Kód írása regiszterszinten

Név: Stefán Kornél

Dátum: 2024. 11. 25 18:00

Mérőhely: 7 bal

Ajánlott irodalom

A jegyzőkönyvhöz tartozó PPT (MicLab-Registers.pptx)

EFM8BB1 Reference Manual: https://www.silabs.com/documents/public/reference-manuals/efm8bb1-rm.pdf

Laboratory Practical: http://eta.bibl.u-szeged.hu/901/1/2011_0104_szte_6_laboratory_practical.pdf

Honlap: http://www.inf.u-szeged.hu/noise/Education/MicLab/

Jegyzőkönyv készítése

A jegyzőkönyvek az órán végzett munka dokumentálására szolgálnak. A letölthető minta jegyzőkönyvet kell kiegészíteni a megfelelő információkkal: név, dátum, mérőhely (pl. 3. jobb), a feladatokhoz tartozó esetleges kifejtendő válaszokkal, valamint a kódok lényeges részével.

A jegyzőkönyveket a CooSpace-en kell feltölteni, külön pdf formátumban csatolni kell a jegyzőkönyvet (a fájl neve a következő mintát kövesse: NagyJ.KissB.03.pdf), egy külön zip fájlban pedig a kódokat (*.c, *.cwg). Amennyiben probléma merül fel a beadás során, az anyagokat az oktató e-mail címére kell elküldeni, levél tárgya legyen pl. MicLab 03.

1. feladat – LED0 bekapcsolása regiszterszinten

Írjon egy olyan C kódot, ami bekapcsolja az EFM8BB1 fejlesztőkiten lévő LED0 nevű LED-et. Ehhez csak a Reference Manual használható, a konfigurátor nem. A feladathoz egy „Empty C Program” típusú projektet hozzon létre. A feladatra akkor jár teljes a pontszám, ha a regiszter értékei esetén a konstansok helyett a beépített enumok használatával oldjuk meg. Az enumok az SI_EFM8BB1_Register_Enums.h fájlban találhatók.

Tipp: A jegyzőkönyvhöz tartozó PPT-ben megtalálhatók a szükséges regiszterek.

A program részekre bontott forráskódja (Main.c, Interrupts.c, ha van):

Main.c

```
#include <SI_EFM8BB1_Register_Enums.h>

#define WD_DISABLE_A 0xDE
#define WD_DISABLE_B 0xAD

#define LED0 P1_B4

void SiLabs_Startup (void)
{
  // Disable the watchdog here.

  // Erre nincs konstans az Enumsban.
  WDTCN = WD_DISABLE_A;
  WDTCN = WD_DISABLE_B;

  XBR2 |= XBR2_XBARE__ENABLED;
```

```
  P1MDOUT |= P1MASK_B4__COMPARED;
}

int main (void)
{
  LED0 = 0;

  while (1) {}
}
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

2. feladat - Timer 3 használata 16 bites módban, interrupt engedélyezéssel

Írjon egy programot regiszterszinten, ami a Timer 3 segítségével villogtatja a LED0-át 10 Hz-es frekvenciával. A Timer 3-at 16 bites módban használja és engedélyezze a hozzá tartozó interruptot. Az interrupt rutint is írja meg (number_of_interrupt: TIMER3_IRQn), amiben a LED0-hoz tartozó bit kapcsolgatása történik. A reload érték képlettel legyen implementálva a kódban, ne csak egy konstansként, melynek „bemenete” legyen a timer kívánt túlcsordulási frekvenciája. (A képlethez segítséget talál a Miclab-utmutato.pdf-ben).

Tippek: a Timer 3 használatához interrupt módban a TMR3CN0 (TR3, TF3H bitek) és a TMR3RLH, TMR3RLL reload regiszterek írása szükséges, valamint a Timer 3-hoz tartozó interrupt engedélyezése az EIE1 regiszterben és az összes interrupt engedélyezése az IE regiszterben. A TF3H overflow flaget a megfelelő bitművelettel kell törölnünk, úgy hogy ne írjuk felül a többi helyiértéken lévő biteket, mert a Timer 3 control regisztere nem bitcímezhető.

A program részekre bontott forráskódja (Main.c, Interrupts.c, ha van):

Main.c

```
#include <SI_EFM8BB1_Register_Enums.h>
#include <SI_EFM8BB1_Defs.h>

#define WD_DISABLE_A 0xDE
#define WD_DISABLE_B 0xAD

#define SYSCLOCK 24500000u
#define SYS_PRESCALER 8u

#define TIMER3_PRESCALER 12u
#define TIMER3_BASE_CLOCK (SYSCLOCK / (SYS_PRESCALER * TIMER3_PRESCALER))
#define TIMER3_TARGET_HZ 10u
#define TIMER3_SIZE 65536u
#define TIMER3_RELOAD (TIMER3_SIZE - (TIMER3_BASE_CLOCK / TIMER3_TARGET_HZ))

#define BYTE_SIZE 8u

#define LED0 P1_B4

void SiLabs_Startup (void)
```

```
{
  // Erre nincs konstans az Enumsban.
  WDTCN = WD_DISABLE_A;
  WDTCN = WD_DISABLE_B;

  XBR2 |= XBR2_XBARE__ENABLED;

  P1MDOUT |= P1MASK_B4__COMPARED;

  IE_EA = 1;
  EIE1 |= EIE1_ET3__ENABLED;

  TMR3CN0 |= TMR3CN0_TR3__RUN;
  TMR3CN0 &= ~TMR3CN0_T3SPLIT__BMASK;
  TMR3CN0 &= ~TMR3CN0_T3XCLK__BMASK;

  TMR3RLH = TIMER3_RELOAD >> BYTE_SIZE;
  TMR3RLL = TIMER3_RELOAD;
}

int main (void)
{
  while (1) {}
}

void timer3_interrupt (void) interrupt TIMER3_IRQn
{
  TMR3CN0 &= ~TMR3CN0_TF3H__BMASK;

  LED0 = ~LED0;
}
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

Megjegyzések