Pedo-Watch: Final Project Written Report

By: Juan Garza, Son Nguyen, Oscar Reynozo

ELEC 327

Due: May 4, 2022

Professor Kemere

Design Concept:

We sought to design a simple energy-efficient electronic pedometer popularly used among fitness and health enthusiasts. For our final project design, we introduce the Pedo-Watch. Our watch utilizes the interrupt and SPI configuration capabilities of the MSP430G2553 microcontroller to detect and display the user's number of steps. In addition, we incorporated a pedometer three-click sensor to see the number of steps the person has taken and a color TFT screen to display the time of the day and the number of steps that the user has taken. Additionally, we implemented two-button functionality to be able to reset or set the step count and current time.

Bill of Materials:

- Texas Instruments Launchpad
- 20-pin MSP430G2553
    - Datasheet: https://elec327.github.io/assets/documents/msp430g2553.pdf
- Adafruit 1.14” 240x135 color screen with the ST7789VW TFT driver
    - Link: https://www.adafruit.com/product/4383
    - Datasheet: https://cdn-learn.adafruit.com/assets/assets/000/082/882/original/ST7789VW_SPEC_V1.0.pdf?1571860977
- Pedometer 3-Click Sensor
    - Link: https://www.mikroe.com/pedometer-3-click
    - Datasheet: https://download.mikroe.com/documents/datasheets/KX126-1063.pdf
- Two buttons
- Two 1k Ohm resistors
- 100k Ohm resistor
- 4.7k Ohm resistor
- Two 47nF capacitors
- Two 0.1uF capacitors

SPI Configuration:

Generally, our Pedo-Watch works by configuring the KX126-1063 IC, our pedometer three-click sensor, and the Adafruit 1.14” 240x135 color screen with the ST7789VW TFT driver, our color TFT display. We decided to use the SPI interface because our pedometer and screen require a high-speed clock to check and update the number of steps taken and the current time. For our SPI configuration, the MSP430G2553 serves as the master device. Meanwhile, the screen and pedometer serve as slave devices. With the exception of the pedometer, each slave device usually has a dedicated chip select pin connection to the MSP430G2553 to determine the slave device that the MSP430G2553 is writing to at any given time. In this case, when providing input or receiving an output from a device, the MOSI (Master-out, Slave-in) and MISO (Master-in, Slave-out) configurations determine the direction in that information is being read or written.

Pedometer Configuration and Implementation:

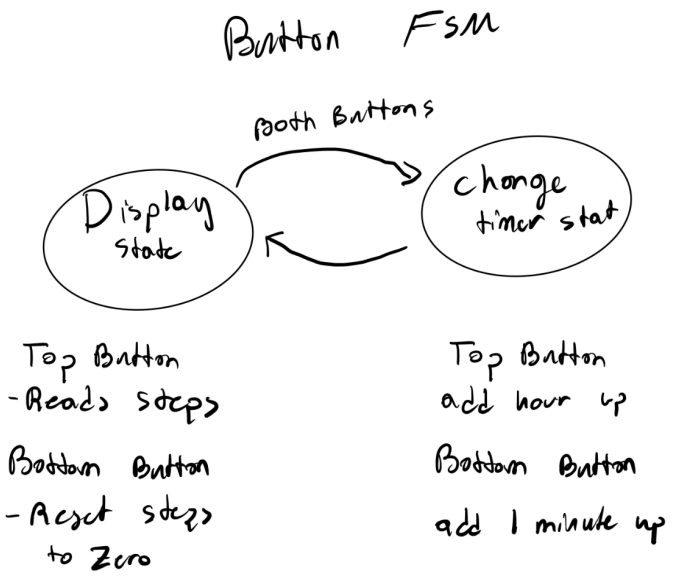
With the KX126-1063's SPI configuration, we utilized a 0.5 MHz clock signal to determine whether to read or write at every positive edge of the clock signal. When given a read or write command at every positive edge of the clock signal, the command must modify the IC's registers to access settings specified by the chip's datasheet. The MSP430G2553 must first set the pedometer to software reset mode, which requires control register one and control register 2 to be changed. Once in software reset, the sensor can be configured into a pedometer function and count steps. The KX126-1063 IC can function as both a pedometer and an accelerometer. Under the pedometer configuration mode, the acceleration register has access to ten different control registers PED_CNTL_XX; these control how the pedometer counts steps, including the threshold for calculating steps. In addition, there are two 8-bits registers used to represent the step count, one register corresponding to the high values (8 most significant bits of steps) and another register corresponding to the low values (8 least important bits of the step count). When the displacement reaches a certain threshold where the sensor accelerates and decelerates from the threshold value, a step is detected, and the step counter increments. In addition, there are configurations for the interrupt pin; these can set pins to be active or low for when the pedometer counts above a threshold of steps.

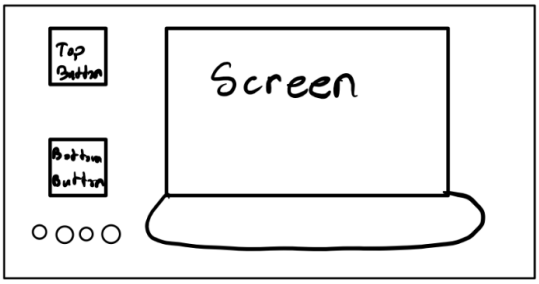
TFT Screen Configuration and Implementation:

Moreover, the TFT screen's SPI configuration works similarly to the KX126-1063's SPI configuration. However, we configured it with a 1 MHz clock signal for writing or reading data at the positive edge of each clock cycle. To set up the TFT screen, we must first initialize a software reset, then configure the master, phase, and synchronous modes in the specified order by sending commands specified in the datasheet and modifying the TFT screen's control registers. Once the TFT screen can receive bytes for displaying text, the CASET and RASET commands must be used to specify the row and column when drawing a pixel. The RAMWR (for writing to the RAM) command writes its corresponding byte value to the UCB0TXIFG buffer, and a byte indicates the pixel's color. Thus, to write messages to the screen, each pixel in the screen must be iterated through and colored appropriately to form text on the screen with a specific text font and color. Utilizing this framework for writing text onto the TFT screen, we used a custom font for a microcontroller and itoa function in C to convert the current number of steps and time from an integer to a string displayed onto the TFT screen as the values update. Once the screen successfully updates the number of steps and time as part of our minimally viable product, we incorporate animations with color changes and transitions to make the TFT screen more appealing and entertaining. Unlike the step counter, the time counter continuously incremented every second and timed appropriately using an interrupt delay.

Buttons Implementation and FSM:

The last component of the system implemented was the two buttons' active low functionality. These buttons operate in a finite state machine. The initial state is *displaying,* where the top button corresponds to reading the step registers and updating the count if there are steps stored in the sensor. The bottom button corresponds to resetting the step count to zero. When both buttons are pressed simultaneously, the TFT display is set to *a time-setting* state. The top button increments the hour value of the current time, and pressing the bottom button minute value of the current time. Conditionals in the code ensure that decreasing the time does not go below 00:00 and increasing the time does not go above 23:59. Once in time-setting mode, the user can leave the time-setting mode and set the time chosen by pressing both buttons at the same time again. A state machine diagram is drawn below.

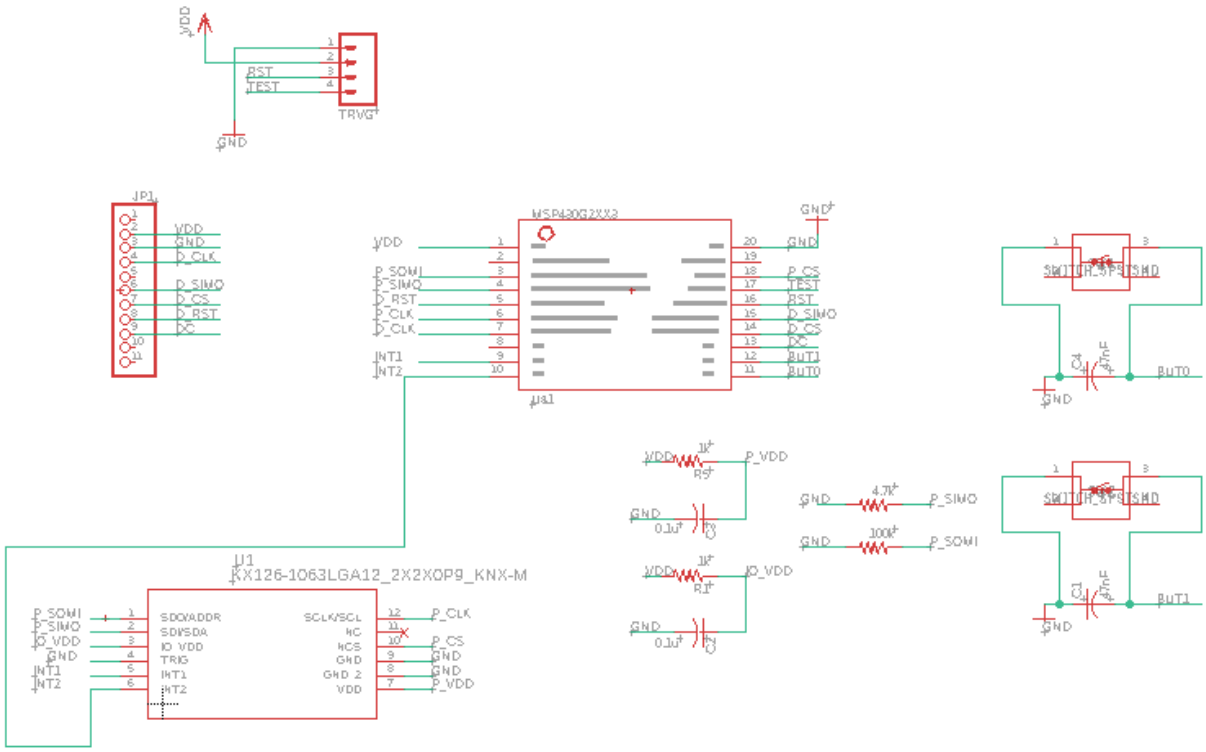




PCB Design:

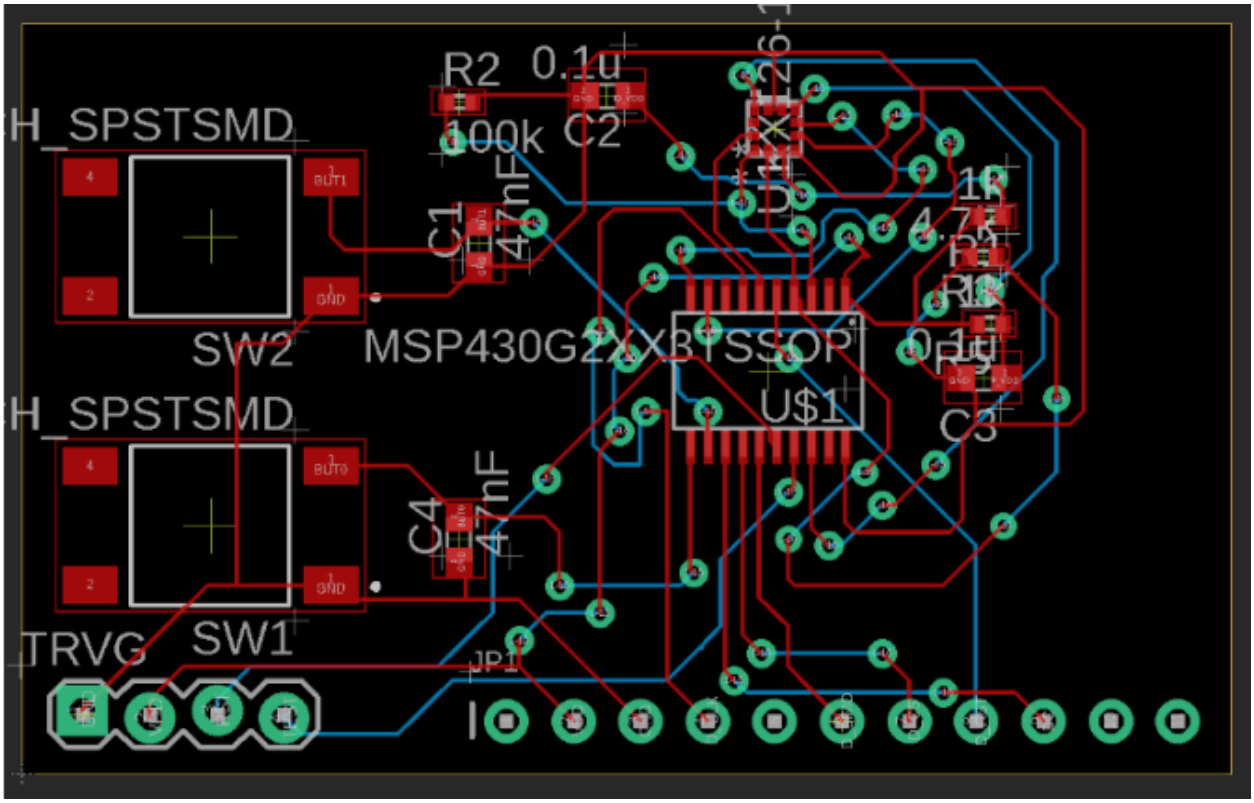
For our PCB design, aside from including the MSP430G2553, the KX126-1063, the display's pins, and the buttons, we also had the 4-pins (GND, VDD, RST, and TEST) connecting to the launchpad for uploading code and resistors/capacitors needed for some of the pedometer and button connections. Here below are images of our Eagle PCB schematic and board files:

Eagle PCB Schematic File



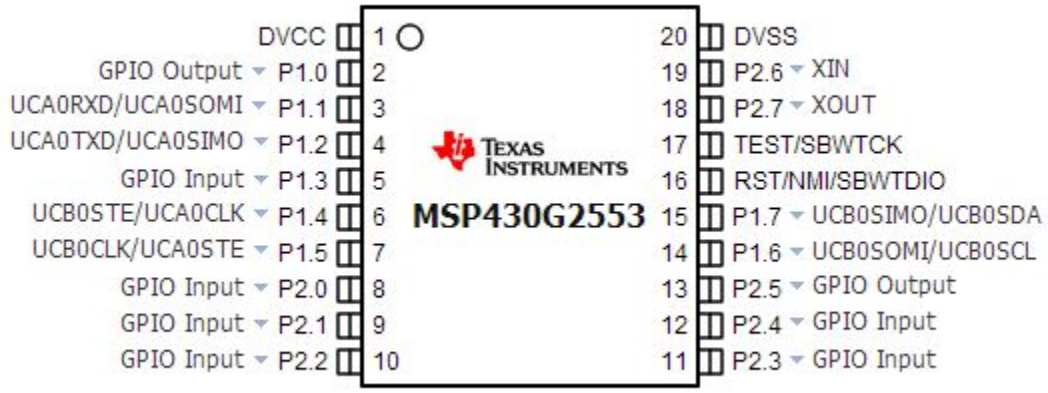
The 47 nF capacitors connected to the buttons eliminate noise and ensure a smooth transition from high to low. Further, the 1k ohm resistor and 0.1 uF capacitor voltage dividers supply 2.2 V to the KX126-1063 chip. Lastly, the 4.7k ohm and 100k ohm resistors reduce the noise in the data transmitted from SDO and SDI pins in the KX126-1063 chip.

Eagle PCB Board File

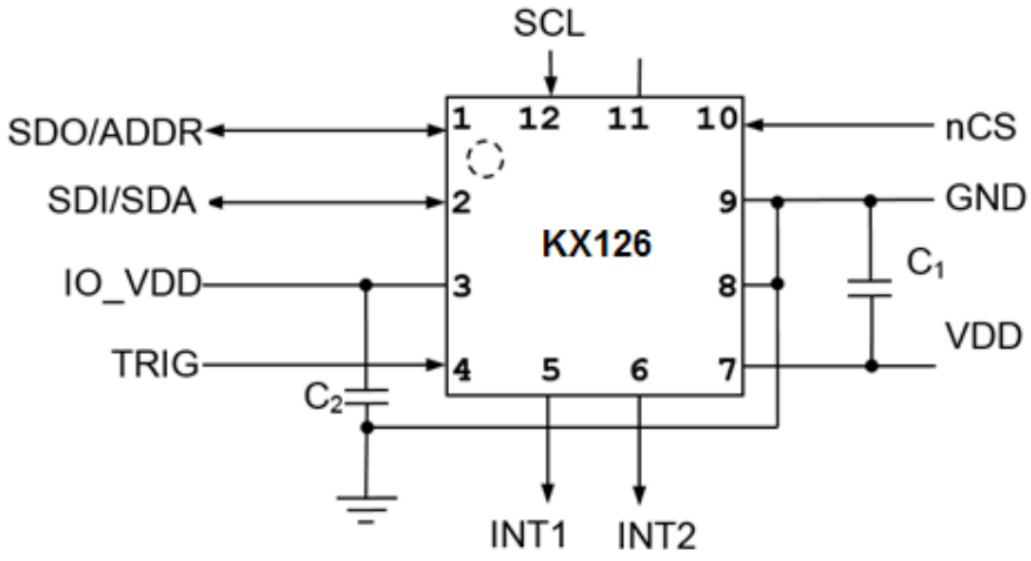


Only the DVCC and DVSS pins of the microcontroller are shared among the different components of the PCB to supply voltage and ground. Except for the vacant pins P1.0, P2.0, and P2.6, the rest of the microcontroller's pins are allocated to a specific purpose for a single component. Lastly, for reference, here are the pin layouts for the 20-pin MSP430G2553 microcontroller and KX126-1063 IC, along with each pin's use:

MSP430G2553 Pin Layout



KX126-1063 Pin Layout



Low-Power Implementation:

We achieved our goal of making the Pedo-Watch a low-power system by extensively utilizing interrupts throughout our system when awaiting inputs or information to be read, written, or transmitted. Instead of the delay cycles function that keeps the MSP430G2553 on while awaiting a delay, we created a delay function that configures the appropriate CCR0 interrupt settings that puts the MSP430G2553 to sleep wakes up the MSP430G2553 using the LPM3 bits. Additionally, we employ the active low button interrupt to update the number of steps displayed on the screen after the user's every twelve steps to increase the pedometer's efficiency when updating the step count. Further, utilizing the interrupt for delays reduces the time that the MSP430G2553 is left powered on, reducing the amount of power it consumes.

Challenges:

Our main challenge throughout developing the Pedo-Watch was being able to send bytes to our pedometer and screen given the lack of available C libraries and sample code for interfacing our components with the MSP420G2553. However, we "reverse engineered" the commands needed to send information to each device. To achieve this, we found a sample working Arduino code for both the screen and the pedometer, where we uploaded the code to an Arduino and probed the MISO and SCLK pins of each component to read the bytes of each command transmitted. Each byte's equivalent hexadecimal value had a corresponding command in the datasheet. Once the MSP430G2553 writes the correct commands to each component, the MSP430G2553 can successfully write data to each component. Additionally, we experienced significant difficulties with finding the right timing for reading signals from the KX126 Sensor, we were one read buffer behind which caused initial confusion.

Team Member Contributions:

Juan Garza:

- Wrote hardware configuration for the pedometer and implemented functions to initialize the pedometer and detect steps with user motion
- Implemented button functionality with resetting the time/step count and setting time depending on buttons pressed
- Soldered final PCB

Son Nguyen:

- Worked on “reverse engineering” screen configuration by probing screen and running sample Arduino code while finding corresponding commands in the screen’s datasheet.
- Developed 1st PCB
- Wrote hardware configuration for the screen as well as implemented functions to initialize the display and send bytes, color, and draw pixels to the display.
- Integrated both pedometer and screen functionalities to run simultaneously
- Assisted in soldering final PCB

Oscar Reynozo:

- Worked on “reverse engineering” screen configuration by probing screen and running sample Arduino code while finding corresponding commands in the screen’s datasheet.
- Significantly researched and wrote notes for the team on the SPI interface for the MSP430G2553
- Developed 2nd and Final PCB

- Wrote written report, slides for presentation, website, and recorded/edited presentation videos
- Wrote code for text fonts and functions for writing letters and strings of text to the screen
- Assisted in soldering final PCB

Final Project Presentation Link: https://www.youtube.com/watch?v=Jgsx6iXTma4&t=416s

Final Project Demonstration Link: https://www.youtube.com/watch?v=IDnrIIwVhfU

Github Link (includes code and PCB files):

https://github.com/JuanLikesRice/Pedo_watch_team

Sources:

- Fonts for microcontrollers: https://jared.geek.nz/2014/jan/custom-fonts-for-microcontrollers
- Adafruit 1.14” 240x135 TFT LCD Display Arduino Sample Code:

  https://learn.adafruit.com/adafruit-1-14-240x135-color-tft-breakout/arduino-wiring-test