Vážení zákazníci,

dovolujeme si Vás upozornit, že na tuto ukázku knihy se vztahují autorská práva, tzv. copyright.

To znamená, že ukázka má sloužit výhradně pro osobní potřebu potenciálního kupujícího (aby čtenář viděl, jakým způsobem je titul zpracován a mohl se také podle tohoto, jako jednoho z parametrů, rozhodnout, zda titul koupí či ne).

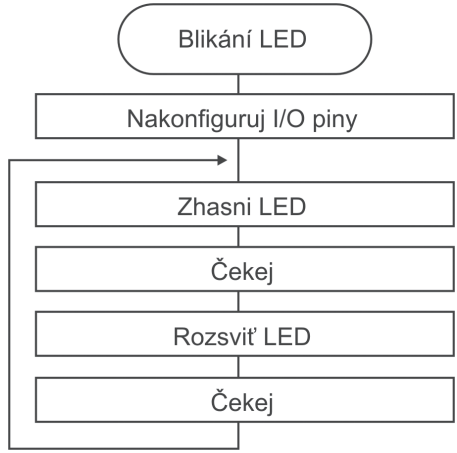
Z toho vyplývá, že není dovoleno tuto ukázku jakýmkoliv způsobem dále šířit, veřejně či neveřejně např. umisťováním na datová média, na jiné internetové stránky (ani prostřednictvím odkazů) apod.

redakce nakladatelství BEN – technická literatura
redakce@ben.cz

1.2 Realizace čekání pomocí jednoduché programové smyčky

Příklad 3: Chceme-li, aby dítě blikalo baterkou v co nejpřesnějším intervalu, řekneme mu: „Rozsviť, pak počítej: „Deset, devět, ..., dva, jedna, nula“. Zhasni, a opět počítej: „Deset, devět, ..., dva, jedna, nula“. Rozsviť, a tak pořád dokola.

Doba, po kterou dítě počítá, je čekání, jehož délku určuje číslo, od kterého počítání začíná. Je to podobné jako počítání při startu rakety. Náš mikrokontrolér to bude dělat stejně. Jeho program tedy bude pracovat podle následujícího vývojového diagramu.

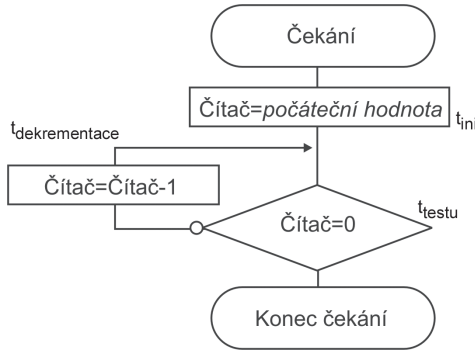


Obrázek 1.3 Vývojový diagram blikání LED

Nyní se budeme podrobněji věnovat jednotlivým blokům tohoto vývojového diagramu. Konfigurace I/O pinů, zhasnutí a rozsvícení LED by nám již nemělo dělat problémy.

Novinkou je pro nás čekání. Budeme se tedy věnovat tomuto úkolu.

Znázorněme si pomocí vývojového diagramu, jak bude čekání probíhat.



Obrázek 1.4 Vývojový diagram čekání

1

Na začátku nastavíme do čítače ***počáteční hodnotu***, například 10. Poté otestujeme, zda je tato hodnota rovna nule. Pokud ne, snížíme stav čítače o jedničku (dekrementujeme obsah čítače). Nyní bude obsah čítače roven devíti. Opět otestujeme obsah čítače, není-li nula, opět dekrementujeme jeho obsah. To provádíme tak dlouho, dokud není splněna podmínka, že je obsah čítače roven nule.

Z činnosti vyplývá výpočet doby čekání:

$$t_{celková} = t_{init} + t_{testu} + n \times (t_{dekrementace} + t_{testu})$$

kde

$t_{celková}$ *– celková doba čekání,*

t_{init} *– doba potřebná k naplnění čítače počáteční hodnotou,*

t_{testu} *– doba potřebná k otestování zda je obsah čítače roven nule,*

$n \times (t_{dekrementace} + t_{testu})$ *– doba provádění smyčky násobená dobou dekrementace čítače a testu.*

Platnost vzorce si můžeme ověřit tak, že do čítače nastavíme počáteční hodnotu nulu. V tom případě se provede pouze nastavení počáteční hodnoty a test, zda je čítač vynulován. Provedení bude trvat $t_1 = t_{init} + t_{testu}$.

Pokud jako počáteční hodnotu nastavíme jedničku, pak se provede nastavení počáteční hodnoty, jedenkrát se provede smyčka a nakonec test na nulový obsah čítače ukončí čekání. Provedení těchto činností bude trvat:

$$t_2 = t_1 + 1 \times (t_{testu} + t_{dekrementace}).$$

Když nastavíme počáteční hodnotu na dvojku, provede se nastavení počáteční hodnoty, dvakrát smyčka a nakonec test na nulový obsah čítače ukončí čekání. Provedení těchto činností bude trvat $t_3 = t_1 + 2 \times (t_{testu} + t_{dekrementace})$.

Vzorec pro výpočet doby čekání můžeme upravit do tvaru, který bude pro nás výhodnější

$$t_n = t_1 + n \times t_2$$

$$t_1 = t_{init} + t_{testu}$$

$$t_2 = t_{dekrementace} + t_{testu}$$

Není to až tak složité, že?

Jak tedy realizovat čekání v našem mikrokontroléru?

Jako čítač použijeme univerzální registr mikrokontroléru.

Již víme, že registry mikrokontroléru jsou osmibitové. Do univerzálního registru lze uložit číslo složené z osmi nul a jedniček (osmibitové číslo ve dvojkové soustavě, tedy číslo 0 až 255).

Poznámka:

V běžném životě jsme zvyklí pracovat s desítkovou soustavou, tedy s čísly složenými z číslic 0, 1, ... 9.

Například číslo 1987 v nám dobře známé desítkové soustavě umíme rozložit na součet jednotlivých číslic násobený jejich váhou.

$$1 \times 10^3 + 9 \times 10^2 + 8 \times 10^1 + 7 \times 10^0 = 1987$$

Váha jednotlivých číslic je zde označena šedě.

*V desítkové soustavě je váha 10^n, kde n je pozice číslice počítáno od nuly. Váhu 10^0 má **nejméně významná číslice** (jednotky).*

Ve dvojkové soustavě je to podobné. Číslo je složené z dvojkových číslic, tedy z nul a jedniček. Váha jednotlivých číslic je 2^n.

*Nejméně významná dvojková číslice má váhu 2^0 a obvykle se označuje anglickou zkratkou **LSb** (last significant bit).*

Například dvojkové číslo 10110101 umíme rozložit na součet jednotlivých číslic násobený jejich váhou.

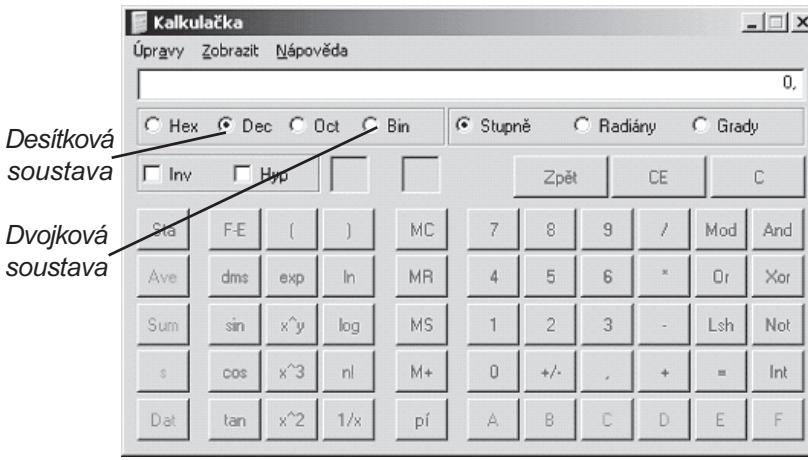
$$1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 181$$

Když takto vytvořený součet provedeme, dostaneme desítkovou hodnotu dvojkového čísla. Dvojkové číslo 10110101 odpovídá desítkovému číslu 181.

Co z toho tedy plyne?

Jakékoliv číslo, které umíme vyjádřit v desítkové soustavě, lze vyjádřit číslem ve dvojkové soustavě.

Není třeba se děsit a složitě se učit převody mezi jednotlivými soustavami. Součástí operačního systému Windows je kalkulačka. Nastavíme-li v ní ***Zobrazit – Vědecká****, pak zde lze snadno provádět snadno převody mezi soustavami.*



Do kalkulačky zapíšeme číslo například v desítkové soustavě (aktivní ***Dec****) a pak přepneme volbu na* ***Bin****. Tím je číslo převedeno z desítkové soustavy do dvojkové. Podobně se převádí číslo z dvojkové soustavy do desítkové, popřípadě pro převody i do soustavy osmičkové (****Oct****), nebo šestnáctkové (****Hex****).*

Nejmenší celé dvojkové číslo, které do registru můžeme uložit, je 00000000 (nula i v desítkové soustavě).

Největší číslo pak 11111111, tedy 255 v desítkové soustavě.

Použijme-li jako čítač osmibitový registr, jsme schopni realizovat čekání 0 až 255 cyklů (průchodů čekací smyčkou).

Jak je vidět na *obrázku 1.4*, algoritmus čekání není moc složitý. Pojďme jej naprogramovat a podívejme se, jak to poběží v mikrokontroléru.