Milestone 1: Communicating with Will Byers

Dario Colon, Joe Foderaro, Nick Gorab

Rowan University

December 3, 2017

1 Abstract

Inspired by the illuminated alphabet wall seen in the Netflix Original show ”Stranger Things”, the purpose of this lab is to implement the techniques and functions we have learned so far through the semester. Using an MSP430 microprocessor and a UART connection, a string of values must be decoded and then passed on, mimicking the glowing lights from ”Stranger Things.” Our processor will be daisy-chained with 25 other processors (For all 26 letters of the alphabet) in order to decode the value given and output any message that the ”Upside down” may throw at it. The assignment, with its specific requirements, brought about careful design choices which are to be discussed in this Application Note.

2 Introduction

Every day, everyone uses an embedded system. Whether it’s a refrigerator, an elevator, or really anything that starts with the word ”smart”, they are all around you. To put it simply, an embedded system is a combination of processors and electronic circuits that act as the middle-ground between a user and the real world, making life easier. The goal of this lab was to design a system that takes a user input message, and spells it out via a string of LEDs, each LED corresponding to a different letter. A more relatable example of this is a ticker message that can be found on a billboard or on the side of a building. A message is put in and signals are sent to an array of LEDs, spelling out any combination of letters that is processed. Furthermore, these types of systems do not only have to control LEDs. The signals can control a motor or a switch. The possibilities are endless and, as advanced as this project may seem, this is just the beginning.

3 Background

One main design decision that had to be made when starting this project was which processor to use. The Texas Instruments MSP430 series was a standard starting point but there is a wide variety of boards that fall under this class and they all have different specifications associated with them. Some have very fast processors (up to 25 MHz) and some have slower processors (1 MHz). Some have many capture/compare registers (MSP430F5529 has 5 registers) and some have not as many (most of the other boards have 3). The list goes on but the point is there are design constraints associated with this project.

One important design constraint that had to be taken into consideration was the amount of capture/compare registers available for use at one time. Due to the functionality of the code, at least 4 registers had to be simultaneously available. That alone eliminated several prospective boards that we could use for the final product. Between that and board speed, there was no better option than the MSP430F5529.

4 Evaluation and Results

4.1 Board Selection

The first step in designing the software for this lab was to select a microprocessor to use. There were many factors which went into this decision, with the main one being ease-of-use. After taking into account all of the possible options provided to us, the MSP430F5529 board was chosen.

This board was chosen due to the number of Capture/Compare registers it has to offer. The MSP430F5529 has five Capture/Compare registers under Timer_A, which are paramount when it comes to controlling the LEDs. Some of the other boards, like the MSP430G2253, only contained three of these registers which would have added significantly more to the program.

4.1.1 Software Architecture

The code which was implemented in this lab is fairly simple. It contains multiple functions which initialize the important parts of the code, and then a main function which executes all of them. A list of these functions is found below.

ledInit
: Establishes the correct pins as an output, and multiplexes the RGB LED pins with the pins relating to their respective Timer_A CCR registers.

uartInit
: Designates pins for UART RX/TX and sets BAUD rate to 9600. The source clock is set to SMCLK, and interrupts are enabled for UART.

pwmInit
Enables the clock and sets up the correct behavior for the CCR Registers. Clock is set to SMCLK, running in Up-Mode, with a pre-divider of 4.

main
Calls all of the previous functions, and then enters Low-Power Mode and enables the Global Interrupt.

The next main part of the code is the interrupt vector for UART. It is here where the data received is processed. There is a case statement which depends on the byte, and each state will process the data accordingly. For more information about the code, see the ”milestone1.c” file, as it is fully commented.

4.2 Hardware

In order to power the LEDs at their recommended values, external circuitry (”Convert-O-Box”) was required. This was done through the use of MOSFET transistors to drive the current for the LEDs. A circuit schematic can be found below in Figure 1.

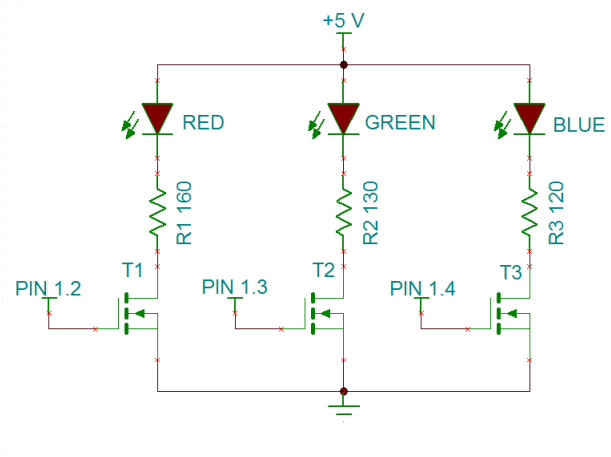

Figure 1: RGB LED External Circuitry.

MOSFETs were used for this schematic in order to put the least amount of stress on the microprocessor. If BJTs were to be used current would have to be provided for amplification to happen, meanwhile MOSFETs require just a voltage, which integrates better with the microprocessor GPIO pins.

This circuit will supply each of the LEDs up to a maximum of 20mA, which is the recommended value according to the datasheet.

5 Conclusions

After studying the MSP430 for a little over a month, all of the cumulative skills have been put to the test in this project. After digesting the requirements set for the lab, a design choice was made and the MSP430F5529 was chosen due to its CCR numbers. Additionally, another design choice needed to be made for the external hardware, and MOSFETs were used as they are more integrable with microprocessors, as they deal with voltages. After the code was compiled, a program was left that will be able to light up the LED according to the information presented to it, as well as pass along all of the other information to the next device.