Refridgerator Temperature Monitor &

Alarm System

System Specifications and Functional Description

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Date | Description of Change | Author |
| - |  | Initial Release | R.McMillin |
|  |  |  |  |

**Contents**

[1. Overview 4](#_Toc7328923)

[1.1. Purpose 4](#_Toc7328924)

[1.2. Basic Description 4](#_Toc7328925)

[2. Power Supply 5](#_Toc7328926)

[2.1. Voltage Input Range 5](#_Toc7328927)

[2.2. Reverse Polarity Protection 5](#_Toc7328928)

[2.3. Switch 5](#_Toc7328929)

[2.4. Current consumption estimation 5](#_Toc7328930)

[2.5. Voltage Sense 5](#_Toc7328931)

[3. MCU 5](#_Toc7328932)

[3.1. MCU 5](#_Toc7328933)

[3.2. Analog Voltage Source and Reference 5](#_Toc7328934)

[3.3. System clock 6](#_Toc7328935)

[3.4. Sleep/Low Power Mode 6](#_Toc7328936)

[3.5. I2C 6](#_Toc7328937)

[3.6. UART 6](#_Toc7328938)

[3.7. SPI 6](#_Toc7328939)

[3.8. PWM 6](#_Toc7328940)

[4. Real Time Clock 6](#_Toc7328941)

[5. Temperature Measurement 7](#_Toc7328942)

[6. Display Board 7](#_Toc7328943)

[6.1. Light Threshold Detection 7](#_Toc7328944)

[6.2. Segment LED display 7](#_Toc7328945)

[6.3. Power Switch 7](#_Toc7328946)

[7. Memory and Data Storage 7](#_Toc7328947)

[8. Wifi 7](#_Toc7328948)

# Overview

## Purpose

The purpose of this document is to specify how the system should function and explain in detail how the circuits work. It will also contain descriptions and math for passive part selection.

This document will be updated as the design changes.

## Basic Description

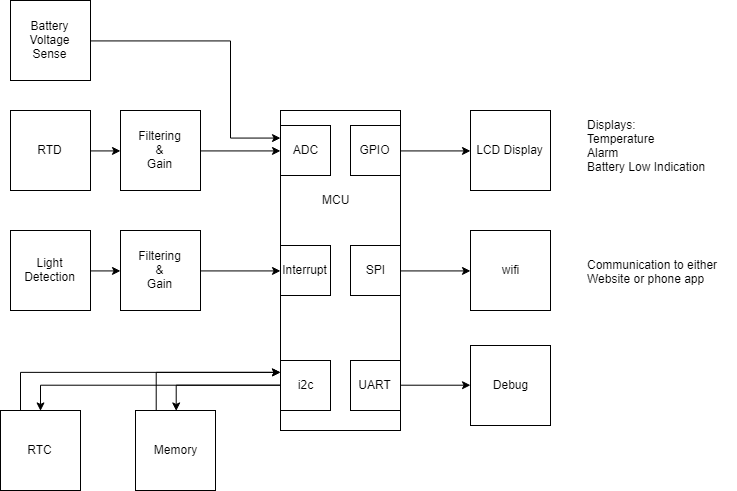


Figure 1 - Hardware Block Diagram

* All parts selected will have an operating temperature range of -40C to 85C or better value.
* MCU will measure temperature from an RTD
* Light detection, from opening the door, will trigger the MCU to wake up. This feature can be disable through the phone app should the freezer be in a bright area.
* Time stamps and measured temperature will be stored in EEPROM
* Battery voltage will be measured and an alert sent to email and the front panel if the voltage is too low
* The front panel will contain a power switch, 4 digit 7-segment display and a photodiode for light sense
* Wifi communication will be provided by the AtwinC1500 wifi module

# Power Supply

## Voltage Input Range

The system is designed to work over the range 1.8V to 3.3V. Two AA batteries in series should sufficiently power this device.

## Reverse Polarity Protection

Reverse polarity protection is provided by P channel mosfet, Q1.

## Switch

S1 is a panel mount switch that will toggle the power on and off

## Current consumption estimation

Include current consumption estimation, estimated power on time

|  |  |  |
| --- | --- | --- |
| Part | Current Consumption Active | Current Consumption Sleep |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Total |  |  |

Estimated run time

* 1 wake up per hour for 5 seconds
* 2AA batteries in series

Some Math

## Voltage Sense

The supply voltage will be measured by a voltage divider, R14 and R15. The output is not high enough to mee the minimum input current of the AVR ADC. Probably should consider an op.

# MCU

## MCU

The MCU that provides the logic in the control system is the Microchip Atmega328pb 8bit microcontroller. ADC, I2C, SPI, PWM, and UART peripherals are used.

## Analog Voltage Source and Reference

The analog voltage source is filtered by an inductor L1 and capacitor C3.

The ADC voltage reference of 1.225V is provided by shunt voltage reference D1. Current is limited by R4. The minimum current is ? and the maximum is ?

Some Math

## System clock

Precision clock speed is not required, so the internal 8MHz oscillator will be used and will be divided by 8 for a frequency of 1MHz. This keeps the MCUs speed under the maximum of 4MHz at 1.8V.

## Sleep/Low Power Mode

The MCU will spend most of its time in Sleep/Low Power mode to save battery power. The RTC will trigger periodic wake ups on the MCU’s interrupt pin. The photodiode will trigger an interrupt to wake up the MCU when the freezer/fridge door opens.

## I2C

The i2c bus runs at 100khz with 10k pull ups. This bus provides communication between the MCU and the RTC/EEPROM.

## UART

The uart pins will be broken out to a header for debug purposes

## SPI

The SPI protocol is used to communicate with the Wifi module

Unsure what frequency to use, SPI mode?

## PWM

PWM is used to dim the brightness of the 7 segment LEDs to conserve battery power.

# Real Time Clock

The real time clock triggers an interrupt on its MFP pin. After an interrupt, the RTC will update it’s alarm to increment by the period it checks the temperature.

The RTC is connected to the i2c bus and runs at 100khz

Crystal Load cap calculation:

Where:   
Ce = external capacitors (6 to 9pF optimized)  
CL = Crystal load capacitance 7pF (Y1)  
Cs = Stray capacitance, estimated 5pF

I haven’t decided if I will add battery backup.

# Temperature Measurement

The transducer to measure the temperature is a 100 ohm platinum RTD which is excited by a 500uA constant current source provided by U4.1, U5.1 and R10. R6=R7=R8=R9. R11 and C6 form a low pass filter with cutoff frequency 159Hz. This is far below the ADC sample frequency, but 10k and 0.1uF are used to save money and reducing the number of different passive values. U4.2 amplifies the voltage across the RTD. The working range of the RTD is -? To ? The min and max values stay away from the rails of the op amp. U5.2 is kept spare incase the design is changed to an external RTD and lead compensation is required.

# Display Board

## Light Threshold Detection

## Segment LED display

## Power Switch

# Memory and Data Storage

Data is stored in EEPROM that is connected to the i2c bus and runs at 100khz

|  |  |  |
| --- | --- | --- |
| Code | Variable Name | # of bits |
| SV | Supply Voltage | 10 |
| SC | Temperature | 10 |
| DT | Date/Time stamp | 32 |
| P | Parity bit | 1 |

Data storage Bit Field

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
| 1 | DT:Min:6 | DT:Min:5 | DT:Min:4 | DT:Min:3 | DT:Min:2 | DT:Min:1 | DT:Min:0 | DT:Hr:5 |
| 2 | DT:Hr:4 | DT:Hr:3 | DT:Hr:2 | DT:Hr:1 | DT:Hr:0 | DT:Day:5 | DT:Day:4 | DT:Day:3 |
| 3 | DT:Day:2 | DT:Day:1 | DT:Day:0 | DT:Mth:4 | DT:Mth:3 | DT:Mth:2 | DT:Mth:1 | DT:Mth:0 |
| 4 | DT:Yr:7 | DT:Yr:6 | DT:Yr:5 | DT:Yr:4 | DT:Yr:3 | DT:Yr:2 | DT:Yr:1 | DT:Yr:0 |
| 5 | SV:9 | SV:8 | SV:7 | SV:6 | SV:5 | SV:4 | SV:3 | SV:2 |
| 6 | SV:1 | SV:0 | T:9 | T:8 | T:7 | T:6 | T:5 | T:4 |
| 7 | T:3 | T:2 | T:1 | T:0 | CRC | CRC | CRC | CRC |

The total number of data points that can be store is ?

# Wifi