

Příprava laboratoře a elektronických opor
předmětu Programovatelné automaty

Arrangement of laboratories and electronical
supports for subject Programmable controllers

Václav Liška

Diplomová práce
2005



Univerzita Tomáše Bati ve Zlíně
Fakulta technologická

Vložit oficiální zadání diplomové/bakalářské práce

Děkuji svému vedoucímu, panu Ing. Tomáši Sysalovi, Ph.D., za vstřícnost a pomoc při řešení této diplomové práce. Dále děkuji pánům Petru Dvořákovi a Ing. Aloisi Mynaříkovi za neocenitelnou pomoc při řešení dílčích technických problémů.

Prohlašuji, že jsem na celé diplomové práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně, 09. 06. 2005

.....

jméno diplomanta

Abstrakt

Náplní této diplomové práce je návrh a zpracování laboratorních úloh pro předmět Programovatelné automaty. Pro tento účel byl zvolen reálný model fontány, pro jeho řízení byl použit programovatelný automat SAIA PCD2.

V práci jsou podrobně popsány model fontány, ovládací skříň, programovatelný automat SAIA PCD2 a dále vývojové prostředky pro programovatelný automat a pro vizualizaci procesu na PC. Pro cvičení v rámci laboratoří předmětu bylo navrženo několik praktických úloh s různou úrovní obtížnosti.

Jako elektronická opora předmětu Programovatelné automaty byla vytvořena webová aplikace dostupná na adrese www.plc.unas.cz. Popis její struktury, obsahu a prostředků použitých při jejím vývoji je rovněž součástí práce.

Summary

Main target of this thesis is to suggest and prepare laboratory practise for subject Programmable logic controllers. For this purpose the model of fountain was chosen. This model was controlled by PLC SAIA PCD2.

This thesis contains detailed description of the fountain model, control case, PLC SAIA PCD2, development means for PLC and for process visualisation on PC. In order to practice skills in laboratories there were suggested several exercises on different sophistication levels.

As an electronical support for subject Programmable controllers there had been created a web application available at www.plc.unas.cz. Description of it's structure, content and means used for its development is part of the thesis as well.

OBSAH

ÚVOD.....	6
1 PŘÍPRAVA LABORATOŘE	7
1.1 SCHÉMA ZAPOJENÍ	7
1.2 MODEL FONTÁNY	8
1.2.1 Stručný popis fontány.....	8
1.2.2 Čerpadla.....	9
1.3 OVLÁDACÍ SKŘÍŇ FONTÁNA (OSF)	10
1.3.1 Konstrukce.....	10
1.3.2 Regulace otáček pomocí PWM	11
1.3.3 Pulzně šířková modulace (PWM).....	13
1.4 PROGRAMOVATELNÝ AUTOMAT	15
1.4.1 Programovatelný automat obecně	15
1.4.2 Programovatelný automat SAIA PCD2.....	19
1.5 VÝVOJOVÉ PROSTŘEDÍ PG4	24
1.5.1 Knihovna projektů	24
1.5.2 Správce projektu	25
1.5.3 Function Plan – FUPLA (FBD).....	26
1.6 CONTROL WEB 2000.....	29
1.6.1 Virtuální přístroje	29
1.6.2 Vytváření programů.....	30
1.6.3 Napojení aplikace na reálné zařízení	32
1.6.4 Připojení k PLC Saia	34
1.7 ÚLOHY PRO LABORATORNÍ CVIČENÍ.....	36
1.7.1 Úloha č. 1.....	36
1.7.2 Úloha č. 2.....	38
1.7.3 Úloha č. 3.....	40
1.7.4 Úloha č. 4.....	42
2 ELEKTRONICKÁ OPORA PŘEDMĚTU	44
2.1 POUŽITÉ TECHNOLOGIE	44
2.2 APLIKACE PRO VÝVOJ.....	46
2.2.1 WWW-server.....	46
2.2.2 Jazyk PHP.....	46
2.2.3 HTML editor	46
2.3 STRUKTURA A ČLENĚNÍ STRÁNEK	47
2.4 OBSAH.....	49
ZÁVĚR.....	52
SEZNAM POUŽITÝCH ZDROJŮ	53
SEZNAM POUŽITÝCH SYMBOLŮ, ZKRATEK A TERMÍNŮ.....	54
SEZNAM OBRÁZKŮ.....	55
SEZNAM TABULEK	56
SEZNAM PŘÍLOH.....	57

ÚVOD

Průmyslová praxe dávno ukázala, že použití programovatelných automatů již zdaleka není jen při řešení logických úloh a nahrazování reléových řídicích obvodů. Díky své flexibilitě, spolehlivosti, širokým možnostem uplatnění a v neposlední řadě také ceně se prosazují ve stále nových oblastech automatizace. PLC snadno nahradí rozsáhlé reléové obvody či analogové PID regulátory. Běžně se můžeme setkat s použitím PLC na pozici adaptivního regulátoru. Velkou výhodou programovatelných automatů je mimo jiného i to, že se s nimi poměrně snadno pracuje. Vývoj nových aplikací není nijak složitý a obvykle netrvá příliš dlouho.

V rámci laboratorních cvičení předmětu Programovatelné automaty je v současné době pro demonstraci možností PLC využíváno vývojové prostředí Mosaic firmy TECO a.s. Kolín. Velkou výhodou programu Mosaic je, že umožňuje vytvářet a odlaďovat nové aplikace bez nutnosti připojení počítače k vlastnímu PLC. Pro pochopení základů psaní programů je tento přístup ideální. S rostoucí znalostí problematiky programovatelných automatů nicméně vzniká potřeba ověřit si dovednosti v praxi, při řízení skutečných zařízení.

K tomuto účelu byl vybrán reálný model fontány, který byl následně připojen k programovatelnému automatu. K řízení tohoto modelu byl zvolen pomocí programovatelný automat SAIA PCD2.

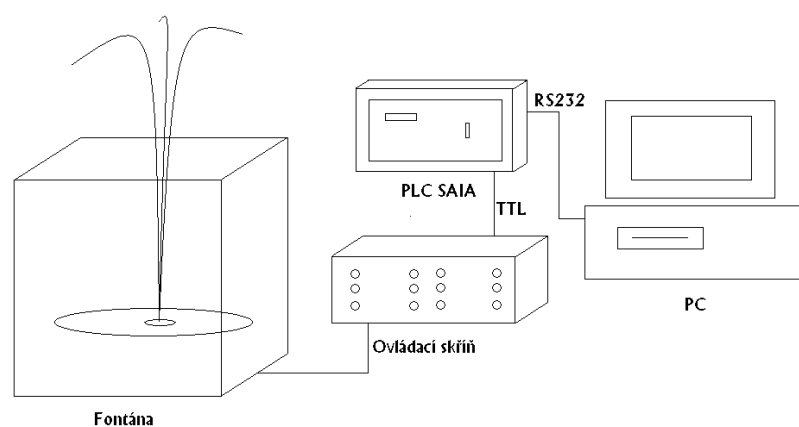
Samotný výběr modelu velmi pěkně ukazuje, jak široké uplatnění programovatelných automatů je – od řízení zařízení a strojů, s nimiž jsme denně ve styku a u nichž je kladen přísný důraz na funkčnost a spolehlivost, až po fontánu, která uspokojuje spíše touhu po estetice a nevšední podívané. Všechny tyto úkoly zvládají PLC stejně snadno.

Elektronická opora předmětu ve formě webové aplikace dostupné na adrese www.plc.unas.cz je určena pro posluchače předmětu Programovatelné automaty a dále pro studenty v kombinované formě studia. Kromě základních informací o předmětu jsou na stránce k dispozici také studijní materiály, učební pomůcky a v neposlední řadě jsou zde zveřejněny aktuální informace vztahující se ke studiu.

1 PŘÍPRAVA LABORATOŘE

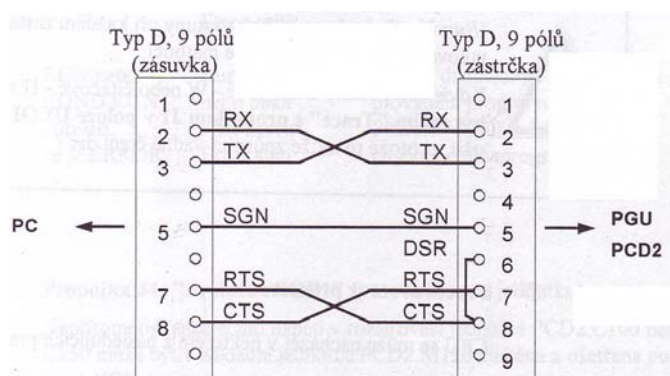
1.1 Schéma zapojení

Cílem navrhovaných úloh je umožnit řízení modelu fontány pomocí vizualizačního panelu v PC. Aby bylo možné tento požadavek splnit, je nutné propojit PC s PLC pomocí sériové linky RS-232. TTL signál z programovatelného automatu je poté přiveden do ovládací skříně a po jeho zpracování je odeslán do fontány.



Obr. 1 Schéma zapojení

Komunikace mezi PLC a PC probíhá po sériové lince pomocí tohoto zapojení:



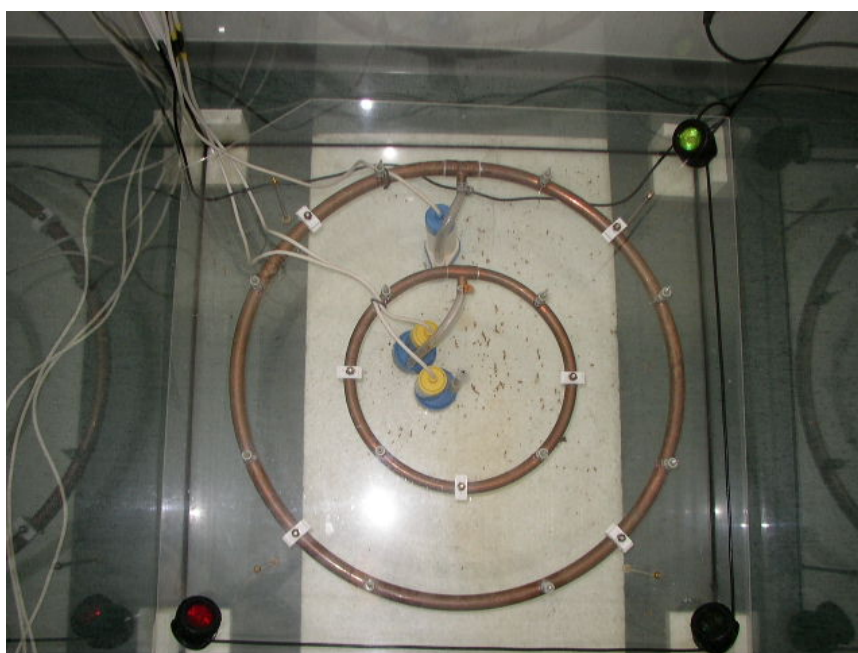
Obr. 2: RS-232 propojení

1.2 Model fontány

1.2.1 Stručný popis fontány

Použitý model fontány má tvar krychle o rozměrech 80x80x120 cm. Obvodové stěny a dno modelu jsou vytvořeny ze skleněných desek tloušťky 5 mm.

Uvnitř krychle jsou umístěny tři okruhy, přičemž každý z těchto okruhů představuje samostatný funkční celek obsahující čerpadlo a trysky. Uprostřed je umístěn první okruh - centrální vodotrysk s vlastním čerpadlem. Dále od středu se nachází měděná skruž se čtyřmi tryskami představující druhý okruh. Třetí okruh je tvořen měděnou skruží s největším obvodem a osmi tryskami.



Obr. 3: Model fontány

1.2.2 Čerpadla

V modelu jsou použita vysoce výkonná nízkonapěťová čerpadla typu BWV 12 l a BWV 22 l od firmy Barwig Wasserversorgung.

Silnější čerpadlo typu BWV 22 l bylo použito pro připojení na třetí okruh a má tyto parametry:

- napájení 12V SS; 3,7 až 4,2 A
- maximální průtok 22 l/min
- maximální dopravní výška 14 m (1,4 bar)
- vhodné pro hadice rozměru od 10 mm
- rozměry 80x135 mm
- hmotnost 290 g



Obr. 4 Čerpadlo 22 l

Slabší čerpadlo typu BWV 12 l použité pro první a druhý okruh je charakterizováno těmito parametry:

- napájení 12V SS; 2,1 až 2,9 A
- maximální průtok 12 l/min

- maximální dopravní výška 10 m (1,0 bar)
- vhodné pro hadice průměru 10 mm
- rozměry 64x100 mm
- hmotnost 260 g



Obr. 5 Čerpadlo 12 l

1.3 Ovládací skříň FONTÁNA (OSF)

1.3.1 Konstrukce

OSF byla sestrojena v předchozí BP *Počítačové řízení fontány* k výkonovému ovládání tří čerpadel a jedné skupiny světel. Mechanicky je připravena pro ovládání až třech samostatných světelných okruhů (zatím uvnitř skříně nerealizováno).

OSF obsahuje 4 nezávislé generátory šířkových pulzů a 4 výkonové spínací obvody. Protože proudová spotřeba použitých motorků čerpadel je při nízkém napájecím napětí poměrně vysoká (je uvažováno s proudem až 8 A pro jedno čerpadlo), a klasické napájecí zdroje by byly příliš objemné, bylo pro napájení čerpadel využito hotových spínaných zdrojů z vyřazených počítačů PC. Spínané zdroje při stejném výstupním výkonu mají podstatně menší hmotnost a objem, než zdroje klasické. OSF tedy obsahuje 3 samostatné spínané zdroje +12 V, každý z nich je využit k napájení jednoho čerpadla a jednoho světelného okruhu (použité osvětlovací žárovky mají nižší proudovou spotřebu než čerpadla).

Skříň je tvořena robustní kovovou konstrukcí a plechovým krytím.

Čelní panel OSF je vyobrazen na obr. 6.



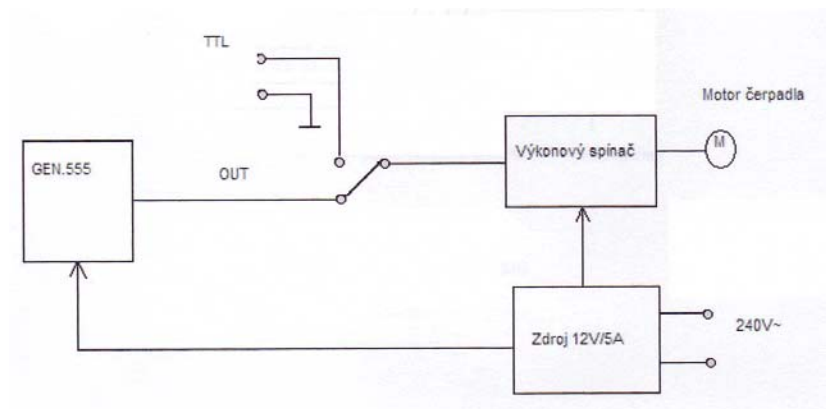
Obr. 6 Ovládací skříň

Otáčky a tím čerpací výkon a výtlač použitých čerpadel lze řídit buď manuálně ovládacími prvky na čelním panelu OSF, nebo externě přivedenými signály TTL. K přepínání MAN/EXT slouží 6 přepínačů, k přivedení externích řídicích signálů 6 konektorů CINCH. Správnou funkci tří zabudovaných zdrojů signalizují 3 kontrolní žárovky.

1.3.2 Regulace otáček pomocí PWM

K manuálnímu řízení či testování připojených čerpadel či světel slouží jednoduchý generátor, který vyrábí pulzně šířkově modulovaný (PWM) obdélníkový signál s regulovatelnou šířkou pulzů.

Blokové schéma jednoho z šesti kanálů OSF je na obr. 7



Obr. 7 Schéma regulačního obvodu

Všechny čtyři vestavěné generátory mají totožné zapojení a jejich schéma je uvedeno v Příloze P4.

Bylo použito známé zapojení jednoduchého generátoru s časovačem NE555. Frekvence pulzů je dána kombinací C1, R1, P1 a v našem případě je přibližně 77 Hz. Vlastní regulace šířky pulzů v poměru 1:99 je možná změnou nastavení potenciometru P1, přičemž výsledná frekvence se příliš nezmění. Z výstupu OUT časovače jsou šířkově modulované pulzy vedeny přes přepínač S1 na vstup jednoduchého spínacího obvodu tvořeného tranzistorem TIP120. K ochraně výkonového tranzistoru před vznikajícími napěťovými špičkami a ke snížení rušení motorku slouží paralelně připojená dioda D5 a kondenzátor C2.

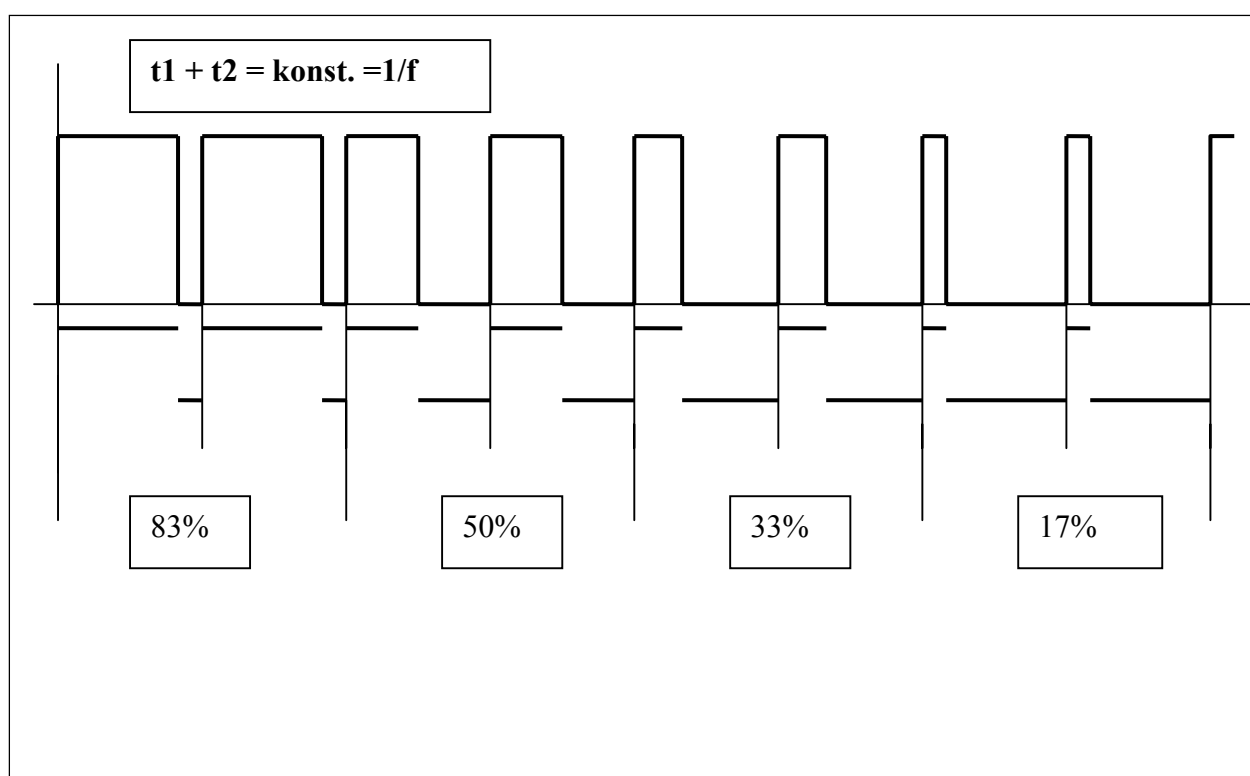
Přepínač S1 slouží k přepínání ovládacích pulzů z vnitřního generátoru (poloha MAN) nebo externího zdroje pulzů logické úrovně TTL (poloha EXT).

Čerpadla nebo světla se připojují na příslušným popisem opatřenou svorkovnici, která je umístěná na zadním panelu OSF.

1.3.3 Pulzně šířková modulace (PWM)

Otáčky stejnosměrného motoru čerpadla a tím i průtok vody lze řídit pomocí změny napájecího napětí. Při výrazném snížení napájecího napětí však dochází ke značnému poklesu výkonu čerpadla, což je jev nežádoucí. Proto je mnohem výhodnější řídit otáčky stejnosměrného motoru pomocí pulzně šířkové modulace (PWM). Motor je napájen přerušovaným napětím jmenovité hodnoty 12 V. Tímto způsobem lze snadno měnit otáčky motoru prakticky od nuly do maxima. Při nízkých otáčkách přitom nedochází k výraznému poklesu kroutícího momentu, jako je tomu u změny otáček pomocí změny napájecího napětí.

Pulzně šířková modulace s konstantní frekvencí f obdélníkových pulzů je jednou z metod regulace výkonu. Změnou šířky ovládacích pulzů lze i poměrně jednoduchými prostředky dosáhnout požadované změny např. při regulaci výkonu topných těles, osvětlení, otáček malých stejnosměrných motorů. Princip spočívá ve změně doby t_1 , po kterou je ovládané zařízení v akci (např. žárovka svítí) a doby t_2 , po kterou je toto zařízení v klidu (žárovka nesvítí). Délka pulzů, tj. součet $t_1 + t_2$, je konstantní. Změnou poměru t_1/t_2 neboli tzv. střídy pulzů lze dosáhnout regulace výkonu teoreticky od nuly do maxima.



Obr. 8 Pulzně šířková modulace

1.4 Programovatelný automat

1.4.1 Programovatelný automat obecně

Programovatelný automat je programovatelný řídicí systém přizpůsobený pro řízení průmyslových a technologických procesů nebo strojů. Nejčastěji se označuje zkratkou PLC (Programmable Logic Controller).

Program v PLC je vykonáván v cyklech, kdy jsou na začátku přečteny vstupy a na konci jsou vloženy výstupní hodnoty na výstupní bránu. Z toho je patrné, že program neumí zachytit změnu vstupu uprostřed cyklu a stejně tak změna na výstupu se projeví až po proběhnutí programové smyčky. Výhodou může být v omezené míře eliminace rušení na vstupu, nevýhodou však riziko nezachycení krátkého impulsu nebo jeho náběžné resp. sestupné hrany .

Z hlediska konstrukčního řešení můžeme PLC dělit na Mikro PLC, Kompaktní PLC a Modulární PLC.

Konstrukční a elektrické provedení PLC

Základní modul - v základním modulu modulárních PLC (rám, nosná deska, lišta) bývá umístěna centrální procesorová jednotka (CPU), napájecí jednotka, systémová a napájecí sběrnice a několik volných pozic pro zásuvné periferní jednotky. Pokud počet volných pozic v základním modulu nepostačuje, bývají k dispozici rozšiřující moduly. Rozšiřující moduly se připojují pomocí tzv. expanzní jednotky.

Centrální procesorová jednotka - je jádrem celého PLC, do jisté míry určuje jeho výkonnost. Může být jednoprocessorová nebo víceprocesorová. U víceprocesorových systémů bývají použity matematické koprocessory nebo vstupně výstupní procesory, někdy též komunikační procesory, pro urychlení práce používá rychlý bitový procesor. Důležitým charakteristickým parametrem CPU je operační rychlost posuzována podle tzv. doby cyklu, což je doba zpracování 1000 (1k) instrukcí. Pohybuje se řádově od desítek mikrosekund až k desetinám milisekund.

Paměťový prostor se může dělit na paměť uživatelskou, systémovou a paměť dat. Do uživatelské paměti se ukládá uživatelský program. Tato paměť bývá typu EPROM nebo EEPROM s kapacitou řádově od desítek kB až po jednotky MB u modulárních PLC, u kompaktních PLC spíše v desítkách kB. V systémové paměti je umístěn systémový program. Tato paměť bývá rovněž typu EEPROM. V samostatné jednotce může být umístěna přídatná uživatelská paměť. V paměti dat typu RAM jsou umístěny uživateli dostupné registry, čítače, časovače a většinou i vyrovnávací registry pro obrazy vstupů a výstupů. Počet těchto registrů výrazně ovlivňuje možnosti PLC. Adresovatelný prostor vymezený pro vstupy/výstupy omezuje počet připojitelných periferních jednotek.

Binární vstupní jednotky slouží k připojování vstupů s dvouhodnotovým charakterem signálu např. tlačítka, přepínače a koncové spínače.

Vstupní binární modul zajišťuje většinou tyto funkce:

- ochranu všech konkrétních vstupů PLC před poškozením či zničením chybným napětím či přepětím
- odfiltrování krátkodobých rušivých impulsů (např. pomocí zpoždění signálu)
- galvanické oddělení obvodů vstupního modulu od centrální jednotky (pomocí optočlenů)
- signalizace stavu vstupů (pomocí LED diod na čelním panelu jednotky)

Binární výstupní jednotka plní tyto funkce:

- galvanické oddělení signálu přicházejícího z CPU od signálu předávaného z výstupní jednotky akčním členům (pomocí optočlenů)
- zesílení signálu na potřebnou úroveň
- ochrana výstupu před zkratem
- signalizace stavu výstupů (pomocí LED diod)

Kombinované jednotky binárních vstupů a výstupů mohou být využity pro optimální doladění sestavy, ale především jsou určeny k multiplexovanému buzení a snímání matice tlačítek do rozsahu 8 x 8 nebo až 16 pater plochých zadávacích přepínačů.

Analogové vstupní jednotky slouží pro převod analogového vstupního signálu určité úrovně na jemu odpovídající binární číslo. Lze k nim připojit např. snímače teploty, vlhkosti, tlaku, síly, hladiny, rychlosti, ale i inteligentní přístroje s analogovými výstupy. Důležitou součástí je A/D převodník s rozsahem 8 nebo 12 bitů. Sortiment bývá široký, např. jednotky pro určité typy čidel – pro termočlánky.

Analogové výstupní jednotky slouží pro ovládání různých akčních členů či zařízení, které je ovládáno spojitým vstupním signálem, jako např. spojitě servopohony, frekvenční měniče, ručkové měřicí přístroje apod. Nezbytnou součástí je D/A převodník, většinou má rozsah 8 nebo 12 bitů. Analogové výstupy jsou buď napěťové nebo proudové. Proudové výstupy mohou být aktivní nebo pasivní. U aktivních nemusí mít připojený akční člen svůj zdroj proudu a je napájen přímo z analogového proudového výstupu automatu. Připojený akční člen má v tomto případě omezený maximální odpor.

Rychlé čítače, odměřovací a polohovací jednotky jsou určeny k měření a řízení polohy, k řízení dráhy a rychlosti pohyblivých částí strojů a manipulačních mechanismů. Čítačové jednotky jsou určeny k čítání pulsů, jejichž perioda je srovnatelná nebo kratší, než je smyčka programu PLC.

Polohovací jednotky jsou určeny pro snímání polohy a řízení jedné nebo dvou souvislých os, případně pro řízení pohybu po naprogramované dráze. Parametry pohybu jsou zadávány z PLC na základě programu.

Komunikační jednotky většinou rozšiřují počet asynchronních sériových komunikačních kanálů. U některých systémů jsou k dispozici i jednotky dálkových přenosů umožňující dálkové přenosy dat přes modem nebo přes radiomodem, RS 232, RS 422, RS 485.

V **počítačovém modulu**, kompaktním s PC, lze standardními počítačovými prostředky řešit úlohy, které nejsou pro PLC typické, např. složité a rychlé výpočetní algoritmy, grafické a geometrické úlohy, zpracování a archivace velkého množství dat, databázové úlohy, výkonné komunikace, napojení PLC do počítačových sítí, připojení standardních počítačových periférií, paměťových karet PCMCIA, apod.

Programové vybavení

Při své činnosti používá programovatelný automat dva typy programového vybavení:

- systémový program (operační systém)
- uživatelský program, v němž je zapsán algoritmus řešení dané úlohy

Systémový program (operační systém) programovatelných automatů je řešen pro každý automat poněkud odlišně a je závislý na technickém vybavení automatu. Je schopen zpracovávat instrukce příslušné danému typu automatu. Podle typu automatu musí být schopen zajistit v nejjednodušším případě režim jednoúlohového zpracovávání uživatelského programu bez přerušení, u složitějších systémů víceúlohové zpracovávání (multitasking), u nejvyspělejších systémů s několika procesory pak víceúlohové zpracovávání na všech dostupných procesorech (multitasking i multiprocessing).

Uživatelské programy lze vytvářet buď pomocí samostatných přístrojů, tzv. programátorů, nebo pomocí PC. Pro většinu PLC je k dispozici speciální programové vybavení pro tvorbu uživatelských programů na PC. V něm je možno konfigurovat celý systém, provádět vlastní zápis a editaci programu, editovat alokační tabulky proměnných. Bývá nabízena syntaktická kontrola programu po překladu do jazyka strojových instrukcí. Vyspělejší systémy poskytují možnost odlaďování programu bez přímé účasti PLC v simulačním režimu.

K programování nabízejí PLC systémy specializované jazyky, původně navržené pro snadnou, názornou a účinnou realizaci logických funkcí. Ustálily se tři třídy jazyků pro programování PLC.

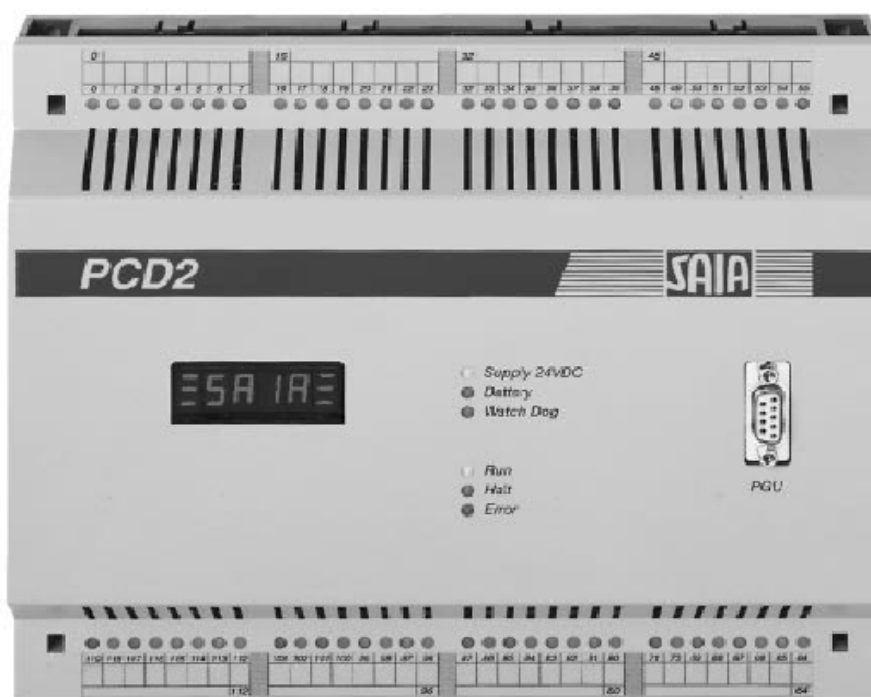
- Jazyk mnemokódů je obdobou assembleru u počítačů a je také strojově orientován, tzn. že každé instrukci PLC systému odpovídá stejně pojmenovaný příkaz jazyka. Dovolují nejlépe přizpůsobit úlohu možnostem PLC a vytěžit maximum z jeho instrukčního souboru.
- Jazyk kontaktních reléových schémat je grafický program, ve kterém se základní logické operace zobrazují ve formě obvyklé pro kreslení schémat při práci s reléovými a kontaktními prvky.
- Jazyk logických schémat je také grafický. Základní logické operace popisuje obdélníkovými značkami. Vychází vstříc uživatelům zvyklým na kreslení logických schémat.

Některé systémy pracují ještě s dalšími jazyky, například s jazykem C, s textovým jazykem, vycházejícím z Pascalu. [1]-[3]

1.4.2 Programovatelný automat SAIA PCD2

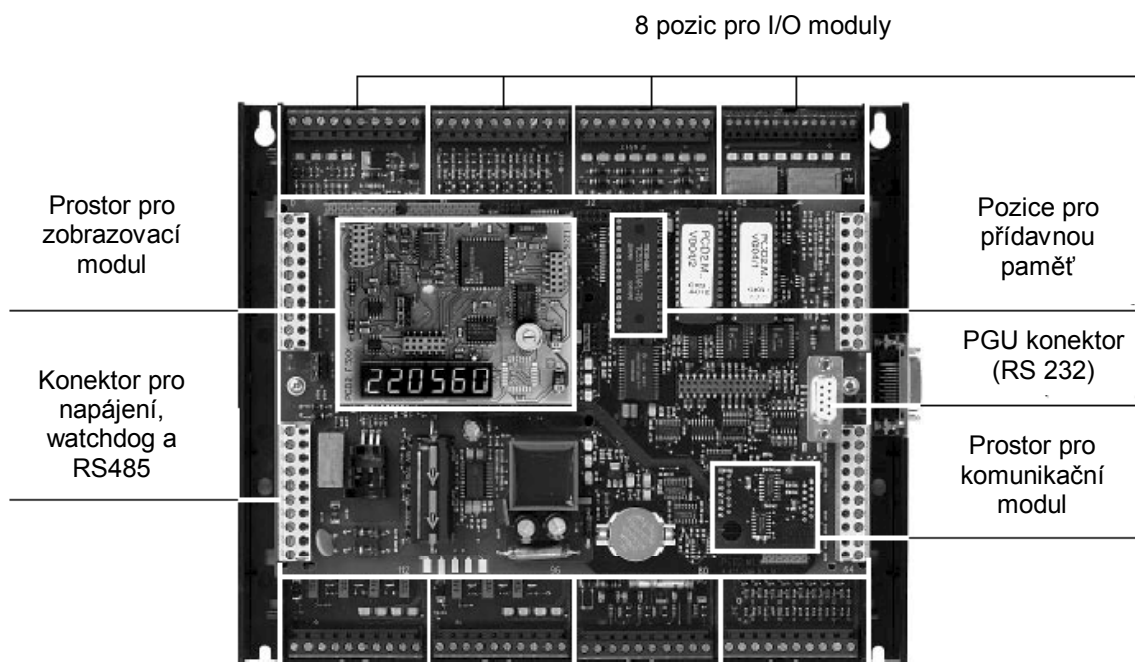
Architektura PCD2

SAIA PCD2 je programovatelný automat kompaktní konstrukce. Základní jednotka obsahuje hlavní desku se všemi aktivními prvky, za ní je sběrnice deska s konektory pro připojení vstupně-výstupních modulů – čtyři na horním okraji a čtyři na dolním. Celek je uzavřen v ploché skřínce, jež umožňuje přístup jen ke konektoru PGU (Programming Unit) pro připojení programovacího zařízení a svorkovnicím jednotlivých modulů na stranách přístroje. Světlovody přivádějí od LED diod na plošných spojích na povrch optickou indikaci stavu základní jednotky a jednotlivých vstupů a výstupů.



Obr. 9 Programovatelný automat SAIA PCD2

Po sejmutí krytu se zpřístupní svorkovnice pro napájení přístroje 24V=, komunikační rozhraní a kontakty relé watchdogu. Nyní lze na kontakty sběrnice desky zasunout I/O moduly. Na hlavní desce jsou také dvě pozice k instalaci rozšiřujících piggybacků s komunikačními a zobrazovacími obvody a patice pro rozšíření paměti (RAM i EPROM). Při vypnutí napájení jsou paměti zálohovány knoflíkovým článkem.



Obr. 10 Hlavní deska SAIA PCD2

Datová komunikace mezi hlavní deskou a přídavnými moduly probíhá pomocí adresovaných kanálů. Každý I/O modul má vyhrazeno 16 adres, podle pozice, na níž je instalován. Adresace začíná v levém horním rohu sběrnice desky od bazové adresy 0 a pokračuje ve směru hodinových ručiček. Pro každý modul je bazová adresa zvyšována vždy o 16. Toto pravidlo platí vždy, nezáleží na tom, zda modul využívá 16, 8, 6 nebo jen 4 adresy. V případě potřeby připojení více než osmi modulů lze použít rozšiřující jednotku, která obsahuje sběrnici s dalšími osmi pozicemi s bazovými adresami 128 – 240. Viz obr. 15 v příloze A.

Pozn.: Protože obvodu watchdog na hlavní desce je interně přiřazena adresa 255, nesmí být na adrese 240 instalován modul, který používá všech 16 adres.

Dodávané typy vstupně-výstupních modulů

- *komunikační rozhraní* (RS 422, RS 485, RS 232, proudová smyčka 20mA) (jako piggyback)
- *Hodiny reálného času + displej* (jako piggyback)
- *Dvouhodnotové vstupní a výstupní moduly* – 4 až 8 vstupů nebo výstupů, 24V= nebo 250 V~, proud až 2A. Galvanicky neoddělené i optočleny oddělené vstupy, výstupy s tranzistory MOSFET nebo relé.
- *Analogové vstupní a výstupní moduly* s 4 až 8 kanály s rozlišením A/D, D/A převodníků 8 až 12 bitů pro signály 0 – 10V, -10 – 0V, -10 – 10V, 0 – 20mA, -20 – 0mA, -20 – 20mA, 4 – 20mA a pro přímé připojení teploměrů Pt, Ni 100, 1000
- *Čítací a polohovací moduly*
- *Zobrazovací moduly a terminály*

Typy datových prostředků využitelných v programech

Data a hodnoty, které program používá jsou striktně rozděleny do datových typů. Do paměti nelze přistupovat libovolně, ale vždy jen k prvku určitého typu a pořadového čísla (adresy) v rámci typu. Tabulka II. obsahuje přehled použitelných typů dat.

Tabulka I. Přehled datových typů prostředků programovatelného automatu SAIA

Kód typu	Typ	Popis	Rozsah adres
I	Vstup (Input)	Hodnota fyzického dvouhodnotového vstupu I/O modulu (adresa podle umístění modulu)	0..8191
O	Výstup (Output)	Slouží k nastavení nebo přečtení stavu fyzického dvouhodnotového výstupu I/O modulu (adresa podle umístění modulu). I a O sdílejí stejné adresy.	0..8191
F	Flag	Binární proměnná	0..8191
T	Časovač (Timer)	Proměnná k nastavení nebo čtení stavu časovače	0..450
C	Čítač (Counter)	Proměnná k nastavení nebo čtení stavu čítače. T a C sdílejí adresy, je možno si zvolit, co bude čítač a co časovač.	0..1599
R	Registr (Register)	Obecný číselný registr (32 bitů). Uložená hodnota může být celé nebo reálné číslo	0..4095
K	Konstanta (K constant)	Podle přítomnosti desetinné tečky se rozlišují celá a reálná čísla	
X	Text	Slouží k ukládání řetězců do paměti	0..7999
DB	Datový blok (Data Block)	Slouží k ukládání bloků dat do paměti. Výhodné pro větší množství dat, používají se speciální instrukce. X a DB sdílejí stejné adresy, je nutno zvolit, zda půjde o řetězec nebo data.	0..7999

Uvedený rozsah adres platí vždy jen v rámci typu (kromě výjimek sdílení adres). Např. Flag F15 je uložen jinde než registr R15. V programech lze samozřejmě využívat možnosti přiřazení symbolických označení (např. direktivou EQU v Instruction List).

Struktura uložení programu

Podobně jako k datům a proměnným lze přistupovat pouze systematicky pomocí adres v rámci datového typu, program se musí zapisovat formou bloků různých typů podle určení. Tabulka II. obsahuje přehled použitelných bloků.

Tabulka II. Přehled typů programových bloků

Kód typu	Typ bloku	Popis	Rozsah číslování
COB	Cyklický organizační blok (Cyclic Organization Block)	Části programu, umístěné v těchto blocích, jsou cyklicky prováděny (postupně se vykoná COB0, COB1 až po COB15 a znova od COB0)	0..15
XOB	Blok výjimky (Exception Organization Block)	K jednorázovému spuštění bloků tohoto typu dochází při výjimečných událostech přiřazených každému číslu XOB. Např. XOB0 se provede při výpadku napájení, XOB13 při výskytu chyby, XOB16 při studeném startu PCD	0..31
PB	Programový blok (Program Block)	Blok podprogramu (procedury). Volá se z jiného bloku a po provedení se vrací zpět.	0..299
FB	Funkční blok (Function Block)	Funkce – použití stejné jako u PB, při volání lze navíc předávat parametry.	0..999
SB	Sekvenční blok (Sequential Block)	Blok pro programování sekvenčních algoritmů (Graftec), obsahuje IST, ST a TR	0..31
IST	Úvodní krok (Initial Step)	Provádí se na začátku sekvence definované SB	0..1999
ST	Krok (Step)	Blok obsahující krok – dílčí část sekvence	0..1999
TR	Přechod (Transition)	Blok obsahující podmínky, na jejichž základě se rozhoduje o přechodu na další krok	0..1999

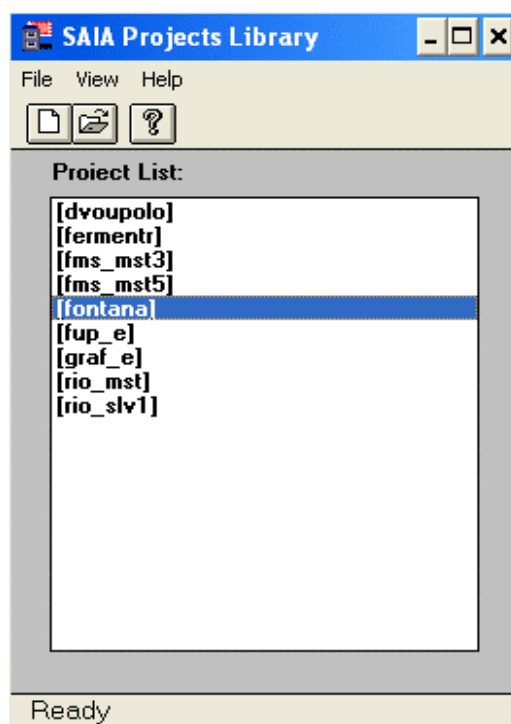
V programu se nesmějí vyskytovat bloky stejného čísla v rámci typu. [4]

1.5 Vývojové prostředí PG4

K tvorbě programů pro rodinu programovatelných logických automatů SAIA PCDx výrobce dodává vývojové prostředí PG4 běžící na PC s operačním systémem MS Windows 3.x, 95, NT. (Předošlá verze PG3 byla určena pro MD-DOS). Jedná se o ucelený balík nástrojů pro tvorbu jednotlivých rutin v různých programovacích jazycích, jejich kompilování, správu knihovny takto vzniklých objektů, linkování programů až správu celých projektů. Pomocí PG4 lze navíc on-line připojením k PCD měnit konfiguraci logického automatu, nahrávat do jeho operační paměti vytvořené programy, tyto spouštět a za chodu monitorovat na obrazovce PC či přímo krokovat až na úrovni nejzákladnějších funkčních bloků.

1.5.1 Knihovna projektů

Po spuštění vývojového prostředí SAIA PG4 se objeví okno knihovny projektů (SAIA Projects Library). Toto je nejvyšší úroveň, zde se k další práci vybírá jeden ze samostatných již vytvořených projektů nebo se zakládá nový.



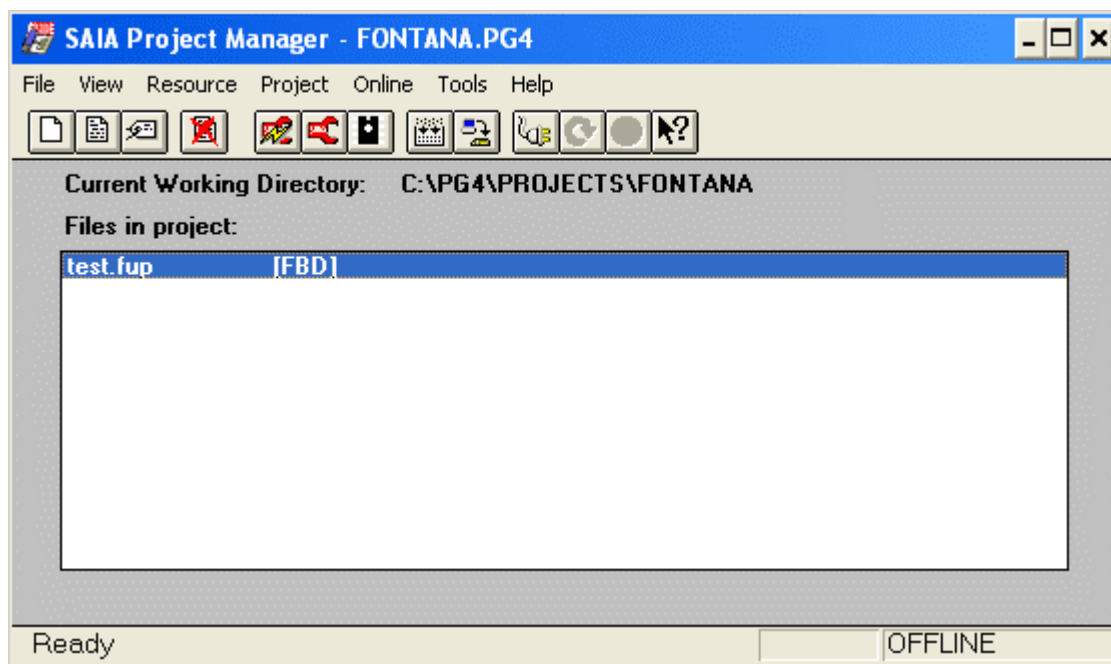
Obr. 11 Okno knihovny projektů

Po zvolení jednoho z projektů s ním lze začít pracovat ve Správci projektu (SAIA Project Manager).

1.5.2 Správce projektu

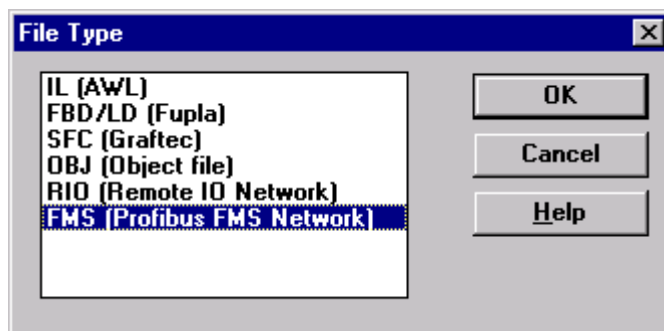
Okno správce projektu obsahuje seznam programových modulů v tomto projektu obsažených, z nichž se sestavuje výsledný program. V případě potřeby lze samozřejmě určitý modul ze sestavování vyjmout, aniž by se musel zcela odstranit z projektu.

Správce je jádrem PG4. Odtud lze linkovat program, připojit se on-line k PCD, nahrát do něj program a následně jej spustit, krokovat. K dispozici jsou zde také nástroje ke konfigurování paměťových zdrojů PCD – rozvržení prostoru RAM pro dynamické proměnné, čítače, časovače, uživatelské texty a datové bloky. Online Configurator umožňuje přímo nastavovat systémové prostředky logického automatu - vnitřní obvod reálného času, parametry komunikačních rozhraní, případně zablokovat další přístup k paměti PCD heslem (pro zabránění neoprávněného kopírování či modifikace software).



Obr. 12 Okno správce projektu

Při vytváření nového objektu se volí programovací jazyk, v němž bude vytvořen, a tím i editor, který se pro něj bude používat:



Obr. 13 Výběr typu programového objektu

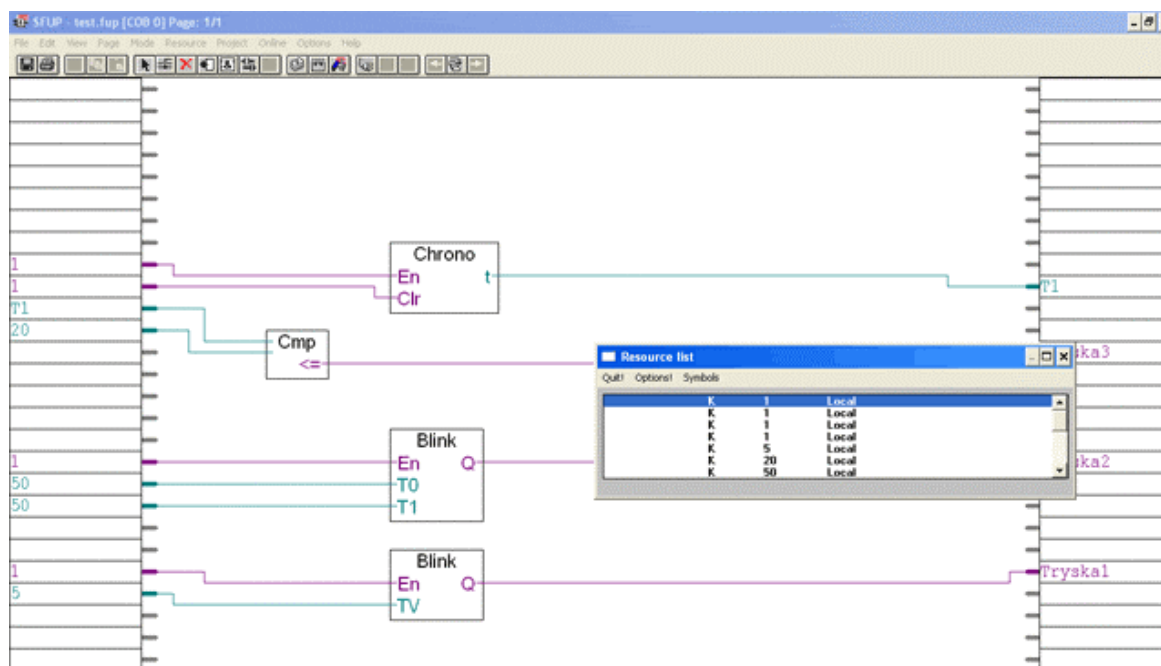
- **IL (AWL)** - Instruction List (assembler), editor AWL (textový editor)
- **FBD/LD** - Function Block Diagram/Ladder Diagram (schéma funkčních bloků), editor FUPLA (Function Plan)
- **SFC** - Grafcet – grafický jazyk pro programování sekvencí bloků

Dalšími možnostmi jsou:

- **OBJ** - předem zkompilovaný objekt k zahrnutí do výsledného programu
- **RIO** - Remote IO Network
- **FMS** - Profibus FMS Network – objekty připojené ze vzdáleného zdroje po síti

1.5.3 Function Plan – FUPLA (FBD)

FUPLA (Function Plan) je grafický jazyk založený na makrech s výhodou dynamického přiřazování datových prostředků.



Obr. 14 Okno editoru FUPLA

Do „sběrnice“ na levém okraji strany se vepisují vstupní data bloku – konstanty, registry, flagy, vstupy, výstupy, čítače, časovače apod. a to buď absolutně, tj. kódem datového typu a číslem, nebo symbolicky. Podobně pravá „sběrnice“ obsahuje prostředky přijímající výstupní hodnoty.

Seznam všech použitých datových prostředků a jim přiřazených symbolů lze vyvolat výběrem *Resource->Resource List...* z hlavního menu. Nebyl-li registr, flag, časovač nebo čítač definován jako externí a nespécifikuje-li se jeho číslo, bude mu přiřazeno dynamicky v okamžiku sestavování programu. Oblasti adres, ze kterých může překladač v tomto případě čerpat, se nastaví vyvoláním *Resource->Dynamic Distribution...* z menu Správce projektu (Project manager). Datové prostředky z takto rezervovaných oblastí by neměly být použity absolutní adresací ve zbytku programu.

Program je sestaven z funkcí, reprezentovaných obdélníky s popisem, vstupy vlevo a výstupy vpravo (FBox). Předávání dat mezi nimi definují spojnice. Funkce se provádějí postupně z levého horního rohu dolů, při zachování logické návaznosti. Jsou definovány v knihovnách, dodávaných s vývojovým prostředím, v nichž je uložen jim odpovídající kód ve formě výpisu instrukcí a symbolickým vyjádřením vstupujících a vystupujících dat. Při překládání programu se použije tento kód jako makro (překopíruje se) a symboly se

nahradí konkrétními čísly datových prostředků. Opět se čerpá z oblastí rezervovaných pro dynamické přiřazování.

Striktně se rozlišují číselné typy hodnot – binární, celočíselné a s plovoucí desetinnou čárkou – a to i barvou spojnic. Např. celočíselný výstup nelze přivést na vstup očekávající reálné číslo, musí se použít konverzní funkce.

Každý soubor v jazyce FUPLA může obsahovat více bloků. K vybrání jednoho z nich ke zobrazení v editačním okně slouží seznam bloků, vyvolaný z menu *File->Block List...*. Volba *Properties...* umožňuje nastavit typ (COB, XOB, PB ...) a číslo bloku, *New...* zakládá nový blok.

Každý blok může mít rozsah více stran. Listování mezi nimi, přidávání nových a jejich organizaci umožňují funkce z nabídky *Page*.

Jazyk FUPLA je přehledný, programování v něm je jednoduché, ale některé nedostatky přinášejí omezení:

nelze použít skoky v programu, podmíněné ani nepodmíněné, což zcela znemožňuje realizaci cyklu

ze stejného důvodu lze podmíněně jen volat podprogramy, i jednoduchý program se tak může stát velmi nepřehledným

při volání funkčních bloků (FB) jim nelze předat parametry, což je degraduje na procedury (PB)

Jako vyšší jazyk přirozeně FUPLA oproti *Instruction List* vytváří redundantní kód, to je však samozřejmá daň za používání knihoven funkcí k univerzálnímu použití.

[5]

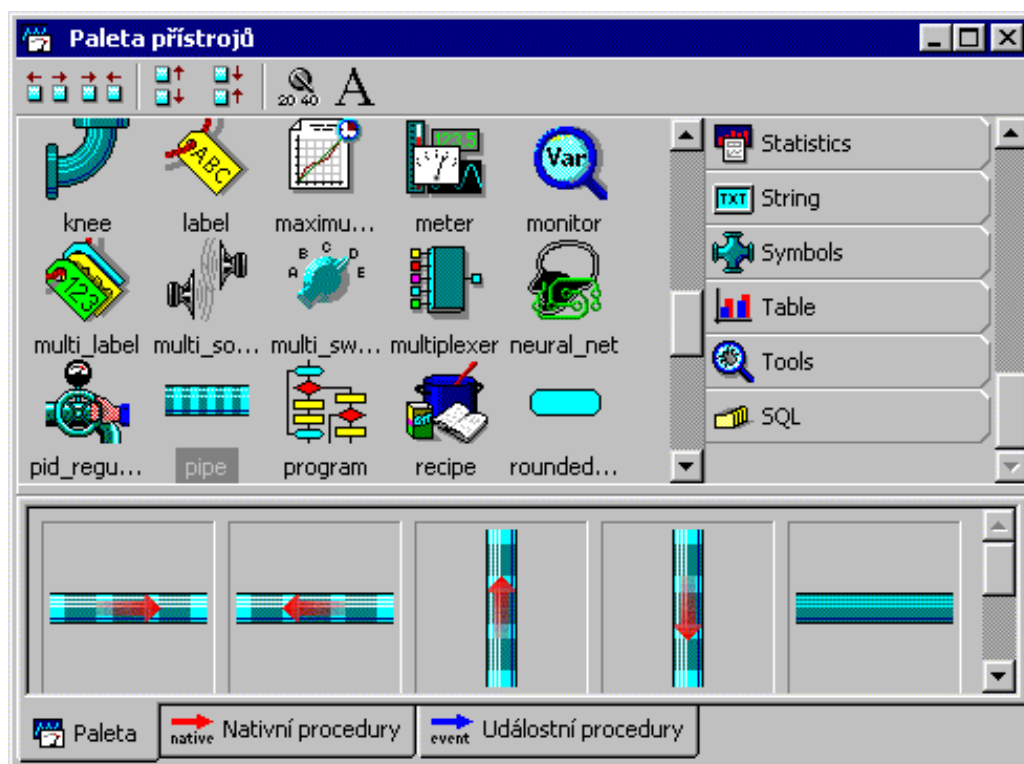
1.6 Control Web 2000

Control Web 2000 je univerzální nástroj pro vývoj a nasazování vizualizačních a řídicích aplikací, aplikaci sběru, ukládání a vyhodnocování dat, aplikací rozhraní člověk-stroj. Unikátní objektově-orientovaná komponentová architektura zajišťuje aplikacím systému Control Web 2000 nejširší rozsah nasazení od prostých časově nenáročných vizualizací až po řídicí aplikace reálného času.

1.6.1 Virtuální přístroje

Aplikační program v prostředí systému Control Web je sestaven z jednotlivých virtuálních přístrojů. Každý virtuální přístroj je zcela samostatnou programovou komponentou. Tato koncepce přináší řadu velmi podstatných výhod:

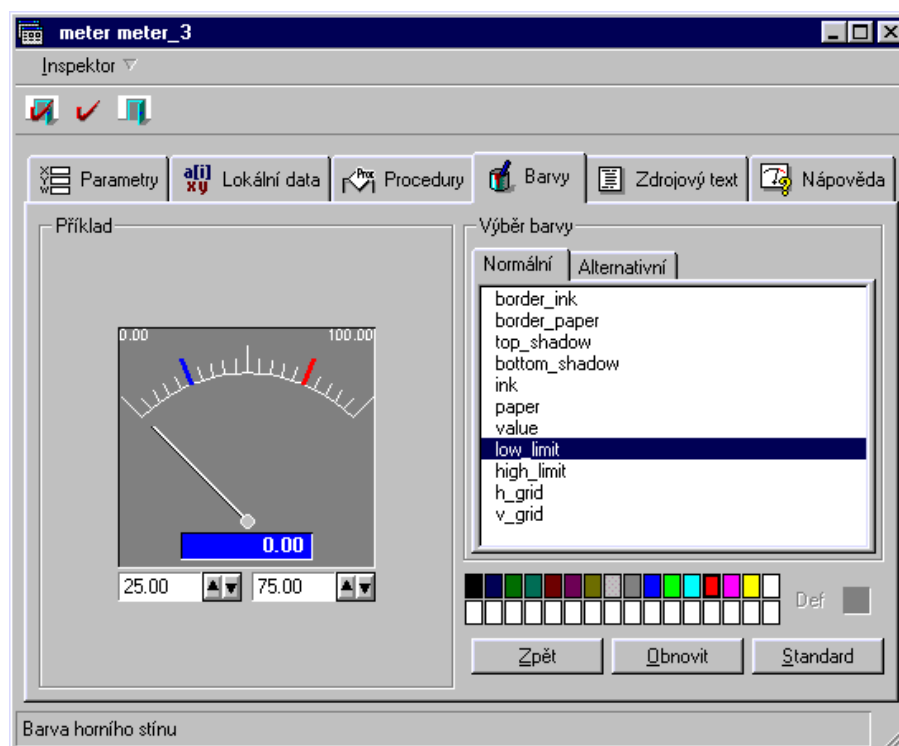
- Systém je trvale zcela otevřený - není nijak omezen ani předem dán počet a typ používaných virtuálních přístrojů. Virtuální přístroje je možno kdykoliv libovolně doplňovat a mohou dělat cokoliv, co jejich autor uzná za vhodné. Výrazové prostředky pro tvorbu aplikací jsou zcela neomezené.
- Aplikace pracuje plným výkonem nativního strojového kódu – podoba aplikace v paměti počítače – a tedy i její výkon – je zcela ekvivalentní napevno napsanému programu např. v C++.
- Aplikace jsou přenositelné nejen mezi jednotlivými verzemi systému, ale i mezi platformami – veškeré komponenty jsou vždy optimalizovány pro operační systém, na kterém aplikace běží.



Obr. 15 Virtuální přístroje v okně Paleta přístrojů

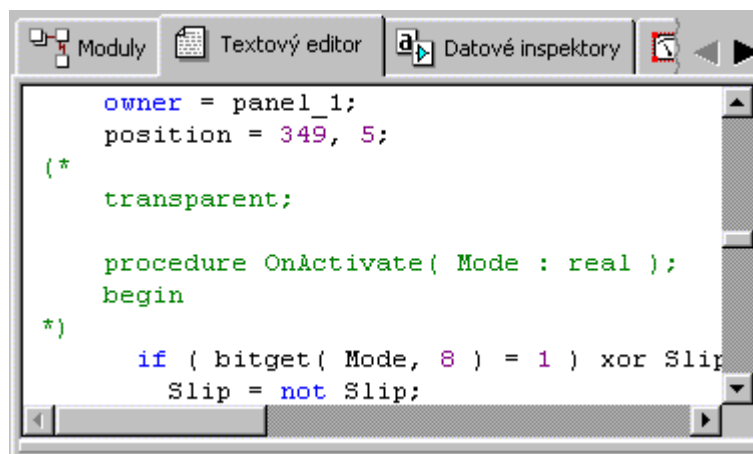
1.6.2 Vytváření programů

Nové programy je možné v aplikaci Control Web 2000 vytvářet pouze pomocí výběru virtuálních přístrojů, definování jejich vlastností a propojení s dalšími přístroji pomocí používaných proměnných. V krajním případě se lze snadno obejít i bez psaní jakéhokoliv zdrojového kódu – aplikace jej vytváří automaticky.



Obr. 16 Inspektor virtuálního přístroje

Control Web 2000 přirozeně umožňuje i psaní aplikací v textovém modu.



Obr. 17 Textový editor

Optimální je kombinovat obě metody programování – lze tak ušetřit spoustu času a zároveň dosáhnout přesně požadovaných výsledků.

1.6.3 Napojení aplikace na reálné zařízení

K uschování i předávání dat v systému **Control Web** slouží proměnné a kanály, společně nazývané **datové elementy**. Datový element je vyhrazené místo v paměti počítače, ve kterém je umístěna nějaká hodnota. Všechny datové elementy je v rámci aplikace nutné nejprve deklarovat (zapsat jejich jméno, typ, apod.) a teprve poté je možno je používat. Hodnoty datových elementů je možno číst i zapisovat. Virtuální přístroje si tedy mohou předávat data prostě tak, že jeden virtuální přístroj hodnotu datového elementu zapíše a druhý ji přečte.

Každá proměnná má nějaký datový typ — druh dat, které proměnná uchovává. Dvoustavovou logickou hodnotu nelze ukládat do číselné proměnné a číselnou proměnnou zase nelze ukládat do proměnné uchovávající text. V rámci systému **Control Web** existují následující typy datových elementů:

- Typ **boolean** : Velikost 1B. Dvoustavový datový element, může nabýt hodnot **true** (pravda) nebo **false** (nepravda).
- Typ **shortint** : Velikost 1B. Umožňuje přenášet hodnoty -128–127.
- Typ **shortcard** : Velikost 1B. Umožňuje přenášet hodnoty 0–255.
- Typ **integer** : Velikost 2B. Umožňuje přenášet hodnoty -32768–32767.
- Typ **cardinal** : Velikost 2B. Umožňuje přenášet hodnoty 0–65535.
- Typ **longint** : Velikost 4B. Umožňuje přenášet hodnoty -2147483648–2147483647.
- Typ **longcard** : Velikost 4B. Umožňuje přenášet hodnoty 0–4294967295.
- Typ **shortreal** : Velikost 4B. Umožňuje přenášet hodnoty $\pm 1.2\text{E}-38$ – $3.4\text{E}+38$.
- Typ **real** : Velikost 8B. Umožňuje přenášet hodnoty $\pm 2.3\text{E}-308$ – $1.7\text{E}+308$.

- Typ **string** : Velikost 256B. Libovolný řetězec znaků (maximálně 256 znaků).
- Typ **buffer** : Blok obecných dat.

Kanály

Do kanálů je možné zapisovat nebo je číst. Pokud se čte hodnota proměnné, stačí přečíst data z paměti, která je s proměnnou spojena. U kanálu je to poněkud složitější. Má-li být přečtena hodnota z kanálu, vyrozumí o tom jádro systému Control Web ovladač, který je s kanálem spojený. Ovladač potom nastaví do kanálu hodnotu. Podobně je tomu i při zápisu do kanálu.

I když je v principu možné kanály číst i do nich zapisovat, ne všechna vstupně/výstupní zařízení dokáží předávat data oběma směry. Proto je možné i kanály omezit podle směru přenosu dat.

- **Vstupní kanály** přenášejí data ze vstupně/výstupního zařízení do aplikace. Vstupní kanály lze pouze číst, pokus o zápis do vstupního kanálu způsobí chybu.
- **Výstupní kanály** přenášejí data z aplikace do vstupně/výstupního zařízení. Do výstupních kanálů lze samozřejmě zapisovat, ale lze je i číst. Čtení výstupních kanálů ale nezpůsobí žádnou komunikaci, pouze je vrácena naposledy zapsaná hodnota.
- **Obousměrné kanály** slouží ke čtení i zápisu dat z/do ovladače.

Ovladače

Ovladač je programová komponenta, která spojuje aplikaci v systému **Control Web** s konkrétním vstupně/výstupním zařízením. Aplikace může využívat služeb neomezeného

počtu ovladačů současně. Ovladač dostává požadavky na čtení nebo zápis dat prostřednictvím kanálů. Tento požadavek přetransformuje do takové podoby, aby mu bylo schopno porozumět vstupně/výstupní zařízení.

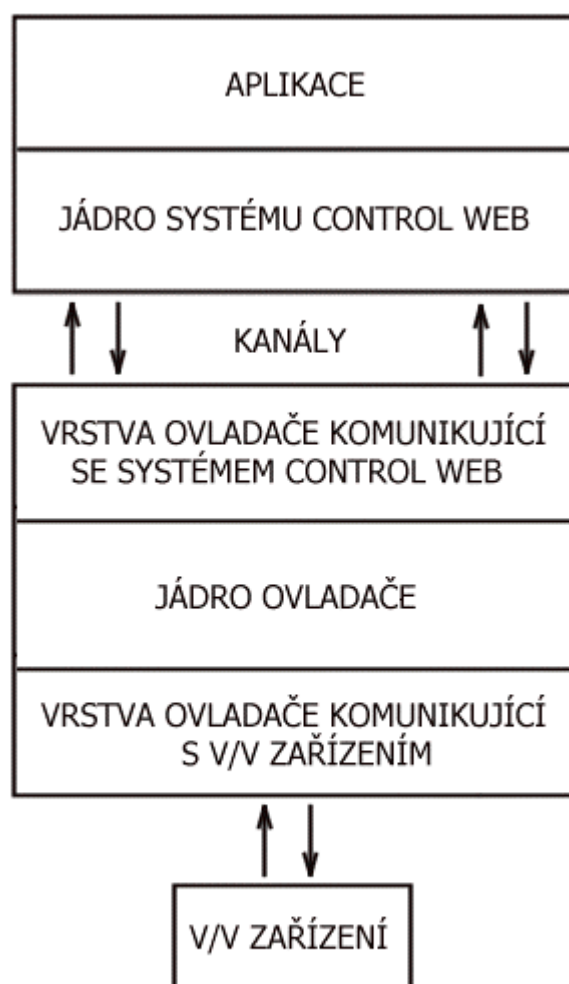


Schéma propojení aplikace systému Control Web s v/v zařízením.

Obr. 18 Propojení aplikace s v/v zařízením

[6]

1.6.4 Připojení k PLC Saia

Aby bylo možné zajistit komunikaci programu Contol Web 2000 s programovatelným automatem SAIA, je nutné nainstalovat odpovídající ovladač pro PLC SAIA a poté nastavit konfigurační soubory ovladače.

Ovladač se do systému nainstaluje spuštěním instalačního souboru s názvem *setup.exe*. Po dokončení instalace je nutné do adresáře, ve kterém je umístěn program CW200, nakopírovat všechny soubory, jež jsou součástí složky CD2 dodávané s instalací ovladače.

Dále je nutné zapsat údaje o ovladači do systémového registru – to zajistí spuštění souboru *SCOMM32.reg*. Aby bylo možné ovladač používat, je nutné vždy před spuštěním aplikace Control Web 2000 spustit také program *SCODRV32.exe*.

Ovladač pracuje s tzv. mapovacím a parametrickým souborem. Mapovací soubor obsahuje informace o kanálech, které je možné používat, včetně standardně přidělených čísel. Tento soubor nemůžeme měnit – je charakteristický pro každý ovladač.

Zasahovat lze do parametrického souboru. Ten obsahuje informace o použitém protokolu, dále o portu, ke kterému je zařízení připojeno a v neposlední řadě také o uživatelem definovaných kanálech.

Při definování kanálů je třeba vycházet z informací uložených v mapovacím souboru. Pokud je toto respektováno, není problém nadefinovat si vlastní bloky kanálů, určit jejich datové typy a instanci, kam budou signály zapisovány, resp. odkud budou načítány.

Příklad mapovacího souboru viz. Příloha P2.

Příklad parametrického souboru viz. Příloha P3.

1.7 Úlohy pro laboratorní cvičení

1.7.1 Úloha č. 1

Vytvořte program pro ovládání modelu fontány pomocí programovatelného automatu SAIA. Ovládání vizualizujte v programu Control Web 2000. Program musí zajistit možnost zapnout a vypnout všechna čerpadla fontány stiskem jednoho tlačítka na vizualizačním panelu. Stav chodu čerpadel musí být na panelu indikován pomocí odpovídajícího grafického prvku.

Vypracování

Úloha sestává ze dvou částí – v první části je nutné vytvořit program pro PLC v prostředí PG4. Program zajistí zápis informací z vizualizačního panelu do registru PLC, jejich zpracování a zápis na výstup.

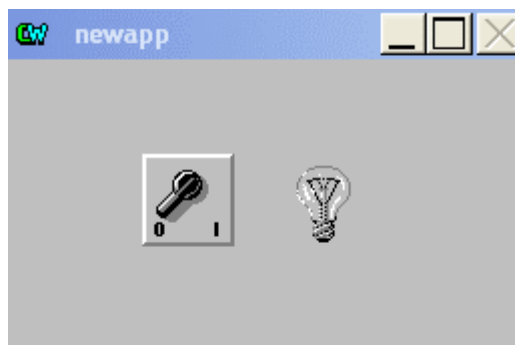


Obr. 19 Program v jazyce FUPLA Úlohu č.1

Informace z vizualizačního panelu je zapsána do proměnné *Zapnuto*. Jelikož jsou data do registru zapsána jako datový typ integer a výstup má být binárního typu, je nutné data před zápisem na výstup konvertovat pomocí funkčního bloku *Integer LSB to 1-bit*.

Hodnota proměnné *Zapnuto* je zapsána na všechny tři výstupy (tj. *Cerpadlo1*, *Cerpadlo2* a *Cerpadlo3*).

Ve druhé části je pak nutné vytvořit program v Control Webu 2000. Tento program umožní ovládání PLC a fontány pomocí grafických prvků na obrazovce počítače. Výstupem jsou informace předávané registru PLC.



Obr. 20 Panel pro Úlohu č. 1

Panel se v tomto případě skládá pouze ze dvou grafických prvků (virtuálních přístrojů) – spínače a indikátoru. Výstupem spínače je hodnota 0 nebo 1 zapisovaná do kanálu s názvem *Zapnuto*. Pomocí mapovacího a parametrického souboru (viz. Přílohy P6 a P7) ovladače je možné definovat jednotlivé kanály, datové typy a také to, kam bude jejich hodnota předávána. Kanál *Zapnuto* je datového typu boolean.

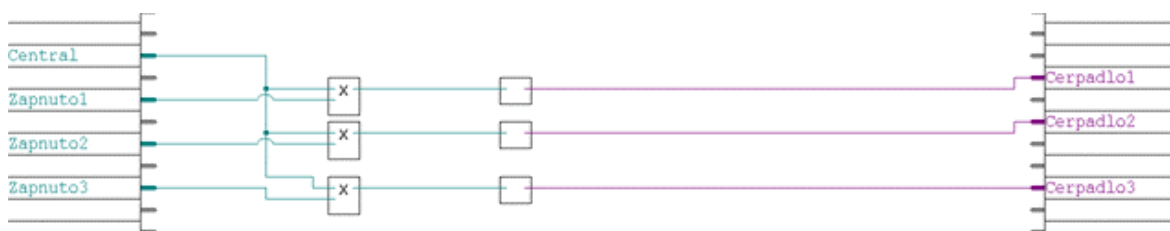
Hodnotu v kanálu *Zapnuto* vyhodnocuje indikátor znázorněný jako žárovka. V případě hodnoty rovné jedné žárovka svítí, v případě hodnoty rovné nule nikoliv.

1.7.2 Úloha č. 2

Vytvořte program pro ovládání modelu fontány pomocí programovatelného automatu SAIA. Ovládání vizualizujte v programu Control Web 2000. Každé čerpadlo musí být možné zapnout a vypnout vlastním tlačítkem. Program musí umožňovat také centrální zapnutí a vypnutí všech čerpadel jediným společným tlačítkem. Signalizujte stavy jednotlivých čerpadel pomocí indikátorů. Ošetření výstupů je zajištěno programem v PLC.

Vypracování

Úloha sestává ze dvou částí – v první části je nutné vytvořit program pro PLC v prostředí PG4. Program zajistí zápis informací z vizualizačního panelu do registru PLC, jejich zpracování a zápis na výstup. Vyhodnotí také stav centrálního spínače.

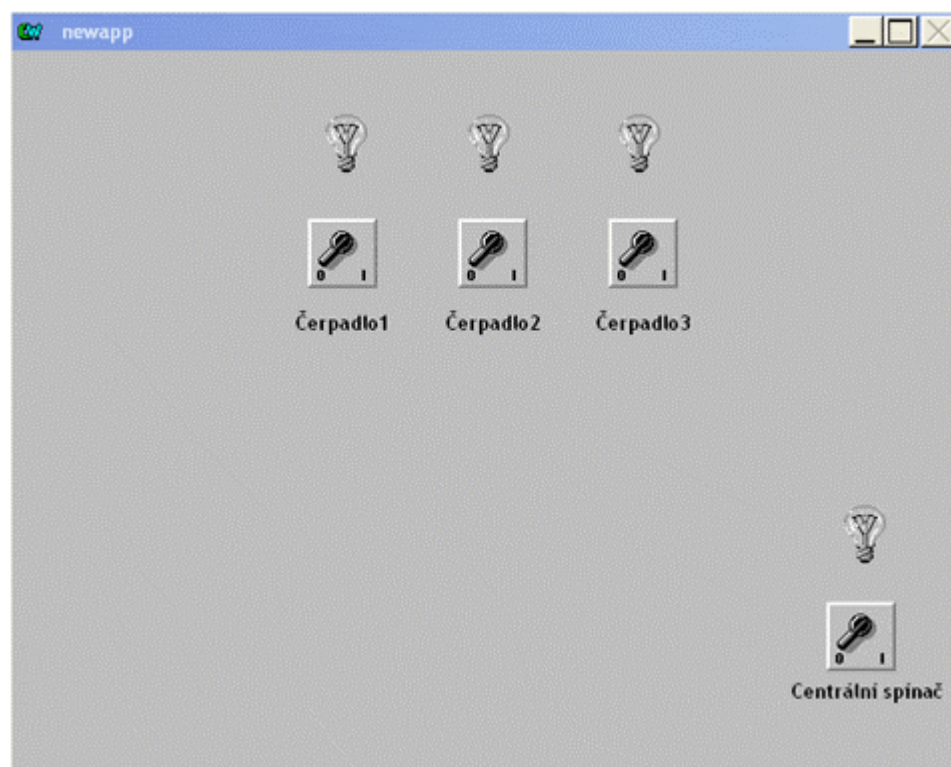


Obr. 21 Program v jazyce FUPLA Úlohu č.2

Pomocí kanálů ovladače jsou do registrů (*Central*, *Zapnuto1*, *Zapnuto2*, *Zapnuto3*) zapsány informace o stavech jednotlivých spínačů. Abychom dosáhli toho, že čerpadla budou spuštěna jen v případě, kdy je spuštěn také centrální spínač, je nutné vynásobit registry *Zapnuto1*, *Zapnuto2* a *Zapnuto3* hodnotou zapsanou v registru *Central*.

Poté je nutné konvertovat datový typ integer na datový typ boolean a hodnoty zapsat na výstup.

Ve druhé části je pak nutné vytvořit program v Control Webu 2000. Tento program umožní ovládání PLC a fontány pomocí grafických prvků na obrazovce počítače. Výstupem jsou informace předávané registru PLC.



Obr. 22 Panel pro Úlohu č. 2

Panel obsahuje tři spínače a indikátory pro jednotlivá čerpadla a dále jeden centrální spínač s vlastním indikátorem. Hodnoty informující o stavu jednotlivých čerpadel a centrálního spínače jsou zapisovány do kanálů *Zapnuto1*, *Zapnuto2*, *Zapnuto3* a *Central*. Všechny jsou typu boolean.

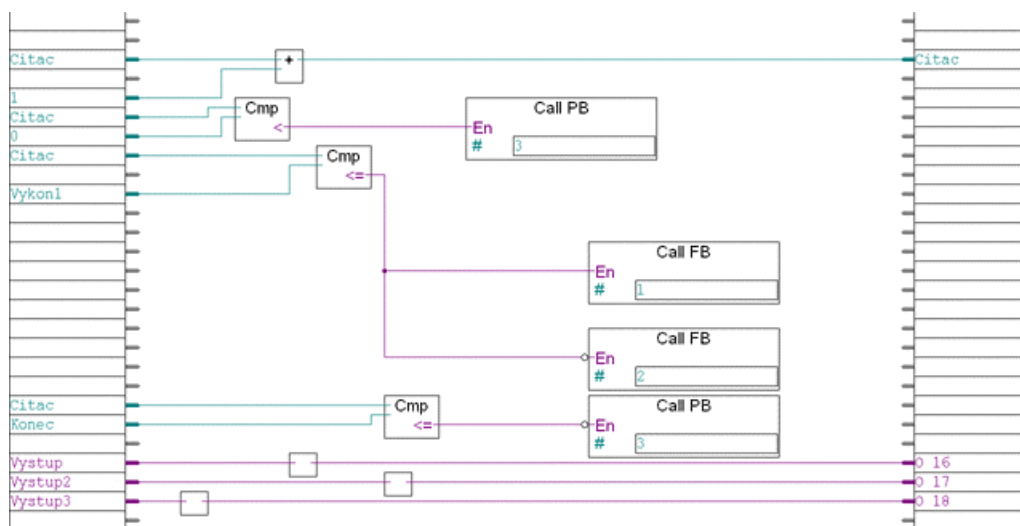
Tyto kanály jsou také vyhodnocovány odpovídajícími indikátory.

1.7.3 Úloha č. 3

Vytvořte program pro ovládání modelu fontány pomocí programovatelného automatu SAIA. Ovládání vizualizujte v programu Control Web 2000. Výkon každého čerpadla musí být možné řídit pomocí ovládacího prvku v programu Control Web 2000.

Vypracování

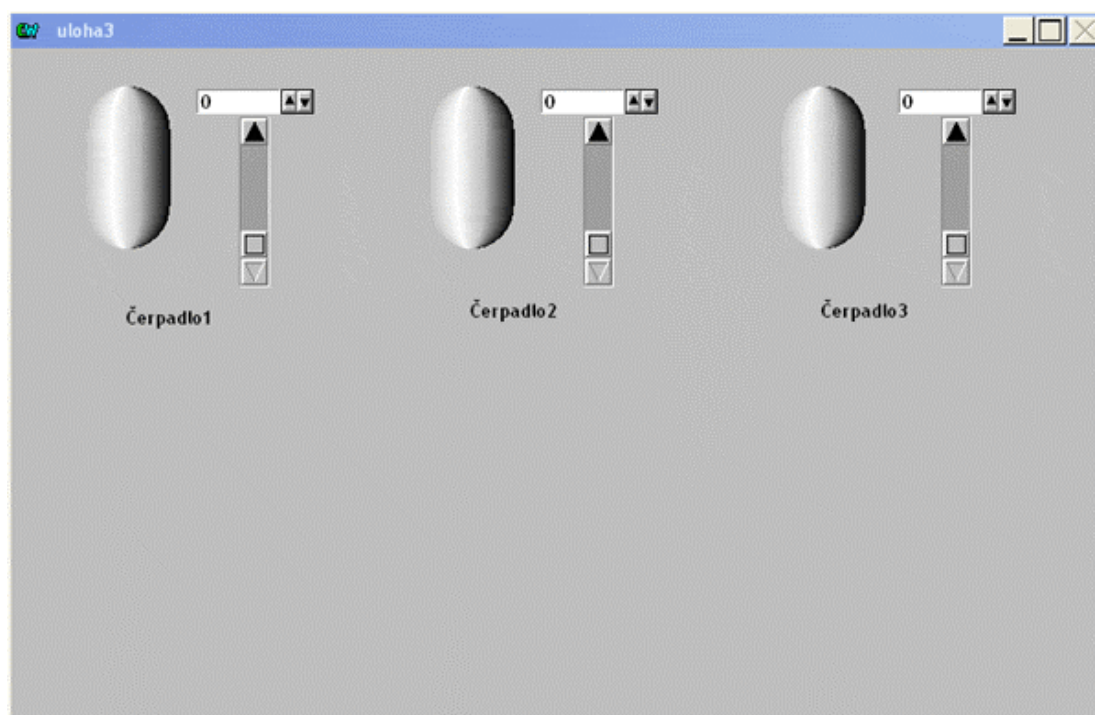
Úloha sestává ze dvou částí – v první části je nutné vytvořit program pro PLC v prostředí PG4. Program zajistí zápis informací z vizualizačního panelu do registru PLC, jejich zpracování a zápis na výstup.



Obr. 23 Hlavní programová smyčka v jazyce FUPA

Aby bylo možné regulovat výkon jednotlivých čerpadel plynule a v požadovaném rozsahu, je nutné vytvořit si tzv. časovou bázi, která definuje, kolik otoček bude trvat náš cyklus. Experimentálně bylo zjištěno, že je výhodné použít bázi s počtem 100 otoček. Tato hodnota je zapsána do proměnné *Konec*. Proměnné *Vykon1*, *Vykon2*, *Vykon3* definují počet otoček, během nichž je čerpadlo zapnuto – na výstupu PLC je zapsána logická 1. Rozdíl hodnot *Konec* a *Vykon1* (resp. *Vykon2*, resp. *Vykon3*) určuje počet otoček, během nichž je čerpadlo vypnuto – na výstup PLC je zapsána logická 0.

Druhou částí úlohy je vytvoření vizualizačního panelu v programu Control Web 2000. Panel musí umožnit vizualizaci ovládání jednotlivých čerpadel a zároveň předává registrům v PLC žádané hodnoty pro řízení čerpadel.



Obr. 24 Panel pro Úlohu č. 3

Ovládání čerpadel je realizováno pomocí virtuálního přístroje *Control*, o indikaci výkonu se stará virtuální přístroj *tank*. Přístroje *Control 1-3* předávají svou hodnotu do PLC pomocí kanálů *Výkon1-3*.

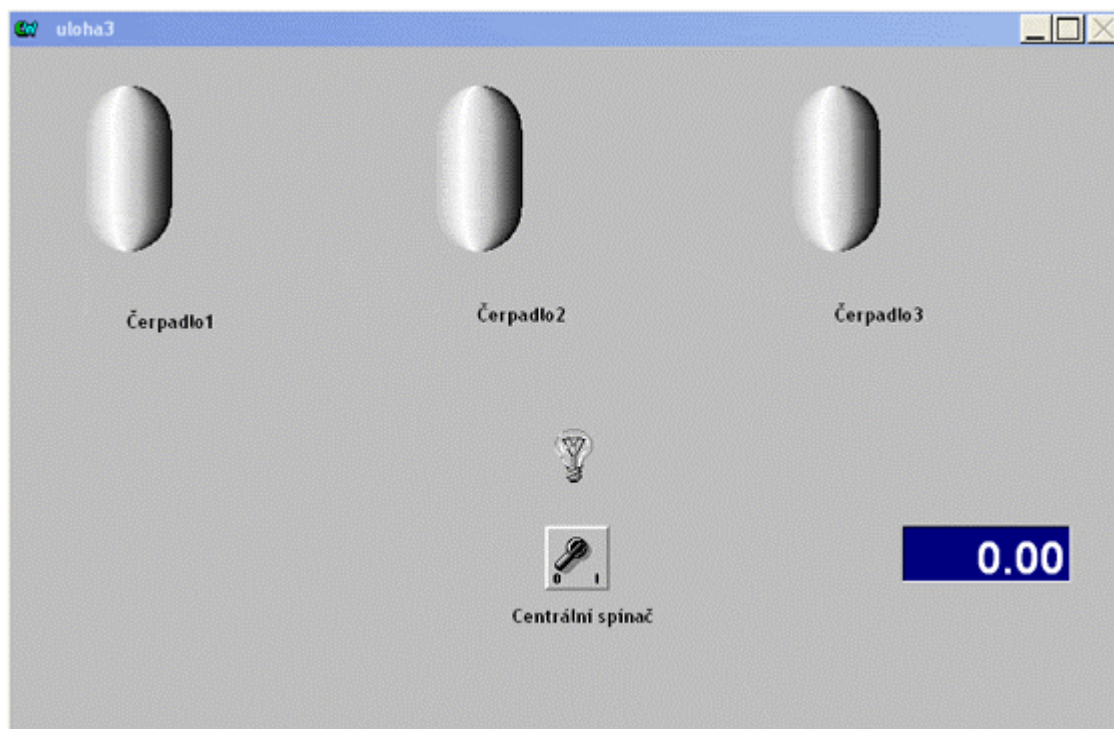
1.7.4 Úloha č. 4

Pomocí programovatelného automatu SAIA a programu Control Web 2000 vytvořte obdobu Křížkovy fontány. Po spuštění programu by se měla zapnout cyklicky se opakující sekvence, kdy se nezávisle na sobě mění výkony všech tří čerpadel. Vytvořte sekvenci tak, aby působila estetickým dojmem.

Vypracování

Při řešení této úlohy lze s výhodou použít program pro PLC vytvořený pro Úlohu č.3. Tento program bude zpracovávat signál doručený do PLC z programu Control Web 2000.

Samotnou sekvenci signálů lze velmi jednoduše naprogramovat v aplikaci Control Web 2000. V podstatě lze použít pouze jeden virtuální přístroj – *switch*. Pro názornost byly ale na panel umístěny také přístroje *tank1-3*, *meter* a *indicator*. Získáme tak lepší přehled o probíhajících operacích.



Obr. 25 Panel pro Úlohu č. 4

Po sepnutí „Centrálního spínače“ je spuštěna cyklicky se opakující sekvence trvající 50 s. Vlastní průběh sekvence je možné naprogramovat pomocí procedury u přístroje *switch*. Následující zdrojový kód popisuje ovládání jednoho čerpadla:

```
procedure OnActivate();
begin
  if Central                / kontrola zapnutí přístroje
  then
    if sekundy=0
    then Vykon1=10; / inicializace hodnoty při startu
    end;
    sekundy=sekundy+1;
    if sekundy>5          / úrovně výkonu v jednotlivých časových úsecích
    then Vykon1=15;
    end;
    if sekundy>10
    then Vykon1=25;
    end;
    if sekundy> 20
    then Vykon1=35;
    end;
    if sekundy>25
    then Vykon1=50;
    end;
    if sekundy>30
    then Vykon1=40;
    end;
    if sekundy>35
    then Vykon1=30;
    end;
    if sekundy>40
    then Vykon1=25;
    end;
    if sekundy>45
    then Vykon1=15;
    end;
    if sekundy=50          / ukončení sekvence
    then sekundy=0;
    end;
  else
    sekundy=0;              / nastavní výkonu při vypnutí přístroje
    Vykon1=0;
    end;
end_procedure;
```

2 ELEKTRONICKÁ OPORA PŘEDMĚTU

Jako elektronická opora pro laboratorní cvičení budou sloužit webové stránky umístěné na adrese www.plc.unas.cz. Stránky mají studentům poskytnout přístup k veškerým materiálům potřebným pro studium předmětu Programovatelné automaty a dále pro absolvování laboratorních cvičení.

Jsou na nich proto k dispozici jak učební texty, tak i zadání laboratorních cvičení včetně návodů k jejich řešení. Stránky zároveň nabízí několik zajímavých elektronických pomůcek.

2.1 Použité technologie

Stránky jsou z velké části statické, proto bylo možné pro jejich použití použít převážně pouze HTML kódu. Pro řešení některých dílčích problémů souvisejících zejména s členěním stránky a načítáním jejího obsahu do požadovaných částí stránky bylo ale nutné použít programovatelný jazyk PHP. Kód PHP je ve zdrojovém textu vždy oddělen od ostatního kódu pomocí znaků „<?php“ a „?>“.

Grafické jednodušství stránky bylo dosaženo použitím CCS kaskádových stylů.

Členění stránky index.php

Jak již bylo zmíněno v kapitole 2.5.1 a 2.5.1, stránka je rozdělena do několika oblastí, přičemž jednotlivé stránky se zobrazují v *Hlavní části*. Toho bylo docíleno tak, že soubor index.php obsahuje pouze jakousi osnovu (kostru) stránky, do které jsou načítány jednotlivé podstránky podle toho, ve které části webu se uživatel aktuálně nachází.

Takového členění je možné dosáhnout použitím příkazu jazyku PHP – *include*, který obecně slouží pro načtení skriptu ze souboru. Pro vložení *záhlaví* a *zápatí* do stránky lze jednoduše použít tento kód `include "/header.php";` resp. `include "/footer.php";`.

Aby bylo zajištěno to, že bude do *Hlavní části* načten obsah požadované stránky, je nutné předcházející zápis mírně modifikovat. Nejprve je potřeba vytvořit a inicializovat pole s názvem *seznam*, které bude obsahovat adresy všech podstránek:

```
$seznam=Array("uvod.php","prednasky.php","laboratore.php","otazky.php","texty.php","  
odkazy.php","kontakt.php","carn1.php","mapa.php");
```

Příkazu *include* potom budeme předávat index odpovídajícího prvku pole:

```
include "$seznam[$page]";
```

Proměnná *\$page* představuje hodnotu indexu pole. Tato je načtena vždy na kliknutí na odkaz s příslušnou podstránkou ve stránce *Panel s nabídkou*.

Příklad předávání hodnoty proměnné *\$page*:

```
<a href="index.php?page=1">Přednášky</a>
```

Kompletní zdrojový kód souboru *index.php* je uveden v Příloze P1.

Jednotlivé stránky

Jednotlivé stránky jsou načítány do *Hlavní části* pomocí předávání hodnoty proměnné *\$page*. Tyto stránky jsou tvořeny pouze pomocí HTML kódu a vzhledem k jejich přehledné struktuře je lze poměrně snadno a rychle editovat a aktualizovat.

Velmi dobře to lze pozorovat na příkladu souboru *news.php*, pomocí nějž lze zveřejnit novou aktualitu editací pouze jednoho řádku textu.

2.2 Aplikace pro vývoj

2.2.1 WWW-server

Pro testování funkčnosti aplikace napsané v jazyce PHP je nutné mít přístup k www-serveru – buďto na internetu nebo offline na lokálním počítači. V případě práce offline je nezbytné mít nainstalován některý z tzv. personal www-serverů. V toto případě byl zvolen server *Apache*. Jeho výhodou je dostačující výkon a také to, že jde o freeware a lze jej používat zcela zdarma.

2.2.2 Jazyk PHP

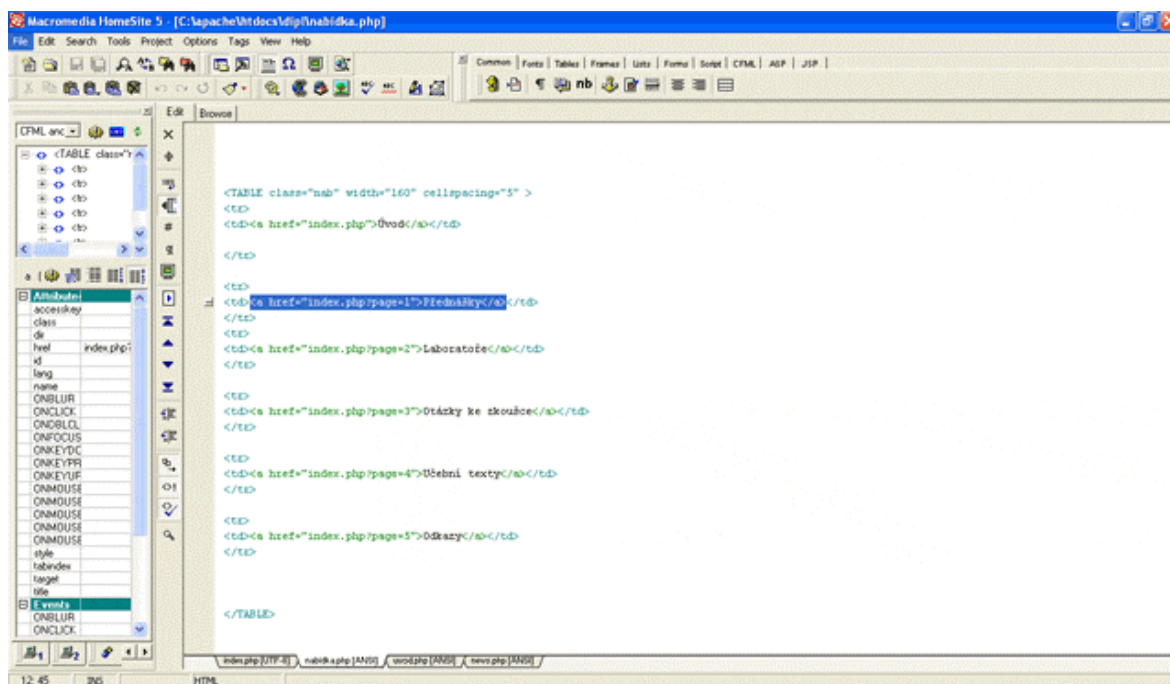
Pro programování v jazyce PHP, resp. pro interpretaci zdrojového kódu je nutné mít v systému nainstalovánu jeho podporu. Program lze rovněž používat zcela zdarma a je k dispozici například na internetové adrese <http://www.php.net>.

Pro správné fungování je nutné ještě upravit inicializační soubor *php3.ini* a nastavit podporu PHP v konfiguračním souboru WWW-serveru *srm.conf*.

2.2.3 HTML editor

V zásadě je možné veškerý zdrojový kód psát v jakémkoliv textovém editoru. Bez problémů si lze vystačit i se standardním poznámkovým blokem. Nicméně pro pohodlnější práci je výhodnější použít některý ze specializovaných HTML editorů, které nabízí spoustu užitečných funkcí a vlastní práci mohou výrazně usnadnit.

Při práci na webových stránkách pro předmět Programovatelné automaty byl použit program *Macromedia HomeSite* verze 5.



Obr. 26: Editor Homesite

Hlavní část slouží pro psaní zdrojového kódu. Přitom je možné použít nástroje umístěné v záhlaví, případně v levé části obrazovky. Mezi nástroji lze najít například zrychlené vkládání nejčastěji používaných částí HTML kódů, odkazů, tabulek obrázků a podobně.

2.3 Struktura a členění stránek

Při tvorbě stránek bylo nutné kromě vlastního obsahu dbát také na jeho logické členění a srozumitelnou strukturu. Tomu bylo přizpůsobeno rozvržení stránek a umístění všech navigačních prvků. Byl zvolen spíše jednoduchý vzhled, umožňující rychlé nalezení požadovaného obsahu.

Jednotlivé stránky jsou členěny na tyto části:

- *Záhlaví*

Nachází se v horní části stránky a obsahuje pouze logo fakulty. V budoucnu je možné jej upravit podle potřeby přidat například vyhledávání, okno pro přihlášení a podobně.

- *Panel s nabídkou*

Panel je umístěn v levé části stránky a představuje hlavní navigační prvek. Společně se záhlavím a zápatím je součástí každé stránky a obsahuje odkazy na jednotlivé podstránky webu.

- *Hlavní část*

Nejdůležitější část stránky, umístěná doprostřed obrazovky a zabírající nejvíce prostoru. V této části je zobrazován vlastní obsah jednotlivých podstránek, na které se lze dostat prostřednictvím panelu s nabídkou.

- *Zápatí*

Umístěno ve spodní části všech stránek. Dotváří grafický celek stránky a obsahuje některé důležité odkazy, např. *Kontakt* – odkazuje na stránku s kontaktními údaji vedoucího předmětu a *email* – přesměruje uživatele do emailového klienta a umožňuje odeslat email správci webu.

- *Panel Aktuality*

Tento panel je umístěn pouze na hlavní stránce (*Úvod*), v její pravé části. Pomocí *Aktualit* je možné informovat studentům rychle a snadno předávat důležité informace, které souvisí s předmětem.



Obr. 27: Úvodní stránka

2.4 Obsah

Obsah stránek koresponduje s členěním zobrazeným v panelu s nabídkou a je rozdělen do těchto částí:

- *Úvod*

Úvod je zobrazen při každém příchodu na stránku. Obsahuje základní údaje o předmětu, případně informuje o aktuálních novinkách důležitých pro posluchače předmětu. Velkou část stránky zabírá ilustrační obrázek charakterizující předmět Programovatelné automaty. Obrázků je k dispozici několik a jsou dynamicky měněny s každým načtením stránky.

- *Přednášky*

Kompletní přehled přednášek přednášených během semestru. Přednášky jsou uloženy jako prezentace programu MS PowerPoint a studenti si je mohou dle potřeby stáhnout na svůj počítač.



Obr. 28: Přednášky

- Laboratoře

Ke každé laboratorní úloze je možné zde najít zadání, ukázkové vypracování úlohy, případně další materiály – pomůcky, fotografie atd.



Obr. 29: Laboratorní úlohy

- Otázky ke zkoušce

V této sekci lze nalézt otázky ke zkoušce a podmínky k získání zápočtu.



Obr. 30: Otázky ke zkoušce

- Učební texty

Studijní materiály k předmětu. Jsou zde k dispozici materiály o programovatelných automatech obecně, o vizualizačním software obecně a dále pak popis jednotlivých PLC a programů dělených dle výrobců.

Samostatnou část tvoří aktivní aplikace ke stažení – program pro minimalizaci Carnaughovy mapy a pro demonstraci uložení čísla ve formátu float.



Obr. 31: Učební texty

- Odkazy

Byly vybrány užitečné odkazy související s předmětem - odkazy na výrobce programovatelných automatů, vizualizačního software. Poslední skupinu odkazů tvoří linky na vybrané stránky univerzity.



Obr. 32: Odkazy

ZÁVĚR

Cílem této diplomové práce bylo připojit zadaný reálný model zařízení k programovatelnému automatu, zajistit jeho funkčnost a připravit vše pro to, aby na zařízení mohly být vedeny laboratorní úlohy předmětu Programovatelné automaty.

Přes prvotní potíže, způsobené zejména technickým stavem reálného modelu, se tohoto cíle podařilo dosáhnout.

Model fontány lze ovládat buďto manuálně pomocí připojené ovládací skříně nebo pomocí PLC SAIA. Programy do programovatelného automatu jsou nahrávány z osobního počítače přes sériovou linku. Ovládání je vizualizováno na obrazovce počítače pomocí programu Control Web 2000.

Pro model byly vytvořeny laboratorní úlohy různých úrovní obtížnosti. Díky tomu si mohou studenti osvojit v laboratorních cvičeních základy programování PLC a naučit se také pokročilejším programovacím postupům.

Veškeré materiály potřebné pro studium předmětu a pro absolvování laboratorních cvičení bude možné najít také na nově vytvořené webové stránce na adrese www.plc.unas.cz. Kromě učebních textů a zadání laboratorních úloh jsou na nich k dispozici také velmi užitečné aktivní aplikace.

Do budoucna je samozřejmě možné připravit velké množství dalších laboratorních úloh pro zvolený reálný model. Lze modifikovat současné zadání, případně zvolit zcela nové. Velmi zajímavou variantou by jistě bylo vytvořit program ovládající chod fontány podle hrající hudební skladby.

Rovněž webové stránky je možné dál postupně vylepšovat a přidávat do nich dle potřeby další materiály. Jejich struktura byla navržena tak, aby bylo její doplňování a aktualizace co nejjednodušší.

SEZNAM POUŽITÝCH ZDROJŮ

- [1] Martinásková, M., Šmejkal, L.: Řízení programovatelnými automaty, Vydavatelství ČVUT, Praha 1998
- [2] Martinásková, M., Šmejkal, L.: Řízení programovatelnými automaty II, Vydavatelství ČVUT, Praha 2000
- [3] Šmejkal, L., Martinásková, M.: PLC a automatizace, Nakladatelství BEN – Technická literatura, Praha 1999
- [4] SAIA-Burgess Electronics: Hardware – Řady PCD1 a PCD2. Vydání 26/737 C5, 1998
- [5] Hornych, J.: Aplikace PLC automatů pro řízení enzymatické hydrolýzy koželužských odpadů, Diplomová práce VUT FT, Zlín 1999
- [6] Bílý, R., Cagaš, P., Cagaš, R., Hladůvka, D., Kolařík, M., Sobotík, J., Zálešák, M., Zgarba, Z.: Control Web 2000, Computer press, Praha 1999
- [7] Lukačovič, P.: Počítačové řízení fontány, Bakalářská práce FT UTB, Zlín 2004
- [8] Kosek, J., : HTML Tvorba dokonalých WWW stránek, Grada Publishing, Praha 1998
- [9] Broža, P., Buranský, I.: Programování WWW stránek pro úplné začátečníky, Computer Press, Brno 2003
- [10] Kosek, J.,: PHP Tvorba interaktivních internetových aplikací, Grada Publishing, Praha 1999

SEZNAM POUŽITÝCH SYMBOLŮ, ZKRATEK A TERMÍNŮ

T_{vz}	vzorkovací perioda
U	akční zásah
A/D	analogově / digitální
D/A	digitálně / analogový
FBox	(Function Box) Obdélník, reprezentující v jazyce FUPLA funkci
FUPLA	Grafický programovací jazyk
GRAFTEC	Grafický sekvenční programovací jazyk
IL	Instruction List – forma zápisu programu jako výpis instrukcí
I/O	Vstupně / výstupní
LED	Svítivá dioda
PCD	Process Control Device – programovatelné automaty fy SAIA
piggyback	Modul pro nasunutí na základní desku PLC k rozšíření jejích funkcí
PLC	Programmable Logic Controller – programovatelný automat
PROFIBUS	Vyšší komunikační sběrníkový protokol
RS232,422,485	Standardy sériového rozhraní
TTL	Standard binárního signálu
Watchdog	Obvod v PLC, rozpínající v případě přerušení běhu programu kontakty relé pro provedení nouzové akce (např. vypnutí napájení řízeného zařízení)
WWW	World Wide Web - internetová služba umožňující propojení stránek pomocí odkazů
HTML	HyperText Markup Language – jazyk pro tvorbu WWW stránek
Link	Odkaz
PHP	Personal Home Page Construction Kit – programovací jazyk
OSF	Ovládací skříň Fontána

SEZNAM OBRÁZKŮ

Obr. 1 Schéma zapojení	7
Obr. 2: RS-232 propojení.....	7
Obr. 3: Model fontány	8
Obr. 4 Čerpadlo 22 l	9
Obr. 5 Čerpadlo 12 l	10
Obr. 6 Ovládací skříň.....	11
Obr. 7 Schéma regulačního obvodu.....	12
Obr. 8 Pulzně šířková modulace	14
Obr. 9 Programovatelný automat SAIA PCD2.....	19
Obr. 10 Hlavní deska SAIA PCD2	20
Obr. 11 Okno knihovny projektů.....	24
Obr. 12 Okno správce projektu.....	25
Obr. 13 Výběr typu programového objektu.....	26
Obr. 14 Okno editoru FUPLA	27
Obr. 15 Virtuální přístroje v okně Paleta přístrojů	30
Obr. 16 Inspektor virtuálního přístroje	31
Obr. 17 Textový editor.....	31
Obr. 18 Propojení aplikace s v/v zařízením.....	34
Obr. 19 Program v jazyce FUPLA Úlohu č.1	36
Obr. 20 Panel pro Úlohu č. 1	37
Obr. 21 Program v jazyce FUPLA Úlohu č.2.....	38
Obr. 22 Panel pro Úlohu č. 2	39
Obr. 23 Hlavní programová smyčka v jazyce FUPA	40
Obr. 24 Panel pro Úlohu č. 3	41
Obr. 25 Panel pro Úlohu č. 4	42
Obr. 26: Editor Homesite.....	47
Obr. 27: Úvodní stránka.....	48
Obr. 28: Přednášky	49
Obr. 29: Laboratorní úlohy	50
Obr. 30: Otázky ke zkoušce.....	50
Obr. 31: Učební texty.....	51
Obr. 32: Odkazy.....	51

SEZNAM TABULEK

Tabulka I. Přehled datových typů prostředků programovatelného automatu SAIA	22
Tabulka II. Přehled typů programových bloků	23

SEZNAM PŘÍLOH

- P 1 : Zdrojový kód souboru index.php
- P 2 : Mapovací soubor saia.dmf
- P 3 : Parametrický soubor saia.par
- P 4 : Schéma zapojení OSF

```
<?php
```

```
$seznam=Array("uvod.php","prednasky.php","laboratore.php","otazky.php","texty.php","odkazy.php","kontakt.php","carn1.php","mapa.php");  
if ($page<=0) $page=0;
```

```
$obrazky=Array("kolaz.gif","omron.gif");
```

```
$cislo=Rand(0,1);
```

```
?>
```

```
<?php
```

```
include "../header.php";
```

```
?>
```

```
<br>
```

```
<!-- obsah hlavni stranky - zacatek //-->
```

```
<table width="760" border="0" align="center" cellpadding="0" height="90%">
```

```
<!-- zacatek leveho sloupce//-->
```

```
<tr><td valign="top">
```

```
<?php
```

```
include "../nabidka.php";
```

```
?>
```

```
</td>
```

```
<!-- konec leveho sloupce//-->
```

```
<!-- zacatek stredniho slouce//-->
```

```
<td valign="top" >
```

```
<?php
```

```
include "$seznam[$page]";
```

```
?>
```

```
</td>
```

```
<!-- konec stredniho slouce//-->
```

```
<!-- zacatek praveho slouce//-->
```

```
<?php if ($page<=0){
```

```
echo "<td width=150 valign='top' align='right'>";
```

```
echo "<table class='nab'><tr>";
```

```
echo "<td align='center' class='text'><b>Novinky</b>";
```

```
echo "<br><br><br>";
```

```
include "../news.php";
```

```
echo "</td></tr>";
```

```
echo "</table>";
```

```
echo "</td>";
```

```
}
```

```
?>
<!-- konec praveho slouce/-->
</tr>
</table>
<!-- obsah hlavni stranky - konec //-->

<!-- footer - zacatek/-->
<?php
include "./footer.php";
?>
<!-- footer - konec/-->
```

```
begin
  1 real input
  10 real input
  11 real input
  12 string input
  13 real input
  14 real input
  15 real input
  16 real input
  17 boolean output
  18 boolean output
  20 real input
  21 real input
  22 string input
  23 real input
  24 real input
  25 real input
  26 real input
  27 boolean output
  28 boolean output
  100 - 119 real bidirectional
  200 - 263 boolean bidirectional
  900 - 917 real bidirectional
  1000 - 1007 boolean bidirectional
  1100 - 1107 boolean input
  1200 - 1207 boolean input
end.
```

[ComPort]

Com = com2

Protocol = p800

SBusMode = parity

BaudRate = 9600

Timeout = 200

RTS = enable

[Channels]

;Block = Station, ChannelFrom, ChannelTo, Media, Offset, [Subtype], [Bidirect]

Block = 01, 100, 100, R, 0, int8, bidirect

Block = 01, 100, 100, R, 5, int8, bidirect

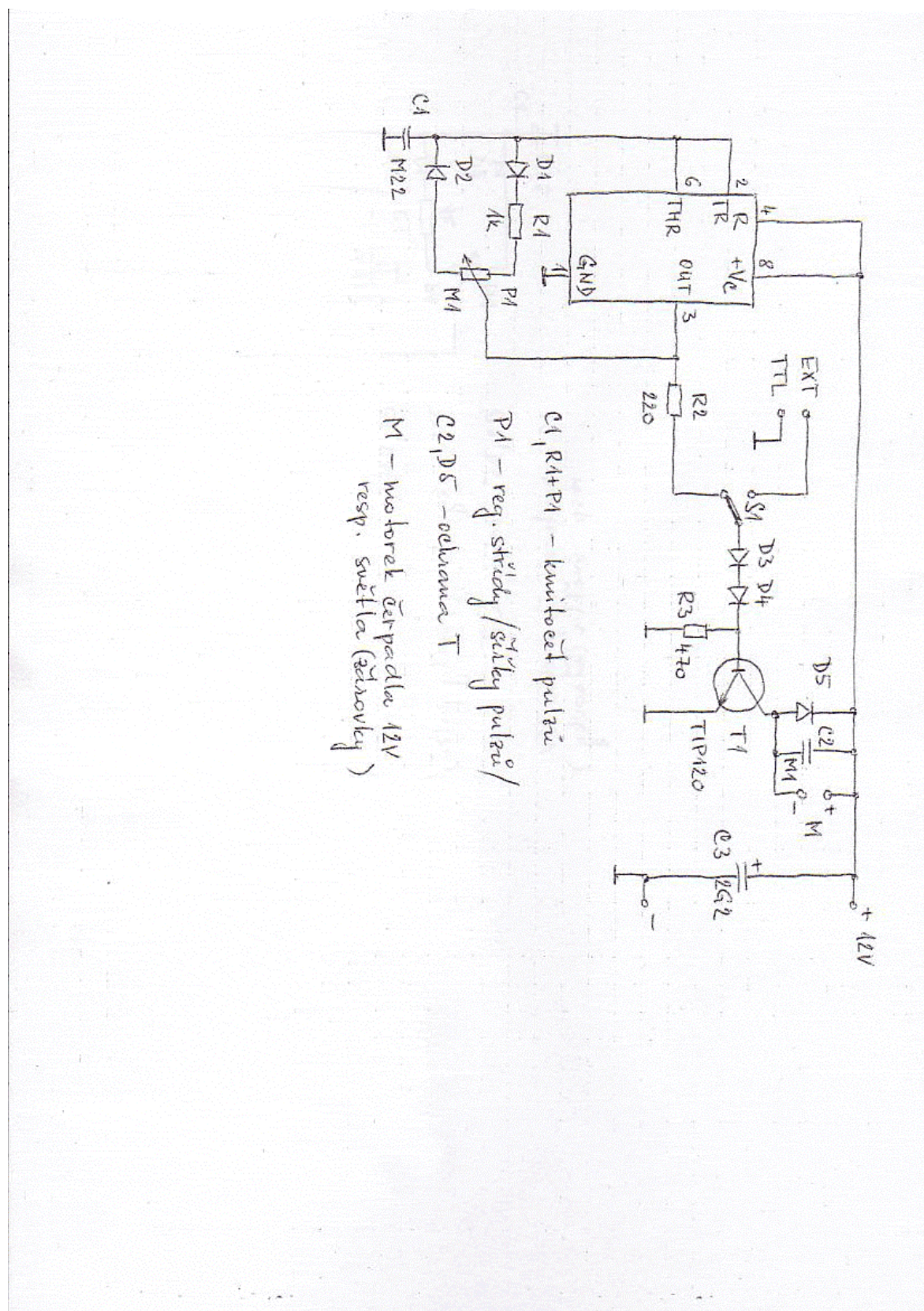
Block = 01, 100, 100, R, 6, int8, bidirect

Block = 01, 200, 200, R, 1, bits, bidirect

Block = 01, 201, 201, R, 2, bits, bidirect

Block = 01, 202, 202, R, 3, bits, bidirect

Block = 01, 203, 203, R, 4, bits, bidirect



Příloha P4: Schéma zapojení OSF