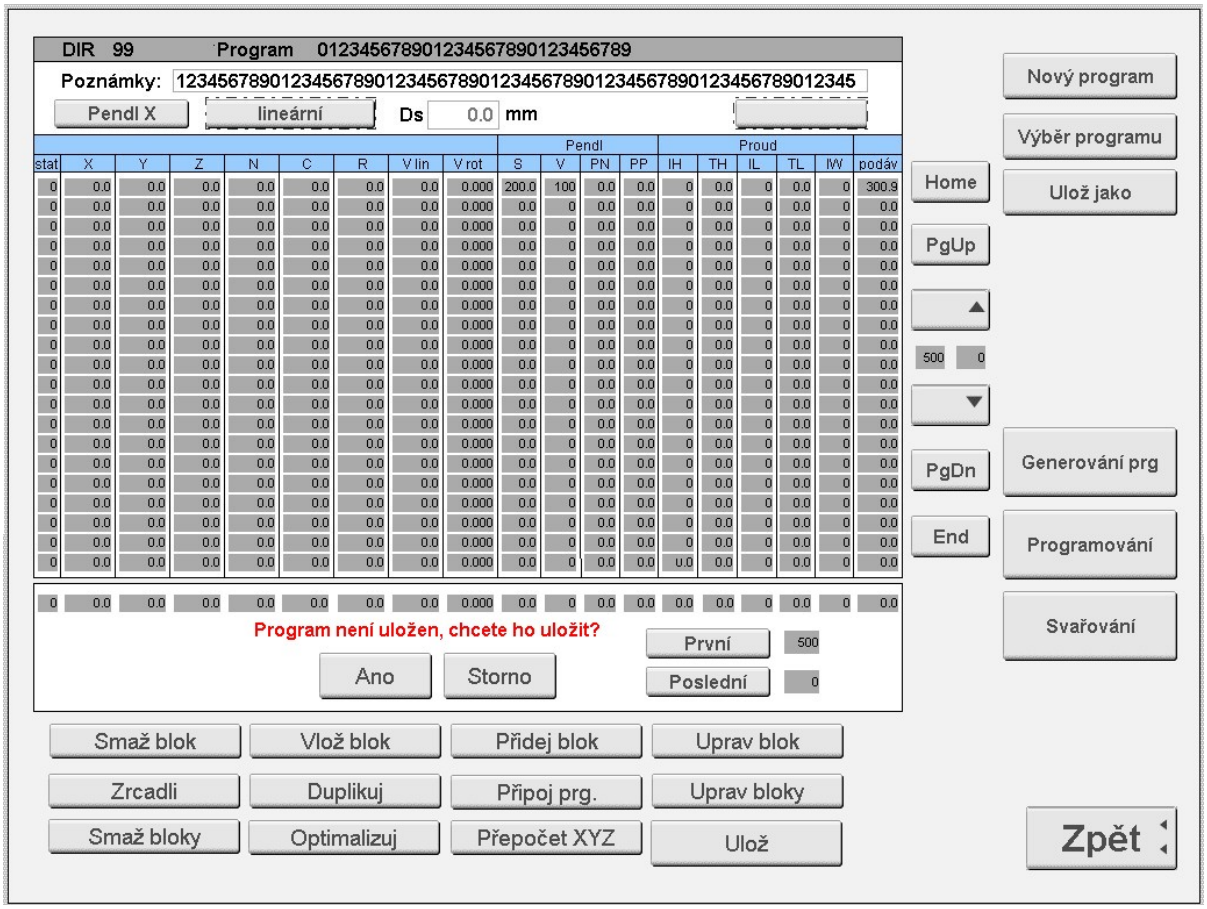
- Pokud máme obě rychlosti nulové a máme proud IH nenulový, nastává chyba.

Dokud všechny čtyři chyby obsluha stroje neodstraní, aplikace bude čekat stále ve stejném case. Až po odstranění chyb program skočí do další části zdrojového kódu a můžeme svařovací program uložit, případně s ním jinak nakládat.

Když jsme si popisovali jednotlivé sloupce, byla zmínka o automaticky generovaném statusu, který může nabývat hodnot 1 až 3. Status závisí například na rotační a lineární rychlosti. Status má hodnotu 1, pokud je nenulová lineární rychlost. Hodnoty 2 nabývá, pokud jsou obě rychlosti nulové a status 3 je v případě, že je nenulová rotační rychlost.

Proto musíme podle změn jednotlivých rychlostí měnit i hodnoty statusu.

Nové rozhraní editoru programů si můžete prohlédnout na obrázku 4.2.



Obrázek 4.2: Výsledek BP - stav manažeru programů

4.4 Export svařovacího programu do CSV

Každý svařovací program je uložen v datové struktuře a tudíž si běžně nemůžeme prohlédnout uložený program v počítači. Abychom takto uživatele neomezovali, rozhodli jsme se pro možnost exportovat programy do formátu CSV, který je snadno otevíratelný například v programu MS Excel.

Prvotní nápad byl umístit export CSV do manažeru programů, kde by si uživatel vybral požadovaný program v jednom z deseti adresářů, dal „Kopírovat do CSV“ a program by se vyexportoval do otevřeného adresáře v druhém okně manažeru. Tento návrh jsme ovšem upravili, protože by se nám pomíchaly CSV soubory s programy určenými pro svařování.

Založili jsme tedy úplně novou obrazovku, která je dostupná z hlavního menu. Vzhledově je velmi podobná manažeru programů.

Postup kopírování je však stejný - v pravém okně si uživatel vybere program, stiskne tlačítko „Kopírovat do CSV“ a soubor se vyexportuje do adresáře otevřeného v levém okně. Levé okno však načítá jiné adresáře než je obvyklé. Načtený adresář je vytvořený speciálně pro CSV soubory a žádné jiné. Tyto adresáře jsou totožné jak na pevném disku panelu, tak i na USB.

Opačné kopírování, tedy z CSV do naší struktury není zatím možné. Muselo by se provádět nespočetné množství kontrol CSV souboru, aby při spuštění programu na stroji nedošlo k poškození jak stroje, tak navařované součásti. Nicméně i touto variantou se časem budeme zabývat. Není to však záležitost pár týdnů, nýbrž měsíců a dlouhodobého testování mimo běžný provoz.

4.4.1 Postup při exportu

Jak bylo již zmíněno, v pravé části manažeru exportu CSV (dále pouze manažer CSV) si uživatel najde svařovací program, který chce exportovat. Stiskem na tlačítko „Kopírovat do CSV“ se spustí proces vytváření souboru v druhé části manažeru, respektive na pevném disku panelu nebo na USB flash disku, podle toho, kam si uživatel vybral kopírovat.

Nejdříve se vytvoří název vyexportovaného programu. Ten se bude skládat z původního názvu souboru a koncovky „csv“. Vytvoříme soubor s tímto názvem v požadovaném adresáři a disku.

V případě, že soubor již existuje, vyskočí hlášení, jestli chceme soubor přepsat.

Do souboru budeme zapisovat hlavičku programu (název a typ programu, datum a čas poslední změny, atd.) a data v CSV formátu. Zapsanou hlavičkou mírně narušíme pravidelnou strukturu CSV, ale v našem případě to nevadí.

Popisovali jsme si, že program se skládá z jednotlivých bloků programu (řádků) a každý blok má 12-19 sloupců s různými daty. I nyní se musíme řídit tím, zda daný typ stroje umožňuje všechny sloupce, nebo jich má méně. Proto musíme kontrolovat, abychom neexportovali i sloupce, které ve skutečnosti zákazník na svém stroji vůbec nemá.

Nejdříve do souboru zapíšeme hlavičku CSV dat, oddělené středníkem a dále zapisujeme samotné parametry, též oddělené středníkem.

Po zapsání celého programu si může uživatel ve svém počítači program prohlížet a archivovat.

5 Generování programů

Do této doby existovaly pouze dvě možnosti, jak vygenerovat nový program. Generování programu válec a obdélník. Pojem generování není příliš přesný. Zde se jedná o něco mezi plněním šablony a skutečným generováním. Ve firmě se však ustálil pojem generování, proto ho budeme používat i zde.

Důvodů, proč implementovat nové způsoby, je hned několik.

1. Musíme být o krok napřed, než konkurence.
2. Plníme přání zákazníkům.
3. Chceme být levnější.
4. Chceme docílit co největší přesnosti návaru.

V České republice má firma přibližně pět velkých konkurentů, proto musíme sledovat jejich výrobní program. Naštěstí pro nás má většina konkurenčních firem jeden nedostatek. Chybí jim programátoři, nebo mají pouze externí programátory. Proto je pro ně náročnější implementovat rozšíření jejich software. S tím souvisí i skutečnost příchodu nových zákazníků do naší firmy se specifickým zadáním.

Snahou všech výrobců je mít co nejnižší ceny. Lidská práce a vyškolení je příliš drahé, proto se vyplatí investovat do nového software. V našem případě je leckdy výnosnější implementovat novou možnost generování svařovacího programu, než zaplatit kvalitního svářeče, který bude navařovat v ručním režimu. Ruční režim také přináší i vetší riziko nepřesného návaru. Pokud to opravdu není zkušený svářeč, velice snadno může dojít ke špatnému navaření.

Nyní si ukážeme nové možnosti generování svařovacích programů.

5.1 Import CNC programů

S poptávkou na koupi nového automatu přišla i podmínka importovat vlastní soubory vygenerované z konstruktérských a obráběcích programů typu CAD/CAM. Prozatím máme tři typy souborů: soubory s příponou „txt“, s příponou „h“ (zde se nejedná o hlavičkové soubory programovacího jazyka C) a s příponou „nc“.

Všechny tři typy souborů mají určitou strukturu, proto jsem se rozhodla uplatnit znalosti konečných stavových automatů. Vznikly tedy tři konečné automaty, jejichž úkolem je načítat souřadnice os, statusy a proudy do struktury. V případě, že hořák nebo polohovadlo není v nulové poloze, je možné souřadnice os připočítat ke generovaným souřadnicím ze souboru stiskem tlačítka „Aktuální poloha“. Dále si můžeme libovolně nastavit proudy, nebo ponechat jejich implicitní nastavení.

Podle těchto souřadnic se řídí hořák při navařování. O vzniklých konečných automatech se více dočtete v následujících podkapitolách. Rozebereme si však pouze dva, protože jejich struktury jsou velmi podobné.

Vzhledem k tomu, že import CNC programů je jakýmsi generováním programů, umístění bylo jasné. Na import se dostaneme ze stránky editoru programů přes tlačítko generování programů a dále si pak můžeme vybrat variantu generování. Přičemž nyní je přidána i tato možnost. Na obrázku 5.1 je vyobrazeno grafické rozhraní importu CNC programů.

Prvním krokem při vytváření nové obrazovky bylo vytvořit její grafický vzhled. Použili jsme podobné prvky jako jsou na jiných obrazovkách, aby uživatelé nebyli zmateni jiným ovládáním. Pro výběr souboru jsme použili obdobu manažeru programů, přes který si vybíráme soubor pro importování. Při výběru programu se můžeme rozhodnout mezi pevným diskem panelu nebo

CompactFlash diskem. Na obou discích jsou vytvořeny speciální adresáře, kde jsou nahrané soubory určené pro import. Na pevném disku panelu je to adresář „cnc“ a na CompactFlash disku adresář „usbcnc“.

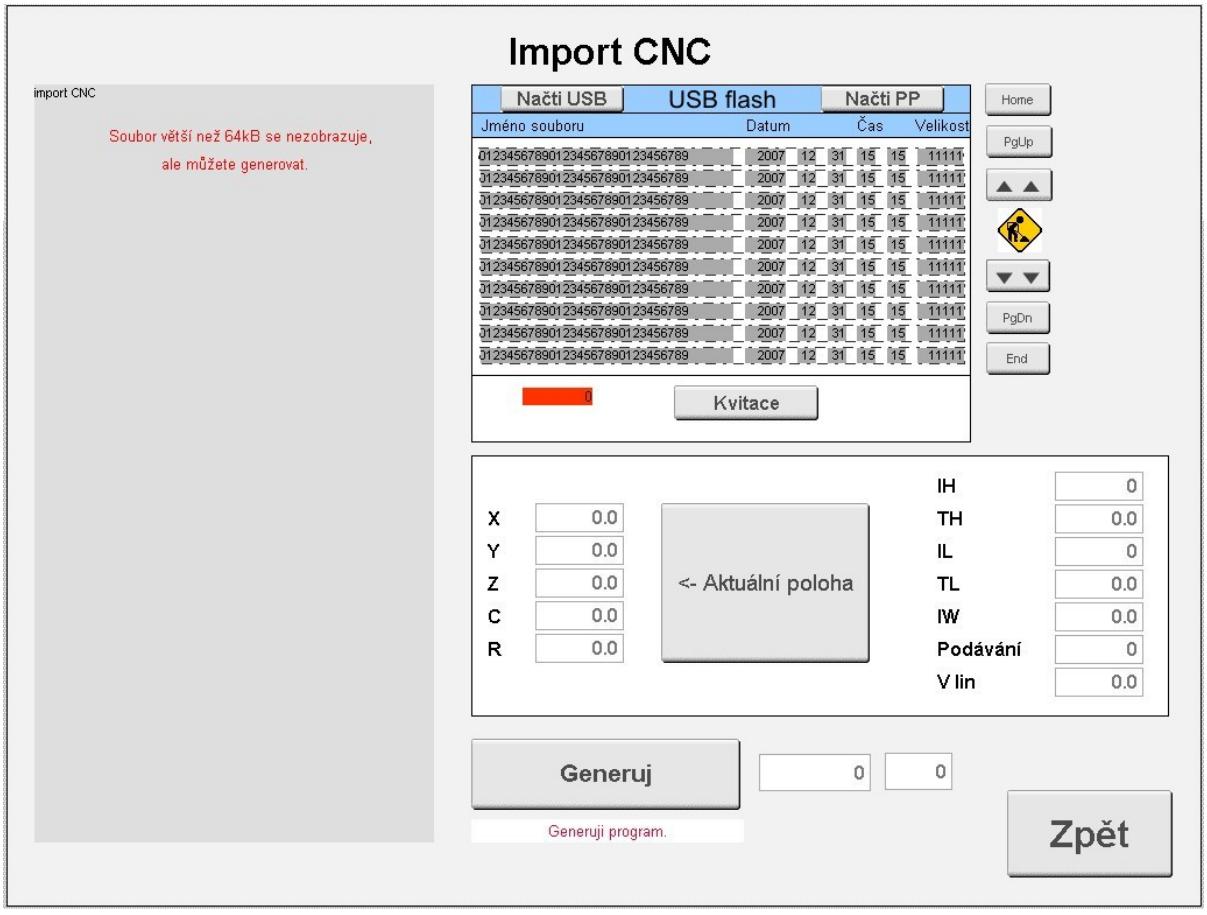
Pro pohyb mezi řádky jsou zde stejné prvky jako u manažeru nebo editoru programů. Jedná se tedy o tlačítka „Page Up“, „Page Down“, šipka nahoru a dolů o jeden řádek, na začátek a na konec adresáře. Ani zde nebylo zapomenuto na dotykové najetí na řádek.

Výběr požadovaného souboru potvrdíme stiskem tlačítka „Načti“. Pokud vše proběhne v pořádku a program nevyhlásí chybu při otevírání, celý soubor se zobrazí v editačním okně v levé části obrazovky. Toto editační okno je grafickou komponentou Automation Studia. Lze v něm otevřený soubor podle potřeby upravovat. Nesmíme však zapomenout na jedinou podmínku a tou je maximální počet bloků programu.

V tomto kroku, kdy máme načtený soubor, program odblokuje tlačítko „Generuj“ a spustí se generování programů, konkrétně zmíněné konečné automaty podle toho, jakou příponu načtený soubor má. V případě, že zadáme soubor s neznámou příponou, dojde k vyhlášení chyby a generování neproběhne.

Po ukončení generování, jehož průběh můžeme sledovat v políčku napravo od tlačítka „Generuj“, se stiskem tlačítka „Zpět“ dostaneme zpátky na editor programů, kde budou vygenerovaná data z importovaného souboru načtena v buňkách editoru. Nyní můžeme soubor upravovat a ukládat.

Podle velikosti načteného souboru zabere generování samotného programu poměrně dost času. Stává se, že obsluha mezitím dělá jiné věci a nevšimne si, že generování je již u konce. Pokusili jsme se přidat zvukové upozornění na konec generování. Tento návrh jsme však v praxi zamítli, protože samotný stroj vydává okolo 70 dB a to nepočítáme okolní spuštěné stroje, rádio a podobně. Zvuková výstraha je tudíž málo slyšitelná.



Obrázek 5.1: Grafické rozhraní importu CNC programů

5.1.1 Konečný automat pro soubory s příponou txt

Pokud zdrojový kód rozezná u souboru příponu txt, spustí se následující konečný automat. Pro ukázku níže vkládám část souboru určeného pro import, který je vstupem pro náš konečný automat.

Soubory jsou generované z programů typu CAD/CAM, což jsou programy pro konstruování a obrábění. Mají vždy stejnou strukturu a proto se použití konečného automatu samo vybízelo.

Každý txt soubor má hlavičku o několika řádcích, jejichž počet není vždy stejný. Každý řádek začíná levou hranatou uvozovkou „[“ a končí pravou hranatou uvozovkou “]”.

Po hlavičce následují další řádky, které pro nás nejsou důležité, proto je přeskočíme. Významné řádky začínají až v momentě, kdy na řádku narazíme na G00, nebo G01. Po těchto třech znacích začínají souřadnice, které budou určovat pohyb hořáku po navařované součásti. Souřadnice jsou vždy v pořadí X, Y, Z, přičemž ne na každém řádku se musí vyskytovat všechny souřadnice. Na řádku může být ještě souřadnice F, kterou ale nepotřebujeme. Pokud se souřadnice nenachází, znamená to, že se hodnota souřadnice nezměnila a platí předchozí načtená.

Po symbolu souřadnice (X, Y, Z, F) bezprostředně následuje číselná hodnota. Na konci souboru je obvykle pár nepodstatných řádků, kterých si nebudeme všímat.

Ukázkový soubor

```
[ PRODUKT:  C:\Documents and Settings\MarkoKuk\UNIOR
  ZRECE\ORODJARNA\DOBAVITELJI_ORODIJ\ROSCO_KSK\VARJENJE_KSK\
  14224_varjenje_junij 2010\programi varjenje.prt ]
[ IZDELAL:  MarkoKuk ]
[ DATUM:  Fri Jun 18 13:46:08 2010 ]
[ INT1 ]
N10 G17
N11 G90
N12 G71
N13 M891
N14 G733D8=33
N15 G00 X-169.278 Y38.184
N16 G00 Z2.
N17 G00 Z0.0
N18 G01 X-168.294 F1000
N19 G01 X-167.629 Y38.106
N20 G01 X-167.535 Y38.082
N21 G00 X-165.969 Y37.712
atd.
```

Pro implementaci konečného automatu jsme původně využili konstrukci switch, přičemž téměř každý stav konečného automatu měl svůj „case“. Ukázalo se však, že při importování velmi dlouhého souboru je tato konstrukce příliš pomalá, protože v každém „case“ trávíme 10 ms. Tato doba je dána zařazením úlohy v patřičné třídě. Místo konstrukce switch jsme nakonec použili for cyklus s podmínkami if-else, které nemusí čekat. Vyskytl se ovšem problém s vypršením časového limitu, který je 30 s na jeden „case“. Přesto, že používáme if-else, nacházíme se v určitém kroku switche a čekáme, dokud for cyklus neskončí. U dlouhých souborů nestačí 30 sekund na procházení celého souboru, takže program spadne do takzvaného „Service mode“. Odtud je jen jedna cesta zpět – vypnutí a zapnutí stroje, což je v praxi nepřijatelné. Řešením bylo periodicky vyskakovat po určité době do jiného case a zase zpět a pokračovat v cyklu.

Nyní si ukážeme, jak pracuje automat (obr. 5.2). Ze souboru je načítán znak po znaku. Protože zde máme trochu jiné možnosti pro práci se soubory a načítání znaků, nemůžeme vždy používat klasické funkce jazyka C, ale musíme využít i B&R knihovny.

Víme, že podstatnými znaky jsou levá hranatá uvozovka, znak G nebo ostatní znaky. Pokud tedy automat narazí ve stavu 0 na znak „[“, skáčeme do stavu 20. V tomto stavu zůstaneme do doby, kdy na vstup přijde znak „]“. Tím nám hlavička končí a skáčeme zpět do stavu 0.

Pokud ve stavu 0 automat narazí na písmeno „G“, načteme další písmeno. Když je načtená číslice „0“, načteme další znak, jinak opět skočíme do stavu 0.

Je-li dalším znakem 0, znamená to, že jsme načetli G00 a status příslušného řádku bude 2. V případě, že už máme načtený alespoň jeden řádek, uložíme si předchozí načtené souřadnice do struktury. Tyto hodnoty pak buď přepíšeme nebo nepřepíšeme, podle toho, zda se budou na řádku vyskytovat všechny souřadnice.

Je-li načteným znakem 1, máme načteno G01 a status řádku je 1. Opět uložíme hodnoty souřadnic do struktury, jako v případě načtení G00. Navíc ještě do struktury uložíme všechny parametry, které zadal uživatel v obrazovce importu CNC programů.

Po načtení G00 nebo G01 skáčeme do stavu 10 a načteme všechny bílé znaky krom znaku odřádkování. Vzhledem k tomu, že soubor má stálou strukturu, víme, že znak načtený po G01/G00 bude jeden znak mezery (znak mezery v KA na obrázku 5.2 znázorněn jako "_"), proto nemusíme dělat žádné speciální kontroly. Přesto v případě většího počtu mezer zůstává automat ve stavu 10 a načte všechny mezery, dokud nenarazí na některý symbol souřadnice (X, Y, Z, F) nebo na odřádkování.

Odřádkování znamená skok do stavu 0 a také posunutí indexu řádku o jeden výš.

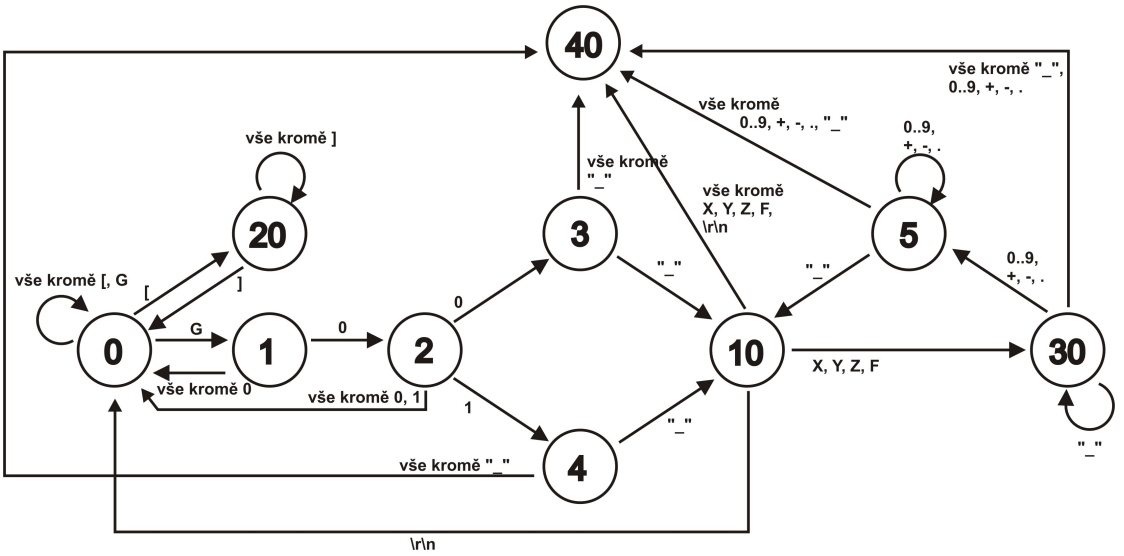
Symbol souřadnice nás odskočí na krok 30 a také si automat zapamatuje, o jakou souřadnici se jedná.

Krok 30 přeskakuje případné mezery mezi symbolem souřadnice a samotnou hodnotou. Až narazí na znak číslice, tečky nebo znaménka minus, automat se zastaví v tomto kroku a načte zbytek čísla. Po načtení čísla se rozhodne, k jaké souřadnici hodnota patří a uloží se do struktury.

Následuje znovu krok 10, ve kterým načteme buď další symbol souřadnice, nebo odřádkování.

Ačkoli text procházející automatem má pevně danou strukturu a nemělo by se stát, že se objeví nějaký nečekaný znak, pro jistotu existuje i chybový stav 40, do kterého se v takových případech spadne.

Tímto způsobem se postupně načtou všechny podstatné řádku v souboru, hodnoty se uloží do struktury a na obrazovce editoru se na vyseparované hodnoty můžeme podívat, případně upravit a začít navařovat.



Obrázek 5.2: Grafické znázornění konečného automatu

5.1.2 Konečný automat pro soubory s příponou h

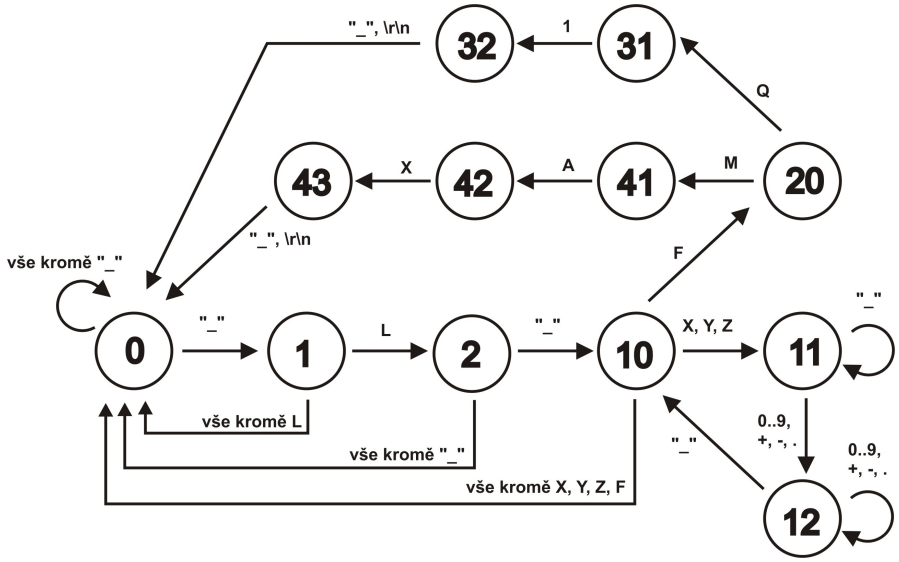
Nyní si ukážeme ukázku souboru s příponou h. Je velmi podobný souboru s příponou txt, protože i tento soubor byl vygenerován na základě programu typu CAD/CAM.

I tento soubor obsahuje hlavičku, ovšem v jiném formátu, a nelze tak jednoduše poznat začátek a konec hlavičky jako v předchozím případě. Proto budeme hlavičku brát za nepodstatné řádky, dokud nenarazíme na řádek, kdy se po číslu označující řádek, vyskytne písmeno „L“. „L“ musí být z obou stran obklopené mezerou, což automat kontroluje. Opět se načtou předchozí hodnoty souřadnic do struktury, pokud máme už alespoň jeden řádek načtený. Tyto souřadnice se buď přepíší novými hodnotami, nebo zůstanou.

V dalším kroku začneme načítat symboly souřadnic, a jejich hodnoty se uloží do struktury. Až narazíme na znak ukončení řádku, načteme do struktury parametry, které zadal uživatel na obrazovce importu CNC programů. Jedná se o obdobný postup, jako u souborů s hlavičkou txt.

U souborů s příponou txt se o statusu rozhodovalo na základě G00 nebo G01, zde automat rozhoduje podle toho, zda je za souřadnicemi FQ1 (status 1) nebo FMAX (status 2).

Tímto způsobem načteme zbylé řádky. Úmyslně zde nepopisuji konečný automat tak podrobně, jako u předchozího případu, protože se jedná o velmi podobný automat s malými úpravami.



Obrázek 5.3: Grafické znázornění konečného automatu

Ukázkový soubor

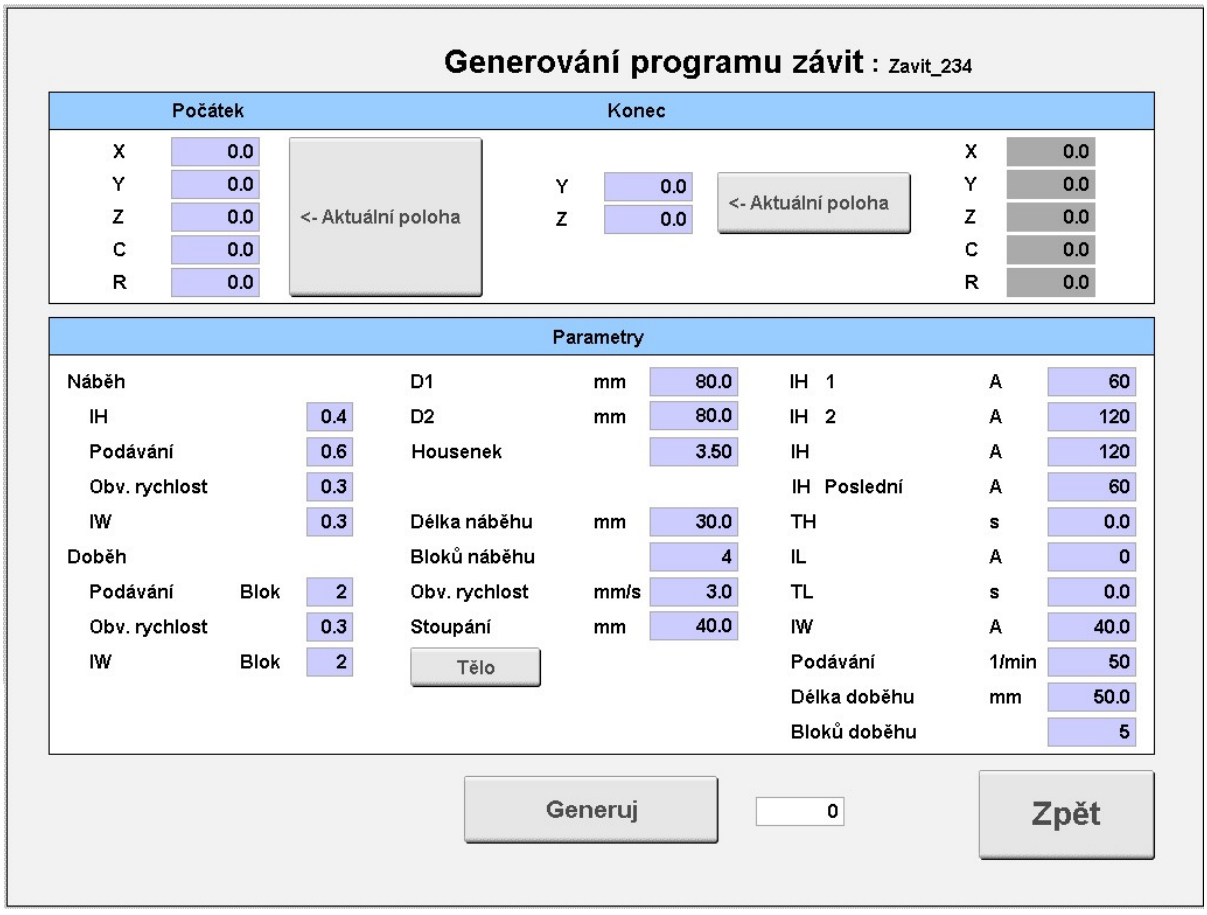
```
0 BEGIN PGM PRAVITKO MM
1  ; ( DATUM ZPRACOVANI: 09.06.2009)
2  ; ( CAS ZPRACOVANI: 12:40:21)
3  ; (KSK CESKA TREBOVA)
4 BLK FORM 0.1 Z X-10 Y-10 Z-10
5 BLK FORM 0.2   X+10 Y+10 Z+10
6 ; OZNACENI:  01141604
7  ;( )
```

```
8  TOOL CALL 1 Z S2000
9  M3
10  FN 0: Q1 =+500 ; POSUV XY
11  FN 0: Q2 =+100 ; POSUV Z
12 L X198.938 Y7. FMAX M8
13 L Z150. FMAX
14 L Z4.8 FMAX
15 L Z-.2 FQ1
16 L Y12.
17 L X196.75 Y8.25
atd.
```

5.2 Generování programu závit

Závit se navařuje na dílech tvaru válec. Typickým příkladem takového navařování je šnek na obrázku 5.5.

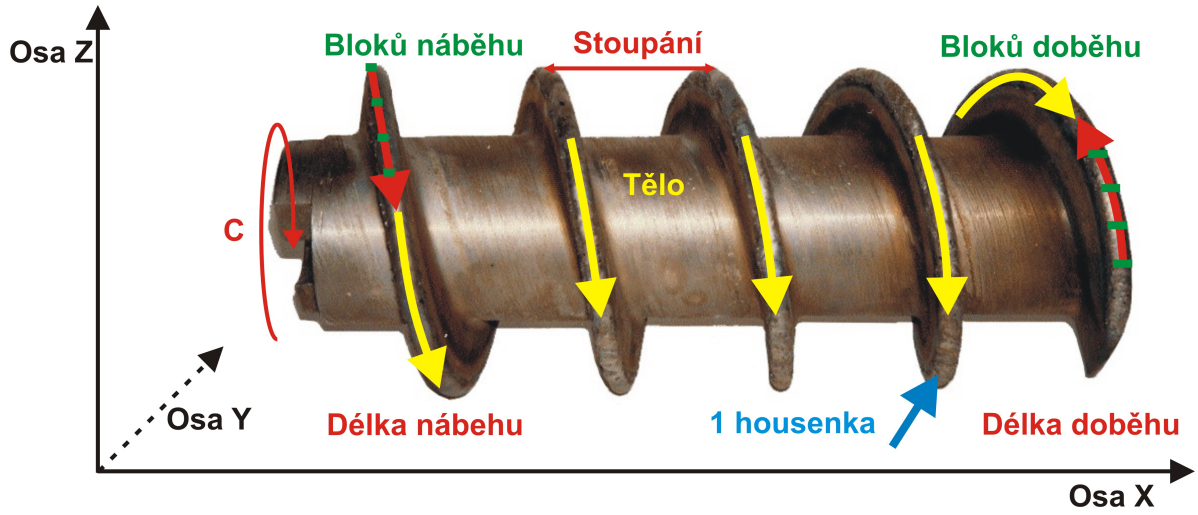
Grafické ztvárnění obrazovky (obr. 5.4) je podobné jako u generování obdélníku nebo válce. Uživatel zadá několik parametrů a pokud jsou všechny hodnoty v pořádku, tzn. nesvítí červeně, může spustit generování programu. Generování probíhá maximálně v řádu desítek vteřin.



Obrázek 5.4: Grafické rozhraní generování programu závit

5.2.1 Princip generování

Každý svařovací program závitu začíná nájezdem, poté pokračuje náběhem, tělem, doběhem a dojezdem (ob. 5.5).



Obrázek 5.5: Příklad závitu – šnek

Nájezd je najetí hořáku a polohovadla na souřadnice, které si obsluha stroje navolila. Opakem nájezdu je dojezd. Je to tedy vyjetí hořáku do uživatelem zadaných souřadnic.

Nájezd na souřadnice je dán následovně:

$$X = počátekX\,,$$
$$Y = počátekY\,,$$
$$Z = počátekZ\,,$$
$$C = počátekC\,,$$
$$R = počátekR.$$

Souřadnice Y a R se v průběhu navařování nemění. Souřadnice Z popojede až na konci celého procesu, v dojezdu. Výpočty budeme provádět pouze pro souřadnice X a C, dále pak výpočty proudů a podávání.

V dalších krocích proběhne samotné navařování součásti.

Po najetí hořáku na souřadnice začneme počítat hodnoty náběhu. Náběh se skládá z bloků náběhu a v každém bloku musíme hodnoty přepočítávat a ukládat.

Důležitým parametrem je otáčení polohovadla C, které vypočítáme následovně:

$$C = počátekC + blok \times stupeňC\,,$$

$$stupeňC = \frac{360 \times náběhDélka}{přepona \times náběhBloků},$$

$$přepona = \sqrt{obvod^2 + stoupání^2},$$

kde *blok* je aktuální blok, ve kterém se nacházíme, *obvod* je obvod navařovaného dílce a *stoupání* je vzdálenost mezi dvěma body na ose X.

Souřadnice osy X se musí také v každém bloku přepočítávat, aby hořák plynule popojížděl.

$$X = počátekX + (úhel \times úsek \times blok),$$

kde *blok* je aktuální blok, ve kterém se nacházíme, *úsek* je délka jednoho bloku (délka náběhu/bloků náběhu) a *úhel* je úhel závitu.

Intenzita navařování je dána mimo jiné proudem. Proud pozvolna stoupá do určité hodnoty dle vzorce:

$$IH = pomProud \times blok + proud1 \times náběhProud,$$

$$pomProud = \frac{proud1 - proud1 \times náběhProud}{náběhBloků},$$

kde *blok* je aktuální blok, ve kterém se nacházíme, *proud1* je proud zadaný uživatelem v ampérech a *náběhProud* je koeficient v době náběhu.

Nyní něco málo k vysvětlení. Nastavíme-li koeficient proudu v době náběhu na 0,8, *proud1* 100 A a máme dva bloky náběhu, pak během těchto dvou bloků musí proud vystoupat na hodnotu 100 A z hodnoty 80 A. K hodnotě 80 A přijdeme jednoduchým způsobem. Koeficient 0,8 znamená 80% z hodnoty 100 A, což je právě našich 80 A. V prvním bloku náběhu musí proud vystoupat z 80 A na 90 A a v druhém bloku z 90 A na 100 A. V obou blocích se musí měnit i souřadnice otáčení polohovadla a osy X, aby se nenavařovalo pořád na stejném místě. K navařování nestačí pouhý proud, ale také přídavný materiál, například prášek, nebo drát. Podávání prášku, drátu a obvodové rychlosti se počítá obdobným způsobem jako proud IH.

Pokud máme nastavenou pulzaci pomocí parametrů IL (hodnota nízkého proudu), TL (čas nízkého proudu) a TH (čas vysokého proudu), proud bude skokově klesat z hodnoty IH na hodnotu IL, vždy po dobu buď vysokého, nebo nízkého proudu.

Po náběhu pokračuje tělo, ve kterém se již parametry příliš nemění. Hodnota proudu IH zůstane taková, jaká byla v posledním bloku náběhu, tedy 100 A. Totéž platí pro podávání prášku nebo drátu a obvodové rychlosti. Pouze otáčení stolu polohovadla a souřadnice osy X se přepočítává. V každé housence, což je jedno otočení závitu, se souřadnice X posune o velikost stoupání a stůl se pootočí o 360°.

Na tělo navazuje doběh, což je opačný proces náběhu, s tím rozdílem, že se stůl polohovadla začne otáčet obráceně a souřadnice osy X také změní směr.

$$C = předchozíC - 360 \times \frac{úsek}{obvod},$$

kde *úsek* je délka jednoho bloku doběhu a *obvod* je obvod navařované součásti.

$$X = předchozíX - úhel \times úsek,$$

kde *úhel* je úhel závitu.

Proud a podávání prášku, případně drátu začne také klesat.

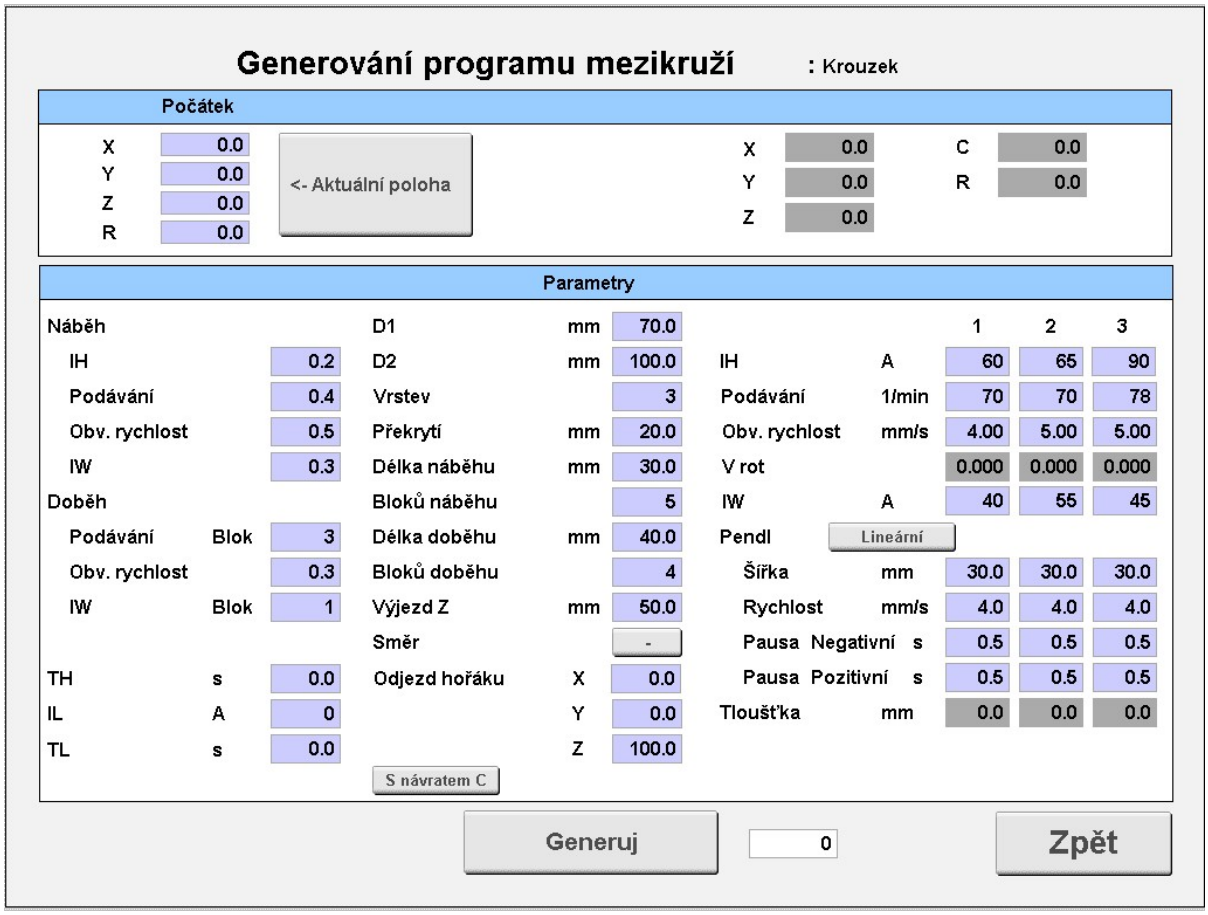
Poslední fází procesu je dojezd, kdy už nenavařujeme, pouze se hořákem odjede na požadované souřadnice a stůl polohovadla se otočí na úhel zadaný uživatelem.

5.3 Generování programu mezikruží

Poslední novou metodou, jak vygenerovat program, je metoda generování mezikruží. Příklad mezikruží si můžete prohlédnout na obrázku 5.6.

Při návrhu obrazovky jsme opět použili uživatelům známé prostředí, jako mohou vidět u generování programu závit, obdélník či válec. Prohlédnout si ho můžete na obrázku 5.7. I zde zadáváme poměrně velký počet parametrů, pomocí kterých vypočítáme, jak se bude pohybovat a chovat hořák a polohovadlo. Všechny parametry mají přednastavené hodnoty, které můžeme měnit. Je určený i rozsah hodnot, jakých může parametr nabývat. Pokud je některý z parametrů rozsvícen červeně, znamená to, že takto zadaná hodnota být nemůže a nelze spustit generování.

Obrázek 5.6: Ukázka mezikruží – sedla



Obrázek 5.7: Grafické rozhraní generování programu mezikruží

5.3.1 Princip generování

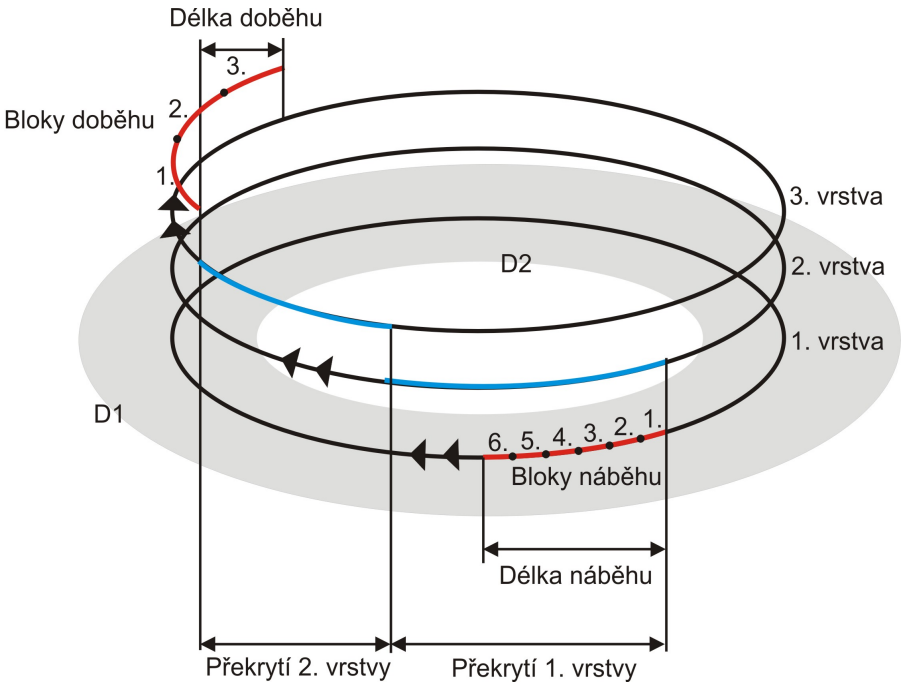
Na obrázku 5.6 je typický příklad mezikruží, tedy sedlo. Obrázek 5.8 ilustruje postup návaru. Každé mezikruží má dva průměry – vnější průměr D1 a vnitřní průměr D2. Mezikruží se navařuje v jedné, ve dvou nebo ve třech vrstvách nad sebou.

První fází je nájezd, kdy pouze uložíme uživatelem zadané souřadnice.

$$X = počátekX$$
$$Y = počátekY$$
$$Z = počátekZ$$
$$R = počátekR$$

Souřadnice X, Y a R se po celou dobu navařování nemění a souřadnice Z se mění v překrytí (viz níže). Po skončení navařování se osy X, Y a Z pouze posunou na uživatelem zvolené souřadnice (viz níže).

Pokud navařujeme všechny tři vrstvy, první vrstva se skládá z náběhu a těla. Náběh je rozdělen na bloky. Délku i počet bloků si může uživatel nastavit. Pro tuto část návaru si uživatel dále nastaví proud, podávání prášku a obvodovou rychlost.



Obrázek 5.8: Nákres mezikruží

V každém bloku náběhu vypočítáme osu C (poloha stolu polohovadla ve stupních) podle vzorce:

$$C = blok \times 360 \times \frac{úsek}{obvod},$$

kde *C* je osa, *blok* je aktuální blok, ve kterém se nacházíme, *úsek* je délka jednoho bloku (délka náběhu / bloků náběhu) a *obvod* je středový obvod (Π × (D1 + D2)/2).

Hodnotu vysokého proudu vypočítáme dle vzorce:

$$IH = pomProud \times blok + Proud1 \times ProudNáběh,$$

$$pomProud = \frac{Proud1 - Proud1 \times ProudNáběh}{blokůNáběhu},$$

kde *Proud1* je nastavený proud pro první vrstvu, *ProudNáběh* je koeficient proudu v době náběhu a *blokůNáběhu* je počet bloků náběhu.

Obdobně jako u závitu, nastavíme-li *ProudNáběh* na 0,5 (tzn. 50%), *Proud1* na 100 A a máme 5 bloků náběhu, v prvním bloku začne proud na 50% hodnotě proudu první vrstvy, což je 50 A. Během zbylých bloků musí proud postupně a rovnoměrně vystoupat, aby ke konci posledního bloku byl proud 100 A. V našem případě to znamená, že v druhém bloku bude proud stoupat ze 60 A na 70A, ve třetím ze 70 A na 80 A, ve čtvrtém bloku z 80 A na 90 A a v pátém bloku z 90 A na 100 A.

Hodnotu podávání a obvodové rychlosti vypočteme obdobným způsobem jako proud, ale s parametry pro podávání a obvodovou rychlost. V případě, že stroj umožňuje i proud drátem, vypočteme tímto způsobem i IW.

Parametry pendlu zůstávají po celou dobu náběhu stejné.

Další fází je tělo první vrstvy. V těle pokračujeme s parametry, které jsme vypočítali jako koncové parametry posledního bloku náběhu. Parametry pendlu zůstávají v první vrstvě také stejné. Pouze osa C, tedy poloha stolu polohovadla, se otočí na 360°.

Po těle nastává překrytí. S osou C jsme na 360°, což znamená, že jsme se otočili na polohu, kde jsme začali. Překrytí tedy nastává v místě, kdy začal náběh. Délka překrytí však může být jiná, než délka náběhu.

Nastavíme osu Z, což je pohyb hořáku směrem nahoru. Polohu stolu polohovadla vypočítáme dle vzorce:

$$C = předchozíHodnotaC + 360 \times \frac{délkaPřekrytí}{obvod}$$

Hodnoty ostatních parametrů zůstanou stejné jako v těle první vrstvy.

Další fáze je tělo druhé vrstvy. S osou C se opět přetočíme o 360°. Zbylé parametry necháme, jak je uživatel zadal pro druhou vrstvu.

Nyní se proces opakuje. Prošli jsme tělem druhé vrstvy a následuje další překrytí. Překrytí začíná v místě, kde skončilo předchozí překrytí. Po tomto posledním překrytí následuje třetí a zároveň poslední tělo návaru. Tělo návaru opět končí v místě, kde skončilo předchozí překrytí. Jediné, co se mění, jsou parametry, které uživatel zadal pro třetí vrstvu.

Tak jako na začátku navařování byl náběh, na konci máme doběh. Doběh začíná v místě, kde skončilo poslední překrytí a je dlouhé podle nastavení uživatele. I u doběhu nastavujeme počet bloků doběhu.

Proud bude v doběhu postupně klesat z hodnoty zadané ve třetí vrstvě na hodnotu *KTRANSFERMIN*, což je konstanta, kterou si uživatel zvolit nemůže. Pokud tedy máme například doběh se třemi bloky, hodnotu proudu ve třetí vrstvě máme nastavenou na 150 A, musíme postupně během tří bloků klesat ze 150 A na konstantu *KTRANSFERMIN* podle vzorce:

$$IH = Proud3 - pomProud \times blok,$$

$$pomProud = \frac{Proud3 - KTRANSFERMIN}{bloků Doběhu},$$

kde *Proud3* je nastavený proud ve třetí vrstvě, *blok* je blok, ve kterém se aktuálně nacházíme a *blokůDoběhu* je počet bloků doběhu.

Dále musíme spočítat podávání a proud drátem, pokud jej stroj podporuje. Obojí končí v nule. Uživatel si jen nastaví, v jakém bloku doběhu se tak stane. Pokud nastaví hned první blok, není co počítat a uložíme nulu hned do prvního bloku dojezdu a všechny ostatní budou též nulové. V opačném případě, když nastaví jiné bloky, musíme použít vzorec:

$$Podávání = Podávání3 - PomPodávání \times blok\,,$$

$$PomPodávání = \frac{Podávání3}{podaváníDoběh},$$

kde *Podávání3* je podávání nastavené ve třetí vrstvě.

Obdobně jako podávání spočítáme i proud drátem.

Výpočet obvodové rychlosti bude podobný jako při náběhu, pouze místo přičítání budeme odečítat.

$$ObvRychlost = ObvRychlost3 - PomRychlost \times blok\,,$$

$$PomRychlost = \frac{ObvRychlost3 - ObvRychlost3 \times ObvRychlostNabeh}{blokůDoběhu},$$

kde *ObvRychlost3* je obvodová rychlost nastavená ve třetí vrstvě.

Pro výpočet polohy stolu polohovadla v jednotlivých blocích použijeme vzorec:

$$C = předchocíBlokC + 360 \times \frac{úsek}{obvod},$$

kde *úsek* je délka jednoho bloku (délka doběhu / bloků doběhu) a *obvod* je středový obvod (Π × (D1 + D2)/2).

Na začátku celého procesu byl nájezd, nyní je třeba ukončit proces odjezdem. Odjezd si uživatel nastaví sám pomocí os X, Y a Z. Hodnoty těchto souřadnic pouze přičteme k předchozím hodnotám. Výsledné souřadnice jsou tedy relativní, nikoli absolutní.

Stůl polohovadla mohl skončit v jakékoli poloze, proto ho natočíme na celé otáčky dle vzorce:

$$otáček = \frac{předchozíC}{360,}$$

$$otáček = round\,(otáček),$$

$$C = otáček \times 360,$$

To je celý proces naváření kroužků. Způsob, jakým počítáme jednotlivé parametry se snaží imitovat ruční naváření kroužku.

Jak bylo zmíněno, kroužky můžeme navařovat v jedné, ve dvou nebo ve třech vrstvách. Předchozí rozbor byl ukázán pro tři vrstvy návaru. Pokud zvolíme vrstev méně, proces bude kratší, ale postup navařování stejný.

6 Implementace, testování a provoz software

6.1 Implementace

Při programování v Automation Studiu jsem se potýkala s mírnými odchylkami s programovacím jazykem C. Práce se soubory, se kterými jsem velmi často během modernizace pracovala, je odlišná od klasického C. B+R mají vlastní knihovny pro práci se soubory.

Veškeré grafické komponenty se musí programovat. Neexistují zde předprogramované komponenty jako například ve vývojových programech pro programovací jazyky Java či C++.
Z tohoto důvodu jsme například tlačítka pro posun mezi řádky (Up, Down, Page Up, Page Down, Home a End) museli naprogramovat ručně, ačkoli to není u vývojových programů příliš běžné.

6.2 Testování

Testování všech částí nově vytvořeného nebo zmodernizovaného software probíhá nejdříve v rámci dotykové obrazovky, bez svařovacího automatu. Každá část se testuje samostatně a později jako celek.

Důležité je vyladit veškeré zviditelňování, zneviditelňování a zamykání tlačítek a vstupů, aby při určitých činnostech nemohlo dojít k problémům. Například ke stisku tlačítka „Svařování“ v době, když ještě není načtený celý program.

Velký důraz klademe na data v programech. Nesmí se stát, že při některé operaci část dat ztratíme, přepíšeme nebo použijeme jiná. Pokud bychom například při editaci programu uložili data jinak, než vidí obsluha stroje, při nejmenším by to mohlo při spuštění programu poškodit navařovanou součást, v horším případě hořák nebo ramena stroje.

Po důkladném otestování software na testovacím panelu můžeme software nahrát do panelu navařovacího automatu, který je zatím používán pouze ve firmě. Obsluha je proškolena a seznámena s novinkami. V této fázi testování musí být svářeči obezřetní a při každém náznaku možné chyby si píší poznámky. V případě, že se jedná o závažnější chybu, je potřeba problém vyřešit okamžitě.

Pokud vyladíme všechny chyby, můžeme software použít i mimo firmu. Bohužel se nám stalo, že software zklamal i za běžného provozu. Naštěstí nikdy nešlo o závažné chyby, spíše o detaily, které jsme snadno vyřešili.

V dnešní době je zmodernizovaný software instalován na navařovacích automatech v následujících podnicích:

- KSK Česká Třebová – navařovací automat PPC 250 PTM
- MSA Dolní Benešov – navařovací automat PPC 250 HDP
- Armatury Group Kravaře – navařovací automat PPC 250 PTM
- LDM Česká Třebová – navařovací automat PPC 250 PTM
- UNIOR Kovaška industrija d.d., Slovinsko – navařovací automat PPC 250 PTM
- Neckmolde, Portugalsko – navařovací automat PPC 250 GMR

Jak je vidět z názvů automatů, jedná se o řadu PPC 250, ale o různé typy. Všechny typy automatů mají společný základ, liší se pouze v počtu os a možnostech využití.

- Typ GMR je využíván na sklářské formy a má všech šest os pro pohyb ramen a polohovadla.

- Typ PTM je největší ze všech typů, ale chybí mu osa N pro naklopení hořáku. Má tedy pět os. Využívá se hlavně na navařování průmyslových armatur.
- Typ HDP je menší a jednodušší typ automatu. Má pouze osy X, Y, Z a C. Využívá se pro navařování průmyslových armatur. Automat umožňuje navařovat nejen práškem, ale i horkým drátem, který se dá použít na některé typy materiálu navařovaných součástí (klíny a sedla šoupátek). Drát je také výrazně levnější než prášek.

6.3 Testování manažeru programů

6.3.1 Ladění

Co se týče chyb manažeru, vyskytly se pouze chyby se špatným odkrýváním tlačítek, což zapříčinilo zablokování, nebo naopak odblokování funkcí. V manažeru se tak stalo, že nešlo kopírovat soubory do prázdného adresáře.

Chyba, která sice nevadila funkčnosti, ale spíše vzhledu, se vyskytla při dotykovém rolování. Pokud jsme měli jedno okno prázdné a v druhém vyhledávali dotykem, v prázdném okně začaly chaoticky vyskakovat prázdné řádky. Problém byl vyřešen pomocí statusů neviditelnosti. Nejdříve byly řádky znevidtelňovány pouze pomocí barev, nikoli statusů. Po přeprogramování problém zmizel.

Závažnější problém se stal při přejmenování souborů. Pokud uživatel přejmenuje název souboru, který pak začíná mezerou, B+R knihovny nedokáží soubor identifikovat a úplně znemožní přístup k tomuto souboru. Dokonce ani vzdáleně přes FTP se nepodařilo soubor upravit ani odstranit. Proto bylo nezbytné uživatelům zakázat název souboru začínající bílým znakem. V případě, že uživatel zadá takový název, bude upozorněn hlášením.

6.3.2 Výsledky testování

Při testování manažeru programů jsme se zaměřili zejména na čas strávený při načítání adresářů, hledání souborů a využití více funkcí najednou. Testování probíhá vždy na adresářích s maximálním možným počtem souborů, což je 500.

Některé výsledky nebyly podle očekávání. Ačkoli jsme přeprogramovali načítání adresářů a přidali možnost načítat adresář v levé i v pravé části manažeru zároveň, zklamalo hardwarové vybavení panelu. Procesor panelu má dostatečný výkon na řízení technologických procesů, pro které je procesor primárně určen. Nicméně pro kancelářské aplikace, jako například náš manažer programů, kdy je zapotřebí větší paměti, je zcela nevyhovující. Tato nemilá skutečnost zapříčinila téměř stoprocentní využití procesoru při načítání velkého adresáře, proto při současném načítání levé i pravé části manažeru programů jsme nedosáhli téměř žádného urychlení. Na výsledky testování se můžeme podívat do tabulky 6.1, na první řádek.

Útěchou za nevydařené načítání může být nalezení požadovaného souboru, kdy rozdíl je obrovský. Dříve, pokud jsme se chtěli dostat na poslední soubor v adresáři, bylo nutné se po jednom souboru pomocí šipek dostat až na konec adresáře. Nyní stačí pouze jeden dotyk. Na druhém řádku v tabulce 6.1 máme výsledky.

Při vyhledávání souboru, který se nachází zhruba v polovině adresáře, máme také nemalé zrychlení a komfort při hledání. Nemusíme rolovat po jednom, ale skáčeme po stránkách a poté buď po jednom souboru nebo dotykem na příslušný soubor. Na třetím řádku tabulky 6.1 jsou naměřené výsledky.

	Operace	Počet souborů	Původní stav min:s:ms	Nynější stav min:s:ms
1	Načtení adresářů v levé i v pravé části zároveň	500	05:40:00	05:26:00
2	Přesunutí se v adresáři z prvního souboru na poslední	500	01:48:00	00:00:35
3	Přesun v adresáři z prvního souboru na soubor v půli adresáře	500	00:58:00	00:09:00

Tabulka 6.1: Časová efektivita

6.4 Testování editoru programů

6.4.1 Ladění

Odstraňování chyb editoru programů bylo komplikovanější, protože opravou jedné chyby, nebo modernizací nějaké funkce, se objevil nový problém.

Ve většině případů stačilo zablokovat uživatelům tlačítka tak, aby při určité operaci nebyli schopni tuto operaci narušit jinou operací.

Uživatelé v editoru dostali možnost editovat jednotlivé sloupce pouhým dotykem, při testování stroje v provozu se však zjistilo, že pokud změníme více sloupců naráz, data se neuloží. Bylo to právě kvůli špatnému užívání editoru, kdy měl svářeč zapnutou funkci upravování dat v jednom řádku a zároveň upravoval pomocí dotyku. Tyto dvě operace se navzájem rušily, proto jsme je vzájemně vyloučili.

Objevil se také problém s otevíráním některých svařovacích programů. Vzhledem k tomu, že některé programy šly otevřít a některé ne, logicky jsme hledali chybu v datech svařovacích programů. Později se ukázalo, že zdrojový kód, který léta fungoval a otevíral soubory, obsahuje chybu, protože B+R nepatrně pozměnilo knihovny pro práci se soubory.

Již zmíněným problémem se staly názvy svařovacích programů začínajících bílým znakem. Knihovny tyto soubory neumí zpracovat, proto jsme uživateli zablokovali možnost takto pojmenovat svařovací program. Pokud bychom tento problém neodstranili včas, mohlo dojít k úplnému zničení špatně pojmenovaných programů.

Při výběru svařovacího programu, který má být v editoru otevřen, musíme nejprve vybrat adresář. Pokud obsahuje adresář větší množství souborů, načtení trvá déle. Během této doby jsme měli viditelné tlačítko „Ano“ a „Storno“. Tlačítko „Ano“ slouží pro otevření programu a pokud jsme ho zmáčkli v okamžik, kdy se adresář teprve načítal, celý panel se zablokoval a vyžadoval restart. Za běžného provozu by to znamenalo zapnout a vypnout stroj. Tlačítko „Ano“ jsme proto při načítání adresáře zablokovali. Tlačítko „Storno“ také nefungovalo správně, nicméně by nezpůsobilo kolaps. Při načítání adresáře zmáčknutím tlačítka „Storno“ je logické, aby se načítání ukončilo, ale abychom zůstali ve výběrové obrazovce. Bohužel tomu tak nebylo a tlačítko způsobilo pouhý návrat do editační obrazovky a když jsme chtěli znovu otevřít svařovací program, automaticky se začal načítat adresář, jehož načítání jsme zrušili. Bylo tedy nutné naprogramovat několik kontrol a hlídat, kdy je tlačítko „Storno“ použité jako rušení načítání a kdy jako návrat do editační obrazovky.

6.4.2 Výsledky testování

Testování přineslo předchozí chyby, které byly následně odstraněny. Novější prostředí mírně urychlilo a hlavně ulehčilo práci s editorem. Svařovací programy jsou snáze editovatelné. V jednotlivých řádcích editačního okna navíc probíhají kontroly zadaných hodnot, takže svářeči již nemají ve svých programech nesmyslné hodnoty.

Při editování většího počtu hodnot dosahujeme urychlení. Výsledky měření jsou shrnuty v tabulce 6.2.

Pro názornost ukazujeme na prvním až čtvrtém řádku tabulky výsledky měření v konkrétních blocích programu, aby bylo vidět, jak důležitým prvkem jsou tlačítka pro pohyb mezi řádky (Home, End, Page Up, Page Down). Editování probíhalo vždy v rámci jednoho bloku (řádku) programu.

Na pátém až osmém řádku tabulky jsou dosažené výsledky při editování různých hodnot v různých řádcích.

	Operace	**Počet bloků (řádků) programu**	**Původní stav min:s:ms**	**Nynější stav min:s:ms**
1	Editace tří hodnot v 1. bloku programu	22	00:10:53	00:08:92
2	Editace tří hodnot v 21. bloku programu	22	00:21:14	00:09:77
3	Editace tří hodnot v 50. bloku programu	84	00:33:41	00:12:47
4	Editace tří hodnot v posledním bloku programu	137	00:48:96	00:10:71
5	Editace vždy jedné hodnoty v pěti blocích programu v rozmezí 1. - 20. blok	137	00:35:76	00:12:64
6	Editace vždy tří hodnot v pěti blocích programu v rozmezí 1. - 20. blok	137	00:51:34	00:35:52
7	Editace vždy jedné hodnoty v pěti blocích programu v rozmezí 61. - 80. blok	137	01:02:34	00:16:59
8	Editace vždy tří hodnot v pěti blocích programu v rozmezí 61. - 80. blok	137	01:28:34	00:42:99

Tabulka 6.2: Časová efektivita

6.4.3 Generování programu závit

Generování programů a nejen závitu, je obecně problémové. Snahou vygenerovaného programu je, aby program simuloval ruční navařování, což bývá obtížné.

Generování závitu bylo zatím testováno pouze v teoretické rovině. Kontrolovali jsme, zda vygenerované hodnoty souhlasí s trajektorií, po které se hořák a polohovadlo pohybuje, zda hodnoty proudů a podávání odpovídají reálným hodnotám, které se při navařování používají.

Shodli jsme se, že vygenerované hodnoty by mohly být použitelné, takže v nejbližší době budou provedeny první reálné testy. Nemůžeme však očekávat, že hned první návar bude proveden dokonale.

6.4.4 Generování programu mezikruží

Generování mezikruží prošlo dlouhým vývojem a implementačními změnami. První testování proběhlo opět v teoretické rovině. Zpočátku jsme se domnívali, že vygenerované hodnoty budou správné. Při testování v reálu se ukázalo, že návar ve fázi mezi náběhem a tělem a mezi tělem a doběhem má ne úplně správné rozložení návaru. Bylo tedy potřeba mírně upravit výpočty.

Při druhém testování s novými propočty byly přechody mezi náběhem a tělem a mezi tělem a doběhem již správné, nicméně nyní byly ne zcela ideální přechody mezi tělem vrstvy a překrytím. Drobné úpravy tento problém odstranily a nyní je generování mezikruží téměř dokonalé.

Díky navařování kroužků tímto způsobem nyní svářeč nemusí po celou dobu navařování být u stoje a může se tak věnovat i jiné práci.

6.4.5 Import CNC

Testování importu CNC bylo bezproblémové. Soubor, ze kterého jsou data importována, si můžeme prohlédnout a porovnat s hodnotami, které nám konečný automat rozebral.

Napoprvé automat nečinil přesně to, co jsme očekávali. Občas se nám hodnoty ukládaly k jiným souřadnicím a podobně.

Jediným problémem bylo určit hlavičku CNC souboru, která nebyla příliš rozeznatelná od počátku dat.

Veškeré problémy však byly úspěšně odhaleny a importování je zcela funkční a používané.

Ačkoli se tato část softwaru může zdát zbytečná, opak je pravdou. Pokud by nešlo programy importovat, musel by někdo celý program ručně přepsat do stroje, což by znamenalo několikahodinovou přípravu, možnost zavlečení chyb a s tím vynaložené finanční náklady.

7 Závěr

Mojí snahou bylo zmodernizovat uživatelské rozhraní dotykového panelu, využívaného v průmyslové výrobě na plazmových navařovacích automatech tak, aby uživatelé měli snadnější práci s tímto rozhraním. Jednalo se zejména o manažer a editor svařovacích programů. Některé změny byly provedeny i mimo tyto dvě části, nicméně nelze úplně rozlišit co je, a co není součástí manažeru a editoru programů. Software je celek a tak se k němu musíme chovat.

Software se vyvíjel postupně a tudíž i instalace na automaty nebyla jednorázová. Zpočátku, když jsme v naší firmě instalovali první β-verze, ohlasy od svářečů nebyly nikterak povzbuzující. Objevilo se značné množství chyb, které nebyly odhaleny při testování mimo provoz. S postupem času se však tyto chyby odstranily a konečně jsme se dočkali dobrých výsledků.

V této době software pracuje téměř bezproblémově. Občas se nějaký problém s uživatelským rozhraním vyskytne. Naštěstí jsou to menší problémy, které příliš nevadí funkčnosti, a nemusíme se tedy bát, že nám budou volat například z Portugalska, abychom přijeli a chybu napravili.

Nyní tedy mohu s klidem konstatovat, že dosavadní modernizace byla úspěšná, nikoli konečná. Do budoucna se počítá s dalšími modernizacemi. Přibudou další možnosti generování programů, větší kontroly zadávaných hodnot do editačního okna a kontroly souborů určených pro import CNC programů. Nemilou funkcí pro svářeče bude evidence času, která má za úkol kontrolovat, jak intenzivně se na stroji pracuje. V neposlední řadě bude možnost přesného navážení přídavného materiálu pomocí váhy na základě vybraného svařovacího programu. Tyto a mnohé další změny software ještě čeká.

Touto prací jsem získala mnoho cenných zkušeností, jak s lidmi, tak se stroji. Ačkoli mým prvotním záměrem nebylo pochopit, co a jak se navařuje, nové poznatky mě neminuly. Velkým přínosem pro mě bylo poznání vývojového prostředí B+R Automation Studio.

Literatura

[1] Power Panel 300/400 : User´s manual [online]. Version 2.30. [Rakousko] : Bernecker + Rainer Industrie-Elektronik Ges.m.b.H., July 2010 [cit. 2011-01-22]. Dostupné z WWW: <http://www.br-automation.com/downloads_br_productcatalogue/BRP4440000000000000128690/MAPP300.400-ENG%20V2_30.pdf>

[2] KRŠKA, Zdeněk; ŠEVČÍK, Stanislav; KUČERA, Ladislav. KSK, s.r.o. [online]. Česká Třebová : 2010 [cit. 2011-01-22]. Svařovací stroje. Dostupné z WWW: <http://www.kskct.cz/web/web.php?jazyk=cz&odkaz=stroje>

[3] B&R - Products [online].Bernecker + Rainer Industrie-Elektronik Ges.m.b.H., c2011 [cit. 2011-01-22]. B&R - Perfection in Automation. Dostupné z WWW: <http://www.br-automation.com/cps/rde/xchg/br-productcatalogue/hs.xsl/products_ENG_HTML.htm>

[4] ŠEVČÍK, Stanislav. Plazmové navařování práškem a horkým drátem. MM Průmyslové spektrum. 2000, č. 10, s. 2.

[5] B&R Help Explorer. Version 3.0.80. [Rakousko] : Bernecker + Rainer Industrie-Elektronik Ges.m.b.H. July 2010. Doustupné se zakoupením software.

[6] KERNIGHAN, Brian; RITCHIE, Dennis M; ŠŤÁVA, Zbyněk. *Programovací jazyk C*. Vyd. 1. Brno : Computer Press, 2006. 286 s. ISBN 80-251-0897-x.

[7] BÍLEK, Petr. *Sally* [online]. 2003-08-29, 2008-10-02 [cit. 2011-03-15]. Programování v jazyku C/C++. Dostupné z WWW: <http://www.sallyx.org/sally/c/>.

Seznam příloh

Příloha 1. CD obsahující:

- Bakalářskou práci ve formátu PDF
- Vybrané zdrojové kódy
- Videonahrávka o generování mezikruží a jeho navařování