

# ADC használata, feszültségmérés

MICLAB-04

Név: Pilter Zsófia, Vad Avar

Dátum: 2024.10.07.

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 03.

## 1. feladat – Potenciométer állásának mérése

Digitalizálja a kiegészítő boardon lévő potenciométer állását polling mód használatával (szoftveres időzítéssel). A Vref legyen a VDD (3,3 V). Ügyeljen rá, hogy az ADC belső órajele a lehető legnagyobb frekvenciájú legyen, de maximum 12,250 MHz. Az A/D konvertert 10 bites módban használja és legyen jobbra igazítva. A mért ADC kódot számolja át százalékba, így a potenciométer egyik szélső állásához 0% a másik szélső állásához 100% fog tartozni. Az eredményt debug módban a Variables ablakban lehet megtekinteni.

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="ADC_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="ADC_0" propertyId="adc.configuration.gaincontrol" value="1x
gain"/>
```

```

        <property object="ADC_0" propertyId="adc.configuration.sarclkdiv" value="0"/>
        <property object="ADC_0" propertyId="adc.control.enableadc" value="Enabled"/>
        <property object="ADC_0"
propertyId="adc.multiplexerselection.positiveinputselection" value="ADC0.15
(P1.7)"/>
        <property object="ADC_0" propertyId="adc.view.view" value="Advanced"/>
        <property object="DefaultMode" propertyId="mode.diagramLocation" value="100,
100"/>
        <property object="P0.6" propertyId="ports.settings.inputmode" value="Analog"/>
        <property object="P0.6" propertyId="ports.settings.iomode" value="Analog
I/O"/>
        <property object="P0.6" propertyId="ports.settings.skip" value="Skipped"/>
        <property object="P1.7" propertyId="ports.settings.inputmode" value="Analog"/>
        <property object="P1.7" propertyId="ports.settings.iomode" value="Analog
I/O"/>
        <property object="P1.7" propertyId="ports.settings.skip" value="Skipped"/>
        <property object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar"
value="Enabled"/>
        <property object="VREF_0" propertyId="ABPeripheral.included" value="true"/>
        <property object="VREF_0" propertyId="vref.hidden.voltagereferenceselect"
value="VDD pin"/>
        <property object="VREF_0"
propertyId="vref.voltagereferencecontrol.selectvoltagereference"
value="Unregulated VDD"/>
        <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>

```

## Main.c:

```

//=====
// src/feladat05-01_main.c: generated by Hardware Configurator
//
// This file will be updated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!!
//=====

//-----
// Includes
//-----
#include <SI_EFM8BB1_Register_Enums.h> // SFR declarations
#include "InitDevice.h"

#define ADC_MAX 1024.0f

```

```

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

//-----
// 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)
{
    // $[SiLabs Startup]
    // [SiLabs Startup]$
}

uint16_t ADCconv(void)
{
    uint16_t adcData;
    ADC0CN0_ADINT = 0;          // clear flag
    ADC0CN0_ADBUSY = 1;        // start conversion
    while (!ADC0CN0_ADINT);     // wait for end of conversion
    adcData = ADC0;
    return adcData;
}

float ADCtoPer(uint16_t d)
{
    float adcDataPer = 0.0f;
    adcDataPer = d / ADC_MAX;
    return adcDataPer;
}

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

    while (1)
    {
        ADCtoPer(ADCconv());
        // $[Generated Run-time code]
        // [Generated Run-time code]$
    }
}

```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

## 2. feladat – Potenciométer feszültségének mérése I

Módosítsa az előző programot úgy, hogy a Variables ablakban a potenciométer kimenő feszültségét számolja ki mV egységben. (A potenciométer egy feszültségosztónak tekinthető, aminek a bemenetére 3,3 V van kötve. A feszültségosztó képletére nincs szükség, csak az ADC kód-feszültség átalakítás képletére.)

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="ADC_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="ADC_0" propertyId="adc.configuration.gaincontrol" value="1x
gain"/>
    <property object="ADC_0" propertyId="adc.configuration.sarclockdivider"
value="0"/>
    <property object="ADC_0" propertyId="adc.control.enableadc" value="Enabled"/>
    <property object="ADC_0"
propertyId="adc.multiplexerselection.positiveinputselection" value="ADC0.15
(P1.7)"/>
    <property object="ADC_0" propertyId="adc.view.view" value="Advanced"/>
    <property object="DefaultMode" propertyId="mode.diagramLocation" value="100,
100"/>
    <property object="P1.7" propertyId="ports.settings.inputmode" value="Analog"/>
    <property object="P1.7" propertyId="ports.settings.iomode" value="Analog
I/O"/>
    <property object="P1.7" propertyId="ports.settings.skip" value="Skipped"/>
    <property object="PBCFG_0" propertyId="pbcfg.settings.enablecrossbar"
value="Enabled"/>
    <property object="VREF_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="VREF_0" propertyId="vref.hidden.voltagereferenceselect"
value="VDD pin"/>
    <property object="VREF_0"
propertyId="vref.voltagereferencecontrol.selectvoltage" value="Unregulated VDD"/>
    <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"/>
  </modeTransition>
</device:XMLDevice>
```

```

        <property object="RESET &#x2192; DefaultMode"
propertyId="modeTransition.target" value="DefaultMode"/>
    </modeTransition>
</device:XMLDevice>

```

## Main.c:

```

//=====
// src/feladat05-02_main.c: generated by Hardware Configurator
//
// This file will be updated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!!
//=====

//-----
// Includes
//-----
#include <SI_EFM8BB1_Register_Enums.h>           // SFR declarations
#include "InitDevice.h"
// $[Generated Includes]
// [Generated Includes]$

#define ADC_MAX 1024.0f
#define VDD 3300.0F

//-----
// 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)
{
    // $[SiLabs Startup]
    // [SiLabs Startup]$
}
uint16_t ADCconv(void)
{
    uint16_t adcData;
    ADC0CN0_ADINT = 0;           // clear flag
    ADC0CN0_ADBUSY = 1;         // start conversion
    while (!ADC0CN0_ADINT);      // wait for end of conversion
    adcData = ADC0;
    return adcData;
}

float ADCToMv(uint16_t d)
{
    float adcDataMv = 0.0f;
    adcDataMv = d * VDD/ADC_MAX;
    return adcDataMv;
}

//-----
// main() Routine
// -----

```

```

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

    while (1)
    {
        ADCtoMv(ADCconv());
        // $[Generated Run-time code]
        // [Generated Run-time code]$
    }
}

```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

### 3. feladat – Potenciométer feszültségének mérése II

Digitalizálja a kiegészítő boardon lévő potenciométer állását interrupt mód használatával, 50 Hz mintavételi rátával, Vref legyen a VDD (3,3 V). Ügyeljen rá, hogy az ADC belső órajele a lehető legnagyobb frekvenciájú legyen, de maximum 12,250 MHz. A megszakításkezelő függvényben csak az A/D konverter adatának változóba mentése történjen. Az A/D konvertert 10 bites módban használja és legyen jobbra igazítva. A mért ADC kódot alakítsa át mV egységbe. Az eredményt debug módban a Variables ablakban lehet megtekinteni.

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="ADC_0" propertyId="ABPeripheral.included" value="true"/>
        <property object="ADC_0" propertyId="adc.configuration.gaincontrol" value="1x
gain"/>
        <property object="ADC_0" propertyId="adc.configuration.sarclkdiv" value="0"/>
        <property object="ADC_0" propertyId="adc.control.enableadc" value="Enabled"/>
        <property object="ADC_0" propertyId="adc.control.startofconversion"
value="Timer 2 overflow"/>
        <property object="ADC_0"
propertyId="adc.multiplexerselection.positiveinputselection" value="ADC0.15
(P1.7)"/>
        <property object="ADC_0" propertyId="adc.view.view" value="Advanced"/>

```

```

    <property object="DefaultMode" propertyId="mode.diagramLocation" value="100,
100"/>
    <property object="INTERRUPT_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.extendedinterruptenable1.enableadc0conversioncompleteinterru
pt" value="Enabled"/>
    <property object="INTERRUPT_0"
propertyId="interrupt.interruptenable.enableallinterrupts" value="Enabled"/>
    <property object="P1.7" propertyId="ports.settings.inputmode" value="Analog"/>
    <property object="P1.7" propertyId="ports.settings.iomode" value="Analog
I/O"/>
    <property object="P1.7" propertyId="ports.settings.skip" value="Skipped"/>
    <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.enablelowbyteinterrupt" value="Enabled"/>
    <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="50"/>
    <property object="TIMER16_2"
propertyId="timer16.initandreloadvalue.timerreloadvalue" value="60432"/>
    <property object="TIMER16_2"
propertyId="timer16.reloadhighbyte.reloadhighbyte" value="236"/>
    <property object="TIMER16_2" propertyId="timer16.reloadlowbyte.reloadlowbyte"
value="16"/>
    <property object="TIMER16_3" propertyId="ABPeripheral.included" value="true"/>
    <property object="TIMER_SETUP_0" propertyId="ABPeripheral.included"
value="true"/>
    <property object="TIMER_SETUP_0" propertyId="timer_setup.timer0.mode"
value="Mode 2, 8-bit Counter/Timer with Auto-Reload"/>
    <property object="VREF_0" propertyId="ABPeripheral.included" value="true"/>
    <property object="VREF_0" propertyId="vref.hidden.voltagereferenceselect"
value="VDD pin"/>
    <property object="VREF_0"
propertyId="vref.voltagereferencecontrol.selectvoltagereference"
value="Unregulated VDD"/>
    <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>

```

Main.c:

```

//=====
// src/feladat05-03_main.c: generated by Hardware Configurator
//
// This file will be updated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!!
//=====

//-----
// Includes
//-----
#include <SI_EFM8BB1_Register_Enums.h>           // SFR declarations
#include "InitDevice.h"
// $[Generated Includes]
// [Generated Includes]$

#define ADC_MAX 1024.0f
#define VDD 3300.0F

volatile uint16_t adcData = 0U;
//-----
// 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)
{
    // $[SiLabs Startup]
    // [SiLabs Startup]$
}

float ADCToMv(uint16_t d)
{
    IE_EA = 0;
    float adcDataMv = 0.0f;
    adcDataMv = d * VDD/ADC_MAX;
    IE_EA = 1;
    return adcDataMv;
}

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

    while (1)
    {
        ADCToMv(adcData);
        // $[Generated Run-time code]
        // [Generated Run-time code]$
    }
}

```



```
}
```

Interrupts.c:

```
//=====
// src/Interrupts.c: generated by Hardware Configurator
//
// This file will be regenerated when saving a document.
// leave the sections inside the "$[...]" comment tags alone
// or they will be overwritten!
//=====

// USER INCLUDES
#include <SI_EFM8BB1_Register_Enums.h>

#define ADC_MAX 1024.0f
#define VDD 3300.0F

extern uint16_t adcData;

//-----
// ADC0EOC_ISR
//-----
//
// ADC0EOC ISR Content goes here. Remember to clear flag bits:
// ADC0CN0::ADINT (Conversion Complete Interrupt Flag)
//
//-----
SI_INTERRUPT (ADC0EOC_ISR, ADC0EOC_IRQn)
{
    ADC0CN0_ADINT = 0;
    adcData = ADC0;
}
```

Az elkészült programot be kell mutatni!

A gyakorlatvezető ellenőrizte:

- Igen
- Nem

A program működött:

- Igen
- Nem

## Megjegyzések