EE4930 Advanced Microprocessor

Section 011/021, Winter 2022/23

Professor: Kerry R. Widder, Ph.D.

Electrical Engineering and Computer Science Department

Milwaukee School of Engineering

Laboratory 5 : “Low Power”

Name: Jorge Jurado-Garcia

Courses: EE 4930 011

Date: 01/27/2023

# Objectives

This lab objective is to implement the MSP432 in low power mode to measure the temperature of an eagle nest for a minimum of 70 days.

# Description

For this lab, I implemented the low power mode to use LPM3. To enter LPM3 I had to set the SCB peripheral SCR register to SLEEPDEEP\_MASK. This had to be set before entering sleep mode. The temperature sensor will not work when in LPM3 mode only the Watchdog timer will work and can also allow us to wake up from the sleep mode. This was setup for the MSP432 to wakeup every 4 minutes and start the analog to digital conversion. End of sequence was set for the A2D converter to save power consumption and time as well as pins not being used to have a pull-down resistor to further lower the power consumption.

# Conclusion

This lab was a success, I decided to use the watchdog timer since it seemed more easier to work with rather than the RTC clock signal. During the process of implementing the low energy system I had trouble with getting the amount of energy being used above 1 month. This was manly due to not turning off my GPIO pins to pulldown mode and having extra wires connected to my MSP board. These extra pins would draw current from the MCU and increase the current consumption of the board. Once I disconnected any unused pins and disabled most of my peripherals the battery increased by 15 days. Another main issue that I forgot to keep in mind was when running the trace model, I was always in debug mode instead of free run mode. Being in debug mode will use more energy needed for the JTAG circuitry and limit the possible battery life of the board. Overall, I was able to get the MSP432 battery life to **2 years and 22 days**. When attempting to gather data from my MCU I was not able to correctly gather the impulses from my MSPboard due to SWDO connection error. Therefore, I have decided to not do ignore this section due to time constraints. But showed the overall equation needed to calculate the battery life.

# Attachments

## Schematic

Diagram, schematic

Description automatically generated

## Energy Trace

### Watchdog Timer 4 minutes

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, chart

Description automatically generated with medium confidence

### Watchdog Timer(16 seconds)

Graphical user interface, application

Description automatically generated

A picture containing chart

Description automatically generated

A screenshot of a computer

Description automatically generated

## Scope Traces

None

## Calculations

Equation to Use:

Where:

B­I­ = Battery lifetime in hours

B­c ­= Batter capacity in mAh

L­a­ = Device Current Consumption when awake in mA

L­s­ = Device Current Consumption when in Sleep mode in mA

t­a­ = Time spent awake in seconds

t­s­ = Time spent in Sleep mode in seconds.

## Main

/\* Copyright (C) 2022 MSOE

\*

\* All Rights Reserved

\* You may not use, distribute or modify this code without the

\* express written permission of MSOE

\*

\* Contact Info

\* jorgejuradogarcia2@gmail.com

\* 608-312-5950

\*

\*/

**#include** <stdio.h>

**#include** "msp432.h"

**#include** "msoe\_lib\_all.h"

**#include** "defines.h"

**#include** <MSP432P401R\_GPIO.h>

//Preprocessors

**#define** LAB5\_ON TRUE

**#define** CLEAR 21

**#if** LAB5\_ON

**#include** "EE4930\_LAB5.h"

//global variables for Lab5

**volatile** **float** temp=0;

**volatile** **int** adc\_value=0;

**volatile** **char** Ready = FALSE;

// Defines

**#define** MAX\_12BIT\_ADC\_RANGE 4096

**#endif**

/\*

\* main.c

\*/

**int** **main**(**void**){

**#if** LAB5\_ON

WDT\_A->CTL = WDT\_A\_CTL\_PW | WDT\_A\_CTL\_HOLD;

// set unused pins to pullup/down enabled to avoid floating inputs

P1->REN |= 0xFF;

P2->REN |= 0xFF;

P3->REN |= 0xFF;

P4->REN |= 0xFF;

P5->REN |= 0xFF;

P6->REN |= 0xFF;

P7->REN |= 0xFF;

P8->REN |= 0xFF;

P9->REN |= 0xFF;

P10->REN |= 0xFF;

Set\_ports\_to\_out();

Init\_PCM(); //setup for power control module

init\_CS(); //setup for clock system

//Init the watchdog timer

Init\_Watchdog();

//Init the Temperature Sensor

Init\_Temp();

//Interrupt setup

NVIC->ISER[0] |= ( 1 << *ADC14\_IRQn* ); //NVIC for ADC14 at ISER[24]

NVIC->ISER[0] |= ( 1 << *WDT\_A\_IRQn*); //NCI for Watchdog atISER[3]

\_\_enable\_interrupt();

ADC14->CTL0 |= ADC14\_CTL0\_SC; // start a new ADC conversion

//Processor Peripherals

//System control register

SCB->SCR = SCB\_SCR\_SLEEPDEEP\_Msk;

**while**(1)

{

\_\_wfi(); // wait in LPM3 for watchdog

**while**(!Ready);

Ready = FALSE;

P4->OUT &= ~BIT5; // end indication

**printf**("Temperature Value: %f F\n", temp);

}

**#endif**

} // end main

**#if** LAB5\_ON

**void** **ADC14\_IRQHandler**(**void**)

{

**int** readIV = ADC14->IV; // Reading the IV register to cl ear

ADC14->CLRIFGR0 = ADC14\_CLRIFGR0\_CLRIFG0;

adc\_value = ADC14->MEM[3];

**float** adc\_voltage = ( ( (**float**)(adc\_value)/MAX\_12BIT\_ADC\_RANGE ) \* (3300.0f));

**float** temp\_c = adc\_voltage/10 - 50; // y = x10mV/C - 75

temp = ( temp\_c \* (9.0f / 5.0f) ) + 32.0;

Ready = TRUE;

}

// Interrupt Handler for the watchdog timer

**void** **WDT\_A\_IRQHandler**(**void**)

{

// wakes up from LPM3 into LPM0

P4->OUT |= BIT5; // indicate temp reading is available

//start a new conversion

//sleep the board

ADC14->CTL0 |= (ADC14\_CTL0\_ENC | ADC14\_CTL0\_SC); // start a new ADC conversion

}

**#endif**

## EE4930\_LAB5.c

/\*

\* EE4930\_LAB5.c

\*

\* Created on: Jan 22, 2023

\* Author: jurado-garciaj

\*/

// Included files

**#include** <stdio.h>

**#include** "msp432.h"

**#include** "defines.h"

**#include** "msoe\_lib\_misc.h"

// Drivers

**#include** <MSP432P401R\_GPIO.h>

**void** **Init\_Watchdog**( **void**)

{

//Configure WDT\_A

// use password must be written or the WDT resets

// Using the BCLK clock source

// Setting watchdog as interval timer mode

//Clear count

WDT\_A->CTL = WDT\_A\_CTL\_PW | WDT\_A\_CTL\_SSEL\_\_BCLK | WDT\_A\_CTL\_TMSEL

| WDT\_A\_CTL\_CNTCL // clear the count

| WDT\_A\_CTL\_IS\_2;

**return**;

}

**void** **init\_CS**( **void** )

{

//what until the PCM control module has finished setting up

**while**( (PCM->CTL1 & 0x100) == 0x100 );

CS->KEY = CS\_KEY\_VAL; //unlock

CS->CTL1 = CS\_CTL1\_SELM\_\_REFOCLK | CS\_CTL1\_SELS\_\_REFOCLK

| CS\_CTL1\_SELA\_\_REFOCLK; // source REFOCLK for BCLK

CS->CTL0 = 0x010000;

CS->CTL2 = 0x010003;

CS->CTL3 = 0x0000BB;

CS->KEY = 0;

**while**( (CS->STAT & 0x02000000) == 0 );

**return**;

}

//Power control module

/\*

\* Allows for the optimization of power

\* This can work in conjunction with the Clock System

\* to control the power setting of the device and hence consumption

\* All peripherals and processors go to the PCM and then translate to the

\* Power Supply System and ClockSystem

\* Different Power Modes, Active MOde, Sleep Mode LPM0, LPM3 LPM4

\* For the lowest power consumption possible one must put the MCU in LPM3.5 or

\* LPM4.5 this only allows the board to have the WDT and RTC peripheral on.

\*/

**void** **Init\_PCM**( **void** )

{

// unlock PCM/PMR

// Entering LMPM3 when setting up the peripheral

// Then when only activating the watchdog timer going into LPM4

// Optional state is also LPM3.5 instead.

//Watchdog is possible in LPM3.5

// Operating state LDO\_VCORE0 setting set to 0

// Power Mode LPMR\_LPM3

// This is for Peripherals enabling

PCM->CTL0 = PCM\_CTL0\_KEY\_VAL | PCM\_CTL0\_LPMR\_\_LPM3 |

PCM\_CTL0\_CPM\_\_AM\_LF\_VCORE0 |PCM\_CTL0\_AMR\_\_AM\_LF\_VCORE0;

**return**;

}

**void** **Init\_Temp**( **void** )

{

// Potentiometer for Room Temperature

//A0

GPIO\_setAsInputPin(GPIO\_PORT\_P5, GPIO\_PIN2);

// input on A3 into MEM0

P5->SEL0 |= BIT2; // use with A3

P5->SEL1 |= BIT2; // use with A3

//setup the ADCPeripheral

// Setting up the control register

ADC14->CTL0 |= ADC14\_CTL0\_SHT0\_5 | ADC14\_CTL0\_SHP | ADC14\_CTL0\_SSEL\_\_MCLK

| ADC14\_CTL0\_ON | ADC14\_CTL0\_CONSEQ\_0;

// Sampling time, S&H=96, ADC14 on, SMCLK, with 14 Bit Resolutions

// make sure read the reset operation first

// CSTARTADDx this gets the start address; these bits select which

//ADC14 conversion

// memory registers is used for a single conversion or for the first

//conversion in a sequence.

// setting the memory to MEM0

ADC14->CTL1 &= ~(ADC14\_CTL1\_RES\_2 | ADC14\_CTL1\_RES\_1);

ADC14->CTL1 |= (ADC14\_CTL1\_RES\_2|ADC14\_CTL1\_PWRMD\_2);

ADC14->CTL1 |= (3 << ADC14\_CTL1\_CSTARTADD\_OFS); //Using MEM3

//Memory conversion control 0 register being set to get data for A3

ADC14->MCTL[3] |= ADC14\_MCTLN\_INCH\_3|ADC14\_MCTLN\_EOS;

//EOS help to make sue the other channels do not get commuted

// Enabled interrupt information for IER0 register

ADC14->IER0 |= ( ADC14\_IER0\_IE3 );

// enable the conversion

ADC14->CTL0 |= ADC14\_CTL0\_ENC;

//use pin

// pin P4.5

GPIO\_setAsOutputPin( GPIO\_PORT\_P4, GPIO\_PIN5 );

GPIO\_setOutputLowOnPin( GPIO\_PORT\_P4, GPIO\_PIN5 );

}

## EE4930\_LAB5.h

/\*

\* EE4930\_LAB5.h

\*

\* Created on: Jan 22, 2023

\* Author: jurado-garciaj

\* Specifications:

\* The code must use the A/D to measure temperature in degrees F

\* (Resolution of 0.1 deg. F min.)

\*

\* Using a TMP36 sensor. The temperature reading must be stored global

\* Variable that the wireless radio can access, and the radio must be

\* Notified that a new value is available for transmission

\* High for some short time interval, then back low).

\* Temperature readings taken about every ten minutes transmitted

\* To the base station.

\*

\* Battery is a CR2032 lithium battery with a capacity of 210 mAh.

\*

\* Use the Energy Trace tool to analyze your code in terms of its time spent in various modes.

\*

\* Use a series resistor (e.g., 10 Ohms) in the power connection to measure the

actual power supply current of the system on an oscilloscope

\*

\* Calculate the estimated battery life of your system using scope data.

\*/

**#ifndef** APP\_EE4930\_LAB5\_H\_

**#define** APP\_EE4930\_LAB5\_H\_

//use an ADC converter to measure the temp connect to Pin5.2

**void** **Init\_Temp**( **void** );

**void** **Init\_Watchdog**( **void**);

**void** **init\_CS**( **void** );

**void** **Init\_PCM**( **void** );

**#endif** /\* APP\_EE4930\_LAB5\_H\_ \*/