FPV DRONE SHOOTING GAME

By

YIXUAN WANG

JIARONG BAI

KAINAN YU

Final Report for ECE 445, Senior Design, spring 2019

TA: David Null

01/May/2019

Project No. 53

Abstract

Our project, the FPV (first person view) drone shooting game is a game that allows drone-players to practice their flying skills with little environment requirement through a fake air fighting scene. This report includes how we implement this drone-game, how we choose each module for its purpose and what we have accomplished.

Contents

1. Introduction

In recent years, the drone industry and the drone community has grown noticeably due to drone's playable features and functionalities. The increasing number of people who enjoy playing drone also creates a series of raising problems for the drone community, such as the limited playground (most major city has banned drones) and limited ways to play around with these drones. By creating the FPV drone shooting game, drone players will be able to practice their drone-flying skill on a less-qualified (lower requirements: less space required and less concern with the government's restriction towards the drone) playground. And drone players will be able to practice their drone-flying skills with more enjoyments from the competitions between players.

In this project report, we will summarize the inner logics, implementations, modules selections, requirements, and the costs of our drone-shooting system.

Our top-level block diagram is described in figure 1 on the next page. It contains three main parts. They are: **power supply**, **control unit** and the **video processing unit**.

Power supply: We will be using the drone's own power to supply our system with the use of the power distribution board (PDB) and some linear voltage regulators (3.3V and 5V).

Control unit: The control unit contains the logics of our drone shooting game, like how we send the orders to the drone to tell the drone to start the game or to shoot, what will happen if the drone gets shot by another drone and the condition of winning the game.

Video processing unit: This is where we convert the gaming content, like the health points & score points, nearby enemy drones and winning status to the FPV goggle and screen.

Our top-level block diagram stays almost the same throughout the semester, and we designed and implemented each component according to this block diagram (figure 1). We successfully implemented and tested the three components, that they are all working correctly according our design individually. However, due to the highly unstableness of our RF modules and PCB soldering defects, we were unable to combine the three components of this system together as a complete system.

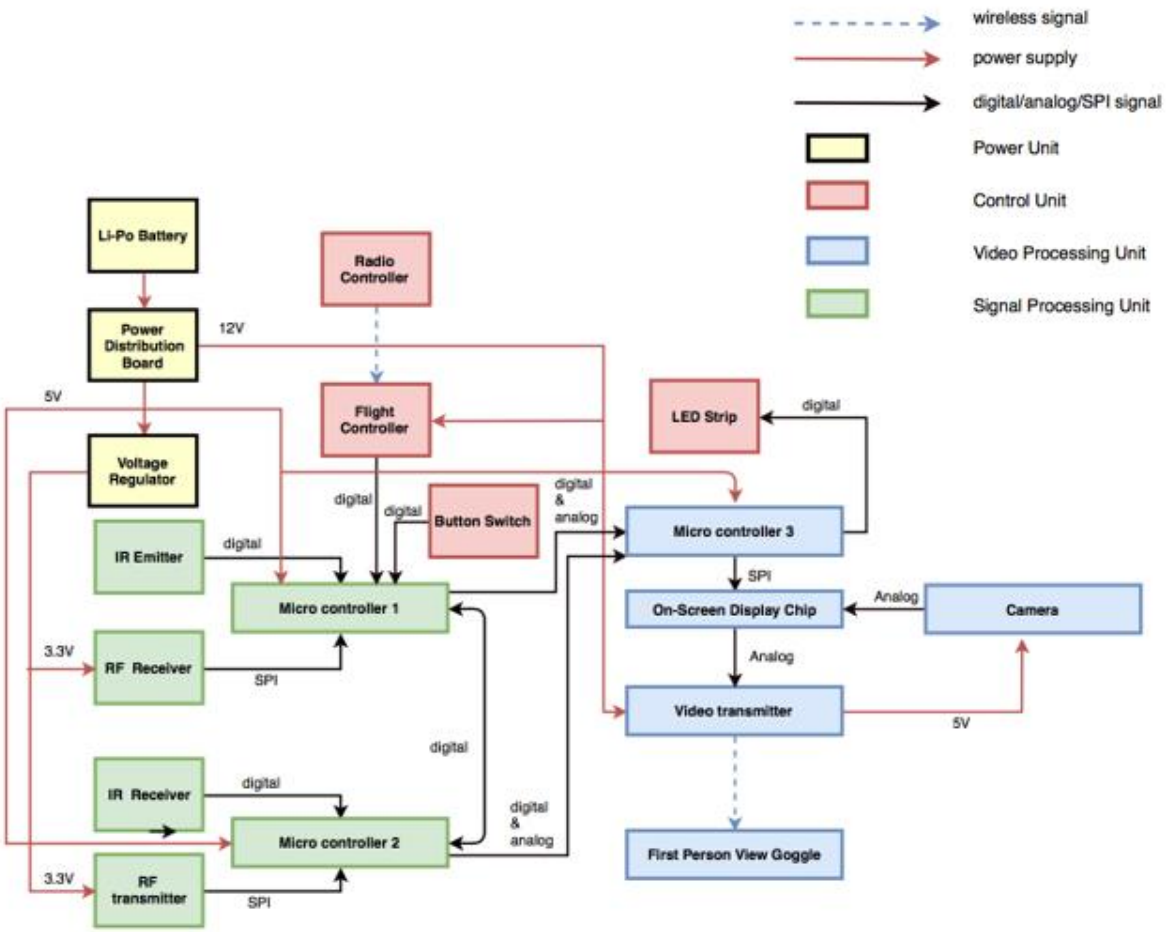

Figure 1： Top-level block design

2 Design

Our drone shooting system consists of three major parts: the power system that distribute power to our system, the control logic that handles the gaming logic and the video processing unit that display the gaming and drone's status on goggle and LED screen. They are discussed in details as below.

2.1 Power distribution board (PDB)

2.11 Design ideas

Since a drone has limited of space and weights of carrying, applying additional power inputs on the drone will be unrealistic. Having additional power inputs not only puts on extra weight, but also takes up extra space. We will have to limit the total extra weight and space for our system.

We tackle this problem by using drone's own power system to supply the power of our drone shooting game's system. We can do so because the drone's own power system has abundant power that we can draw from, and our drone shooting system requires relative low power inputs.

The purpose of the power system is to make sure that all modules can work correctly without any power issue. We want to avoid the risk of frying the circuit with too much power being fed. And also, we want to make sure that each component is working without any malfunctions under its specific power requirement.

Since all of our modules can work under the voltage of 3.3V to 5V. We will be using the 3.3V linear voltage regulators and 5V linear voltage regulator extensively.

2.12 Module selection

Since all of our modules can work under the voltage of 3.3V to 5V. We will be using the 3.3V linear voltage regulators and 5V linear voltage regulator extensively. Our drone will carry a Li-po battery that will be served as the power input for our system. The Li-po battery has a voltage output of 12V. We choose the module TSP763 as our 3.3V linear voltage regulator and L78L as our 5V linear voltage regulator.

2.2 Control unit

The control unit handles the gaming logics. What will the drone be using for the shoot, what will happen if one drone gets shot by the other drone and what to determine a drone's winning status.

2.21 Design ideas

We are using ATmega328p for our micro-processors. The good side about the ATmega328p is that we can read the serial output of our modules through Arduino, making it easier to test and debug.

For the shooting part, we will be using TSAL6100 and TSOP4838 as our IR communication tool. The IR emitter TSAL6100 is a good choice because of it can be encoded, can only be receive within limit angle and can transmit longer than usual IR emitters.

For the shooting feedback and enemy nearby status, we will be using nrf24l01 RF transceiver as our module.

For the center information processing unit, we will process all the gaming logic data and then send to the video processing unit. The video processing unit will display these data accordingly and then display it on the goggle and LED screen.

2.22 IR communication unit

We will be using IR communication to mimic the shooting scenario. In order to create a shooting tool that has all the key characteristics of a bullet, we should have our IR communication based on the TSAL6100 to satisfy these important characteristics.

A bullet usually travels very fast and is able to shoot the target within unnoticeable time-interval. Luckily IR communication has this characteristic because IR signals travel near the speed of light. Assuming two drones are separated by a distance of 50 meters, which is the max distance for our game. And IR signals are traveling at the speed of 2/3 of light, which is slower than most of the IR actual travelling speed. The total time of IR travel is calculated as follows (2.21):

$$time\ of\ travel\ (in\ seconds) = \frac{distance\ (\ in\ meters)}{speed\ of\ IR} = \frac{50}{200000000} \\ = 2.5 * 10^{-7}\ seconds \qquad (2.21)$$

We can see from the equation (2.21) that the IR communication can be done between two drones within unnoticeable time.

A bullet usually travels in a straight line. In order to better mimic the bullet's travelling path, we should be using the laser communication tool. But when we consider the real-world scenario, using laser would increase the game's difficulty as well as the implementation difficulty greatly. Aiming a tiny spot (standard laser receiving area is around 5mm * 5mm) in 30 meters will require significant efforts. Also, we will have to map the drone with abundant laser sensors, which will cost a lot of unnecessary weights and money. The IR communication is better for 2 reasons. The first reason is that the IR emitter module we choose, TSAL6100, has the angle of half density of 15 degrees. Which means that this IR emitter will be able to transmit IR signals in a small angle direction. This reduces the efforts of aiming greatly. The

second reason is that at the receiving end, we only need to put 2 IR receivers to receive the IR signals, which can reduce the total weight for the entire system.

The IR emitter is connected to the PD3 port of ATmega328p (Figure 2), which can be encoded with the drone's ID. The drone being shot can decode the IR signal it receives to determine which drone is shooting at it. IR emitter and receiver can both work under the voltage from 3V to 5V. And we connected the 3 IR emitters in parallel and each to a 330ohm resistor to enhance the range of IR communication (Figure 2). For the receiving end, we use two IR receivers to better receive the IR incoming signals in all directions (Figure 3).

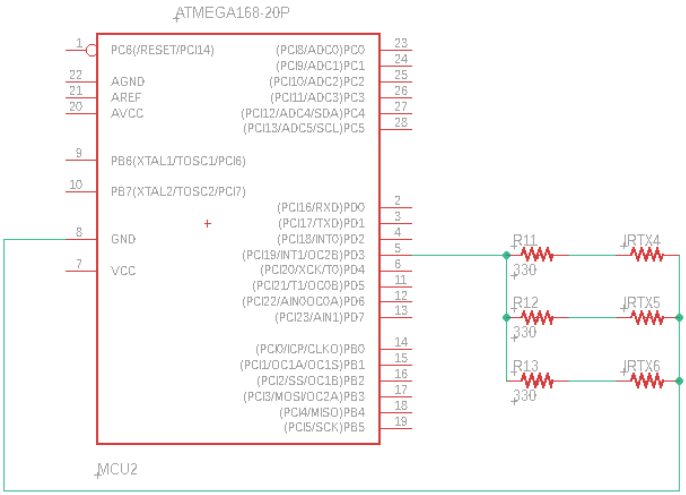

Figure 2: IR emitters circuit

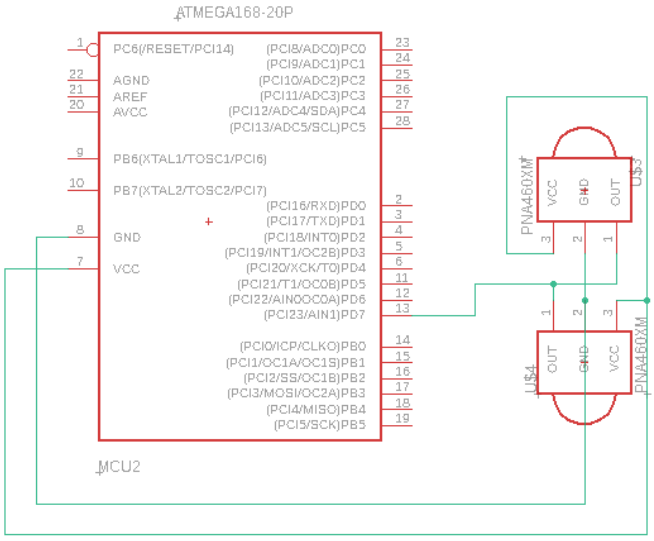

Figure 3: IR receivers circuit

The next step is to encode and decode the IR signals. We can use the IR transmission protocol to encode our IR signal. There are a lot of IR transmission protocols available online. Including the NEC protocol and Sony protocol.

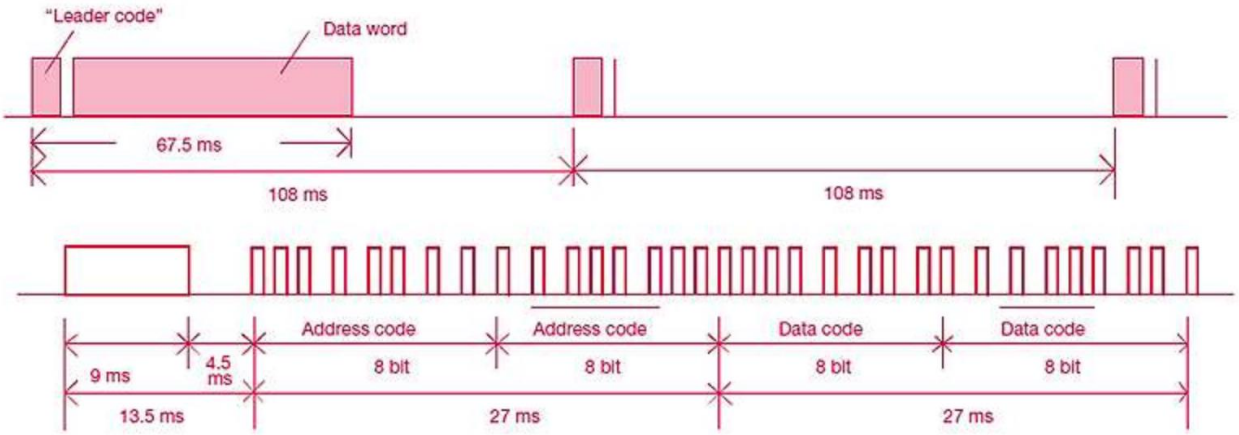

Figure 4: IR transmission protocol

The figure 4 is how the IR transmission protocol works. Basically, you have the "Leader code" and the "Data word". The first step to complete the IR transmission is to send the "Leader code". Once the receiving end detects the "Leader code", it will start to decode the data. In our case, the "data word" contains the player's ID. Once one drone gets hit, it will know which drone fired the shot.

What makes the IR transmission good for our project? We can answer this by stating that the IR signals have different wavelengths compared to the natural light. In figure x, we can see that most of the natural lights have the wavelength from 400 nm to 800nm. Our IR has the wavelength around 940 nm which will not have interference with the natural lights. This makes it possible to play our drone shooting game outdoor.

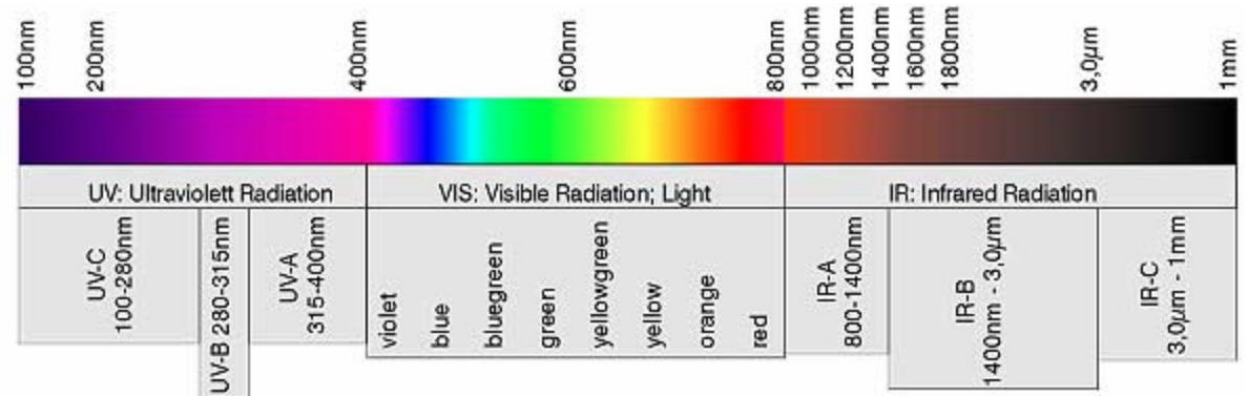

Figure 5: Wavelengths of different lights

2.23 RF communication unit

The RF module is used to transmit and receive location signals and hit feedback signals. These two signals need to be sent as fast as possible to all directions, so we choose RF signal to achieve these goals. The location signal is used to indicate whether there are enemy nearby. The hit feedback signal is sent once the player receive a IR attack signal from enemy and send an acknowledge back with attacker ID. The message is displayed in following form: 'LOCX' and 'ATTX'. For location signal, the X is set as player ID. If a player receives a location signal it need to be verified such that the signal is not sent by itself. The X for hit feedback signal is set as attacked ID decoded from IR received signals. If a player receives this signal the X should match with player ID such that the signal can be recognized as valid. The module we choose for this signal is a transceiver module that can switch between transmitter and receiver. In our initial design we used only one RF module. But later we found that due to the game logic, while RF module listens to any location signal it should also be able to send location signal at the same time. So we increase the number of RF modules and use separate micro controllers to control them since each of them requires SPI interface with micro controller.

2.24 Center processing unit (combined)

The center processing is where the gaming logic is stored, which takes all the gaming logic data as the inputs, process them and then feed them to the video processing unit and the LED display. The gaming logic are programmed as follows:

Player can select which drone he or she wants to play as by toggling the switches on the IR emitter and RF receiver circuit. There are 4 drone IDs available for selection. We accomplished this by using the two toggle switch by feeding the "HIGH and LOW" data to the ATmega328P in the IR emitter and RF receiver circuit . Right after the game has started, each drone will have a health points of 4, the score points of 0 and all 4 LEDs which indicate the remaining health point on. If drone A is being shot by done B, the health point of drone A will be decreased by 1, the score point of drone B will be increased by 1 and one LED will be turned off.

If one drone loses all the health point, then this drone is not able to shoot anymore. This drone loses the game. If there is only one drone left with health points greater than 0. Then this drone wins the game.

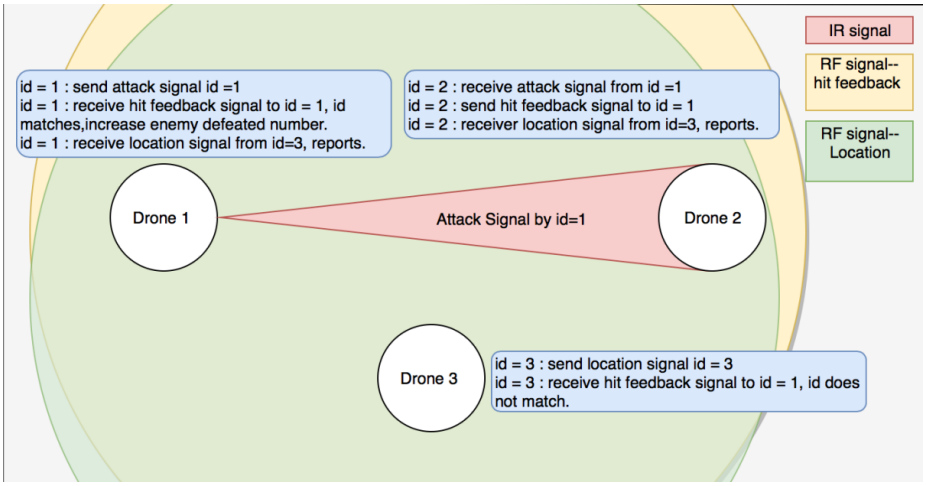

Figure 6: Multiplayer Interactions example

2.3 Video processing unit

The video processing unit will display the gaming logic to the screen and the player's goggle, making it easier for the player to know the in-game info. What we want to accomplish is something like figure with the specific layout as

2.31 design idea

The video processing unit utilizes the Atmega328 chip to receive and transmit the data obtained from camera and send it to max7456, the on-screen-display chip. The chip overlay the data onto the original graphics from camera and creates the user/player interface of the game. The MAX7456 single-channel monochrome on-screen display (OSD) generator lowers system cost by eliminating the need for an external video driver, sync separator, video switch, and EEPROM (a read-only memory whose contents can be erased and reprogrammed using a pulsed voltage.). The MAX7456 is preloaded with 256 characters and pictographs and can be reprogrammed in-circuit using the SPI (serial peripheral interface) port. After reading the datasheet. For the whole game logic, there are various basic game parameters we want to present: health-point, number of drones we have shot down, warnings when enemy is around, which player we are, and which drone attacked us (Figure 5). And most importantly, did we win or lose the game (Figure 6). The whole interface is shown once game on signal is sent from the controlled. And after a game on logo showing up shortly, the whole interface starts up. The left upper corner shows health-points, right-up corner shows attacker ID, and player ID (to support the multiplier mode, we utilized the digital pins and analog pins for assigning player ID for the drones, and it ideally supports up to 4 players) the right corner is how many drones we have shot down (Figure 5). The health-point stars at 4 health points. And when the player loses all the health points, the interface will go into lose status and flash (show "wasted" on screen, inspired by a game called GTA). And when the drone wins the game (all the enemies are eliminated), the screen will clean up and show "You Win".

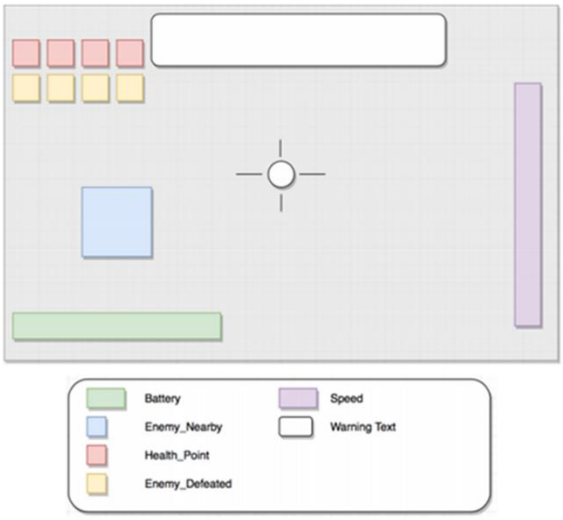

Figure 7: Specific screen layout for our game

Figure 8: A graphic overlay example

The video interface consists of different characters and symbols. And these different elements are stored in character pixel map. Each character pixel map has 18 X 12 pixels. Each pixel has 2 bits. Which means there are 3 bytes in a row and there are 18 rows. So, the whole map is 54 bytes. There are 3 color choice for the pixels (00 for white, 01 for black, and x1 for gray). Each of these characters pixel map is stored in character address memory. The character memory is 64bytes X 256 rows. Byte 0-53 is for the character pixel map data, and byte 54-63 are unused. The screen display area has 30 X 16 locations. To display the right character or symbol data onto the screen, we just need to put the right character address into the display area address (0-8 bit) and output a display signal to the display port.

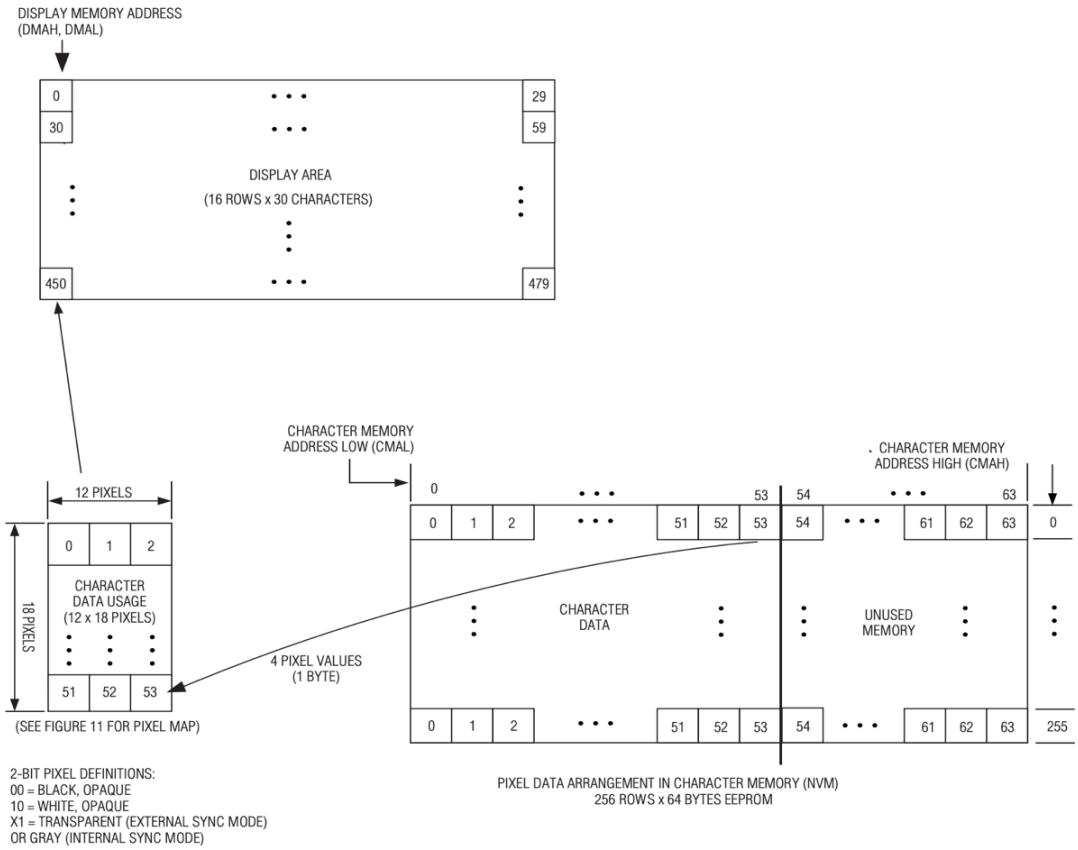

Figure 9: Graphic overlay design procedure

3. Design Verification

3.1 Power system

The method we used to test is to connect each module to the multimeter at ECEB 2072 Lab to test the voltage once applying the voltage to the PCBs. The voltage of each module when connected the PCB with power are listed in the following table (Table 1). And they are all working correctly after we supplied them with the voltage inputs from the PCB board.

Name	Working voltage	Voltage tested
TSOP4838	2.7V to 5.5V	5.0V
TSAL6199	2.7V to 5.5V	5.0V
LED display	1.7V to 3.3V	3.3V
ATmega328P	5V	5V
Nrf24l01	1.7V to 3.3V	3.3V

Table 1: Required voltage and tested voltage for each module

3.2 Control unit

The purpose of the control unit is to store the gaming logic and gaming actions. Gaming actions includes “Game on”, “Game end” and “send attack”. And the gaming actions includes actions like “send attack” and “hit feedback”. (Appendix A, Table 1)

We are able to encode the IR signal differently by toggling the switches on the PCB board. We connected the receiver end’s atmega328p serial output to the laptop to see the decoded IR message. And after toggled the switch, the IR signals being transmitted changed accordingly. We separate the IR part of the circuit to test it individually (Figure 9).

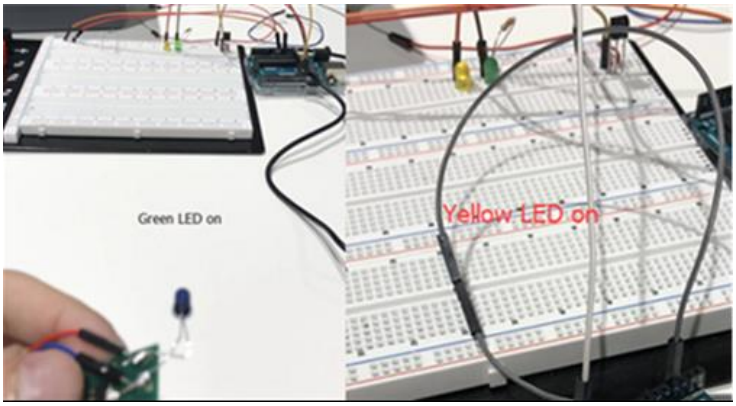

Figure 10: Image of IR encode/decode testing

In this verification process, we only encode 2 different 2 IR signals using Sony encoding protocol. If the receiving end receives and decides the first encoding value, the green LED light will turn on and yellow light will turn off. If the receiving end receives and decodes the second encoding value, the yellow LED light will turn off and the yellow light will turn off. (Figure 9). Please noted that this just for the testing for the ability to encode the IR message differently for 2 different IR message.

The complete 4 different IR encodings are verified when combined the IR and RF transceivers together (Figure 10). We are able to use RF transceiver to send back the hit feedback and surrounding drone signals. We connected the ATmega328p serial outputs from the IRTX&RFRX (Figure 12) and IRRX&RFTX (Figure 11) board to the laptop. And when shooting with the encoded IR code player4, the IR&RF transceivers can successfully transmit the enemy nearby and hit feedback signals. (Figure 10).

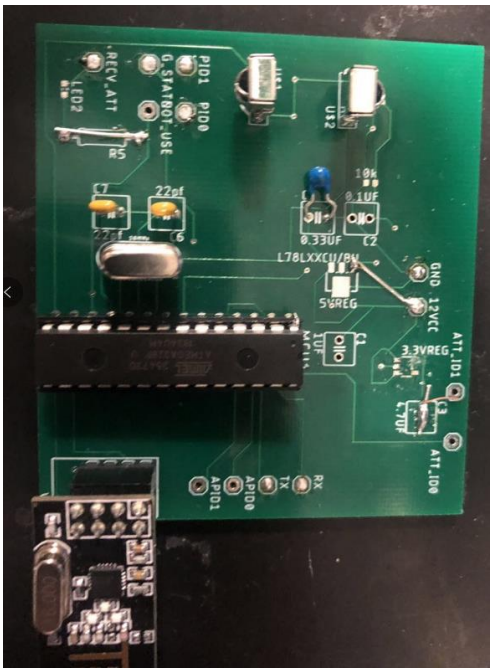

Figure 11: PCB for IRRX&RFTX

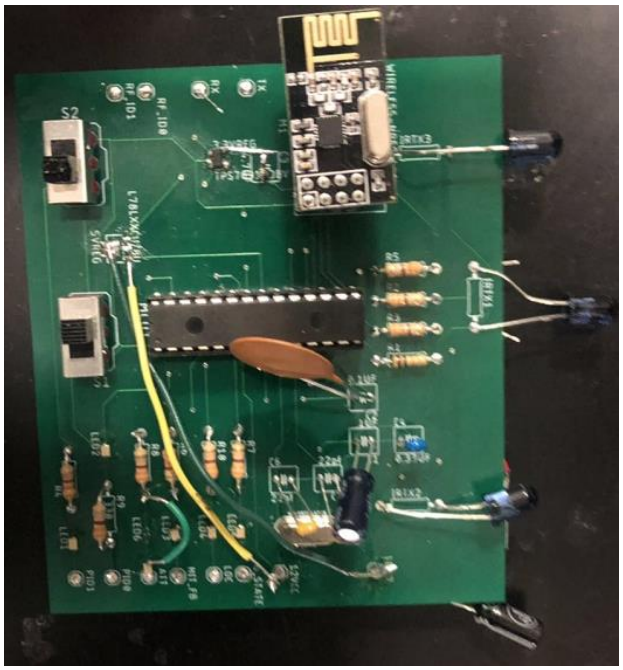

Figure 12:PCB for IRTX&RFRX

Figure 13: IR&RF testing output

3.3 video processing unit

We tested the video processing unit after we finished our PCBs for the IR and RF modules. And when feeding the video processing unit with the correct data, both the screen and the goggle will display the correct gaming contents.

In detail, we want to display We want to display the battery, speed, enemy nearby, health points and enemy defeated (AKA score points) on the specified location on the screen (Figure 13).

We tested our video processing unit by feeding the right data for the in-game display, our screen displayed the desired outputs. All features in the requirements (Figure 6), are included (Figure X).

When feeding the data for the end-game display when one player wins the game, our screen displayed the winning screen (Figure X).

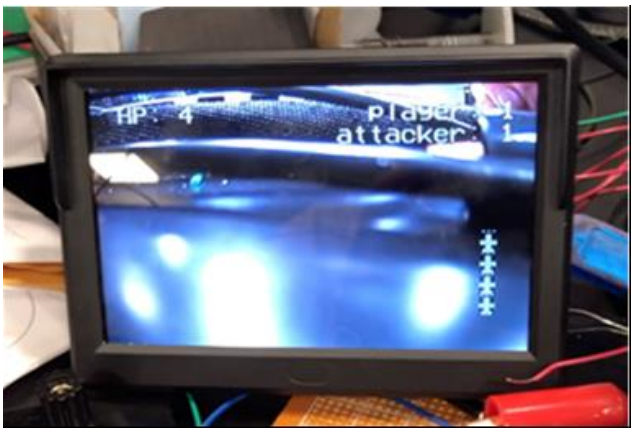

Figure 14: On screen display for specific layouts

Figure 15: Wining screen

4. Costs

4.1 Parts

Table X Parts Costs

Part	Manufacturer	Retail Cost ($)	Bulk Purchase Cost ($)	Actual Cost ($)
TSAL6100	VISHAY	0.24	6.84	12.24
HS0038B	GIMAX	2.43	10.69	10.69
TSOP4838	VISHY	2.55	12.55	12.55
MCP2120	MICROCHIP	1.89	8.99	13.79
Arduino	Arduino	34.16	68.32	68.32
Atmega328	MICROCHIP	3.42	20.48	20.85
max7456	Sparkfun	2.32	4.64	4.64
nRF24l91	MakerFocus	1.98	11.98	11.98

4.2 Labor

NAME	OBJECTIVE	HOUR	SALARY($)/HOUR	TOTAL($)
RF transceiver	Set up the RF communication	40.00	30.00	1200.00
IR transceiver	Set up the IR communication	30.00	30.00	900.00
OSD design	UI design	40.00	30.00	1200.00
Assemble	Assemble all parts	20.00	90.00	1800.00
Test	Test each part	50.00	40.00	2000.00

4.3 Total

6413.44 is our total cost in dollars.

5. Conclusion

5.1 Accomplishments

IR communication: We are able to encode the IR signal with the player's ID and decode the IR signal in order to read the player's ID. Also, we are able to send the IR signal in the selected direction when the distance of the IR transmission is longer than 1 meter.

RF communication: We are able to transmit the enemy nearby signals, hit feedback signals without noticeable delay successfully.

Video processing: for the Video processing part, we have accomplished all the desired interface we want to show including health point, enemy around warning, player ID, attacker ID, and numbers of drones we have shot down.

5.2 Uncertainties

Video processing part:

The interface gives wrong character sometime, I thought there was an overflow of the buffer and returned the wrong or random character, but after several trial of on screen displaying, the resulted characters are the same, which means it is not random character, and therefore it might be displace of the character order. And turns out, the preprogramed alphabet was indeed in some weird order. Since the loop is really fast in Arduino. To avoid multiple signal in short time, we used a previous signal to check if the current signal is only given once, but such logic is not giving the desired interface. To address the issue, I added some delay in the game to make the timeline correct and solved the problem.

5.3 Ethical considerations

We will formally and properly site the sources of the unitized data and info. And for our own development data, we will share most of the code and resources that we think might contribute to DIY drones industry. And since we will use the goggles for FPV shooting, there might be personal privacy recorded in game players' sight. All the video in the goggles will only be used for project developing and private images if show upon the videos, will remain confidential. We are responsible for the information that is sent through our technology. This spread of valuable knowledge is an implementation of the IEEE Code of Ethics, #5: "To improve the understanding of technology; its appropriate application, and potential consequences" [1]. We hope to bring education and communication to the most remote corners of the world. Unfortunately, risks surrounding the spread of information include piracy and mental health. Every day, people pirate music, movies, and even books via the conventional internet - and there is no reason to believe that our network will be any different. We are not explicitly giving out the tools to commit piracy or copyright infringement of any kind, but in a decentralized network it is impossible to track with any degree of certainty what information is shared. This would go against #7 and #9 of the IEEE Code of Ethics - the people committing piracy are not properly crediting the work of others, and they could be injuring the copyright holders by sharing content without paying for it [1]. We do not currently have a solution to this - we do not believe it would be the right course of action to limit the utility of our network simply because we anticipate a small subset of our users engaging in

piracy. On the internet, where a certain degree of anonymity is assured, there are fewer barriers to behaviors like cyberbullying. This type of harassment will adversely affect the mental health of those on the receiving end. It is entirely possible that the network will be used to discriminate by race, gender, or sexual orientation, violating #8 of the IEEE Code of Ethics, “to treat fairly all persons and to not engage in acts of discrimination based on race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression” [1]. We plan to introduce the ability for node owners to “ban” certain devices from services hosted on their node. The “banned” user would have no knowledge of this action; their packets would simply not return a response as the node hosting the service would throw them away instead of processing them. We believe this is the best course of action - any harassment can be stopped by an automated system, and the harasser(s) will never know that their messages aren’t being delivered. Our mitigation techniques align with the IEEE Code of Ethics, #1: “To accept responsibility...” [1]. There are many risks that present themselves as a consequence to access and free communication, but we believe that the advantages of open resources, which include free education and the potential for economic development, far outweigh the potential negative effects.

5.4 Future work

IR communication: For the emitter part, we will add more IR emitters to increase the distance and intensity of IR transmission. We will connect all IR emitters in parallel in order to do so. For the receiver part, we will add two more IR receivers. Adding two more IR receivers will enable our drone to receive IR messages from all directions. Whereas our current version can only support back and front receiving.

RF communication: We will choose a more stable RF module

Video processing part: we’d like to show fancier interface in the future. To accomplish such goal, we need to change to a more updated version of on-screen-display chip to create the special effects and more RGB pixels. And if possible, we would also include radar into the graph to show not only a warning that enemy is around, but also the direction in which enemy is coming from

References

[1]. Ieee.org, "IEEE IEEE Code of Ethics", 2016. [Online]. Available:http://www.ieee.org/about/corporate/governance/p7‐8.html. [Accessed: 20‐ Feb‐2018].

[2]. Atmega328p datasheet:2018[Online].Available:http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf[Accessed: 20‐ Feb‐ 2018].

[3]. Atmega328p bootloader:2018[Online].Available:https://www.instructables.com/id/Burning-the-Bootloader-on-ATMega328-using-Arduino-/[Accessed: 20‐ Feb‐ 2018].

[4]. MAX7456 datasheet [Online].http://freedatasheets.com/datasheet-download/1c272b384139988aac37d442d8555c1d/MAX7456 [Accessed: 28‐ Feb‐ 2018].
[5]. Arduino chinese tutorial:2018[Online].Available:https://zhuanlan.zhihu.com/p/36839509[Accessed: 20‐ Feb‐ 2018].

[6]. HOW TO CHOOSE OSD FOR QUADCOPTER 2018[Online].Available:https://oscarliang.com/best-osd-quadcopter-fpv-data-on-screen-display-video/[Accessed: 20‐ Feb‐ 2018].

[7] DIY OSD Discussion blog [Online] https://www.rcgroups.com/forums/showthread.php?1776186-Diy-osd-%28MAX7456-version-OpenSource%29 [Accessed: 22‐ mar‐ 2018].

[8] Rafael Reinhold, Coverage range analysis of IEEE 802.15.4a IR-UWB for reliable data transmission in wireless sensor networks. 7-8 Oct. 2013. [E-book] https://ieeexplore.ieee.org/abstract/document/6663772

[9] Podkalicki, Ł. (2019). ATtiny13 - IR remote to control LEDs | Łukasz Podkalicki. [online] Blog.podkalicki.com. Available at: https://blog.podkalicki.com/attiny13-ir-remote-to-control-leds-nec-proto/ [Accessed 25 Mar. 2019].

[10] Learn.sparkfun.com. (2019). IR Communication - learn.sparkfun.com. [online] Available at: https://learn.sparkfun.com/tutorials/ir-communication/all [Accessed 25 Mar. 2019].

Appendix A Requirement and Verification Table

Part name	Requirements	Verification
RF communication between Radio controller and the MCU on the drone	Radio controller is able to send "Game On" signal to the MCU on the drone	Check the screen display if the "Game on" Mode is enabled when flip the game on button on the radio controller
	Radio controller is able to send the "Attack signal" to the "IRTX&RFRX" PCB on the drone	1: check if the drone that is being shot at has lose health point on the display screen 2: Check if the IR receiver can detect the IR signal from the drone.
	Radio controller is able to within 20meters range (we can do 50 if the space is allowed)	Using the "Game On" method to check. Move the drone 20 meters away from the radio controller.
	Radio controller will not have noticeable delay when controlling the drone.	We can observe instant feedback from the display screen when an action is performed
IRTX&RFRX Module	Able to receive the RF signal from the Radio controller "Send attack"	Can be seen from the gameplay
	Able to send the IR signal through the IR emitter on the PCB once receive the "send attack" signal	If pointed to the IR receiver PCB, the health point of another drone will decrease by 1.
	Able to send the IR signal at a distance of at least 10 meters.	Put 2 drones 10 meters away to see if the IR signal can successfully transmit.
	Able to receive the IR signal at the receiving end and able to distinguish the different IR signals	Player will know being shot by whom
	After being shot, the drone being shot at will send back a hit signal to the drone shooting, indicating that the shot was successfully landed	on the player 1 screen display, the score point will add one if the shot is successful

	Able to know if the drone has successfully landed the shot	If the health point of one drone is decreased by 1, then the drone shoots the IR signal will have 1 game point earned.
OSD	graphics can be modified to the desired interface	if the graphics are showing up, whereas the interface is not showing the correct info, there must be something wrong with the coding part.
	interface should show the real-time health-point, enemy around signal or other interface features with acceptable graphical delay	interface graphic will show up the current health of the airdrome. Hit signal is sent by enemy air-drone from IR transmitter, and the RF transmitter will transmit the shot-on-target signal to the air-drone under attack. To make sure the transmitting rate has no significant delay, we will let IR transmitter of air-drone to shoot at the IR receiver. If the graphics of HP (health point) for air-drone B are in an acceptable delay range.

Table 2: Requirement and verification

Appendix B: Circuits and PCB design

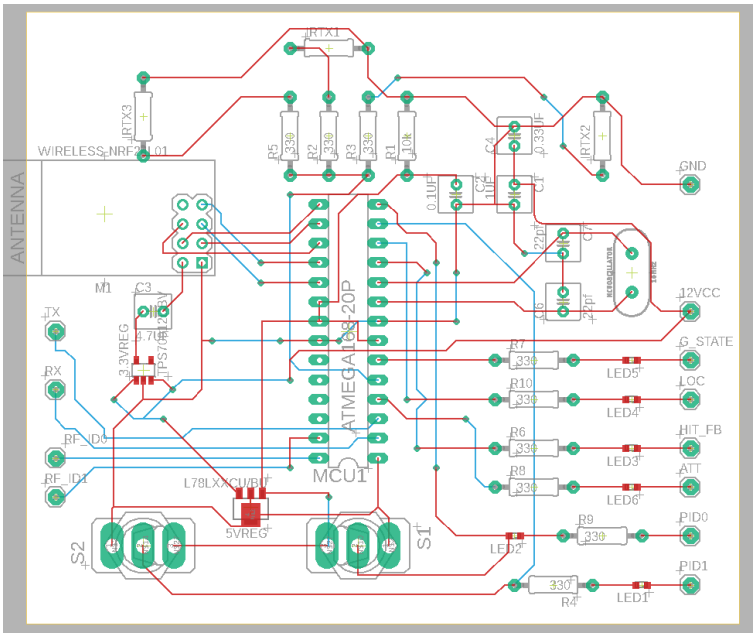

Figure 17:Circuit design for IR emitter and RF receiver

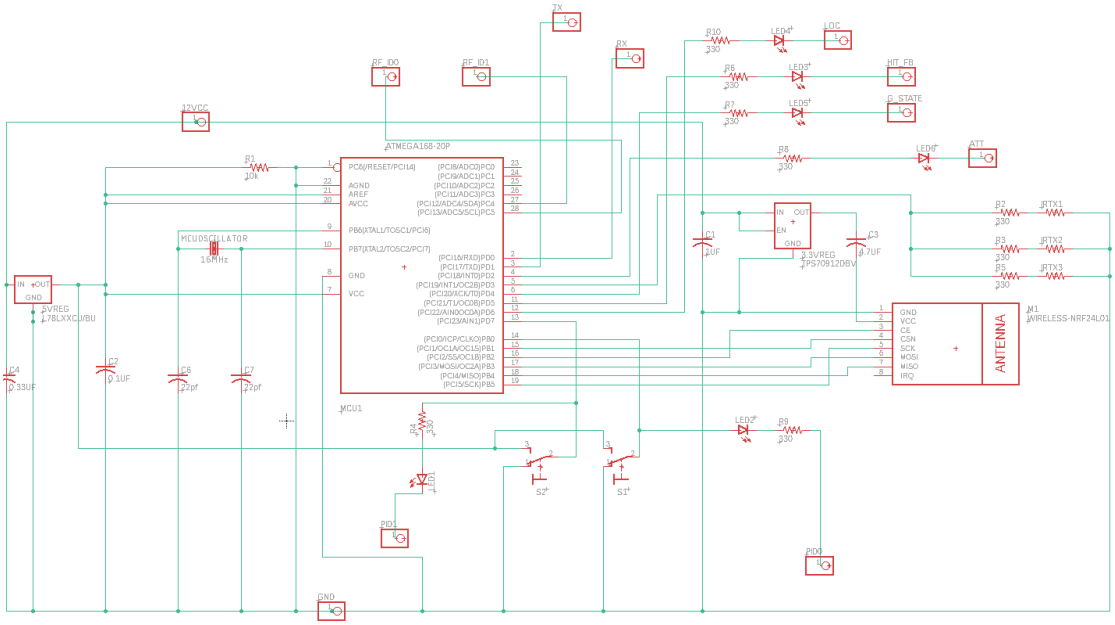

Figure 16: PCB design for IR emitter and RF receiver

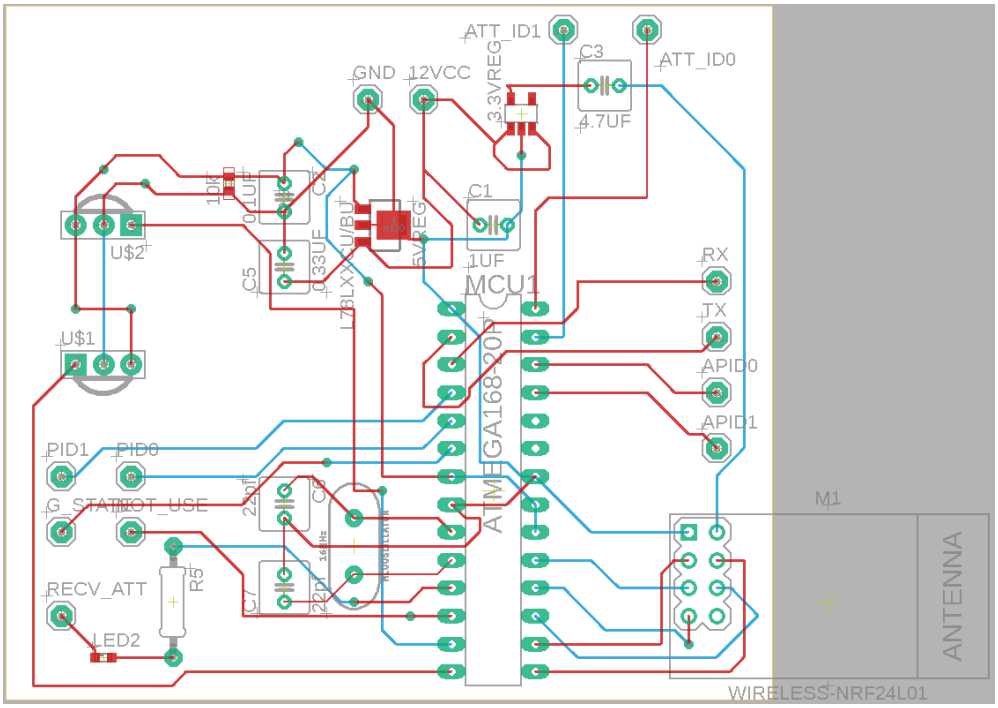

Figure 19:PCB design for IR Emitter and RF receiver

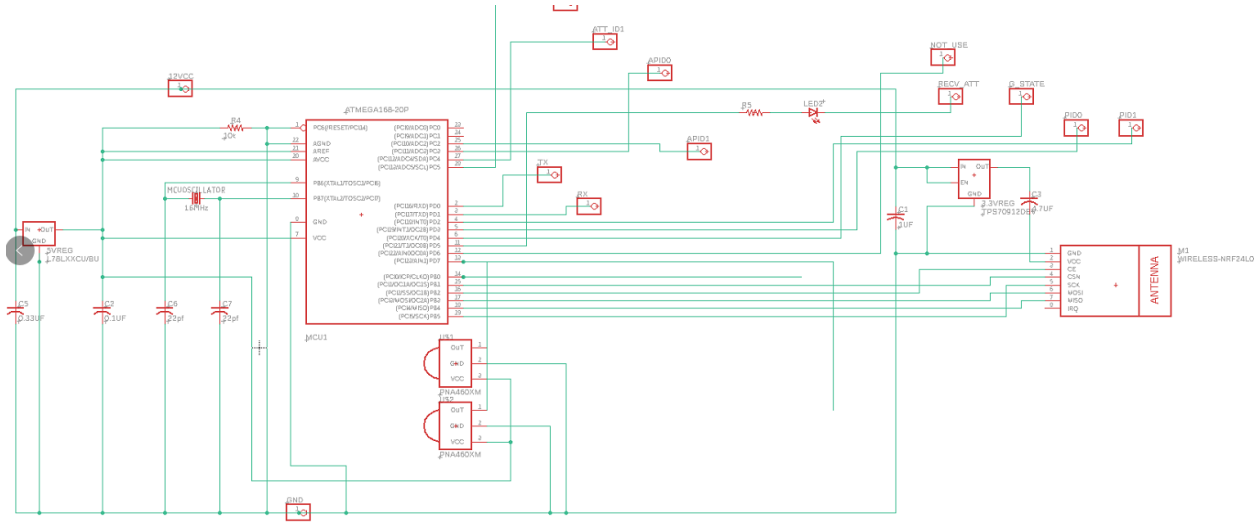

Figure 18:Circuit design for IR emitter and RF receiver

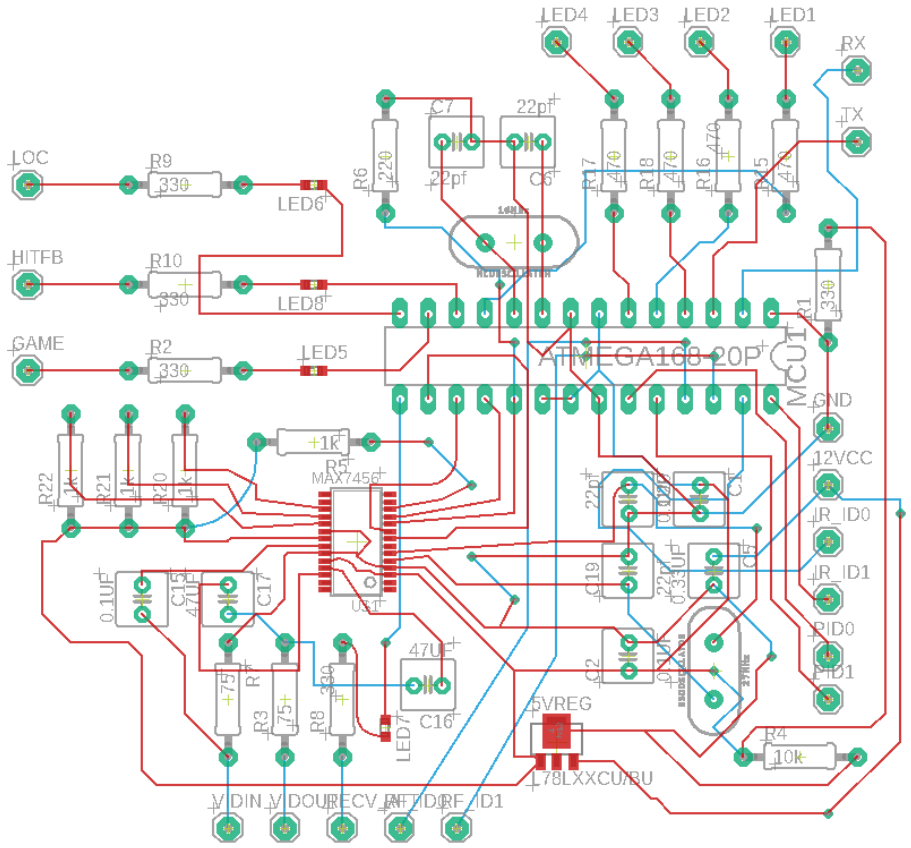

Figure 20:PCB for video processing unit

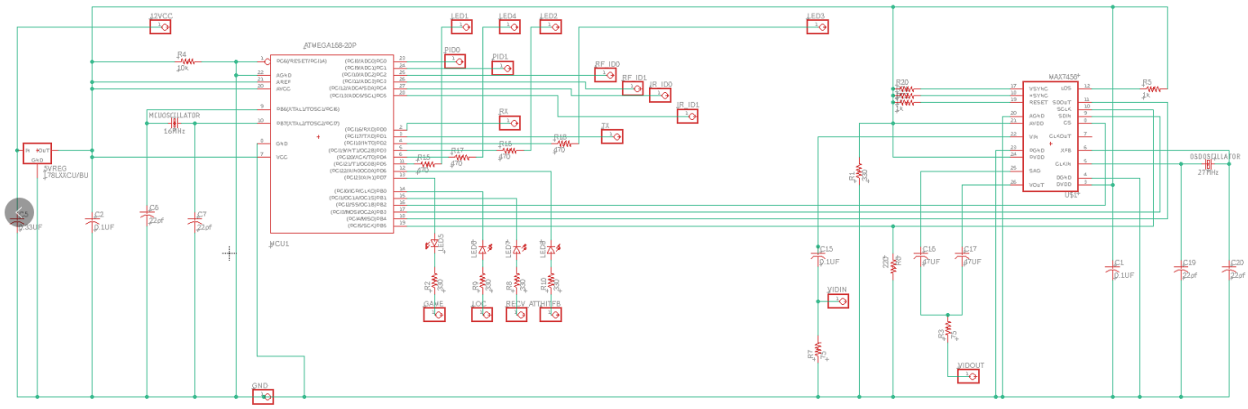

Figure 21:Circuit for video processing unit