PCA - PWM out

MICLAB



Név: Stefán Kornél

Dátum: 2024. 11. 18. 18:00

Mérőhely: 7 bal

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.o3.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 o3.

<u> 1. feladat – PCA konfigurálása, LED vezérlése PWM módban</u>

A Port I/O Mapping ablakban engedélyezze a PCAo_CEXo kimenetet. Pin-skipp segítségével tolja el a PCAo_CEXo-át a P1.4-be. Konfigurálja be a PCA perifériát 8 bites módban. Az órajelét a Timer o overflow adja. A Cannel o legyen Edge-aliagned PWM módban.

Készítsen egy programot, ami a PCA segítségével 1 másodperces periódusidővel villogtatja az EFM8BB1LCK panelen lévő LEDo-át 25%-os kitöltési tényezővel. Ehhez először a Timer o-át konfigurálja be, majd a programban állítsa be PCA capture/compare regiszterét a megfelelő értékre.

Tipp: az 1 s-os periódusidőt el lehet érni a megfelelő beállításokkal a konfigurátorban.

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

Config

```
cproperty object="P1.3" propertyId="ports.settings.skip" value="Skipped"/>
  cproperty object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar" value="Enabled"/>
  cproperty object="PCACH_0"
propertyId="pcach.pcachannelcapturecomparemode.enablechannelcomparatorfunction" value="Enabled"/>
  cproperty object="PCACH_0"
propertyId="pcach.pcachannelcapturecomparemode.enablechannelpulsewidthmodulationmoden"
value="Enabled"/>
  <property object="PCACH_0" propertyId="pcach.pcacontrol.channelcapturecomparemode"</pre>
value="Predefined 8~11-bit pulse modulator"/>
  value="250.000 Hz"/>
  cproperty object="PCA_0"
propertyId="pca.pcacountertimerconfiguration.pcaclockfrequencyintegervalue" value="250"/>
  mS"/>
  value="Timer 0 overflow"/>
  cproperty object="TIMER01 0"
propertyId="timer01.timer0mode2:8bitcountertimerwithautoreload.targetoverflowfrequency" value="250"/>
  cproperty object="TIMER01_0"
propertyId="timer01.timer0mode2:8bitcountertimerwithautoreload.timerreloadvalue" value="1"/>
  value="SYSCLK / 48"/>
  Counter/Timer with Auto-Reload"/>
  cyroperty object="TIMER_SETUP_0" propertyId="timer_setup.timer0.timerrunningstate" value="Timer is
Running"/>
  <property object="TIMER_SETUP_0" propertyId="timer_setup.timer0.timerswitch1:runcontrol"</pre>
value="Start"/>
  cproperty object="TIMER_SETUP_0" propertyId="timer_setup.timer01control.timer0runcontrol"
  cproperty object="WDT_0" propertyId="ABPeripheral.included" value="true"/>
  cproperty object="WDT_0" propertyId="wdt.watchdogcontrol.wdtenable" value="Disable"/>
 </mode>
 <modeTransition>
  value="DefaultMode"/>
 </modeTransition>
</device:XMLDevice>
Más kódra nincs valójában szükség, de main.c
#define MAGIC VALUE 64
int main (void)
{
 // Call hardware initialization routine
 enter_DefaultMode_from_RESET();
 PCA0CPH0 = MAGIC_VALUE;
 while (1)
  // $[Generated Run-time code]
  // [Generated Run-time code]$
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- <u>Igen</u>
- Nem

2. feladat – LED fényerejének változtatása nyomógombbal

A kiegészítő panelen lévő SW1 nyomógombbal 0%-os kitöltési tényezőtől 75%-os kitöltési tényezőig 4 lépésben növelje a LEDo fényerejét gombnyomásra. Az utolsó utáni lépés után ismét az első következzen. A PWM frekvenciája legyen 100 Hz feletti. Amig az SW1 le van nyomva, addig folyamatosan léptesse a fényerőt egy timer interruptban. A 4 lépés megtételéhez szükséges idő legyen 1 másodperc.

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"</pre>
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">
   cproperty object="CROSSBAR0" propertyId="xbar0.pca0.cex" value="CEX0"/>
   <property object="DefaultMode" propertyId="mode.diagramLocation" value="100, 100"/>
<property object="INTERRUPT_0" propertyId="ABPeripheral.included" value="true"/>
   <property object="P0.4" propertyId="ports.settings.skip" value="Skipped"/>
<property object="P0.5" propertyId="ports.settings.skip" value="Skipped"/>
   cyroperty object="PCACH_0" propertyId="pcach.pcachannel.capturecomparehighbyte" value="11"/>
cyroperty object="PCACH_0" propertyId="pcach.pcachannel.capturecomparehighbyte" value="11"/>
cyroperty object="PCACH_0"
propertyId="pcach.pcachannelcapturecomparemode.enablechannelcomparatorfunction" value="Enabled"/>
   cproperty object="PCACH_0"
propertyId="pcach.pcachannelcapturecomparemode.enablechannelpulsewidthmodulationmoden"
value="Enabled"/>
   <property object="PCACH_0" propertyId="pcach.pcacontrol.channelcapturecomparemode"</pre>
value="Predefined 8~11-bit pulse modulator"/>
   cproperty object="PCA_0" propertyId="ABPeripheral.included" value="true"/>
   value="255.208 kHz"/>
   cproperty object="PCA_0"
propertyId="pca.pcacountertimerconfiguration.pcaclockfrequencyintegervalue" value="255208.333333"/>
```

Main.c

```
// Includes
#include <SI_EFM8BB1_Register_Enums.h>
                                               // SFR declarations
#include "InitDevice.h"
// $[Generated Includes]
// [Generated Includes]$
#define ONBOARD BTN P0 B2
#define STEPS 4u
#define BUTTON DEBOUNCE 64u
#define MAGIC FLAG OFF 0
enum {
 RELEASED.
 PRESSED,
enum {
 HANDLED,
 UNHANDLED
enum {
 STEP0,
 STEP25,
 STEP50,
 STEP75,
static uint8_t button_last_status = RELEASED;
static uint8 t button_status = RELEASED;
static uint8_t button_counter = 0;
static uint8_t led_step_status = HANDLED;
static uint8_t pwm_steps[STEPS] = {0u, 64u, 128u, 192u};
static uint8_t current_step = STEP0;
//-----
// 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)
  // $[SiLabs Startup]
  // [SiLabs Startup]$
// main() Routine
int main (void)
  // Call hardware initialization routine
  enter_DefaultMode_from_RESET();
  PCA0CPH0 = pwm_steps[current_step];
  while (1)
    // $[Generated Run-time code]
    // [Generated Run-time code]$
          if (!ONBOARD_BTN)
                   if (button_last_status == RELEASED)
                           button_last_status = PRESSED;
                   else if (button_counter < BUTTON_DEBOUNCE)</pre>
                   {
                           ++button_counter;
                   }
                   else
                   {
                           button_status = PRESSED;
                           button_counter = 0;
                   }
          }
          else
                   if (button_last_status == PRESSED)
                           button last status = RELEASED;
                   }
                   else if (button_counter < BUTTON_DEBOUNCE)</pre>
                   {
                           ++button_counter;
                   }
                  else
                   {
                           button_status = RELEASED;
                           button_counter = 0;
                   }
      if (button_status == PRESSED)
          TMR2L = 0;
          TMR2H = 0;
          TMR2CN0_TF2H = MAGIC_FLAG_OFF;
          while(!TMR2CN0_TF2H);
          TMR2CN0_TF2H = MAGIC_FLAG_OFF;
          led_step_status = UNHANDLED;
          button_status = RELEASED;
      }
      if (led_step_status == UNHANDLED)
          current_step = (current_step + 1) % STEPS;
          PCA0CPH0 = pwm_steps[current_step];
          led_step_status = HANDLED;
      }
  }
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- <u>Igen</u>
- Nem

A program működött:

- Igen
- Nem

3. feladat – LED fényerejének állítása potenciométerrel

Mérje folyamatosan az ADC-vel a kiegészítő panelen lévő potenciométert interrupt módban és a mért értékkel arányosan változtassa a LEDo kitöltési tényezőjét o ás ~100% között. A PWM frekvenciája legyen 100 Hz feletti. Az ADC kód legyen 8 bites és legyen balra igazítva. a Voltage Reference legyen 3,3 V. Elég a felső 8 bitet kiolvasni (ADCoH regiszter).

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"</pre>
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">
    cproperty object="ADC_0" propertyId="ABPeripheral.included" value="true"/>
    value="Left justified"/>
    <property object="ADC_0" propertyId="adc.configuration.enable8bitmode" value="8-Bit mode"/>
<property object="ADC_0" propertyId="adc.configuration.gaincontrol" value="1x gain"/>
<property object="ADC_0" propertyId="adc.configuration.resolution" value="8-bit"/>
    value="ADC0.15 (P1.7)"/>
    cproperty object="ADC_0" propertyId="adc.view.view" value="Advanced"/>
    cproperty object="CROSSBAR0" propertyId="xbar0.pca0.cex" value="CEX0"/>
    <property object="DefaultMode" propertyId="mode.diagramLocation" value="100, 100"/>
    propertyId="interrupt.extendedinterruptenable1.enableadc0conversioncompleteinterrupt" value="Enabled"/>
    <property object="INTERRUPT_0" propertyId="interrupt.interruptenable.enableallinterrupts"</pre>
value="Enabled"/>
    <property object="P0.0" propertyId="ports.settings.skip" value="Skipped"/>
    <property object="P0.1" propertyId="ports.settings.skip" value="Skipped"/>
<property object="P0.2" propertyId="ports.settings.skip" value="Skipped"/>
    cproperty object="P0.3" propertyId="ports.settings.skip" value="Skipped"/>
    <property object="P0.7" propertyId="ports.settings.skip" value="Skipped"/>
<property object="P1.0" propertyId="ports.settings.skip" value="Skipped"/>
    <property object="P1.0" propertyId="ports.settings.skip" value="Skipped"/>
<property object="P1.1" propertyId="ports.settings.skip" value="Skipped"/>
<property object="P1.2" propertyId="ports.settings.skip" value="Skipped"/>
cproperty object="P1.3" propertyId="ports.settings.skip" value="Skipped"/>
cproperty object="P1.4" propertyId="ports.settings.iomode" value="Digital Push-Pull Output"/>
cproperty object="P1.4" propertyId="ports.settings.outputmode" value="Push-pull"/>
```

```
<property object="P1.7" propertyId="ports.settings.inputmode" value="Analog"/>
  <property object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar" value="Enabled"/>
  cproperty object="PCACH 0"
propertyId="pcach.pcachannelcapturecomparemode.enablechannelcomparatorfunction" value="Enabled"/>
  cproperty object="PCACH_0"
property Id = "pcach.pcachannel \overline{l} capture compare mode.enable channel pulse width modulation moden"
value="Enabled"/>
  <property object="PCACH_0" propertyId="pcach.pcacontrol.channelcapturecomparemode"</pre>
value="Predefined 8~11-bit pulse modulator"/>
  value="255.208 kHz"/>
  cproperty object="PCA_0"
propertyId="pca.pcacountertimerconfiguration.pcaclockfrequencyintegervalue" value="255208.333333"/>
  uS"/>
  <property object="PCA_0" propertyId="pca.pcacountertimerruncontrol.pcarun" value="Start"/>
  Running"/>
  value="4"/>
  cproperty object="TIMER16_2" propertyId="timer16.initandreloadvalue.timerreloadvalue"
value="1734"/>
  <property object="TIMER16_2" propertyId="timer16.reloadhighbyte.reloadhighbyte" value="6"/>
<property object="TIMER16_2" propertyId="timer16.reloadlowbyte.reloadlowbyte" value="198"/>
cproperty object="TIMER16_3" propertyId="ABPeripheral.included" value="true"/>
  <property object="TIMER SETUP 0" propertyId="ABPeripheral.included" value="true"/>
  value="Unregulated VDD"/>
  </mode>
 <modeTransition>
  cproperty object="RESET → DefaultMode" propertyId="modeTransition.source" value="RESET"/>
  cproperty object="RESET → DefaultMode" propertyId="modeTransition.target"
value="DefaultMode"/>
 </modeTransition>
</device:XMLDevice>
```

Interrupts.c

```
PCA0CPH0 = ADC0H;
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- <u>Igen</u>
- Nem

A program működött:

- <u>Igen</u>
- Nem

Megjegyzések

A második feladatban megoldani azt, hogy a Timer 2 pont akkor kezdődjön és pergésmentes is legyen azért nem kis meló volt, remélem tetszik. :D