Portland State UNIVERSITY

Erebus Labs

STEM SENSORS

SOFTWARE PLAN

Version 1.0

1/22/2014

SCOTT LAWSON

BRYAN BUTTON

CHRIS CLARY

MAX COPE

VERSION HISTORY

Version #	Implemented By	Revision Date	Reason
1.0	Scott Lawson	1/22/2014	Initial Release

NOTE TO READER

This is a template obtained from:

http://www2.cdc.gov/cdcup/library/templates/default.htm

Template Name: Product Design

UP Template Version: 12/31/07

TABLE OF CONTENTS

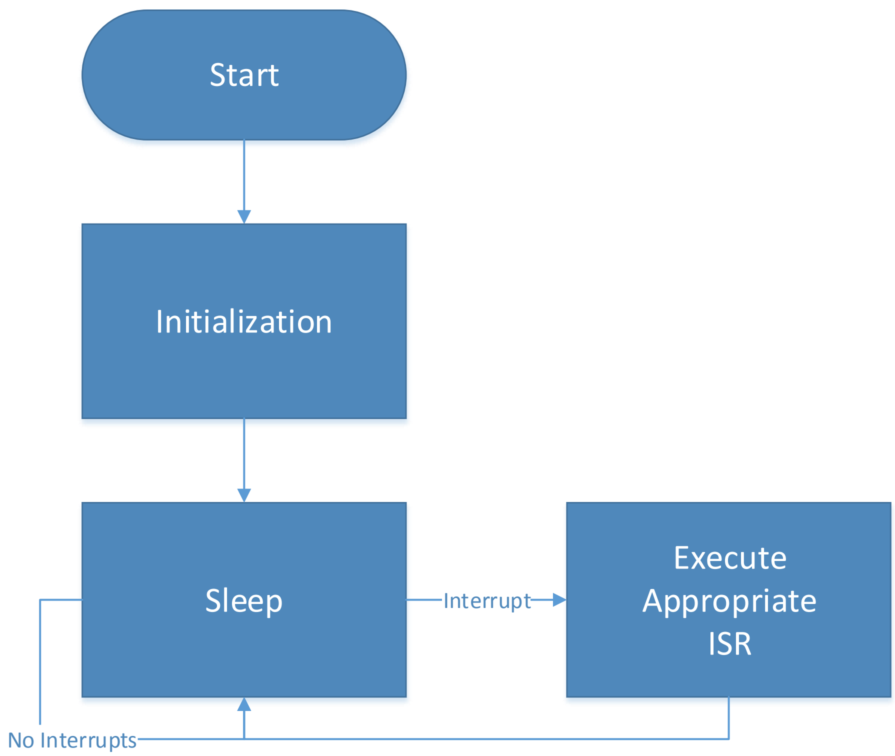
1 INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

This document is a high-level description of how the software is planned to be implemented for the Erebus Labs STEM Sensors device.

1.2 OVERVIEW

The Ereubs Labs STEM Sensors device firmware will be written in standard C compliant with ISO/IEC 9899:2011. When a microcontroller is selected for the device, the development environment will be specified here. Upon reset, the controller will operate as follows:



1.3 THEORY OF OPERATION

The STEM Sensor will contain a null-body loop as its MAIN function. All functionality will be interrupt-driven. The system will remain in sleep mode except when connected to a computer via a USB interface, or when sampling the ADC. The system will be woken from sleep by a watchdog timer to drive ADC sampling. Interrupt service routines are described in Section 3 of this document, and the user interface when connect to a computer is described in Section 4 of this document.

2 INITIALIZATION TASKS

After a restart, the chip must be initialized. There are several individual components that will require register modifications:

2.1 GENERAL

- Clocks
- Oscillators
- GPIOs including LEDs and buttons
- Real-time clocks

2.2 EEPROM

- Set pointers to key memory locations
- Ensure EEPROM can be globally accessed via functions
- I^2C/UART/SPI configuration if off-chip

2.3 USB

- Set Device / Interface Descriptors
- Enable USB interrupt assertions from USB controller

2.4 ANALOG TO DIGITAL CONVERTER

- Enumerate Sensors or retrieve sensor IDs from EEPROM
- Set Op-Amps appropriate for sensor
- Configure ADC clock

2.5 WATCHDOG TIMER

- Retrieve frequency settings from EEPROM
- Enable interrupt assertions
- Start timer

2.6 POWER MANAGEMENT

- Set sleep mode parameters
- Allow waking up from USB and watchdog timer

2.7 INTERRUPTS

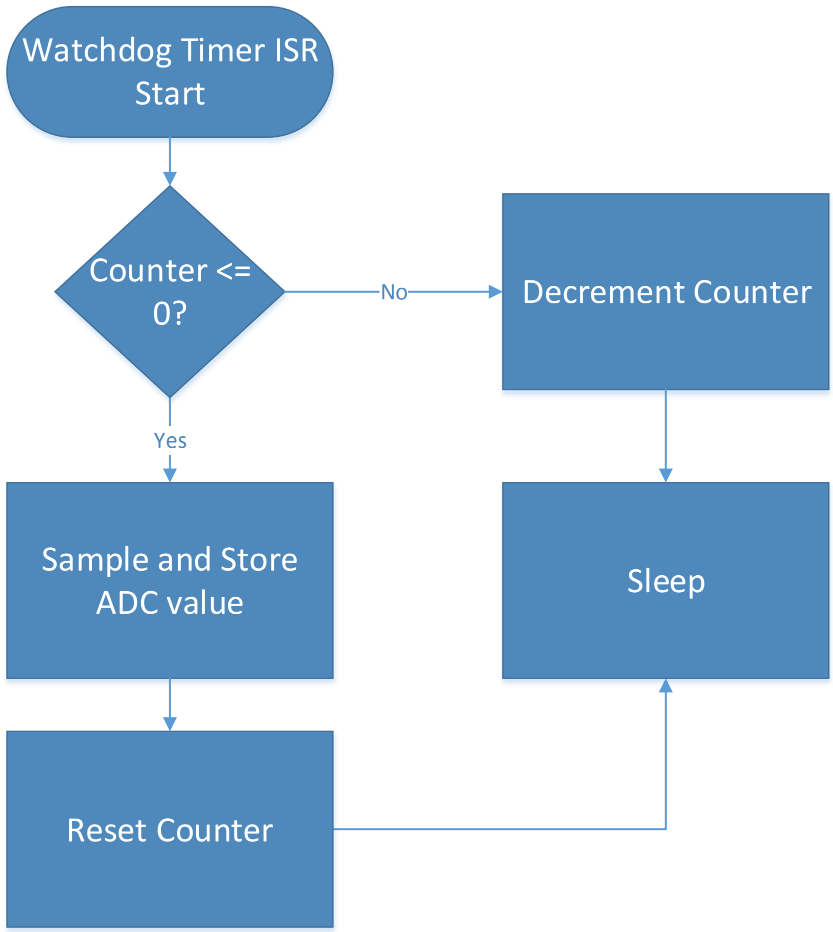
- Set USB, Low-power, and watchdog timer interrupts in interrupt controller
- Map interrupt vectors to ISRs
- Enable global interrupts

3 INTERRUPT SERVICE ROUTINES

3.1 WATCHDOG ISR

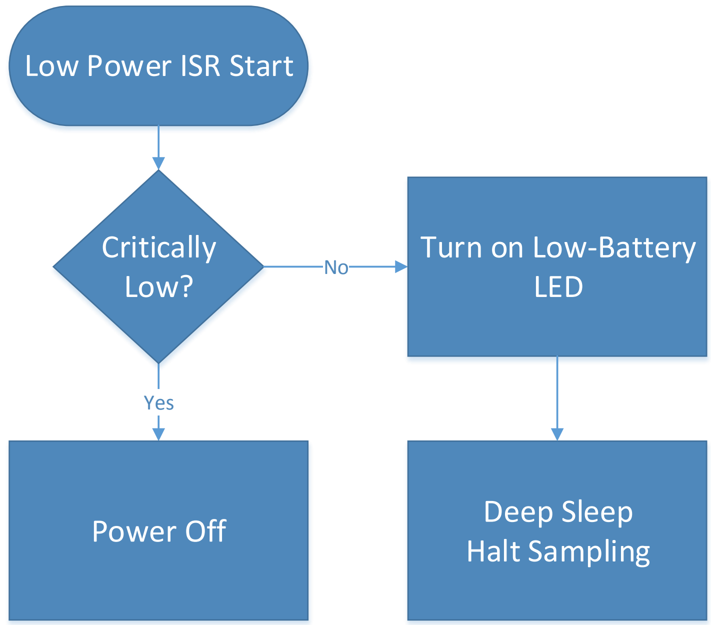
The microcontrollers under consideration for use in the system allow a watchdog timer to wake the system periodically with a period of between 1 second and ~20 seconds. To achieve the sampling frequency configured by the user, a counter will be maintained in SRAM. This counter will be decremented each time the system is woken from sleep. When the counter reaches zero, the ADC will be sampled, its value stored, and the counter reset.

The initial value of the counter will be equal to Sample Period / Clock Period.



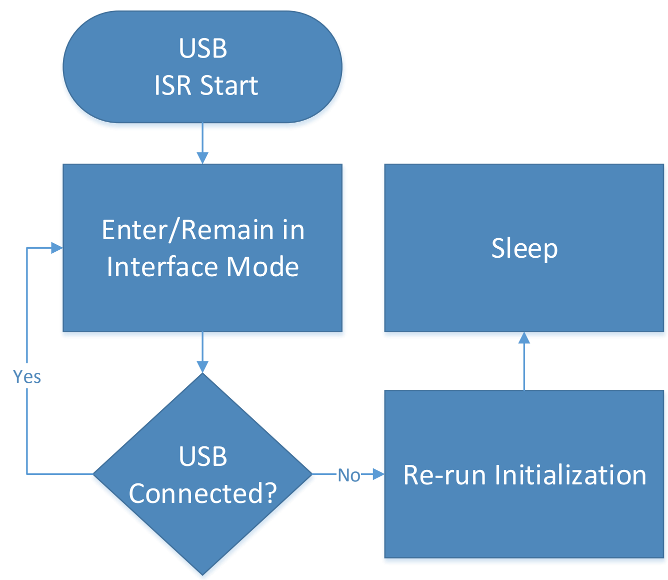
3.2 LOW POWER ISR

If the selected microcontroller is equipped, there will be a low-power interrupt asserted when the power supply drops below a designated threshold. The low power interrupt will determine if the system is critically low on power. If yes, the system will shut down entirely to protect the microcontroller from failure. If no, then turn on a warning LED to the user.



3.3 USB ISR

The system will enter an interface mode when plugged into a computer. The sampled data will be made available from the EEPROM, and the user will be able to reconfigure data collection settings.

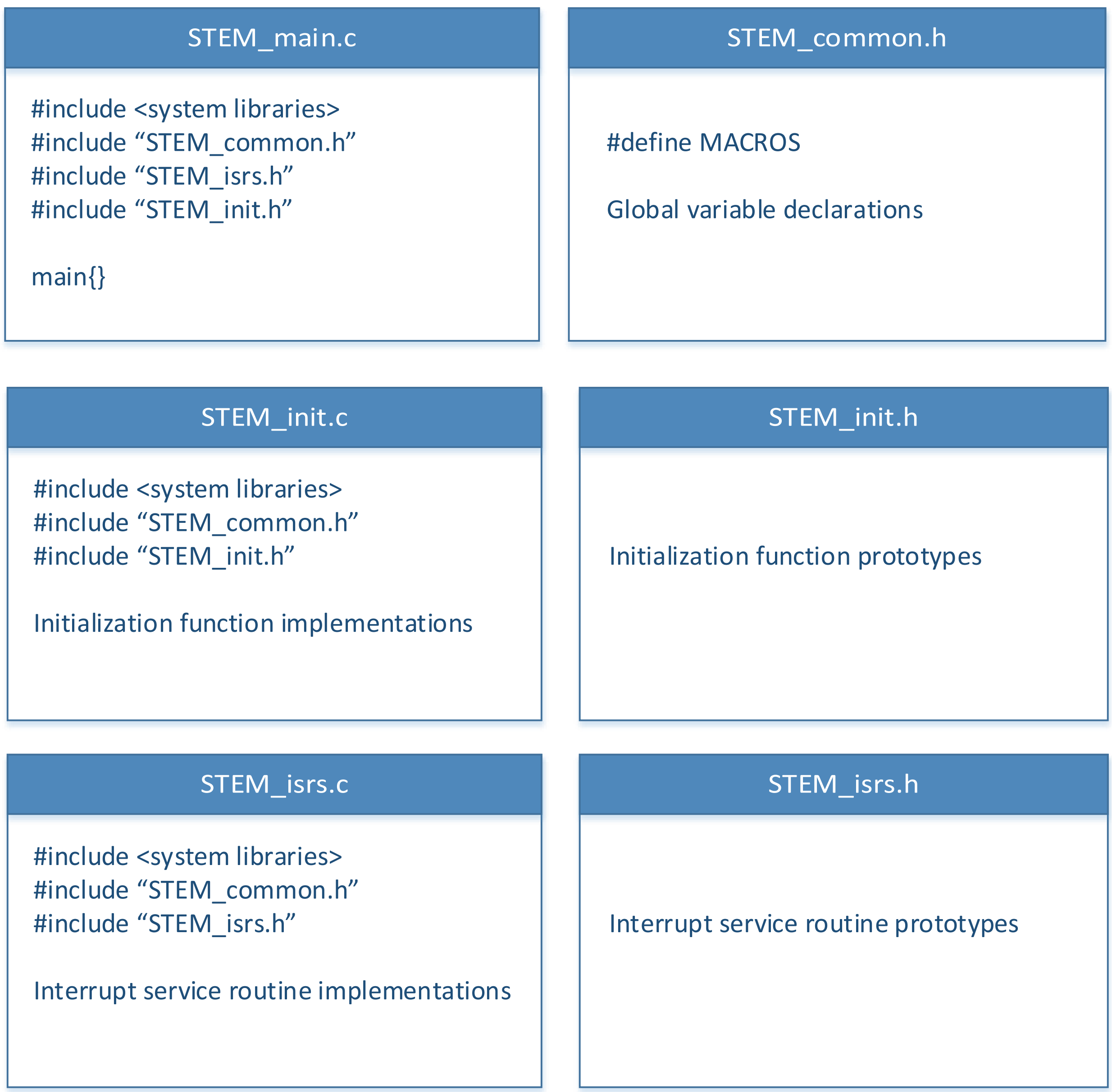


4 COMPUTER INTERFACE

The structure of the user interface is still under discussion as of the date of this writing. It will likely consist of a GUI that allows the user to download EEPROM data points and display simple graphs. Additionally, the data will be available in a raw .CSV file for importing to other applications suitable for more advanced students.

The user interface will also allow the user to configure settings such as the frequency and duration of data collection.

5 FIRMWARE FILE STRUCTURE



6 DATA STRUCTURES

To maximize the use of limited memory, the unsigned binary value read directly from the ADC will be stored. The system will not perform conversions or calculations with this data on board the base unit.

Once the microcontroller is selected, the encoding scheme will be described in this document. The encoding scheme will also contain an identifier for the sensor used and a date/time stamp from the real-time clock. The entire encoded data point will be uploaded to the computer when connected and the computer-based software will translate the data points into human-readable data.

APPENDIX A: TERMINOLOGY

A.1 ACRONYMS

Acronym	Meaning
ADC	Analog-to-Digital Converter
BOM	Bill of Materials
CO	Carbon Monoxide
EEPROM	Electrically Erasable Programmable Read-Only Memory
EPL	The Portland State University Engineering and Prototyping Lab
I^2C	The Inter-Integrated Circuit communication protocol
ISR	Interrupt Service Routine
K-12	Kindergarten through 12^{th} grade school
LED	Light Emanating Diode
PCB	Printed Circuit Board
SI	Silicon
SPI	Serial Peripheral Interface Bus
STEM	Science, Technology, Engineering and Math
USB	Universal Serial Bus

A.2 SYSTEM ARCHITECTURE

Base Unit

The central device that manages power, communication, and data storage, and has one or more sensors attached to it.

Sensor

The individual data collection devices such as VOC detectors and thermometers that are attached to the base unit.

User Interface

The program that will be run on a laptop or desktop computer that allows the user to view and interact with the data collected.

System

The operational product comprised of base units with attached sensors and a user interface.