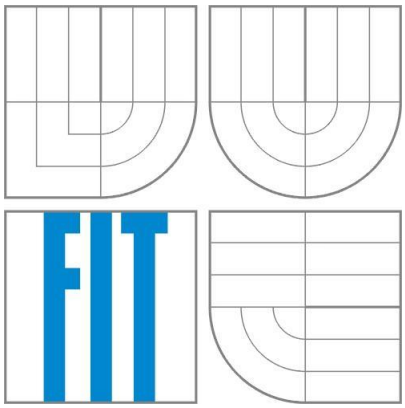
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



Dokumentace k projektu do předmětů IMP

SIMULACE V CW: SVĚTELNÉ NOVINY

Sára Škutová xskuto00@stud.fit.vutbr.cz
xskuto00

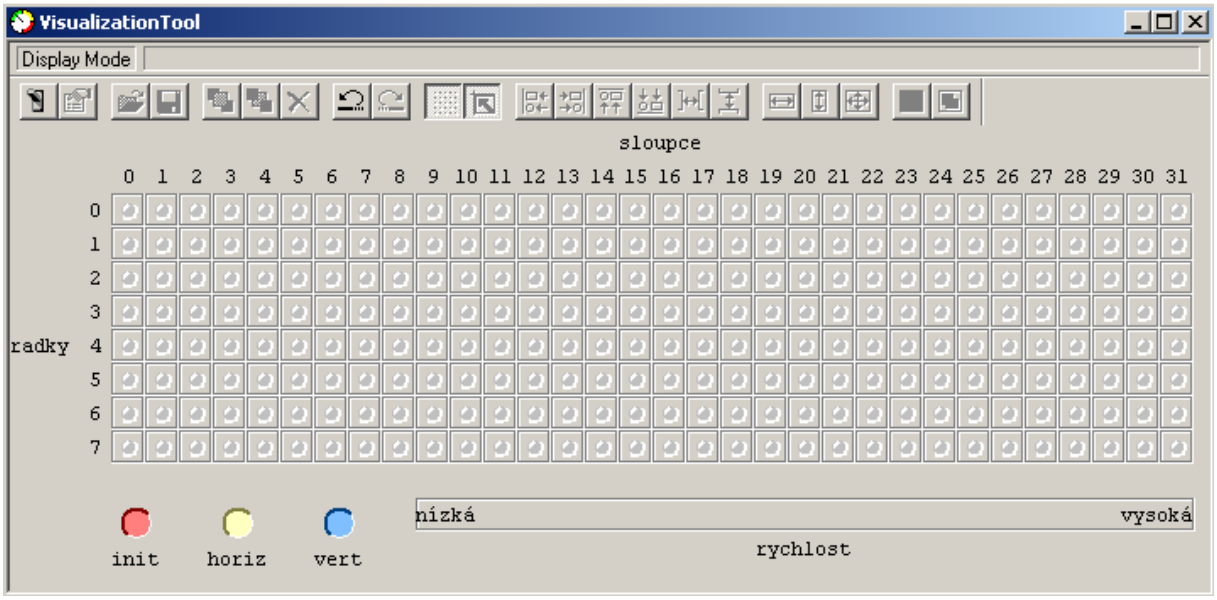
V Brně 15. 12. 2014

Obsah

1. Zadání projektu

Cílem bylo naprogramovat pro True-Time Simulator aplikaci Světelné noviny určenou pro zobrazování textu na displeji složeného z LED diod. Textem v tomto případě je myšleno školní login autora.

Uživatel bude aplikaci ovládat pomocí prvků nástroje Visualization Tool, jejich konfigurace byla ponechána na autorovi (více v části Implementace).



Obrázek 1: Rozložení prvků nástroje Visualization Tool

2. Popis funkcionality jednotlivých prvků

2.1. LED diody

Uspořádané do matice o 8 řádcích a 32 sloupcích. Slouží k zobrazování loginu uloženého v paměti mikrokontroléru. Každý ze znaků je reprezentován bitmapou v rastru o velikosti 8 řádků a 8 sloupců, přičemž nejlevější a nejpravější sloupec plní funkci mezer. Login je ve formě malých písmen.

2.2. Tlačítko init

Po stisku tlačítka init se aplikace dostane do počátečního stavu, ve kterém jsou na dispeji viditelné 4 nejlevější znaky loginu.

2.3. Tlačítko horiz

Po stisku tlačítka horiz bude na displeji rotovat login autora ve směru zprava doleva.[1]

2.4. Tlačítko vert

Po stisku tlačítka vert bude na displeji rotovat login autora ve směru shora dolů.[2]

2.5. Rychlost rotace

Rychlost rotace je měnitelná pomocí táhla rychlosti.

3. Implementace

Aplikaci bylo možné implementovat pomocí programovacích jazyků C nebo asembler přičemž cílovou platformou byl mikrokontrolér HC(S)08. Já jsem si vybrala k implementací převážně jazyk C, ale v jedné funkci jsem použila vnořený asembler. Popis jednotlivých funkcí bude následovat. Použité proměnné a konstanty budou zhodnoceny v oddílu Analýzy paměťových nároků.

3.1. Konfigurace prvků nástroje Visualization Tool

Pro správnou funkcionalitu aplikace bylo nutno jednotlivé prvky správně namapovat. Já jsem jednotlivé prvky namapovala na dané adresy v paměti a pak jsem k nim přiřadila potřebné proměnné. Jejich detailnější popis naleznete v oddílu Analýzy paměťových nároků.

3.1.1. LED

Matice LED byla namapováná od adresy[3] 0x0200 a mapovala jsem po sloupcích (původní mapování o 0x100 se ukázalo problematické). K této adrese bylo následně přiřazeno pole LED, jež obsahuje login autora. Za zmínku stojí, že jsem jednotlivé diody mapovala po bitech, čímž jsem zmenšila paměťové nároky aplikace.

[1] Login autora je xskuto00, 00 znamená rotování ve směru zprava doleva, jiná čísla představují rotování zleva doprava

[2] Login autora je xskuto00, 00 nebo 01 znamená rotování ve směru shora dolů, jiná čísla představují rotování zdola nahoru

[3] Všechny adresy jsou v šestnáctkové formě

3.1.2. Tlačítka init, horiz a vert

Byly postupně namapovány na tyto adresy 0x250, 0x251 a 0x252. Tyto adresy reprezentují proměnné init, horiz a vertik.

3.1.3. Rychlost rotace

Táhlo rychlosti je namapováno na adresu 0x253. Reprezentováno proměnnou rychlost. Maximální hodnota táhla je nastavená na 100. Při spuštění aplikace je táhlo nastaveno na 50 (čili 50% maximální rychlosti rotace).

3.2. Zobrazení loginu a reakce na tlačítko init

Implementováno jako funkce led_init. V této funkci se jednotlivé indexy pole LED (sloupce matice LED diod) inicializují potřebnou bitovou hodnotou. Po dokončení inicializace je v paměti celý login autora a na displeji jsou zobrazeny první čtyři znaky.

3.3. Horizontální rotace / reakce na tlačítko horiz

Využívá funkci led_horiz. V této funkci se všechny indexy pole LED přepisují svými následujícími indexy (např. LED[0] = LED[1]….), až na poslední index, který se přepíše hodnotou na indexu 0. Dalším polem, které se tady objevuje je LED_help, jež je využíváno při vertikální rotaci a pro její správnou funkcionalitu se také musí podrobit horizontální rotaci.

3.4. Vertikální rotace / reakce na tlačítko vert

Představuje funkci led_vertik. Jelikož v jazyce C nelze provádět bitovou rotaci využila jsem možnost vložit do kódu kousky asembleru, pro lepší pochopení následuje kus daného kódu:

```
/*Každý sloupec binarně rotuji, jelikož rotace přechází přes carry musím přenastavovat carry*/
/*Velice doufam, ze se nic nezkazi*/
 for (i = 0; i < 64; i++) {
     pomoc = LED[i];

     if (LED_help[i] == 0)  /*Pomocné pole dle něhož nastavuju příznak carry*/
         __asm CLC;         /*C = 0*/
     else
         __asm SEC;         /*C = 1*/

     asm {
         ROL pomoc        /*Rotace v daném sloupci*/

         BCC nula         /*Testování příznaku carry na hodnotu 0*/

         MOV #1, $80      /*Vložení hodnoty 1 na adresu 80, na adrese 80 je pomocná proměnná pomoc_carry*/
         JMP konec_carry

         nula:
           MOV #0, $80 /*Vložení hodnoty 0 na adresu 80, na adrese 80 je pomocná proměnná pomoc_carry*/

         konec_carry:
     }


    LED_help[i] = pomoc_carry;  /*vložení proměnné s hodnotou carry na daný index*/
    LED[i] = pomoc;             /*sloupec po binární rotaci*/

 }
```

Obrázek 2: Algoritmus vertikální rotace s vnořeným asemblerem

3.4.1. Dodatečný komentář k algoritmu vertikální rotace

K vertikální rotaci využívám instrukci ROL, která nejvíce významný bit vloží do příznaku carry, celou posloupnost bitu posune a na nejméně významný bit vloží hodnotu z carry. Jelikož se daný příznak neustále mění, ukladám si jeho hodnoty do pomocného pole LED_help dle něhož pak při dalším voláním funkce led_vertik nastavuji příznak carry. Před spuštění aplikace a při stisknutí tlačítka init je pole LED_help vynulováno. Za zmínku stojí, že se vertikální rotaci podrobuje i neviditelná část loginu.

3.5. Rychlost rotace

Zrychlení/zpomalení rotace simuluje funkce led_delay. Funkce ta zpomaluje rychlost provádění pomoci cyklu, jenž používá hodnotu s táhla rychlosti.

3.6. Testy na stisknuté tlačítka

Jelikož momentální verze simulátoru již nepodporuje vyvolávání přerušení, musí se pro testování stisku tlačítka použít technika jménem polling, což označuje neustále dotazování na stav tlačítka. Při detekci stisku se nastaví pomocná proměnná, jenž zajistí, že se požadovaná operace bude provádět do detekce jiného tlačítka[4]. Testování probíhá v hlavní programové smyčce.

4. Analýza paměťových nároků

4.1. Použité proměnné, jejich typ a význam hodnot

4.1.1. Globální proměnné

LED: pole typu byte[5] o velikosti 64, uchovává se tam login autora, používá se při vypisování na displej, pevně přiřazeno k adrese 0x0200.

LED_help: pomocné pole typu unsigned char o velikosti 64, používá se jako pomoc u vertikální rotace, uchovávají se tam hodnoty příznaku carry.

pomoc_carry: pomocná proměnná typu byte jejíž adresa je používána při přiřazování v úseku napsaném v asembleru (u vertikální rotace), přiřazena k adrese 0x00B0.

[4] Toto pouze u rotování, při stisku init se funkce led_init provede pouze jednou, pak se čeká na stisk dalšího tlačítka

[5] Typ byte je ve skutečnosti další typ vytvořený pomocí typedef a typu unsigned char

init, horiz, vertik: proměnné typu byte, postupně přiřazené k adresám 0x0250, 0x0251 a 0x0252, hodnota jiná než nula, indikuje stisknutí tlačítka

rychlost: přiřazená k adrese 0x0253, proměnná typu byte, uchovává v sobě hodnotu nastavenou na táhlu rychlosti.

pom_init, pom_horiz, pom_vertik: pomocné proměnné typu int, které se nastavuji při detekci stisku daného tlačítka.

4.1.2. Lokální proměnné

V této aplikaci se lokální proměnné používají pouze jako podpora u cyklu nebo jako dočasné uložiště hodnot.

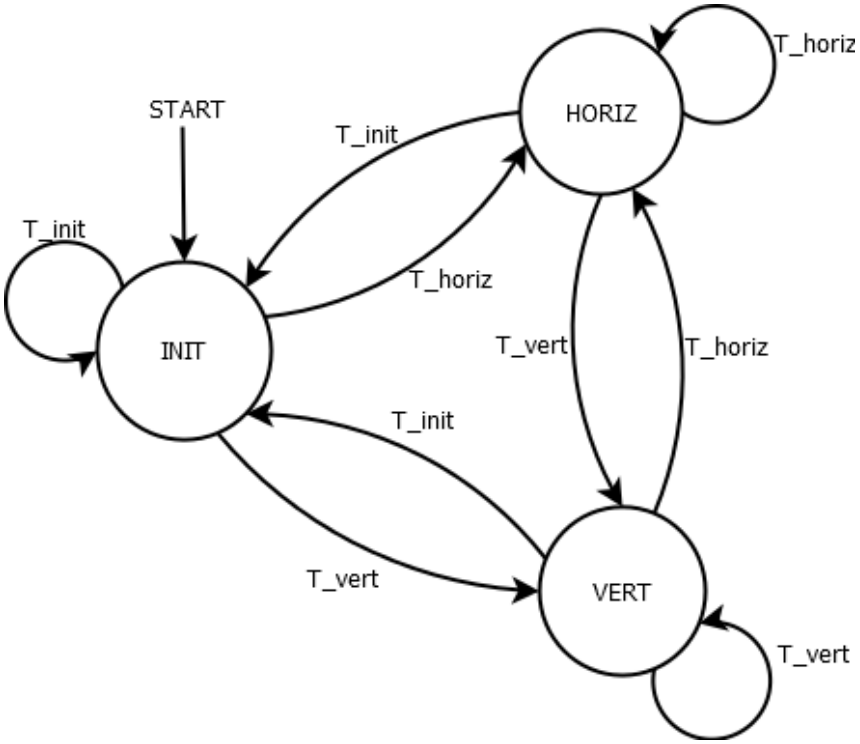
4.2. Metriky kódu

4.2.1. Soubor main.c

Nároky na RAM: 139B
Nároky na Flash: 581B
Velikost kódu: 250 řádků

5. Slovní a grafický popis činnosti aplikace

Při spuštění aplikace se inicializují potřebné proměnné, táhlo rychlostí se nastaví na 50% své maximální velikosti a na obrazovce se zobrazí 4 počáteční znaky loginu autora. Aplikace následně testuje stisk jednotlivých tlačítek. Při stisku init se aplikace vrátí do původního stavu. Tlačítko horiz vyvolá horizontální rotaci loginu a tlačítko vert rotaci vertikální. Pomocí táhla rychlosti je možné ovládat rychlost rotací. Následující strana obsahuje stavový diagram.



Obrázek 3: Stavový diagram aplikace[6]

6. Známe problémy

Špatné testování stisku tlačítek a realizace zrychlení/zpomalení způsobují, že aplikace někdy nereaguje na stisk tlačítek. Je to způsobeno tím, že test stisku je pouze v jednom místě programu a pokud nedojde v dané chvíli ve stisku tlačítka, aplikace to nezaznamená. Problém je možno vyřešit: a) změnit tlačítka na přepínače, b) po stisknutí tlačítko chvíli podržet, c) vytvořit lepší algoritmus.

7. Závěr

Ačkoliv hardwarové předměty nemám příliv v oblibě a tento projekt se mi v polovině semestru zdál neřešitelný, po shlédnutí videa popisujícího tuto aplikaci a přečtení studentského fóra fituška, se všechno objasnilo a práce na projektu mě začala i bavit. Jenom namapovat všechny ty LED diody bylo trochu na zabití.

[6] T_init – stisk tlačítka init, T_horiz – stisk horiz, T_vert – stist tlačítka vert