RECONFIGURABLE AERIAL COMPUTING SYSTEM: DESIGN AND DEVELOPMENT

Yixin Gu

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

August 2018

Gu, Yixin. *Reconfigurable Aerial Computing System: Design and Development*. Doctor of Philosophy (Electrical Engineering), August 2018, 116 pp., 12 tables, 111 figures, 1 appendix, 39 numbered references.

In situations where information infrastructure is destroyed or not available, on-demand information infrastructure is pivotal for the success of rescue missions. In this paper, a drone-carried on-demand information infrastructure for long-distance WiFi transmission system is developed. It can be used in the areas including emergency response, public event, and battlefield.

In years development, the Drone WIFI System has developed from single-CPU platform, twin-CPU platform, Atmega2560 platform to NVIDIA Jetson TX2 platform. By the upgrade of the platform, the hardware shows more and more reliable and higher performance which make the application of the platform more and more exciting. The latest TX2 platform can provide real time and thermal video transmission, also application of deep learning of object recognition and target tracing. All these up-to-date technology brings more application scenarios to the system. Therefore, the system can serve more people in more scenarios.

ACKNOWLEDGEMENTS

I would never have been able to finish my dissertation without the guidance of my committee members, help from friends, and support from my family.

I would first like to thank my major advisor Dr. Shengli Fu, who has the patience and passions throughout my journey in the doctoral program. I would like to express my deepest appreciation, for his excellent guidance, caring, and patience, and for providing me with an excellent atmosphere for doing research in last seven years. His expertise in communication, wireless networking, and embedded systems improved my research skills and prepared me for future challenges.

I would also like to thank my committee member, Dr. Yan Wan, who has advised me in almost all aspects in my doctoral research. I am grateful to Dr. Xinrong Li for his helpful suggestions and valuable comments during my Ph.D. program. In addition, I am grateful to Dr. Shu-Shaw Wang for providing advice from both industrial and academic point of view.

I want to thank all my lab mates for their help, support, and cooperation. Finally, I want to express special thanks to my family for a solid source of inspiration and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The objective of this research is to explore the drone as a platform for various applications involving computing and communication. In last decade, the drones or unmanned aerial vehicles (UAVs) have seen strong potential in areas of transportation [1], environmental monitoring [2], public safety [3-5], and agriculture [6]. It also attracts tremendous research interests in communication [7-11], control [12-14], computer science [15-17], antennas [18], and cyber-physical system (CPS) [19-20].

One challenge for drone applications and research is that researchers have to design and develop the system from scratch, which typically involves multi-disciplinary topics such as power, mechanics, control, and programming. This imposes researchers' extra effort in addition to the focused research interests. Motivated by this opportunity, this research proposes to design and develop a reconfigurable aerial computing system which is characterized by modular design, plug-and-play, and integration with communication and control.

As an example, for the proposed framework, we developed the "drone-carried WiFi" system to provide on-demand broadband communication for emergency response. Disasters such as earthquake and hurricane could strike down communication infrastructures which not only disconnect victims but also block the first responders to a full picture on the extent of damage. In our work [5, 21, 22], we propose the drone-carried WiFi system for a temporary communication solution for both survivors and rescue teams. For example, through the WiFi networks, victims can communicate with outside world as soon as possible. They provide their locations and heathy status such that rescue teams can find them to provide medical services as quick as possible. Also, the rescue teams can obtain latest commands from operation center to perform rescue tasks. As the system comply with the standard WiFi protocols, people could access the service as long as they have WiFi terminal such as smartphone, laptop, and tablet, which are very common communication gadgets. The broad band provided by the system also enables services like real-time video transmission for a more efficient communication.

As shown in Fig. 1.1, the drone-carried WiFi system consists of two identical drones. Each drone carries with a directional antenna, so the two antennas can communicate to each other when they face to each. Because the antennas have very narrow RF waveform beam, so the transmission data rate is quite sensitive to azimuth degree. Therefore, to guarantee high data rate, each directional antenna is installed on a rotary platform driven by a DC motor to keep the two antennas always face to each other precisely.

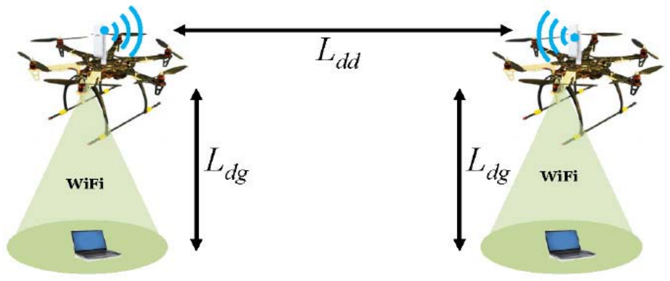


Fig. 1.1: Drone WiFi system structure[*]

Fig. 1.2 shows the network topology. One directional antenna is configured in AP mode which connects the Internet through a RJ45 cable. The other directional antenna is

[*] Fig. 1.1 is reproduced from: [5] Y. Gu, M. Zhou, S. Fu, and Y. Wan, "Airborne WiFi networks through directional antennae: An experimental study", IEEE Wireless Communications and Networking Conference (WCNC), 1314-1319, 2015 and [22] J. Xie, F. Al-Emrani, Y. Gu, Y. Wan, and S. Fu, "UAV-carried long-distance Wi-Fi communication infrastructure", *AIAA Infotech@ Aerospace*, 2016.

configured in Station mode, which broadcasts WiFi signal to ground users (smartphone, laptop, etc.). Other accessories include power supply, DC motor, micro controller

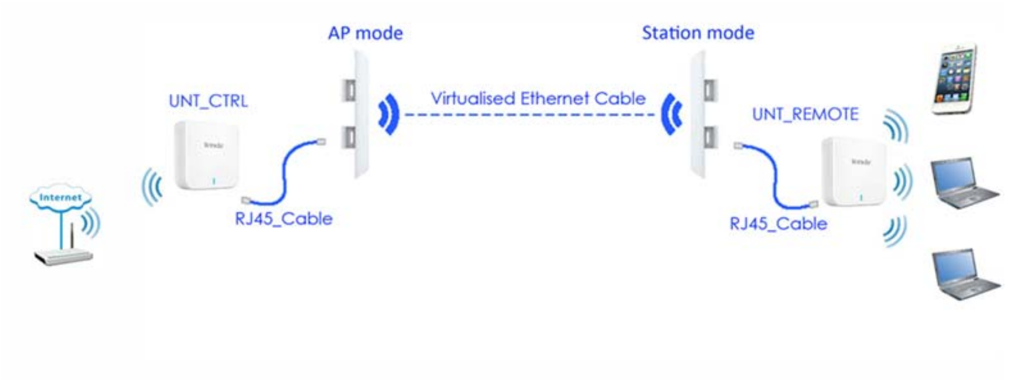


Fig. 1.2: Network topology

In this dissertation, we first introduce an implementation of drone-carried WiFi system with a simple Arduino microcontroller. While the Arduino version can support full function for the drone operation and WiFi communication, it is not powerful enough for computation intense applications due the limited capacity of Arduino microcontroller. We then develop the system with Jetson TX2 from NVIDIA [23]. The Jetson TX2 is the latest GPU module designed specifically for UAVs and AI computing at the edge. We describe in detail on the design and development of Jetson TX2 carrier board, which could be installed over different drone systems. Field tests and measurements are also provided in this research.

CHAPTER 2

SINGLE-CPU PLATFORM

2.1 Directional Antenna, Wireless Router and Drone Selection

2.1.1 Directional Antenna

Comparing to omni-directional antennas, directional antennas have longer transmission distance due to the focus of transmission energy in one direction. Since the antennas are deployed over small UAV, special considerations are needed including the size, weight, and power consumption. After comparison of commercial available products, Nano Station2 made by Ubiquity [24] is selected due to the overall performance in terms of bandwidth, weight, and size. The Nano antennas works at 2.4GHz frequency range, supports 802.11b/g with data rate up to 54Mbps. It has light weight of 180g and low power consumption of 15V x 0.2A. These features make Nano Station2 a good solution for the drone platform.

Fig. 2.1: Nano Station2 directional antenna [24]

2.1.2 Wireless Router

Tenda A6 (Fig. 2.2) is a small and powerful wireless router. The technical specifications include 5cm X 5cm size and 5V x 0.3A power consumption [25]. It supports 802.11b/g/n with two built-in omni-directional antennas. It also has three different working modes: router, repeater and WiFi client. In our system, the router mode will be enabled such that it could relay the signal from Nano Station2 to the devices on the ground.

Fig. 2.2: Tenda A6 wireless router [25]

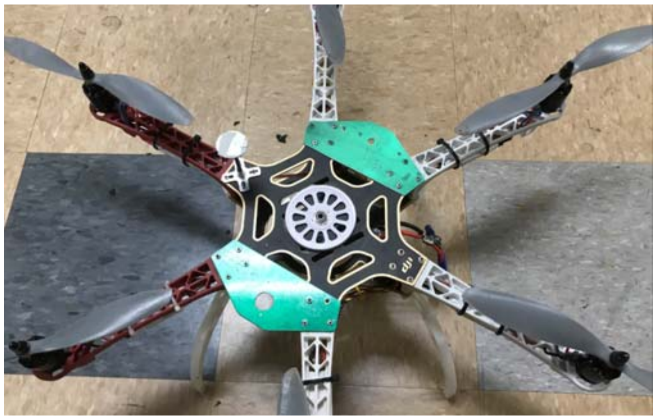


Fig. 2.3: DJI F550 drone [26]

2.1.3 Drone Selection

The drone used in the project is DJI F550 hexcopter [26] (Fig. 2.3). Comparing to quadcopter, the hexcopter is more stable with more payload. The upper and lower frame is PCB which means it is easy to install the rotary antenna part (Fig. 2.4).

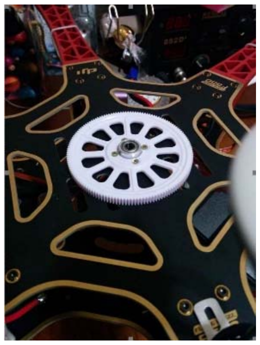


Fig. 2.4: Installation of gear on upper frame

Fig. 2.5: DJI NAZA flight controller [26]

The material of the six arms is ABS (Acrylonitrile butadiene styrene) plastic. It has features including light weight, strong, cheap and easy for replacement etc. It also comes with NAZA flight controller (Fig. 2.5) kit including GPS, barometer and accelerometer sensors to provide stable operation experience.

For the battery, two 11.1V 7500mAH Li-Ion cells are configured in parallel to provide up to 30 minutes hovering time (no payload). As the whole directional antenna system has weight of 480g. With this payload, the two Li-Ion cells could provide 18 minutes flight time. The batteries are charged by the EOS720i (Fig. 2.6) twin-way balanced charger with 600W switching power adapter. The two cells can be fully-charged within one hour. For the discharge, the first battery alarm is set at 3.7V/each cell, 11.1V total and the drone alarms by slow blinking red on the tail LED; the second alarm is set

3.5V/each cell, 10.5V total and the drone lands itself automatically. After landing, the batteries can recover a little bit from 10.5V to usually 11V. Charging indicates the 10.5V leaves about 1000MAh inside of the batteries so they're not totally drained to avoid premature death.

Fig. 2.6: EOS720i battery charger [32]

Fig. 2.7: FUTABA 72M RC

Due to the FAA regulation, the drone system still need a remote controller (RC) which is operated on the ground. To avoid possible interference, a classic 72MHz RC (Fig. 2.7) is used instead of up-to-date frequency-hopping-spread-spectrum (FHSS)

2.4GHz equipment. The 2.4GHz FHSS RC uses the same frequency as the WiFi signal, which introduces interference to each other. As shown in Fig. 2.7, the FUTABA 72MHz 9-channel RC is selected for the project.

2.2 System Description

In this system, a single micro controller: Atmega328P made by ATMEL is adopted to build a system [28]. In this system, the micro controller is used to read current magnetometer azimuth degree to see how much bias it is comparing to north, calculate the error and drive a DC geared motor to offset the difference so that the antenna always points to the north (block diagram see Fig. 2.8). The antenna sub-system is a rotary platform which is detachable to the drone. Therefore, even if turning the drone, the magnetometer on the antenna reads current azimuth degree, calculate the difference and drive the motor toward the opposite direction to offset the difference.

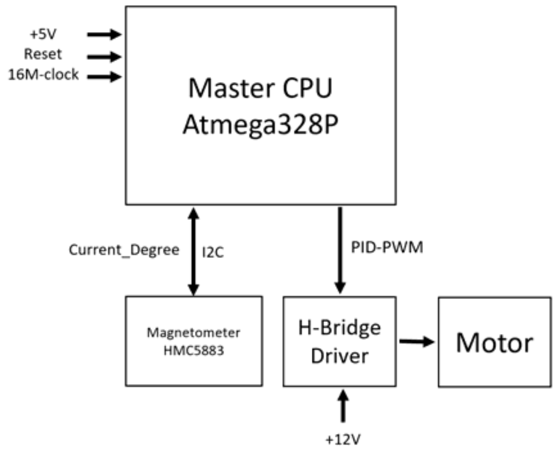


Fig. 2.8: Block diagram of single-CPU platform

To flexibly control the antenna degree, in case not pointing to north, a XBEE wireless module is integrated in the system so it communicates with a ground controller.

Fig. 2.9 shows the system diagram. On one hand, the XBEE on the drone sends information including altitude, height and battery voltage to the ground controller; on the other hand, the ground controller sends command to drone to choose a new azimuth degree other than 0 degree (north).

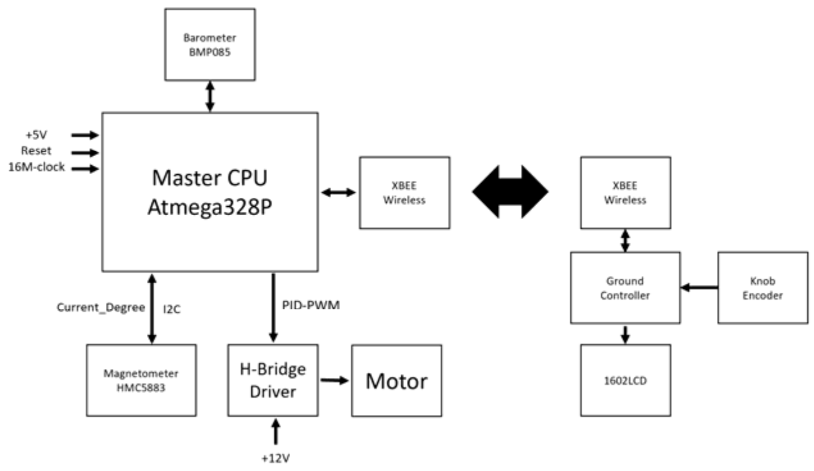


Fig. 2.9: System diagram of single-CPU system

2.3 Circuit and PCB Design

The antenna turntable PCB design and fabrication are shown in Fig. 2.10. The Atmega328P CPU has a 16MHz crystal, reset circuit and +5V power supply. The I2C bus connects two devices: BMP085 barometer and HMC5883 magnetometer. The barometer occupies address 0x77 and the magnetometer occupies address 0x1E so they can be separately visited by the microcontroller. The Atmega328P has 20 GPIOs; however, only 5 of them could support pulse width modulation (PWM) output. Therefore, the two motor PWM outputs are limited in these 5 GPIOs. The H-bridge circuit is consisted by all transistor circuit which is easy to build and repair.

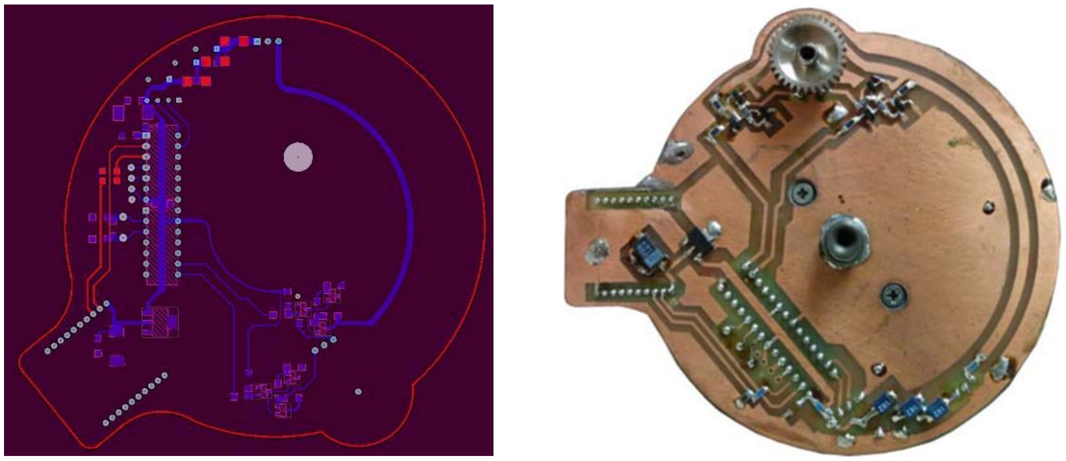


Fig. 2.10: Altium10 designed PCB

The power is designed to be two-way DC-DC power supplies. One way is a DC-DC boost circuit LM2596 to increase the 12V source to 15V output for the Nano antenna. The other way is to convert the 12V input into +5V (up to 3A) through a DC-DC buck power supply LM2577. The +5V is allocated to power the micro controller, the magnetometer and XBEE module on the main PCB. In Fig. 2.11, the power circuit module is small enough to be developed inside of the Nano antenna, so it doesn't occupy any extra space.

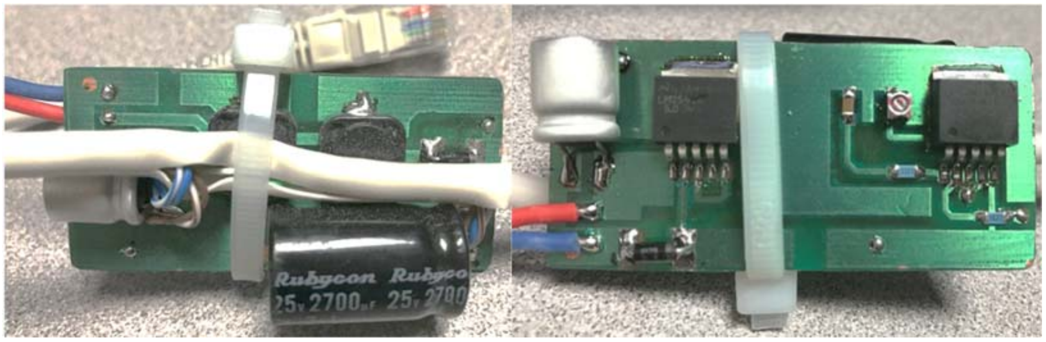


Fig. 2.11: Twin-way DC-DC power converter

Also, the 12V battery is directly connected to the H-Bridge circuit to drive a geared motor. The geared motor drives the antenna system either clockwise (CW) or count clockwise (CCW).

The ground controller (Fig. 2.12) is also developed with the Atmega328P. It has two external components: LCD1602 and XBEE wireless module. The 1602LCD is used to display information including altitude, height and battery voltage. Before taking off, the air pressure will be read by barometer on the drone and sent to the ground controller. At this time, press the blue button to set current pressure as 0. Once taking off, the new air pressure will be subtracted the saved ground air pressure and will show the current height. Rotating the black knob, a new azimuth degree changes from 0. Press the black knob to send the new degree once it's determined. The antenna on the drone will be driven to the new azimuth target degree.

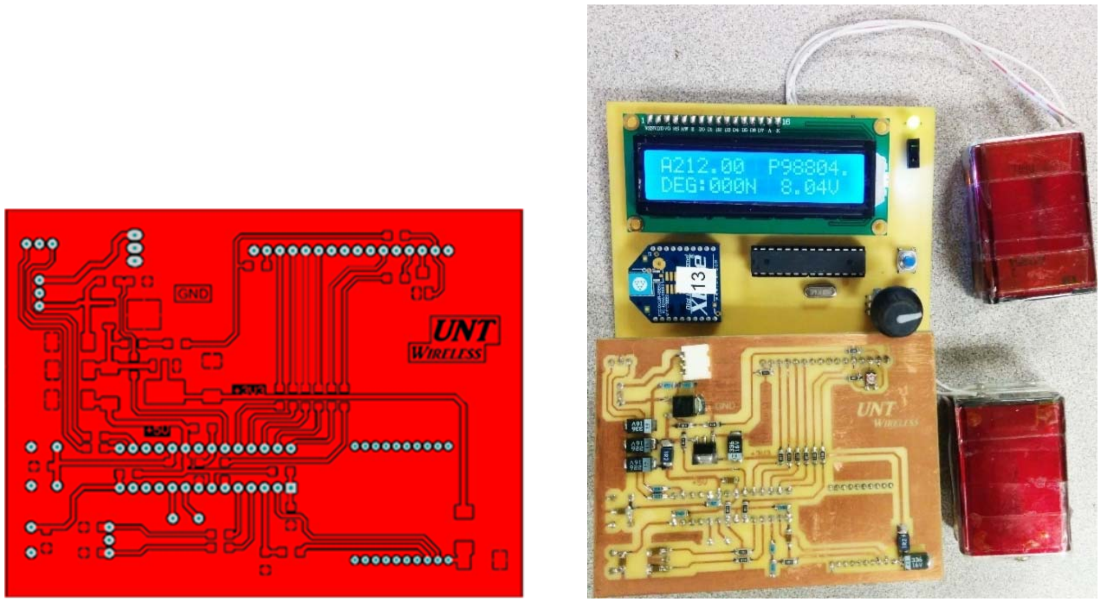


Fig. 2.12: Ground controller

2.4 Assembling

The upper frame of the drone is installed with a white 144-tooth gear to provide opposite torque to drive the antenna assembly. In the middle of the gear, a 5mm bearing is installed in a bearing holder to hold a 5mm shaft connected with the turntable. To hold the shaft firmly, another 5mm bearing and holder are installed on the lower frame of the drone. Once the geared motor on the antenna system rotates, the 32-tooth gear attached on the motor will drive the white gear for rotation.

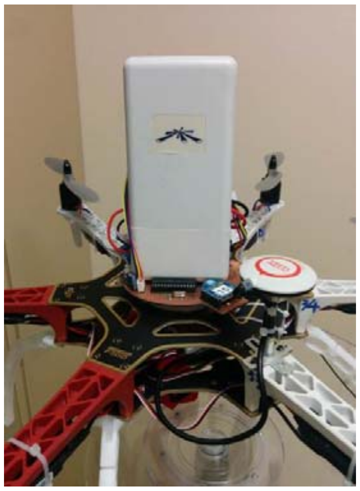


Fig. 2.13: Assembling of antenna

2.5 Software Design

Fig 2.14 shows flow chart of the control protocol. Each function will be called in main loop by sequence. The main loop is checking the magnetometer and tuning the PWM signal to drive the motor. There’s a one second timer in the main loop. Once the timer triggers, the XBEE checks whether there’s any incoming new degree demand. If yes, the new degree replaces the current degree and the antenna locks at this new degree. Also, the XBEE sends current battery voltage and barometer read out to the ground controller. Then, the ground controller calculates the height and displays all these information on the LCD screen.

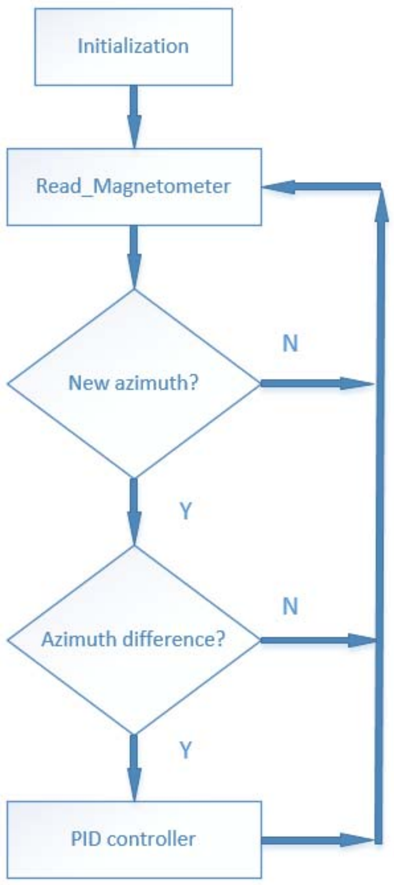


Fig. 2.14: Flow chart of control protocol

2.6 Test and Conclusion

After assembling, a test is performed. Once powering on, the motor drives the antenna to the north. If rotating the drone horizontally, the antenna rotates to opposite direction to maintain facing north. Even if rotating the antenna continuously CW and CCW, the antenna can response to opposite direction immediately.

Turns on the ground controller, the LCD on the controller displays three information about air pressure, height and battery voltage. Before taking off, the ground air pressure is “98990” which means ground air pressure 0.98x106 Pa. After taking off, the drone sends back air pressure “99079”. By calculating height equals air pressure*3.2808 – 275, LCD shows height: 17M. If a new azimuth is desired, rotating the knob to show correct degree and press it to send new azimuth. The knob is actually a rotating encoder. The micro controller uses two interrupts INT0 and INT1 to read the pulses; once desired degree appears, press the knob to transmit the new degree. E.g. the new degree 90 is sent to the drone, the antenna will rotate to the east and maintains the east.

Table 2.1: Throughput Summary: (Unit: Mbps)

Ldd (Unit: M)	UAV Status	Median	Average	ST Deviation
150	Static	42.4	44.2	6.0
	Hover	40.1	36.2	11.9
300	Static	36.1	35.9	5.7
	Hover	19.9	18.4	9.2
1000	Static	12.4	13.4	7.0
	Hover	8.2	7.8	6.3

CHAPTER 3

TWIN-CPU PLATFORM

3.1 Exist Issue in Single CPU Platform

In the previous version of design, an Atmega328P is used as the core controller to read magnetometer and output PWM signal to drive turntable motor so that the directional antenna installed on the turntable could point to the desired degree. However, the desired degree is pre-defined before flight or manually determined during flight. The two drones should be fixed along a straight line during hovering in the sky. Once either drone deviates from the line, the communication quality between the two directional antennas deteriorates. The communication can even fail if the deviation is too large (see Fig. 3.1).

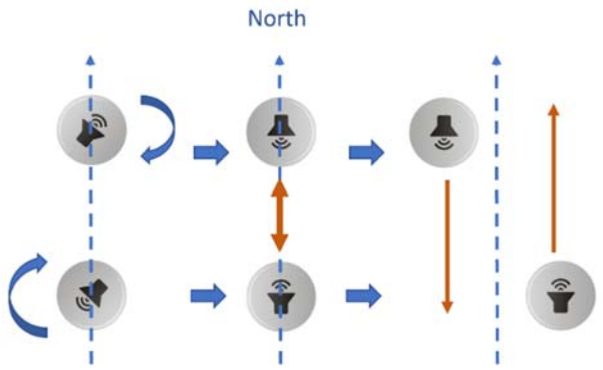


Fig. 3.1: Single-CPU alignment shortage

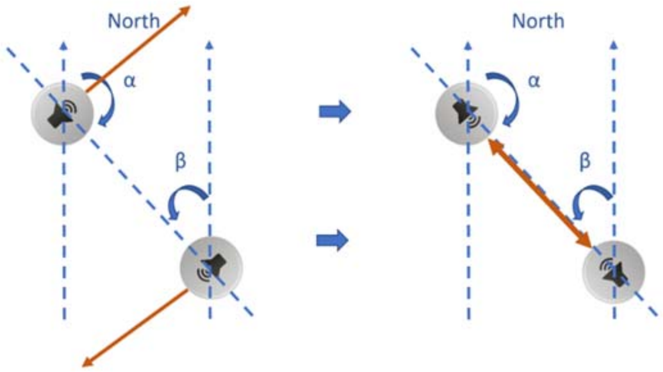


Fig. 3.2: Twin-CPU platform

3.2 Solution Concept

Due to the cons, a GPS module is considered a good solution to be integrated into the new design and provides precise coordinate locations in Fig. 3.3. In this design, an EM506 GPS module is selected. The GPS is built by SiRF Star IV high performance GPS chipset and has very high sensitivity (-163dbm). It has compact size 30.00 x 30.00 x 10.70 mm and weights 23g. It supports NMEA V3.0 outputs GGA, GSA, GSV, RMC, VYG, GLL, ZDA formats.

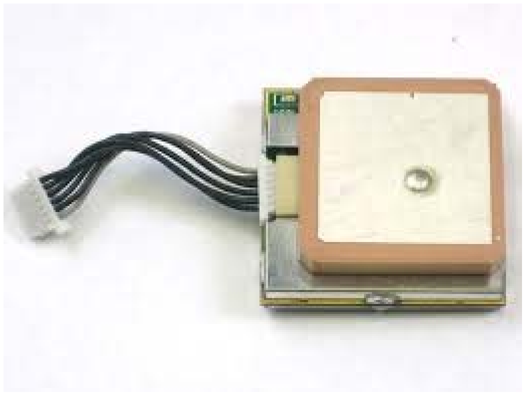


Fig. 3.3: EM506 GPS module [33

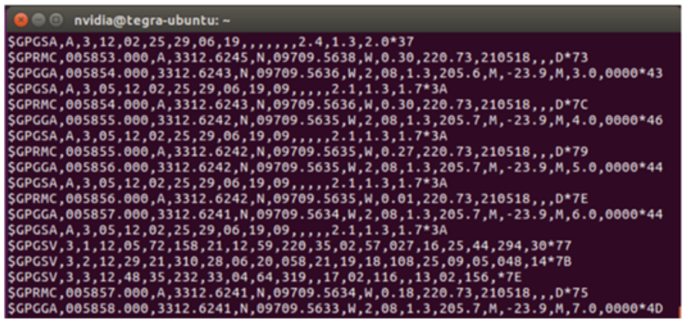
To illustrate the usage of GPS, a screenshot is provide in Fig. 3.4, where the location is obtained from the output of GPS:

```
$GPGGA, 005858.000, 3312.6241, N, 09709.5633, W, 2,08,1.3,
205.7, M, -23.9, M, 7.0 ,0000*4D
```

In this output,

- GP represents that it is a GPS position (GL denotes GLONASS).
- GGA represents that it contains information of latitude and longitude.
- 005858.000 is the time stamp: UTC time in hours, minutes and seconds which means 00:58:58. Time zone of Dallas is -6 plus DST 1 hour, so the real time is 19:58:58.
- 3312.6241, N: means North latitude DDMM.MMMMMM, after conversion: 33+12/60+0.6241/60=33.210401.

- 09709.5633, W: means West longitude DDDMM.MMMMMM, after conversion: 97+9/60+0.5564/60=97.159388.
- 2: means ”Differentially correct coordinate”, also: 1: means “uncorrected coordinate” 4:” RTK Fix coordinate”, 5: “RTK Float”.
- 08: means numbers of satellites used in the coordinate.
- 1.3: means the HDOP (horizontal dilution of precision).
- 205.7 means: altitude of the GPS
- M: means unit of altitude, Meter or Feet
- -23.9: means geoidal separation (subtract this from altitude to obtain Height Above Ellipsoid(HAE))
- M: means unit of altitude, Meter or Feet
- 7.0: means the age of the correction
- 0000: means correction station ID
- *4D: means the checksum



```
nvidia@tegra-ubuntu: ~
$GPGSA,A,3,12,02,25,29,06,19,,,,,,,2.4,1.3,2.0*37
$GPRMC,005853.000,A,3312.6245,N,09709.5638,W,0.30,220.73,210518,,,D*73
$GPGGA,005854.000,3312.6243,N,09709.5636,W,2,08,1.3,205.6,M,-23.9,M,3.0,0000*43
$GPGSA,A,3,05,12,02,25,29,06,19,09,,,,,2.1,1.3,1.7*3A
$GPRMC,005854.000,A,3312.6243,N,09709.5636,W,0.30,220.73,210518,,,D*7C
$GPGGA,005855.000,3312.6242,N,09709.5635,W,2,08,1.3,205.7,M,-23.9,M,4.0,0000*46
$GPGSA,A,3,05,12,02,25,29,06,19,09,,,,,2.1,1.3,1.7*3A
$GPRMC,005855.000,A,3312.6242,N,09709.5635,W,0.27,220.73,210518,,,D*79
$GPGGA,005856.000,3312.6242,N,09709.5635,W,2,08,1.3,205.7,M,-23.9,M,5.0,0000*44
$GPGSA,A,3,05,12,02,25,29,06,19,09,,,,,2.1,1.3,1.7*3A
$GPRMC,005856.000,A,3312.6242,N,09709.5635,W,0.01,220.73,210518,,,D*7E
$GPGGA,005857.000,3312.6241,N,09709.5634,W,2,08,1.3,205.7,M,-23.9,M,6.0,0000*44
$GPGSA,A,3,05,12,02,25,29,06,19,09,,,,,2.1,1.3,1.7*3A
$GPGSV,3,1,12,05,72,158,21,12,59,220,35,02,57,027,16,25,44,294,30*77
$GPGSV,3,2,12,29,21,310,28,06,20,058,21,19,18,108,25,09,05,048,14*7B
$GPGSV,3,3,12,48,35,232,33,04,64,319,,17,02,116,,13,02,156,*7E
$GPRMC,005857.000,A,3312.6241,N,09709.5634,W,0.18,220.73,210518,,,D*75
$GPGGA,005858.000,3312.6241,N,09709.5633,W,2,08,1.3,205.7,M,-23.9,M,7.0,0000*4D
```

Fig. 3.4: Sample of GPS NMEA information

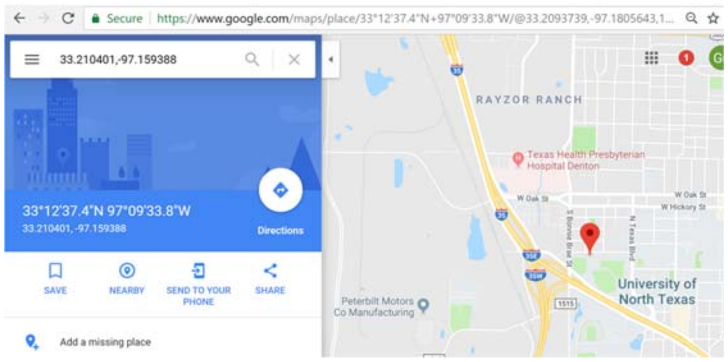

Secure
https://www.google.com/maps/place/33°12'37.4"N+97°09'33.8"W/@33.2093739,-97.1805643,1...
33.210401,-97.159388
33°12'37.4"N 97°09'33.8"W
33.210401,-97.159388
Directions
SAVE
NEARBY
SEND TO YOUR PHONE
SHARE
Add a missing place
RAYZOR RANCH
Texas Health Presbyterian Hospital Denton
W Oak St
W Hickory St
University of North Texas
Peterbilt Motors Co Manufacturing

Fig. 3.5: GPS coordinates on Google map

The location information could be put over the google map. As shown in Fig. 3.5, the location is displayed in google map after input of the converted latitude and longitude. Therefore, by extracting appropriate bit from the string “GPGGA”, the latitude and longitude are acquired. If the two controller Atmega328p can get both its own local GPS coordinate and remote GPS coordinate, it’s possible to calculate the degree of the azimuth.

First, the Em506 uses UART to communicate with micro controller by bit rate: 4800bps which means 4800/8=600 characters per second. The update rate of the GPS module is 1Hz. Also, the GPS module outputs all NMEA data every time by 266 characters, “GPGGA” 78 characters. The code reads all 266 characters by 443mS, find “GPGGA” and finally output extracted latitude and longitude. During the 443mS, the micro controller is totally occupied and can do nothing else. However, it’s in charge of reading magnetometer and drive the turntable motor by 20Hz at least to drive the motor smooth. Hence the fast operation of the magnetometer and motor drive couldn’t cope with GPS reading function.

Second, to exchange GPS info between two antennas before they establish wireless communication, a pair of wireless module XTEND 900 [27] are used in both ends (see Fig. 3.6).

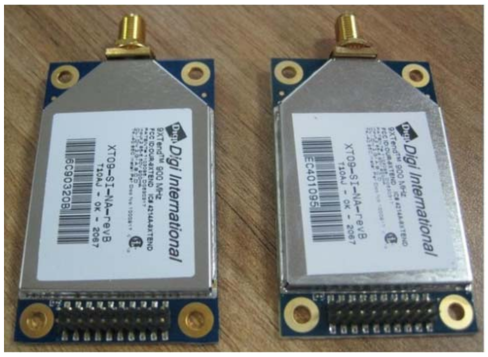


Fig. 3.6: XTEND 900 wireless module [27]

Once GPS is locked, the local GPS coordinate will be sent to each other. However, this is also an UART device which needs certain time to send and receive data. By using 9600bps, the XTEND 900 sends out 19 bytes latitude and longitude data, it takes at least 16mS. However, receiving them takes at least 50mS because a 2mS pause will be added between every two successive characters to help separate them.

As a result, a twin-CPU architecture is taken into consideration. The basic concept showed in Fig. 3.7 is using one Atmage328p as a slave CPU which in charge of slow devices, i.e. GPS and XTEND 900. This CPU does nothing but keeps reading GPS and XTEND900; also checks whether there's anything to send to XTEND900. Once it obtains newly arrived GPS and XTEND900 data, the data is saved into 3 variables: GPS_Lat, GPS_Lon and XTEND_RX. Since this CPU has both GPS coordinates, it performs calculation of the coordinates and saves the result into variable: azimuth_degree. At meantime, another Atmage328p works as master and motor server. It reads both current magnetometer and azimuth_degree from slave CPU as the two inputs of the motor PID controller and finally outputs PWM waveform to motor H-bridge driver to drive turntable motor. The two CPU communicate by the built-in UARTs. Because the CPU Atmega328p has only one hardware UART; therefore, two more soft UARTs are created by library: SoftSerial on the slave CPU. Finally, the slave CPU has three UARTs. Two soft UARTs used by GPS and XTEND900; one hard UART communicates with master CPU. At master CPU side, the only one hardware UART communicates with the slave CPU.

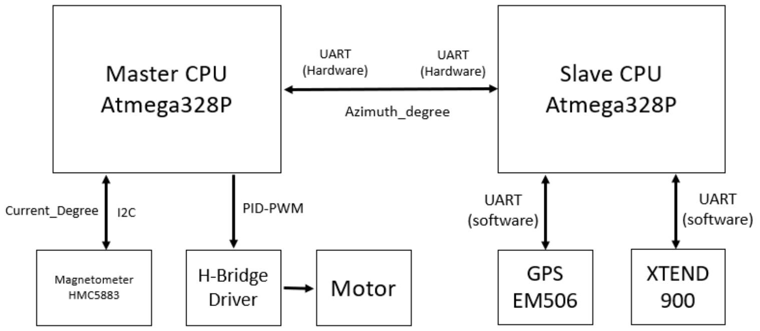


Fig. 3.7: Block diagram of twin-CPU platform

3.3 Circuit and PCB Design

On the design of the circuit, because there're two identical CPUs are adopted simultaneously, both CPUs have identical 16MHz crystals, reset circuit and share one +5V power supply. The previous two-way DC-DC power adapter is used again in this design. The power consumption is listed in Table 3.1. In the table, the maximum power consumption is +12V0.75A when the XTEND is sending data. The maximum current for +5V power supply is 0.88A when XTEND is sending data. The DC-DC buck adapter can output current up to 3A at 5V so 5V power supply is powerful enough to power the whole system. Note the efficiency for the DC-DC buck adapter is calculated by 90%.

Table 3.1: Total power consumption of the V2.0 System

Device	Power consumption	
Atmega328p x 2	+5V 50mA	
GPS EM506	+5V 50mA	
Tenda A6	+5V 300mA	
Nano directional antenna	+12V 300mA	
XTEND 900	+3.3V 50mA (RX)	+3.3V 300mA (TX)
DC motor	+12V 10mA	
Total (DC-DC efficiency:0.9):	+12V0.55A	+12V0.81A

The PCB is designed to be double layer: top-layer and bottom layer as shown in Fig.3.8.

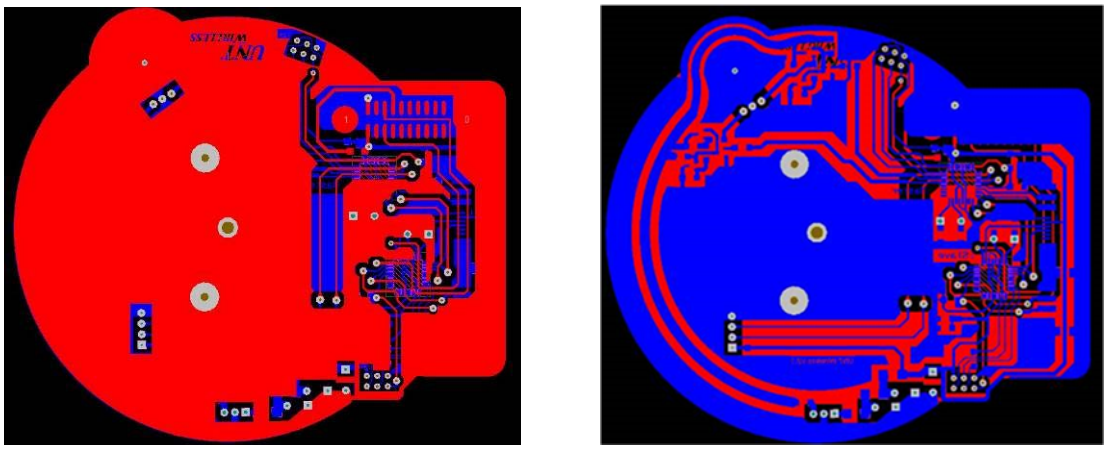


Fig. 3.8: Altium designed PCB

3.4 Software Design

The system block diagram is shown in Fig. 3.9. The slave CPU keeps serving GPS and XTEND. Once both GPS data are available, it performs calculation of the azimuth degree and stores it to be visited by the master CPU. The master CPU serves reading magnetometer and driving motor. It also keeps reading azimuth degree from the slave CPU. Once the master CPU obtains a new azimuth degree, it replaces the old azimuth by the new one and drives the Nano antenna towards the new azimuth target degree. The code design adopted modular structure so it’s convenient to debug each part by comment/uncomment each function in main loop. Once all functions are tested well, enable them all to run entire code.

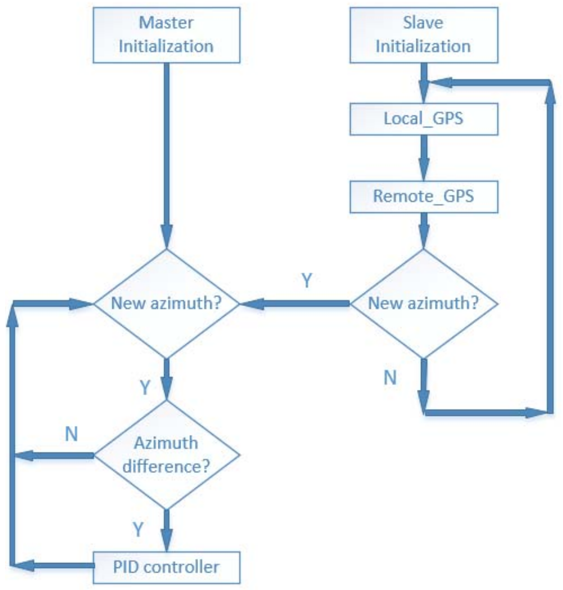


Fig. 3.9: Flow chart of software

3.5 Test and Conclusion

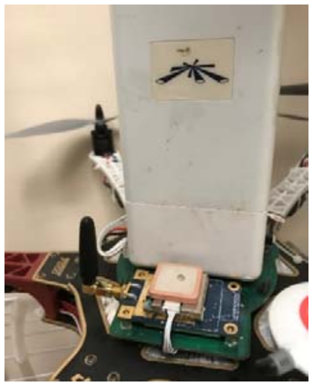


Fig. 3.10: Drone WiFi system twin-CPU platform

After final assembling in Fig. 3.10, a test is performed to verify the overall performance of the turntable. Once powering on, the master CPU drives the antenna towards the north and the GPS module seeks for GPS signal. After 30 seconds, the GPS signal locks with flashing a tiny red LED. Next, the two drones exchange their GPS coordinates. Then, the slave CPU has all variables ready to be read. The master CPU keeps reading the UART to obtain new azimuth degree. Once it is available, the master

CPU immediately drive the antenna to the target degree that the other antenna is located. Test flight verifies the system works well. The only issue is the slow GPS update rate. When a drone is flying, the antennas on both ends rotate once a second despite the drone has moved a lot from one location to another. Therefore, the system is considered suitable to the application which doesn’t move too fast/relative statistic.

CHAPTER 4

ATMEGA2560 PLATFORM

4.1 Exist Issue in Twin-CPU Platform

Previously, a twin-CPU platform consisted by two core controllers: Atmega328P performed final test. Because of the low data rate and long string output, the function of GPS takes quite a long time to extract latitude and longitude. Although a dedicated CPU is used to address this part, the low data rate deteriorates the antenna tracking performance. Also, the GPS and XTEND900 are connected via software simulated serial ports which are not stable enough from time to time. If the read out of the GPS or XTEND900 has a wrong bit, the system will have to drop it and wait for the next data frame which further deteriorates antenna's performance. Therefore, in this design, an advanced CPU with more hardware UARTs will be taken into consideration. After comparison, Atmega2560 is selected because it has 4 hardware UARTs, 86 GPIOs, 16-channel 10-bit ADC, 4 8-bit PWM channels, I2C, SPI and lots of other resources. Hence, this Atmega2560 showed in Fig. 4.1 will be designed to connect all peripherals.

Fig. 4.1: Atmega2560 micro controller [28]

4.2 Solution Concept

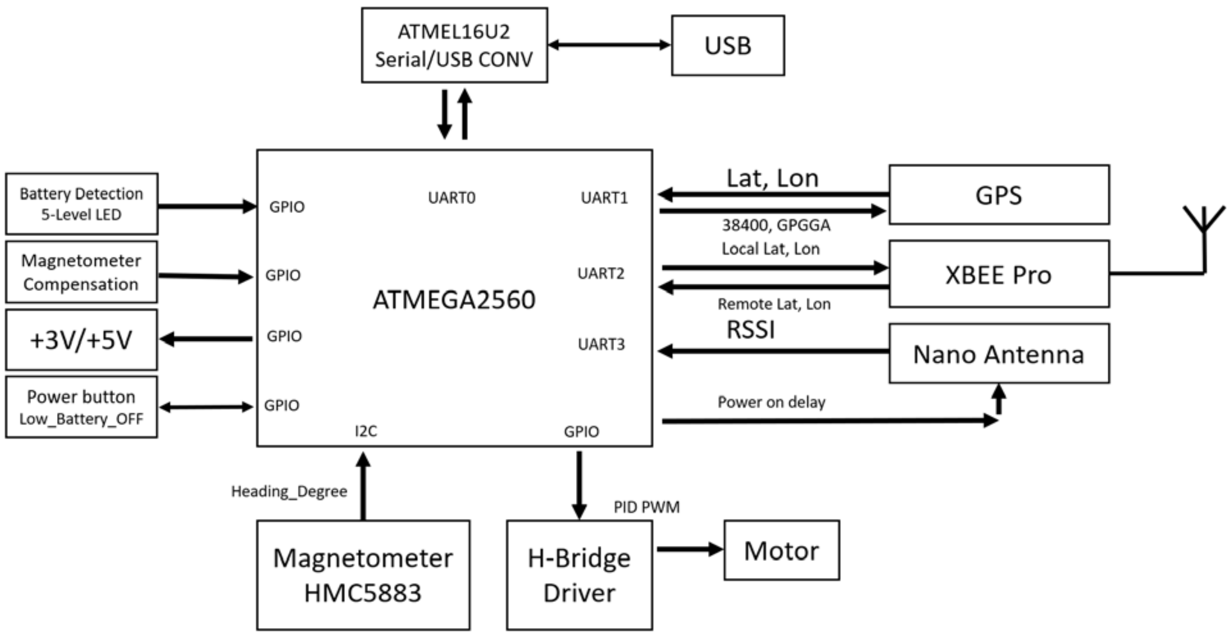


Fig. 4.2: Block diagram of Atmega2560 platform

In this design of Fig. 4.2, a single Atmega2560 8-bit processor is used as core controller. Because it has sufficient hardware resources, it’s suitable to provide various interfaces to serve the project.

4.2.1 Device Upgrade

4.2.1.1 Nano LOCO M5 5GHz Directional Antenna

Nano LOCO M5 is a 5G band directional antenna. It has built-in 13dBi MIMO antenna to ensure the transmission range up to 10 km. It supports bandwidth up to 300Mbps. It also has same dimensions so totally compatible to previous 1st version but better performance and less interference at 5G band.

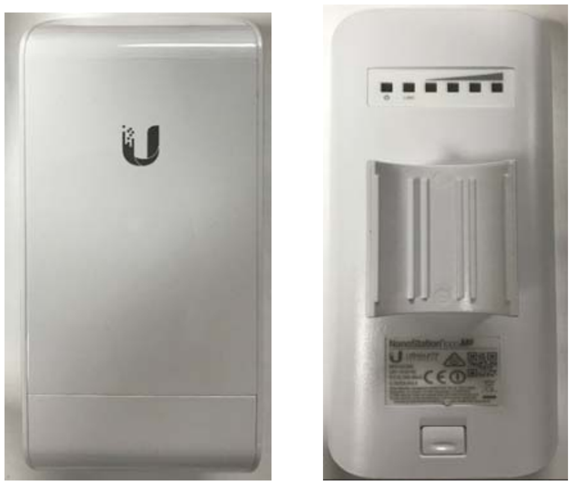


Fig. 4.3: Nano LOCO M5 directional antenna [24]

4.2.1.2 Wireless Router

In the previous design, a TENDA A6 mini wireless router is exploited in the system. The TENDA A6 is single 2.4GHz band wireless router. It is preferred to have both 5GHz and 2.4GHz available in case one band is congested. HUAWEI RS323 [29] is such a device to support both 2.4GHz and 5GHz and considered to be a replacement in the design of this version (see Fig. 4.4). It has an on-panel push bottom to easily convert between 2.4GHz and 5GHz by long press.

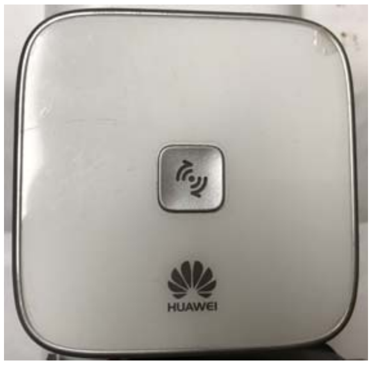


Fig. 4.4: HUAWEI WS323 mini wireless router [29]

The WS232 has three working modes: WiFi router, WiFi repeater, and WiFi client. It is very easy to switch modes by a short press over the push-button (Fig. 4.5).

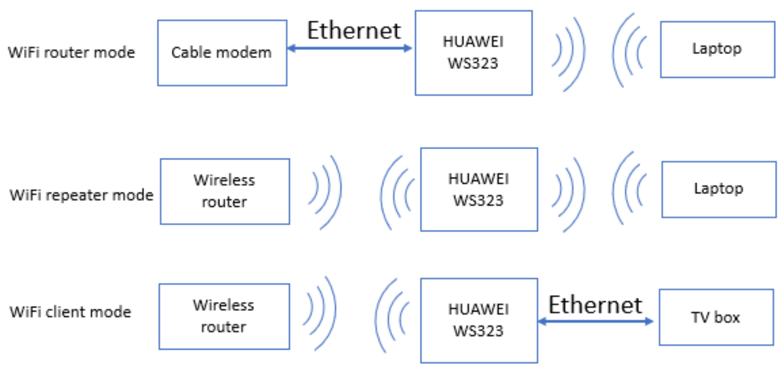


Fig. 4.5: Three modes of HUAWEI WS232

In the following a detail description on the three modes is provided [29]:

4.2.1.2.1 Wireless Repeater Mode

Also called "range extender". In this mode, there's an exist wireless router connected with internet service provider (ISP). However, the users like laptops, smarts phones or tablets are too far away to connect with the wireless router, for example in a big house. In this situation, the HUAWEI WS323 wireless router can work as a "ranger extender" to receive exist WiFi signal and transfer it further to these users by another SSID. To cover more no signal areas, it's applicable to apply more WS323s, each WS323 serves a specific small area. In this mode the WS323 also supports (Dynamic Host Configuration Protocol) DHCP which distributes IP addresses to peripherals. When powering on, the WS323 automatically scans wireless spectrum to avoid crowed channels. Users can also long press the front bottom to select 2.4GHz band or 5GHz band.

4.2.1.2.2 Wireless Router Mode

In this mode, WS323 can turn wired internet router to be wireless router. Most ISPs provide PPPOE service of broad band internet service. In the configuration of WS323, the interface has option to choose PPPOE service by entering username/password so the WS323 can dial over PPPOE service and establish an internet connection. Once WS323 receives available internet service on its WLAN port, its LAN port provides internet connection to devices need services.   For the wireless part, encryption system like WEP, WPA, WPA2 are all supported. Other ISP services like community Ethernet   service, the configuration on WLAN port of the RS323 only needs to be “receive service from uplink device” so that the WLAN port automatically obtains an IP address and transfer services to wireless LAN port.

4.2.1.2.3 WiFi Client Mode

In this mode, the WLAN wireless port works exactly same with the mode 1. WS323 receives internet service, it’ll be transferred to wired WLAN port. The property of the wired port is LAN instead of WLAN. Such application is quite suitable for early network devices (E.g. TV BOX, Microsoft XBOX) without wireless connection. Under this mode, the WS323 works as a wireless adapter to convert wireless internet to be wired internet service.

In above three working modes, the wireless router mode is applied in the current design. Two Nano LOCO M5 directional antennas communicate to each other on two separate drones, each antenna connects with a HUAWEI WS323 wireless router to broadcast WiFi service to each area underneath of the drone.

4.2.1.3 XBEE Module and UART

The processor has four UART ports which are used by GPS, XBEE Pro, Nano antenna and USB converter. In the previous design, the GPS is connected with software simulated serial port. Although the SoftSerial has inaccurate timing, there's a dedicated slave processor working for it. In current design, timing will be critical for each function to properly work one after another. The EM506 GPS supports data rate up to 38400bps which is 8 times faster than original 4800bps, but it can only implement on hardware UART. Also, NMEA info is not all useful for the project. Therefore, information other than "GPGGA" can be disabled via UART1 to decrease time occupation. Finally, the GPS coordinates takes: 78*1/ (38400/8) =16mS (16mS VS 443mS). The XBEE Pro 900HP occupies UART2 to replace the previous XTEND 900 [27] to save size and power consumption.

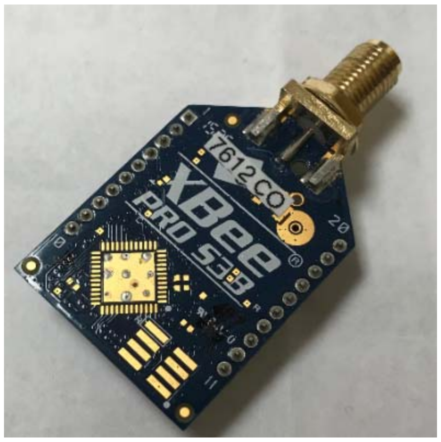


Fig. 4.6: XBEE Pro 900HP [27]

Table 4.1: XBEE Pro 900HP specification

	Indoor	**Outdoor**
10Kbps	Up to 2000ft (610m)	Up to 9miles (15.5km)
200Kbps	Up to 1000ft (305m)	Up to 4miles (6.5km)
Transmit Power	Up to 24 dBm (250mW) software selectable	
Receive Sensitivity	-110 dBm @ 10 Kbps, 101 dBm @ 200 Kbps	

(table continues)

	Indoor	Outdoor
Power Supply:	2.1 – 3.6VDC	
Current: in mA(TX/RX)	215mA / 29mA	
Spread Spectrum:	FHSS (Software selectable channels) at 900MHz	

Table 4.1 shows the transmit and receive performance of the XBEE Pro 900HP. It weights 8g, 70% lighter than the weight of the XTEND900 (23g). The size of the XBEE of 32mmx24mm, 2.8 times smaller than XTEND 900 (60mmx36mm). The Ordinary core controller sends GPS coordinate once a second, totally 19 bytes like: longitude -97.123456 and latitude 33.123456. Even choosing 10kbps, the data rate is fast enough to transmit these bytes. Also, the XBEE module transmit further by sending lower data rate. Therefore, the XBEE Pro 900 HP modules are configured with “10kbps” and “max transmit power”. By configuring data rate 115200bps of XBEE UART showed in Table 4.2, the transmit time reduces from TX: 19mS to 1.3mS; RX: 55mS to 37mS.

Table 4.2: XBEE Pro 900HP performance improvement

	XTEND900	XBEE Pro 900HP	Improvement
UART Data Rate	9600bps	115200bps	12 times higher
Bytes	1200Bps	14400Bps	12 times higher
TX 19 Bytes (Lat, Lon)	16mS	1.3mS	12.3 times faster
RX 19 Bytes (Lat, Lon)	55mS	37mS	33%

UART3 is used to communicate with Nano LOCO M5 directional antenna. This port is directly connected with the serial port of the Nano antenna to read the received signal strength indicator (RSSI). The RSSI will be applied in the function of RSSI assisted alignment module. In case of indoor exhibition without capable GPS signal, the RSSI assisted alignment algorithms can be used to align two directional antennas only by signal strength without GPS assistance.

Nano LOCO M5 directional antenna is running with LINUX2.3 kernel, by performing command in console:

```
$ mca-status | grep signal= |cut -c8-10 >/dev/ttyS0
```

Signal strength can be obtained via Nano's UART ttyS0 so UART3 on the Atmega2560 receives it. However, the command disappears when powers over. In order to perform it every time when powers on, it can be saved in RSSI file. so that the command will be executed automatically at every booting.

```
1)   Create a file: RSSI
          #!/bin/sh
          while [ 1 -lt 2 ]
          do
               mca-status   |   grep   signal=   |cut   -c8-
               10 >/dev/ttyS0
          done
2)   Upload it to Nano LOCO M5 by WINSCP, put it at path:
/etc/init.d/rssi
3)   Modify    the    boot    file:    /etc/rc.local,    add
/etc/init.d/rssi before exit0.
4)   cd /etc/init.d, $sudo update-rc.d rssi default 95
```

4.2.2 A Potential Issue

Because the compact size of the PCB design, quite a lot of components are crowed together. The GPS module is severely interfered by Ethernet controller built-in the Nano LOCO M5 antenna. Since the distance between these two parts is fixed, the timing for powering on each component is critical to handle this issue. The reason is that the GPS takes about 30 seconds to lock satellite signal. During this 30 seconds, it couldn't survive from any strong interference. Otherwise, it will never receive any signal. However, once the signal is locked, a tiny or even medium interference cannot stop its working. Solution is: powers on the Nano LOCO M5 later than GPS for 1 minute, so the Ethernet controller generates noise after GPS is stable. For some special weather like cloudy or weak satellite signal which needs more time to lock signal, 30 more seconds will help.

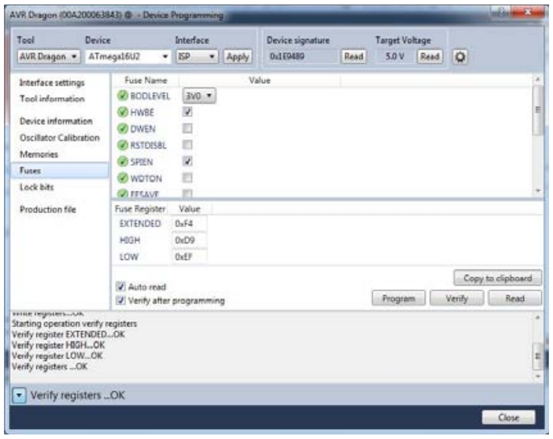
UART0 designed for USB connector doesn't exist in the previous design. The CPU ATmega328p used in twin-CPU platform has only one hardware UART which is connected each other for data communication. In case of upload new sketches, the UART must be disconnected and used to accept new code. Of course, RESET, +5V and Ground are necessary to be wired as well. Therefore, two 6-pin headers are designed close to each CPU. Although such design simplifies the circuit design, it brings lot of trouble. For example, an USB programmer and a bunch of jumpers have to be always prepared for uploading code. Hence, the USB port is considered and integrated in the new design.

Another Atmel 8-bit micro controller: Atmega16U2 will play a key role to bridge the Atmega2560 and host computer. The Atmega16U2 comes with a USB port. By uploading specific USB converter firmware, it can work as a UART-USB converter. However, new brought chips including Atmega16U2 and Atmage2560 cannot be applied into system directly.

4.2.3 Fuse Bit Configuration

4.2.3.1 Atmega16U2

In ATMELSTUDIO 6.0, enter tool-device programming, connect an AVRISP programmer with Atmega16U2 by SPI connection. First, configure fuse bit: High 0xD9, Low 0xEF, Extended: F4, then upload firmware (Fig.4.7).


AVR Dragon (00A200063843) @ - Device Programming

Tool	Device	Interface	Device signature	Target Voltage
AVR Dragon	ATmega16U2	ISP	0x1E9489	5.0 V

Fuse Name	Value
BODLEVEL	3V0
HWBE	✓
DWEN	
RSTDISBL	
SPIEN	✓
WDTON	
EESAVE	

Fuse Register	Value
EXTENDED	0xF4
HIGH	0xD9
LOW	0xEF

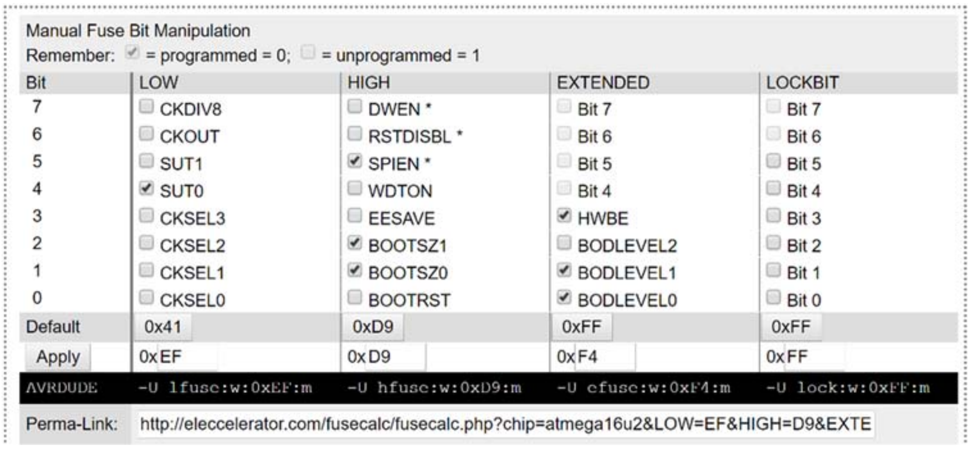
[✓] Auto read
[✓] Verify after programming

```
Starting operation verify registers
Verify register EXTENDED...OK
Verify register HIGH...OK
Verify register LOW...OK
Verify registers ...OK
```

Verify registers ...OK

Fig. 4.7: Fuse bit configuration of Atmega16U2

In Atmega16U2 fuse configuration Fig. 4.8, SUT0 stands for select start-up time, SPIEN stands for upload firmware by SPI bus, BootTSZ0/1 stands for bootsize:256words, 4 pages, application flash section:0x0000-0xEFF, bootloader flash section: 0xF00-0xFFF, end application section: 0xEFF and boot reset address 0xF00, HWBE: hardware boot enable, BODLEVEL0/1: 001, 4.0V keep RESET low when power supply is insufficient < 4.0V.


Manual Fuse Bit Manipulation

Remember: ✓ = programmed = 0; ☐ = unprogrammed = 1

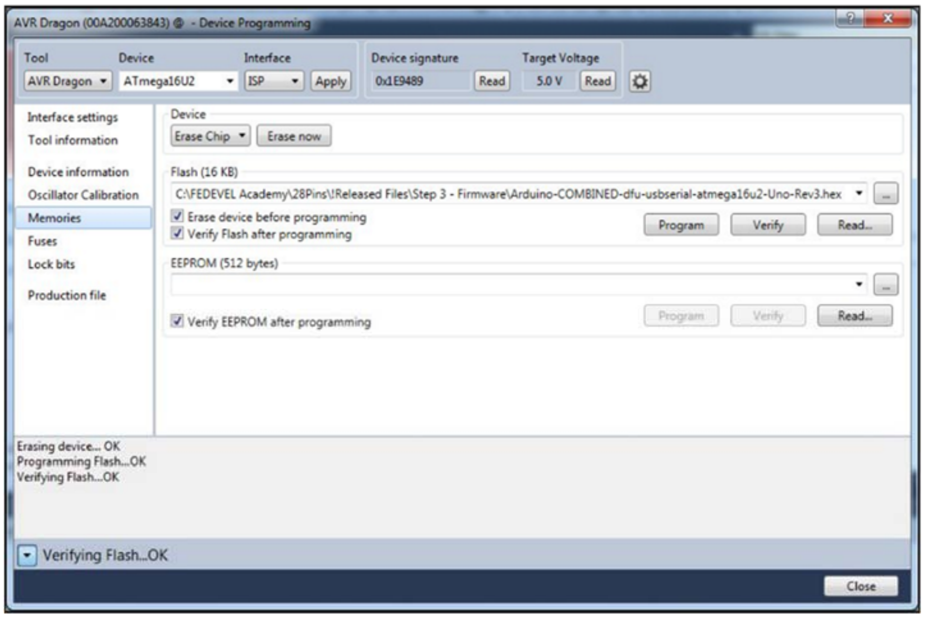
Bit	LOW	HIGH	EXTENDED	LOCKBIT
7	☐ CKDIV8	☐ DWEN *	☐ Bit 7	☐ Bit 7
6	☐ CKOUT	☐ RSTDISBL *	☐ Bit 6	☐ Bit 6
5	☐ SUT1	✓ SPIEN *	☐ Bit 5	☐ Bit 5
4	✓ SUT0	☐ WDTON	☐ Bit 4	☐ Bit 4
3	☐ CKSEL3	☐ EESAVE	✓ HWBE	☐ Bit 3
2	☐ CKSEL2	✓ BOOTSZ1	☐ BODLEVEL2	☐ Bit 2
1	☐ CKSEL1	✓ BOOTSZ0	✓ BODLEVEL1	☐ Bit 1
0	☐ CKSEL0	☐ BOOTRST	✓ BODLEVEL0	☐ Bit 0
Default	0x41	0xD9	0xFF	0xFF
Apply	0x EF	0x D9	0x F4	0x FF
AVRDUDE	-U lfuse:w:0xEF:m	-U hfuse:w:0xD9:m	-U efuse:w:0xF4:m	-U lock:w:0xFF:m

Perma-Link: http://eleccelerator.com/fusecalc/fusecalc.php?chip=atmega16u2&LOW=EF&HIGH=D9&EXTE

Fig. 4.8: Fuse bit detail of Atmega16U2

Firmware:

After fuse configuration, upload a proper firmware “Arudino-COMBINED-dfu-UsbSerial-atmega16u2-Atmega2560-R3.hex” (Fig. 4.9).



AVR Dragon (00A200063843) @ - Device Programming

Tool	Device	Interface	Device signature	Target Voltage
AVR Dragon	ATmega16U2	ISP	0x1E9489	5.0 V

Device: Erase Chip, Erase now

Flash (16 KB): C:\FEDEVEL Academy\28Pins\!Released Files\Step 3 - Firmware\Arduino-COMBINED-dfu-usbserial-atmega16u2-Uno-Rev3.hex

☑ Erase device before programming
☑ Verify Flash after programming

EEPROM (512 bytes)

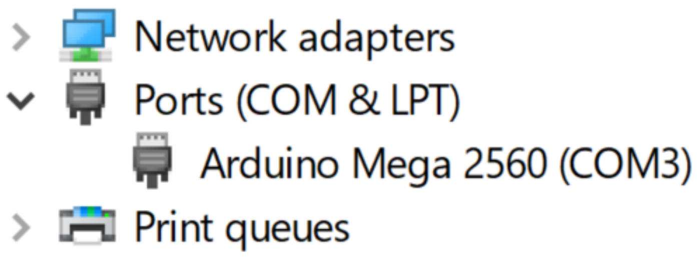
☑ Verify EEPROM after programming

```
Erasing device... OK
Programming Flash...OK
Verifying Flash...OK
```

Verifying Flash...OK

Fig. 4.9: Firmware configuration of Atmega16U2

Now connect the Atmega16U2 to host computer, “Arduino Mega2560 COM3” appears (Fig. 4.6). When uploading code, COM3 will be selected to upload code.

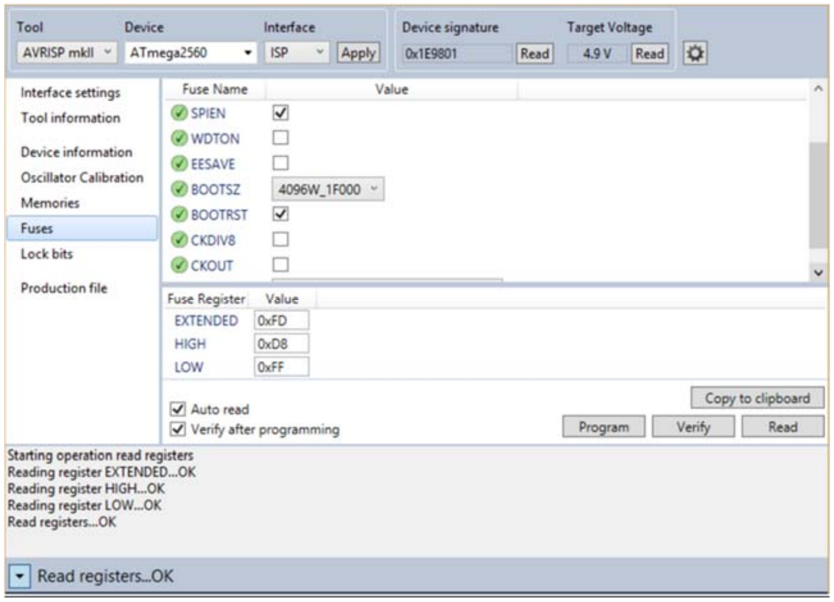


Network adapters
Ports (COM & LPT)
Arduino Mega 2560 (COM3)
Print queues

Fig. 4.10: COM port in system

4.2.3.2 Atmega2560

In ATMELSTUDIO 6.0, enter tool-device programming, connect an AVRISP programmer with Atmega2560 by SPI connection. First, as showed in Fig.4.11, configure fuse bit: High 0xD8, Low 0xFF, Extended: FD, then upload firmware.



Tool: AVRISP mkII | Device: ATmega2560 | Interface: ISP | Apply | Device signature: 0x1E9801 | Read | Target Voltage: 4.9 V | Read

Interface settings
Tool information
Device information
Oscillator Calibration
Memories
Fuses
Lock bits
Production file

Fuse Name	Value
SPIEN	✓
WDTON	☐
EESAVE	☐
BOOTSZ	4096W_1F000
BOOTRST	✓
CKDIV8	☐
CKOUT	☐

Fuse Register	Value
EXTENDED	0xFD
HIGH	0xD8
LOW	0xFF

✓ Auto read
✓ Verify after programming

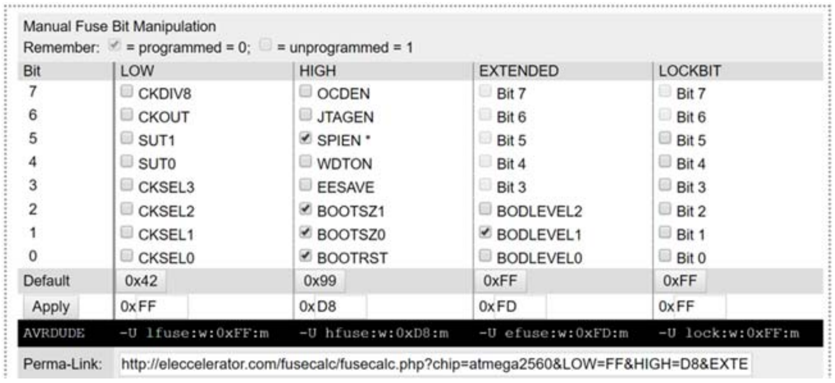
Copy to clipboard | Program | Verify | Read

```
Starting operation read registers
Reading register EXTENDED...OK
Reading register HIGH...OK
Reading register LOW...OK
Read registers...OK
```

Read registers...OK

Fig. 4.11: Fuse bit configuration of Atmega2560

In Atmega2560 fuse configuration (Fig. 4.12), SPIEN stands for upload firmware by SPI bus, BootTSZ0/1 stands for bootsize:4096words, 320pages, application flash section:0x0000-0x1EFFF, bootloader flash section: 0x1F800-0x1FFF, end application section: 0x1EFFF and boot reset address 0x1F000, BOOTRST: once reset the device jumps to bootloader head address and start, BODLEVEL0/1/2: 101, brown-out reset 2.7V.



Manual Fuse Bit Manipulation
Remember: ✓ = programmed = 0; ☐ = unprogrammed = 1

Bit	LOW	HIGH	EXTENDED	LOCKBIT
7	☐ CKDIV8	☐ OCDEN	☐ Bit 7	☐ Bit 7
6	☐ CKOUT	☐ JTAGEN	☐ Bit 6	☐ Bit 6
5	☐ SUT1	✓ SPIEN *	☐ Bit 5	☐ Bit 5
4	☐ SUT0	☐ WDTON	☐ Bit 4	☐ Bit 4
3	☐ CKSEL3	☐ EESAVE	☐ Bit 3	☐ Bit 3
2	☐ CKSEL2	✓ BOOTSZ1	☐ BODLEVEL2	☐ Bit 2
1	☐ CKSEL1	✓ BOOTSZ0	✓ BODLEVEL1	☐ Bit 1
0	☐ CKSEL0	✓ BOOTRST	☐ BODLEVEL0	☐ Bit 0
Default	0x42	0x99	0xFF	0xFF
Apply	0x FF	0x D8	0x FD	0x FF
AVRDUDE	-U lfuse:w:0xFF:m	-U hfuse:w:0xD8:m	-U efuse:w:0xFD:m	-U lock:w:0xFF:m

Perma-Link: http://eleccelerator.com/fusecalc/fusecalc.php?chip=atmega2560&LOW=FF&HIGH=D8&EXTE

Fig. 4.12: Fuse bit detail of Atmega2560

By uploading bootloader of stk500boot_v2_mage2560.hex (Fig. 4.13), the chip can accept all code designed for it. Once powering on, the Atmega2560 runs bootloader then jumps to user program.

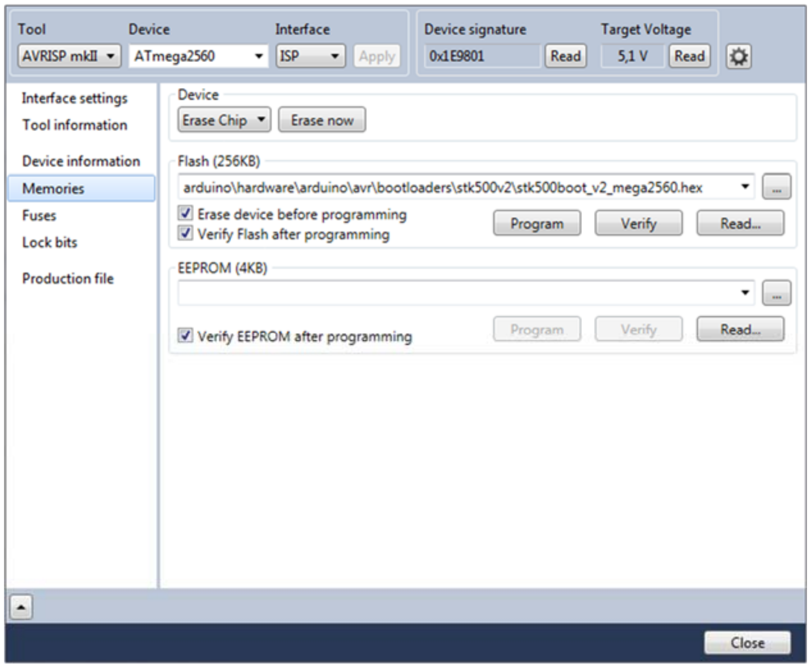

Tool: AVRISP mkII
Device: ATmega2560
Interface: ISP
Device signature: 0x1E9801
Target Voltage: 5,1 V
Device: Erase Chip
Flash (256KB): arduino\hardware\arduino\avr\bootloaders\stk500v2\stk500boot_v2_mega2560.hex
☑ Erase device before programming
☑ Verify Flash after programming
EEPROM (4KB):
☑ Verify EEPROM after programming

Fig. 4.13: Firmware configuration of Atmega2560

4.3 Circuit and PCB Design

4.3.1 Software Power-On/Off Circuit

Different with traditional hardware switch, a push bottom is designed to power on/off the system illustrated in Fig.4.14. Basically, such push bottom in a system is used to change voltage level between High and Low. For example, click it to change temperature setting on the AC controller. The push bottom typically couldn’t hold current over 1A; otherwise, the contactor goes high temperature, oxidizing and blacken. By software assistance, the push bottom is capable to power on/off system.

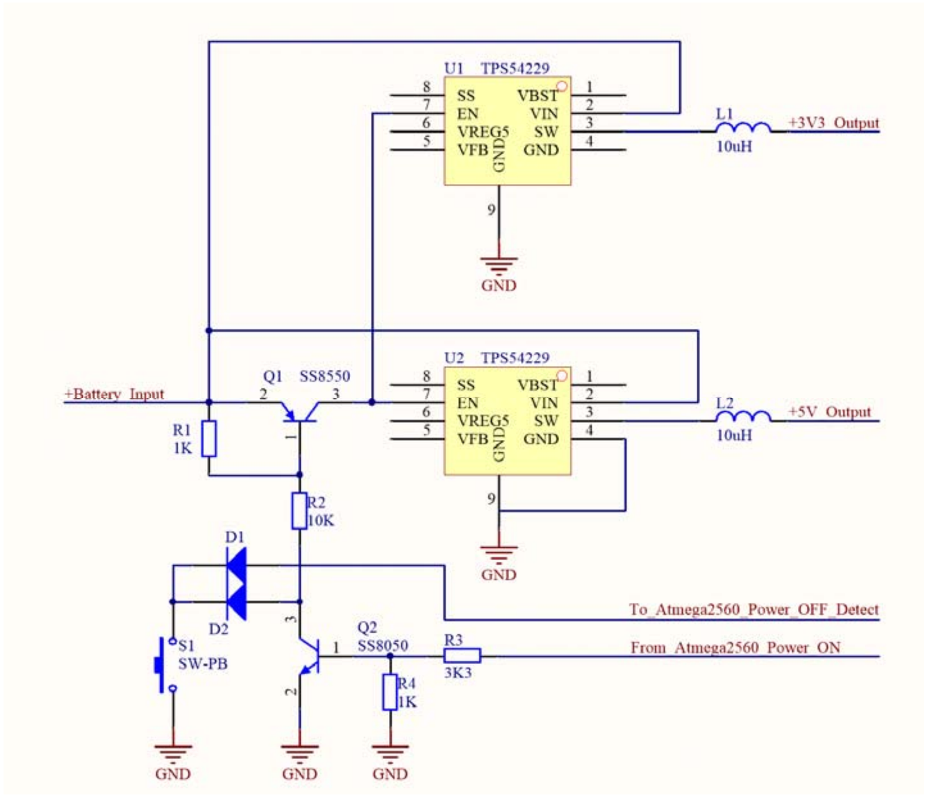


Fig. 4.14: One-key power-on/off circuit

As illustrated in the diagram, when S1 is push down, the input battery voltage 8.4V generates 1V at R1 and turns on Q1, so battery voltage enters EN pins of the chip U1 and U2 TPS54229. U1/2 are usually connect with +Battery_Input from battery and by default the EN pins are Low which means the chips shut down. Once they receive EN high level, both chips work and generate +5V and +3.3V to power the whole system. The +5V powers Atmega2560, Huawei wireless and GPS module. The +3.3V powers magnetometer and XBEE. When +5V reaches Atmega2560, it boots and check battery voltage. As long as the battery voltage is greater than 6.0V, the Atmega2560 sends out a Power_ON signal to trigger Q2. Q2 turns on shorts R2 to ground so that R1 remain 1V to keep Q1 on, now release the S1 to keep system working. The Atmega2560 keeps checking battery voltage once a second and show battery level on LED array. If it finds the voltage is lower than 6.0V, the Power_On signal will be disabled so that the Q1 will turn off; both U1/U2 power off and whole system shuts down. If the system is working correctly while the S1 is pressed manually, the action will be monitored by Atmega2560 and the Power_on signal will be disabled as well to shut down the whole system. In order to avoid mis-pressing of S1, a simple algorithm is used to check the pressing action. If the system checks the push button is pressed over 2 seconds, the Power_On signal will be disabled.

4.3.2 Synchronous DC-DC Buck Converter

The TPS54229 is a DC-DC synchronous buck converter. It enables system designers to complete a design with a cost effective, low component count, low standby current solution. It has wide input voltage range from 4.5V to 18V, output 0.76 to 7.0V 2A, standby current as low as 10uA. Regarding the choice of the battery, concurrently Li-Ion cells are most suitable for portable device because it has high energy density. To the project, antenna weight is critical to the drone's flying time. To supply the Nano antenna directly and supply the other system, a 7.4V Li-Ion battery pack is selected. The selected battery pack has 7.4V 1900mAh capacity which is sufficient for 2 hours' continuous working. Comparing to traditional DC-DC adapter, the synchronous buck converter has 10% more efficiency from 80% to 90%. Fig 4.15 compares two different working principles.

In the traditional DC-DC step-down (buck) circuit, when switch S1 turns on, current flows through L1. The polar across the inductor is positive at left and negative at right. At this moment, the diode D1 doesn't affect circuit. When the switch S1 turns off, the inductor L1 generates a reverse voltage which is negative at left and positive at right.

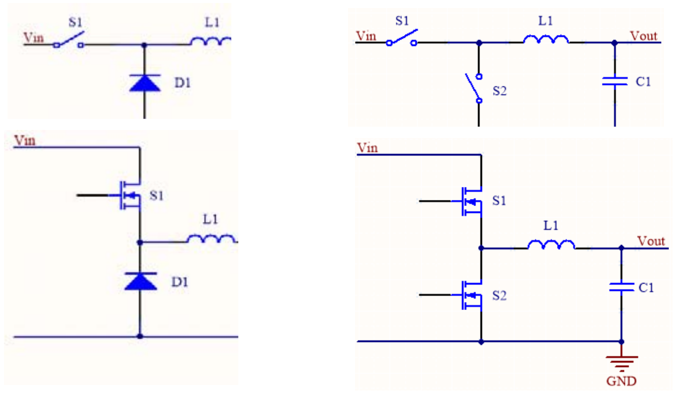


Fig. 4.15: Principle of DC-DC buck and synchronous circuit

This voltage generates current flows through the load and returns to negative polar through the diode D1.The duty rate of the S1 determines how much percent of the voltage will be added to the inductor and finally provided to the load. The capacitor C1 is designed to smooth the ripple of the output voltage and makes the DC waveform as flat as possible. Since the switching frequency of the S1 is up to 1MHz, the output DC has same frequency of the ripple, the filter capacitor C1 should select solid capacitors or high-frequency tantalum capacitors other than traditional aluminum electrolytic capacitors. The traditional aluminum electrolytic capacitors are designed for 50Hz/60Hz application; it can generate extra heat or even explosion if used in high-frequency applications. The diode in the illustration should select high-speed, fast reverse and low forward drop Schottky diode. General purpose diode has hundreds of nS reverse time while Schottky has a few pS reverse time; former has about 0.7V forward drop voltage while latter has only 0.2V ordinary.

In the synchronous buck converter, there're two improvements. First, although the Schottky diode is used in buck converter, it still wastes a little power. In the next illustration, the diode is replaced by a switch S2; however, it will be controlled precisely which means it only turns on at necessary moment with almost zero power loss. The former Schottky diode always turns on as long as the voltage on its anode is higher than cathode by 0.2V; it has 0.2V loss once turning on. The latter only turns on at precise moment and theoretically no loss. In The next illustration, the switch S2 is replaced by a MOSFET. Currently, MOSFET transistors are popular in various application especially in power circuit design because it has extremely high input impedance which means it drains no current from drive source. By technical improvement, original the MOSFET needs over 10V Vgs to be turned on but in recent year this value drops to about 2V. For example, NTJD4001N has very low Drain-to-source ON Resistance (Ron) 1.0Ω at 4.0V, 1.5 Ω at 2.5V.

TPS54229 in the system is a such kind of synchronous buck converter. See application in Fig. 4.16. Therefore, in the application diagram of TPS54229, by properly selecting R1 and R2 pair in Eqn. 4.1, the buck converter can be adjusted to output any voltage in range 0.76-7.0V.

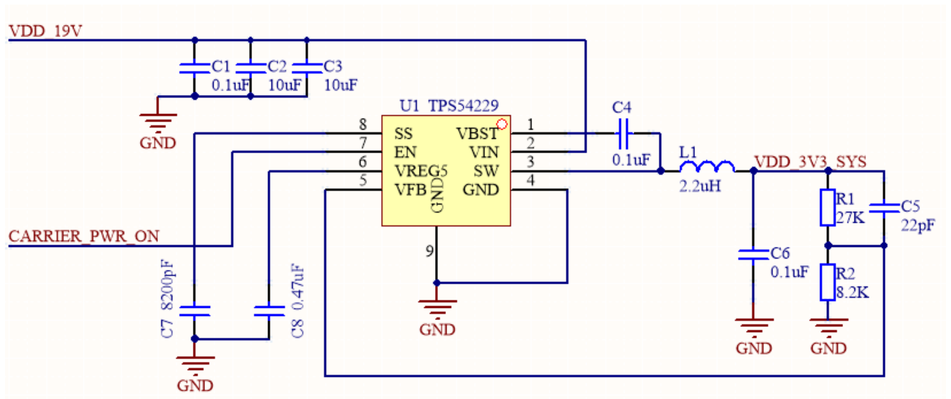


Fig. 4.16: Application of TPS54229

$$V_{out} = 0.765 \times \left(1 + \frac{R1}{R2}\right) \qquad \text{(Eqn. 4.1)}$$

Once the +5V and +3.3V are established, the +5V powers Atmage2560 core controller. It takes 1 second to boot from its own firmware and run pre-uploaded code. In the code, once battery voltage is verified greater than 6.0V, the Atmage2560 outputs a Power-on signal to replace the power button so that the bottom can be released while power remains on. Since the CPU takes 1.5S from accepting power +5V to output Power-on signal, it can also avoid miss touching. Only intended pressing bottom down for over 1.5S can trigger the system.

4.3.3 Battery Events

1: Battery voltage check. In the design of the system, a 7.4V (two 3.7V Li-Ion cells connects in serial) battery pack with easy plug cartridge is equipped on the antenna system. On the Atmage2560, a 10-bit ADC port A0 is set to read sample voltage from the battery pack. The ADC port can sample up to 5V signal, but the battery voltage is up to 8.4V. To solve this issue, a voltage divider circuit is designed to measure the divided voltage on the lower resistor. Since the two resistors have exactly same resistance(10K+10K), the voltage on the lower resistor indicates the voltage across the two resistors is 2 times of the measured voltage. A 10-bit ADC can convert up to 1024 levels which has resolution of 5.0/1024 = 5mV. It’s precise enough to measure battery pack voltage.

On one hand, easy plug design will be convenient to replace drained battery and restore working order. On the other hand, Li-Ion cells couldn’t be drained lower than 6V (3V each cell); otherwise, the cells will be swollen and shorten its life. Hence, the battery management block of the function in the code monitors the voltage of the battery pack, then perform:

A: show battery capacity by lighting 5 LEDs in Table 4.3. The voltage of fully charged battery pack is 8.4V and drains no lower than 6.0V. So, the range 6-8.4V is divided into 5 level, 0.5V each level which means:

Table 4.3: LED array for battery indicator

	LED1	LED2	LED3	LED4	LED5
8.4V-7.9V	🟩	🟩	🟩	🟩	
7.9V-7.4V		🟩	🟩	🟩	
7.4V-6.9V			🟩	🟩	
6.9V-6.4V				🟩	
6.4V-6.0V					🟥
<6.0V	Shut down system: forbid power on; On system, shut off actively				

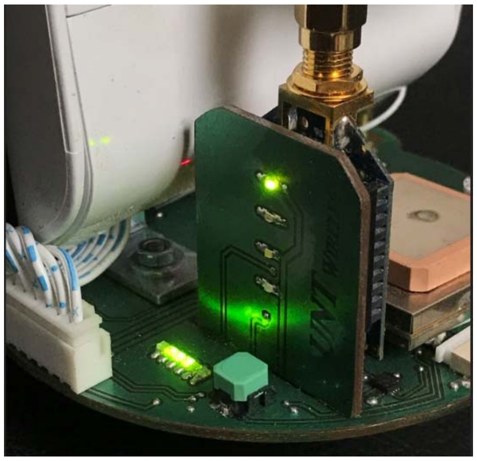


Fig. 4.17: Battery-level indicator

The green LEDs in Fig. 4.17 show corresponding level of the battery capacity. Once the battery voltage is lower than 6.4V, all green LEDs off and Led5 turns on. The LED5 is a red LED with 1Hz oscillator built-in. Once pulling the IO port low (sink current mode, positive pin of the LED connects with +5V power supply), the red LED blinks by itself.

B: If the system is running with voltage drops gradually, the LEDs turn off one by one. Finally, the red blinks and further the Atmega2560 shuts off the Power-On signal; therefore, the whole system powers off to protect battery pack.

4.3.4 Li-Ion Smart Charger

To charge a drained Li-Ion battery, several attentions should be paid. See Fig. 4.18. Initially, the battery voltage is lower than 6.0V, a pre-charge will be applied. In pre-charge stage, a 1/10 charging current (in the system 1/10X900mA=90mA) will be applied to the drained battery to conditioning/activate the battery. When the battery voltage reaches 0.7 x battery voltage, charging shifts to stage 2. In the second stage, the battery pack voltage is quite low, i.e. 6.0V. If charge it by 8.4V (well-charged voltage), the charging current will be huge e.g. 3A-5A due to big voltage difference between battery and charger. The huge current generates excessive heat and dries battery electrolyte so that reduces battery capacity. Therefore, at this stage when charging current is huge, constant-current should be applied. The voltage difference between battery and charger should be kept same dynamically so that battery voltage increases, the voltage of charger increases correspondingly. In this stage, battery voltage increases gradually while charging current is constant. When the battery closes to 8.4V, the charging shifts to the third stage. In this stage, charging current decreases gradually while battery voltage remains 8.4V. The charger provides a 1/10 charging current to compensate the battery capacity, called “trickle charge”.

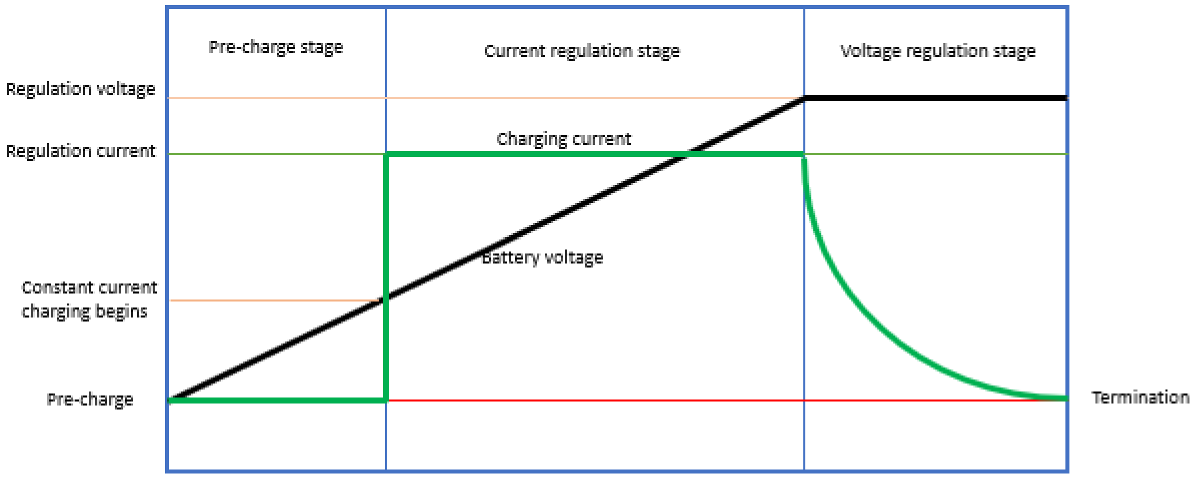


Fig. 4.18: Three-stage of constant-current Li-ion charger

The battery pack comes from NIKON digital camera and the charger has two constant charging designs:

4.3.4.1 Linear Constant-Current Charging Controller BQ2057W

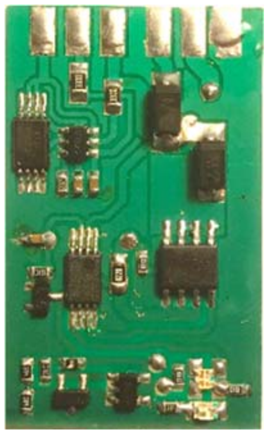


Fig. 4.19: BQ2057W linear 3-stage Li-ion constant-current charger

BQ2057W is a linear 2 Li-ion-cell smart constant-current charger made by Texas Instruments. It detects battery voltage to determine the charging stage and measure the 0.22OHM sense resistor to check charging current to sense the charging status. It adjusts an external MOS-FET transistor to control charging current. See Fig. 4.19. The BQ2057W outputs two signals to drive two LEDs. A red LED turns on when charging. It turns off when charging is done and turns on a green LED. At this point, “trickle charge” begins. Upper left of the PCB is a battery protection circuit. In case the battery is charged over 8.4V or discharged lower than 6.0V, the battery will be disconnected from circuit. Applying a charging voltage can restore the status of the battery. The lower part of the PCB is a battery voltage detection circuit. It uses an operational amplifier to compare battery voltage with reference voltage. If battery voltage is greater than reference voltage, the operational amplifier outputs LOW, nothing happens. When battery voltage is lower than reference voltage, the operational amplifier outputs HIGH to drive a red LED. This LED has built-in 1Hz oscillator, it blinks itself when HIGH received. The following Fig.4.20 shows the charging procedure of BQ2057W by sampling battery voltage and charging current by Arduino UNO R3 development board.

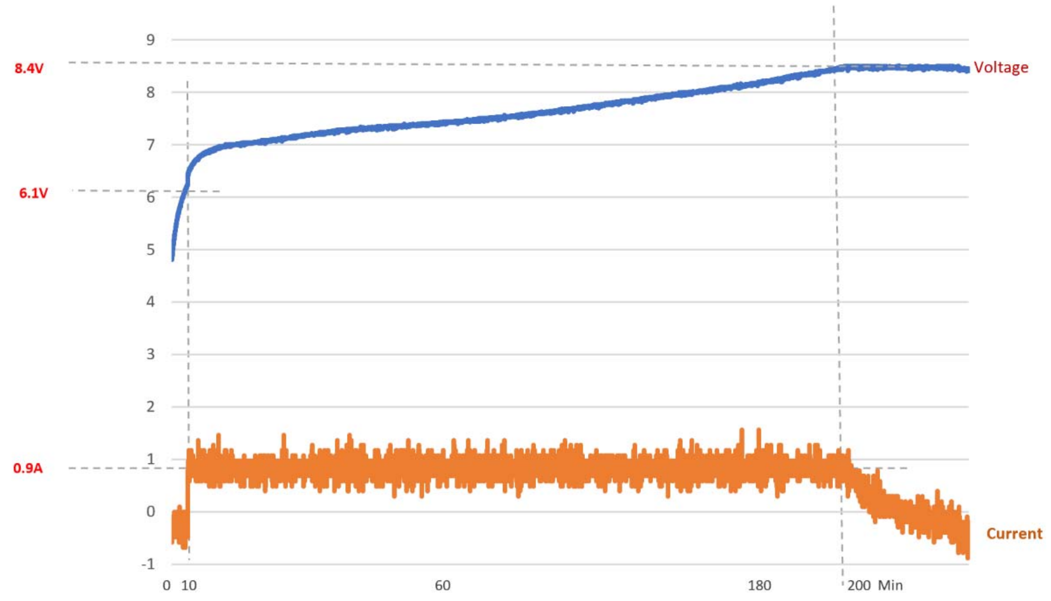


Fig. 4.20: BQ2057W charging monitor

4.3.4.2 PWM Constant-Current Controller: BA24103A

Fig. 4.21 is another version of the smart Li-Ion-cell charging circuit is consisted by BQ24103A made also by Texas instruments. The BQ24103A has built-in PWM MOS-FET transistor to adjust output voltage. Therefore, the chip itself generates heat. The chip has a bottom heat sink pad to be soldered on PCB to help heat sink. The charging theory is identical to the previous BQ2057W. The charging procedure is also sampled for comparison by Arduino UNO R3.

Fig. 4.21: BQ24103A PWM 3-stage Li-ion constant-current charger

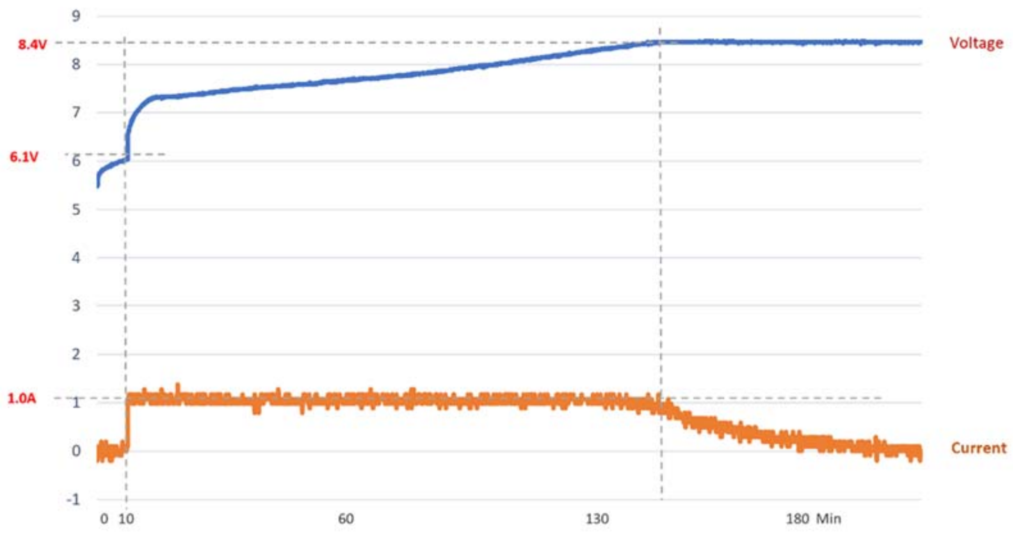


Fig. 4.22: BQ24103A charging monitor

Fig. 4.22 indicates the charging current of BA24103A is smoother comparing to BQ2057W due to PWM controlled more precisely. Further, the BQ24103A generates less heat then BQ2057W. In the design of BW2057W, the external MOS-FET transistor works in linear status, the unwanted voltage is totally transferred to be heat. Therefore, the BQ24103A has higher power efficiency and be used in the antenna battery charger.

4.3.5 Magnetometer and Compensation

Previously a LSM303 magnetometer provides heading degree for the reference of the Nano directional antenna. However, the LSM303 has built-in accelerometer to measure 3-axies to compensate the output heading. Therefore, even the magnetometer is placed on an uneven surface, the readout heading remains precision. When the scene changes to drone, the accelerometer detects the drone's high-frequency vibration and compensates the heading output by mistake. The wrong heading makes the antenna shakes a lot during flight. Although the accelerometer can be disabled in code, another magnetometer HMC5883 is taken into consideration which has no built-in accelerometer. Another reason for choosing it is that the HMC5883 has smaller size and less ball pins which is easier for surface mounting. Test result agrees with expectation, the Nano antenna tracks smoothly and much better than previously.

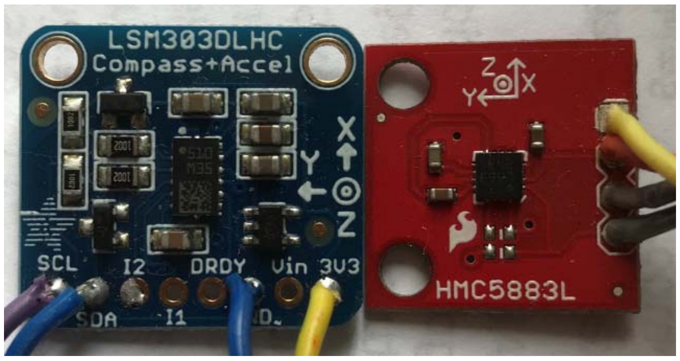


Fig. 4.23: Magnetometer LSM303 [35] and HMC5883 [34]

The original HMC5883 comes with a breakout board with size 20mmX20mm. Due to compact design, the breakout cannot be directly installed on the PCB. The chip of HMC5883 is integrated into the design and finally mounted on the surface of the PCB. The HMC5883 has 16-LLC surface mount technology (SMT) package so it's quite small in size: 2mmX2mm. To mount it onto the PCB, a tiny layer of solder paste will be painted covering the 16 pins. Be aware of the high-temperature solder paste should be diluted 5 times by toluol; otherwise, the balls will slide all around before melting precisely on each pad of the chip. Also, the balls will melt each other, or several balls will melt together on one same pad. 16X0.3mm pin balls are carefully placed on each pad of the chip if without assistance of a specific stencil. Next, heat the balls by hot air at 250-degree centigrade. Because the balls melt at about 160-180 degree centigrade, the nozzle of the hot air gun should be placed about 5CM away on the top of the chip. All balls melt after heating about 5 seconds and bond with each pad. Next, apply same high-temperature solder paste on the footprint of the chip on the PCB, flip the chip and precisely place it on the footprint. Be very careful of the position of No.1 pin. Hot air gun 250-degree centigrade heats 5 seconds, balls melt. The chip can be observed sink a little bit. Push the chip a tiny bit by a tweezer to see whether it can return to original place. If so, soldering finished, see Fig. 4.24.

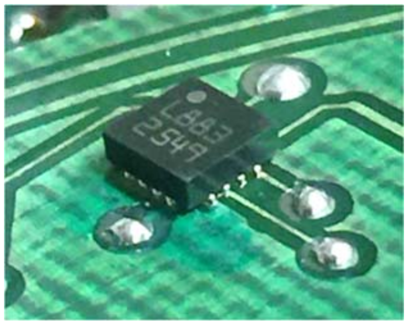
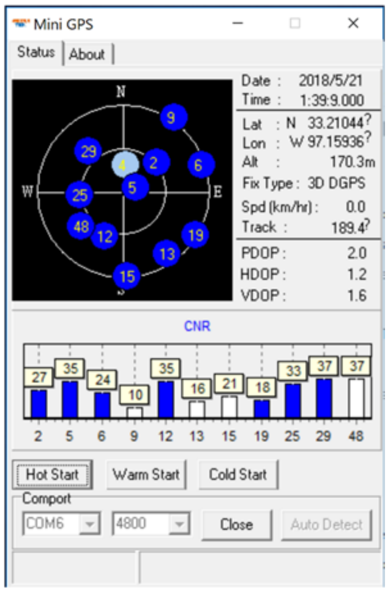


Fig. 4.24: HMC5883 and soldering

Another scene needs to be emphasized is that the manual adjustment of the magnetometer when in exhibition. This project receives lots of attentions so is invited to attend quite a few exhibitions. However, indoor exhibition usually has no strong enough GPS signal. Even with the assistance of the RSSI alignment, the Nano antennas couldn't align precisely due to reasons like magnetic deviation or indoor signal reflection/refraction. Therefore, there should be a simple and direct way to fix this issue temporarily. Here, a potentiometer is integrated into the design. It's connected with analog input A1 port to read the divided voltage of the potentiometer. A 10-bit ADC of the Atmage2560 converts it into integer value to be added to the heading target variable. The mid-point of the potentiometer is set to be zero compensation, clockwise positive and counter clockwise negative compensation to the heading degree so that the antennas azimuth can be tuned actively during exhibitions. This potentiometer can also be used to manually adjust any of other variables according to requirements.

4.3.6 GPS and Nano Delay Power-On

The GPS receiver works at frequency of 1.575GHz; the signal comes from 26 satellites on earth orbit as high as 26000KM. The signal is so tiny which requires high sensitive low noise amplifier (LNA) to amplify the signal. The receiving sensitivity of the EM506 GPS module is as high as -163dbm but is also easy to be interfered. A separate test confirms the time for cold start the GPS, it takes 30 seconds to fix the satellite signal. However, the GPS takes even 5 minutes it can hardly fix the signal when installed onboard. By analyzing the GPS signal from MINI GPS software Fig. 4.25, when moving the GPS towards Nano antenna, the signal drops.



Date : 2018/5/21
Time : 1:39:9.000
Lat : N 33.21044?
Lon : W 97.15936?
Alt : 170.3m
Fix Type : 3D DGPS
Spd (km/hr) : 0.0
Track : 189.4?
PDOP : 2.0
HDOP : 1.2
VDOP : 1.6
Comport COM6 4800

Fig. 4.25: MINI GPS real-time viewer

A close look indicates the Ethernet controller with 24Mhz crystal generates huge interference. The 65th and 66th (1.575GHz/24Mhz=65.6) harmonic submerge the GPS channel and blocks the initialization of the signal fixing. The isolation of the Ethernet controller by copper sheet doesn't work; hence, a delay power-on solution is applied. The observation confirms: if powers on the Nano antenna after the GPS is signal fixed, the Ethernet controller reduced the GPS signal 20% but doesn't block the GPS signal. The remain GPS signal is still good enough to be shared and calculated by system. In the code, the GPS powers on simultaneously with system but the Nano antenna powers on 60s later than the GPS considering bad weather the GPS takes longer than 30s. Therefore, a power control circuit is added to system to control the power supply of the Nano antenna on and off.

4.3.7 Nano Antenna Power Supply

The Nano antenna used in the system is Nano LOCO M5 5GHz directional antenna. The original power adapter comes with the antenna is a 24V1A switching power adapter. Originally when choosing the battery pack, a 24V battery pack or DC-DC boost power supply are considered. However, disassembling and analyzing the Nano antenna shows the 24V battery is not necessary. From the circuit analysis in Fig. 4.26, the input 24V DC is converted to be +5V. This +5V powers DAC and other circuit; also, the +5V is further converted to be +3.3V to power CPU AR9243. Both two DC-DC buck converters are implemented by RT8813. The RT8813 is a wide-input-range DC-DC power supply chip. It supports input voltage ranging from 5.5V-26V, output current is subject to external SMT MOSFET transistors; basically, 3A is capable.

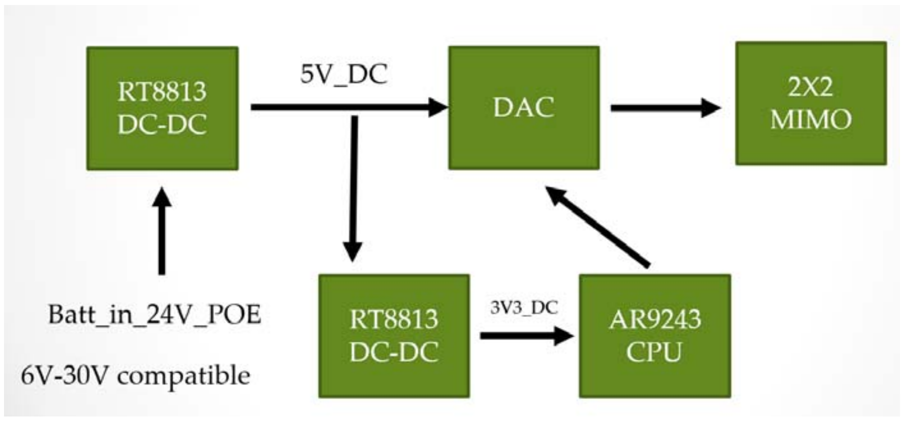


Fig. 4.26: Power supply of Nano LOCO M5

The common sense of the Li-Ion cell is that it cannot be drained lower than 3.0V and cannot be charged over 4.2V so the battery always carries with built-in protection circuit to guarantee cell voltage within range of 3.0-4.2V. To increase output voltage, several cells can be connected in serial. Thus, the battery pack can be cell numbers multiply 4.2V as 8.4V, 12.6, 16.8V… are considered in applications. In this project, the battery should choose 8.4V (6V the lowest) or 12.6V (9V the lowest) to fit RT8813's minimum input requirement 5.5V. Comparing the 8.4V(2S) and 12.6V(3S) battery packs, the 2S is better because 2S is consisted by two cells which has less weight than 3S. Also, the drone’s hovering time is around 30 minutes, more battery capacity and working time contributes nothing to the drone performance. Measurement indicates the whole antenna system consumes 600mA at 8.4V plus DC motor’s current consumption; therefore, a NIKON 2S 1900mA8.4V battery showed in Fig. 4.27 will be suitable. On one hand, the battery can power the antenna directly without any buck/boost circuit; on the other hand, 8.4V is easy to be converted to +5V and +3.3V by buck converters TPS54229.

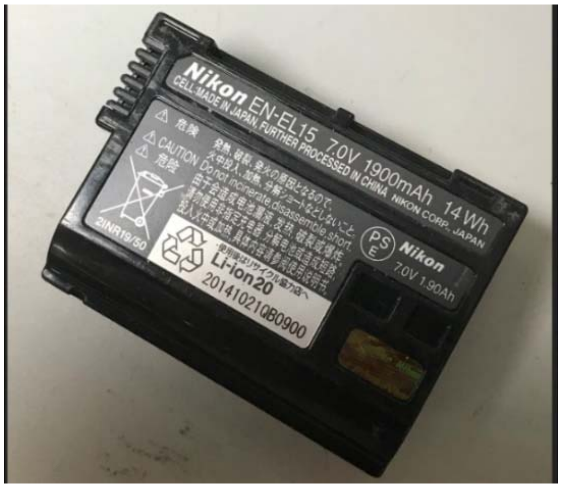

Nikon EN-EL15 7.0V 1900mAh 14Wh
CELL MADE IN JAPAN, FURTHER PROCESSED IN CHINA NIKON CORP. JAPAN
CAUTION Do not incinerate, disassemble, short.
Li-ion20
20141021QB0900
Nikon 7.0V 1.90Ah

Fig. 4.27: Nikon 2S 1900mA Li-ion battery [36]

4.4 Software Design

In the software design, a flow chart in Fig. 4.28 is designed. When powering on, the Atmega2560 drives antenna towards the north and waits for GPS signal for be ready. 30 seconds later, GPS signal locks and exchange coordinates by XBEE modules. Then, the two antennas align to each other. 60 seconds later, the antennas power on and begin communication to each other. Once GPS fail to lock satellite signals, RSSI function launches. The antennas can align to each other by assistance of the RSSI.

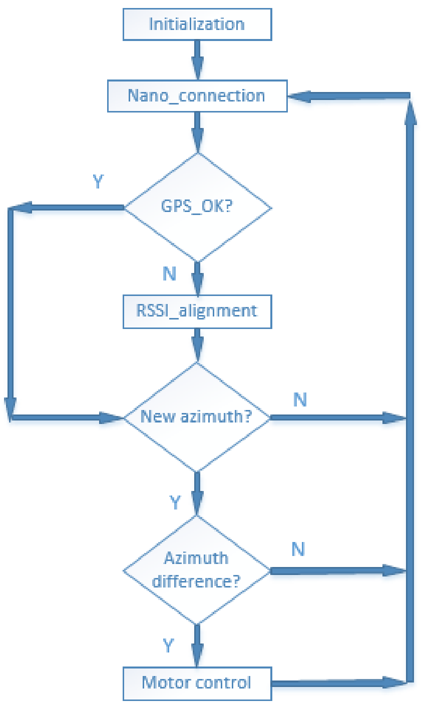


Fig. 4.28: Flow chart of software

4.5 RSSI Assisted Azimuth Algorithm

The two Nano antennas in the system are aligned by assistance of GPS coordinates. In case there's no or very weak GPS signal, the RSSI assisted azimuth algorithm can be adopted to align two antennas. The basic theory is that the two Nano antennas rotates by different speed. Say, one 0.3 round/second; the other one 1 round/second. Theoretically, the two antennas always have chance to be face to face. Therefore, when each antenna is rotating, the received signal strength indicator (RSSI) is also being recorded. In the code, once the RSSI is greater than the previous one, the current azimuth degree and RSSI will be recorded. After rotating for first 60 seconds called “coarse alignment”, the two antennas can roughly find each other. In the next 60 seconds called “fine alignment”, the two antennas perform narrow 60 degree scan to determine the best RSSI azimuth degree. All above operation is performed when GPS signal is unavailable. The system turns to GPS assisted alignment once GPS signal is obtained. The Atmega2560 monitors the GPS signal and determine which alignment method will be performed.

4.6 Drone Integration and Assembling

To fit two different drone platforms: DJI Phatom3 and TAROT650, the antenna assembly has two external designs. Fig. 4.29 shows the design for DJI Phatom3.

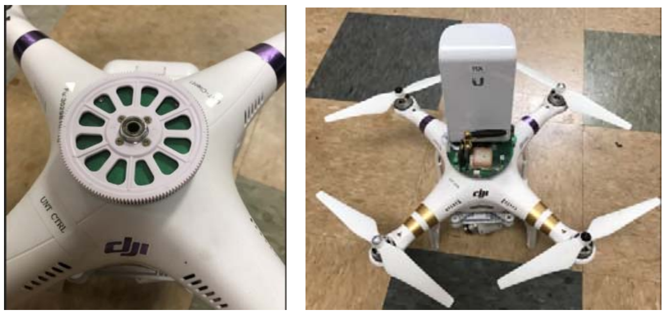


Fig. 4.29: Antenna assembly with DJI Phatom3 [26]

This design is for indoor exhibition, compact in size and convenient to present. The built-in battery of the Phatom3 provides 18 minutes flying time without payload; 8minutes flying time with antenna assembly. The weight of the antenna assembly is 450g.

To extend flying time in the air, a stronger drone: TAROT 650 is also designed to carry the antenna assembly as showed in Fig. 4.30.

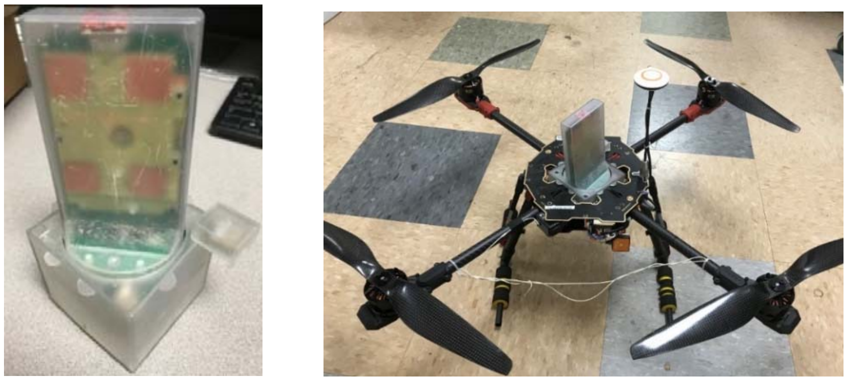


Fig. 4.30: TAROT 650 with antenna assembly [37]

The TAROT 650 is built by all carbon material which has characters of strong in structure and light in weight. The drone equips four 15-inch carbon propellers which has relatively higher efficiency comparing to previous DJI F550 and Phatom3.

The antenna assembly is also modified to better fit for the TAROT 650. The external case, mechanical parts and PCB are re-designed to make the whole system all-in-one. Therefore, the antenna assembly looks more systematic; it’s easy to plug-and-play even for ordinary users.

Test fly verifies that the drone can hover in the air for 28 minutes without payload, 18 minutes with antenna assembly.

4.7 Test and Conclusion

Concurrently, the codes is designed to cope with both indoor and outdoor situations. For the indoor situation, the system performs RSSI assisted alignment algorithm when GPS signal is unavailable. After one minutes, the two antennas can align roughly; two minutes later, they can align precisely. In rare situations the alignment is not so satisfying because the in indoor reflection signal, the potentiometer can be tuned to align the antennas precisely.

For the outdoor situation, the system performs Nano antenna delay power-on algorithm. The GPS module begins to work once the system is power on. The Nano antenna powers on 60 seconds later. During the 60 seconds, the GPS has enough time to lock satellites’ signal, so the alignment is quite precise. Due to the low flash rate of the GPS, the system is not suitable for fast movement tracking while low speed alignment is perfect.

CHAPTER 5

DIRECTIONAL ANTENNA DESIGN FOR HUAWEI WS323 WIRELESS ROUTER

5.1 Exist Issue in the Original Design of the Wireless Router

In the previous design, a TENDA A6 mini wireless router is integrated in the system. TENDA A6 is a single 2.4GHz 802.11 B/G/N 150Mbps wireless router. Because 2.4GHz band is being more and more crowded than before, it's necessary to replace it by a 2.4GHz/5GHz twin-band wireless router. After investigation, a HUAWEI WS323 turns out it's capable. WS323 in Fig. 5.1 supports 802.11 AB/G/N protocols, transmission rate up to 300Mbps. It has built-in multiply-input/multiply-output (MIMO) antenna system on 2.4GHz and 5GHz for more stable wireless and wider coverage.

Fig. 5.1: HUAWEI WS323 mini wireless router

The broadcasting pattern of the HUAWEI RS323 is omni-directional antenna. The omni-antenna has equal transmission power at each direction. However, when flying in the sky, broadcasting to the upper part of the drone is useless. Therefore, this part of the RF energy is wasted. The whole RF energy should be concentrated at certain degree, i.e. 120^o. Hence, the original omni-antenna built-in in the WS323 should be replaced by a direction antenna. Data rate measurement of the HUAWEI 2.4GHz and 5GHz at 5M, 15M, 25M and 35M shows in Fig. 5.2 and Fig. 5.3. In 2.4GHz band, at 5M distance, the data rate is 36Mbps average. By increasing of distance, the data rate drops obviously. At the maximum distance of 35M, the data rate is around 1M. The transmission doesn't show obvious directional characteristic.

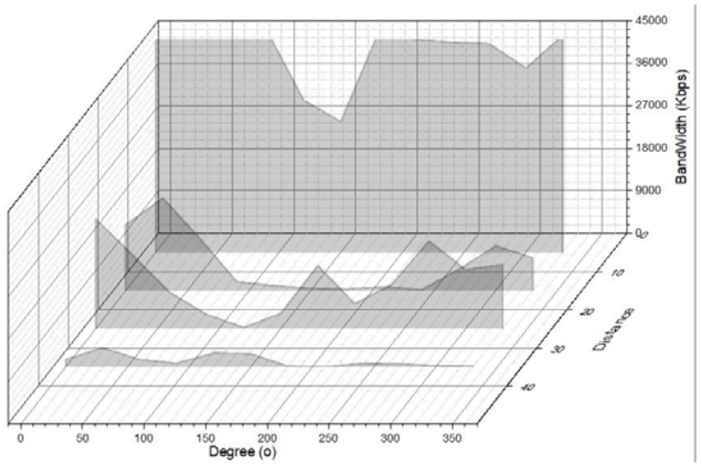


Fig. 5.2: Data rate of HUAWEI 2.4GHz

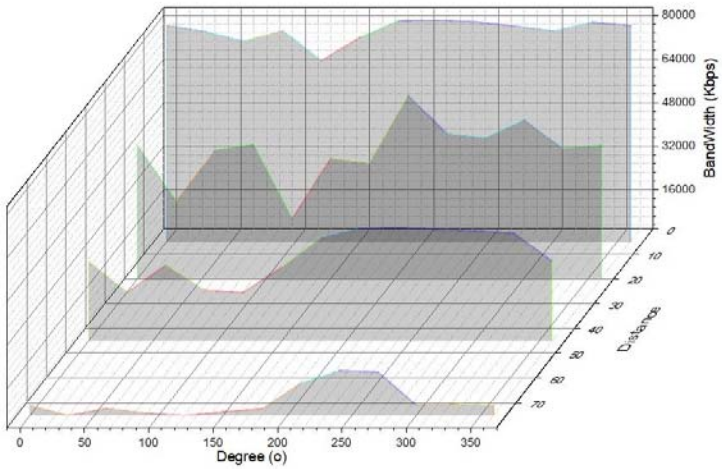


Fig. 5.3: Data rate of HUAWEI 5GHz

5.2 Solution Concept

Fig. 5.4 shows the design of the 2.4G/5G dual-band directional antenna based on the methods in [30]. The antenna is combined by long dipole is designed for lower band at 2.4GHz while the two short dipoles are used for higher band at 5GHz. The antenna is fabricated by ROGERS RT5880 with dielectric constant ϵ=2.2 and thickness t=0.5mm. All three dipoles are excited by a coupling microstrip line at the backside. The RF signal is fed by a coaxial cable linked with HUAWEI WS323 whose original omni-antennas are removed.

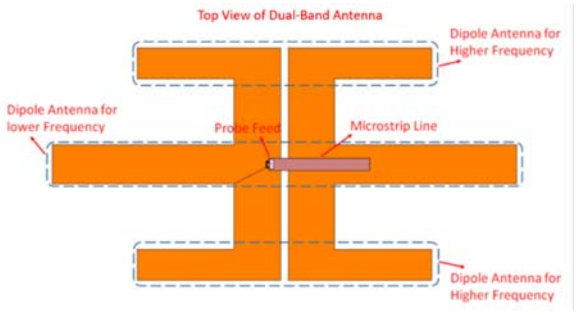

Top View of Dual-Band Antenna
Dipole Antenna for Higher Frequency
Dipole Antenna for lower Frequency
Probe Feed
Microstrip Line
Dipole Antenna for Higher Frequency

Fig. 5.4: Design of directional antenna for HUAWEI WS232 in IE3D

The side view of the antenna is shown in Fig. 5.5.

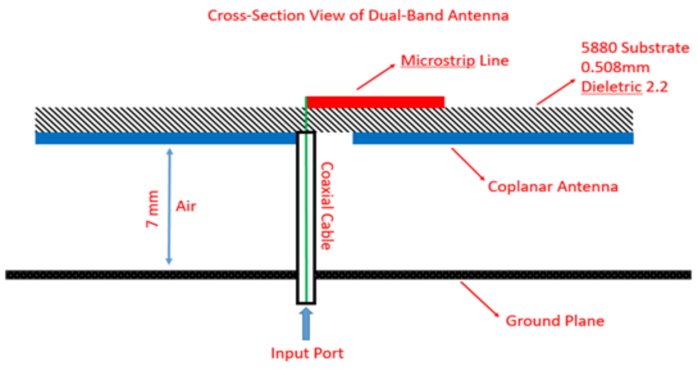


Fig. 5.5: Layer structure of the directional antenna

5.3 Fabrication

The fabricating the directional antenna is illustrated below in Fig. 5.6, itched double-sided PCB of RT5880. The Fig. 5.7 shows the assembling of the parts and obtains the antenna assembly.

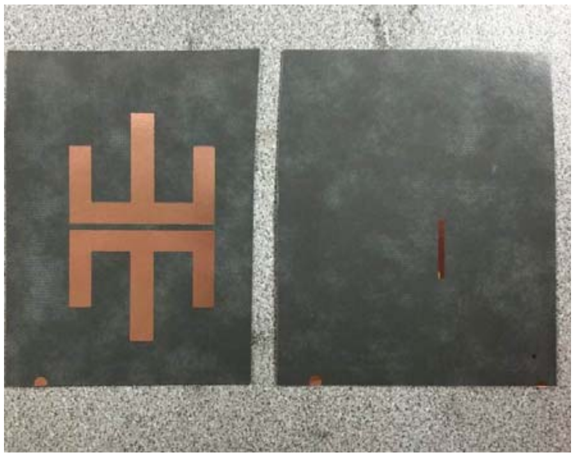


Fig. 5.6: Fabrication of the antenna

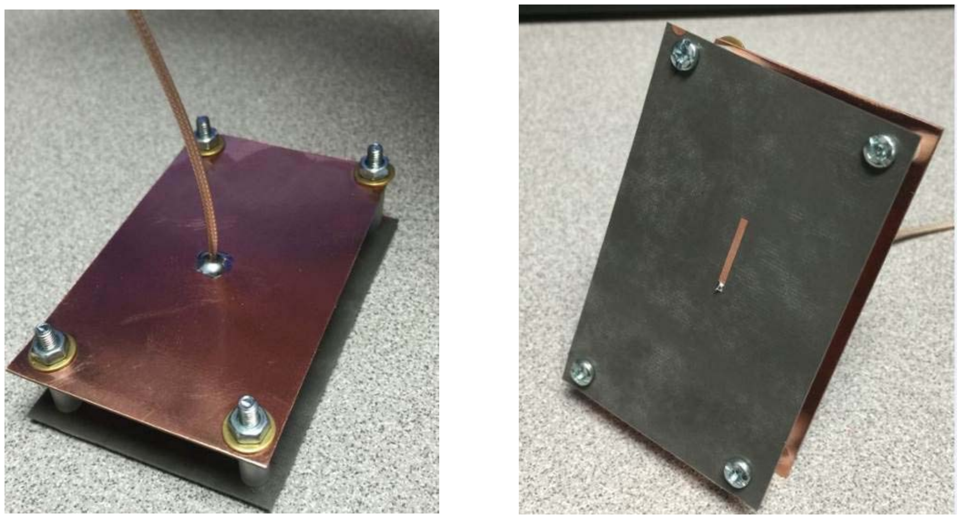


Fig. 5.7: Assembling of the antenna

5.4 IE3D Simulation and Design

The simulation in IE3D is showed in Fig. 5.4. Red curve is the computer simulation under provided parameters. The black curve is the measurement result of the fabricated antenna. The figure shows excellent performance at 2.4GHz and 5GHz. The antenna has 10dB gain at 2.4GHz from frequency 2.39GHz to 2.52GHz, 4.93GHz to 6.13GHz.

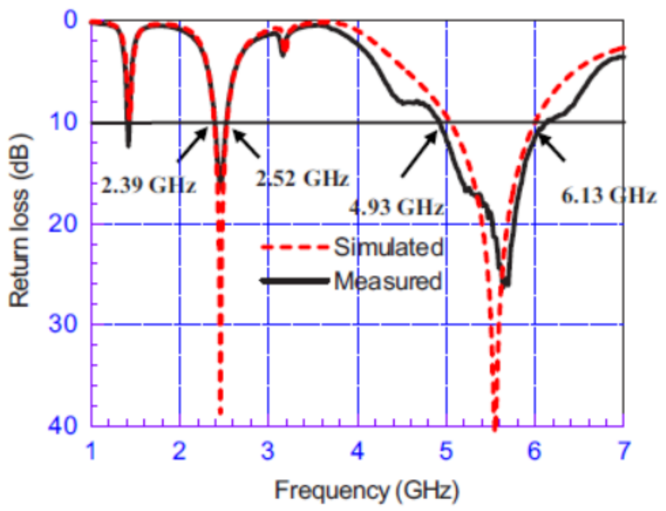


Fig. 5.8: Simulation and measurement at 2.4/5GHz

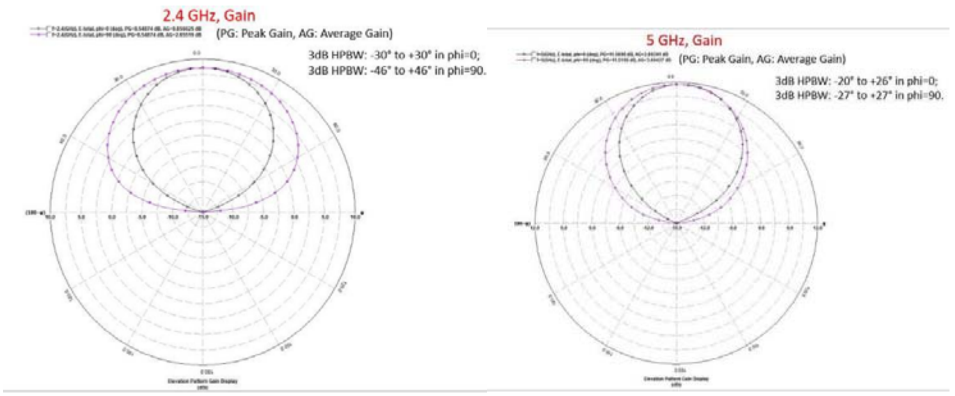


Fig. 5.9: Gain at two bands

5.5 Test and Conclusion

Test and measurement are done with different distance at 2.4GHz: 25M, 50M, 75M and 100M; 5GHz: 25M, 75M, 100M and 200M. The test software is JPERF V2.02. Two DELL XPS9550 laptops with CPU i7-6700HQ, 16GB DDR4 memory, 512GB SSD hard drive and Windows 10 Home 64-bit operation system are used in the test. One laptop running JPERF server end connects directly with HUAWEI WS323 with Ethernet cable. The other laptop connects with HUAWEI by 2.4GHz/5GHz respectively and runs JPERF client with configuration: UDP connection and 30 seconds period. The 2.4GHz band test result is illustrated in Fig. 5.10.

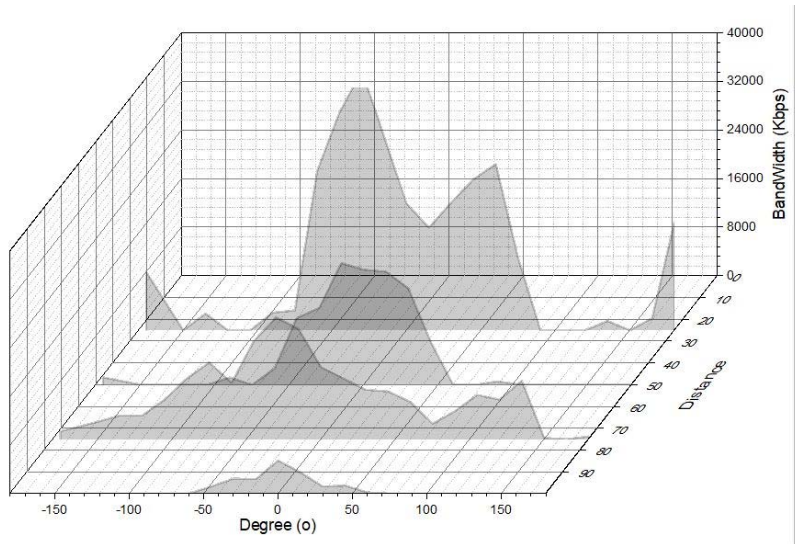


Fig. 5.10: Directional antenna 2.4GHz data rate

The test result clearly shows the directional characteristic at band 2.4GHz which has strong signal and high data rate at range of -50 to +50 degree. The maximum transmission range is up to 100M which still connectable and has about 3Mbps data rate at the very front of the figure 100M.

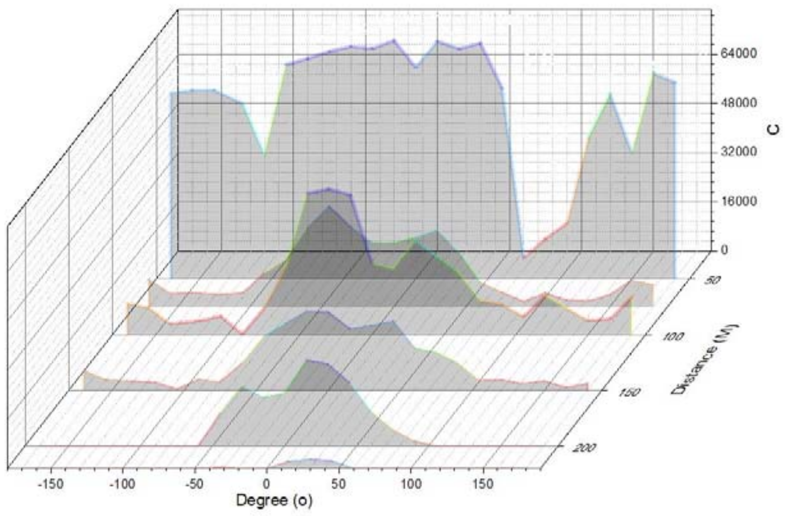


Fig. 5.11: Directional antenna 5GHz data rate

The test result of 5GHz band in Fig. 5.11 also shows the directional characteristic which has strong signal and data rate at range of -50 to +50 degree. The maximum transmission range is up to 220M which still connectable and has about 1Mbps data rate at the very front of the figure 220M.

Table 5.1: Comparison of HUAWEI omni-antenna and directional antenna

	Omni-antenna		**Directional-antenna**	
Band	2.4G	5G	2.4G	5G
Antenna gain (dB)	3	3	10	10
Max range (M)	35	75	100	200
Max Data rate at max range (kbps)	4049	18157	5451	27967
Improvement			186%	180%

From Table 5.1 comparison, it's obvious that the designed directional antenna has significant performance improvement over the original antennas built-in the HUAWEI WS323. The data indicates that the 5GHz band with high-gain antenna can provide premium data service. The 5GHz band on the directional antenna provides up to 220M transmission range which is more than enough to be applied on the drone. In the future, more simulation and modification will be performed to further improve its performance, size and structure.

CHAPTER 6

NVIDIA JETSON TX2 PLATFORM

6.1 NVIDIA Jetson TX2 Introduction

Jetson TX2 is released by NVIDIA in 2017 (see Fig. 6.1). Being an (Artificial Intelligence) AI computing platform, TX2 runs super-fast and has very high power-efficiency [23]. The TX2 platform uses NVIDIA PASCAL architecture is especially suitable for application of robots, drones, intelligent camera and portable medical devices etc. It totally covers all functions of its predecessor TX1; at meantime is ready for more complex and advanced deep learning neural network. It comes with UBUNTU16.04 operation system and NVIDIA JetPack3.2 (Jetson Software Development Kit). The JetPack3.2 is an all-in-one software package with libraries, APIs, documentations and samples to develop projects as soon as possible; also, it contains latest version of TensorRT and cuDNN for machine learning development. The operation system running on Jetson TX2 is LINUX UBUNTU16.04 64-bit and JetPack3.2 R28.2.

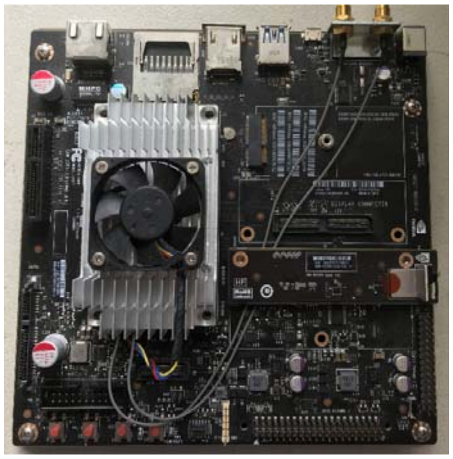


Fig. 6.1: NVIDIA JETSON TX2 CPU and platform [23]

Following is a comparison of TX2 and its predecessor TX1:

Table 6.1: Specification of TX1 and TX2

	Jetson TX2	**Jetson TX1**
GPU	NVIDIA Pascal™, 256 CUDA cores	NVIDIA Maxwell ™, 256 CUDA cores
CPU	HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2	Quad ARM® A57/2 MB L2
Video	4K x 2K 60 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (12-Bit Support)	4K x 2K 30 Hz Encode (HEVC) 4K x 2K 60 Hz Decode (10-Bit Support)
Memory	8 GB 128-bit LPDDR4 59.7 GB/s	4 GB 64-bit LPDDR4 25.6 GB/s
Display	2x DSI, 2x DP 1.2 / HDMI 2.0 / eDP 1.4	2x DSI, 1x eDP 1.4 / DP 1.2 / HDMI
CSI	Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.2 (2.5 Gbps/Lane)	Up to 6 Cameras (2 Lane) CSI2 D-PHY 1.1 (1.5 Gbps/Lane)
PCIE	Gen 2 | 1x4 + 1x1 OR 2x1 + 1x2	Gen 2 | 1x4 + 1x1
Data Storage	32 GB eMMC, SDIO, SATA	16 GB eMMC, SDIO, SATA
Other	CAN, UART, SPI, I2C, I2S, GPIOs	UART, SPI, I2C, I2S, GPIOs
USB	USB 3.0 + USB 2.0	
Connectivity	1 Gigabit Ethernet, 802.11ac WLAN, Bluetooth	
Mechanical	50 mm x 87 mm (400-Pin Compatible Board-to-Board Connector)	

From Table 6.1, the built-in CSI bus can connect hi-definition 1080 cameras; PCIE can connect with extra SSD storage device; USB2.0 and HDMI ports can connect wireless keyboard/mouse/screen to perform on-board debugging; Ethernet port can connect Nano LOCO M5 directional antenna; all these properties enable this development board to work as a next higher-level platform. By using this platform, lots of amazing jobs cannot be implemented in previous Atmega2560 platform will come true including target recognition, target tracing, thermal image detection etc. Also, the TX2 platform supports 5 UARTs which are ready for GPS module, Nano antenna RSSI reading and more applications.

In terms computing NVIDIA Jetson TX2 supports:

- Deep learning: TensorRT, cuDNN, NVIDIA DIGITS procedures
- Computer vision: NVIDIA Vision, OpenCV
- GPU calculation: NVIDIA CUDA, CUDA library
- Multi-media: ISP supporting, camera image, Video CODEC

Also, NVIDIA provides support for (Robotic Operation System) ROS compatibility [31], OpenGL, advanced development tools etc.

6.2 Onboard Test

6.2.1 Device Tree Binary (DTB) and Kernel Compilation

DTB and kernel of the LINUX system are two key points of hardware/software debugging. As long as hardware is added or software updates, DTB and kernel are to be upgraded. In TX2 platform, The DTB modification is done on host computer with the following steps:

1) Convert `/boot/dtb/tegra186-quill-p3310-c03-00-base.dtb` (binary file) to source file *.dts by DTC tool. This method is suitable for modifying an exist working DTB.

   To install dtc for first time:

   ```
   sudo apt-get update
   sudo apt-get install device-tree-compiler
   ```

   Convert DTB to DTC with

```
Sudo dtc -I dtb -O dts c03-base.dts tegre186-quill-p3310-
c03-00-base.dtb
```

This will generate DTS source file c03-base.dts.

Now modify the DTS file by

```
gedit: sudo gedit c03-base.dts
```

and add new device or code. Then convert it back to DTB file by:

```
sudo dtc -I dts -O dtb tegre186-quill-p3310-c03-00-
base.dtb c03-base.dts
```

After the conversion, flash it back to carrier board.

On host computer, force the carrier board into RECOVERY mode: press power button to power the carrier, press and hold the RECOBVERY button, click RESET button, release RECOVERY button. Connecting a USB AB debug cable, "lsusb" to confirm a "Nvidia" device is found. In terminal, enter directory `64_tx2\Linux_for_tegra`, `perform "sudo ./flash.sh -r -k kernel-dtb jetson-tx2 mmcblk0p1"` to upload new DTB. Reboot the carrier to check whether new DTB works.

2) A new DTB can be created by TX2 sources on host computer.

   Download GCC toolchain: https://developer.nvidia.com/embedded/dlc/l4t-gcc-toolchain-64-bit-28-2

Place it at: home/yx/toolchain/install

1) Download L4T 28.2 at: https://developer.nvidia.com/embedded/dlc/l4t-sources-28-2

   Place it at: home/yx/work/tx2 //Unpack each folder
   Create folder "linux_for_tegra" under tx2
   Create two folders under linux_for_tegra: out and modules

2) Create environment variables.

```
export HOME_TOP=/home/yx
export BUILD_TOP=$HOME_TOP/work/tx2
export
CROSS_COMPILE=$HOME_TOP/toolchain/install/bin/aarch64
-unknown-linux-gnu-
export KERNEL_DIR=$BUILD_TOP/kernel/kernel-4.4
export KERNEL_SUB=kernel/kernel-4.4/
export
TEGRA_KERNEL_OUT=$BUILD_TOP/linux_for_tegra/out
export

TEGRA_MODULES_OUT=$BUILD_TOP/linux_for_tegra/modules
```

3) cd $BUILD_TOP

```
sudo make -C $KERNEL_SUB O=$TEGRA_KERNEL_OUT mrproper   //clean
up the previous build
sudo  make  -C  $KERNEL_SUB  O=$TEGRA_KERNEL_OUT  ARCH=arm64
CROSS_COMPILE=$CROSS_COMPILE  tegra18_defconfig  //create  the
configuration
sudo  make  -C  $KERNEL_SUB  O=$TEGRA_KERNEL_OUT  ARCH=arm64
CROSS_COMPILE=$CROSS_COMPILE TEGRA_KERNEL_OUT=$TEGRA_KERNEL_OUT
dtbs  //compile DTS
sudo  make  -C  $KERNEL_SUB  O=$TEGRA_KERNEL_OUT  ARCH=arm6
CROSS_COMPILE=$CROSS_COMPILE TEGRA_KERNEL_OUT=$TEGRA_KERNEL_OUT
zImage  //compile kernel
```

4) Find newly created DTB file at:

`TEGRA_KERNEL_OUT/arch/arm64/boot/dts/tegra186-quill-p3310-1000-c03-00-base.dtb`

Copy this file to:
`yx\Downloads\64_TX2\Linux_for_Tegra\kernel\dtb`

`Find newly created kernel file at:`

`TEGRA_KERNEL_OUT/arch/arm64/boot/Image`

Copy this file to a flash drive

5) Install Jetpack 3.2

https://developer.nvidia.com/embedded/dlc/jetpack-l4t-3_2 (log-in may required)
Run: JetPack-L4T-3.2-linux-x64_b196.run

Follow steps to upload Jetpack 3.2 to TX2 and has 64_tx2/Linux_for_tegra directory ready because it contains flash command to upload DTB.

6) Upload DTB to carrier board:

```
cd yx\Downloads\64_TX2\Linux_for_Tegra\
sudo  ./flash.sh  -r  -k  kernel-dtb  jetson-tx2
mmcblk0p1
```

Reboot the carrier board to check whether new DTB works

7) Upload kernel to carrier board:

Copy the kernel image to carrier /boot/dtb to replace the original image file, reboot to check whether new kernel works

6.2.2 Enable SPIDEV3.0

By default, the SPI bus is disabled by Jetpack R3.2. To enable it, both DTB and kernel should be re-generated. By using above method might be a little complex, a simple way as follow:

Kernel generating:

1) Download source code:

```
git clone
http://github.com/jetsonhacks/buildJetsonTX2Kernel.git
cd buildJetsonTX2Kernel
/getKernelSources.sh
```

2) Configure kernel:

```
cd /usr/src
cd /kernel/kernel-4.4/
cd /arch/arm64/configs/
sudo  gedit  tegra18_def   add  line  “CONFIG_SPI_SPIDEV=m”  under
“CONFIG_SPI_TEGRA114=y”.
```

3) Build the kernel:

Build new .conf file

```
cd /usr/src/kernel/kernel-4.4
```

```
sudo make tegra18_defconfig
```

Build new kernel:

```
cd ~/buildJetsonTX2Kernel
sudo ./makeKernel.sh
```

Ensure the SPIDev.ko Kernel module is copied under /lib/modules

```
sudo cp /usr/src/kernel/kernel-4.4/drivers/spi/spidev.ko
/lib/modules/$(uname -r)/kernel/drivers/
```

Update modular dependency and overwrite the kernel Image:

```
sudo depmod
sudo ./copyImage.sh
```

DTB generating:

1) Convert DTB to DTS on host computer assuming NVIDIA official L4T Jetpack is installed:

```
cd 64_TX2\Linex_for_Tegra\kernel\DTB
sudo dtc -I dtb -O dts -o c03-base.dts tegra186-quill-
p3310-1000-c03-00-base.dtb
```

2) Modify the DTS file and add SPI code on carrier board:

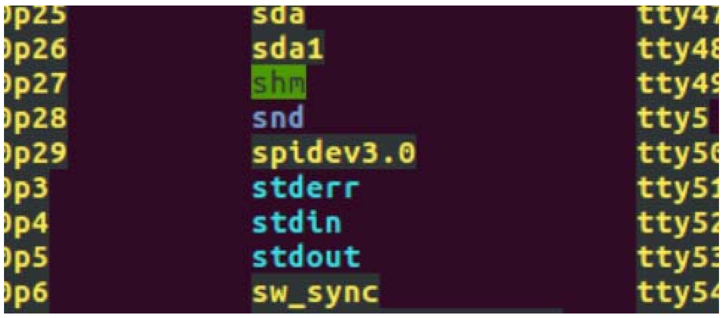
```
sudo gedit c03-base.dts
Find:  spi@3240000{
     compatible = "nvidia,tegra186-spi";
     reg = <0x0 0x3240000 0x0 0x10000>;

     linux,phandle = <0x80>;
     Add:     spi@0 {
          compatible = "spidev";
          reg = <0x0>;
          spi-max-frequency = <0x1312D00>;
          nvidia,enable-hw-based-cs;
          nvidia,cs-setup-clk-count = <0x1e>;
          nvidia,cs-hold-clk-count = <0x1e>;
          nvidia,rx-clk-tap-delay = <0x1f>;
          nvidia,tx-clk-tap-delau = <0x0>;
          };
          };
```

3) Covert the DTS back to DTB and upload the original DTB:

```
sudo dtc -I dts -O dtb -o trgea186-quill-p3310-1000-c03-
00-base.dtb c03-base.dts
```

Under 64_TX2/Linex_for_Tegra, perform "`sudo ./flash.sh -r -k kernel-dtb jetson-tx2 mmcblk0p1`", reboot and verify.

4) After rebooting, the SPI is enabled if a "SPIDEV3.0" is found by running "ls /dev" in terminal as showed in Fig. 6.5.



```
p25          sda          tty47
p26          sda1         tty48
p27          shm          tty49
p28          snd          tty5
p29          spidev3.0    tty50
p3           stderr       tty51
p4           stdin        tty52
p5           stdout       tty53
p6           sw_sync      tty54
```

Fig. 6.2: SPIDEV3.0 in DEV directory

6.3 Customize Compact Carrier Board for Nvidia Jetson TX2

Because the TX2 carrier board has powerful performance and high power-efficiency, it's perfect if it can be adopted in the drone project. However, the size of the carrier board is as big as 170mmx170mm which is too big to be contained by current drones. To meet the project's requirements, the carrier board should possess following features as listed in Fig.6.3:

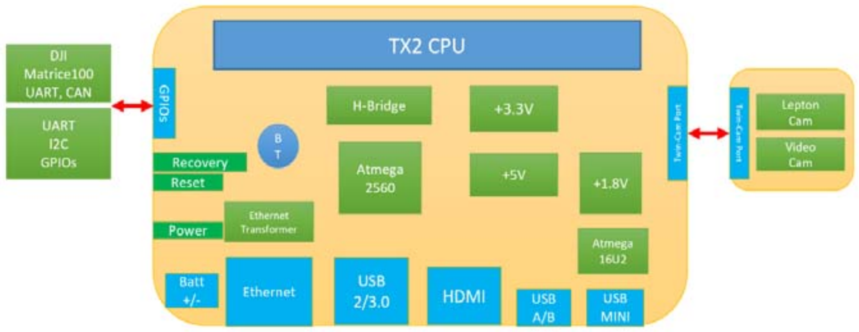


Fig. 6.3: Requirement for the TX2 custom carrier board

1) Ethernet port: to communicate with Nano directional antenna. In case USB port doesn’t work, the Ethernet port can also be used to login by SSH or Putty.

2) USB2/3.0 port: to connect a wireless keyboard/mouse receiver so user can login in desktop to debug software. In case more USB devices are needed, an USB hub can be connected to expend port. This port supports USB2.0 and 3.0 so that in the future a high-definition camera or storage device can hook with.

3) HDMI: to connect an external LCD screen to perform video preview and system debug. For extended application, this port can also be used to capture incoming video.

4) USB micro debug port: only use to upload system image file, DTB and operation system.

5) Mini USB port: to connect host computer to debug of Atmega2560. The port can monitor all variables of the Atmega2560 and upload sketches.

6) GPIOs: to communicate with DJI Matrice 100 drone via UART and CAN bus so the TX2 can command the drone and share certain information including GPS and so on; therefore, previous GPS in the system can be removed. Other GPIOs in this connector includes UART, I2C, CAN and 4 GPIOs can be reserved for future use.

7) Twin-cam port: to link with a daughter twin-cam board which installed a thermal camera: Lepton and a video camera: SONY OV5693. Therefore, the application in the TX2 can perform target tracing and human body detection.

8) Three button of RECOVERY, RESET and POWER: to conveniently turns on the system, reset the system and force the system to accept uploaded new code.

9) Size of the board should be limited within 100mmx100mm to fit current drone.

6.3.1 Introduction of Commercial Carrier Boards

Actually, there are quite a few commercial customized carrier boards listed in Table 6.2 for TX2 are available for comparison.

Table 6.2: Comparison of TX2 carriers on market

Functions	UNT Carrier	CTI Orbitty	DCDZ XCB-Lite	AUVIDEA J100	AUVIDEA J120	Rebotnix RB130
HDMI	√	√	√	√	√	√
eDP (external LVDS LCD)	X	X	√	X	X	X
DSI	X	X	√	X	X	X
LED	√	√	√	√	√	X
UART1/2	√	√	√	√	√	X
Fan connector	√	√	√	√	√	X
Push button(GPIOs)	√	√	√	X	X	X
USB Debug	√	√	√	X	√	X
USB2.0	√	√	X	√	X	X
USB3.0	√	√	√	√	√	√
Ethernet GB	√	√	√	√	√	√
Battery charger	X	X	√	X	X	X
Auto-Power-On	X	X	√	√	√	X
MicroSD	X	√	√	√	√	√
CSI camera	√	X	√	√	√	X
Power, Reset and Recovery buttons	√	√	X	X	√	√
I2C	√	√	X	X	√	√
I2S (Audio)	X	X	X	√	√	√
PCIe	X	√	X	√	√	√
CAN BUS	X	X	X	X	√	√
SATA	X	X	X	X	X	√
Others	Arduino2560	Video input	12GPIOs, Touch screen, 6 cameras	N/A	IMU MPU9250	JTAG
Dimension (in mm)	90x78	87x50	87x63	87x50	110x50	110x74
Weight(g)	53	41	34.6	26	40	N/A
Price: (USD)	80	174	220	235	282	706

However, none of these carriers can totally meet the requirements of this drone project. Therefore, effort is made to design a suitable customized carrier board. It has compact size, sufficient connectors and powerful performance.

6.3.2 Version 1 Carrier Board

In the very first design of the carrier board, it doesn't come with too many features but supports two-way DC-DC buck converter to generate +3.3V and +5V, Ethernet port and USB port. If all these features pass test, more features will be added to the board.

6.3.2.1 Circuit and PCB Design

The carrier is designed in Altium 17.0 with 4-layer structures. Upper layer: signal, second layer: VCC and ground, third layer: ground, lower layer: signal. Fig.6.4 illustrates Altium simulated PCB design and fabricated PCB with all components soldered. The Ethernet and USB ports have quite a few high-speed differential pair wires so it's necessary to tuning each pair to be equal length. For example, there're four pairs of differential signals in Ethernet port. Not only should two wires in each pair be tuned at same length, but also each pair has same length to guarantee all signals arrive target at same time, no delay to each signal. Therefore, lots of serpentine wirings are used to keep all differential pairs same length. The differential pair tolerates ±5 mil difference. E.g., in Fig. 6.5, there're 3 pairs of D1(P, N), D2(P, N) and D3(P, N), the longest is D1: (1000mil, 990mil), D2: (900mil, 890mil) and D3: (800mil,790mil. After length tuning, the final length: D1: (1000mil, 1002mil), D2: (998mil, 1001mil) and D3: (999mil, 1005mil are acceptable by Altium 17.0.

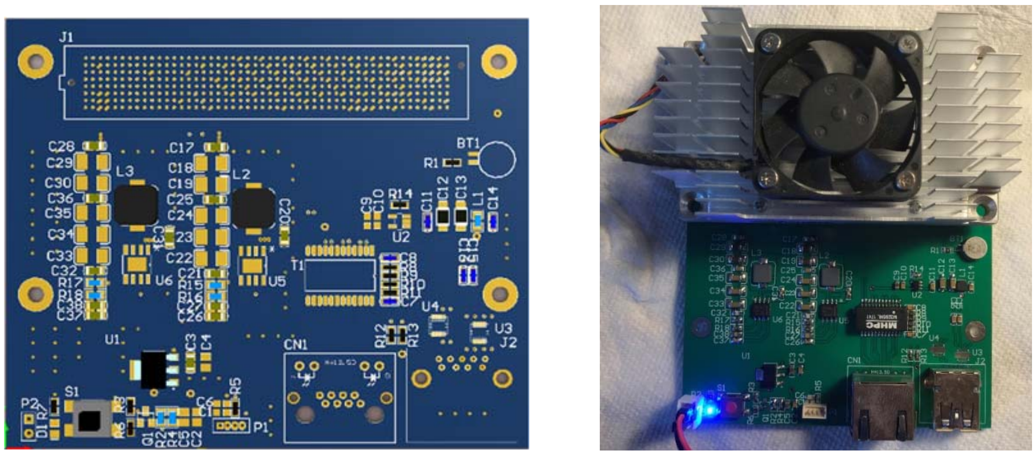


Fig. 6.4: TX2 carrier board V1.0 designed in Altium 17.0

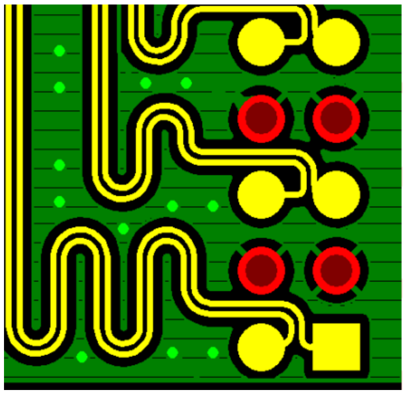


Fig. 6.5: Differential pair and serpentine wiring

6.3.2.2 Test and Result

6.3.2.2.1 USB2/3.0 Enabling

The original TX2 carrier board (Fig. 6.6) has two INA3221 chips to monitor system powers including +3V system, +5V system, +19V input and +3V3 RUN RAIL, 1.8V system, M.2 SSD 3.3V.

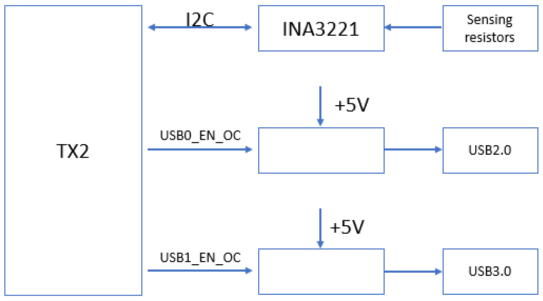


Fig. 6.6: TX2 USB block diagram

The two INA3221s monitor all these power suppliers by high-side sensing resistors to evaluate current in each power supply and send this information to TX2. If TX2 module verifies each current value within working range, it enables USB controller and turns on A17 USB0_EN_OC and A18 USB1_EN_OC to power two USB ports (USB2.0/3.0) onboard. To simplify the design of the custom carrier board, no INA3221 is used in V1.0.

However, without shunt current information, the TX2 module refuses to enable USB controller and no enable signals to turn on USB power. In the V1.0 carrier board, the USB port couldn't obtain any power supply voltage. Even manually place a jumper to short +5V and USB power supply, no USB device can be detected.

To address this issue, the device tree must be modified to fit the current hardware configuration.

STEP:

1) Download GCC toolchain: https://developer.nvidia.com/embedded/dlc/l4t-gcc-toolchain-64-bit-28-1

   Place it at: home/yx/toolchain/install (Unpack each file except TX1)

2) L4T 28.2: https://developer.nvidia.com/embedded/dlc/l4t-sources-28-2

   Place it at: home/yx/work/tx2

   create folder "linux_for_tegra" under tx2

   create two folders under linux_for_tegra: out and modules

3) Edit: `hardware/nvidia/platform/t18x/common/kernel-dts/t18x-common-platforms/tegra186-quill-power-tree-p3310-1000-a00-00.dtsi by $sudo gedit tegra186-quill-power-tree-p3310-1000-a00-00.dtsi`
4)
5) Modify vbus-3-supply from vdd_usb2_5v to battery_reg. The modifications are as follows:

```
pinctrl@3520000 {
     vbus-0-supply = <&vdd_usb0_5v>;
     vbus-1-supply = <&vdd_usb1_5v>;
     - vbus-2-supply = <&vdd_usb2_5v>;
     + vbus-2-supply = <&battery_reg>;
     vbus-3-supply = <&battery_reg>;    // Modification
          here
     vddio-hsic-supply = <&battery_reg>;
     avdd_usb-supply = <&spmic_sd3>;
     vclamp_usb-supply = <&spmic_sd2>;
```

```
        avdd_pll_erefeut-supply = <&spmic_sd2>;
        };
```

Then, re-generate device tree binary(DTB) and flash it to TX2.

6) Create environment variables.

```
export HOME_TOP=/home/yx
export BUILD_TOP=$HOME_TOP/work/tx2
export CROSS_COMPILE=$HOME_TOP/toolchain/install/bin/aarch64-
        unknown-linux-gnu-
export KERNEL_DIR=$BUILD_TOP/kernel/kernel-4.4
export KERNEL_SUB=kernel/kernel-4.4/
export TEGRA_KERNEL_OUT=$BUILD_TOP/linux_for_tegra/out
export TEGRA_MODULES_OUT=$BUILD_TOP/linux_for_tegra/modules
```

7) cd $BUILD_TOP

```
sudo make -C $KERNEL_SUB O=$TEGRA_KERNEL_OUT mrproper
     //clean up the previous build
sudo make -C $KERNEL_SUB O=$TEGRA_KERNEL_OUT ARCH=arm64
     CROSS_COMPILE=$CROSS_COMPILE     tegra18_defconfig
     //create the configuration
sudo make -C $KERNEL_SUB O=$TEGRA_KERNEL_OUT ARCH=arm64
     CROSS_COMPILE=$CROSS_COMPILE
     TEGRA_KERNEL_OUT=$TEGRA_KERNEL_OUT            dtbs
     //compile and generate DTS
sudo make -C $KERNEL_SUB O=$TEGRA_KERNEL_OUT ARCH=arm64
     CROSS_COMPILE=$CROSS_COMPILE
     TEGRA_KERNEL_OUT=$TEGRA_KERNEL_OUT          zImage
     //compile and generate kernel
```

8) Find newly created DTB file at:

```
TEGRA_KERNEL_OUT/arch/arm64/boot/dts/tegra186-quill-
p3310-1000-c03-00-base.dtb
```

9) Install L4T JetPack3.1 on host computer and locate path:

Download/64_TX2/linux_for_Tegra_TX2/kernek/DTB, copy the above dtb file to this path.

10)Set TX2 to RECOVERY mode by:

Press power button to turn on the TX2, immediately press and hold RECOVERY button and press RESET button once and release the RECOVERY button; then hook a micro USB cable. Type “lsusb” in terminal on host computer to confirm “NVIDIA USB Device” appears.

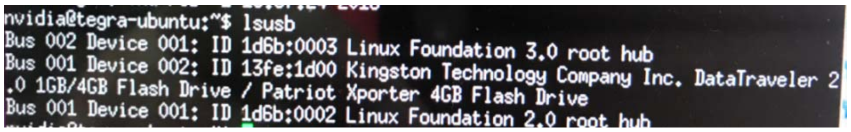
11) On host computer, enter path /Download/64_TX2/linux_for_Tegra\, run in terminal:

```
$sudo ./flash.sh -r -k kernel-dtb jetson-tx2 mmcblk0p1
(which “mmcblk0p1” stands for embedded flash NAND memory)
```

12) Restart TX2 and the USB is ready for use.

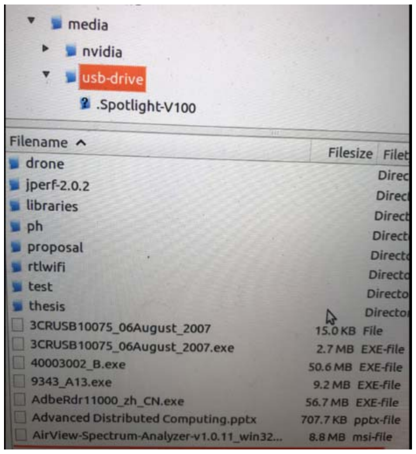
6.3.2.2.2 USB2/3.0 Test

First, a 1GB Kingston flash drive is inserted into the USB port in Fig. 6.7. Typing “lsusb” can show the flash drive is successfully recognized at bus 001 which is a USB2.0 port. Filezilla confirms the USB2.0 is working correctly in reading the flash drive in Fig. 6.8.



```
nvidia@tegra-ubuntu:~$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 13fe:1d00 Kingston Technology Company Inc. DataTraveler 2
.0 1GB/4GB Flash Drive / Patriot Xporter 4GB Flash Drive
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

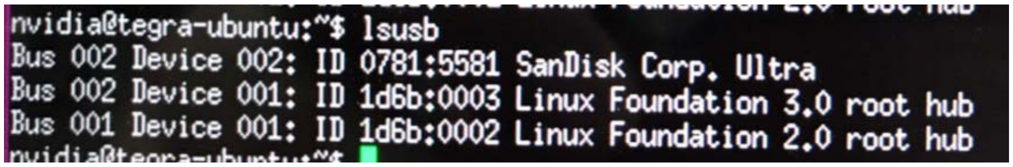
Fig. 6.7: USB2.0 test



```
▾ media
  ▸ nvidia
  ▾ usb-drive
      .Spotlight-V100

Filename ^                                        Filesize  Filet
drone                                                       Direc
jperf-2.0.2                                                 Direc
libraries                                                   Direct
ph                                                          Direct
proposal                                                    Directo
rtlwifi                                                     Directo
test                                                        Directo
thesis                                                      Director
3CRUSB10075_06August_2007                          15.0 KB  File
3CRUSB10075_06August_2007.exe                       2.7 MB  EXE-file
40003002_B.exe                                     50.6 MB  EXE-file
9343_A13.exe                                        9.2 MB  EXE-file
AdbeRdr11000_zh_CN.exe                             56.7 MB  EXE-file
Advanced Distributed Computing.pptx               707.7 KB  pptx-file
AirView-Spectrum-Analyzer-v1.0.11_win32...          8.8 MB  msi-file
```

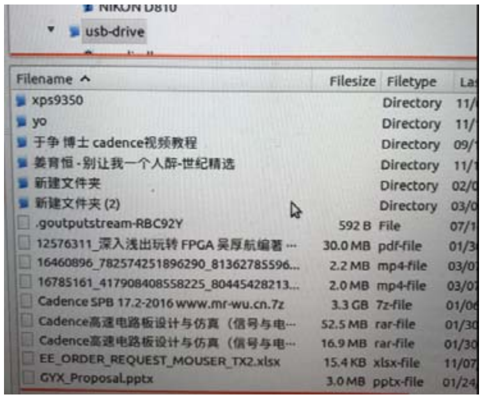
Fig. 6.8: USB2.0 test by Filezilla

Next, a SANDISK 32GB flash drive is connected with the USB port. Successfully, the flash drive can also be recognized at bus002 in Fig. 6.10 in Putty which is a USB3.0 bus. Filezilla confirms it reads and writes the flash drive successfully in Fig. 6.10.



```
nvidia@tegra-ubuntu:~$ lsusb
Bus 002 Device 002: ID 0781:5581 SanDisk Corp. Ultra
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
nvidia@tegra-ubuntu:~$
```

Fig. 6.9: USB3.0 test



Filename	Filesize	Filetype	La
xps9350		Directory	11/
yo		Directory	11/
于争 博士 cadence视频教程		Directory	09/
姜育恒 -别让我一个人醉-世纪精选		Directory	11/
新建文件夹		Directory	02/0
新建文件夹 (2)		Directory	03/0
.goutputstream-RBC92Y	592 B	File	07/1
12576311_深入浅出玩转 FPGA 吴厚航编著 ···	30.0 MB	pdf-file	01/3
16460896_782574251896290_81362785596...	2.2 MB	mp4-file	03/0
16785161_417908408558225_80445428213...	2.0 MB	mp4-file	03/0
Cadence SPB 17.2-2016 www.mr-wu.cn.7z	3.3 GB	7z-file	01/0
Cadence高速电路板设计与仿真（信号与电···	52.5 MB	rar-file	01/30
Cadence高速电路板设计与仿真（信号与电···	16.9 MB	rar-file	01/30
EE_ORDER_REQUEST_MOUSER_TX2.xlsx	15.4 KB	xlsx-file	11/07
GYX_Proposal.pptx	3.0 MB	pptx-file	01/24

Fig. 6.10: USB3.0 test by Filezilla

6.3.3 Version 2 Carrier Board

Based on the carrier board version V1.0, more features will be added to the board. The new features include HDMI, Atmega2560, USB AB debug port, mini USB port for Atmega2560, GPIO extend connector and twin-cam support.

6.3.3.1 Circuit and PCB Design

The schematic diagram is consisted by 8 separate sheets. They're: ABCD socket of TX2, EFGH socket of TX2, Ethernet_Port, USB2/3 and USB micro, Accessories, 3V5V DC-DC power converter, Atmega2560 and mini USB and HDMI_Port. Also, there's a twin-cam schematic for daughter board. The test of the carrier V1.0 corrected several design issues and appended above new features confirmed on TX2 carrier board. PCB layer structure maintains the original 4-layer, but the wiring complexity is much higher due to much more added components on this version. Following Fig6.11 is the both sides of simulated PCB by Altium17.0

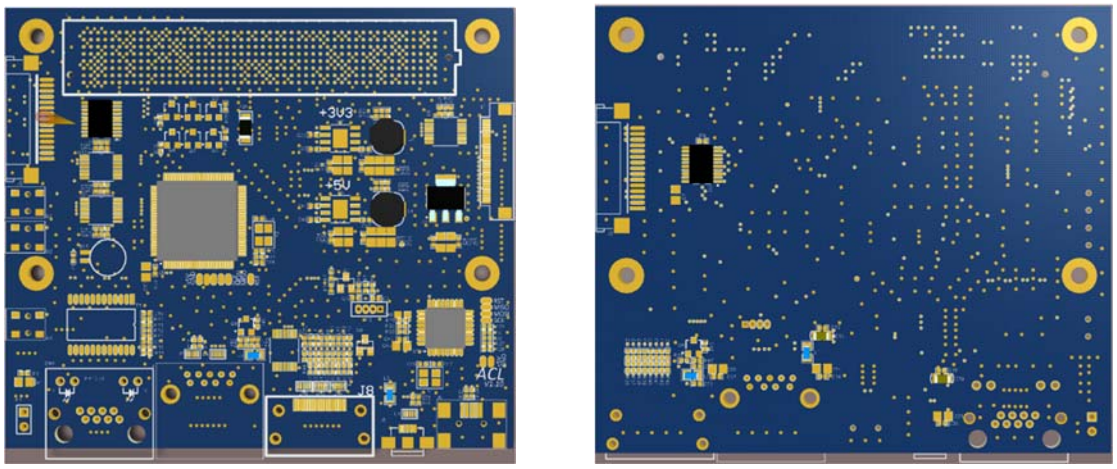


Fig. 6.11: Altium 17.0 designed PCB V2.0

Next, Fig. 6.12 is the fabricated PCB and daughter board. Fig 6.13 is the soldered system; motherboard and daughter board bonded by a 30-pin flexible cable.

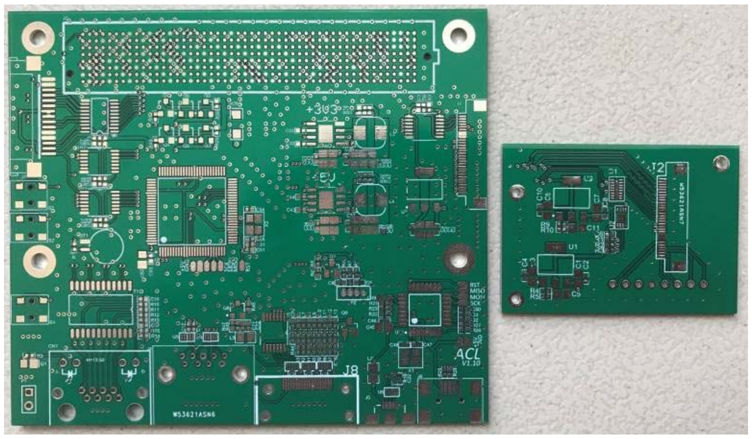


Fig. 6.12: Fabricated PCB V2.0 and twin-camera daughter board

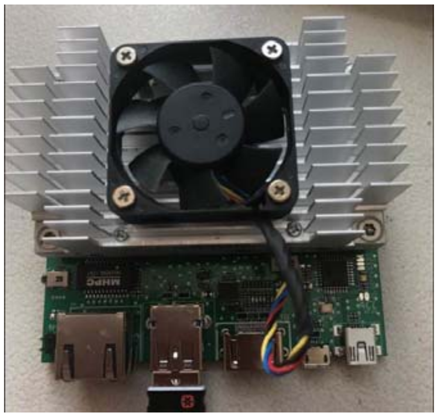
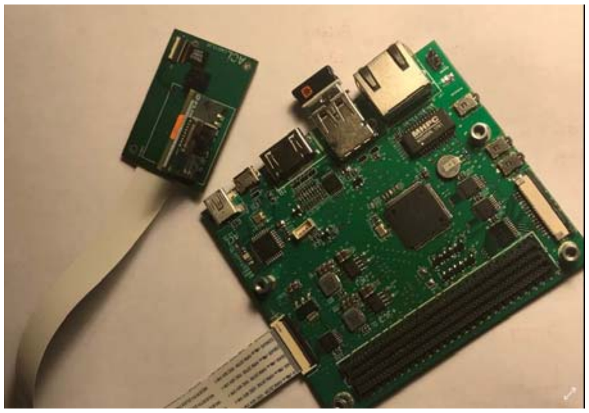



Fig. 6.13: Assembled PCB V2.0

6.3.3.2 I2C Port Expender TCA9539

In the second version of carrier board V2.0, a twin-camera daughter board is designed to connect with the main board. Test indicates quite a few issues due to simplification of the original design. In the original design, two TCA9539s are integrated in the circuit to expend IOs. TX2 module connects TCA9539 with I2C bus expends 16 ports per chip. The ports include quite a few variables work for cameras including: CAM_VDD_1V8_EN, CAM_VDD_1V2_EN, CAM2_RST, VDD_SYS_EN and so on. Because removing of the two chips, these variables are failed to be created. Consequently, the system reports error during booting and the onboard camera SONY OV5693 couldn’t be recognized. To handle this issue, the kernel of the Linux will be modified a lot and compiled again to avoid detection of these variables when booting. All related variables detection is disabled but still pop up issues including VANA error, VIF error, MCLK error, platform data error etc. In Fig. 6.14, the two TCA9539 have to be installed back to the TX2 carrier board to avoid too many complains from TX2 and therefore, everything works fine.

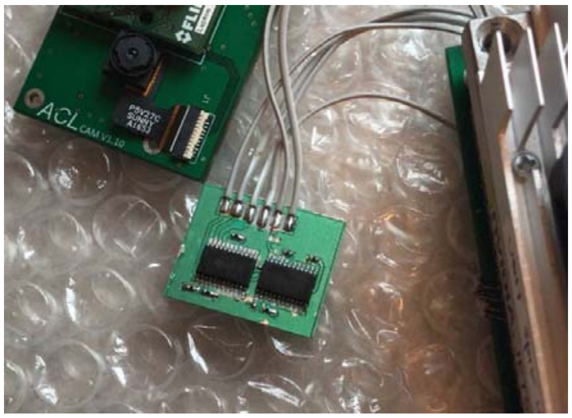


Fig. 6.14: Attached PCB for TCA9539

6.3.3.3 I2C and Magnetometer LSM303

To test the I2C bus, a I2C device is needed to be connected to the bus to verify the design is correct. LSM303 is a typical I2C device so it’s applicable to test the I2C bus. The I2C bus of the TX2 module is 1.8V tolerant while the magnetometer LSM303 has 3.3V IO voltage so a level-shifter chip is needed to convert between 1.8V and 3.3V. In the design a TXS0108 8-channel level-shifter made by TI is integrated to handle the job. The chip converts not only I2C bus but also GPIOs. Soldering the SDA and SCL wires of the LSM303 to pads of the TX2 carrier just finishes hardware connection in Fig. 6.15.

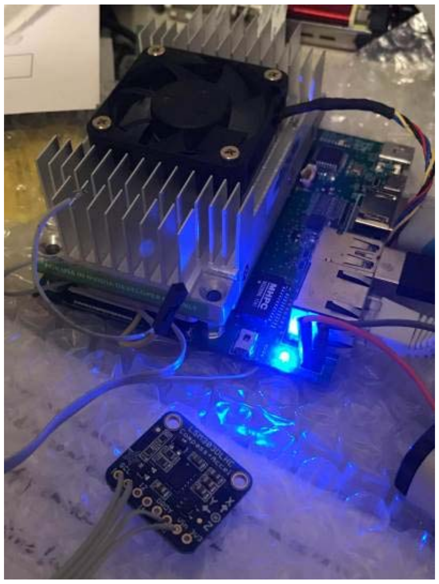
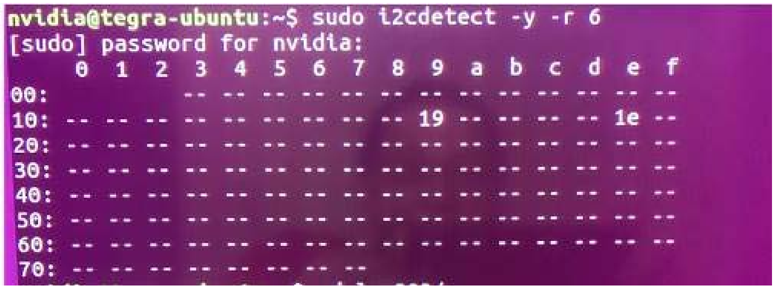


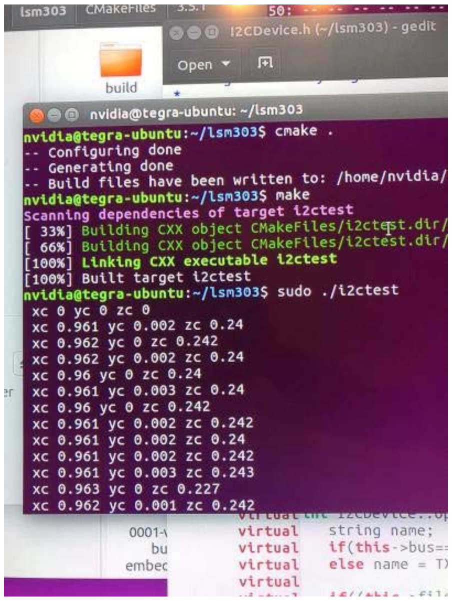
Fig. 6.15; TX2 V2.0 test with I2C LSM303

Software test begins with downloading driver of LSM303 online. Step: “cmake .” to generate makefile; “make” to read makefile just generated and build executable program. In Fig.6.16, “i2cdetect -y -r 6” detects the LSM303 device on 6th I2C(total 8-I2C on TX2) channel, address 19 stands for magnetometer; address 1E stands for accelerometer. In Fig. 6.17, “i2ctest” is the final generated file which can be executed to drive and read 3-axis magnetic fields.



```
nvidia@tegra-ubuntu:~$ sudo i2cdetect -y -r 6
[sudo] password for nvidia:
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- 19 -- -- -- -- 1e --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

Fig. 6.16: i2cdetect



```
nvidia@tegra-ubuntu:~/lsm303$ cmake .
-- Configuring done
-- Generating done
-- Build files have been written to: /home/nvidia/
nvidia@tegra-ubuntu:~/lsm303$ make
Scanning dependencies of target i2ctest
[ 33%] Building CXX object CMakeFiles/i2ctest.dir/
[ 66%] Building CXX object CMakeFiles/i2ctest.dir/
[100%] Linking CXX executable i2ctest
[100%] Built target i2ctest
nvidia@tegra-ubuntu:~/lsm303$ sudo ./i2ctest
 xc 0 yc 0 zc 0
 xc 0.961 yc 0.002 zc 0.24
 xc 0.962 yc 0 zc 0.242
 xc 0.962 yc 0.002 zc 0.24
 xc 0.96 yc 0 zc 0.24
 xc 0.961 yc 0.003 zc 0.24
 xc 0.96 yc 0 zc 0.242
 xc 0.961 yc 0.002 zc 0.242
 xc 0.961 yc 0.002 zc 0.24
 xc 0.961 yc 0.002 zc 0.242
 xc 0.961 yc 0.003 zc 0.243
 xc 0.963 yc 0 zc 0.227
 xc 0.962 yc 0.001 zc 0.242
```

```
virtual    string name;
virtual    if(this->bus==
virtual    else name = T
virtual
```

Fig. 6.17: LSM303 driver installation

The I2C port number should be specified in driver file to fit the hardware configuration. The default value is bus 0 or 1 which only fits Raspberry Pi3.

6.3.3.4 Twin-Camera Test

6.3.3.4.1 SONY OV5603 Camera Test

Previously to simplify the circuit design, two TCA9539 chips are removed from the schematic. However, the system complains a lot of variables missing issues which less likely to be addressed easily. Therefore, the two chips are installed on a small PCB and connected back to the system and everything goes well. Consequently, in the device folder, a SONY OV5693 camera is successfully recognized as “Video0”. The SONY OV5693 is a sensor with 13 million pixels. It supports resolution including 2592x1944, 2592x1458, 1920x1080, 1280x720, 640x480 and so on. A simple way to test the camera is run “nvgstcapture-1.0” in terminal and video appears. The command “nvgstcapture-1.0” supports quite a few functions including CW/CCW 90/180 rotation, white-balance mode, saturation, video-recording, image-multi shot, video encoder configuration and so forth. It’s flexible for users to adjust these parameters as per requirements. Fig. 6.18 shows a live video screen shot.

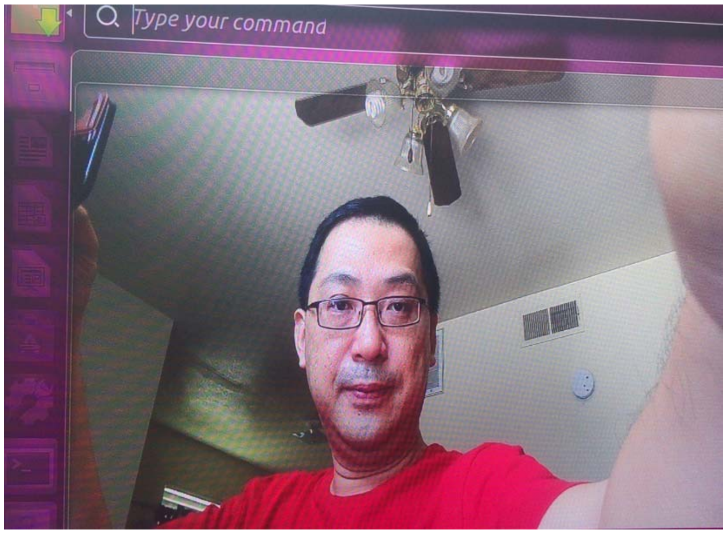


Fig. 6.18: SONY OV5693 camera real-time video

6.3.3.4.2 FLIR LEPTON Thermal Camera Test

A Lepton thermal camera in Fig. 6.19 made by FLIR is also be integrated into the system. The Lepton is a radiometric-capable low wavelength infra-red camera with resolution 80x60 pixels. Each pixel can perform accurate (resolution 0.05°C) and non-contact temperature measurement. It's the smallest thermal sensor in the world, as small as 10mmX10mm, weights 0.55g. The measurement of the Lepton covers -10-450°C±5°C It's a SPI bus camera, tested with driver as following:

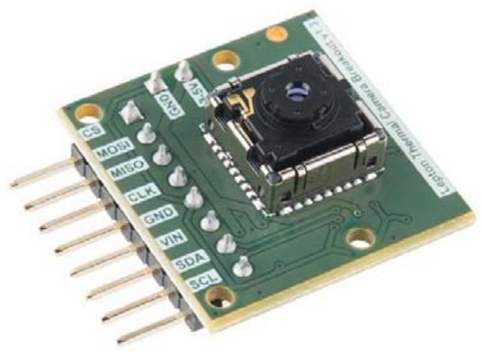


Fig. 6.19: FLIR Lepton thermal camera [38]

To test the Lepton thermal camera, there're three requisites.

1: Enabling SPIDEV3.0

By default, the SPI bus is disabled in system. However, Lepton is a SPI bus camera. It's necessary to enable the SPI bus before reading data from it. From previous configuration including DTB and kernel compilation, the SPIDEV3.0 in TX2 system is ready for user.

2: OpenCV

OpenCV (Open Source Computer Vision Library) is a widely used toolbox in image processing under PYTHON environment. However, OpenCV is not an installed software comes with the TX2 operation system. To install it: “sudo apt-get install python-opencv python-numpy” in terminal.

3: Lepton driver

There’s no applicable driver for Jetson TX2, so the driver for Respberry Pi3 is referred to.

1): Download Lepton driver for RespberryPi3:

git clone https://github.com/groupgets/pylepton.git

2): Extract the package

3): Install driver by “sudo python setup.py install”

4): Run infrad.py test code by “sudo python infrad.py” to preview the thermal image Fig.6.20.

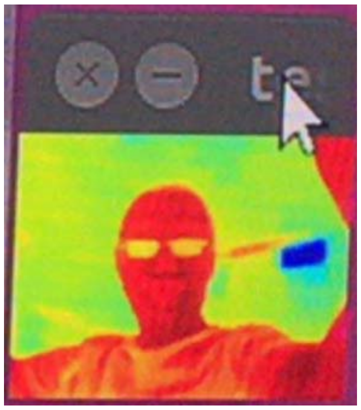


Fig. 6.20: FLIR Lepton thermal camera image

6.3.3.5 UART Constrain and Level-Shifter Chip Selection

As listed in Table 6.3, the TX2 platform supports 5 UARTs. Each port has specific usage and purpose. To apply the platform in the Nano antenna design, each port is examined to ensure nothing wrong happens when customized carrier board is fabricated.

UART0/1/2 are optimized by NVIDIA by using DMA to reduce latency; otherwise, high-speed transmission will be problematic. Therefore, these three UARTs are priority among all five UARTs. The other two UARTs: UART3 and UART7 have constrains not be driven or pull high during power-on; otherwise, it will result in boot failure. These ports should be leave alone at least when booting. For the ttyTHS0; however, lacking detail info on selecting sources from J10 or J21, test result is not so satisfying. UART1 and UART2 finally win the design. The former is specified for communication with drone Matrice 100; the latter is specified with Arduino2560. Right now, the design includes a hardware UART to provide two-way communication between Atmel and TX2 processors. Be aware of the UART1 also has an issue of boot-strapping which is required to leave open, no pull high during power-on; certain prevention is taken care to against the possible booting issue.

Table 6.3: TX2 platform carries with 5 UARTs

Jetson TX2 Pins (Tegra Functions) TX, RX Pins	I/O Block	Typical Usage	NVIDIA Optimiz ation	Wiring	Constrains	Design
UART0 (UART1) H12, G12	DEBUG	Debug	ttyTHS0	Selector: J10 6(TX_1.8V), 8(RX_1.8V) J21 8(TX_3.3V), 10(RX_3.3V)	Tested, RX not stable	No connection
UART1 (UART3) D9, D10	AO	Serial Port	ttyTHS2	J17 4(TX_3.3V), 5(RX_3.3V)	D9 RAM_code strapping, must not be driven or pull H or L during power-on, violation results in malfunction	Matrice 100 communicati on
UART2 (UART2) B16, B15	UART	M.2 socket for external WLAN / BT	ttyTHS1	J18 32(TX_3.3V), 34(RX_3.3V)	Working perfect	Arduino2560 communicati on
UART3 (UART4) H10, H9	CONN	Misc. Available if not used for on-module WLAN / BT (selected by on-module multiplexor)	No, ttyS0	J21 8(TX_3.3V), 10(RX_3.3V)	H10 must not be driven or pull H during power-on, violation results in boot failure	No connection
UART7 (UART7) D8, D5	AO	2nd Debug/Misc.	No, ttyS1	J10 27(TX_1.8V),29(RX_1.8 V)	D8 must not be driven or pull H during power-on, violation results in boot failure	No connection

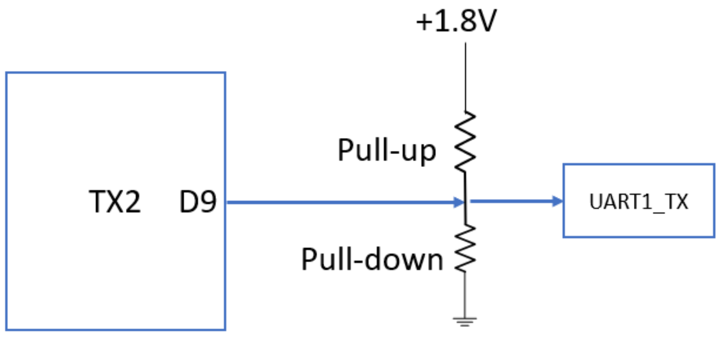


Fig. 6.21: UARTs and Pull/down resistors

From the Fig. 6.21 one pull-up and one pull-down resistors are already connected on the D9 line of UART1_TX. A pull-up resistor is to lock an uncertain signal to be HIGH by providing current from power supply. While a pull-down resistor is to release current from port to ground so that the port level can be locked LOW. For the connection with pull-up and pull-down resistors exist simultaneously, stands for no specific purpose which is fit for the situation of “no driven or pull H or L during power-on”. The UART1 will communicate with Atmega2560 to share GPS info to steer antenna so that a level shifter circuit is needed because TX and RX pins on TX2 UART1 is 1.8V tolerance cannot directly communicate with 5V serial port of Atmage2560. Otherwise, when 5V TX of Arduino2560 is sending data to 1.8V RX of TX2 UART1; the RX couldn’t bear such high voltage and generate excessive heat then burn the RX port. The level shifter circuit includes two ways: TX and RX. To reduce the circuit, an integrated circuit TXB0104 in Fig. 6.22 is integrated in the design. TXB0104 is a four-channel level shifter. It has a enable pin; once set HIGH (refer to VccA), it automatically detects portA and portB for signals and guide the signal from portA to portB or portB to portA.

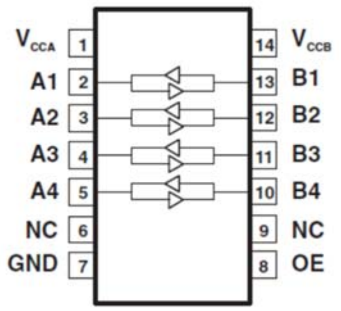


Fig. 6.22: Internal structure of TXB0104 [39]

The situation is that the TX2 couldn't boot when the level ship TXB0104 is soldered into the circuit in Fig. 6.23. After investigation, the "automatic direction sensing" function is believed to be the source of the issue. The TXB0104 continuously checks the both ports in turn. Once it detects signal on the port, it automatically opens the channel and sets the detected signal as signal source. When booting, the TXB0104 detects noise on portB then converts the noise as a signal and lowers its amplitude 2.8 times. Because the compact design of the PCB, the noise generated by DC-DC power supply is easily detected by TXB0104 and accidently changes the booting status of the TX2 module so that the TX2 has boot failure or no desktop issue.

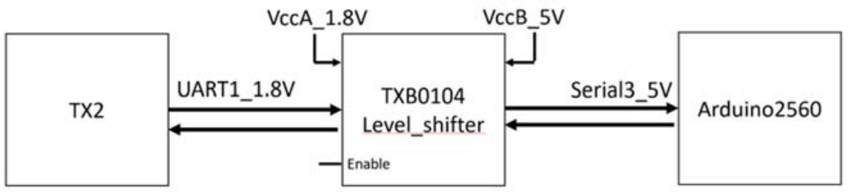


Fig. 6.23: TXB0104 level-shifter

Solution is obvious. Either eliminating the noise or disable the "automatic direction sensing" function. Although the TX pin of the UART1 is vulnerable to interference, it's an output port. If it can be defined as "output-only" as signal source, it should not be affected by external signal. According to this theory, a manual-direction controlled level shifter: 74LVC2T45 by Texas Instrument is found to be suitable. See Fig. 6.24.

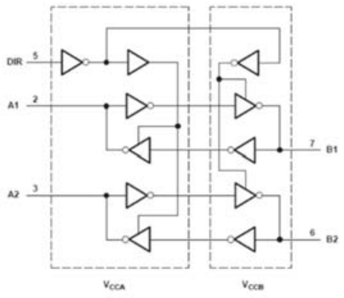


Fig. 6.24: 74LVC2T45 level-shifter internal structure [39]

74LVC2T45 is a twin-channel level shifter operates from 1.65V-5.5V. In Fig. 6.25, the pin "DIR" indicates the direction between portA and portB. The "DIR" sets HIGH (refer to VccA) convers data from portA to portB; sets LOW converts portB to portA.

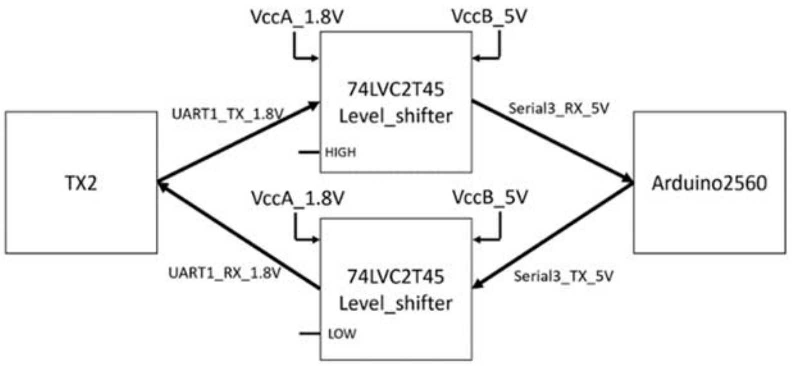
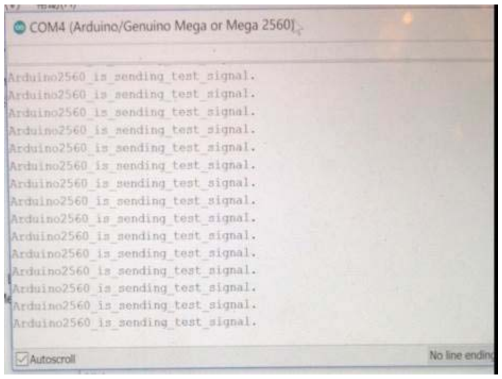


Fig. 6.25: 74LVC2T45 bridges data between TX2 and Atmega2560

The V2.0 carrier board has quite a few channels need level shifters converting between 1.8V and 3.3V, 1.8V and 5V, 3.3V and 5V. To increase integration and reduce components, multi-channel level shifter chips are used to satisfy this demand. The TXB0104/0108 can still be used in GPIO and UART application of level-shifting.

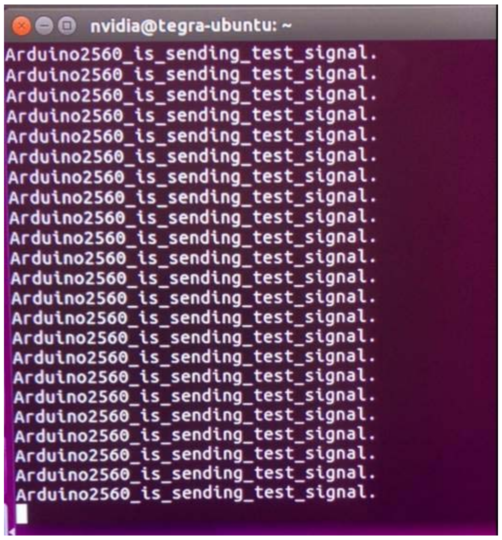
6.3.3.6 Atmega2560 Test with TX2 UART

The test results after level-shifter replaced is excellent. Booting issue handled; also, TX2 UART to Atmega2560 Serial3 communicates well. After power-on, this UART1 port connects with the UART port of drone Matrice100. To simplify the test, an Atmega2560 development board is programmed to communicate with UART1 by its serial port. In the Fig. 6.26, the Atmega2560 is programmed to continuously sending string ”Atmega2560 is sending test signal” via its serial port3. This port connects UART1 on the TX2. In the Fig. 6.27, a MINICOM is running in TX2 terminal and is configured with “ttyTHS2, 9600, 8N1” to monitor the UART1 port. From terminal of TX2, the string shows correctly.


COM4 (Arduino/Genuino Mega or Mega 2560)

```
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
```

Fig. 6.26: Atmega2560 sends string to TX2

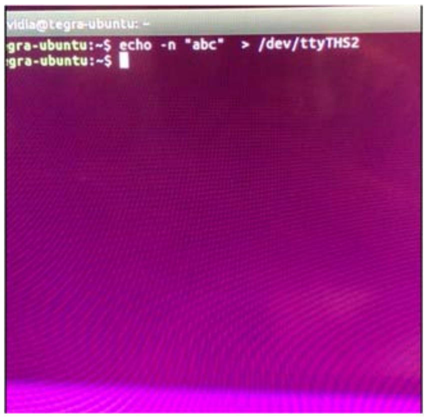

nvidia@tegra-ubuntu: ~

```
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
Arduino2560_is_sending_test_signal.
```

Fig. 6.27: TX2 receives string from Atmega2560

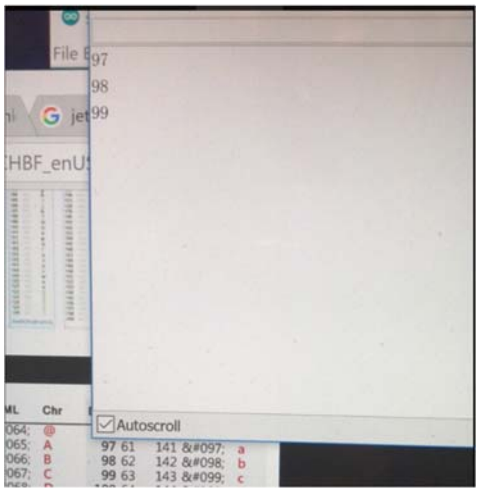
Above illustrations indicate the Atmega2560 can successfully communicate with Jetson TX2.

Another test is sending ascending integer value from Atmega2560 to TX2 in Fig. 6.28, in TX2 terminal the data receives correctly in Fig. 6.29. This test also confirms 115200kbps data rate runs well. The third test is sending characters "ABC" from TX2 to Atmega2560, TX2 in terminal shows "97, 98, 99". By checking ASCII table, "97, 98, 99" stand for "a b c".



```
vidia@tegra-ubuntu: ~
gra-ubuntu:~$ echo -n "abc"  > /dev/ttyTHS2
:gra-ubuntu:~$
```

Fig. 6.28: TX2 sends string to Atmega2560



```
97
98
99
```

```
064;  @        97 61   141 &#097;   a
065;  A        98 62   142 &#098;   b
066;  B        99 63   143 &#099;   c
067;  C
```

Fig. 6.29: Atmega2560 receives string from TX2

6.3.3.7 GPIO Test

TX2’s GPIO configuration is defined in /kernel/kernel-4.4/include/dt-bindings/gpio/tegra186-gpio.h

```
GPIOs 320-511, platform/2200000.gpio, tegra-gpio
GPIOs 256-319, platform/c2f0000.gpio, tegra-gpio-aon
/* GPIOs implemented by main GPIO controller */
#define TEGRA_MAIN_GPIO_PORT_A 0
#define TEGRA_MAIN_GPIO_PORT_B 1
#define TEGRA_MAIN_GPIO_PORT_C 2
#define TEGRA_MAIN_GPIO_PORT_D 3
#define TEGRA_MAIN_GPIO_PORT_E 4
#define TEGRA_MAIN_GPIO_PORT_F 5
#define TEGRA_MAIN_GPIO_PORT_G 6
#define TEGRA_MAIN_GPIO_PORT_H 7
#define TEGRA_MAIN_GPIO_PORT_I 8
#define TEGRA_MAIN_GPIO_PORT_J 9
#define TEGRA_MAIN_GPIO_PORT_K 10
#define TEGRA_MAIN_GPIO_PORT_L 11
#define TEGRA_MAIN_GPIO_PORT_M 12
#define TEGRA_MAIN_GPIO_PORT_N 13
#define TEGRA_MAIN_GPIO_PORT_O 14
#define TEGRA_MAIN_GPIO_PORT_P 15
#define TEGRA_MAIN_GPIO_PORT_Q 16
#define TEGRA_MAIN_GPIO_PORT_R 17
#define TEGRA_MAIN_GPIO_PORT_T 18
#define TEGRA_MAIN_GPIO_PORT_X 19
#define TEGRA_MAIN_GPIO_PORT_Y 20
#define TEGRA_MAIN_GPIO_PORT_BB 21
#define TEGRA_MAIN_GPIO_PORT_CC 22
#define TEGRA_MAIN_GPIO_PORT_DD 23

#define TEGRA_MAIN_GPIO(port, offset) \
     ((TEGRA_MAIN_GPIO_PORT_##port * 8) + offset)

/* GPIOs implemented by AON GPIO controller */
#define TEGRA_AON_GPIO_PORT_S 0
#define TEGRA_AON_GPIO_PORT_U 1
#define TEGRA_AON_GPIO_PORT_V 2
#define TEGRA_AON_GPIO_PORT_W 3
#define TEGRA_AON_GPIO_PORT_Z 4
#define TEGRA_AON_GPIO_PORT_AA 5
#define TEGRA_AON_GPIO_PORT_EE 6
#define TEGRA_AON_GPIO_PORT_FF 7

```

```
#define TEGRA_AON_GPIO(port, offset) \
     ((TEGRA_AON_GPIO_PORT_##port * 8) + offset)
```

As offset for TEGRA_MAIN_GPIO is 320, it's easy to calculate the GPIO number by providing port number. E.g. known TX2 pin number B19 is GPIO3_PJ.06: 320 + (9 * 8 + 6) = 398; hence, the port number GPIO3_PJ.06 reflects GPIO398.

For offset of TEGRA_AON_GPIO is 256, it's easy to calculate the GPIO number by providing port number. E.g. known TX2 pin number G14 is GPIO3_PAA.02: 256 + (5 * 8 + 2) = 298; hence, the port number GPIO3_PAA.02 reflects GPIO298.

In the system design, 4 GPIOs are added for future usage. They're B19(GPIO389), F2(GPIO398), G14(GPIO395) and H13(GPIO388).

GPIO389 test:

```
$sudo su
$/sys/class/gpio echo 389> export
$/sys/class/gpio/gpio389/cat direction   //in
$/sys/class/gpio/gpio389/echo out > direction
$/sys/class/gpio/gpio389/cat direction   //out
$/sys/class/gpio/gpio389/cat value // 1
$/sys/class/gpio/gpio389/echo 0 >  value
$/sys/class/gpio/gpio389/cat value //0
```

A green LED in Fig. 6.30 is connected in "sink current" mode so set GPIO 0/Low triggers it.

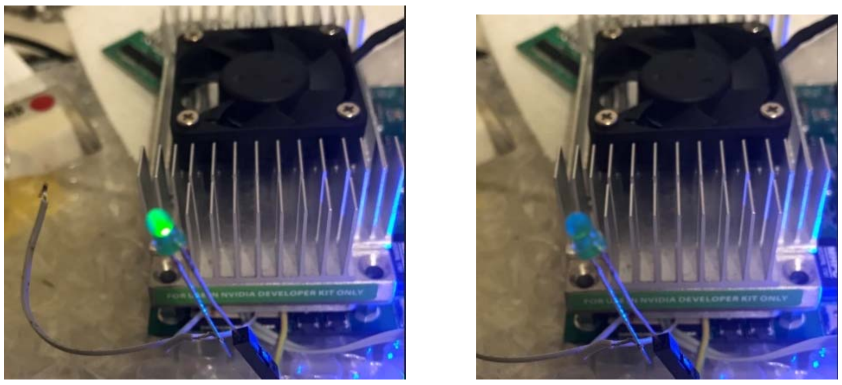


Fig. 6.30: GPIO389 test with a green LED

GPIO PWM: In this Nano antenna system, PWM signal is adopted to control antenna rotating motor and gimble motor in the future. For this application, NVIDIA provides two solutions:

1) PWM generated by Atmel chips. It's easy and best option so that image processing in TX2, and TX2 sends information to Atmel controller to perform actual PWM motor drive. Because PWM is a real-time low-level operation needs hard-real time consistent timing, it's more suitable than application processor TX2.
2) PWM directly generated by TX2. This is more complex and might not provide precise timing signal.

Hence, the PWM motor drive circuit will be implemented by Atmega2560.

6.3.4 Version: V3.0

The carrier V3.0 corrects quite a few issues found in V2.0 and adds two chips TCA9539. The design further shrinks the size from 90x78 to 90x66, 15% smaller but more components. This version pays more attention to the boot RAM code strapping to avoid boot failure issue and no desktop issue. Also, this version adds CAN bus transceiver SN65HVD257 to implement high-speed communication with DJI Matrice 100. Fig. 6.31 is the Altium 17.0 simulated PCB; Fig. 6.32 is the comparison of V2 and V3 to show how much space is reduced.

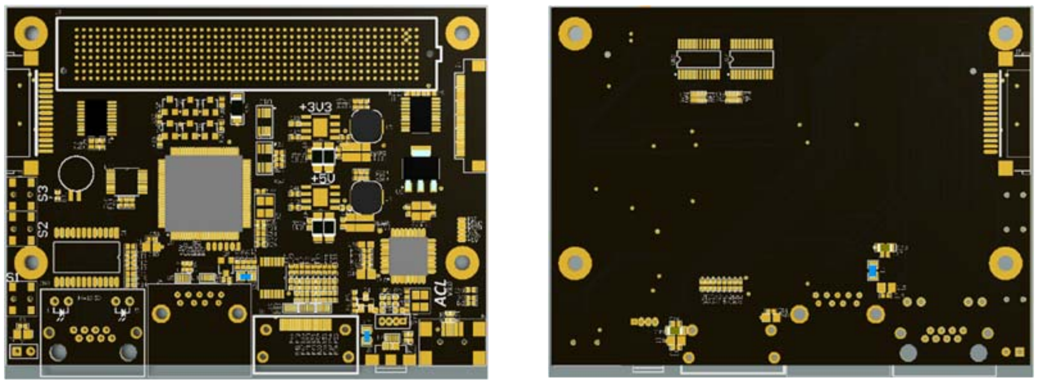


Fig. 6.31: Top layer and bottom layer of V3.0 PCB

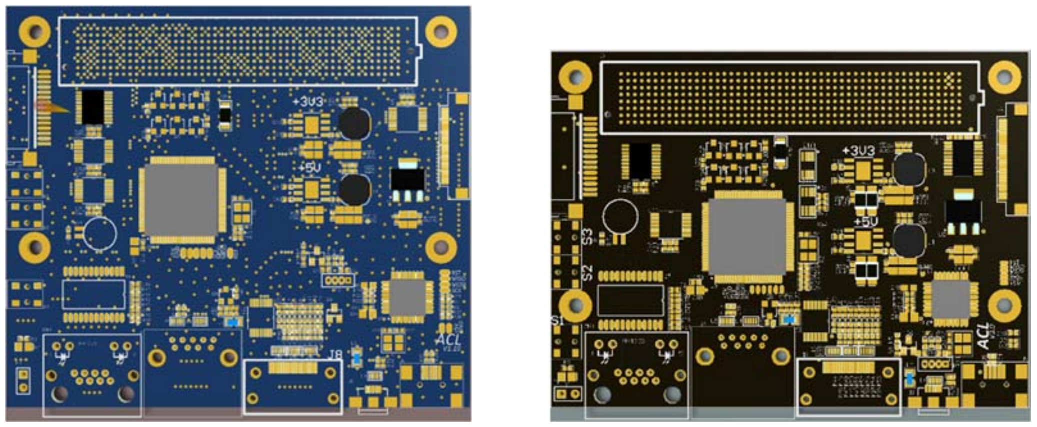


Fig. 6.32: Comparison of V2.0 and V3.0 PCBs

After soldering, the carrier board looks like Fig. 6.33. The top layer is almost covered by various of components; the bottom layer has much less components but quite a few wirings of differential pairs. Fig. 6.34 shows the whole system after assembling.

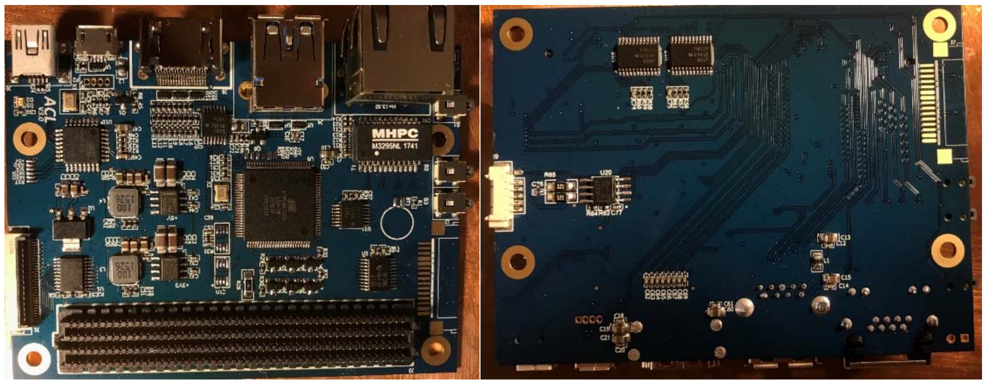


Fig. 6.33: TX2 V3.0 carrier board

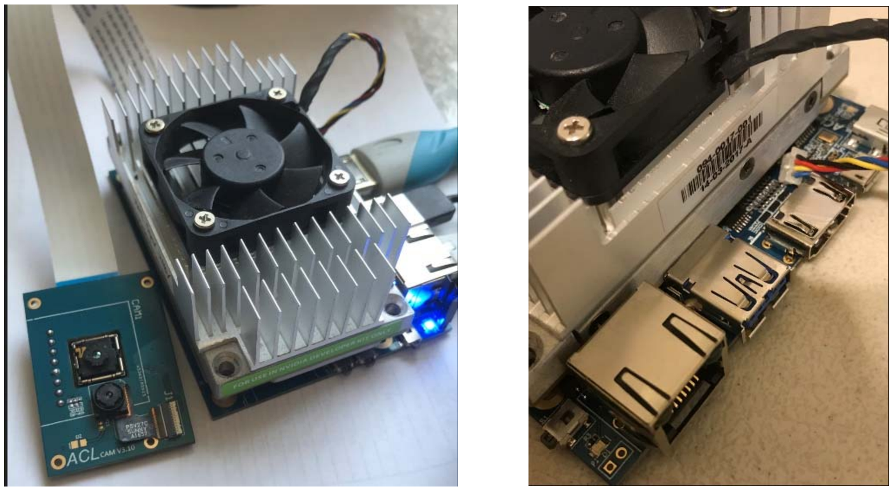


Fig. 6.34: Assembling finish TX2 carrier board

Fig. 6.35 shows a combination of SONY OV5693 and Lepton thermal video test on local TX2 carrier board V3.0. The Lepton image is re-sized to be 800x600 by function cv2.re_size. The resolution of OV5693 is also 800x600. The two videos demonstrate excellent working performance.

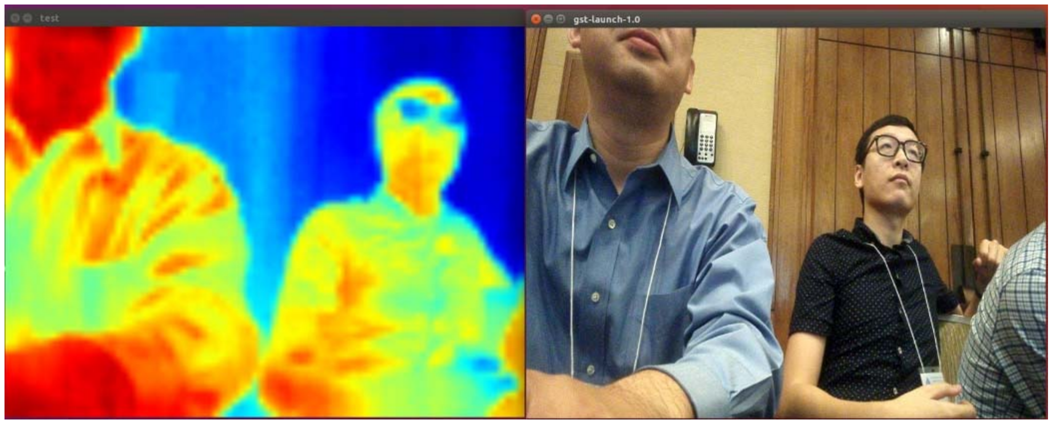


Fig. 6.35: Combining SONY OV5603 and Lepton thermal video

The Fig. 6.36 measures the thermal distribution of TX2 carrier board. The picture indicates the CPU is the main source of the heat. Red indicates hot object surface temperature while blue means cold object surface temperature.

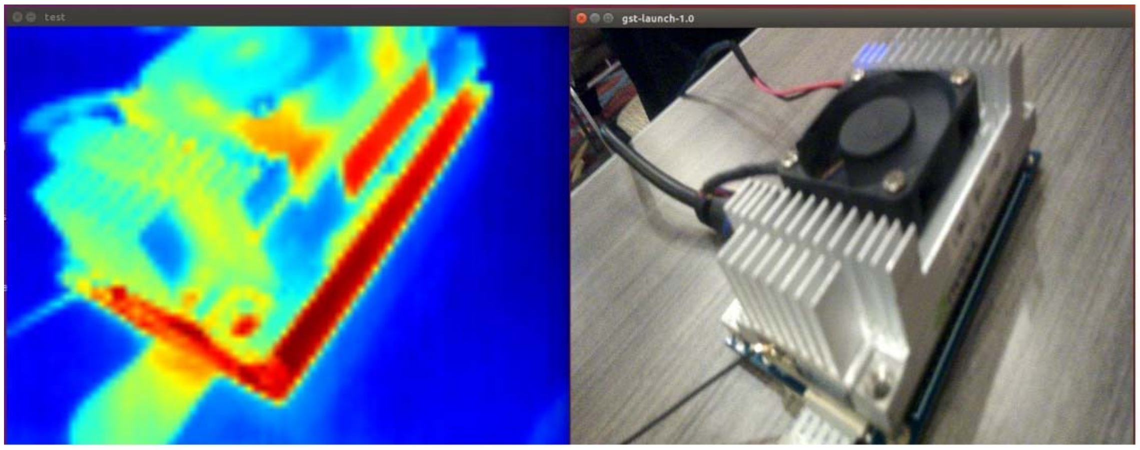


Fig. 6.36: TX2 temperature inspection

6.3.5 Application on Nvidia Jetson TX2

6.3.5.1 Ubuntu OS and ROS Installation

New TX2 module doesn't come with Ubuntu operation system. Download Jetpack3.2 (see Fig. 6.37) will install Ubuntu 16.04 and NVIDIA development packages together.

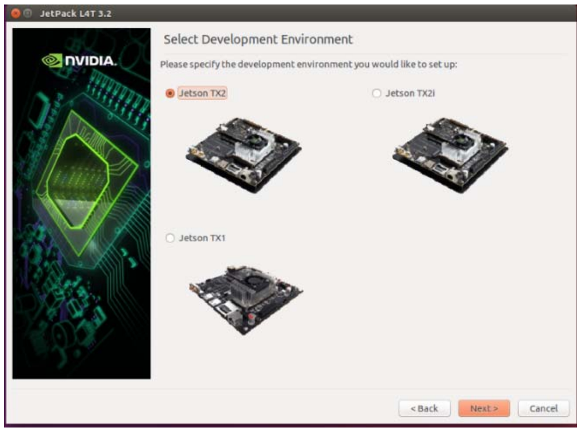

JetPack L4T 3.2
NVIDIA
Select Development Environment
Please specify the development environment you would like to set up:
Jetson TX2
Jetson TX2i
Jetson TX1
< Back
Next >
Cancel

Fig. 6.37: JetPack 3.2 interface [23]

To install robotics operation system (ROS), installation can follow ROS official website, or download an all-in-one package:

```
$ git clone
     https://github.com/jetsonhacks/installROSTX2.git
     $ cd installROSTX2
     $ ./installROSTX2
$ ./setupCatkinWorkspace.sh catkin_ws     // Create working
     space
```

6.3.5.2 Data Sharing between Nodes by ROS

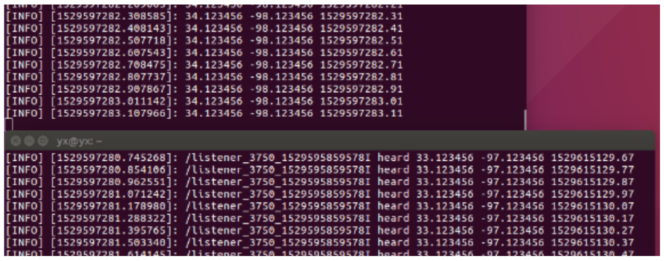
To share data between two or more nodes/computers, publishers and subscribers are created after these devices are linked together inside of one local network.

Table 6.4: Configuration of connected equipment

	TX2 (Server)	**Laptop1**	**Laptop2**	**Laptop N**
ROS_MASTER_URI	http://192.168.1.35:11311			
ROS_IP		192.168.1.2	192.168.1.3	192.168.1.N
etc\hosts	192.168.1.35 tegra-ubuntu 192.168.1.2 laptop1 192.168.1.3 laptop2 192.168.1.N laptopN	192.168.1.35 tegra-ubuntu 192.168.1.2 laptop1 192.168.1.3 laptop2 192.168.1.N laptopN	192.168.1.35 tegra-ubuntu 192.168.1.2 laptop1 192.168.1.3 laptop2 192.168.1.N laptopN	192.168.1.35 tegra-ubuntu 192.168.1.2 laptop1 192.168.1.3 laptop2 192.168.1.N laptopN

After configuration on each device, each device can successfully ping other devices by its name in terminal, e.g. on TX2, ping laptop1 should receive successful response.

Now creating a publisher node “talker1” on TX2 to continuously sending a GPS string topic “SendingTest”, the laptop1 node “listener1” should receive it by subscribing the same topic “SendingTest”. Also, on the laptop1, node “talker1” publishes another GPS string topic “SendingTest2”, on the TX2 a subscriber “listener2” should successfully receive the same topic string. Therefore, the experiment confirms the data exchange between two device, multi-node can be successfully performed as showed in Fig. 6.38. To show the topology of the node/topic relationship, run “rqt_graph” as showed in Fig. 6.39.



```
[INFO] [1529597282.308585]: 34.123456 -98.123456 1529597282.31
[INFO] [1529597282.408143]: 34.123456 -98.123456 1529597282.41
[INFO] [1529597282.507718]: 34.123456 -98.123456 1529597282.51
[INFO] [1529597282.607543]: 34.123456 -98.123456 1529597282.61
[INFO] [1529597282.708475]: 34.123456 -98.123456 1529597282.71
[INFO] [1529597282.807737]: 34.123456 -98.123456 1529597282.81
[INFO] [1529597282.907867]: 34.123456 -98.123456 1529597282.91
[INFO] [1529597283.011142]: 34.123456 -98.123456 1529597283.01
[INFO] [1529597283.107966]: 34.123456 -98.123456 1529597283.11
```

```
[INFO] [1529597280.745268]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615129.67
[INFO] [1529597280.854106]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615129.77
[INFO] [1529597280.962551]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615129.87
[INFO] [1529597281.071242]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615129.97
[INFO] [1529597281.178980]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615130.07
[INFO] [1529597281.288322]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615130.17
[INFO] [1529597281.395765]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615130.27
[INFO] [1529597281.503340]: /listener_3750_1529595859578I heard 33.123456 -97.123456 1529615130.37
```

Fig. 6.38: GPS coordinates exchange test

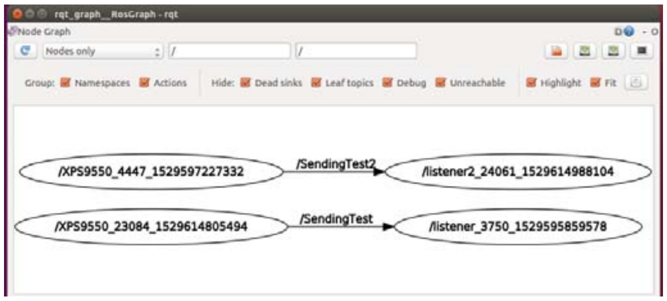

rqt_graph__RosGraph - rqt
Node Graph
Nodes only
Group: Namespaces Actions
Hide: Dead sinks Leaf topics Debug Unreachable
Highlight Fit
/XPS9550_4447_1529597227332
/SendingTest2
/listener2_24061_1529614988104
/XPS9550_23084_1529614805494
/SendingTest
/listener_3750_1529595859578

Fig. 6.39: Topology of the node/topic relationship

6.3.5.3 Access-point implementation on the TX2

Previously a HUAWEI WS323 wireless router connects TX2 and Nano antenna to provide WiFi service. Actually, the HUAWEI is also a tiny LINUX host computer. By adopting the powerful TX2 platform, it's applicable to integrate the HUAWEI wireless router into the TX2. The wireless router function can be implemented by following configuration in TX2:

1) sudo apt-get install hostapd dnsmasq
2) create file /etc/hostapd/hostapd.conf and put following text in it. SSID and wpa_pass phrase can be set here.

```
interface=wlan0
ssid=UNT_Drone_Test

hw_mode=g

channel=3
```

3) Edit /etc/network/interfaces with following text

```
auto lo
iface lo inet loopback
auto wlan0
iface wlan0 inet static
hostapd /etc/hostapd/hostapd.conf
address 192.168.8.1
netmask 255.255.255.0
```

4) Edit /etc/dnsmasq.conf with following

```
interface=lo,wlan0
no-dhcp-interface=lo
dhcp-range=192.168.8.20,192.168.8.254,255.255.255.0,12h
```

5) Add following line in /etc/sysctl.conf file. If this line already exits and is commented out, uncomment it.

```
net.ipv4.ip_forward=1
```

6) Edit /etc/NetworkManager/NetworkManager.conf with following

```
[main]
plugins=ifupdown,keyfile,ofono
dns=dnsmasq
[ifupdown]
managed=false
```

7) Add this line to /etc/rc.local

```
echo 2 > /sys/module/bcmdhd/paramters/op_mode node
```

8) Restart your system and check if hostapd is already running on your system. If it is, kill it and run the following command.

    sudo hostapd -B /etc/hostapd/hostapd.conf

By above setting, a laptop or smart phone can scan the created SSID of the AP; hence, the HUAWEI wireless router is removed.

6.3.5.4 Video Streaming Test (Server and Client)

The video camera used in the system is SONY OV5693 which is successfully mounted to system with “Video0”. Command “nvgstcapture-1.0” can show a preview window to confirm its working status on local TX2 platform. For remote streaming of the video, a RSTP server is created to provide stream service.

On Jetson TX2 server side:

Install gst-rtsp-server-1.8.3

1) $ wgethttps://gstreamer.freedesktop.org/src/gst-rtsp-server/gst-rtsp-server-1.8.3.tar.xz
2) $ tar -xvf gst-rtsp-server/gst-rtsp-server-1.8.3
3) $ cd gst-rtsp-server-1.8.3
4) $ ./configure –libdir=/usr/lib/arm-linux-gnueabihf/
5) $ make
6) $ cd examples/
7) $ ./test-launch "(nvcamerasrc sensor-id=0 ! video/x-raw(memory:NVMM), width=1920, height=1080, framerate=30/1, format=I420 ! nvvidconv flip-method=2 ! video/x-raw, width=720, height=480, framerate=30/1, format=I420 ! timeoverlay ! omxh265enc ! rtph265pay name=pay0 pt=96)"
8) $ sudo iptables -A INPUT -p tcp --dport 8554 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
9) $ sudo iptables -A OUTPUT -p tcp --sport 8554 -m conntrack --ctstate ESTABLISHED -j ACCEPT

To perform a simple test on TX2 desktop, run:

```
$ gst-launch-1.0 rtspsrc location=rtsp://127.0.0.1:8554/test ! 'application/x-
```

```
rtp, media=(string)video' ! decodebin ! videoconvert !
Ximagesink
```

A preview window will appear with real time video. Note the server serves only one client each time so shut it off to perform next remote test.

On the host computer:

```
$ ssh nvidia@192.168.1.9 -L 8554:localhost:8554
```

in which 192.168.1.9 is TX2's IP address. Type "ifconfig" in terminal can verify the IP address in using. Or, check attached device from wireless router interface and find the IP address assigned by system.

Install VLC video player by:

```
$sudo apt-get update
$sudo apt-get install vlc browser-plugin-vlc
```

In terminal run "vlc" to open a VLC window, open media-network: rtsp://localhost:8554/test or rtsp://127.0.0.1:8554/test to display real time video from TX2 camera (Fig. 6.40). Note one server serves only one client. If more clients appear, following method can be applied to support more video traffic.

1): One camera connects one rtsp server with several URI:

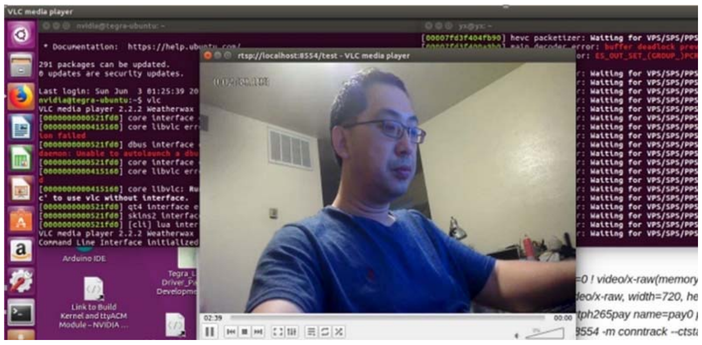
```
server0: rtsp://IP:port/link1
server0: rtsp://IP:port/link2
server0: rtsp://IP:port/lang3
server0: rtsp://IP:port/lang4
```

2): one camera with several rtsp servers with own URI:

```
server0: rtsp://IP:port0/link1
server1: rtsp://IP:port1/link2
server2: rtsp://IP:port2/lang3
server3: rtsp://IP:port3/lang4
```

3): One camera one RTSP server one URI several sub-streams:

server0: rtsp://IP:port/stream (contains sub-streams link1, link2, lang3 lang4)


VLC media player
nvidia@tegra-ubuntu: ~
yx@yx: ~
rtsp://localhost:8554/test - VLC media player
Arduino IDE
Link to Build Kernel and ttyACM Module - NVIDIA ...
02:39
00:00

```
 * Documentation:  https://help.ubuntu.com/

291 packages can be updated.
0 updates are security updates.

Last login: Sun Jun  3 01:25:39 20
nvidia@tegra-ubuntu:~$ vlc
VLC media player 2.2.2 Weatherwax
[0000000000521fd0] core interface
[0000000000415160] core libvlc err
ion failed
[0000000000521fd0] dbus interface
daemon: Unable to autolaunch a dbu
[0000000000521fd0] core interface
[0000000000415160] core libvlc err
d
[0000000000415160] core libvlc: Ru
c' to use vlc without interface.
[0000000000521fd0] qt4 interface e
[0000000000521fd0] skins2 interfac
[0000000000521fd0] [cli] lua inter
VLC media player 2.2.2 Weatherwax
Command Line Interface initialized
```

```
[00007fd3f404fb90] hevc packetizer: Waiting for VPS/SPS/PPS
main decoder error: buffer deadlock prev
or: ES_OUT_SET_(GROUP_)PCR
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
r: Waiting for VPS/SPS/PPS
```

```
=0 ! video/x-raw(memory
deo/x-raw, width=720, he
tph265pay name=pay0
8554 -m conntrack --ctst
```

Fig. 6.40: Captured from host laptop, streaming from TX2 carrier V3.0

6.3.5.5 Lepton Thermal Video Streaming

To receive SONY OV5693 and Lepton thermal video simultaneously, both TX2 (server side) and laptop (client side) are to be configurated.

On TX2 server side:

OV5693 camera: run gst-rtsp-server-1.8.3 by:

./test-launch "(nvcamerasrc sensor-id=0 ! video/x-raw(memory:NVMM), width=1920, height=1080, framerate=30/1, format=I420 ! nvvidconv flip-method=2 ! video/x-raw, width=720, height=480, framerate=30/1, format=I420 ! timeoverlay ! omxh265enc ! rtph265pay name=pay0 pt=96)"

Lepton thermal camera: run “rosrun flir_lepton_camera lepton_video.py” to publish the Lepton image as a topic under “root” environment by “sudo su” due to permission limitation on hardware operation.

On laptop client side:

Under “root” environment by “sudo su” and run “rqt --force” to open a new window. Click “video_image” to show thermal video by subscribing topic ”chatterimg”.

Click “START” bottom on the interface to link OV5693 camera on the server.

To successfully show OV5693 video, care of the modification will be taken.

In function start_clicked(self) in my_module.py,

Self.capture = cv2.VideoCaptur(“rtsp://192.168.1.35:8554/test”), the IP address of the rtsp server should use the verified address in previous test.

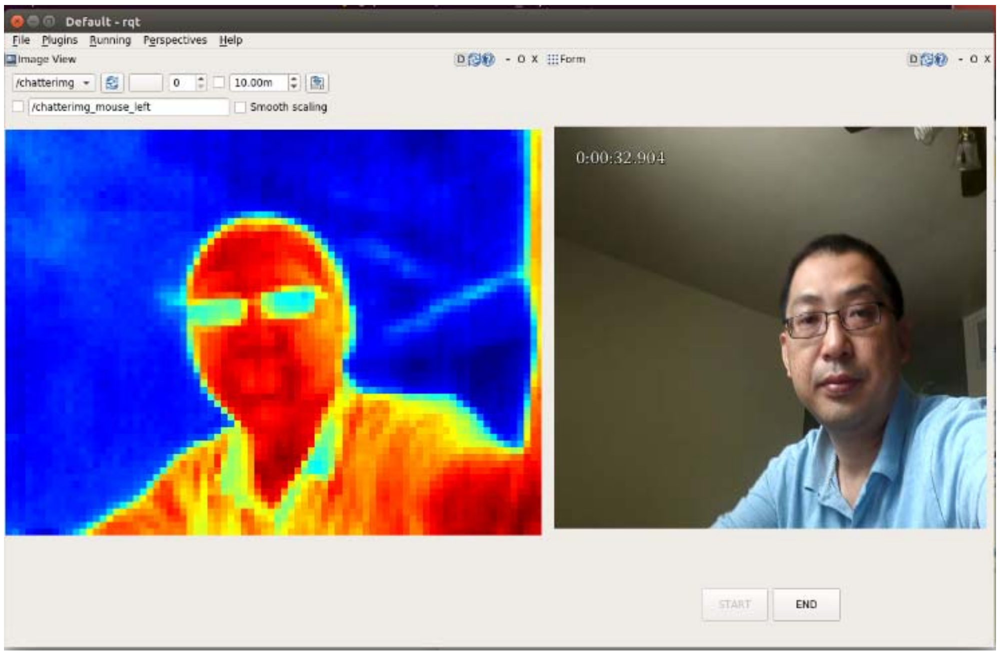


Fig. 6.41: Rqt streaming of SONY OV5693 and Lepton thermal video

So far, the real time video from SONY OV5693 and thermal video from Flir Lepton implemented by RQT is applicable in the project of “Drone WiFi System” to wirelessly transfer from one drone to another (see Fig. 6.41). The laptop connects with the receiver drone can successfully display the two videos at command center.

6.3.5.6 Image Process and Target Tracing

The TX2 platform has powerful GPU to perform continuous and complex computation for TensorRT deep learning applications. To perform Deep Learning on the

TX2 platform, NVIDIA Inference will be installed on the TX2. The Inference is based on TensorRT developed by GOOGLE.

1) Install Jetson Inference repository:

   Install git and cmake tools:

   ```
   $ sudo apt-get install git cmake
   ```

   Clone Jeston Inference repository:

   ```
   $ git clone https://github.com/dusty-nv/jetson-inference
   $ cd jeston-inference
   $ mkdir build
   $ cd build
   $ cmake ../
   $ make
   ```

2) Test object recognition by imagenet

   The object recognition can be performed on TX2 by following:

   ```
   $ ./imagenet-camera googlenet
   ```

   The googlenet is a trained dataset contains 12 core groups over 1000 different classes. In the Fig. 6.42, the drinking water is recognized with "water bottle" with confidence 97.56%; a keyboard is recognized with "computer keyboard" with confidence 74.73%.

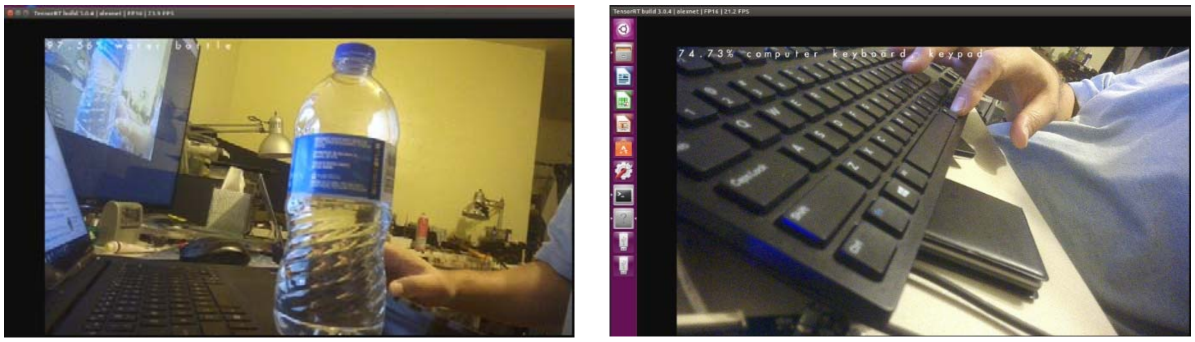


Fig. 6.42: Object recognition

3) Test object recognition and tracing

The object recognition can be performed on TX2 by following:

```
$ ./image-camera coco-bottle
```

The coco is a trained dataset by Microsoft contains 80 object categories over 1.5 million different instances. “coco-bottle” is one of the object categories. In the following figure, the water bottle is flawlessly bounding labelled. In Fig. 6.43, by moving of the object, the bounding label is fixed on the object and is moving correspondingly.

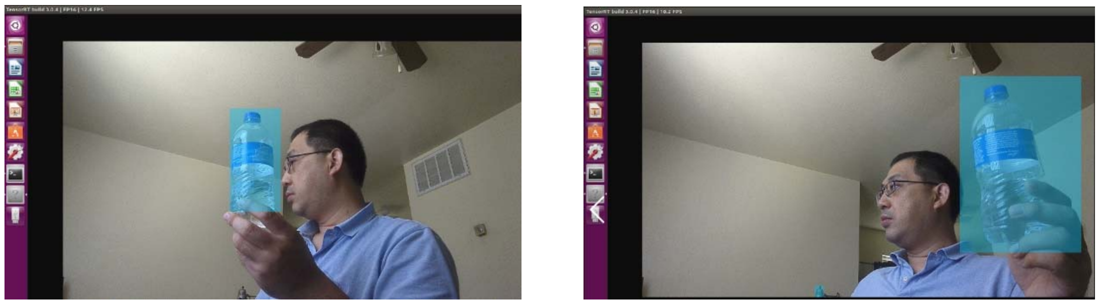


Fig. 6.43: Target tracing

Fig. 6.44 shows the target tracing with human body by:

```
$ ./detectnet-camera pednet   to detect single pedestrian
$./detectnet-camera multiped  to detect multi pedestrian
```

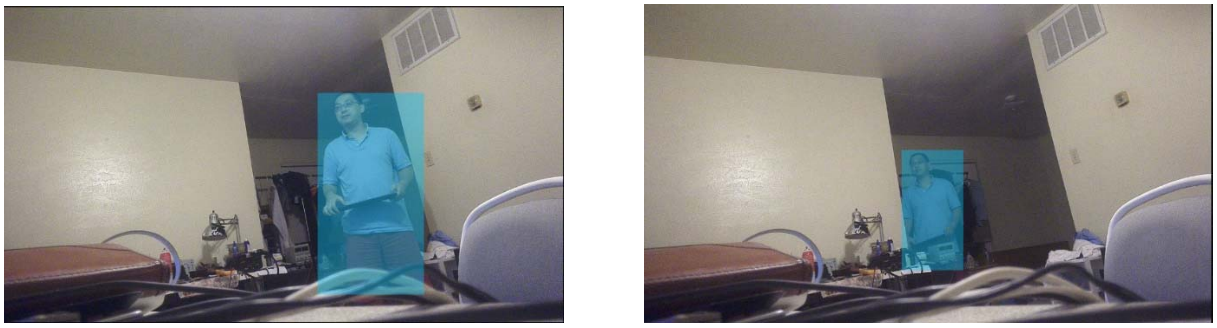


Fig. 6.44: Human body tracing

6.3.5.7 Power Consumption of TX2 Platform

Because the TX2 is a battery-powered portable computational platform, the power consumption is critical to be measured so that a suitable battery can be selected. The current consumption under different working conditions are measured in the Table 6.5.

Table 6.5; Power consumption in different conditions

Voltage: 8.4V	Booting	Desktop	Compile Kernel	ROS+2 CAM Streaming	TensorRT
Current: (A)	0.65	0.42	0.9	0.68	1.5

Note that the current consumption of TensorRT including target recognition and object tracing is 1.5A average (up to 1.9A) at 8.4V (2S Li-ion fully charged). The power consumption is huge. If powering the TX2 carrier board by 2S2200mAh Li-ion cells, the working time is around 1hour. To save power, another option is that moving the TensorRT function to laptop. Once the host laptop operates with LINUX UBUNTU16.04 system with TensorRT installed, it can perform target recognition and object tracing once receives video streaming from TX2. The TX2 just captures real time and thermal video then transmit them to host laptop so the power consumption can be much lower. Therefore, the TX2 can use much smaller and lighter battery to extend its hovering time in the air.

CHAPTER 7

FIELD TEST

The field test is done with 300M, 1000M, 3000M and 5000M. Because the previous V1 antenna using Nano Station2 couldn't communicate over 1000M, no data was obtained in that test. Therefore, the performance test is on range on 300M, 1000M and even more 3000M, 5000M.

On one hand, NanoStation2 is a 2.4G device with 10dB gain patch antenna while Nano LOCO M5 is a 5G device with 13dB gain patch antenna. The gain on LOCO M5 is higher than Station2 so that the transmission range of LOCO M5 is longer and higher bandwidth.

On the other hand, the 3rd version antenna assisted by GPS coordinate alignment is much accurate than the fixed degree alignment which further enhances beam strength of the wireless communication. Also, the carbon frame of the drone helps signal isolation and reduce interference which helps signal quality.

The transmission performance comparison between previous Nano Station2 and current LOCO5 is listed in table 7.1.

Table 7.1: Comparison of Nano Station2 and Loco M5 directional antennas†

Distance (Meter)	V1 with Nano Station2	V3 with Nano LOCO M5
300	18.4Mbps	36Mbps
1000	7.8Mbps	12Mbps
3000	Link lost	2Mbps
5000	Link lost	800kbps

† Data of Table 7.1 have been previous published in part from [5] Y. Gu, M. Zhou, S. Fu, and Y. Wan, "Airborne WiFi networks through directional antennae: An experimental study", IEEE Wireless Communications and Networking Conference (WCNC), 1314-1319, 2015 and [21] J. Chen, J. Xie, Y. Gu, S. Li, S. Fu, Y. Wan, and K. Lu, "Long-range and broadband aerial communication using directional antennas (ACDA): design and implementation," IEEE Transactions on Vehicular Technology, vol. 66, no. 12, pp. 10793-10805, 2017 with permission from IEEE.

From the Table 7.1, the improvement is quite obvious. The V1 system can communicate no more than 1000M while the V3 system still has 800kbps at 5000m. At 300M and 1000M distance, 5G band shows its excellent throughput which is about 2 times at 300M and 1.5 times at 1000M. At 3000M and 5000M, the Nano Station 2 has too weak signal to connect a laptop, so the measurement cannot continue. The Nano station2 has typical 10 dB gain patch antenna while LOCO M5 has 13dB; therefore, the transmission of LOCO M5 is much further.

CHAPTER 8

CONCLUSION AND FUTURE WORKS

The drone-carried WiFi system have been studied throughout my doctoral research. It has been evolved from communication solution to integration of control, computation, communication, and networking. In this 4 years, quite a lot of efforts are made to make the project more reliable, more flexible, and more powerful. Benefitted from technologies advancement, new generation sensors, parts, and products are adopted during the project in last four years. So far, the Jetson TX2 core system is the most advanced platform to provide high quality real time video, thermal video transmission and target detection/tracing. The latest directional antenna Nano LOCO M5 provide premium performance to serve long distance communication. The 300M, 1KM and 3KM test results are very impressive to support broadband WiFi service and video service. By installing on advanced drones like DJI Matrice100, the whole system will be smart and reliable enough to perform various real-world tasks.

In the future, based on this platform, there'll be more application be developed to implement specific tasks. The TX2 hardware platform is built with high efficiency and performance, the application can be flexibly adjusted to meet requirements for different scenarios. The reconfigurable of the system means the system can be used in different scenarios by combining different hardware and software modules so that the system can maximize its performance in different application settings. For example, in the WiFi service application, most modules inside of TX2 will be turned off to save power to extend service time; therefore, the TX2 is only used to exchange GPS data and provide WiFi service. While in the task of rescuing survivors, target detection and tracing modules will be turned on to transmit video back to command center. To further advance the propose system, more field tests and measurements are needed to provide inputs for continuous improvement.

APPENDIX

PERMISSION TO REPRODUCE

Appreciation is extended to the Institute of Electrical and Electronics Engineers (IEEE) for permission granted to papers: [5] Y. Gu, M. Zhou, S. Fu, and Y. Wan, “Airborne WiFi networks through directional antennae: An experimental study”, IEEE Wireless Communications and Networking Conference (WCNC), 1314-1319, 2015 and [21] J. Chen, J. Xie, Y. Gu, S. Li, S. Fu, Y. Wan, and K. Lu, “Long-range and broadband aerial communication using directional antennas (ACDA): design and implementation,” IEEE Transactions on Vehicular Technology, vol. 66, no. 12, pp. 10793-10805, 2017.

Appreciation is also extended to American Institute of Aeronautics and Astronautics (AIAA) for permission granted to paper: [22] J. Xie, F. Al-Emrani, Y. Gu, Y. Wan, and S. Fu, “UAV-carried long-distance Wi-Fi communication infrastructure”, AIAA Infotech@ Aerospace, 2016.

List of Reproduced Contents

REFERENCES

[1] H. Menouar, I. Guvenc, K. Aklaya, A. S. Uluagac, A. Kadri, and A. Tuncer, “UVA-enabled intelligent transportation systems for the smart city: applications and challenges,” IEEE Communication Magazine, pp. 22-28, March, 2017.

[2] B. A. White, A. Tsourdos, I. Askokaraj, S. Subchan, and R. Zbikowski, “Contaminant cloud boundary monitoring using network of UAV sensors,” IEEE Sensors Journal, pp. 1681-1692, October 2008.

[3] A. Merwaday and I. Guvenc, “UAV assisted heterogeneous networks for public safety communications,” in Proceedings of IEEE Wireless Communications and Networking Conference Workshops (WCNCW), pp. 329-334, 2015.

[4] D. Athukoralage, I. Guvenc, W. Saad, and M. Bennis, “Regret based learning for UAV assisted LTE-U/WiFi public safety networks,” in Proceedings of IEEE Global Communications Conference (GLOBECOM), pp. 1-7, 2016.

[5] Y. Gu, M. Zhou, S. Fu, and Y. Wan, “Airborne WiFi networks through directional antennae: An experimental study”, IEEE Wireless Communications and Networking Conference (WCNC), 1314-1319, 2015.

[6] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, “Sensor planning for a symbiotic UAV and UGV system for precision agriculture,” IEEE Transactions on Robotics, vol. 32, no. 6, pp. 1498-1511, 2016.

[7] M. Tortonesi, C. Stefanelli, E. Benvegnu, K. Ford, N. Suri, and M. Linderman, “Multiple-UAV coordination and communications in tactical edge networks,” IEEE Communication Magazine, vol. 50, no. 10, pp.48-55, 2012.

[8] E. Yanmaz, R. Kuschnig, and C. Bettstetter, “Achieving air-ground communications in 802.11 networks with three-dimensional aerial mobility,” in Proceedings of IEEE INFOCOM, pp. 120-124, 2013.

[9] Y. Zeng, R. Zhang, and T. J. Lim, “Throughput maximization for UAV-enabled mobile relaying systems,” IEEE Transactions on Communications, vol. 64, no. 12, pp. 4983-4996, 2016.

[10] G. Zhang, Q. Wu, M. Cui, and R. Zhang, “Securing UAV communications via trajectory optimization,” in Proceedings of IEEE Global Communication Conference (GLOBECOM), 2017.

[11] A. Agogino, C. Holmes Parker, and K. Tumer, “Evolving large scale UAV communication system,” in Proceedings of the 14th annual conference on Genetic and evolutionary computation, pp. 1023-1030, 2012.

[12] A. Hernandez, C. Copot, R. De Keyser, T. Vlas, and I. Nascu, “Identification and path following control of an AR. Drone quadrotor,” in Proceedings of 17th

International Conference System Theory, Control and Computing (ICSTCC), pp. 583-588, 2013.

[13] C. Ashokkumar and G. W. York, “UAV control and simulation using trajectory transcriptions,” AIAA Modeling and Simulation Technologies Conference, 2016.

[14] F. Chen, R. Jiang, K. Zhang, B. Jiang, and G. Tao, “Robust backstepping sliding-mode control and observer-based fault estimation for a quadrotor UAV,” IEEE Transactions on Industrial Electronics, vol. 63, no. 8, pp. 5044-5056, 2016.

[15] A. Vega, C.-C. Lin, K. Swaminathan, A. Buyuktosunoglu, S. Pankanti, Sharathchandra, and P. Bose, “Resilient uav-embedded real-time computing,” in Proceedings of 33rd IEEE International Conference on Computer Design (ICCD), pp. 736-739, 2015.

[16] R. Om. Prakash and C. Saravanan, “Autonomous robust helipad detection algorithm using computer vision,” in Proceedings of International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 2599-2604, 2016.

[17] B. Wang, J. Xie, S. Li, Y. Wan, S. Fu, K. Lu, “Enabling high-performance onboard computing with virtualization for unmanned aerial systems”, in Proceedings of the 2018 International Conference on Unmanned Aircraft Systems, 2018.

[18] J. Alonso, M. S. Perez, “Phased array for UAV communications at 5.5 GHz,” IEEE Antennas and Wireless Propagation Letters, vol. 14, pp. 771-774, 2015.

[19] M. Khan, K. Heurtefeux, A. Mohamed, K. A. Harras, and M. M. Hassan, “Mobile target coverage and tracking on drone-be-gone UAV cyber-physical testbed,” IEEE Systems Journal, 2017.

[20] M. Schmittle, A. Lukina, L. Vacek, J. Das, C. P. Buskirk, S. Rees, J. Sztipanovits, R. Grosu, and V. Kumar, “OpenUAV: a UAV testbed for the CPS and robotics community,” in Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems, pp. 130-139, 2018.

[21] J. Chen, J. Xie, Y. Gu, S. Li, S. Fu, Y. Wan, and K. Lu, “Long-range and broadband aerial communication using directional antennas (ACDA): design and implementation,” IEEE Transactions on Vehicular Technology, vol. 66, no. 12, pp. 10793-10805, 2017.

[22] J. Xie, F. Al-Emrani, Y. Gu, Y. Wan, and S. Fu, “UAV-carried long-distance Wi-Fi communication infrastructure”, AIAA Infotech@ Aerospace, 2016.

[23] NVIDIA Jetson TX Module. [Online]. Available: https://developer.nvidia.com/embedded/buy/jetson-tx2.

[24] UBIAQUITI Networks. [Online]. Available: http://www.ubnt.com.

[25] Tenda Router. [Online]. Available: http://www.tendaus.com.

[26] DJI. [Online]. Available: http://www.dji.com.

[27] DIJI. [Online]. Available: http://www.digi.com/xbee/.

[28] Atmel. [Online]. Available: www.microchip.com.

[29] HUAWEI. [Online]. Available: www.huawei.com.

[30] R. Li, T. Wu, and M. Tentzeris, "A dual-band unidirectional coplanar antenna for 2.4-5-GHz wireless applications", in Proceedings of 2008 Asia-Pacific Microwave Conference, December 2008.

[31] B. Abhishek, K. Keshav, S. Gautham, D. V. Samuel, and S. R. Nair, "Low cost ROS based semi-autonomous drone with position and altitude lock", in Proceedings of IEEE International Conference on Power, Control, Signals and Instrumentation Engineering, September 2017.

[32] Hyperion. [Online]. Available: https://hyperion-world.com/

[33] GlobalSat. [Online]. Available: https://www.globalsat.com.tw/

[34] Sparkfun. [Online]. Available: http://www.sparkfun.com

[35] Adafruit. [Online]. Available: http://www.adafruit.com

[36] Nikon. [Online]. Available: http://www.nikon.com

[37] Tarot. [Online]. Available: http://www.tarot-rc.com/

[38] FLIR. [Online]. Available: http://www.flir.com

[39] Ti. [Online]. Available: http://www.TI.com