Smart Home Energy Monitor

Rodney Tayebwa & Remi Pilon

Course: NET3001 Computer Systems

Professor: Mehdi Niknam

Due on April 10th, 2024

Table of Contents

[Brief description of the system: 3](#_Toc163670679)

[Circuit Diagram: 3](#_Toc163670680)

[Components and topics used in the project: 4](#_Toc163670681)

[Detailed explanation of the system: 4](#_Toc163670682)

[Reflection: 7](#_Toc163670683)

# Brief description of the system:

The Smart Home Energy Monitor is an embedded system designed to monitor and optimize energy usage within a home. It tracks electricity consumption in real-time and provides users with insights to help them make informed decisions about energy usage and conservation.

This project aims to empower homeowners to monitor and manage their energy usage more effectively, promoting energy conservation and cost savings while reducing environmental impact.

# Circuit Diagram:

# Components and topics used in the project:

Mandatory Topics and Components:

GPIO: Used for interfacing with energy monitoring sensors and controlling output devices.

Timers: Scheduled tasks for data logging and reporting.

Interrupts: Handle events such as sensor readings or user inputs.

LEDs: Indicate energy consumption levels or system status.

Push Buttons: Enable user interaction for viewing data or configuring settings.

7-segment Display: Display real-time energy consumption data.

LCD: Display detailed energy usage statistics and system status.

Shift Registers: Expand GPIO capabilities for driving multiple LEDs or displays efficiently.

Optional Topic and Components:

USART: Enable communication with external devices or data logging systems.

Active or Passive Buzzers: Provide audible alerts for high energy consumption levels or system warnings.

Potentiometer: Allow users to adjust display brightness or contrast.

# Detailed explanation of the system:

Include necessary header files for AVR I/O, interrupts, LCD, and timer functionalities

Declare global variable:

received\_char as volatile char

Define functions:

usart\_init()

Set baud rate to 9600

Enable receiver and transmitter

Set frame format: 8 data bits, 1 stop bit

Enable USART receive interrupt

usart\_transmit(char data)

Wait for empty transmit buffer

Put data into buffer, sends the data

usart\_receive()

Wait for data to be received

Get and return received data from buffer

ISR(USART\_RX\_vect)

Update received\_char with received data

lcd\_init()

Initialize LCD in 4-bit mode

Set up display parameters

Wait for LCD initialization to complete

lcd\_cmd(unsigned char cmd)

Send a command to the LCD

lcd\_data(unsigned char data)

Send data to the LCD

lcd\_send\_nibble(unsigned char nibble)

Send a 4-bit data to the LCD

lcd\_trigger()

Send a trigger signal to the LCD

timer1\_init()

Initialize Timer1 with a prescaler of 64

Enable overflow interrupt

button\_init()

Initialize push buttons as inputs with pull-up resistors enabled

display7Segment(uint8\_t combinedState)

Display a combined state on a 7-segment display using a shift register

Main function:

Initialize USART, LCD, shift register pins, push buttons, LEDs, and Timer1

Enable global interrupts

Loop:

Control LEDs based on button states

Display button states on a 7-segment display

Transmit received USART data to the LCD

# Reflection:

During the planning stage of the project, we were torn between doing a smart energy monitor and a thermostat. We didn’t do the thermostat because we did not have the required components for it.

After going through the pros and cons of both systems we decided on a smart energy monitoring system. This is because the smart energy monitoring system allowed us to use different components of the Arduino kit and coding for the project wouldn’t have been as intense as it would be for the thermostat.

One challenge we experienced was trying to get the timers to respond to events within the specified time frames. Another problem we encountered was we were not able to get the buttons to work as well as the LCD screen. We tried using different LCDs to see if that was the problem and that worked.