Portable Thermal Printer

Introduction

Our senior design project name is **Portable Thermal Printer** for ECE 445 Spring 2023.

We are Team 29, with the assistance of TA Shao Hanyin (hanyins) - Gally Huang (ghuang23) - Jason Liu (jliu246) - Kevin An (kqan2)

Objectives and Background

Introduction

Problem: One of the biggest problems surrounding frequent travelers is the issue of portability. Items that are carried along have limits on their weight, cannot consume too much space, and have to compromise on quality. A target area that has been identified by the Hewlett-Packard Company (HP) lies within the commercial printer industry. Printers as a whole have remained relatively unchanged over time with respect to other technologies that have shifted towards more portable means. As such, they remain inconvenient for travelers who need to quickly print items on the go. HP has identified a potential entry into the portable printer market to remain competitive in this industry and find new methods for company innovation.

Solution: Our solution is a portable thermal printer, a system that receives wireless instructions for printing on receipt paper. Users will be able to upload images from their phones or computer that this system can fetch and print.

We will use a field-programmable gate array (an FPGA) to implement our solution because they can stand in place for a real-world application-specific integrated circuit (ASIC) and eventually be developed in an ASIC. It will be utilized as the base of the project. Additionally, we need to have a way to print, so we will be using the internals of a thermal printer along with a microcontroller unit (MCU) to handle the wireless components. Finally, we will be creating our own input/output shield (I/O shield) for the PCB that has the subsystems listed further down on top of it.

Visual Aid:

Goals and Benefits

- Make printing portable in the world where many other technology have already evolved to become more portable.
- Current solutions using wireless connection usually support Bluetooth, which is short-range, or are expensive.

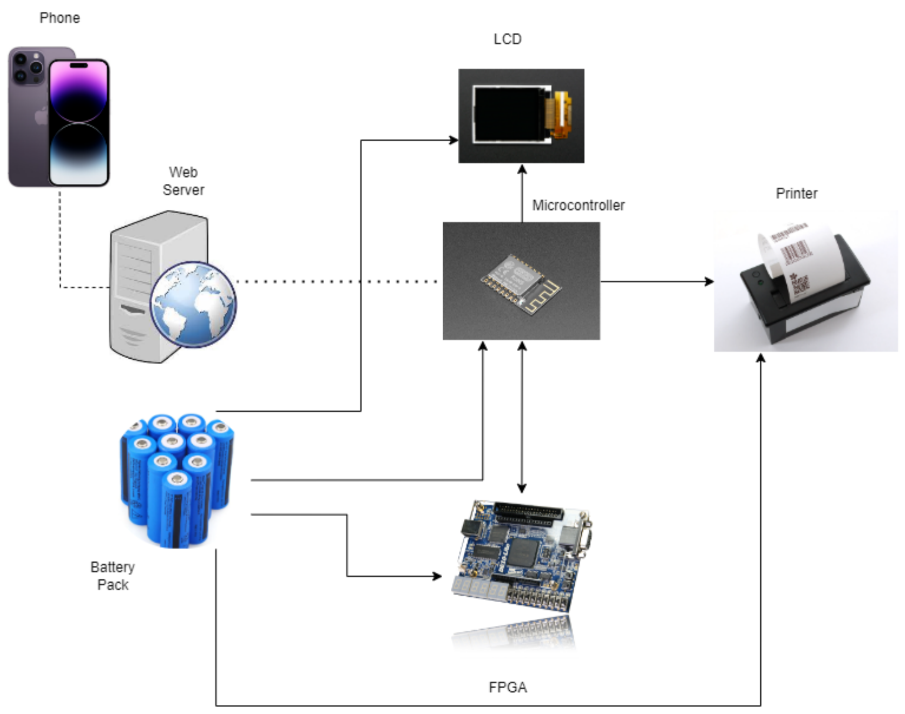


Figure 1: Diagram

- The project allows for printing with a battery system and wireless uploading of image data, making it very portable.

High Level Requirements

- The device design is portable. It should be able to wirelessly and accurately get the user-uploaded image data from a server to the embedded MCU. It should sit as a small footprint of at most 12”x12” as to fit comfortably within a suitcase, allowing for ease of transportation.
- The device itself should also be completely powered by batteries, having an average (if not worst case) battery life of ideally 1.5 or more hours.
- The start to end time, between user upload and completing the printing, should be within ~ 20 seconds so as to not consume too much time for the user.

Design

Block Diagram

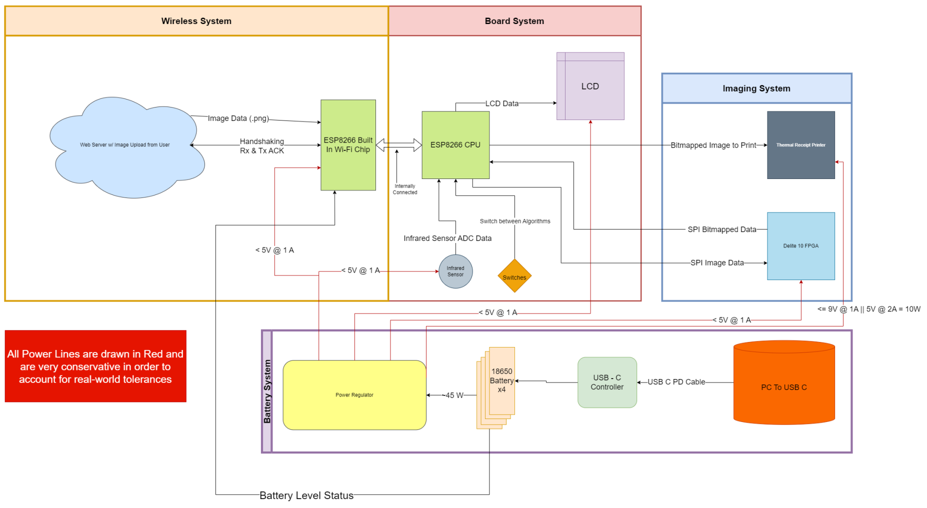


Figure 2: Diagram

Description

Wireless Subsystem

Wireless Subsystem Overview: The purpose of the Wireless Subsystem is to allow the system to wirelessly connect between a server (can be locally hosted on a computer or on the cloud), a user, and the ESP8266 MCU. The benefits of this subsystem add portability for the product and a more “polished” feel for the user, reducing the need for excessive cables and clutter.

There will be a simple web page backed by the server that a user can interface with and upload an image to and upon which, the user can request a connected printer to print the uploaded image. The server will send data to the MCU through a WiFi connection between the MCU and the server. Upon receiving this data, the MCU will further process it.

Wireless Subsystem Requirements: ESP8266 Microcontroller (MCU): - This low-cost MCU will be embedded on the custom PCB (we will design this as an I/O Shield for the system’s FPGA). With respect to the wireless functionality, it is responsible for allowing the printer system itself to stay wireless, as it has a built-in WiFi microchip, enabling simple connection to an application server (discussed below). With the MCU acting as a client to an application server, it can create something such as HTTP requests and receive data from the application that users can upload an image to (can be programmed to perform requests at certain intervals, manual button press, etc.).

- This MCU will then be responsible for delivering the image data to a buffer for the FPGA to further process towards printing, taking advantage of the FPGA for hardware acceleration in implementing DSP algorithms (such as the Floyd-Steinberg dithering algorithm) in order to speed up the image manipulation and cleaning processes. It will also send diagnostic data about the state of the current processes to an LCD display, such as about current battery status, ready for printing acknowledgement, paper jam, etc.

Application server: - A server which allows the user (when connected) to upload an image through a computer or cell phone and enables the MCU to receive the data through a request following the event. The front end can be created with a simple interface (i.e., basic web development through HTML/CSS/JavaScript) that allows users to upload an image, and an on-screen button which flags the image as ready to be delivered to the MCU upon the next request. The back end can be handled with the Django framework and an API which allows the user to actually upload the image on the server and for the MCU to get an encoded version of the to-be-printed image (i.e, through base64 string encoding in a JSON) from the server.

- As mentioned previously, the server can be hosted locally for the scope of this project as-is, especially as a means of saving a consistent amount of money as opposed to hosting on a commercial cloud platform such as AWS or GCP. For large scale implementation, we of course cannot rely on local servers, but this simplifies our testing requirements with a small sample set of users and devices to work with.
- The MCU will require 12 mA of current and anywhere between 3-3.3 V continuously for operation [2], defined in the Power Subsystem how it will

be delivered. The local server, being hosted a computer, will require power delivered through a commercial power adapter supply (i.e., laptop being powered by a laptop charger), however, we allow this to be hidden from view for the user.

Imaging Subsystem

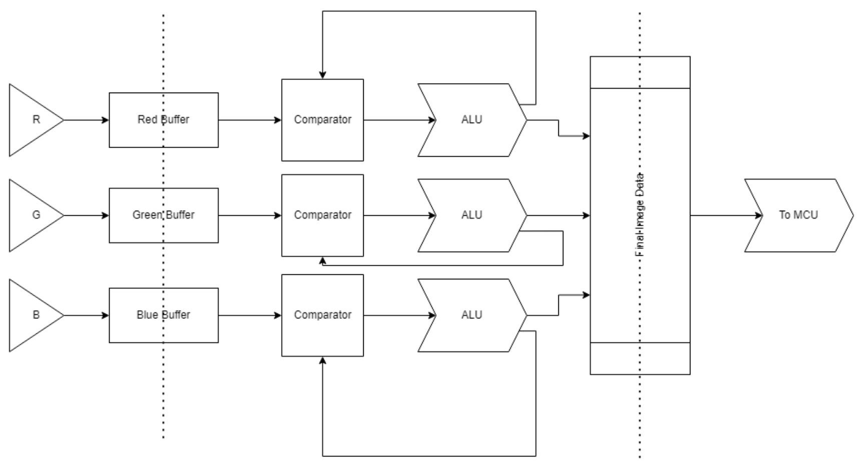
Imaging Subsystem Overview: The Imaging Subsystem allows for three pixels to be converted and mapped into the dithered equivalent after being processed. The processing is done entirely by hardware as this is the "hardware accelerator" portion of our project. This will allow for the images to be printed out at an incredible rate in a very similar fashion to how it is done in industry with consumer grade printers at HP. Additionally, this subsystem includes the thermal printer itself, which takes the hardware accelerator's image and routes it back through the board so that the MCU can send the image to the printer to be printed.

Imaging Processing Subsystem Requirements: DE10-Lite FPGA: - An FPGA will be utilized to simulate the operation of an ASIC which is not openly available to the mass public. FPGAs are commonly used to test HDL code at a very cheap cost compared to a full scale tape out, albeit at a slower clock, so we will attempt to multiply our hardware throughput at the correct proportional speedup rate. The FPGA can run at a speed of 50 MHz [7], while mainstream ASICs can usually run 10-50x faster than this, but this will still be much faster than processing the image through software means (on the cloud or on the MCU).

- The FPGA must take in data through the SPI protocol and be able to send data back out through the SPI protocol as well.
- The FPGA will take in three pixels and run it through a pipeline. Firstly, the FPGA must store all of the RGB (red, green, and blue) values of an image into its onboard memory to prepare it to be processed. While the pixels are being stored into memory, we can start processing some of the data while it is still in the process of gathering data from the MCU. This is because many of the algorithms that will be applied, such as Floyd-Steinberg dithering [6], only requires 5 adjacent pixels for the image to start being processed. We need to set up a state machine that detects whenever a threshold amount of pixels have been loaded into the FPGA, and then, it will start to process this data simultaneously. The third stage of the pipeline is when the data needs to be stored in a final bitmapped processed stage, and then this final image will be sent back out into the MCU and will be ready for printing. This process happens very fast, and doing the math, it should not take more than $3 * (\text{Number of pipeline stages}) * (xy) = 3xy$ clock cycles in order to process a single image, where x and y are the dimensions of the picture.

Diagram of sample algorithm (all are pretty similar except for different ALUs):

Floyd-Steinberg Dithering



- As for the printer, the printer must be able to interact with the MCU correctly. This means that the ports coming out of the MCU has to be connected correctly to the printer. We also need to make sure that the printer receives enough power so it will be run with power through its own dedicated rail, since we expect that at least 10 W will be used by the printer during peak run time.

Board Subsystem

Board Subsystem Overview: The Board Subsystem is the interactive and diagnostic block that allows for the user to check the status of the entire system at a glance.

The primary component is a small 1.8" raw TFT display [4] that displays useful information about the battery level and the status of a printing job for user diagnostics (e.g., completed, failed, paper jam), all of which is processed and delivered from the MCU.

There will also be an infrared receiver sensor that will sense if there is still a supply of thermal paper for the thermal printer to print on. If the sensor detects a change in the paper supply, this information will be sent to the MCU, which will have the LCD print out a visual warning/error and prevent the printing process.

Finally, there will be a switchbox that the user can use. Switches, when turned on and off, will change which algorithm the FPGA will use when processing the image (e.g., Floyd-Steinberg Dithering, Burkes's Dithering [5], etc.).

Board Subsystem Requirements:

- This block contributes to the overall design by providing a reasonable level of user experience. It informs the user of potential issues pertaining to battery life and printer status and allows the user to change between different image processing algorithms based on their needs.
- One requirement for the LCD is that it must be able to refresh its status/display at a decent rate so that monitoring/debugging the system is reasonably convenient for the user ($<$ 5 seconds). If something changes in the status of the system, the LCD should be able to reflect upon this change with little lag.
- While not directly responsible for the Board Subsystem, the MCU is responsible for sending and processing data that is delivered to this subsystem. Without it, the LCD would fail to function and as a result, the Board Subsystem would essentially be rendered useless.

Power Subsystem

Power Subsystem Overview: The power subsystem supplies power to every other subsystem. Namely, it powers components such as the ESP8266 MCU at 3-3.3 V [2], the thermal printer at 5-9 V [3], the FPGA at 5 V [7], the LCD at 3.3 V [4], and the infrared sensor at 3-5 V [10]. Its components are a USB-C controller that will be connected to a PC's USB-C port. This connection will supply power to our four 18650 batteries. We use a regulator system to maintain constant voltage levels to the components stated above. It will also flash the MCU (send program information to the MCU to execute).

This subsystem as a whole is necessary for supporting the continued operations of the entire system, which includes displaying the diagnostic information, the Wireless Subsystem receiving image data, processing image data, and printing.

Power Subsystem Requirements:

- The other subsystems must be powered on with this subsystem at the stated voltage and current levels or with a maximum of -5% deviation.
- It is important that the power system is able to supply the upper conservative limit of 45 W as well, since this would be able to provide enough power to the system in the case of sub components requiring peak power.
- We also must be able to check the current battery level percent of the 18650 batteries on the LCD in the Board Subsystem. This diagnostic data is to be delivered to the Board Subsystem for displaying to the user.

Risk / Tolerance Analysis

- Servers are considered outside the scope of this class, so it may be difficult to implement. Additionally, based on our implementation of accepting data from a user, we can have our local server (and hence, our local device) be susceptible to a cyber attack. Using JavaScript makes us potentially vulnerable to some control flow hijacking, which can allow users to attack our device. Since our code is online, attackers can try to precisely send images to hijack the server.

- The printer itself needs to operate at over 150 degrees Fahrenheit in order to activate the thermal paper, and therefore we must ensure, for the safety of the device for the user, that the specific area intended to be held by the user remains under 120 degrees Fahrenheit throughout operation. The reason for 120 degrees Fahrenheit is because this is generally agreed upon for handheld products as the upper limit of a safe-to-touch temperature [9], and it would be extremely detrimental if the device were to cause harm by exceeding this rating.

Ethics and Safety

We believe that this design is quite safe according to IEEE standards [1] since most of the components are found in everyday objects such as a phone and other consumer-grade products. However, we take some ideas from prior product failures, such as the Samsung S7 battery that exploded and imploded on itself due to a manufacturing defect [8].

Therefore, we should be testing the battery throughly in order to ensure that it is not causing any power overages to occur, which may cause harm to the user. This includes setting up battery current limitations as well as limitations on how much the battery can charge. By limiting the amount of things that the battery can do, this will in turn cause for the most volatile part of the system to be the most safe.

References

[1] IEEE Policies, Section 7 - Professional Activities (Part A - IEEE Policies), IEEE Code of Ethics 2020.

[2] Expressif Systems, “ESP8266EX Datasheet,” 2015. Accessed: Feb. 07, 2023. [Online]. Available: Link

[3] P. Burgess and Adafruit Industries, “Mini Thermal Receipt Printer,” Nov. 2021. Accessed: Feb. 07, 2023. [Online]. Available: Link

[4] Truly Semiconductors Co., “JD-T18003-T01.” Link (accessed Feb. 07, 2023).

[5] T. Helland, “Image Dithering: Eleven Algorithms and Source Code.” Link (accessed Feb. 09, 2023).

[6] justinkraaijenbrink, “Exploiting the Floyd-Steinberg Algorithm for Image Dithering in R,” Medium, Jan. 30, 2021. Link (accessed Feb. 09, 2023).

[7] Terasic, “DE10-Lite User Manual,” Jun. 05, 2020.

[8] R. Rox, “Samsung Galaxy S7 explodes during charge,” Notebookcheck, Sep. 03, 2017. Link (accessed Feb. 08, 2023).

[9] Johns Manville, “Too Hot to Handle?,” www.jm.com, Feb. 25, 2015. Link (accessed Feb. 08, 2023).

[10] Vishay Semiconductors, “TSOP382.., TSOP384.., TSOP392.., TSOP394..,” Feb. 2011. Accessed: Feb. 09, 2023. [Online]. Available: Link