Interrupt

MICLAB-02

Név: Pilter Zsófia, Vad Avar

Dátum: 2024. 09. 16.

Mérőhely: 1 (bal és jobb)

Bevezetés

Az interrupt használatának megismerése.

Ajánlott irodalom

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 02.

1. feladat – LED vezérlése interrupttal I

Írjon egy programot, mely a LED0-át 10 Hz frekvenciával villogtatja, ha a BTN0 nyomógomb le van nyomva. A vezérlési feladatot a Timer 2 interrupt rutin lássa el.

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

Config:

```
<?xml version="1.0" encoding="ASCII"?>
<device:XMLDevice xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:device="http://www.silabs.com/ss/hwconfig/document/device.ecore"
name="EFM8BB10F8G-A-QSOP24" partId="mcu.8051.efm8.bb1.efm8bb10f8g-a-qsop24"
version="4.0.0" contextId="%DEFAULT%">
  <mode name="DefaultMode">
    <property object="DefaultMode" propertyId="mode.diagramLocation" value="100,
100"/>
    <property object="INTERRUPT_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.interruptenable.enableallinterrupts" value="Enabled"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.interruptenable.enabletimer2interrupt" value="Enabled"/>
```

```
    <property object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar"
value="Enabled"/>
    <property object="TIMER01_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER16_2" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER16_2" propertyId="timer16.control.runcontrol"
value="Start"/>
    <property object="TIMER16_2" propertyId="timer16.control.timerrunningstate"
value="Timer is Running"/>
    <property object="TIMER16_2"
propertyId="timer16.initandreloadvalue.targetoverflowfrequency" value="20"/>
    <property object="TIMER16_2"
propertyId="timer16.initandreloadvalue.timerreloadvalue" value="52776"/>
    <property object="TIMER16_2"
propertyId="timer16.reloadhighbyte.reloadhighbyte" value="206"/>
    <property object="TIMER16_2" propertyId="timer16.reloadlowbyte.reloadlowbyte"
value="40"/>
    <property object="TIMER16_3" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER_SETUP_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="WDT_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="WDT_0" propertyId="wdt.watchdogcontrol.wdtinitialvalue"
value="5"/>
    <property object="WDT_0" propertyId="wdt.watchdogcontrol.wdtperiodactual"
value="6.554 s"/>
  </mode>
  <modeTransition>
    <property object="RESET &#x2192; DefaultMode"
propertyId="modeTransition.source" value="RESET"/>
    <property object="RESET &#x2192; DefaultMode"
propertyId="modeTransition.target" value="DefaultMode"/>
  </modeTransition>
</device:XMLDevice>
```

Interrupts.c

```
#include <SI_EFM8BB1_Register_Enums.h>                  // SFR declarations
#define ONBOARD_LED P1_B4
#define ONBOARD_BTN P0_B2

// This routine changes the state of the LED whenever Timer2 overflows.
//
SI_INTERRUPT(TIMER2_ISR, TIMER2_IRQn)
{
   TMR2CN0_TF2H = 0;
   if (!ONBOARD_BTN)
   {
          ONBOARD_LED = !ONBOARD_LED;
   }
   else
   {
          ONBOARD_LED = 1;
   }
}
```

Main:

main.c-ben semmi érdemleges nem volt.

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

2. feladat – LED állapotának billentése nyomógombbal

Írjon egy programot, mely a LED0-át átbillenti az aktuális állapotának az ellenkezőjére, ha megnyomjuk a BTN0-át. A nyomógomb lenyomásakor a véletlenszerű, ún. „pergése" okozta többszöri detektálás elkerüléséhez pergésmentesíteni is kell a nyomógombot.

A megvalósítás részletei:

- A nyomógombot a Timer 2 használatával pergésmentesítse, mely 10 ms-os periódusidővel generál egy interruptot és léptet egy változót (extern uint8_t legyen), amivel az időt mérhetünk.
- A nyomógomb állapotát a főprogramban ellenőrizze. Amennyiben a gomb 50 ms-ig lenyomott állapotban van, úgy minősítse lenyomottnak és billentse át a LED0 állapotát az ellenkezőjére!

Tippek a pergésmentesítés algoritmusához: Ha a gombot lenyomtuk, akkor a program várakozzon 50 ms-ig (tartsa fel a főprogramot a megfelelő C vezérlési szerkezettel), billentse át a LED0-át, majd ismét várakozzon, egészen addig, amíg a gomb lenyomott állapotban van (ezzel el lehet kerülni a LED0 folyamatos billegtetését, ha a gomb sokáig van lenyomva). Ezután ismét várjon 50 ms-ot, hogy a gomb felengedése is pergésmentesítve legyen.

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

Config:

```
<?xml version="1.0" encoding="ASCII"?>
<device:XMLDevice xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:device="http://www.silabs.com/ss/hwconfig/document/device.ecore"
name="EFM8BB10F8G-A-QSOP24" partId="mcu.8051.efm8.bb1.efm8bb10f8g-a-qsop24"
version="4.0.0" contextId="%DEFAULT%">
  <mode name="DefaultMode">
    <property object="DefaultMode" propertyId="mode.diagramLocation" value="100,
100"/>
    <property object="INTERRUPT_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.interruptenable.enableallinterrupts" value="Enabled"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.interruptenable.enabletimer2interrupt" value="Enabled"/>
    <property object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar"
value="Enabled"/>
    <property object="TIMER01_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER16_2" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER16_2" propertyId="timer16.control.runcontrol"
value="Start"/>
    <property object="TIMER16_2" propertyId="timer16.control.timerrunningstate"
value="Timer is Running"/>
    <property object="TIMER16_2"
propertyId="timer16.initandreloadvalue.targetoverflowfrequency" value="100"/>
    <property object="TIMER16_2"
propertyId="timer16.initandreloadvalue.timerreloadvalue" value="62984"/>
    <property object="TIMER16_2"
propertyId="timer16.reloadhighbyte.reloadhighbyte" value="246"/>
```

```
    <property object="TIMER16_2" propertyId="timer16.reloadlowbyte.reloadlowbyte"
value="8"/>
    <property object="TIMER16_3" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER_SETUP_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="WDT_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="WDT_0" propertyId="wdt.watchdogcontrol.wdtenable"
value="Disable"/>
    <property object="WDT_0" propertyId="wdt.watchdogcontrol.wdtinitialvalue"
value="5"/>
    <property object="WDT_0" propertyId="wdt.watchdogcontrol.wdtperiodactual"
value="6.554 s"/>
  </mode>
  <modeTransition>
    <property object="RESET &#x2192; DefaultMode"
propertyId="modeTransition.source" value="RESET"/>
    <property object="RESET &#x2192; DefaultMode"
propertyId="modeTransition.target" value="DefaultMode"/>
  </modeTransition>
</device:XMLDevice>
```

Interrupts.c

```
#include <SI_EFM8BB1_Register_Enums.h>
#define ONBOARD_LED P1_B4
#define ONBOARD_BTN P0_B2

//-----------------------------------------------------------------------------
// TIMER2_ISR
//-----------------------------------------------------------------------------
//
// TIMER2 ISR Content goes here. Remember to clear flag bits:
// TMR2CN0::TF2H (Timer # High Byte Overflow Flag)
// TMR2CN0::TF2L (Timer # Low Byte Overflow Flag)
//
//-----------------------------------------------------------------------------

extern uint8_t tim;

SI_INTERRUPT (TIMER2_ISR, TIMER2_IRQn)
{
      if (!ONBOARD_BTN)
      {
            tim++;
      }
      else
      {
            tim = 0;
      }

}
```

Main.c:

```
#include <SI_EFM8BB1_Register_Enums.h>                  // SFR declarations
#include "InitDevice.h"
#define ONBOARD_LED P1_B4
#define ONBOARD_BTN P0_B2

// $[Generated Includes]
// [Generated Includes]$
```

```
//-----------------------------------------------------------------------------
// SiLabs_Startup() Routine
// ----------------------------------------------------------------------------
// This function is called immediately after reset, before the initialization
// code is run in SILABS_STARTUP.A51 (which runs before main() ). This is a
// useful place to disable the watchdog timer, which is enable by default
// and may trigger before main() in some instances.
//-----------------------------------------------------------------------------
void SiLabs_Startup (void)
{
  // $[SiLabs Startup]
  // [SiLabs Startup]$

}

uint8_t tim = 0;

//-----------------------------------------------------------------------------
// main() Routine
// ----------------------------------------------------------------------------
int main (void)
{
  // Call hardware initialization routine
  enter_DefaultMode_from_RESET();

  ONBOARD_LED = 1;

  while (1)
  {
        if (tim >= 5)
        {
              while (!ONBOARD_BTN);
              ONBOARD_LED = !ONBOARD_LED;
        }
    // $[Generated Run-time code]
    // [Generated Run-time code]$
  }

}
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

Szorgalmi feladat – LED vezérlése interrupttal II

Írjon egy programot, mely a LED0-ás LED-et 5 Hz, 10 Hz és 50 Hz frekvenciával villogtatja. A vezérlési feladatot a Timer 0 interrupt rutin lássa el, a villogási frekvenciák között a BTN0-ás nyomógomb segítségével lehessen váltani. A nyomógombot pergés mentesíteni is szükséges.

Segítség: Mivel a Timer0 8 bites, ezért a kívánt frekvencia elérésére önmagában nem képes, egy timer interrupt rutinban megvalósított számláló bevezetésével érhető el a kívánt frekvencia.

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

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

Megjegyzések