Vysoké učení technické v Brně
Fakulta informačních technologií

Mikroprocesorové a vestavěné systémy

Dokumentace k projektu - Světelná tabule

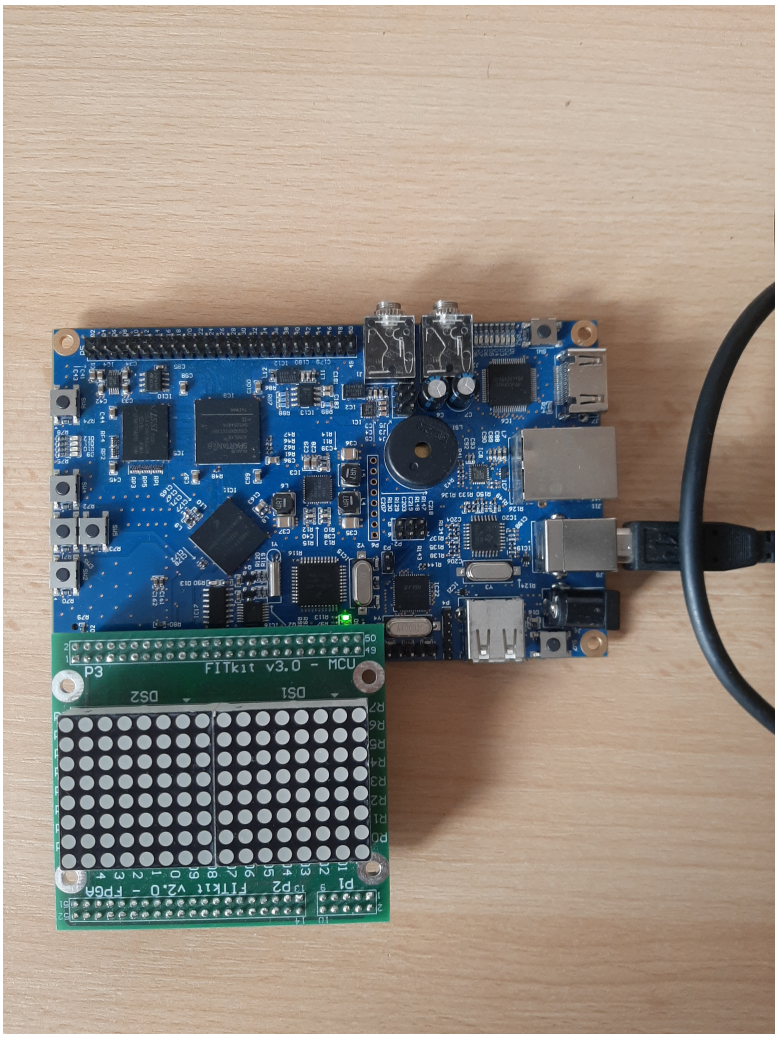
28. listopadu 2022

David Kocman, xkocma08

1 Úvod

Cílem projektu je naprogramovat běžící text na platformě FitKit3 Minerva s použitím přídavné maticové obrazovky. Minerva využívá mikrokontroler Kinetis K60 s jádrem ARM Cortex-M4. Projekt byl vypracován v prostředí Kinetis Design Studio bez využití jakýchkoli dalších knihoven kromě standardního hlavičkového souboru pro vybraný mikroprocesor.

Součástky se propojí přes pin P3 FitKitu3 s pinem P1 světelné tabule. Propojení s počítačem pro „nahrávání“ kódu je docíleno kabelem USB-B.



Obrázek 1: Zapojení FitKit3 s přídavnou maticovou obrazovkou.

Pro vypracování stačí pracovat s portem A a E mikrokontroleru. Port A slouží pro zapínání a vypínání řádků a výběr sloupců. Port E slouží pro nastavování signálu EN pro zapínání a vypínání sloupců a ovládání tlačítek. Při práci jsem se řídil dokumentací mikrokontroléru [1].

Registry využity v projektu:

- **PDDR**: registr byl využíván na nastavování pinů portu A a E jako output.
- **PCRn**: registry slouží na nastavení pinů portu A a E pro GPIO funkcionalitu a pro

vyvolání přerušení tlačítky.

- **PDOR**: slouží pro nastavení output hodnot obou portů.
- **ISFR**: registr slouží pro detekci přerušení při zmáčknutí tlačítka.
- **PDIR**: slouží pro potvrzení přerušení, resp. jestli se nejedná jen o zákmyt.
- **PSOR**: slouží pro opakované nastavení output hodnot output registru.
- **PCOR**: slouží k vyčištění output registru.

Pro projekt byla využita knihovna `MK60D10.h`, která poskytuje standardní rozhraní pro práci s FitKitem3. Využity byly struktury odpovídající výše zmíněným portům a registrům.

Hlavní logikou fungování světelné tabule je rychlé zapínání a vypínání určité kombinace řádků a sloupců pro vypsání konkrétního písmene. Pro vypsání tedy slova musí být písmena adekvátně rozmístěny a v cyklu všechny postupně vykresleny na obrazovku zprava doleva. K implementaci níže.

2 Implementace

Zaměříme se na hlavní funkce programu.

2.1 main

Hlavní smyčka programu, kde na začátku se nakonfiguruje nastavení mikrokontoleru (nastavení portů, přerušení, atd.) a posléze se vstoupí do nekonečné smyčky, kde se čeká na přerušení, při kterém se vypíše text.

2.2 PORTE_IRQHandler

Tahle funkce nám zaručuje vypsání jakéhokoliv slova. Po stisknutí jednoho z 5 tlačítek na FitKitu3 (SW2 až SW6) se program přesune do této funkce. Podle toho, které tlačítko bylo stisknuto, se vypíše adekvátní věta. Zde je potřeba regitrů ISFR a PDIR, se kterýma se porovnává konkrétní maska tlačítka `SWn_MASK`. Po vypsání slova se vynuluje příznak přerušení a program přesune zpět do `main`u a čeká na další přerušení.

Vypsání slova zajišťuje cyklus, který má maximálně `16 + délka_slova` iterací z toho důvodu, aby text po přejetí „zmizel“. V tomto cyklu je i další zanožený cyklus, který zajišťuje to, aby mohlo svítit více sloupců naráz.

Vypsání samotných písmen je řešeno funkcemi `printn`, kde `n` je šířka písmene. Implementace je popsána níže. Příklad vypsání písmene „I“:

```
print3(i, delay, 0x24000000, 0x3F000280, 0x24000000);
```

2.3 printn() funkce

Jak již bylo zmíněno výše, `n` určuje šířku písmena. V programu tohle`n` nabývá hodnot 3,4,5.

Základem jsou podmínky, které určují na jaké pozici se má vykreslit konkrétní část písmene. Začíná na pozici `i`, které udává aktuální pozici v iteraci ve funkci `PORTE_IRQHandler`. Samozřejmě se k této proměnné počítá odsazení písmene od aktuální pozice v iteraci.

```
if(i >= 0 && i < 16)
```

Podmínka výše udává, že se jedná o vykreslení první části písmene. Porovnávání s 0 a 16 docílí efektu, že sloupce nebudou „přetékat“.

V samotném těle podmínek se opakuje následující logika: vyčistit output registr, nastavit požadované hodnoty do output registru, vybrat sloupec funkcí `select_column()`, zapnout odpovídající sloupec funkcí `turnLED()`.

```
PTA->PCOR = GPIO_PCOR_PTCO(0x3F000280);
PTA->PSOR = GPIO_PSOR_PTSO(value3);
column_select((i));
turnLED(delayt);
```

Za zmínku ještě stojí funkce `printEx()`, která vykteslí vykřičník.

2.4 turnLED()

Tato funkce zapne korespondující sloupec pomocí vypnutí a zapnutí `EN` signálu portu E.

2.5 computeDelay()

Funkce, která podle délky slova vypočítá jak dlouho by měla funkce `delay` čekat. Z důvodu hodně LED světel zaplých naráz, velká doba čekání způsobí, že lidské oko by si všimlo přepínání sloupců, protože by se nestíhalo přepínat dostatečně rychle a přišli bychom o výhodu nedokonalosti lidského oka. Při krátkém čekání máme dojem, jako by LED světla svítily konstantně.

2.6 column_select()

Funkce převzata z testovacího projektu. Přeloží konkrétní číslo sloupce pomocí dekodéru 4-na-16.

2.7 delay()

Funkce převzata z testovacího projektu. Slouží k čekání při zapínání a vypínání sloupce.

2.8 SystemConfig()

Funkce převzata z testovacího projektu. Slouží k nastavení prvotní konfigurace pinů mikrokontroleru.

3 Závěr

Způsob ověření vlastností probíhal postupným testováním. Program dokáže vypsat 5 zpráv:

- **Tlačítko SW2**: `HELLO WORLD!`
- **Tlačítko SW3**: `XKOCMA08`
- **Tlačítko SW4**: `IMP GOOD`
- **Tlačítko SW5**: `VUTBR FIT`
- **Tlačítko SW6**: `DAVID`

Představení funkčnosti: video

3.1 Autoevaluace

- **E**: Vybavení jsem si vypůjčil včas na na projektu žačal pracovat přibližně v půlce listopadu - **1b**
- **F**: Projekt jsem vypracoval podle zadání a podporuju 5 běžících textů, které se spustí přerušením. Při implementaci jsem použil jazyk C - **5b**
- **Q**: Zdrojové soubory jsem adekvátně okomentoval a snažil jsem se docílit největší možné dekompozice - **3b**
- **P**: Ve videu jsem ukázal všech 5 zpráv - **1b**
- **D**: Adekvátně jsem popsal úvod do problému, implementaci, ale se závěrem si nejsem moc jistý - **3.5b**

$(K_1 + k_2 * F/5) * (E + F + Q + P + D) = (0.25 + 0.75 * 5/5) * (1 + 5 + 3 + 1 + 3.5) =$ **13.5b**

Reference

[1] Šimek, V.: *FITKit v3.0 schematic*. [online], [viděno 17.11.2022].
URL <https://www.fit.vutbr.cz/~simekv/IMP_projekt_board_FITkit_v3.0_schematic.pdf>