VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

DEPARTMENT OF COMPUTER SYSTEMS

ARM/FITKIT3: APLIKACE MODULU WATCHDOG TIMER (WDOG)

ARM/FITKIT3: WATCHDOG TIMER MODULE (WDOG) APPLICATION

PROJEKTOVÁ DOKUMENTACE

PROJECT DOCUMENTATION

AUTOR PRÁCE **PETR BUCHAL**

AUTHOR

BRNO 2017

Obsah

Kapitola 1

Úvod

Spolehlivost je jednou z klíčových vlastností dnešních zařízení, možná úplně tou nejdůležitější. Můžeme ji maximalizovat spoustou nejrůznějších metod, pokud však k chybám dochází na té nejnižší úrovni, jakékoli snahy na vyšších vrstvách jsou bezcenné. V předmětu IMP se zabýváme mikrokontrolery. Právě jejich správné fungování je klíčovým prvkem spolehlivosti určitého typu zařízení na velmi nízké, a tedy klíčové úrovni.

1.1 Motivace

Spolehlivost mikrokontroleru závisí na práci lidí, kteří navrhují jeho architekturu a poté na těch kteří píšou kód, který na něm běží. Co se ale stane, když některá část mikrokontroleru selže a aplikace přestane plnit svou roli? Odpověď je jednoduchá, pravděpodobně dojde k resetu. Tato skutečnost je způsobená existencí něčeho, čemu se říká WDOG. Tento modul zajišťuje, že v okamžiku, kdy dojde k selhání, resetuje mikrokontrolér. Pochopit jakým způsobem tento modul funguje není složité, ale neexistuje žádná aplikace, která by jednoduchým způsobem demonstrovala jeho funkčnost.

1.2 Cíl práce

Cílem práce je vytvořit aplikaci, která „user-friendly“ způsobem demonstruje funkcionalitu modulu WDOG na mikrokontroleru Kinetis K60. Tato aplikace bude demonstrovat oba dva módy WDOGu (periodický, okénkový) a jako zdroj hodin bude využívat Low-Power Oscilátor.

Kapitola 2

Návrh Aplikace

Aplikace se bude zabývat demonstrací funkcionality modulu WDOG. Ten hlídá, zdali v systému nedošlo k selhání a v jeho případě resetuje systém. Během normální činnosti systému systém posílá WDOGu refresh zprávy, které mu říkají, že systém běží, jak má a že není zaseklý. Pokud se systém zasekne, přestane posílat zprávy WDOGu a ten ho resetuje.

2.1 Popis registrů

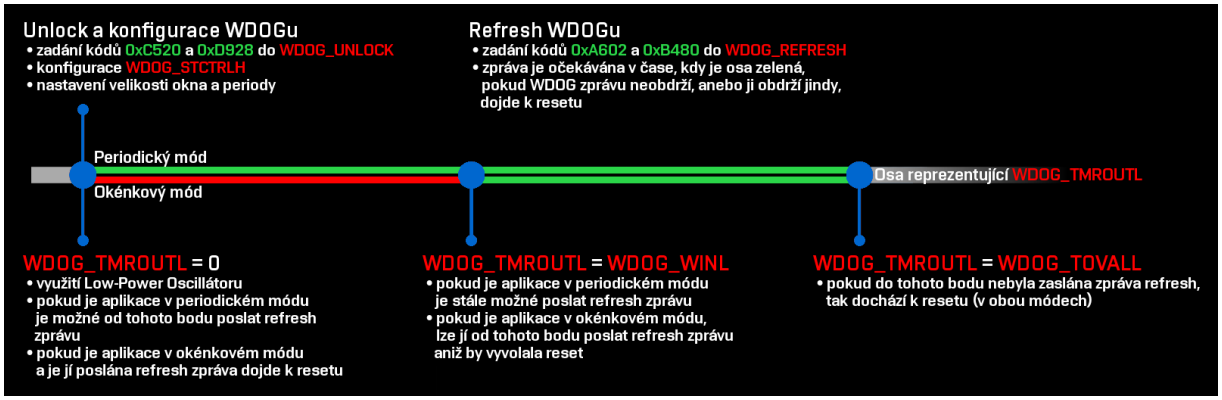
V aplikaci budeme využívat registr WDOG_STCRHL na, zapnutí WDOGu, nastavení Low-Power Oscilátoru jako zdroje hodin a vynutí nebo zapnutí okénkového módu. WDOG_TOVALH a WDOG_TOVALL se použijí na nastavení velikosti periody. WDOG_WINH a WDOG_WINL se použijí na nastavení velikosti okna. Do registru WDOG_REFRESH budeme odesílat refresh zprávy (zápis dvou různých hodnot). Registr WDOG_UNLOCK se využívá pro odemknutí možnosti změny hodnoty registrů WDOGu (zápis dvou různých hodnot). Registry WDOG_TMROUTH a WDOG_TMROUTL nám říkají, jaká doba uběhla od resetu. Registr WDOG_RSTCNT počítá kolikrát během doby, kdy je mikrokontroler zapnut, došlo k resetu.

Register name	Width (in bits)
Watchdog Status and Control Register High (WDOG_STCTRLH)	16
Watchdog Status and Control Register Low (WDOG_STCTRLL)	16
Watchdog Time-out Value Register High (WDOG_TOVALH)	16
Watchdog Time-out Value Register Low (WDOG_TOVALL)	16
Watchdog Window Register High (WDOG_WINH)	16
Watchdog Window Register Low (WDOG_WINL)	16
Watchdog Refresh register (WDOG_REFRESH)	16
Watchdog Unlock register (WDOG_UNLOCK)	16
Watchdog Timer Output Register High (WDOG_TMROUTH)	16
Watchdog Timer Output Register Low (WDOG_TMROUTL)	16
Watchdog Reset Count register (WDOG_RSTCNT)	16
Watchdog Prescaler register (WDOG_PRESC)	16

Obrázek 1: Tabulka obsahující seznam všech registrů WDOGu [1]

2.2 Módy WDOGu

WDOG funguje buďto v periodickém módu,
který se používá na kontrolu správného běhu programu anebo v okénkovém módu, který se používá na zjištění, zdali neběží aplikace na mikrokontroleru jinou rychlostí, než má. Pro periodický režim je důležité nastavení velikosti periody, ve které musí přijít refresh zpráva WDOGu (registry WDOG_TOVALL a WDOG_TOVALH). Pokud v tomto časovém intervalu zpráva nepřijde nastane reset. Okénkový režim očekává refresh zprávu v časovém intervalu od hodnot registrů WDOG_WINH a WDOG_WINL do hodnot registrů WDOG_TOVALH a WDOG_TOVALL, pokud dostane zprávu mimo tento časový interval nebo ji nedostane vůbec, nastává reset.



Obrázek 2: Infografika popisující chod modulu WDOG v aplikaci

2.3 Nastavení hodnot registrů WDOGu

Abychom mohli nastavit jakékoli hodnoty ve WDOG registrech je nutné jej odemknout. Odemknutí umožňují dvě speciální hodnoty 0xC520 a 0xD928, které je třeba zapsat do WDOG_UNLOCK registru. Konkrétně tak, že ve WCT ihned po resetu (konfigurační doba WDOG) se musí zapsat alespoň první hodnota do WDOG_UNLOCK registru a poté v průběhu dvaceti časových cyklů hodnota druhá. Poté nastavujeme hodnoty jednotlivých registrů. V registru WDOG_STCTRLH nastavíme příznak WDOGEN na 1 (zapnutí WDOGu), příznak CLKSRC na 0 (využití LPO jako zdroje hodin), příznak IRQRSTEN na 1 (před resetem se vygeneruje přerušení) a příznak WINEN na 1 pokud chceme pracovat v okénkovém módu (periodický mód je 0). Dále pak nastavíme hodnoty v registrech pro velikost periody a okna. Pokud se při nastavení něco nepovede, WDOG resetuje systém.

2.4 Refresh WDOGu

Refresh WDOGu umožňují dvě speciální hodnoty 0xA602 a 0xB480, které je třeba zapsat do WDOG_REFRESH registru. Tyto hodnoty musí být zapsány po sobě v časovém intervalu dvaceti časových cyklů. Pokud se při refreshi něco nepovede anebo jsou hodnoty zapsány v nevhodné době, WDOG resetuje systém.

2.5 Reset

K resetu vyvolaným WDOGem nedochází pouze nesprávným zasíláním refresh zpráv, ale také nesprávným zapisováním hodnot do registrů. Pokud je v registru WDOG_STCRHL nastaven příznak IRQ_RST_EN dochází ještě před resetem ke změně hodnoty příznaku INTFLG v registru WDOG_STCTRLL na 1, poté k přerušení a poté teprve k resetu.

2.5.1 Případy vyvolání resetu

- Uplynutí periody dříve, než WDOG dostal refresh zprávu (okénkový, periodický mód)
- Brzkým zasláním refresh zprávy v okénkovém módu
- Selhání odemknutí WDOGu během WCT po resetu
- Po odemknutí nebyla změněna hodnota v žádném registru
- Do WDOG_UNLOCK byla zapsána jiná hodnota než hodnota pro odemknutí
- Do WDOG_REFRESH byla zapsána jiná hodnota než hodnota pro refresh WDOGu
- Mezi zapsáním hodnot pro odemknutí je větší mezera než 20 časových cyklů
- Mezi zapsáním hodnot pro refresh WDOGu je větší mezera než 20 časových cyklů

Kapitola 3

Popis implementace

Aplikaci jsem implementoval v jazyce C ve vývojovém prostředí Kinetis Design Studio (KDS) a otestoval na mikrokontroléru Kinetis K60 (s jádrem ARM Cortex-M4). Důležitou částí aplikace je využití semihostingu, ten je použit na zobrazení informačních výpisů pro uživatele. Zdrojový kód bych rozdělil na tři části. První část kódu je věnována definici parametrů, s kterými může být aplikace spuštěna, tuto část mění uživatel podle potřeby (obrázek č. 3). Druhá část kódu je ponechána z demo aplikace na ověření správného propojení FITkitu s počítačem (změna rychlosti blikání LED). Ponechal jsem zde tuto část, protože blikající LED světlo je užitečné na určení doby, kdy je FITkit v provozu (během resetu LED nebliká).

Třetí částí jsou poté funkce pro výpis aktuálních hodnot z registrů a samotné tělo programu. První funkcí pro výpis aktuálních hodnot je `printWDOGRegisters`, která vypíše hodnoty všech WDOG registrů, druhou je poté `printWDOG_STCTRHL`, která vypíše důležité příznaky z registru WDOG_STCTRHL. V těle programu se nejdříve provede odemknutí WDOGu a poté změna hodnot v jeho registrech, tedy nastavení příznaků v registru WDOG_STCTRHL a nastavení velikosti periody a okna. Poté se inicializují hodnoty v dalších modulech mikrokontroleru funkcemi převzatými a poupravenými z demo aplikace. Následně program přejde do nekonečné smyčky, ve které probíhá kontrola, zdali nebylo stisknuto nějaké tlačítko. Pokud bylo stisknuto tlačítko č. 1 (obrázek č. 4), vytiskne se výpis o stavu registrů důležitých pro uživatele. Pokud bylo stisknuto tlačítko č. 2 (obrázek č. 4) odešle se refresh zpráva a uživatel je o této skutečnosti informován výpisem.

3.1 Seznam využitých knihoven

- MK60D10.h
- stdio.h
- string.h

Kapitola 4

Návod na použití

Aplikace nabízí možnost spuštění s několika odlišnými hodnotami parametrů. Tyto parametry se nastavují v kódu před jeho přeložením a nahráním do mikrokontroleru. Můžeme si vybrat, zdali WDOG bude operovat v periodickém nebo okénkovém módu, nepoužitý mód musí zůstat zakomentován. Poté aplikace nabízí nastavení velikosti okna a periody z několika předdefinovaných variant. U těchto variant je napsáno, jakou dobu přibližně trvají, nepoužité varianty je opět nutné zakomentovat. Aplikaci je důležité mít spuštěnou se semihostingem, kvůli jejím výpisům.

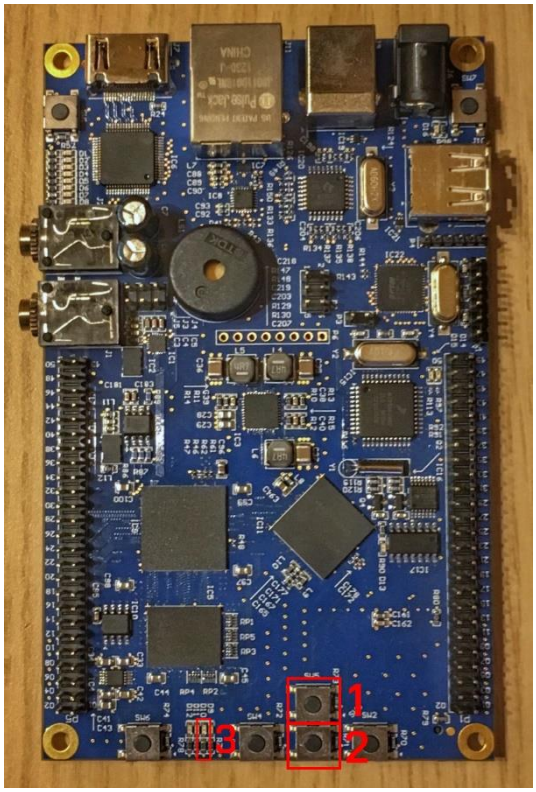
```
/*******    Parametry aplikace    *******/
//nastavuji se celkem tri hodnoty, ktere ovlinuji chod aplikace
//pri vyberu kazde hodoty nehodici zakomentujte a vybranou odkomentujte
//pokud pracujete s okenkovym rezimem velikost periody musi byt vetsi nez velikost okna

/***    Vyber modu    ***/
int wdog_stctrlh = 0b0101;    //periodicky rezim
//int wdog_stctrlh = 0b1101;    //okenkovy rezim

/***    Velikost periody    ***/
int wdog_tovalh = 0; int wdog_tovall = 8191;   //Casova delka periody ~ cca 41s
//int wdog_tovalh = 0; int wdog_tovall = 32767;   //Casova delka periody ~ cca 2m 42s
//int wdog_tovalh = 0; int wdog_tovall = 65535;   //Casova delka periody ~ cca 5m 28s

/***    Velikost okna    ***/
int wdog_winh = 0; int wdog_winl = 4095;   //Casova delka okna ~ cca 20s
//int wdog_winh = 0; int wdog_winl = 16383;   //Casova delka okna ~ cca 1m 22s
//int wdog_winh = 0; int wdog_winl = 32767;   //Casova delka okna ~ cca 2m 42s
```

Obrázek 3: Část kódu, ve které se nastavují parametry programu



Aplikace se ovládá dvojicí tlačítek nacházejících se na obrázku č. 4. Pokud aplikace běží a není ve stavu resetu, bliká dioda č. 3 (obrázek č. 4).

Tlačítko č. 1 (obrázek č. 4) vypíše do terminálu semihostingu relevantní informace o aktuálním stavu registru WDOG_TMROUTH (samotná hodnota, procentní porovnání s nastavenou délkou periody a okna). Tlačítko č. 2 (obrázek č. 4) odešle refresh zprávu do registru WDOG_REFRESH. Při úspěšném přijetí se vypíše zpráva „Refreshed.“. V opačném případě nastává reset a přestává blikat dioda označena číslem 3.

Pokud není refresh zpráva odeslaná v očekávaném časovém intervalu rovněž dochází k resetu.

Obrázek 4: FITkit 3 s označením částí důležitých pro běh programu

Kapitola 5

Závěr

Aplikace byla otestována na prostředku FITkit3 a lze tvrdit, že úspěšně demonstruje chování WDOGu. Věcí, co nám zůstává k diskuzi je jakým způsobem určit, co přesně vyvolalo reset. Jak jsem zmínil v kapitole 2.5.1 reset může být vyvolaný poměrně hodně příčinami. K určení, jaká byla konkrétní příčina, by nám mohla sloužit možnost vyvolání přerušení ještě před resetem prostředku. Poté bychom mohli z funkce, která řeší přerušení, na základě hodnot jednotlivých registrů určit, proč došlu k resetu. Například porovnáním registrů WDOG_TOVALH a WDOG_TOVAL s registry WDOG_TMROUTH a WDOG_TMROUTL bychom mohli zjistit, zdali došlo k resetu z důvodu nezaslání refresh zprávy.

Literatura

[1] K60 Sub-Family Reference Manual. In: *NXP Semiconductors* [online]. 2012 [cit. 2017-12-24]. Dostupné z: http://cache.freescale.com/files/32bit/doc/ref_manual/K60P144M100SF2V2RM.pdf