CanSat

Pico Size Artificial Satellite

A Guidebook for Building Successful CanSat Project

University Space Engineering Consortium **(UNISEC)**

Preface

CanSat provides an affordable opportunity for educators and students to acquire basic knowledge of space engineering and to experience the engineering challenges in building a satellite. In CanSat-based space engineering education, the students will be able to design and build a small electronic payload (mission as well as bus systems) that can fit inside a standard drink can (350 ml) or a little larger sizes. The CanSats will be launched by a rocket or balloon and released in the air. Using a parachute, the CanSat will slowly descend back to the ground as it performs its designated mission, i.e., taking pictures and transmitting telemetry. The data generated in CanSat is usually sent to the ground station in real time by RF communication, or is recorded in the memory of the CanSat so that it can be retrieved after landing.

The CanSat-based space engineering education challenges innovative students to get hands-on experience in a space related project during less than one year and with relatively low cost. As a space engineering project, students will get experience from conceptual design, through integration and testing, and actual operation of the system. This will give the students experience of taking part of one whole project cycle within one year or less. One of the major advantages of the CanSat is the very low life cycle cost of the project. Thus, universities could involve more students to space related projects. The CanSat is small, non-orbiting and with limited complexity, but it is still like a "satellite" in terms of many of the challenges real satellites face. Actually, many universities in Japan who obtained skills, know-how and experiences in CanSat later successfully developed and launched real orbiting satellites. We can say that the CanSat is "the starting point for developing satellites."

The structure of the text book introduces CanSat in a simple and clear way for any person who want to acquire knowledge about basic satellite engineering. It covers all the important aspect of CanSat design, fabrication, testing and operations. It highlights also the similarity between the CanSat and the real satellite. The text book is divided into 7 chapters. Chapter 1 introduces the CanSat and its history. Chapter 2 describes the CanSat based space engineering education. Chapter 3 provides a detailed explanation about the i-CanSat kit, a hands-on training kit, and how to assemble, integrate, test and operate it. Chapter 4 provides a detailed explanations about the CanSat mission system design. Chapter 5 presents the details of advance verification tests that should be conducted for CanSat before flight. Chapter 6 introduces the important aspects of field testing. Last but not least, Chapter 7 presents the field works and project self-evaluation after field tests. It is hoped that this text book will provide the reader with a concrete background about satellite engineering and how they can start their CanSat project. Please enjoy reading.

Revision History

Date	Revised by	Version	Changes
August 28, 2016	Yasuyuki Miyazaki	0	English draft version from the original Japanese version including all chapters except chapter 7 with the following missing sections 2.5, 4.1, 4.2, and 4.3.
September 18, 2016	Mohammed Khalil	1	Comprehensive revision for chapter 3 to act as a standalone manual for i-CanSat-6 kit to be used in CLTP7
October 14, 2016	Mohammed Khalil	2	2nd Major revision for chapter 3 based of feedback from CLTP7 participants and instructors
December 13, 2016	Mohammed Khalil	3	Revised English draft version that include all chapters, missing chapters and missing sections from version 0. It includes also the chapter 3 of version 2. Most of graphs and tables are re-produced
July 20, 2017	Mohammed Khalil	4	This revision includes the following: - Revision of Chapter 2 as per Sakamoto-sensei - Revision of Chapter 4 as per Sakamoto-sensei - nclude HD photo in Chapter 6 as per Nagata-sensei - Minor revision for the remaining portions.

Table of Contents

List of Abbreviation

ADM3202	: Integrated Circuit (IC) for General-purpose RS-232 data link.
BAT	: Battery.
bps	: Bit per second.
BTM	: Bottom.
C	: Electric Capacitance value in units of Farads of [F].
CAM	: CAMera.
CAM/OBC	: The terminal on the CAM board of the interface connector between the CAM and OBC boards.
CAM/XBEE	: The terminal on the CAM board of the interface connector between the CAM and XBEE boards.
CANCAM	: Camera used for i-CanSat6.
CLTP	: Cansat Leader Training Program.
COMM	: COMMunication.
CR	: Carriage Return.
D	: Symbol used to abbreviate the Light Emitting Diode (LED).
DH	: XBee Destination address High.
DIP	: Dual In-line Package, may refer to each individual switch.
DL	: XBee Destination address Low.
E	: Electric potential in units of Voltage or [V].
EEPROM	: Electrically Erasable Programmable Read-Only Memory.
GND	: Ground signal.
GPGGA	: Global Positioning System Fix Data.
GPGLL	: Geographic Position, Latitude / Longitude and time.
GPGSV	: GPS Satellites in view.
GPRMC	: Recommended minimum specific GPS/Transit data.
GPS	: Global Positioning System.
GPS_OUT	: Switch that Outputs the GPS data to PC or OBC.
GPS/PWR	: The terminal on the GPS board of the interface connector between the GPS and PWR boards.
GS	: Ground Station.
HDOP	: Horizontal Dilution Of Precision. A measure of the geometric quality of a GPS satellite configuration in the sky.
I	: Electrical current in units of Ampers or [A].
I/O	: Input/Output.
I^2C	: Inter-Integrated Circuit is a multi-master, multi-slave, single-ended, serial computer bus.
ID	: IDentification number.
IF1	: Interface pin number one.
IF2	: Interface pin number two.
IOx/ANx	: Input/Output ANalog signal number x.
J	: Jumper or connector.
LED	: Light Emitting Diode.

LF	: Line Feed
MAX232	: Integrated Circuit (IC) for General-purpose RS-232 data link
MDL	: Middle.
NMEA	: National Marine Electronics Association is a standard for GPS data format.
OBC	: On Board Computer.
OBC/CAM	: The terminal on the OBC board of the interface connector between the OBC and CAM boards.
OBC/USR-1	: The terminal on the OBC board of the first interface connector between the OBC and USR boards.
OBC/USR-2	: The terminal on the OBC board of the second interface connector between the OBC and USR boards.
PAN	: Personal Area Network.
PC	: Personnel Computer.
PC_RXD	: Receive data to PC.
PC_TXD	: Transmit data to PC.
PCB	: Printed Circuit Board.
PGC	: Standard communication clock for PIC programming.
PGD	: standard communication data for PIC programming.
PIC	: Peripheral Interface Controller/ Programmable Intelligent Computer.
PICkit3	: Electrical device allows debugging and programming of PIC.
PRG	: PRogramming.
PWR	: Power board.
PWR/GPS	: the terminal on the PWR board of the interface connector between the PWR and GPS boards.
PWR/USR	: The terminal on the PWR board of the interface connector between the PWR and USR boards.
R	: Electrical Resistance in units of Ohm or [Ω].
RS-232C	: a standard for serial communication transmission of data.
RXD	: Receive Data.
S	: electrical toggle Switch.
SCL	: I^2C CLock signal.
SD	: Secure Digital card.
SDA	: I^2C DAta Signal.
SEP	: SEParation jumper.
STR	: STRucture.
TNC	: Terminal Node Controller, unit to encode transmitted data to fit the transmission protocol or decodes the received data
TTL	: Transistor–Transistor Logic.
TXD	: Transmit Data.
TXD_OUT	: Switch that outputs of the Transmit Data can be set to PC or XBee.
U	: Integrated Circuit (IC) or assembly of electric circuit.
USB	: Universal Serial Bus.
USR	: User board.
USR/PWR	: The terminal on the USR board of the interface connector between the USR and PWR boards.

USR/OBC-1	: The terminal on the USR board of the first interface connector between the USR and OBC boards.
USR/OBC-2	: The terminal on the USR board of the second interface connector between the USR and OBC boards.
VCC	: Designation that refers to voltage from a power supply.
VPP	: Programming mode voltage.
WGS-84	: The World Geodetic System is the spatial reference system of GPS satellites.
WP	: Write Protection .
XBee	: Wireless communication module used of i-CanSat.
XBEE/CAM	: The terminal on the XBEE board of the interface connector between the XBEE and CAM boards.
X-CTU	: Application to pair two XBees.
XTAL	: Crystal oscillator.

1 | Introduction to CanSat

1.1 Introduction

CanSat is a small satellite analog, as shown in Figure 1-1-a, in which all of the components required for satellite-like functions, such as micro-computers, sensors, actuators, RF transmitters/receivers and GPS, are housed inside a 350-ml soda can. In the initial concept, CanSat's size was restricted to "350ml soda can size", but later larger size CanSats that could be suitable for single launch by a ARLISS model rocket (described later) appeared. This larger size CanSat, as shown in Figure 1-1-b, is called "Open Class" CanSat. In this way, now the definition of CanSat in terms of size and weight limitation is not so strict; within something like 350ml soda can or a little larger sizes.

a) A CanSat of 350 ml soda can size.

b) Open class CanSat of 150 mm diameter.

Figure 1-1: Examples of CanSat.

1.2 History of CanSat

CanSat provides an affordable opportunity for educator and students to acquire basic knowledge of space engineering and to experience engineering challenges in building a satellite. The students will be able to design and build a small electronic payload (mission as well as bus systems) that can fit inside a standard drink can (350 ml) or a little larger sizes. The CanSats will be launched by a rocket or balloon and released in the air. Using a parachute, the CanSat will slowly descend back to the ground as it performs its designated mission, i.e., taking pictures and transmitting telemetry. The data generated in CanSat is usually sent to the ground station in real time by RF communication, or is recorded in the memory of CanSat so that it can be retrieved after landing of CanSat.

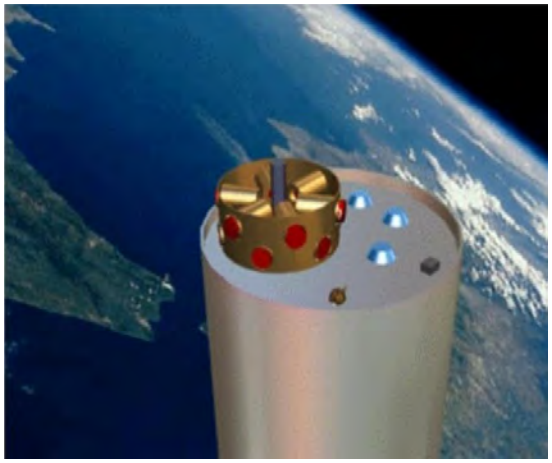
The CanSat-based space engineering education challenges innovative students to get hands-on experience in a space related project during less than one year and with relatively low cost. As a space engineering project students will get experience from conceptual design, through integration and testing, and actual operation of the system. This will give the students experience of taking part of one whole project cycle within one year or less. One of the major advantages of the CanSat is the very low life cycle cost of the project. Thus, universities could involve more students to space related projects. The

CanSat is small, non-orbiting and with limited complexity, but it is still like a "satellite" in terms of many of the challenges real satellites face. Actually, many universities in Japan who obtained skills, know-how and experiences in CanSat later successfully developed and launched real orbiting satellites. We can say that the CanSat is "the starting point for developing satellites."

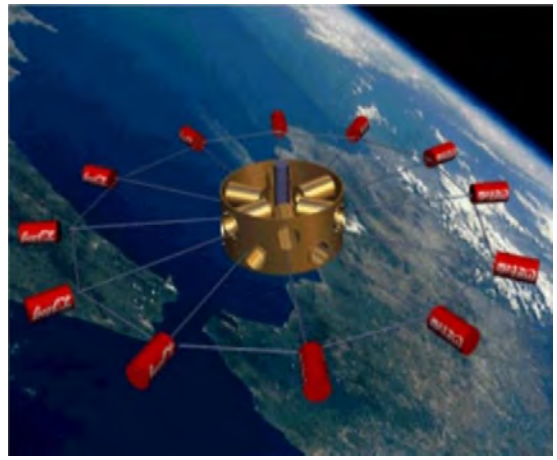
1.2.1 **Birth of CanSat concept**

In November 1998 at the University Space Systems Symposium (USSS) held in Hawaii, Professor Bob Twiggs of Space Development Laboratory of Stanford University at that time, Figure 1-2, suddenly stood up with a Coke-Can in his hand and announced "Let's make a satellite out of this!". This was the moment when the concept of "CanSat" was born. In the initial plan, each university is to develop 350 ml soda can sized pico-satellite, and these satellites are to be launched altogether to the orbit and operated at the next year's USSS, as shown in Figure 1-3. However, the acquisition of launch opportunity for them was found difficulty, and the project was switched to suborbital launch experiment called ARLISS (A Rocket Launch for International Student Satellites) in which CanSats are launched up to 12000 ft (3.6 km) height using US amateur high power rockets developed by AEROPAC, as shown in Figure 1-4. The important requirement for space engineering education is that the fabricated systems should be actually launched and experimented in real environment, even if it is not in Earth orbit. Thanks to the almost voluntary and willing support from a US amateur rocket group AEROPAC, we could obtain valuable opportunity for such real environment experiments.

Figure 1-2: Professor Bob Twiggs.



a) CanSat cluster before deployment.



b) Full deployment of CanSat cluster.

Figure 1-3: Initial CanSat operation concept (1998).

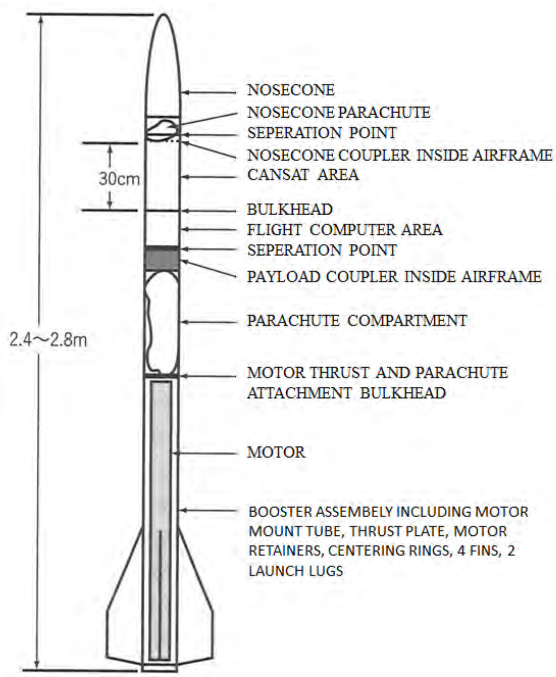


Figure 1-4: AEROPAC high power model rocket.

1.2.2 **ARLISS in Black Desert, USA**

The first ARLISS was held on 11th of September in 1999 in Black Rock Desert in Nevada State, shown in Figure 1-5. University of Tokyo and Tokyo Institute of Technology participated in the experiment with three CanSats each, and Arizona State University and Kennedy Middle School participated from US side. The CanSats, after released from rockets, fly using parachute for about 15 to 25 minutes, during which various satellite related experiments are to be performed. For example, University of Tokyo performed pico-satellite bus experiment including onboard CPU, memory, uplink and downlink, solar power generation and storage, sensors, actuators (reaction wheels), wire cutting mechanism and real time video image acquisition and transmission.

Figure 1-5: Location of Black Rock Desert.

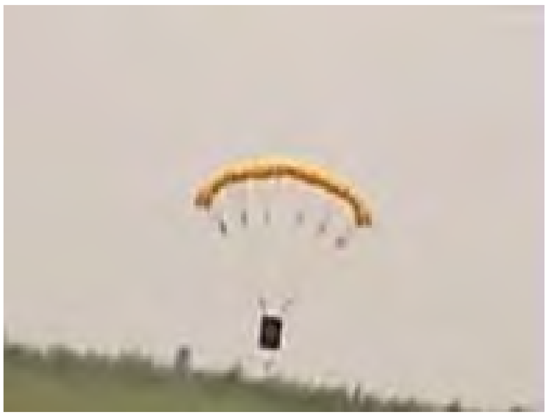
ARLISS has been held annually in Black Rock Desert since 1999, and the number of participating universities and countries grew rapidly. In 2016, thirteen universities from Japan, two universities from South Korea, three universities from USA, one university from Peru, and one university from Egypt attended, with more than 100 students participation. Interested reader can access the ARLISS website at www.arliss.org.

1.2.3 **Comeback competition**

The comeback competition was proposed by Professor Shinichi Nakasuka of University of Tokyo in 2001. In this competition, a CanSat (called "a comebacker" hereafter) with a certain steering mechanism such as parafoil is to, after release in high altitude, come back to a certain target point autonomously without human interaction, and the one which comes nearest to the target wins the competition. Usually the comebacker obtains its position and velocity information using GPS, and the difference of the current position/velocity and the target point's position/direction is feedbacked to calculate the control signal, which is used to pull the one end of the parafoil to change the flight direction. This typical comebacker is called "fly-back type", as shown in Figure 1-6-a, having a parafoil or a fixed wing with control surfaces, but in 2002, a "run-back type", as shown in Figure 1-6-b, appeared which drives back with wheels after landing. In either case, the target position is stored in the onboard memory beforehand in the form of latitude and longitude.

This competition has been providing excellent educational material for satellite system development, because 1) the onboard system should have almost all the satellite bus subsystems, including, onboard computer, memory, communication system, battery, sensors/actuators, and the structure/ mechanism, and 2) the comebacker can only achieve the target if all the elements work well.

The comebacker cannot have any maintenance once it is released in high altitude, and the operations should be fully autonomous. Therefore, every possible situations (such as strong wind, stacking in ground holes, etc.) should be predicted and some countermeasures should be prepared. Moreover, the "competition" type training can give students high motivation and enthusiasm, which further enhance the educational effect. The competition has been held annually since 2001 in USA in ARLISS and the first Japanese domestic competition was held in 2002 using a thermal manned balloon.



a) Fly-back type CanSat.

b) Run-back type CanSat.

Figure 1-6: Different types of CanSats.

1.2.4 **Other CanSat experimental events**

In Japan, Noshiro space event has been annually held in August since 2005, where various types of student experiments such as hybrid rocket launch, model rocket launch and CanSat experiments have been performed. CanSat comeback competition is also organized by UNISEC (University Space Engineering Consortium) in this event, using not rocket launch but a Helium balloon. Some universities also conduct CanSat launch by model rockets or hybrid rockets. In addition, high school students' CanSat competition called “CanSat Koshien" (not a comeback type competition) is also held annually and the winning team of this competition is invited by the sponsor to attend ARLISS. In 2005, at the occasion of International Astronautical Congress (IAC) in Fukuoka, an international comeback competition was held at Kyushu University in Fukuoka, and the first CanSat workshop sponsored by JAXA was held in February 2007 at University of Tokyo.

In Europe, also, several CanSat competitions have been held as ESA is supporting student CanSat activities. Interested reader can access the website at www.cansat.eu. Now many countries start CanSat-based education of space development.

1.2.5 **CanSat Leaders Training Program (CLTP)**

Many Japanese universities have gained experiences and skills of CanSat development through the above mentioned events, with which they started to provide annual CanSat training course for foreign countries since 2011. This program is called CLTP (CanSat Leaders Training Program). In this program, university professors or other similar position persons in other countries who want to start

CanSat education in their home countries can participate. The course is divided into two parts. The first part is online lecture series about satellites systems and CanSats given by Japanese professors in English language for about one month. The second part is a hands-on training for about two weeks in which the participants are invited to Japan at certain Japanese university to assemble, integrate, test, and launch a CanSat kit called “i-CanSat” which is described in details in Chapter 3 | . After completing these two parts, they will be able to teach CanSat development after they go back to their home countries. The first CLTP was held in 2011, and Table 1-1 shows the hosting universities and the participating countries in seven CLTPs.

Table 1-1: Participating Countries in CLTP 1 through 7.

CLTP1 (Wakayama University in Feburary-March, 2011) 12 participants from 10 countries, namely Algeria, Australia, Egypt, Guatemala, Mexico, Nigeria, Peru, Sri Lanka, Turkey (3), Vietnam.
CLTP2 (Nihon University in November-December, 2011) 10 participants from 10 countries, namely Indonesia, Malaysia, Nigeria, Vietnam, Ghana, Peru, Singapore, Mongolia, Thailand, Turkey.
CLTP3 (Tokyo Metropolitan University in July-August, 2012) 10 participants from 9 countries, namely Egypt (2), Nigeria, Namibia, Turkey, Lithuania, Mongolia, Israel, Philippines, Brazil.
CLTP4 (Keio University in July-August, 2013) 9 participants from 6 countries, namely Mexico(4), Angola, Mongolia, Philippines, Bangladesh, Japan.
CLTP5 (Hokkaido University in September, 2014) 7 participants from 5 countries, namely Egypt, Korea (2), Mexico (2), Mongolia, Peru.
CLTP6 (Hokkaido University in August-September, 2015) 8 participants from 8 countries, namely Angola, Australia, Austria (United Nations), Bangladesh, Egypt, Mexico, Tunisia, Turkey.
CLTP7 (Hokkaido University in September-October, 2016) 8 participants from 7 countries, namely Dominican Republic, Egypt, Magnolia, Myanmar, Nepal (2), Peru, Serbia.

1.3 What is required for CanSat experiment?

The following items should be developed or prepared by the time when the CanSat field experiment is planned.

1.3.1 CanSat

CanSat which meets the size and weight requirements should be developed. In ARLISS, 350ml size CanSat should be the same size as 350ml soda can, and the weight limitation is usually less than 350 grams. Three CanSats can be loaded into a carrier, as shown in Figure 1-7, which will be inserted into the compartment below the nosecone part of the ARLISS rocket. Open class CanSat should be smaller than 147 mm × 240 mm size with weight less than 1 kg, which fits the inner diameter and length of the ARLISS carrier, as shown in Figure 1-7. CanSat can have some flexible appendages such as a parachute, parafoil, antenna, etc. outside of the above envelope, but they should also be stowed into the carrier altogether. Of course, this size and weight requirements will be varied according to the technique of how to lift CanSat up into the sky.

Figure 1-7: Carrier with three CanSat loaded (ARLISS).

Another important requirement for CanSat is the safety. The CanSat which is released in the sky should descent to the ground with a certain, not fast, speed. A parachute, parafoil or other descending velocity reduction mechanism should be attached to the CanSat.

1.3.2 **Ground station**

If you want to downlink the CanSat data in real time to the ground, a ground station should be also developed. Usually the experiment is held in a desert or similar open areas, without power supply in most cases. Therefore the ground station should be portable one which does not require external power supply. Figure 1-8 shows one example of ground station, which was used in ARLISS 1999 by University of Tokyo. It consists of a hand-held Yagi antenna, an amateur transceiver and a personal computer with human interface software. This PC should store the downlinked data. If you want to analyze the current CanSat status in real time and send some commands to CanSat, then PC should have data visualizing software application and command generation function.

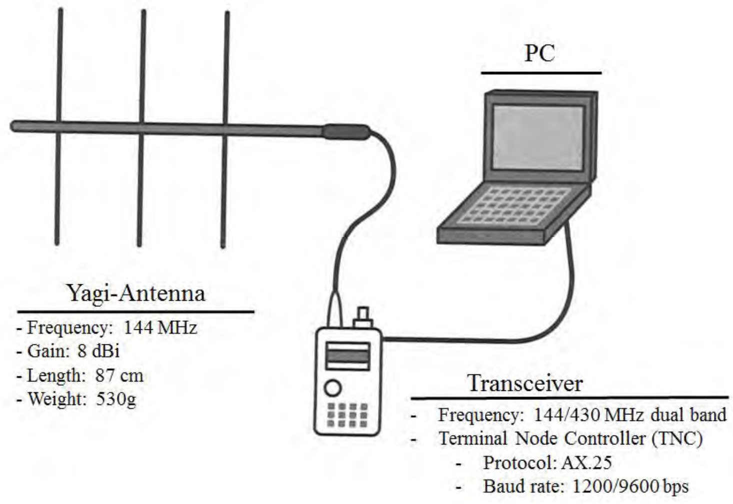


Figure 1-8: Example of portable ground station.

1.3.3 **Way to lift CanSat to a certain altitude**

There are several ways to lift CanSat up to a certain altitude. They can be summarized as follows:

1. **Dropping from building**
   The easiest way is to bring CanSat up to a certain tall building or tower's high floor and drop it. You should be careful so that the CanSat does not hit the building or tower after release.

2. **Tethered balloon or UAV**
   A helium balloon lifts a certain gondola carrying CanSat up into the sky and some remote control mechanism is used to drop CanSat from the gondola. The wires which connect the balloon with the ground should be redundant just in case one of them is cut by heavy load for example by sudden wind, bird attack, etc. a 1 m^3 balloon volume can lift approximately 1 kg payload which consists of gondola, CanSat, and the wires. In Figure 1-9, the bottom door of the gondola can be got open by a servo motor controlled by a remote Radio controller. Using UAV (Unmanned Air Vehicle) to lift CanSat into the sky is a similar method.

3. **Rocket launch**
   A rocket launches a CanSat and releases it in the sky. Figure 1-10 shows the operation performed by ARLISS rockets. Rocket launch provides large vibration and acceleration load and release from rocket usually uses pyro mechanism giving CanSat a large shock load. The CanSat should be tolerant against such loads, which is very similar situation with a real satellite in a launch rocket.

4. **Human flight**
   A man with a CanSat rides a certain flying machine for example airplane, helicopter, thermal balloon, etc and drop CanSat from it. A comeback competition held in Itakura, Japan in 2002

used a manned thermal balloon and CanSats were released at 500 m altitude, as shown in Figure 1-11.

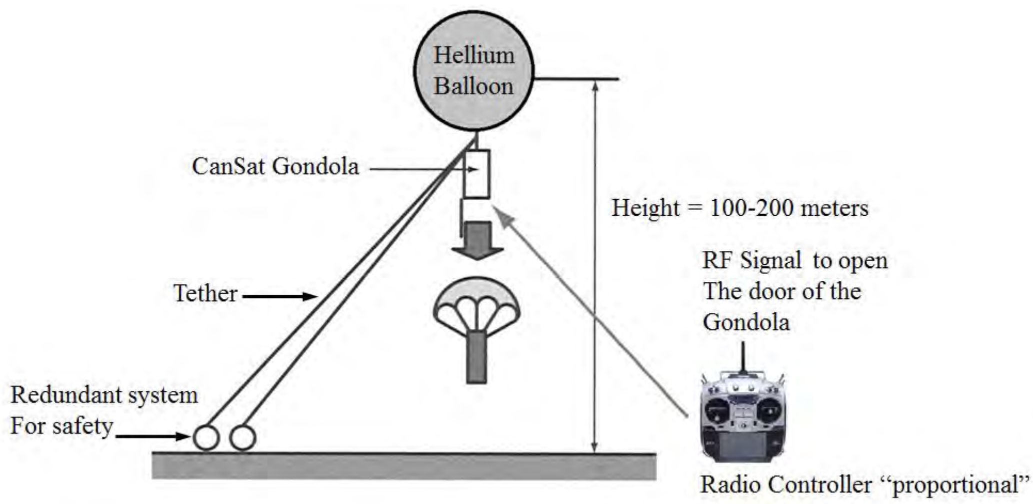


Figure 1-9: Helium balloon method.

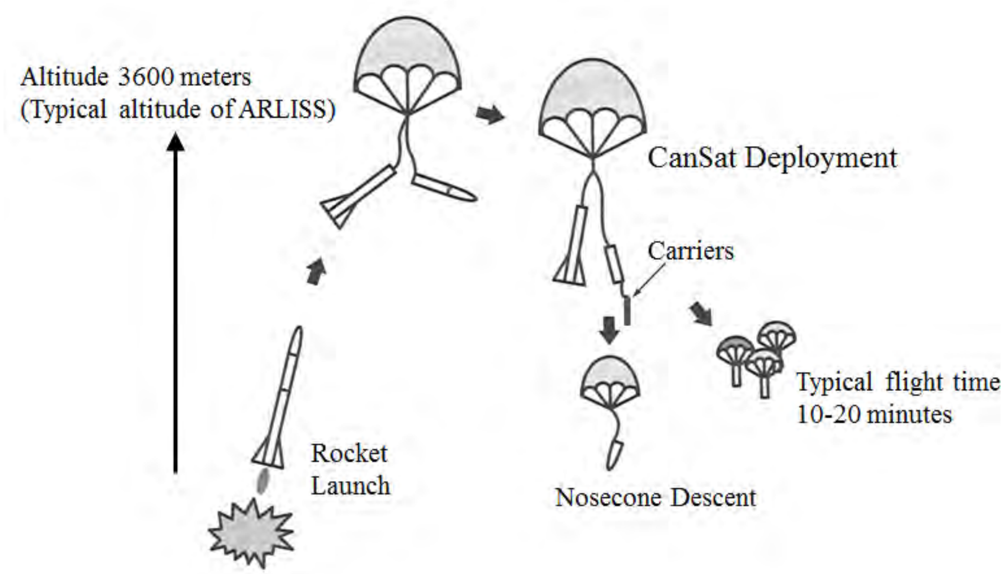


Figure 1-10: Rocket launch method (ARLISS case).

Figure 1-11: Thermal balloon method, Itakura, Japan 2002.

1.4 CanSat missions

Creation of missions is one of the most important step of CanSat project. The following notes should be kept in mind when you create the missions.

1. You should aim at interesting missions, but they should be realizable (within your ability) missions.
   a) The development should be finished within the lime limit (by ARLISS, for example), considering human resource and expertise
   b) You also should consider what you can do with the available parts, equipment, facility and components
2. If you are novice, please start with a very simple mission. The most important thing is to make what really works as designed. It is advised to start with low level and gradually set higher targets.
3. Usually development tasks, for example design, fabrication, check, test, modification, etc., requires more than twice as long time as expected. So you should add sufficient margin in the schedule.
4. You should verify each part of your design with some tests. Don't employ such design which cannot be tested before actual flight experiment.

If you participate in CanSat development for the first time, it will be very difficult to find out what is realizable and what is not within the available time and resources. Therefore one of the recommended strategy is to first start with very simple and easy mission and gradually add additional missions as your project goes on and you feel that you can do more.

The following functions are rather easy to achieve in CanSat. One suggestion is to combine these functions to create your own missions.

a. **Sensors**
   Temperature, pressure, GPS, accelerometer, sun light, gyro, ultra violet, sound, infra red, etc.
b. **Actuators**
   Motor, Nichrome wire to cut something, magnet, utilization of shock of landing, spring, gravity, etc.
c. **ON/OFF switching**
   Triggered by command uplink, timer, other events, etc.
d. **High level actions**
   Guidance/control with GPS, e.g., comeback, image capture, still or motion, LED, moving after landing, etc.
e. **Communications**
   Between many CanSats, CanSat and ground station, etc.

2 | CanSat Based Education

2.1 Relationships between satellite and CanSat

CanSat can be considered "a very small and simple satellite", as it has similar functions with satellites and operates away from the human operators. This fact makes the education based on CanSat very suitable as a first step training towards the real satellite development. On the other hands, CanSat is different from real satellites in several aspects. Therefore it is very important to understand what are similar points between CanSat and satellites and what are the differences, and to take these factors into consideration when learning CanSat. The followings are common features between CanSat and a real satellite

2.1.1 System configuration

Figure 2-1 shows the configuration of CanSat and a satellite system. Both of a satellite and CanSat consist of C&DH system (Command and Data Handling or information management system), mission subsystem, power subsystem, communication subsystem, structure and mechanical subsystems, thermal subsystem and attitude or orbit control subsystem. Only the differences are that CanSat usually does not require thermal control though a satellite needs it, and that some CanSat does not require communication system because acquired data by CanSat can be retrieved after landing of the CanSat. It is also common that whether attitude/orbit control system is required or not depends on the requirement from mission subsystem.

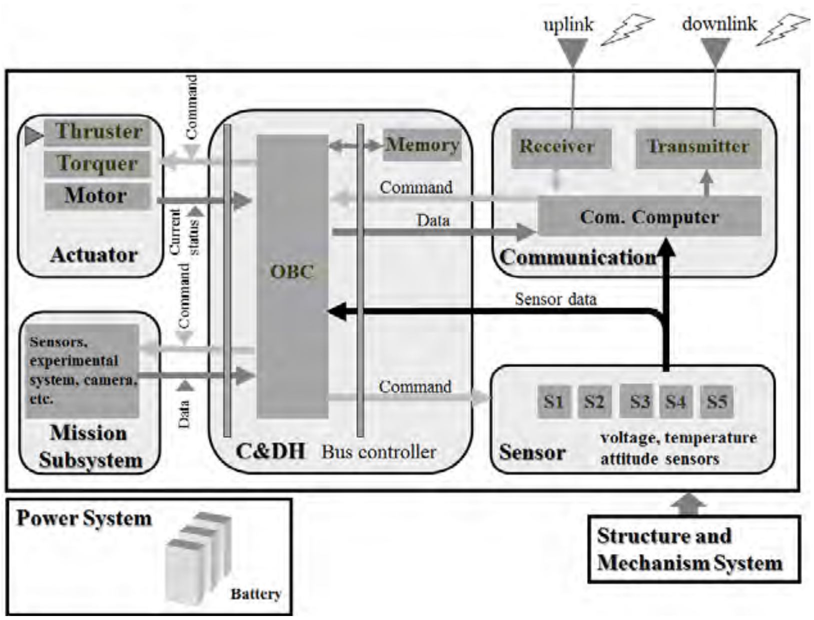


Figure 2-1: CanSat/Satellite system configuration.

2.1.2 **Non-repairable system**

Both of CanSat and real satellites are "non-repairable system"; they cannot be repaired once they are released from a rocket or dropped from a balloon. This feature makes the development of satellites an extremely difficult problem, and such difficulties and how to prepare countermeasures can be learned and trained even in CanSat project. The important concept here is to make the system "robust to disturbances and fault tolerant", whose mechanisms should be implemented in the design and verified by tests even in CanSat.

2.1.3 **Importance of ground operations**

As CanSat and satellites should remotely be operated, training on system development of a ground station as well as ground operation are very important for both. First step of the ground operation is that the human operators on the ground understand the current status of the satellite. In order to do so, telemetry should be designed, i.e., what should be downlinked to the ground station. As the communication speed from satellite to the ground is limited, the information to be downlinked should be carefully selected. And the downlinked data should be displayed on the screen so that human operators easily grasps what happens in the satellite. And sometimes quick uplink command should be generated, verified and uplinked to the satellite with the help of ground station computer. These cycles can also be learned in a CanSat project, though the ground station for CanSat, as shown in Figure 1-8, will be much simpler than ground stations for satellites.

2.1.4 **Development process**

Satellite development usually takes phased approach, including conceptual design with BBM (Bread Board Model) development, PDR (Preliminary Design Review), detailed design with EM (Engineering Model) development, CDR (Critical Design Review), PFM (Proto-Flight Model) development and pre-launch operation, as described in section 2.6. This process can also be simulated to some extent in CanSat project.   High level project management is required for both.

2.1.5 **Important mission creation and definition**

Mission creation and definition is one of the most crucial task for satellites. Meaningfully interesting or advanced yet realizable mission should be carefully designed for both of satellites and CanSat. Not only conceptualization of mission, but also detailed definition of mission requirements and mission sequence is very important.

2.1.6 **Launch environment**

When CanSat is launched by a rocket, similar environment including acceleration, vibration, and shock as real satellite case are exerted on CanSat, and how to make the system tolerant against such environment can be trained in CanSat. Vibration or shock tests are required for CanSat in this case, and how to do such tests and how to make feedback from the test results can also be learned.

On the other hand, the following points are different between CanSat and satellites. These different parts should be additionally learned when you develop real satellites after CanSat.

a) Space environments such as micro-gravity, radiation, vacuum, thermal, ultra-violet, atomic oxygen, etc. are unique for satellites. Such ground tests as thermal test, vacuum test, thermal vacuum test, radiation test are not required for CanSat.
b) Missions which can be performed in CanSat are subset of missions to be done in satellites. The mission period is also usually very short for CanSat, say, several seconds to several minutes. Therefore the missions for CanSat should be such ones that is meaningful even for such short time.
c) Thermal subsystem, which is one of the very important subsystems for a satellite, is usually not required for CanSat.
d) Solar power generation with solar cells, which is one of key parts for satellites, usually is not required for CanSat, because the mission period for CanSat is very short.

The following points are favourable for CanSat, as compared with real satellites.

a) CanSat can usually be retrieved after the experiment, which means that the cause of failures or effect of launch environment or operation can be analysed after the experiment, whose results can be effectively used for the modifications of the same CanSat or the following projects.
b) The system of CanSat is very simple as compared with a satellite, which makes the development cost and time very low and short and makes the system configuration understandable even to novices. Therefore the CanSat projects can be iterated many times within a certain budget and time period.

2.2 What you can learn in CanSat?

As inferred from the previous section, you can learn in CanSat project how to develop a real satellite in a simplified form. What you can learn in CanSat project can be summarized as follows:

1. Mission creation, requirement definition and sequence generation.
2. Satellite architecture design.
3. System analysis (power/weight budgeting).
4. Subsystem design and fabrication.
5. Development process (BBM/EM/FM, Design Review) and Project Management.
6. Assembly, Integration and Test (AI&T).
7. How to do “Field Test” (rocket or balloon, etc.).
8. Ground operation (uplink/downlink/display).

In the following section some explanation about "power budgeting" in point number 3 above for real satellite, as shown in Figure 2-2, and CanSat, as shown in Figure 2-3, is described.

The first task for both cases is to summarize the required power for each component and add them according to the schedule of usage of each component. As a results, the total "Power consumption" line is generated as in shown in Figure 2-2. In a real satellite case, the solar cells generate power during sun light, as shown by the grey line in Figure 2-2, and the difference between power consumption and power generation changes the battery DOD (Depth of Discharge). The solar cells area should be designed so that the battery DOD after one cycle is the same as the DOD at the starting point of this cycle, and the battery capacity should be designed so that the battery maximum DOD is about 20-30% of the full capacity.

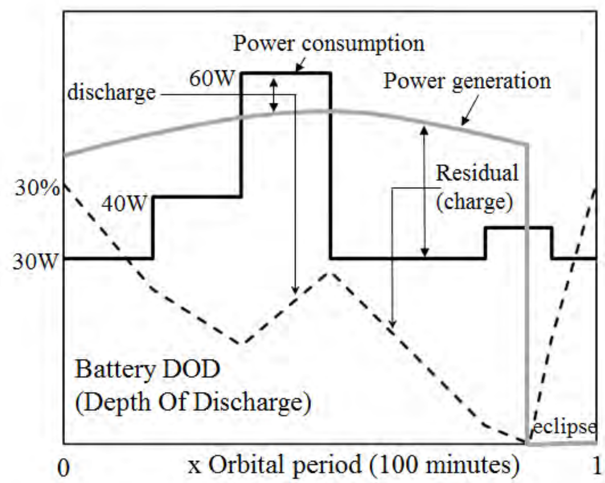


Figure 2-2: Power profile for real satellite.

On the other hand in CanSat case, the power budgeting is very easy. Each component power consumption is accumulated for the mission time period based on power profile as shown in Figure 2-3, and the total required power (with some margin) should be provided by the battery. The required capacity of the battery is estimated in this way.

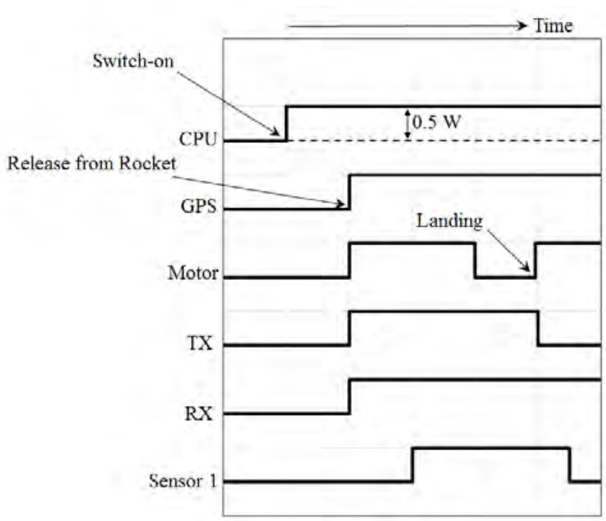


Figure 2-3: Power profile for CanSat.

2.3 Various levels of training

There are many levels of education using CanSat, and the educator should carefully select the options of CanSat based education, considering what kind of skills should be trained and what is the current level of skills and knowledge of the trainees. Table 2-1 summarize several options and Table 2-2 describes the educational effects expected in each option for the different skills to be trained.

Table 2-1: Several options of education using CanSat.

1)	Assemble “kit” with fixed mission, do ground test and launch/balloon experiments. 1.1) Add original mission with new components.
2)	Create mission, obtain (purchase) subsystem components, do ground tests and launch/balloon experiments. 2.1) Design/fabricate some components. 2.2) Design/fabricate all the components.

Table 2-2: Educational effects of each option of CanSat based learning.

Level	Mission Creation	Architecture Design	System Analysis	Subsystem Design	Project Management	AI&T
1)	-	-	-	-	s	s
1.1)	s	s	s	-	s	s
2)	L	L	L		L	L
2.1)	L	L	L	s	L	L
2.2)	L	L	L	L	L	L

AI&T: Assembly, Integration and Test, **s**: Small effect, **L**: large effect

The lowest level, level 1) of education is such option that a certain "kit of CanSat" is purchased or prepared with a fixed mission, and the trainees only assemble and integrate the CanSat as designed and make some flight tests. This is the easiest for the educators, but the trainees do not get training for many important skills and minor educational effect can be expected even for project management and AI&T.

Variation 1.1) of this option using "a kit" is that the only mission part is created, designed and fabricated by the trainees by themselves, which add some educational effects in mission creation, architecture design, system analysis. This option has good educational effect with reasonable cost and time, but the educator should prepare parts and machines to be used for fabricating the mission subsystem which cannot be exactly predicted beforehand, and some level of knowledge to educate mission creation, electric circuit board design and making a certain structure to be used for mission subsystem is required.

In level 2), where no kit of CanSat is used, architecture design and system analysis will be more difficult. If the components (i.e., sensors, actuators, battery, CPUs, transmitters or receivers) are not designed and only purchased from some vendors, then the effort can be focused on system level design and analysis as well as AI&T, and so good educational effects are expected for these skills. However, the skills and knowledge required for educators are far deeper and wider than level 1). In the variation 2.1) where some (not all) components are designed and fabricated by the trainees by themselves, then the skills on subsystem design can be obtained additionally. If all the components are designed and fabricated as in option 2.2), full educational effects for all the skills can be resulted, but the required efforts and skills for educators would be very large.

Considering these pros and cons, the recommended options for CanSat-based education is 1.1) or 2.1) in Table 2-2. We suggest that the educators or developers should choose from these two options considering the available parts or machines, skills and knowledge of the educator and allowable cost and time. In 2.1) case, the subsystem to be designed and fabricated should be in such areas that the educators or developers have sufficient knowledge for design and analysis.

2.4 CanSat system configuration and subsystems

Typical CanSat overall system configuration is depicted in Figure 2-1, which is almost the same as the satellite system configuration except for thermal control subsystem and solar cells. CanSat system consists of the following subsystems, as listed in Table 2-3. The subsystems other than the mission subsystem are called "bus subsystem." "Sensors" and "actuators" are usually categorized in satellites as certain parts of mission subsystem, C&DH subsystem or Attitude/orbit Control Subsystem (ACS), but it would be more convenient to categorize them in an independent subsystem in CanSat case. Attitude and orbit Control Subsystem (ACS) is usually not implemented in CanSat and so deleted from Table 2-3. If the created missions require such subsystem, it must be added.

Table 2-3: CanSat subsystems.

1)	Mission Subsystem
2)	Command & Data Handling Subsystem (C&DH): including software
3)	Power Subsystem (battery charge/discharge system)
4)	Communication Subsystem (including antenna)
5)	Sensors
6)	Actuators
7)	Structure & Accessories subsystem (including parachute or parafoils, etc.).
8)	Ground Station

For detail of each subsystem, please refer to chapter 4 | .

2.5 Teaming

If not only one person but a team with some members collaboratively proceed with CanSat project, then the task assignment to the team members or "sub-team formation" should be carefully designed. There are two directions of sub-team formation which means each team member should do two types of tasks; that is, "subsystem development" and "administrative task".

2.5.1 Sub-team formation according to subsystems

One sub-team is organized for each subsystem as listed in Table 2-3. The number of sub-team members should be decided considering the predicted workload required for each subsystem development. For example, 1) Mission Subsystem and 2) C&DH Subsystem will require larger workload than other subsystems. If the total number of team members are small, then one sub-team is assigned development of several subsystems altogether. For example, 4) Communication Subsystem and 8) Ground Station Subsystem can be combined, and 5) Sensors can be combined with 2) C&DH Subsystem.

2.5.2 Sub-team formation according to administrative work

The following administrative tasks, as listed in Table 2-4, should be done by someone in the team. In real satellite development in large companies, these tasks are usually performed by persons in the administrative divisions, not by development engineers themselves. However in CanSat project, these tasks also should be done by the development engineers, which also provides good education to the trainees.

Table 2-4: Administrative tasks typically required for CanSat project.

a)	Project Manager (PM)
b)	Sub Project Manager (sub PM)
c)	Financial Management (funds raising, budget management, etc.)
d)	Parts/Components Search and Purchase
e)	Documentation and Data Control (internal and external web pages, etc)
f)	Outer Relationships & Promotion (permission, regulations, outreach, etc.)

2.6 Systems engineering

The system engineering discipline can be applied and learned from the CanSat project. It is recommended that developer should be versed in “Systems Engineering” to take holistic approach to understanding the CanSat project. The definition of “Systems Engineering” is diverse. At first, developers might want to start from referring to the basic explanation in the INCOSE Systems Engineering Handbook, International Council on Systems Engineering (INCOSE). Interested reader can access the website of INCOSE (http://www.incose.org/) where the systems engineering handbook is available. In this section, the process to apply systems engineering to the CanSat project is introduced.

2.6.1 Systems engineering processes

To assure mission success of the CanSat project, developers must carry out numerous activities. As it turns out, they can be categorized mainly into three processes. These processes can be defined as follows:

2.6.1.1 System design

First of all, developers must portray the stakeholders. They are people/organizations who affects or influenced by the project. Secondly, developers should extract the requirements from the project needs. Thirdly, to specify the overall function and specifications, it is necessary to decompose the system into smaller individual parts. It is crucial not to forget clarifying the interface between each consisting parts and define the specification in detail.

2.6.1.2 Verification and validation

Developers must guarantee if the systems engineering activities are conducted in accordance with the systems requirements. Also, developers should confirm whether the system design meets the demands of the customer.

2.6.1.3 Systems engineering management

The management engineering processes in a project are used to establish and evolve the project plans, to execute the plans, to assess actual achievement and progress against the plans and to control execution through to fulfilment. Individual technical management processes may be invoked at any time in the life cycle and at any level.

To satisfy the quality, cost and schedule of the system development, system engineers or developers ought to carefully conduct, evaluate and improve each tasks.

The above mentioned three processes can be depicted by a process model diagram called the "V-diagram" shown in Figure 2-4. The left-hand side of the diagram shows how the requirements are broken down into smaller pieces. On the other hand, the right-hand side displays how subsystems are integrated into systems via test and assembly procedures. The dotted lines which connects the system design and verification and validation in each level, system, subsystem and component, express the test plans.

Developers can be flexible in defining the V-diagram. First, if developers represent "the overall Cansat project" as the system, then both "the CanSat" and "the ground station" can be framed as the subsystems. Second, if the developer distinguish "the CanSat" as the system, C&DH system, EPS system and communication system then become the "subsystem". In a similar fashion, the battery, radio communication modules and GPS receivers are assorted to the "component".

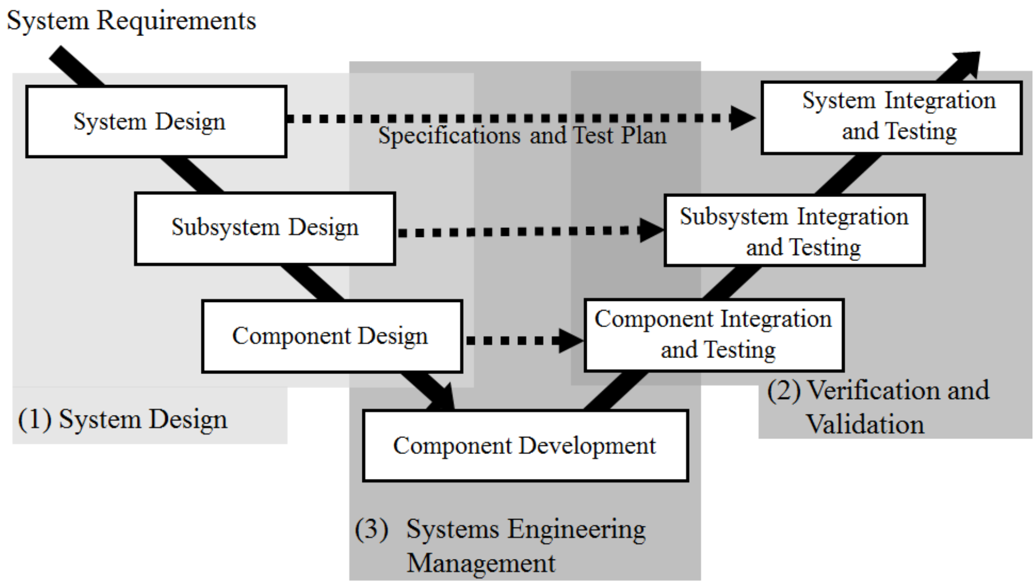


Figure 2-4: System engineering V-Diagram.

2.6.2 **Application to CanSat project.**

In case of applying Systems Engineering to the Cansat project, it is supposed that the system developer follows the following steps:

2.6.2.1 Mission Definition

First, it should be announced what the team wants to conduct by the CanSat project in a broad statement. The statement becomes the mission definition. At first, individual opinions of the team members might be consolidated. In the end, the leader must coordinate the team goal.

2.6.2.2 Requirements Analysis

After the mission definition is set, the mission requirements is clarified. Decomposition of system is conducted and extraction of requirements are carried out for each parts.

Usually in the next step, the method to satisfy the system requirement is chosen. This process is so-called the "System Design". In this course, the subsystem and their requirements are defined.

In turn, the components selection is conducted, which will consist the subsystem and outline their requirements. In the left-hand side of the V-diagram, if the system engineer or developer terminates extracting the requirement in a certain level, the immediate bottom level can be designed. Ideally, the design process is conducted from the top level and flows down to the bottom. However, in reality, that is not always the case.

In requirement analysis, one standard is to categorize requirements into six categories as listed in Table 2-5. In this case, it is important to specify the boundary of each system/subsystem. It is also necessary to characterize each requirement. In this process, it is critical to compose a specification.

Table 2-5: Categories of system requirements.

1.	Function Requirement: Requirement to specify the function to meet the system goal.
2.	Specification Requirement: Requirement to specify the specification to meet the system goal.
3.	External Interface Requirement: Requirement to comply with the external system.
4.	Environment Requirement: Requirement to meet the operation.
5.	Resource Requirement: Requirement to fulfil the system resource.
6.	Physical Requirement: Requirement such as the volume, size and weight of the system.

Requirement specifications are due to change during development phase. According to mission modification, there will be times when requirements increase or decrease. If the schedule is behind, system engineer might consider deleting requirements from the initial specification. On the opposite, when the project is flowing perfectly, the development team might become eager to add new requirements.

2.6.2.3 Architectural Design

In architecture design, the system function and specification are allocated to parts consisting the system. In parallel, the interface among components is defined.

In case of the Cansat project, architecture design is limited to two activities, which is the 1) function design and the 2) physical design. In the former, the system function requirement is divided and illustrated upper level functions into more detail. In the latter, system engineer allocates the functions that were decomposed by the function design to certain components/parts. For example, in such requirement often seen in CanSat as "downlink longitude, latitude and altitude to the ground station during flight," "latitude measurement" and "ground station downlink" are translated to function design. Allocating "latitude measurement" function to the "GPS receiver" becomes the physical design.

2.6.2.4 Verification and validation

"Verification" means "inspecting whether the artifact is designed/manufactured properly according to the upper level specification", while "Validation" means "studying the artifact whether it is designed/manufactured properly reflecting the true voice of the customer". In case of CanSat, the customer and the developer are often the same, so developer is expected to run verification in an intense manner. The purpose of verification is to check whether the "function" extracted from the requirement analysis is working appropriately. In verification, there are methods such as "Inspection," "Analysis," "Demonstration" and "Test."

After learning the assembly of the CanSat in Chapter 3 | , a more close look on Systems Engineering for CanSat is presented in Chapters 4 | and 5 | .

3 | CanSat Assembly, Integration and Testing.

3.1 Introduction

This chapter describes the procedures to assemble, integrate and test (AIT) a CanSat kit. The CanSat kit presented in this chapter is called i-CanSat kit. i-CanSat is an introductory CanSat kit which will provide the new comers to the field of space engineering with the fundamental hands-on experience about the basic satellite subsystems.

i-CanSat originally developed to be used in CanSat Leader Training Program (CLTP). CLTP was launched in 2011 by the University Space Engineering Consortium (UNISEC) to teach the universities and high schools instructors how to teach space engineering using CanSat. The CLTP is offered annually for international participants. The first introduction of i-CanSat was in the third cycle of CLTP (CLTP-3) in July 2012. i-CanSat has been evolved since that time. Table 3-1 summarizes the historical development of i-CanSat. Figures 1-a and 1-b shows the third version (i-CanSat-3) and sixth version (i-CanSat-6) of i-CanSat, respectively. Versions from the first until the fifth are designed to be installed into 350 ml Japanese soda can. The sixth version is designed to be installed into a 500 ml PET bottle or soda can.

Table 3-1: Historical development of i-CanSat.

Version	Date of Release	Brief description
Ver. 1	July, 2012	Designed for CLTP-3. All power is supplied by Li-Po in the camera. Consisting of four boards of TOP, MDL, BTM and OBC.
Ver. 2	June, 2013	Designed for CLTP-4. Consisting of five boards of TOP, MDL, BTM, USR and OBC.
Ver. 3	November, 2013	Two power lines are available. Consisting of five boards of COMM, PWR, USR, CAM and OBC.
Ver. 4	February, 2014	Improved the CAM board.
Ver. 5	July, 2014	Adapted also for 9V dry cell. Improved the CAM board with introduction of CANCAM Ver.1.
Ver. 6	June 2015	Improved for easy assembly. Consisting of six boards of GPS, PWR, USR, OBC, CAM, and XBee. CANCAM Ver.2 was adopted.

This chapter is describing the assembly, integration and testing of the sixth version of i-CanSat (i-CanSat-6) . It was written with the assumption that readers have elementary knowledge in electrical circuits and programming using C-Language. Otherwise, the chapter is self-contained. Section 2 is an introduction to i-CanSat. Basic electrical circuits' theory and terminology will be introduced in section 3. Types of On-Board Computer (OBC) interfaces are introduced in section 4. The basics of soldering technique will be described in section 5. Assembly of the CanSat subsystems and boards are presented in section 6 with clear instructions about soldering the needed parts and modules in each subsystem's main board. Guidelines for simple Assembly, Integration, and Testing (AIT) are presented in section 7.

Section 8 is describing the operation and testing of i-CanSat in details. The firmware development for the i-CanSat is presented in section 9 with sample programs. Section 10 is a brief introduction about launching of i-CanSat and retrieve the flight data. Fabrication of the parachute is described in section 11. General outlines related to the fabrication of Cansat structure is presented in Section 12. Further developments of i-CanSat are discussed in section 13. Last but not least, general precautions about i-CanSat are discussed in section 13. Appendices A, B, and C describe the kit's part list, the sample program, USR board assembly, respectively.

a) i-CanSat, Version 3 (i-CanSat-3) b) i-CanSat, Version 6 (i-CanSat-6)

Figure 3-1: Versions of i-CanSat.

3.2 Configuration Overview

i-CanSat-6 consists of partially cut off circular six Printed Circuits Boards (PCBs). From top to bottom, they are as follows: GPS board for installation of GPS, switches, and indicators, PWR board for installation of on-board battery, USR board for installation of optional modules, e.g., accelerometers, gyroscopes, pressure transduces, …etc., OBC board for installing On-Board Computer (OBC) and memory, CAM board for installing a camera called CANCAM, and XBee board for installing a XBee communication module. Figure 3-2 shows a schematic of the i-CanSat and the locations of each board and electronic parts. All the boards are supplied as a single circuit as shown in Figure 3-3-a and Figure 3-3-b . In all boards, there are preinstalled electronic parts and there are parts need to be soldered. Before start working, the PCBs are detached along the V-cut lines as shown in Figure 3-3-c.

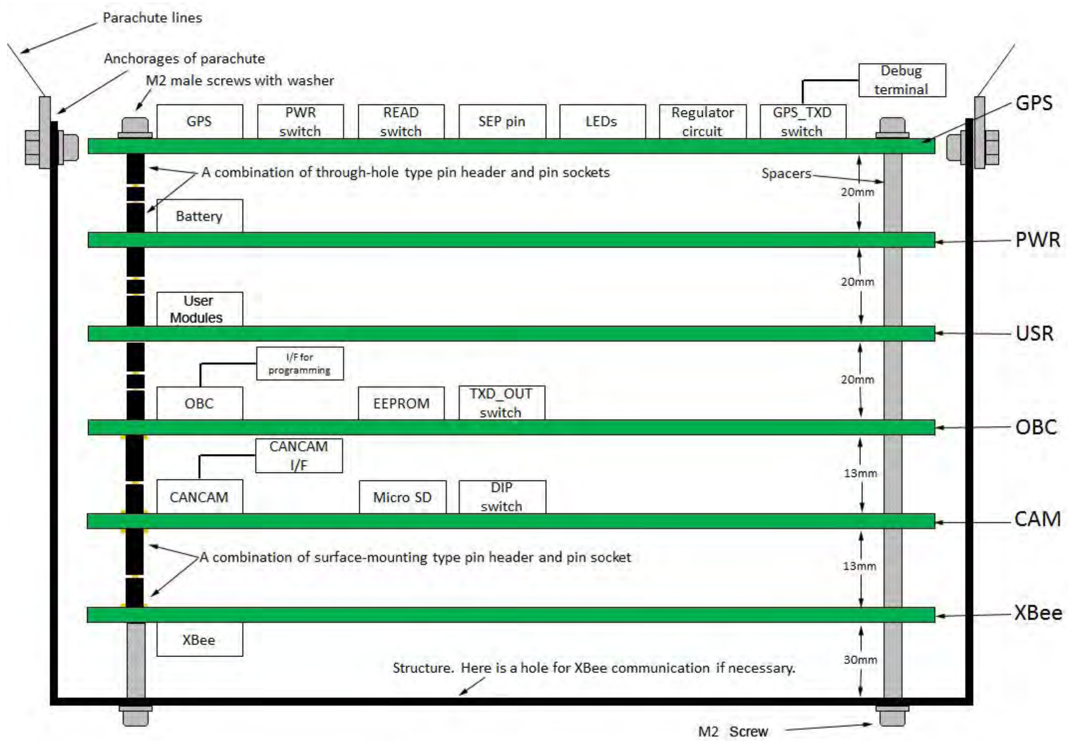
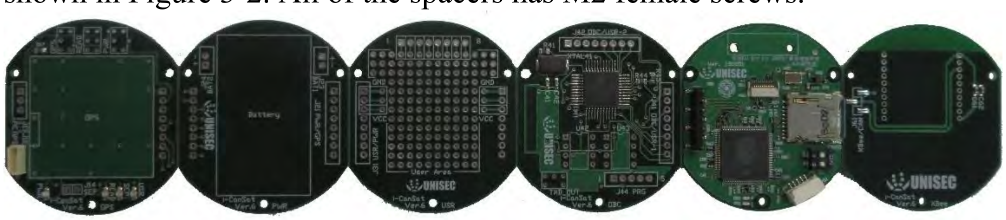


Figure 3-2 Schematic of i-CanSat-6 (not to scale).

The upper surface of the board is distinguished from the lower surface by the written text "i-CanSat Ver.6" beside the board name. This is applied for all boards except the CAM board where the upper surface is distinguished from the lower surface by the SD card slot. It is recommended for the sake of easy handling that the upper surface is marked with tiny colored sticker in all boards during the assembly process, as shown in Figure 3-3-c.

The diameter of all of the boards is 53.4 mm =2,100 mil (1 mil=1/1,000 inch) for the installation into the 350 ml can or 500 ml PET bottle or can. The thickness of all the boards is 1.6 mm, and connection between two boards are formed electrically by the combination of pin header and pin sockets, and mechanically by the spacers of 20 mm or 13 mm in length. The spacer beneath the GPS board is fixed to the GPS board by M2 male screws. For the installation of i-CanSat into the structure of soda can or PET bottle, the spacer beneath the XBee board is fixed to the structure by M2 male screws, as shown in Figure 3-2. All of the spacers has M2 female screws.



a) Upper surafce, all boards attached.

b) Lower surafce, all boards attached.

c) Upper surface, all boards detached.

Figure 3-3: i-CanSat-6 PCBs (from left to right: GPS, PWR, USR, OBC, CAM, and XBee board).

Parachute is fixed on the upper side of i-CanSat, and separation from a career like rocket or balloon is detected by pulling out a jumper, tacked with one of the parachute lines, on the GPS board. The weight of i-CanSat-6, excluding structure, parachute, and the modules on the USR board which are installed by users, is about 150 g. The complete parts list of the kit is described in Appendix A.

3.2.1 **PCB Interfaces**

The main interfaces between the boards and the feature of each board are shown schematically in Figure 3-4. The power lines of the ground (GND) with base potential of 0 V and the power voltage (VCC) adjusted to 3.3V are common to all the boards through the interfaces between the boards. The current goes out from the VCC terminal of the battery and returns to the GND terminal. Commonly the GND side is called “Return” and the VCC side is called “Hot”. The user modules on the USR board are also supplied by the same power lines. User should observe the total required current because the total power consumption, especially current requirements, should not exceed the maximum capacity of both the battery on the PWR board and the voltage regulator on the GPS board. The maximum current for safe operation is 1000 mA or 1A which is decided by the voltage regulator on the GPS board. The features of each board are described as follows:

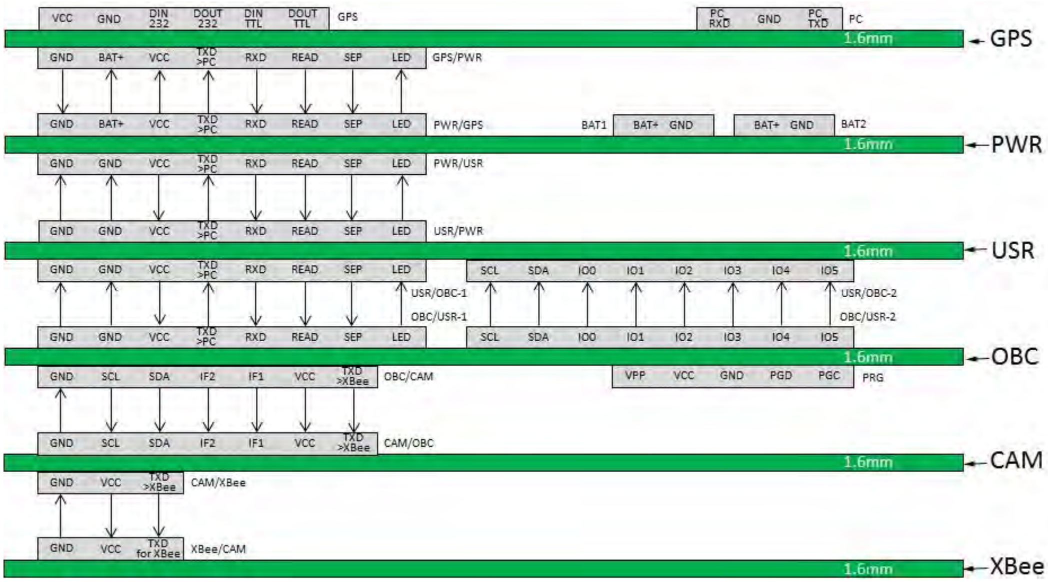


Figure 3-4: i-CanSat boards and PCB Interfaces.

3.2.1.1 GPS PCB

GPS board consists of GPS receiver for positioning, PWR switch, READ switch, SEP jumper, LEDs as indicators, Regulator circuit for voltage regulation, GPS_TXD switch for selecting the destination of GPS receiver output, debug terminal connector which can be connected to PC, and an interface connector (GPS/PWR) for the connection with the lower PWR board.

3.2.1.2 PWR PCB

PWR board consists of Battery and two post connectors, BAT1 and BAT2, and interfaces connectors to the upper GPS board (PWR/GPS) and to the lower USR board (PWR/USR). 006p 9V Alkaline dry cell is suitable for i-CanSat-6, but user can choose his own battery including Lithium battery with a voltage not less than 6 V and a current capacity over 300 mAh. Manganese dry has poor current capacity. Therefore, it is not recommended to use it. Two of the same batteries can be installed on both side the PWR board by double-sided tape, but normally a single battery is sufficient. The hot (+) and return (-) terminals of battery should be matched very carefully with the BAT1 or BAT2 connectors. Any mismatch of power lines can cause damage to the i-CanSat-6

3.2.1.3 USR PCB

USR board is a small universal board with the interval between holes of 2.54 mm or 100 mil. The first interface connector to OBC board is USR/OBC-1. The second interface connector to the OBC board is USR/OBC-2 which has the two I^{2}C bus lines (SCL and SDA) and six Inputs/Outputs (I/Os) ports (RA0/AN0=IO0, RA1/AN1=IO1, RA5/AN4=IO2, RE0/AN5=IO3, RE1/AN6=IO4, RE2/AN7=IO5) for interfacing with sensors that will be installed on the USR board. The third interface connector is to the upper PWR board and is called USR/PWR. As described in section 3.4.6, integrated circuits and modules using I^{2}C bus protocol for communication should have its unique addresses. The EEPROMs or the OBC memory is connected to the OBC through I^{2}C bus. Two chips of EEPROMs can be installed in the OBC board. They have the addresses of 0x00 and 0x01 in hexadecimal or 0 and 1 in decimal. Any module, that uses I^2C bus, will be installed to the USR board should have different address from the EEPROMs.

3.2.1.4 OBC PCB

OBC board consist of PIC microprocessor (PIC16LF877A-I/PT) as On-Board Computer (OBC) and two EEPROMs each has capacity of 1,024 kbit (=128kB). There are two interface connectors to the upper USR board which are OBC/USR-1 and OBC/USR2. The third interface connector is to the lower CAM board (OBC/CAM). There is programming terminal (PRG) installed on the OBC board. This terminal can be connected to the PICKit3 programming kit for the OBC programming. The OBC_TXD switch can change the output of the OBC to be the PC (through wired cable to the GPS board) or the XBee communication module (wireless) installed in the XBee board.

3.2.1.5 CAM PCB

CAM board consists of a camera called CANCAM. The picture files taken by CANCAM is stored into a Micro-SD card in the card slot in the CAM board. Though there is an interface for programming and debugging the CANCAM, it is not used here. The CAM board includes also DIP switch selects the mode of CANCAM. There are two interfaces connectors with the CAM board. The first interface connector is to the upper OBC board (CAM/OBC) and the second interface is to the lower XBee board (CAM/XBee).

3.2.1.6 XBee PCB

XBee board consists of XBee communication module on the lower surface to establish the wireless communication with the ground station or remote PC. There is one interface connector to the upper CAM board which is the XBee/CAM. Since the communication via XBee can affect the GPS receiver greatly, the XBee communication module and GPS receiver should be far from each other in order to minimize the interference between them and that is why the GPS is placed at the top and the XBee communication module is placed at the bottom.

The spacer beneath the XBee board should have at least 30 mm in length to ensure that the XBee antenna is fully extended and not affected by the structure.

3.2.2 Part/Component Identification Number

The electronic components and parts used in the kit are identified according to the following format:

XYZ

Where:

X The type of component or an assembly of components. It can be assigned to any of the following characters:

- C Capacitor
- D LED (Light Emitting Diode)
- J Jumper or connector
- R Resistance
- S Switch
- U Integrated Circuit (IC) or assembly of electric circuit
- XTAL Crystal oscillator

Y The board or subsystem at which the component X is soldered or mounted. Y can have any value from the following number.

- 1 GBS board
- 2 Power board
- 3 User board
- 4 On-Board Computer (OBC) board
- 5 Cancam (Camera) board
- 6 XBee Board

Z is the components number within the board or subsystem.

Example: **J44** is a connector in OBC board and its number is four within the board.

3.2.3 Board Interfacing Connector Naming System

The i-CanSat boards are connected through a number of interface connectors using pin headers and pic sockets as described in section 3.5.5. There are six interface connectors in the i-CanSat-6. The interface connectors has an identification name which has following format:

XXX/YYY – YYY/XXX

Where:

- XXX/YYY is the starting side of the interface at the board XXX and ends at the board YYY.
- YYY/XXX is the ending side of the interface at the board YYY and coming from board XXX.
- XXX or YYY can have the following names:
    - GPS: GPS board
    - PWR: Power board
    - USR: User baord
    - OBC: OBC board
    - CAM: Camera board
    - XBEE: XBee board
- The six interface connectors in i-CanSat-6 are:
    1. GPS/PWR – PWR/GPS (8 pins interface connector)
    2. PWR/USR – USR/PWR (8 pins interface connectors)

3. USR/OBC-1 – OBC/USR-1 (8 pins interface connectors)
4. USR/OBC-2 – OBC/USR-2 (8 pins interface connectors)
5. OBC/CAM – CAM/OBC (7 pins interface connectors)
6. XBEE/CAM – CAM/XBEE (3 pins interface connectors)

3.3 Basics of Electrical Circuits

Knowledge of electrical circuit is a prerequisite for assembling i-CanSat-6. In this section, basic electrical circuit's theory are explained.

3.3.1 **Ohm's Law**

The Ohm's law establishes the relationship between the voltage and current across a conductor. Consider the circuit shown in Figure 3-5. This relationship states that the voltage, *E*, or potential difference across a conductor is proportional to the current, *i*, through it. The constant of proportionality is called the resistance R, mathematically, it can be written as:

$$E = i\,R \qquad \text{Equation 3-1}$$

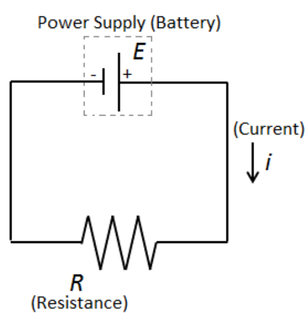


Figure 3-5: An elementary electric circuit.

3.3.2 **Kirchhoff's laws**

In addition to the Ohm's law, there are two important laws to solve electrical circuit problems. They are known as Kirchhoff's laws. The first law is known as the Kirchhoff's current law and the second law is known as Kirchhoff's voltage law.

3.2.1 Kirchhoff's current law

Kirchhoff's current law states that "The sum of current into a circuit node equals the sum of current out of the circuit node". In the node "A" shown in Figure 3-6, the entering value of the current is i_1, this current splits into the current of i_2 and i_3. Mathematically, this can be written as

$$\sum_{NodeA,n=1}^{3} i_n = 0; i_1 = i_2 + i_3 \qquad \text{Equation 3-2}$$

3.2.2 Kirchhoff's Voltage Law

Kirchhoff's voltage law states that "the algebraic sum of the voltage (potential) differences in any loop must equal zero". The loop is any closed circuit. Any complex circuit can be divided into many closed circuits. Consider the circuit shown in Figure 3-6, there are three loops. The upper loop includes the power supply (E), R_1 and R_2. The lower loop includes R_1, R_2, and R_3. The outer loop includes the power supply (E), and R_3. The Kirchhoff's voltage law can be written as follows:

Upper loop:	$E = i_2(R_1 + R_2)$	Equation 3-3
Lower loop:	$i_2(R_1 + R_2) = i_3 R_3$	Equation 3-4
Outer loop:	$E = i_3 R_3$	Equation 3-5

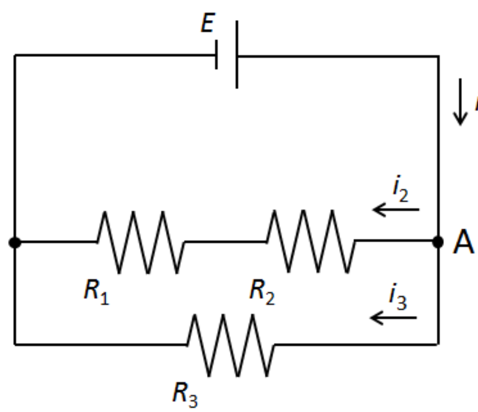


Figure 3-6: Illustration of Kirchhoff's laws.

3.3.3 **Limiting Resistor**

Electric element (voltage element) must have specified voltage to work. On the other hand, there are electric element (current element) must have specified current to work. In current element, if higher current than the maximum rating is supplied, damage of the element may be resulted. To limit the current supplied to an element, current limiting resistor must be connected in series. For example, to apply the current of 10 mA (0.01 A) to an element with the specification of power voltage 5 V, and the specified current 10 mA, insert a current limiting resistor of 5V/0.01 = 500 Ω.

In most of the Light Emitting Diode (LED), the voltage may drop inside the LED. This is called "forward voltage drop". For any LED, the values of the forward current and the forward voltage are writing in the specification sheet. Using these values, the current limiting resistance to be connected to the LED is obtained using the following equation:

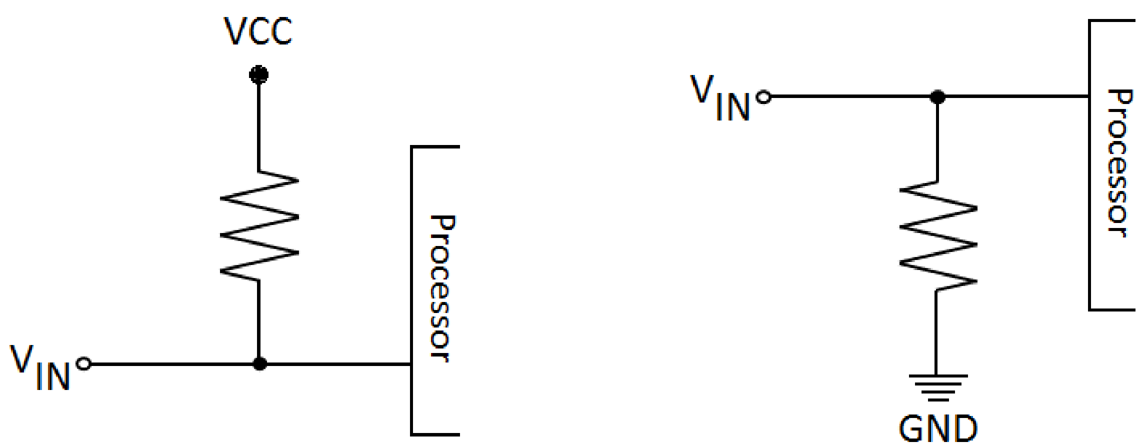
$$\text{Current Limiting Resistance} = \frac{\text{Power Volatge - Forward Voltage}}{\text{Forward Current}} \qquad \text{Equation 3-6}$$

For example, a LED with forward current of 10 mA and forward voltage of 2 V, the current limiting resistance will be (5 - 2) /0.01 = 300 Ω for the power voltage of 5 V.

3.3.4 **Pull-up Resistor and Pull-down Resistor**

For digital circuits and analog circuits, it is desirable to provide a measure to prevent the instability of voltage (potential) at some points. For example, in a digital circuit, the potential stands for the logic or "truth value" and is indicated in 1/0 or H (High level)/L (Low level). High and low are the supplied voltage (VCC) and the zero voltage (GND), respectively. If the potential becomes unstable, the logic will be unstable and result in unintended behavior. Such situation is often caused by fluctuations in the voltage at the terminals of the On-Board Computer (OBC) at startup or the instantaneous fluctuation of voltage. Therefore, the specified logic shall be determined on the circuit before any one of the terminals become unstable. More specifically, a pull-up resistor and a pull-down resistor must be connected to obtain logic 1/High and 0/Low, respectively. The pull-up and pull-down resistors are shown Figure 3-7.

If the voltage at the terminal, V_{IN}, is not specified neither “1” (VCC) nor “0” (GND) and it is required to set the specified logic to "1” at this terminal, adopt the pull-up to connect to VCC through a pull-up resistor, and if it is required to set the specified logic to "0”, adopt the pull-down to connect to GND through a pull-down resistor.



a) Pull-up resistor b) Pull-down resistor

Figure 3-7: Pull-up and pull-down resistors

If the terminal V_{IN} is specified as logic “0” (GND) at the pull-up terminal, current will pass through the pull-up resistor. This current is a loss current and in order to reduce it pull-up resistor should be increased. Same situation occurs if the terminal at the pull-down resistor is set to logic “1” (VCC). The pull-down resistor should have high resistance in order to decrease the loss current.

Because the pull-up or pull-down circuits will form a kind of filter, some time lag may occur to the change of logic at the terminal. To reduce this lag, it will be effective to use the smaller resistance value for pull-up resistor or pull-down resistor. Thus the resistance value shall be properly determined taking into consideration the conflicting conditions of the loss current and the lag time. Commonly the resistances of 10 kΩ or 47 kΩ are used. In I^2C communication, resistance of 2–5 kΩ is commonly used.

3.4 On-Board Computer (OBC) Input / Output Ports

The OBC adopted for i-CanSat-6 is PIC16LF877A microcontroller. PIC16LF877A series normally has five input/output ports. They are used for the input/output interfacing with other devices/circuits. Most of these port pins are multiplexed for handling alternate function for peripheral features on the devices. All ports in a PIC chip are bi-directional. When the peripheral action is enabled in a pin, it may not be used as its general input/output functions. The PIC ports can be configured to interface with others devices/circuits. Not all devices/circuits have same type of interface. The most commonly used types of interface include Digital I/O, Analog I/O, USART, Timer, I²C, and PWM. To configure any port to be input or output, value in its associate "Tri-Sate Enable" or TRIS register must be specified, value "0" for output and value "1" for input. Details about configuration and implementation of the different interface types will be presented in section 3.9. Reader are advised to download and read the specifications sheets of PIC16LF877A microcontroller from Microchip website at www.microchip.com. In the following subsections brief introduction about the most commonly used types of interface is presented.

3.4.1 **Digital I/O**

Digital input/output (I/O) ports can be configured to function as input or output. The port or pin is driven to logic zero or one (0 or +5V) if it functions as output. Microcontroller can detect its logic whether it logic zero or one (0 or +5V) if it functions as input.

3.4.2 **Analog Input**

The PIC16LF877A has a number of analog inputs. They enable the microcontroller to recognize, not only whether a pin is driven to logic zero or one (0 or +5V), but to precisely measure its voltage and convert it into a numerical value, i.e. digital format. The whole procedure takes place in the A/D converter module of the microcontroller. Detailed implementation is presented in section 3.9 with a sample program.

3.4.3 **Timer**

PIC microcontrollers are equipped with one or more precision timing systems known as Timers. They can be used to perform a variety of time precision functions, such as generating events at specific times, measuring the duration of an event, keeping date and time record, counting events, etc. The main component of a timer module is a free running binary counter that increments for each incoming pulse. Since it runs independently, it can count pulses concurrently with the main program execution.

3.4.4 **PWM**

Pulse Width Modulation (PWM) is a technique of controlling the amount of power delivered to an electronic load using an on-off digital signal. The fraction of the period for which the signal is on is known as the duty cycle. The average DC value of the signal can be varied by varying the duty cycle. The duty cycle can be anywhere between 0 (signal is always off) to 1 (signal is constantly on). Suppose, if the signal has +5 V while it is on and 0 V during off condition, then by changing the duty cycle of the signal, any voltage between 0-5 V can be simulated, as shown in Figure 3-8. This method is commonly used for controlling speeds of DC motors, servos and brightness of lamps.

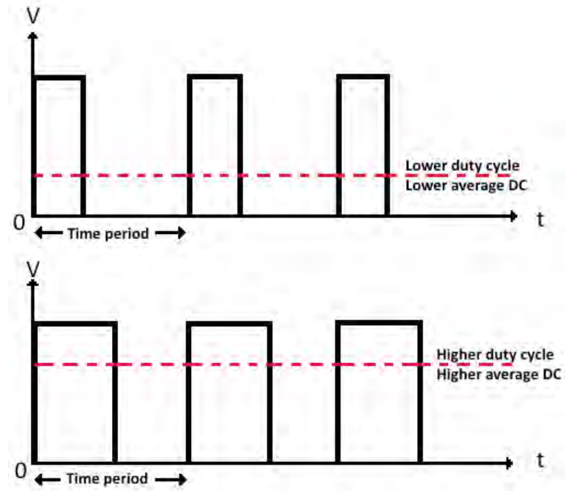


Figure 3-8: Pulse Width Modulation (PWM) signal.

3.4.5 **Universal Synchronous Asynchronous Receiver Transmitter (USART)**

Universal Synchronous Asynchronous Receiver Transmitter (USART) is knows as Serial Communication Interface. Synchronous operation uses a clock and data line while there is no separate clock accompanying the data for asynchronous transmission. Generally, the synchronization operation system is faster for communication. Since there is no clock signal in asynchronous operation, one pin can be used for transmission and another pin can be used for reception. Both transmission and reception can occur at the same time, this is known as full duplex operation. Transmission and reception can be independently enabled. However, when the serial port is enabled, the USART will control both pins and one cannot be used for general purpose I/O when the other is being used for transmission or reception. The USART is most commonly used in the asynchronous mode.

The most common use of the USART in asynchronous mode is to communicate to a PC serial port using the RS-232C protocol. Please note that a driver is required to interface to RS-232C voltage levels and the OBC level (TTL level). OBC serial port should not be directly connected to RS-232C signals of the PC. Conversion from TTL level to RS-232C can be done by using special integrated circuits for example MAX232 or ADM3202.

RS-232C communication protocol is an interface standard of serial communication (a communication format of bit-by- bit transfer in a line). In the asynchronous transmission, there are the systems with flow control and those without it. The asynchronous transmission without flow control will use three terminals, GND to give the base voltage, TXD to transmit, and RXD to receive. The asynchronous transmission is more frequently adopted because of its simplicity.

There is only one module for serial communication terminals on the OBC of i-CanSat-6. It consists of a transmitting pin (TXD) and receiving pin (RXD). The receiving pin is only used to receive data from the GPS module, and the transmitting pin is only used to send data to the XBee communication module. For the asynchronous transmission, the communication speed in the range between 4,800 bps to 38,400 bps is often adopted.

3.4.6 I^2C

The I^2C (Inter-Integrated Circuit) communication is a standard for bus system serial communication. In this standard, there are two lines. They are the clock line (SCL) and data line (SDA) and they are used for communication. The communication speed can reach up to 400 kbps. The most commonly used speed is 100 kbps, and the multiple devices can be arranged on a bus for communication by allocating an ID for each device, as shown in Figure 3-9. However, the bus can communicate with only one device at a time, therefore some precautions are needed, such as specifying the order of communication if multiple devices need to be communicated. In addition, the two communicating devices are under the relationship of master and slave. Normally, once both SCL and SDA drops to the "0"or low level, they cannot return to the "1" or high level by themselves; consequently the pull-up resistor will be required on each line using an external resistor, as shown in Figure 3-9.

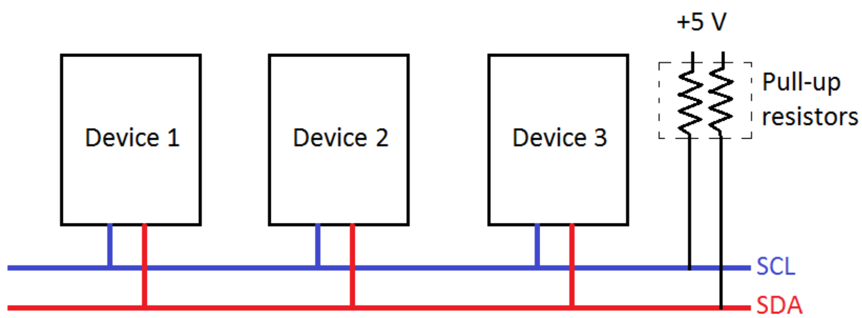


Figure 3-9: I^2C (Inter-Integrated Circuit) communication.

3.5 Basics of Soldering Technique

Soldering is a process to mount the electronic parts on a Printed Circuit Board (PCB). Soldering is secures the electronic parts by melting of a metal with low melting point (solder) using a soldering iron. In this section, the soldering technique and how to practice it are presented.

3.5.1 Tools and Preparation

Adequate preparation and setup are recommended before start soldering because tiny electronic parts will be soldered. Necessary tools are required and must be organized on the worktable so that it can be quickly accessed. This will increase the work efficiency. Recommended set of tools for the soldering process are shown in Figure 3-10. Figure 3-11 shows an example of how you can organize a worktable for soldering. Small electronic parts can be stored in a case, each type of parts have its own compartment in the case. Write the name, type number and number of parts on the cover of the case as shown in Figure 3-12. The circuit diagram must be printed and quickly accessed on the worktable. It is recommended that the specification sheets for each electronic part are available in hardcopy or softcopy. Sometimes, soldering has to be conducted outdoor where the accessibility of electricity is limited. In this situations, hardcopy of the specification sheets is essential.

Figure 3-10: Typical tools for i-CanSat assembly and integration. (1) Digital multimeter, (2) Pincette, (3) Scissors, (4) Nipper(Large), (5) Wire stripper, (6) Long nose pliers, (7) Tape, (8) Solder, (9) Desoldering tool, (10) Soldering iron base with sponge, (11) Small file tool, (12) Screw drivers set, (13) Pliers, (14) Nipper (Small), (15) Crimping tool, (16) Soldering iron tip cleaner, (17) Solder paste, (18) Soldering absorbent wire, (19) Solder flux, (20) Soldering iron, and (21) Tool case.

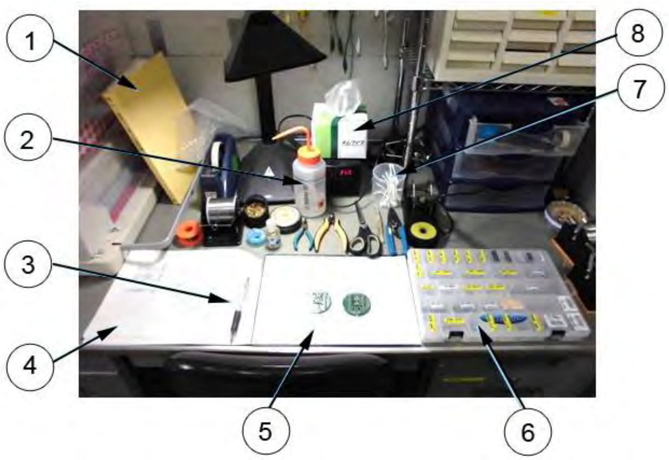


Figure 3-11: Organization of the worktable. (1) Specification sheets file, (2) Ethanol, (3) Pen, (4) Circuit diagram, (5) Work space over insulation sheet, (6) Parts case, (7) Cotton swab, and (8) Waste.

a) Parts' case with cover open. b) Parts' case with cover closed.

Figure 3-12: Electronic part case.

3.5.2 **The Soldering Iron**

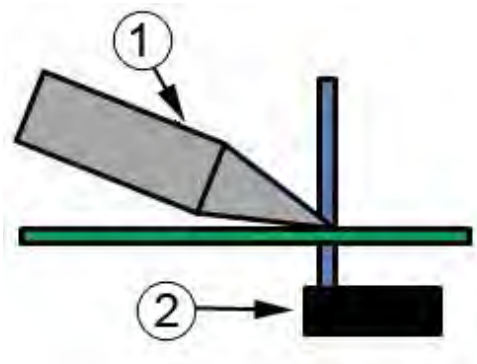
Soldering iron is the main tool used in the soldering. The temperature of the iron tip increased by built in electric heater. The combination of proper tip shape and iron's power (temperature) is important for the precision of soldering. There are types of soldering iron equipped with a temperature control capability. These types are useful because temperature can be changed in accordance with the electronic part to be soldered. Continuous soldering may result in solder remains on the iron's tip and sticking of burnt flux or paste as shown in Figure 3-13-a. This will degrade the reliability of soldering and it must be removed on frequent basis. To remove these remains, the tip of the iron may be cooled for a while using sponge with water. The sponge can be located in the soldering iron base, as shown in Figure 3-13-b. Also wire wool may be used like a cleaner to clean the tip, as shown in Figure 3-13-c.

a) Remains solder on the iron's tip. b) Using sponge of the soldering iron's base. c) Tip cleaning using wire wool.

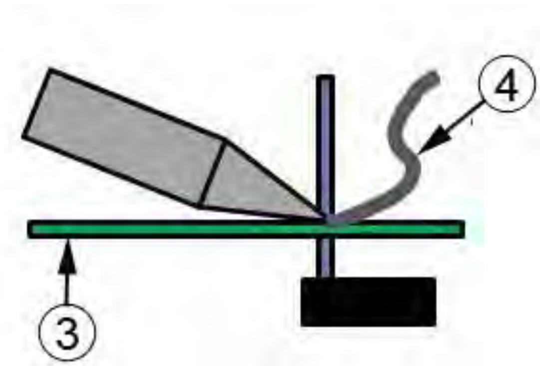
Figure 3-13: Methods of cleaning the soldering iron tip.

3.5.3 **The Soldering Process**

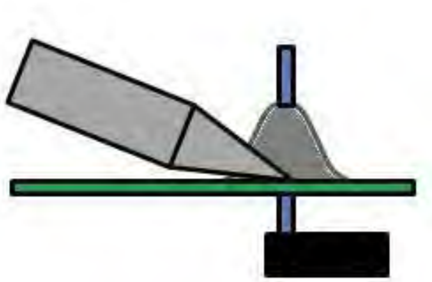
For successful soldering, efficient melting and solidifying technique is required that achieves firm mounting between the electronic parts and board. The heat up of the solder and electronic parts for a long time should be avoided. The sequence of soldering process is depicted in Figure 3-14. Firstly before applying the solder, the temperature of the part's contact point must be increased, as shown in Figure 3-14-a . Secondly, application of solder takes place while holding the soldering iron at the same place, as shown in Figure 3-14-b, thirdly, keep melting the solder by soldering iron until it reaches all the contact area, as shown in Figure 3-14-c. Figure 3-14-d shows the final shape of the successful soldering spot. Each step takes about one second. An awkward shape, swelling at the top, or shrinkage in the contact area with the board will be judged as defective soldering. Defective soldering will not only degrade the reliability of the connection of the parts, but also it will be damaged during the launch of the CanSat or the satellite. Successful soldering requires not only the shape, but also smooth and shining surface condition, as shown in Figure 3-15. In the areas with large print patterns, such as GND connection areas, heat loss is increased, as a result, the solder will not be fused because the temperature is unlikely to rise. Special care must be taken in such situation, by increasing the iron's power for example.



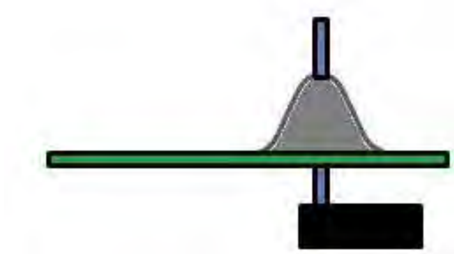
a) Pre-soldering heating.



b) Solder application.



c) Post soldering heating.



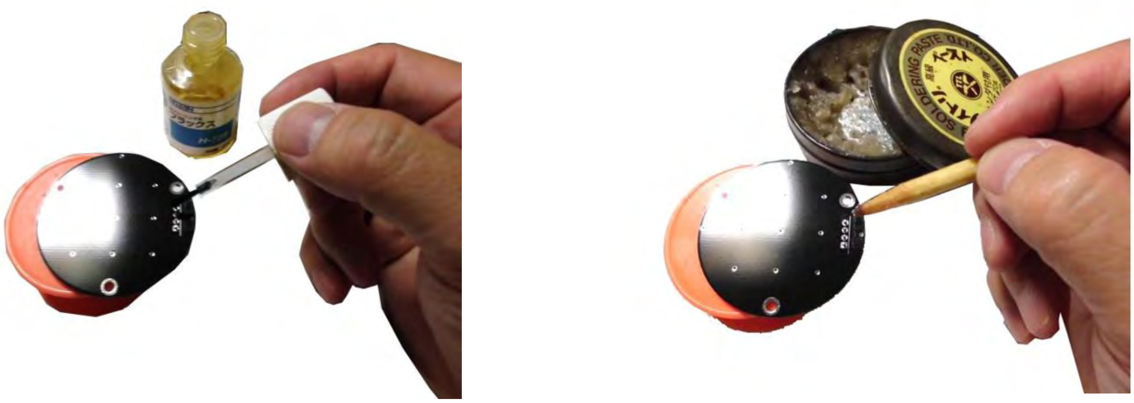
d) Final shape.

Figure 3-14: Sequence of soldering. (1) Soldering iron, (2) Electronic part, (3) PCB, and (4) Solder.

The solder may not be well fused depending on the surface conditions of the board or electronic parts. To solve such a problem, apply pre-soldering with a very small quantity of solder on the point to be soldered on the board or the electronic parts. Another solution is to perform the soldering after applying some quantity of flux or paste to help the solder to adhere, as shown in Figure 3-16. It should be noted that the flux or paste will corrode the solder after solidification, therefore, it is not recommended to use it if highly reliable soldering is required. Excessive flux or past must be avoided.

Figure 3-15: Appearance of successful soldering shapes.

Flux, paste and other ingredient in the solder usually stain the area around the soldering points. It is recommended to clear this area with a cotton bud swamped with ethanol, as shown in Figure 3-17. This will help also removing any electric conduction between the electric terminals in the PCB caused by the application by flux or paste.



a) Application of flux. b) Application of paste.

Figure 3-16: Application of flux or paste to help fusing the solder.

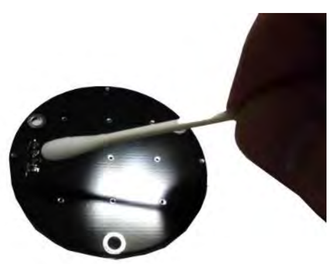


Figure 3-17: Cleaning with ethanol after soldering.

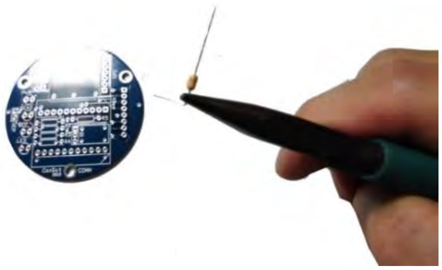
Secure the board to prevent movement during the soldering process. Hold the board on the worktable by touching softly with the fingers, as shown in Figure 3-18. Use the insulation sheet or paper on the worktable to protect the board from damage from short circuit when testing the board. The board testing may be performed on the worktable with power supplied. Therefore, the insulation sheet or paper is required to prevent the short circuit if the worktable is conductive.

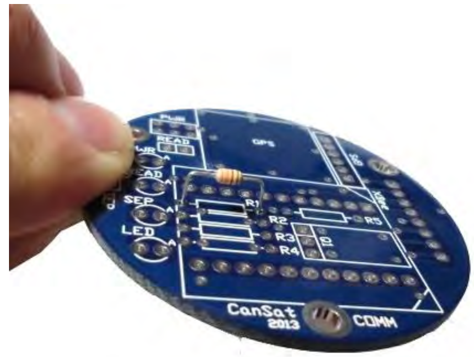
Figure 3-18: Holding the board during the soldering process.

3.5.4 **Soldering Electronic Parts**

There are two types of mounting the electronic parts of i-CanSat-6. They are the through-hole mount type, and the surface mount type. The surface mount type is often used for CanSat. But, if is required to avoid multiple wiring or the effect of strong force that may remove the parts due to vibration during launch, adopt the through-hole mount type. For both types, soldering is required. For the surface mount type of the i-CanSat, the details will be explained at the end of this section as well as in section 3.6.4. In this section, the soldering for the through-hole type is explained.

For soldering the through-hole type, similar to the part shown in Figure 3-19-a, first determine the position of the part in the board and the length of the part's terminals or legs that need to be folded by comparing the pattern on the board. In order to do this, fold the terminals of the main body while holding the folding point with long-nose pliers to prevent the bending force from being applied to the main body of the part. If it is bent without holding the support point, force may be applied to the main body of the part and may cause damage. After properly folding the legs, insert them in the holes in the board, as shown in Figure 3-19-b.



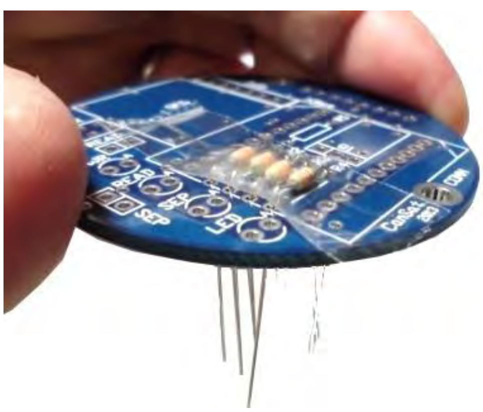


a) Folding the through-hole mount electronic part.

b) Insertion of the through-hole mount electronic part into PCB.

Figure 3-19: Soldering the through-hole mount electronic parts.

Basically, the parts shall be soldered one by one, but if there are parts that have similar shape and located near to each other, the simultaneous soldering can be done. For the through-hole mounting type, the tape can be used to temporarily holding the parts after pressing over the main body of the parts on the board to be attached with it, as shown in Figure 3-20-a. After temporarily holding, cut off the legs in the opposite surface of the board leaving some proper length of about 2-3 mm. Leaving similar length for the parts' legs with improve the general appearance of the board and increase the efficiency of repetitive soldering work. In addition, even very short legs will not be a problem as far as the leg is not buried under the board surface. If the leg cannot completely go through the board thickness, the parts has to be changed.



a) Holding the parts in place using tape.

b) Cutting the legs of the parts to be of equal lengths.

Figure 3-20: Placing the through-hole mount part in the board before soldering.

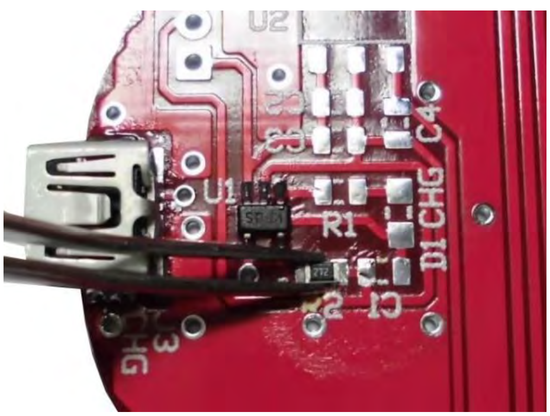
It is recommended not to dispose of the cut off-legs but store them in a small bag, as shown in Figure 3-21. They can be reused in place of the tinned wire for connection between the parts on the universal board similar to the USR board of i-CanSat-6.

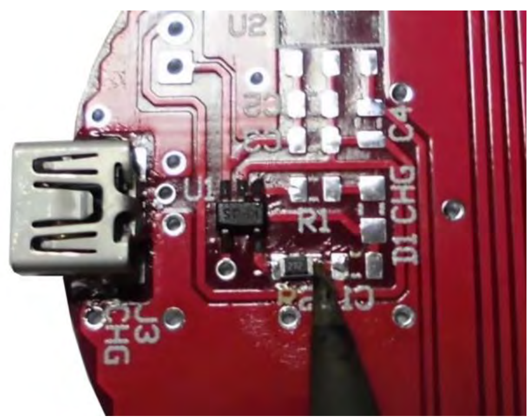
a) Collect the cut-off legs.

b) Store the cut-off legs.

Figure 3-21: Store the cut-off legs for possible usage in the future.

On the other hand, the surface-mount type parts are very small in general. They can blow away easily even by breath. In soldering the surface-mount type part, first it must be fixed on board with pincette, as shown in Figure 3-22-a , and secure the entire bottom edge of the part to the circuit board by viewing its ends and their correspond location on the board. After one terminal is soldered, the part is fixed on the board so that the pincette may be released to solder the other terminal of the part, as shown in Figure 3-22-b. Note that the temperature of the surface-mount type parts easily rises up in soldering because of its small sizes and it melts the solder rapidly.





a) Use pincette to place the part.

b) Soldering the terminal of the surface mount part.

Figure 3-22: Soldering surface mount type electronic parts.

3.5.5 **Soldering the Interface Connector**

The interface connector between the boards in i-CanSat-6 is done using combination pin headers, shown in Figure 3-23-a, and pin sockets, shown in Figure 3-23-b . There are six interfaces in i-CanSat-6 which have been presented earlier in this manual. They are as follow:

1. GPS/PWR – PWR/GPS (8 pins interface connector)
2. PWR/USR – USR/PWR (8 pins interface connector)

3. USR/OBC-1 – OBC/USR-1 (8 pins interface connector)
4. USR/OBC-2 – OBC/USR-2 (8 pins interface connector)
5. OBC/CAM – CAM/OBC (7 pins interface connector)
6. XBEE/CAM – CAM/XBEE (3 pins interface connector)

Pin headers and pin sockets are sold in units of 20 pins, 36 pins, 40 pins, etc. The required number of pins can be cut out from the supplied units. Pin header with pin length of 6.1 mm and insulator height of 2.5 mm and pin socket of insulator height of 6.5 mm are adopted in this kit.

a) Pin header

b) Pin sockets

Figure 3-23: Pin header and pin sockets used for interface between the boards in i-CanSat-6.

The required number of pins can be cut from the supplied units using a nipper for pin headers and pin sockets with gaps, as shown in Figure 3-24. On the other hand, sometime the available pin sockets may have no gap. In this case, the pin socket will be cut just above one of the pin sockets using a nipper leaving the necessary number of pins and wasting one pin, as depicted in Figure 3-25.

a) Cutting the pin header.

b) Cutting the pin socket.

Figure 3-24: How to cut the pin header and pin socket in case gaps exist between pins.

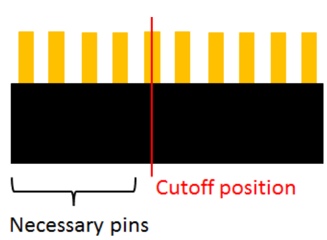
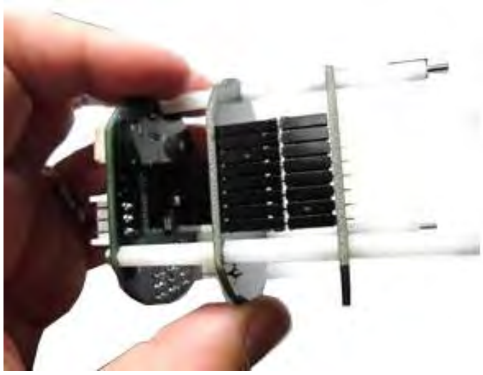


Figure 3-25: How to cut the pin sockets in case no gaps exist between pins.

Assemble the pin headers and pin sockets slightly as shown in Figure 3-26-a. This forms the interface connector. To solder the interface connector at both side of the boards, assembly of the two boards and the interface connector must be done by using the supplied spacers, as shown in Figure 3-26-b. The correct spacer height must be used because the supplied spacers have different lengths. As the screw of spacer is tightened, the pin header and pin sockets are thoroughly mated, as shown in Figure 3-26-c. After confirming that the two boards are fixed, solder the terminals of pin sockets onto the boards, as shown in Figure 3-24-d. This procedure adequately defines the position of pin sockets, and should be adopted to all of the boards' interface connectors.

a) Prepare and assembly of interface pin header and pin socket.

b) Assembly the interface with two boards of i-CanSat-6.



c) Holding the interface with the two boards by fasten the spaces of the boards.

d) Soldering both ends of the interface.

Figure 3-26: Soldering the interface connector between two boards of i-CanSat-6.

3.5.6 **Solderless Connectors**

A number of connectors must be attached for i-CanSat-6. The EH series of JST Mfg. Co., Ltd. is adopted for the i-CanSat-6 for these connectors. The three circuits type of the header (post) and the housing of the EH- series connectors are shown in Figure 3-27-a and Figure 3-27-b, respectively. The connectors have standard pitch of 2.54 mm. The header (post) is soldered to the board and the housing is attached to the cable using metal contacts (terminal), shown in Figure 3-27-c, and a crimping tool, shown in Figure 3-28-a. The metal contacts are of type SHE-001 T-P0.6 from the same company. These metal contacts are attached to the cable wires using a crimping tool and the process of crimping is illustrated in Figure 3-28-b to Figure 3-28-n. After crimping all the contacts to the cable, insertion to the connector housing one by one takes place.

Five connectors of this type are adopted in i-CanSat-6 with their associated accessories. They are as follow

1. The GPS receiver connector to the GPS board (J12)
2. The USB connector on the GPS board (J13)
3. Two battery connector on the PWR board (J23 and J24): Only one will be used.
4. The PRG connector on the OBC board (J44)

a) Connector header or post. b) Connector housing. c) Contacts (terminals)

Figure 3-27: The EH-Type connector.

a) Crimping tool.

b) Choose the contact/terminal size.

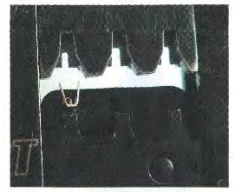
c) Open the tool.

d) Open the flap locator.

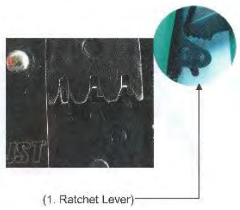
e) Contact has to be inserted until stop. Please pay attention to the correct cross section.

f) Close the flap locator.

g) Check of the flap locator is closed.



h) The contact lies in the center of the respective profile.



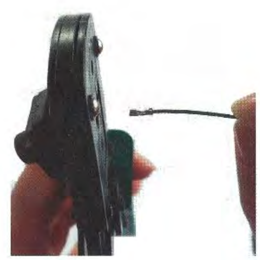
i) Close the tool lightly until the contact is held.

j) Insert the cable until insulation stop blade.

k) Close the tool

l) Open the tool

m) Loosen the flap locator.

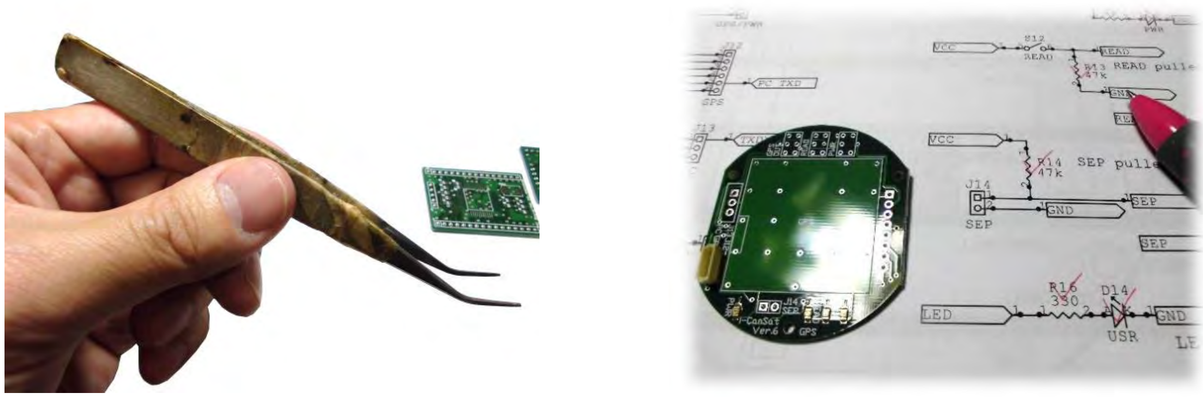


n) Remove the crimped contact.

Figure 3-28: The crimping tool and how to use it.

3.5.7 **General Consideration**

Use a pincette for handling small parts. The pincette shall be insulated with anti-static electricity tape, for example the masking tape, as shown in Figure 3-29-a. After mounting parts, record the progress with check marks on the circuit diagram indicating the completion of mounting, as shown in Figure 3-29-b.



a) A pincette with electrostatic protection. b) Record the progress of the board soldering.

Figure 3-29: Electrostatic protection and recording the progress of soldering.

3.6 Soldering of i-CanSat Boards

In this section, the circuit diagrams of each i-CanSat-6 boards are presented. Firstly, a brief description of each board is presented. Secondly, the pre-installed parts and the parts need to be soldered are presented. Finally, the procedure to solder the electronic parts in the board is described with illustrations. The reader is advised to refer to the section titled "*Part/Component Identification Number*" at section 3.2.2 for explanation about the numbering system used for the electronic parts of i-CanSat-6.

3.6.1 **GPS Board**

3.6.1.1 Description

The circuit diagram of the GPS board is shown in Figure 3-30. From the circuit diagram the functions of electronic parts can be summarized as follows:

- Connectors and Jumpers
  - **J11** is the GPS/PWR interface connector's terminal.
  - **J12** is the GPS receiver interface connector header.
  - **J13** is the USB-serial connector header for PC debugging.
  - **J14** is the SEP pin for the detection of separation from a CanSat carrier, and connected with a jumper pin tacked with one of parachute lines. When the jumper pin inserted, the SEP signal is the same voltage as GND. When the parachute deployed, the parachute's line pulls off the jumper pin from **J14**, and the SEP signal changes to the voltage as VCC and **D13** lights. By checking the voltage of the SEP signal by the OBC, CanSat separation from the carrier can be detected.
- Switches
  - **S11** is the PWR switch to turn on or off the CanSat. When PWR switch closed, meaning ON, **D11** lights, BAT+ is connected to VIN, and VIN (battery voltage) is converted to VCC (3.3 V) , the hot line of whole CanSat by the voltage regulator, **U11**.

- **S12** is the READ switch to set the CanSat mode to either flight mode (FLT) or read mode (READ). When the READ switch opened, meaning OFF, the READ signal is the same voltage as GND through the resistance **R13**. When the READ switch closed, meaning ON, the READ signal is the same voltage as VCC and **D12** lights. In this way, CanSat can be detected whether it is under FLT mode or READ mode.
    - **S13** is the GPS_TXD switch to select the destination of the GPS data to either the OBC or to PC through **J13**. When GPS_TXD switch is OFF, GPS data is sent to OBC. When GPS_TXD switch is ON, GPS data is sent to **J13** (PC).
- Electronic Modules
    - **U11** is a voltage regulator, which converts the voltage supplied by a battery (VIN) on PWR board to 3.3V (VCC).
    - **U12** is GPS receiver, and its line are connected to **J12**. Any GPS can be used if its dimensions can fit into the allocated space on the GPS board and the pin assignment is matched. GT-723F, GMS6-CR6, and GMS7-CR7 made by CANMORE Inc., are compatible, and the data format is NMEA-0183.
- Resistors
    - **R11** is current limiting resistor for D11 (LED). D11 is directly driven from the battery so that R11 is the largest in comparison to R12, R14, and R15.
    - **R12** is current limiting resistor for D12 (LED) to supply a current of 3-4 mA to each LED.
    - **R14** pulls up the SEP signal to VCC.
    - **R15** is 0 ohm resistor.
    - **R16** is current limiting resistor for D14 (LED) to supply a current of 3-4 mA to the LED.
    - **R17** is a protective resistance to stabilize the serial communication.
- Capacitors
    - **C11** is capacitors to stabilize the input and output voltages.
    - **C12** is capacitors to stabilize the input and output voltages.
- Light Emitting Diodes (LEDs)
    - **D11** is an indicator for the **S11** switch or the PWR switch. It lights when power is on.
    - **D12** is an indicator for the **S12** switch or the READ switch. It lights when i-CanSat-6 is in the READ mode.
    - **D13** is an indicator of the SEP or **J14** jumper. When separation is detected or **J14** is pulled off, D13 lights.
    - **D14** is a user LED and it can be controlled through OBC programming. It can be used for debugging and operation of i-CanSat-6.

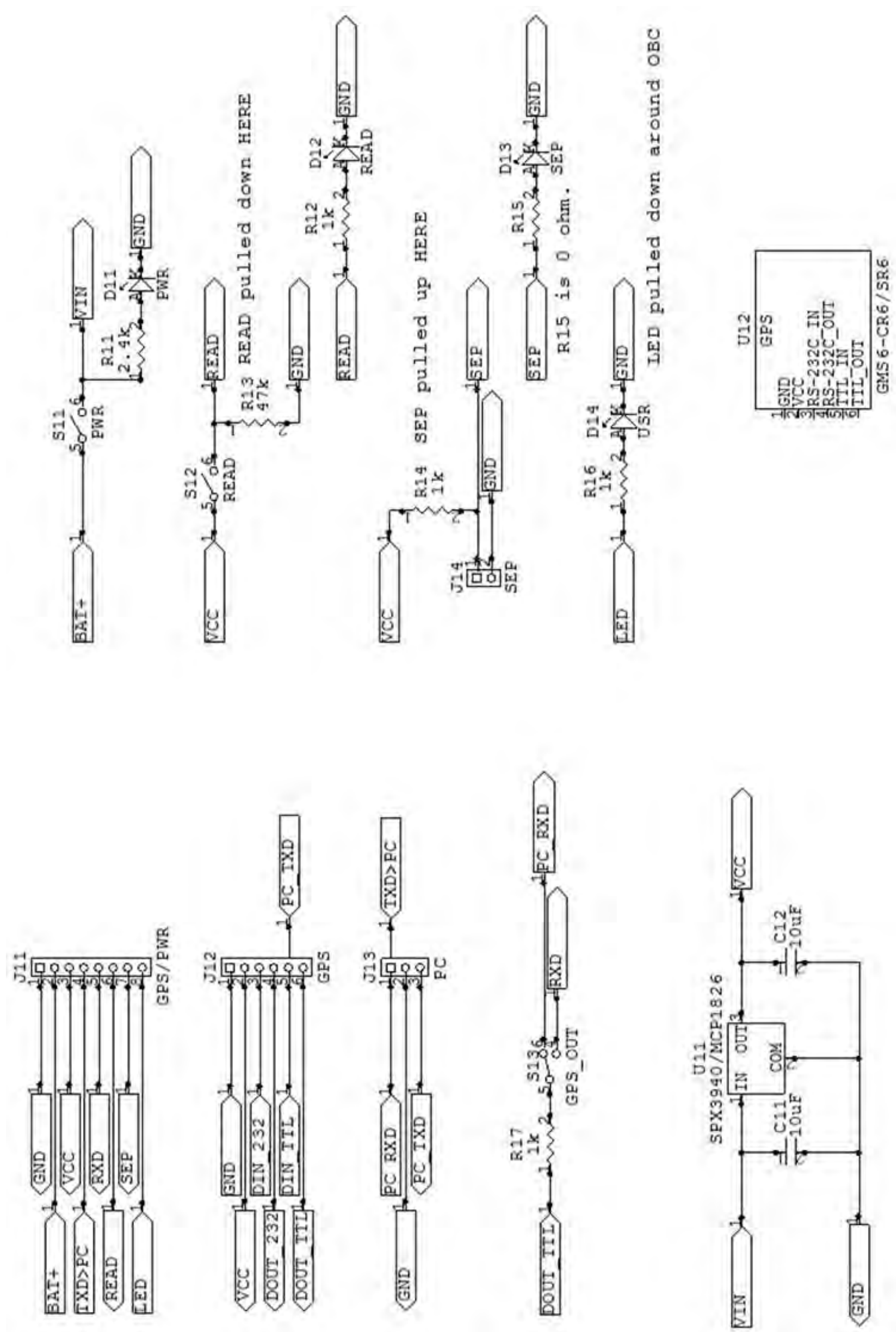
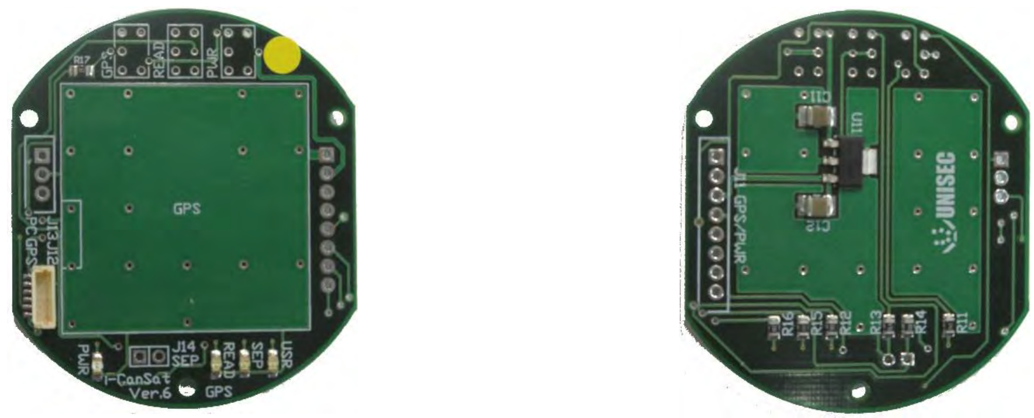


Figure 3-30: Circuit diagram of the GPS board.



a) Upper surface. b) Lower surface.

Figure 3-31: GPS board with the preinstalled parts and before soldering the electronic parts.

3.6.1.2 Pre-installed parts

The following parts are pre-installed to the GPS board, as shown in Figure 3-31:

- Resistors
    - **R12, R13, R14, R15, R16,** and **R17.**
- Capacitors
    - **C11**and **C12**.
- LEDs
    - **D11, D12, D13,** and **D14.**
- Connectors
    - **J12.**
- Electronic modules
    - **U11.**

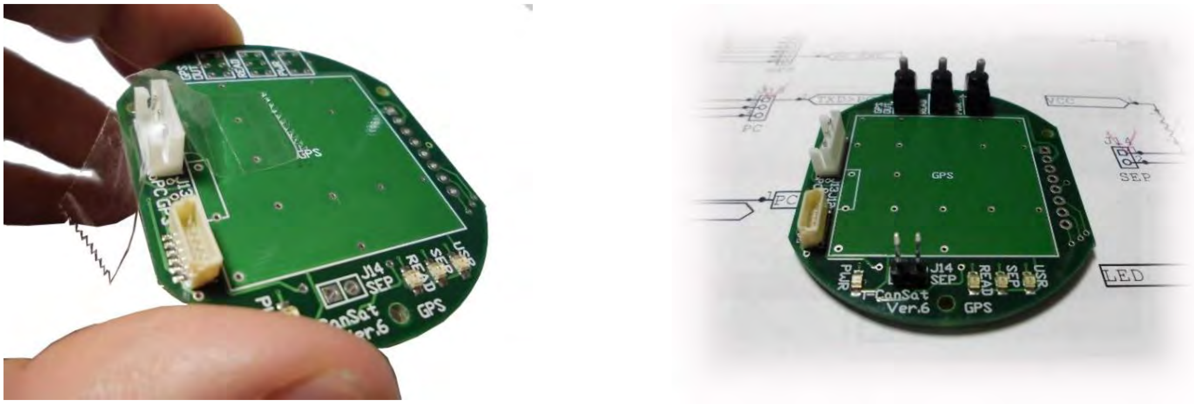
3.6.1.3 Parts to be soldered or mounted

The following parts are need to be soldered or installed on the GPS board:

- Connectors
    - **J11, J13**, and **J14.**
    - The housing connector of **J13.**
- Switches
    - **S11, S12**, and **S13.**
- Electronic Modules
    - **U12.**

3.6.1.4 Soldering guidelines

Generally, start with the small parts when soldering on a board. **J13** is soldered first, as shown in Figure 3-32-a, followed by **S11**, **S12**, **S13**, and **J14**. Tape can be used to hold the parts in place before soldering. After completing the soldering of each part record your progress on the circuit diagram, as shown in Figure 3-32-b.



a) Mounting **J13** connector by tape. b) Record the progress.

Figure 3-32: Soldering and recording the progress of GPS board.

To solder the **J11**, the GPS board, the PWR board and the complete GPS/PWR-PWR/GPS interface connector must be assembled using the 20 mm spacers, as shown in Figure 3-33. This is also allowed the soldering of **J21** on the PWR board. Follow the same procedures described in section 3.5.5. The housing connector of **J13** is mounted using the procedures described in section 3.5.6.

The GPS is very sensitive to static charge, and is susceptible to damage even from the charge of human body. The GPS should therefore be protected with Kapton tape or insulated tape, as shown in Figure 3-35-a. Material-collecting charge or material-containing carbon should not attached to the GPS receiver. However, because no protection is provided by the materials that takes charge, the GPS receiver is obstructed by the tape of material-containing carbon. The cable of the GPS is weak and it must be handled with care to avoid any damage that may result from excessive use or fatigue. The GPS receiver and its cable are shown in Figure 3-34. The GPS is fixed onto the board with double-sided tape. The GPS cable is connected to the GPS connector header, in the GPS itself, and **J12**. The cable should not obscure the GPS and should be fixed to the board if necessary. The final assembly of the GPS board is shown in Figure 3-35.

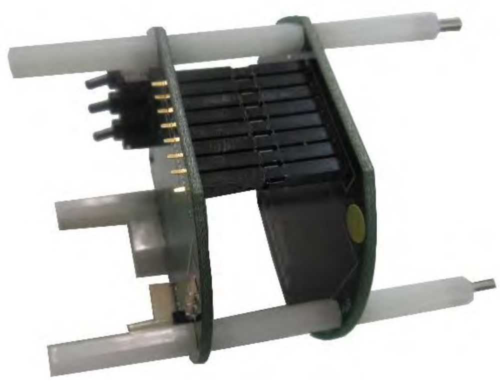
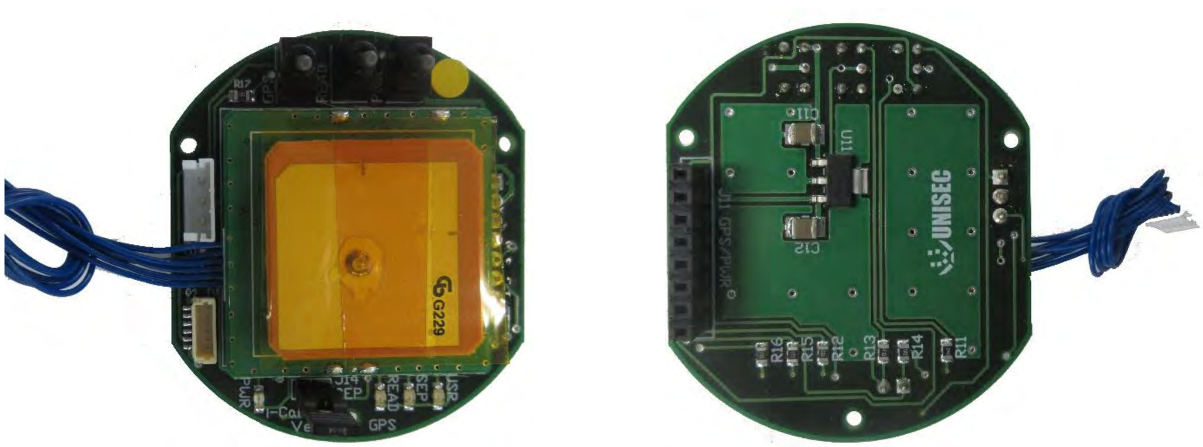


Figure 3-33: Soldering the GPS/PWR-PWR/GPS interface connector.

Figure 3-34: GPS receiver GT-723F made by CANMORE Inc.



a) Upper surface. b) Lower surface.

Figure 3-35: Completion of the installation and the soldering works of the GPS board.

3.6.2 **Power Board**

3.6.2.1 Description

The circuit diagram of the PWR board is shown in Figure 3-36. From the circuit diagram the functions of electronic parts can be summarized as follows:

- Connectors
  - **J21** is the PWR/GPS interface connecter's terminal.
  - **J22** is the PWR/USR interface connector's terminal.
  - **J23** is the header connector of **BAT1** (first battery). Normally one battery connected to either **BAT1** or **BAT2** is sufficient.
  - **J24** is the header connector of **BAT2** (second battery)

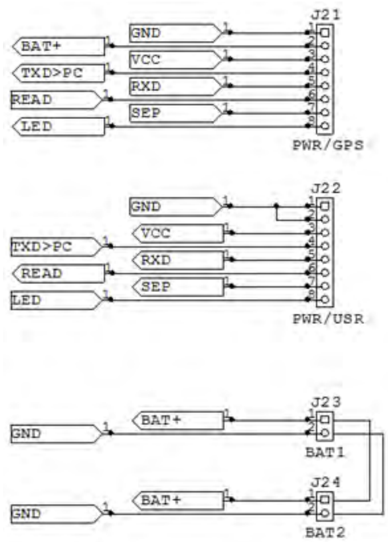
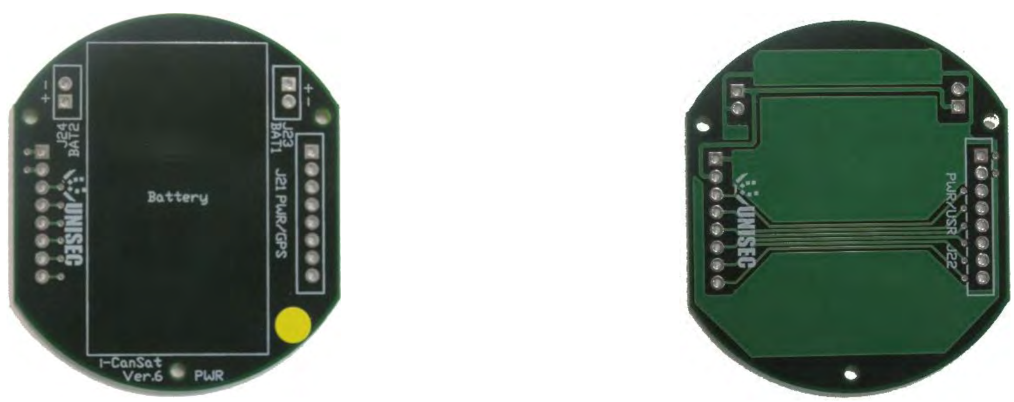


Figure 3-36: Circuit diagram of the PWR board.



a) Upper surface b) Lower surface.

Figure 3-37: PWR board before soldering the electronic parts.

3.6.2.2 Pre-installed parts

There aren't pre-installed parts on the PWR board as shown in Figure 3-37.

3.6.2.3 Parts to be soldered or mounted

The following parts are needed to be soldered or installed on the PWR board:

- Connectors
  - **J22, J23**, and **J24** (**J21** is already soldered during the soldering of **J11** as described in section 3.6.1.4)
  - The housing connectors of **J23** and **J24** (if exist).
- Electronic Modules

- **BAT1**

3.6.2.4 Soldering guidelines

J23 and **J24** can be soldered in the same way as **J13**. To solder **J22**, PWR board, the USR board and the complete PWR/USR-USR/PWR interface connector must be assembled using the 20 mm spacers, as shown in Figure 3-38. This is also allowed the soldering of **J31** on the USR board. Follow the same procedures described in section 3.5.5. The housing connectors of **J23** and **J24** (if exist) is mounted using the procedures described in section 3.5.6. **BAT1** is fixed onto the board with double-sided tape

Figure 3-38: Soldering the PWR/USR-USR/PWR interface connector.

For the battery, the 006P type 9V Alkaline battery is strongly recommended. For continuous operation during the programming of OBC, it is recommended to use external power supply of 9 voltages more than 300 mA of current or supply the power through the PICKit3 as described in section 3.9 . The final assembly of the PWR board is shown in Figure 3-39.

a) Upper surface

b) Lower surface

Figure 3-39: Completion of the installation and soldering works of the PWR board.

3.6.3 **User Board**

3.6.3.1 Description

The circuit diagram of the USR board is shown in Figure 3-40. From the circuit diagram, the functions of electronic parts can be summarized as follows:

- Connectors
  - **J31** is the USR/PWR interface connecter's terminal.
  - **J32** is the USR/OBC-1 interface connector's terminal.
  - **J33** is the USR/OBC-2 interface connector's terminal.

On the USR board, the user is able to install sensor modules and other electronic parts. Terminal labelled VCC and GND are used as the hot and the return, respectively. The electronic modules on USR board can be connected to up to six channels of analog inputs or digital inputs/outputs and the I^2C bus through the USR/OBC-2 interface connector.

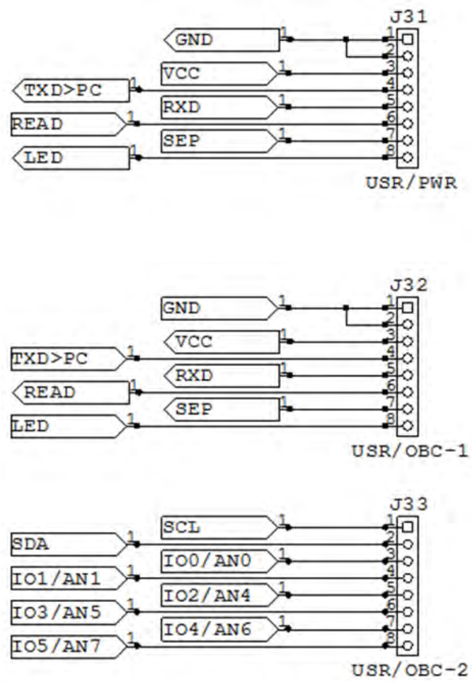


Figure 3-40: Circuit diagram of the USR board.

a) Upper surface. b) Lower surface.

Figure 3-41: USR board before soldering the electronic parts.

3.6.3.2 Pre-installed parts

There aren't pre-installed parts on the USR board as shown in Figure 3-41.

3.6.3.3 Parts to be soldered or mounted

The following parts are needed to be soldered or installed on the USR board:

- Connectors
  - **J32** and **J33** (**J31** is already soldered during the soldering of **J22** as described in section 3.6.2.4)

It is recommended to continue assembly the i-CanSat without install any electronic modules on the USR board. The reason is that basic i-CanSat operations must be conducted first using GPS, XBee, USB-Serial Cable and PICKit3. Then advanced operations using electronic modules installed on the USR board can be done. The reader can refer to the appendix C and section 3.9 for USR board sensor modules and advanced i-CanSat operation, respectively.

3.6.3.4 Soldering guidelines

To solder **J32** and **J33**, USR board, the OBC board, the complete USR/OBC-1 - OBC/USR-1 interface connector and USR/OBC-2 - OBC/USR-2 interface connector must be assembled using the 20 mm spacers, as shown in Figure 3-42. This is also allowed the soldering of both **J41** and **J42** on the OBC board. Follow the same procedures described in section 3.5.5.

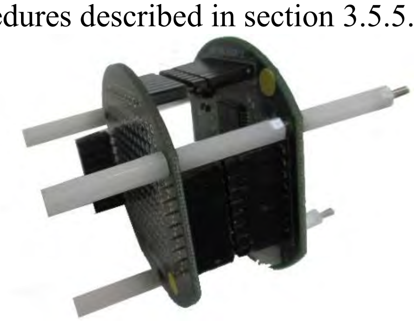


Figure 3-42: Soldering the USR/OBC-1 - OBC/USR-1 and the USR/OBC-2 - OBC/USR-2 interface connectors.

3.6.4 **OBC Board**

3.6.4.1 Description

The circuit diagram of the OBC board is shown in Figure 3-43. From the circuit diagram, the functions of electronic parts can be summarized as follows:

- Connectors
  - **J41** is the OBC/USR-1 interface connecter's terminal.
  - **J42** is the OBC/USR-2 interface connecter's terminal.
  - **J43** is the OBC/CAM interface connecter's terminal. This is a surface mount pin header with 7 pins.
  - **J44** is terminal to write a program to the OBC (**U41**: PIC16LF877A-I/PT).
- Switches
  - **S41** is the TXD_OUT switch used to select the destination to transmit the data in RS-232 communication protocol to either **J13** (PC) or to the XBee communication module.
- Integrated Circuits (ICs).
  - **U41** is the On-Board Computer (OBC) which is PIC16LF877A-I/PT.
  - **U42** is through-hole DIP8 IC type EEPROM (MICROCHIP, 24LC1025-I/P), with storage capability of 1024 kbit (128kB).
  - **U43** is through-hole DIP8 IC type EEPROM (MICROCHIP, 24LC1025-I/P), with storage capability of 1024 kbit (128kB).
    NOTE: EEPROMs (**U42** and **U43**) are connected to the OBC using I^2C interface bus. The IDs for the I^2C communication with these EEPROMs are specified by the terminal A0 and A1 in the EEPROMs, as shown in Figure 3-45. These terminals are connected to the corresponding VCC and GND voltage on the mounting position of **U42** and **U43** so that **U42** has ID (address) of 0x00 (A0 = L and A1 = L) and **U43** has ID of 0x01 (A0 = H and A1 = L). This means that the user have to use these IDs in the program to access the EEPROMs. In the sample program(s) only **U42** is accessed which has the ID = 0x00. Terminal A2 of the EEPORM is always high. The WP of the EEPROM is a pin for write protection, to enable writing then WP must be low (L).
  - **XTAL41** is a crystal oscillator (KYOCERA, CX8045GB1000H0HEQZ1) used to drive the OBC. The oscillator of the i-CanSat-6 is normally rated at 10 MHz. Changing the frequency of oscillator requires to change the values of BRGH and SPBRG parameters, in the sample program to adjust the baud rate of RS-232 communication to 9600 bps.
- Resistors
  - **R41** is required for the crystal oscillator.
  - **R42** is pull-up resistor for RS-232 communication, RXD line.
  - **R43** is pull-up resistor for RS-232 communication, TXD line.
  - **R44** is resistor as per the OBC specification sheet.
  - **R45** is resistor as per the OBC specification sheet.
  - **R46** is pull-up resistor for I^2C communication, SDA line.

- **R47** is pull-up resistor for I^2C communication, SCL line.
- **R48** is pull-up resistor for **D14** (LED) on the GPS board.
- **R49** is pull-up resistor for of the I/F (interface) IF1 line.
- **R410** is pull-up resistor for of the I/F (interface) IF2 line.
- **R411** is a protection for the TXD line.

- Capacitors
  - **C41** is required for the crystal oscillator.
  - **C42** is required for the crystal oscillator.
  - **C43** is bypass capacitor to stabilize the voltage on the power lines to the OBC.
  - **C44** is bypass capacitor to stabilize the voltage on the power lines to the OBC.

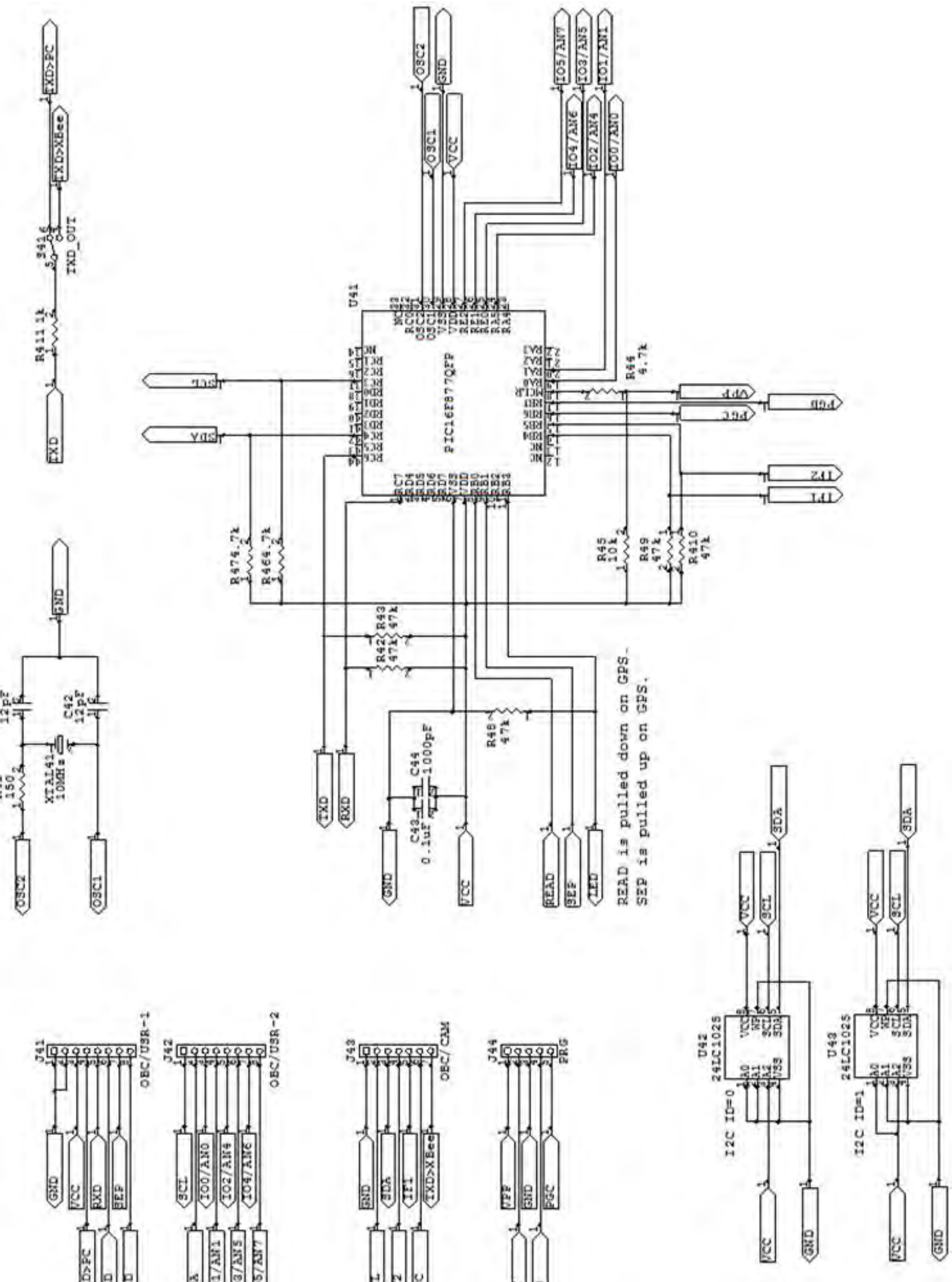
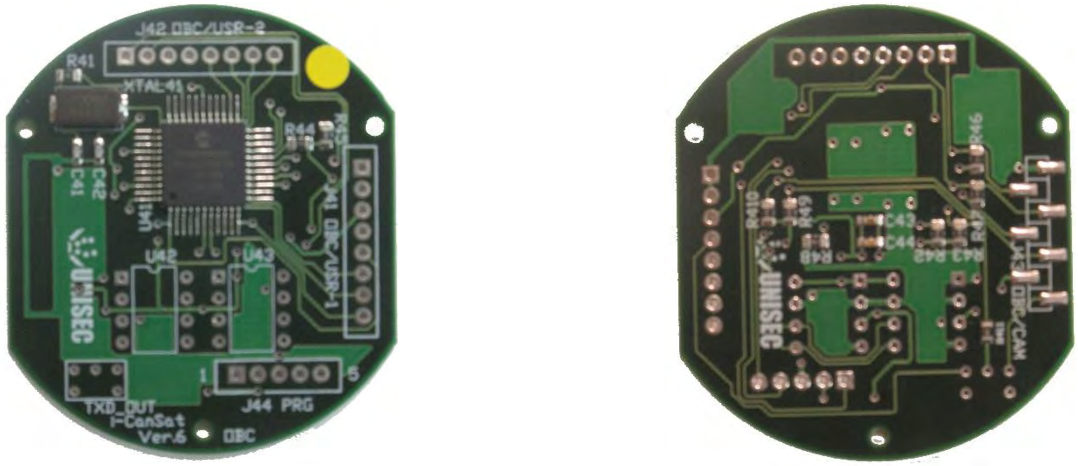


Figure 3-43: Circuit diagram of the OBC board.



a) Upper surface b) Lower Surface

Figure 3-44: OBC board with preinstalled electronic parts and before soldering the electronic parts.

3.6.4.2 Pre-installed parts

The following parts are pre-installed to the OBC board, as shown in Figure 3-44:

- Resistors
  - **R41**, **R42**, **R43**, **R44**, **R44**, **R45**, **R46**, **R47**, **R48**, **R48**, **R49**, **R410**, and **R411**
- Capacitors
  - **C41, C42, C43**, and **C44**.
- ICs
  - **U41** and **XTAL41**

3.6.4.3 Parts to be soldered or mounted

The following parts are needed to be soldered or installed on the OBC board:

- Connectors
  - **J43** and **J44** (**J41** and **J42** are already soldered during the soldering of **J32** and **J33**, respectively, as described in section 3.6.3.4**)**
  - The housing connector of **J44**.
- Switches
  - **S41**
- ICs
  - **U42**

3.6.4.4 Soldering guidelines

U42 should be soldered first because the other parts might become obstacles. Follow the 1st pin marking on the OBC board and the notch marking of the IC, as shown in Figure 3-45 when **U42** is inserted. When mounting the **U42** in the board, make sure that the **U42** legs will not be bent. If any of the legs is bent then it cannot be inserted and it must be straightened first with the proper tool. Countermeasures against static electricity must be conducted. When **U42** is inserted, check the alignment and cohesion of the IC relative to the board. The sequence of soldering is the 1st pin at the beginning, followed by 5th pin, 8th pin, 4th pin, then the 2nd pin, 3rd pin, 6th pin and 7th pin. The same procedure is applied to **U43** if it is decided to install and use it. Avoid excess heat by the soldering iron to the **U42** or **U43** as it might cause damage to the ICs.

J44 is a connector header (top entry and 5 circuits) of type JST B5B-EH (LF)(SN) is soldered in the same way as **J13** keeping its front to outward for easy plugging the program writing cable, as shown in Figure 3-45. The housing connector of **J44** is mounted using the procedures described in section 3.5.6.

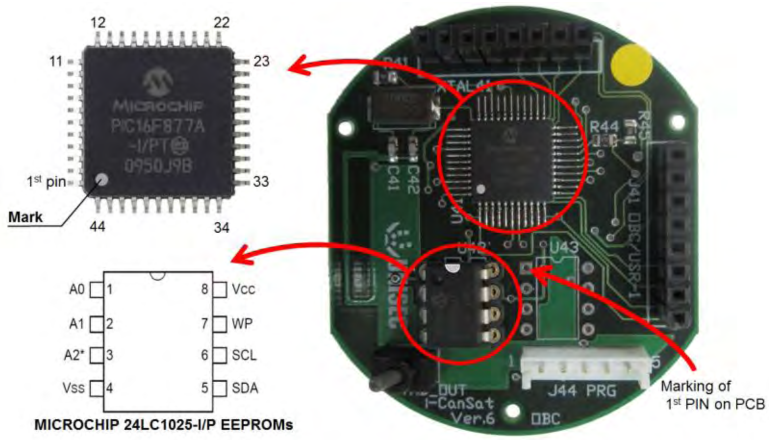


Figure 3-45: Reference pins for both PIC16LF877A-I/PT OBC and MICROSHIP 24LC1025-I/P EEPROMs.

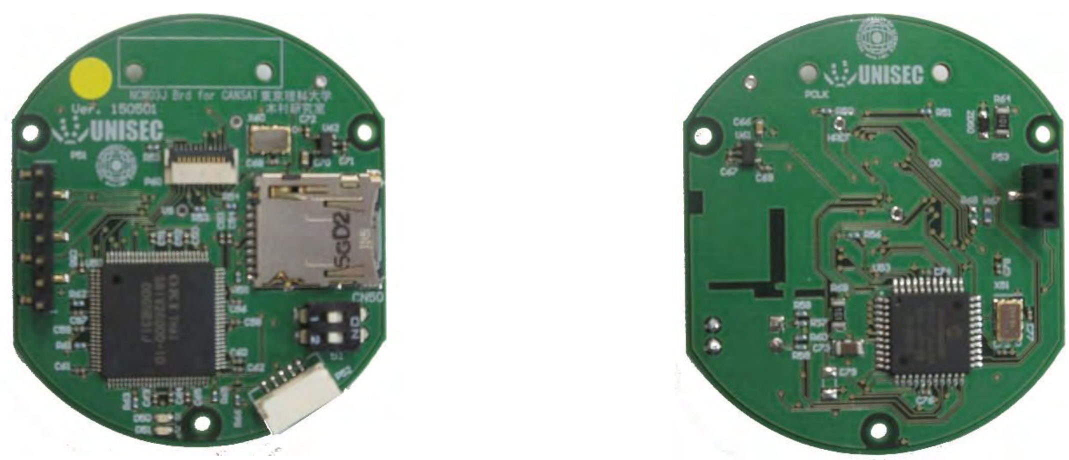
S41 is soldered at the specified position by fitting the terminals to the corresponding holes on the board.

To solder **J43** the pin socket (HIROSUGI FSM-41075-07), OBC board, the CAM board and the complete OBC/CAM-CAM/OBC interface connector (only **J43**, surface mount pin header) must be assembled using the 13 mm spacers, as shown in Figure 3-46. First solder 1st pin, 3rd pin, 5th pin and 7th pin from the outward side then disassemble the CAM board and continue soldering the remaining pins (2nd pin, 4th pin and 6th pin) from inside. Avoid touching of the soldering iron with insulator. No need to add solder as the CAM board already has but if it is not sufficient then it can be added.

Figure 3-46: Soldering the OBC/CAM - CAM/OBC interface connector (**J43**).

3.6.5 **Camera Board**

The CAM board in i-CanSat-6 was developed at Kimura's laboratory of Tokyo University of Science, shown in Figure 3-47.



a) Upper Surface b) Lower surface

Figure 3-47: CAM board before mounting the Camera.

The board has two interface connectors. The first interface is the CAM/OBC – OBC\CAM interface connector (7 pins). Its pin headers already soldered to CAM board. The second interface is the CAM/XBEE – XBEE-CAM interface connector (3 pins) and its pin sockets is also soldered to the CAM board.

In the sample program, the 4th and 5th pins represent IF2 and IF1, respectively, as shown in Figure 3-48, are not used, and all commands from OBC to CAM depends on I^2C communication bus. The 7th pin, TXD>XBee, is to enable transmitting TXD data from the OBC board to the XBee board through one pin of the CAM/XBEE – XBEE-CAM interface connector which has 3 pins (VCC, GND and TXD>XBEE).

A camera on the CAM board, CANCAM, is installed, and is available to take pictures under the modes described in Table 3-2. Table 3-3 shows the commands under I^{2}C mode (SW1=OFF, SW2=ON, of the DIP switch on the CAM board). The picture data is recorded in Micro-SD card on the CAM board.

All interfaces on the CAM board are the surface-mount type pin header and pin socket. The spacers of 13 mm in length should be used for the connection between the OBC and CAM boards as well as between the CAM and XBee boards.

When the Micro-SD card is not inserted in the card slot on the CAM board, or when undefined command is accepted by the CANCAM, a red LED on the CAM board goes on and off. If the command is correct, a green LED turn on.

All electronic parts are pre-installed to the CAM board. The only thing that has to be mounted on the CAM board is the Camera, its housing, and its cable. M2 plastic screws and nuts are used to mount the Camera inside the housing and the housing to the board. The CANCAM has a flat cable that must be inserted to the micro connector with a lock on the board, as shown in Figure 3-48.

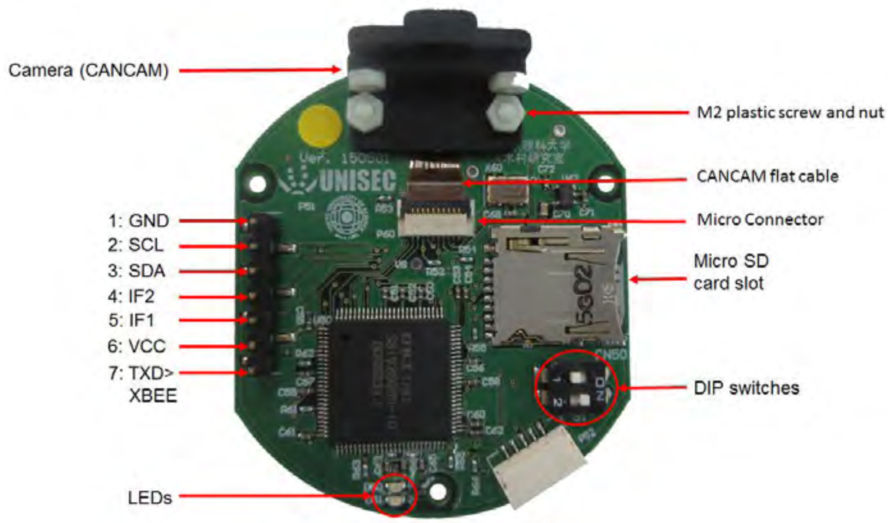


Figure 3-48: CAM board and the interface connector with OBC board pin assignment.

Table 3-2 Setting Modes for the DIP switches of CANCAM on the CAM board

SW1	SW2	Mode	Function
OFF	**OFF**	AUTO Mode	Automatic Capture Mode Images are intermittently captured and stored in Micro-SD Card when the power is supplied
ON	**OFF**	DC Mode	Discrete Control Mode Image capturing process is controlled by the General Purpose Input signal.
OFF	**ON**	I^2C Mode	I^2C Commanding Mode P1-2 is assigned as I^2C clock, and P1-3 is assigned as I^2C data.
ON	**ON**	UART Mode	UART Commanding Mode P1-2 is assigned as UART RX, and P1-3 is assigned as UART TX.

Table 3-3 Commands for CANCAM under I^2C mode

Address	Function	Description
0xF1	Start Continuous	Number of Images (0X00 for Endless)
OxF2	Stop Continuous Image Capture	0x00 (Fixed)
0xF3	Resolution Setting	0x00: High Resolution (640 × 480) 0x01: Low Resolution (160 × 120)
0x12	Imager Setting	BIT[7]-[3]: Reserved BIT[2]: Auto Gain Control 0: Manual 1: Auto (Default) BIT[1]: Reserved BIT[0]: Auto Exposure Control Setting 0: Manual 1: Auto (Default)
0x00	GAIN Setting	Gain Level (0x00 to 0x3F) (Manual Gain Mode Only)
0x10	Exposure Time Setting	Exposure Time (0x00 to 0xFF) (Manual Exposure Time Mode Only)

3.6.6 **XBee Board**

3.6.6.1 Description

The circuit diagram of the XBee board is shown in Figure 3-49. From the circuit diagram, the functions of electronic parts can be summarized as follows:

- Connectors

- **J61** is the XBEE/CAM – CAM/XBEE interface connector (3 pins). It has the VCC, GND and TXD>XBEE lines. The TXD>XBEE signal is the TXD from the OBC and the data in this TXD line is sent to the XBee communication module to transmit it wirelessly to the ground station PC.

- Electric modules
  - **U61** is the XBee communication module. It should be noted here that the frequency used in the XBee communication is radio frequency. Since the available frequencies for radio communication differ from country to country, proper XBee communication module must be selected for legal operations.
- Capacitors
  - **C61** is bypass capacitor.
  - **C62** is bypass capacitor.

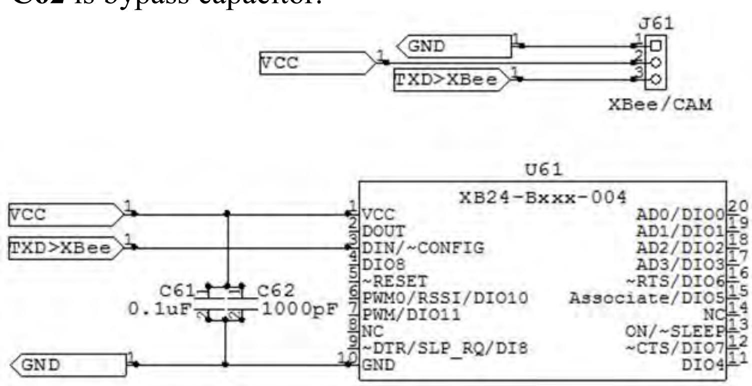
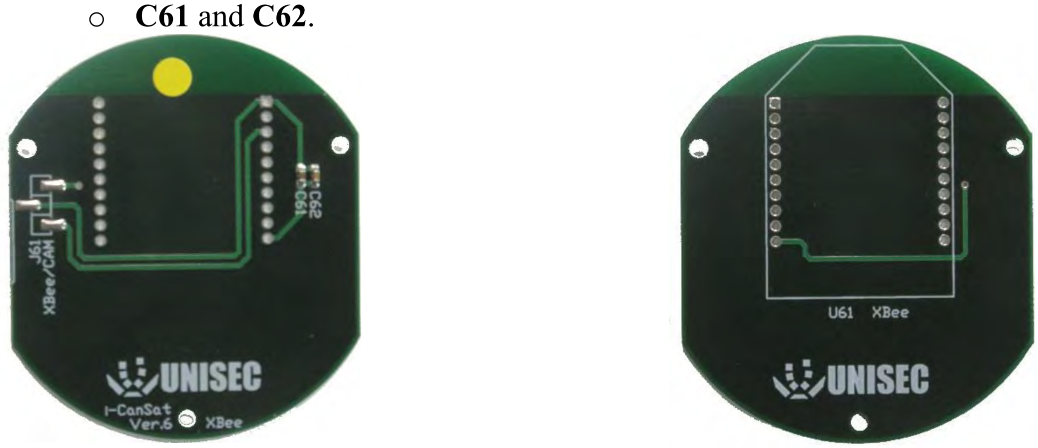


Figure 3-49: Circuit diagram of the XBee board.

3.6.6.2 Pre-installed parts

The following parts are pre-installed to the OBC board, as shown in Figure 3-50:

- Capacitors
  - **C61** and **C62**.



a) Upper surface. b) Lower surface

Figure 3-50: XBee board with the preinstalled electronic parts and before soldering the electronic parts.

3.6.6.3 Parts to be soldered or mounted

The following parts are needed to be soldered or installed on the XBee board:

- Connectors
    - **J61**
- Modules
    - **U61**

3.6.6.4 Soldering guidelines

To solder the **J61**, which is a surface mount pin header (3 pins), CAM board, the XBee board and the complete surface mount pin header (3 pins) of CAM/XBEE-XBEE/CAM interface connector must be assembled using the 13 mm spacers, as shown in Figure 3-51. All pins can be soldered without disassembly because there are no obstacles in the upper surface of the XBee board. The procedures are the same as described in section 3.6.4.43.6.4.4.

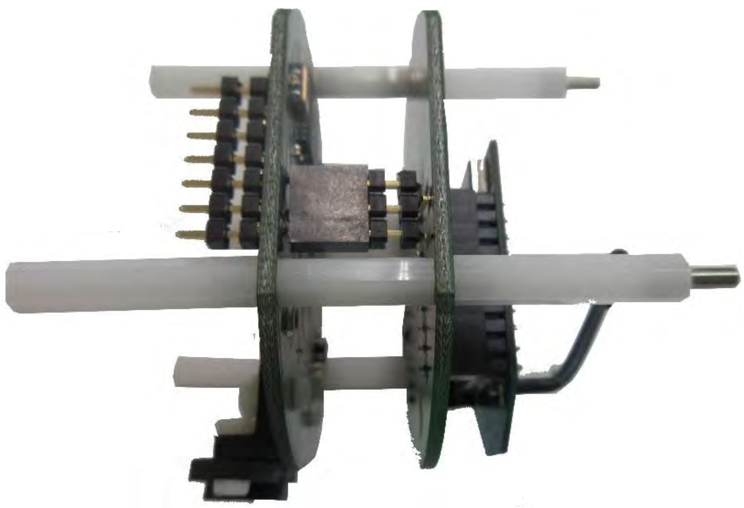
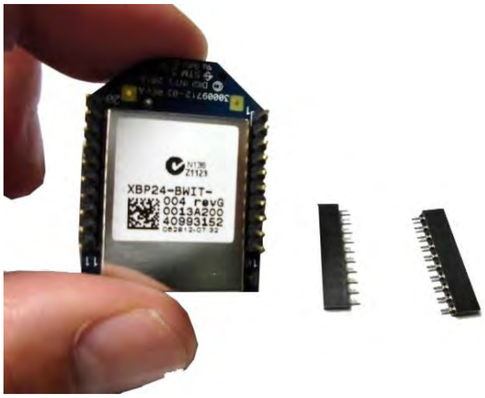


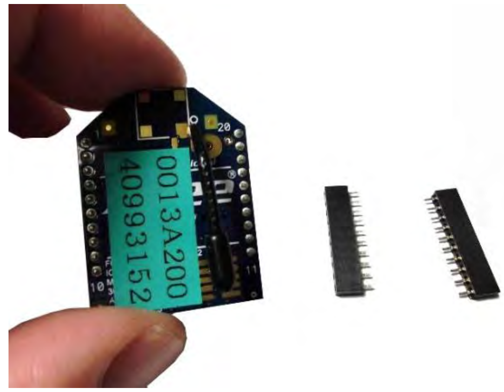
Figure 3-51: Soldering the surface mount pin header of the CAM/XBEE - XBEE/CAM interface connector.

A pair of XBee modules must be used for wireless communication between the i-CanSat-6 and ground station PC. The ground station consists of XBee module, XBee interface board and USB cable. These XBee modules should be configured prior to use them in wireless communication. The configuration procedures are presented in section 3.8.3. This process is known as XBee pairing. To conduct the configuration process two serial numbers for each XBee should be known. These serial numbers are known as SH and SL and they are unique for each XBee. The serial numbers of each XBee are printed on its back side, as shown in Figure 3-52-a.

It is recommended to print those serial numbers on the front of the XBee as shown in Figure 3-52-b. So, there will be no need to pull out the XBee from its pin sockets each time when the serial numbers are needed. This way, the XBee pins can have extended lifetime because they will not be subjected to stress during the pulling out and insertion of XBee communication module.



a) Copy the serial numbers from the back side of XBee module.

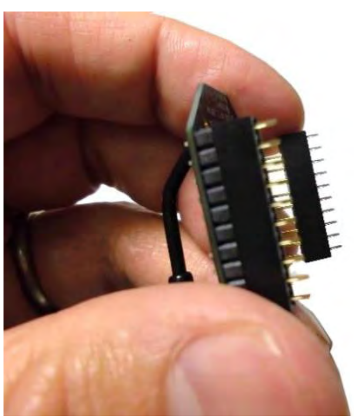


b) Write the serial numbers on the front side of XBee module.

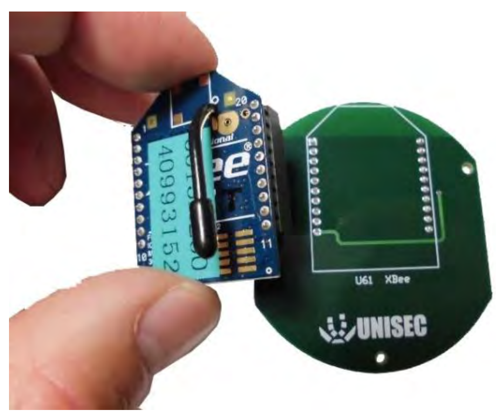
Figure 3-52: Record the serial numbers of each XBee module.

The XBee communication module is mounted on the lower surface of the XBee board via a pair of 2 mm pitch pin sockets, HIROSUGI FSS- 21043-10, as shown in Figure 3-52-b. When soldering the pin sockets onto the XBee board, the pin sockets should be soundly inserted to the XBee communication module first as shown in Figure 3-53-a, then, inserted onto the XBee board as shown in Figure 3-53-b and soldered them. The module should be temporarily held before soldering, so that the orientation of the XBee communication module aligns with the shape printed on the lower surface of the XBee board.

Afterward, the removal of the XBee communication module from the board wouldn't necessary. Since the antenna of the XBee module is delicate, excessive bending due to the removal of the XBee communication module or mishandling might cause fatigue to antenna.



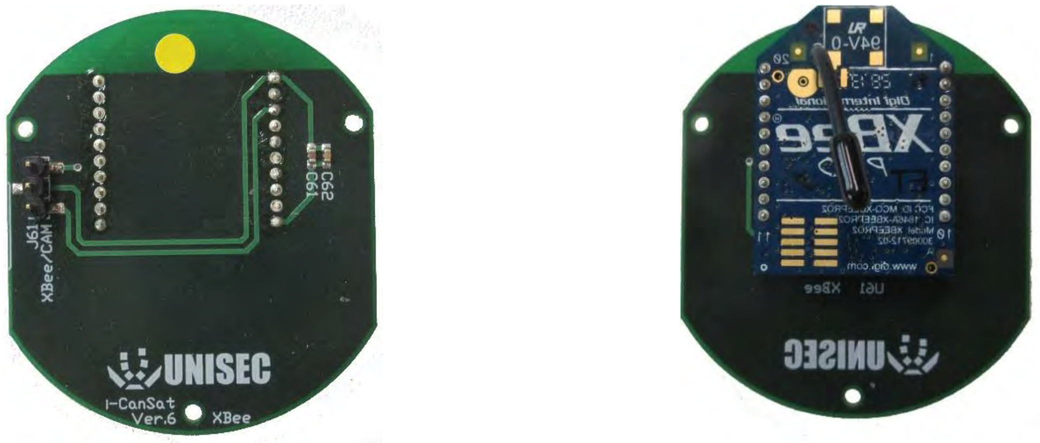
a) Insert XBee pin sockets to the XBee module.



b) Mount the XBee module to the XBee board.

Figure 3-53: Mounting the XBee module to the XBee board.

The final assembly of the XBee board is shown in Figure 3-54.



a) Upper surface b) Lower surface

Figure 3-54: Completion of the installation and soldering works of the XBee board.

3.6.7 **Cables**

A pair of cables are used in debugging and firmware development of i-CanSat-6. The first cable is USB-serial communication cable which is used for debugging. This cable is used to connect the **J13** connector on the GPS board to the PC. Three circuits EH-series connector housing need to be mount on one side of this cable using the contacts and the crimping tool as presented in section 3.5.6 and shown in Figure 3-55.

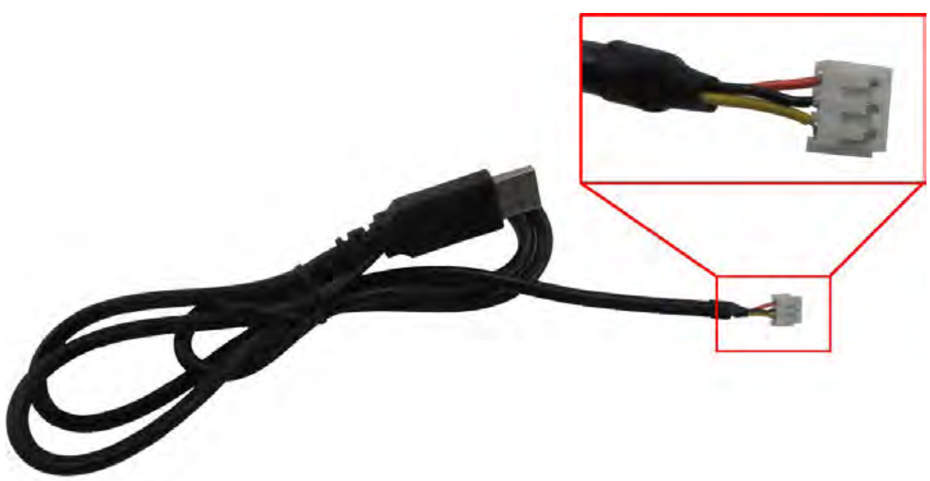


Figure 3-55: Mount the connector housing (3 circuits) to USB – Serial communication cable for PC debugging.

The second cable is the program writing cable and used for firmware development. This cable is used to connect the **J44** connector on the OBC board to PICkit3 development kit pin sockets. Five wires, colored, flat cable is used to connect the PICkit3 to the PC USB port. This colored flat cable is connected to the five circuits' EH-series connector housing using the contacts and crimping tool as presented in section 3.5.6 and shown in Figure 3-56. The other side of the colored flat cable is soldered to the five pin headers as shown Figure 3-56. Pin number 1 of the **J44** connector should be correspond to the pin socket marked with arrow in the PICkit3, as shown in Figure 3-56.

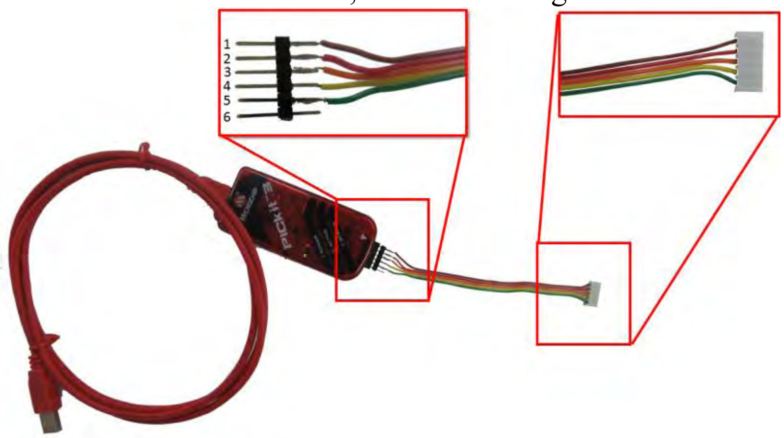


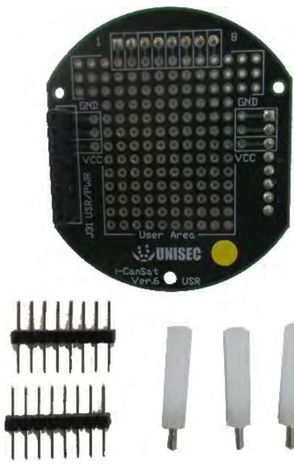
Figure 3-56: Soldering and mounting of connector housing (5 circuits) to the program writing cable through PICKit3 development kit.

3.7 Assembly, Integration and Testing of i-CanSat

After completing the soldering and installation on subsystem level or board level, the complete system assembly or i-CanSat can be done. Prepare each board with its associate spacers and pin header as shown in Figure 3-57. The spacers are used to mechanically fasten each board with respect to other boards as shown in Figure 3-2. As shown in Figure 3-2, most of the spacers are 20 mm in length. The spacers between the OBC/CAM boards and CAM/XBEE boards are of 13 mm length. The spaces attached to the lower surface of the XBee board are of 30 mm length. All spacers are male-female type except the 30 mm length spacers are female-female type.

a) GPS board, 3×20mm spacers, pin headers, and 3×M2 metal screws.

b) PWR board, 3×20 mm spacers, and pin headers.



c) USR board, 3×20 mm spacers, and pin headers.

d) OBC board and 3×13 mm spacers.

e) CAM board and 3×13 mm spacers.

f) XBee boards, 3×30 mm spacers, and 3×M2 metal screws.

Figure 3-57: Preparation for the i-CanSat Assembly and Integration.

3.7.1 **Board Continuity Test**

Before starting the assembly process, each board should pass the continuity test. Continuity testing is done to check if there is a break in a wire or track on a circuit board (open circuit). Continuity testing can also check whether a wire or track is shorted to another wire or track (short circuit). Short circuit between VCC and GND can be very dangerous. In this section, the minimum continuity tests, specifically for short circuit, are described and should be passed before assembly the i-CanSat. Open circuit means that the resistance between the two terminals should be infinity. Any high finite value of resistance, for example 10 MΩ, is considered short circuit. If any abnormal short circuit is found during the continuity test, assembly of the i-CanSat shouldn't be proceeded until the cause of short circuit is identified and removed.

A multimeter can be used for continuity testing. When the continuity function is used on a multimeter, the leads of the multimeter are placed on either end of a wire, track or terminals, the multimeter will make an audible sound if the wire or track is not broken, i.e. it is continuous. The multimeter acts as a buzzer circuit and the leads act as a switch. When the leads are touched together or on either ends of a conductor, the buzzer circuit is completed and the buzzer sounds. Figure 3-58 shows a multimeter setting and display during the continuity test. In open circuit situation, as shown in Figure 3-58-a, the multimeter is not produced any buzzer sound and the display shows the letters "OL" which means Open Leads/Loop. Figure 3-58-b shows the display during the short circuit. The display shows the value of the resistance and buzzer sound is produced.

a) Open circuit b) Closed circuit

Figure 3-58: The multimeter setting and display during the continuity test.

The minimum continuity tests to confirm that the connectors and boards are free from any short circuit are as follow:

1) GPS board:
   a) All the terminals combinations of **J11** connector (1^{st} pin, 2^{nd} pin, and 3^{rd} pin).
2) PWR board:
   a) All the terminals combinations of **J21** connector (1^{st} pin, 2^{nd} pin, and 3^{rd} pin).
   b) 1^{st} (or 2^{nd}) pin and 3^{rd} pin of **J22**.
   c) 1^{st} and 2^{nd} pins of **J23**.
   d) 1^{st} and 2^{nd} pins of **J24**.
3) USR board:
   a) 1^{st} (or 2^{nd}) pin and 3^{rd} pin of **J31**.
   b) 1^{st} (or 2^{nd}) pin and 3^{rd} pin of **J32**.
4) OBC board:
   a) 1^{st} (or 2^{nd}) pin and 3^{rd} pin of **J41**.
   b) 1^{st} and 6^{th} pins of **J43**.
   c) 2^{nd} and 3^{rd} pins of **J44**.
5) CAM board:
   a) 1^{st} and 6^{th} pins of the 7 pins socket on the upper surface.
   b) 1^{st} and 2^{nd} pins of the 3 pins header on the lower surface.
6) XBee board:
   a) 1^{st} and 2^{nd} pins of **J61**.

It is highly recommended to conduct the continuity test for all terminals and combinations. Circuit diagrams are considered the main references to conduct the continuity tests.

3.7.2 **Assembly and Integration**

The i-CanSat assembly can be done after passing the continuity test in section 3.7.1. The following rules should be kept in mind before starting the assembly of the i-CanSat.

- The assembly can start from any board in the i-CanSat. It can start from GPS boards down to the XBee board or vice versa. The sequence must be decided and never change during the execution. This will minimize any human related errors.
- When assembly boards together, the priority should be given to mating the pin headers and pin sockets in all board interface connectors. Then arrange the position of the spacers. It is recommended not to fasten the board's spacers tightly until the lower or upper board assembled to the same spacers.
- The whole assembly should be conducted without the power supplied to the interface connectors. Any short circuit between the VCC and GND terminals may damage to the i-CanSat. The battery can be disconnected by disconnected the **J23** or **J24** connector housing or by disconnecting the battery clip snap connector. This way any short circuit that may occur during the assembly will not damage the i-CanSat.
- Don't use metallic drivers or metallic pincette to pull out or insert any electric connectors.
- The power switch must be in off position before connecting the battery to the i-CanSat, as shown in Figure 3-59.

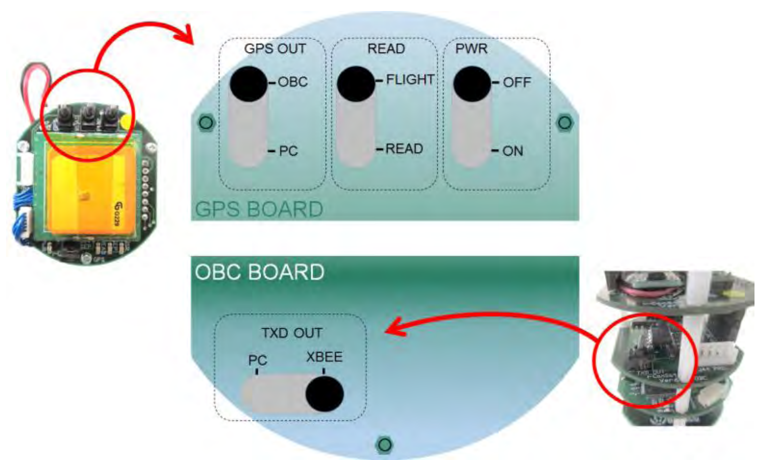


Figure 3-59: Switches setting in GPS and OBC boards

Here is an example of i-CanSat assembly procedures, from the upper board, GPS board, to the lower board, XBee board.

1. Set the position of the switches on the GPS board to be as indicated in Figure 3-59.
2. Assembly the 3×20 mm spacers to the GPS board with the screw driver and don't fasten the screws tightly until the PWR board is assembled to the GPS board as shown in Figure 3-60. This will result in the GPS board subassembly.

Figure 3-60: Assembly of the 3×20 mm spacers to the GPS board: GPS board subassembly.

3. Assembly the PWR board to the GPS subassembly using 3×20 mm spacers, as shown in Figure 3-61. Remember to unplug the battery before assembling the PWR board. Mating the pin headers and pin sockets of the GPS/PWR-PWR/GPS interface connector first. The GPS spacers can be fastened tightly by the screws. This will result in the PWR-GPS subassembly.

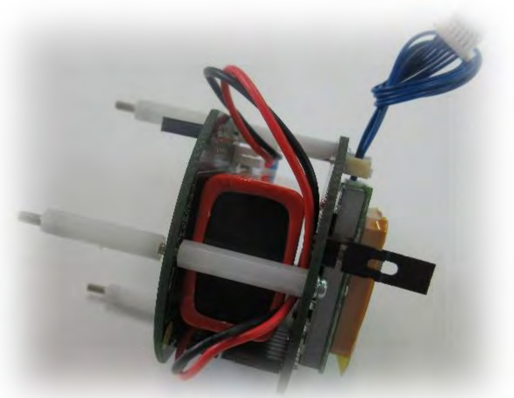


Figure 3-61: Assembly of the PWR and 3×20 mm spacers to the GPS board subassembly: PWR-GPS subassembly.

4. Assembly the USR board to the PWR-GPS subassembly using 3×20 mm spacers, as shown in Figure 3-62. Mating the pin headers and pin sockets of the PWR/USR-USR/PWR interface connector first. The PWR spacers can be fastened tightly. This will result in the USR-PWR-GPS subassembly.

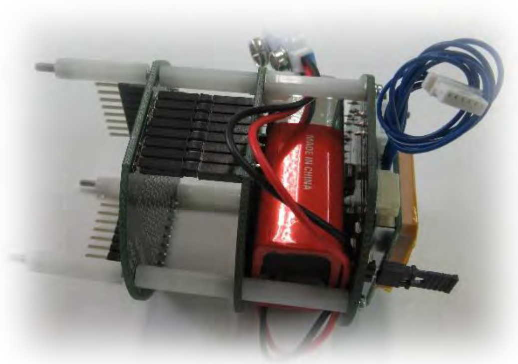


Figure 3-62: Assembly of the USR board and 3×20 mm spacers to the PWR-GPS subassembly: USR-PWR-GPS Subassembly.

5. Assembly the OBC board to the USR-PWR-GPS subassembly using 3×13 mm spacers, as shown in Figure 3-63. Mating the pin headers and pin sockets of the USR/OBC-1 - OBC/USR-1 and USR/OBC-2 - OBC/USR-2 interface connectors first. The USR spacers can be fastened tightly. This will result in the OBC-USR-PWR-GPS subassembly.

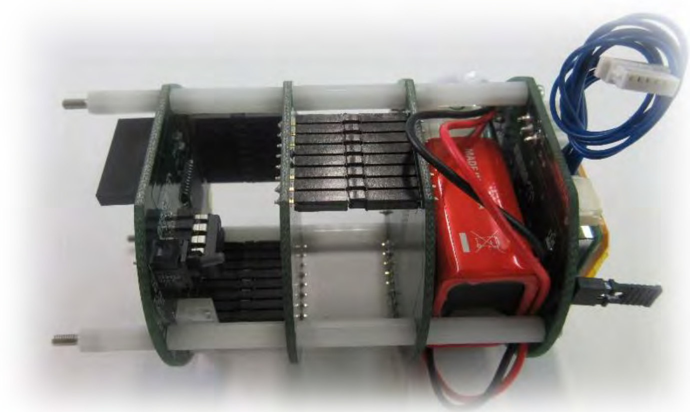


Figure 3-63: Assembly of the OBC board and 3×13 mm spacers to the USR-PWR-GPS subassembly: OBC-USR-PWR-GPS Subassembly.

6. Assembly the CAM board to the OBC-USR-PWR-GPS subassembly using 3×13 mm spacers, as shown in Figure 3-64. Mating the pin headers and pin sockets of the OBC/CAM-CAM/OBC interface connector first. The OBC spacers can be fastened tightly. This will result in the CAM-OBC-USR-PWR-GPS subassembly.

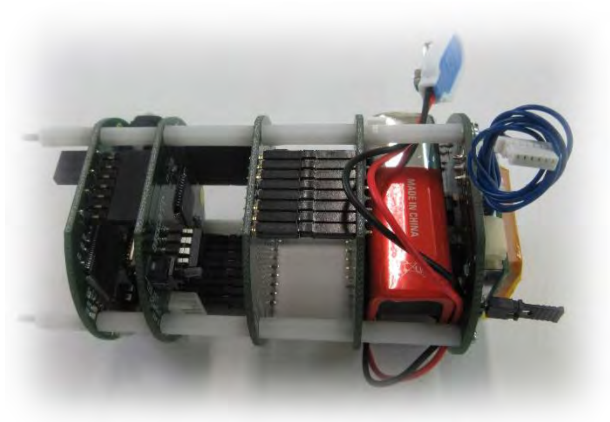


Figure 3-64: Assembly of the CAM board and 3×13 mm spacers to the OBC-USR-PWR-GPS subassembly: CAM-OBC-USR-PWR-GPS Subassembly.

7. Assembly the XBEE board to the CAM-OBC-USR-PWR-GPS subassembly using 3×30 mm spacers, as shown in Figure 3-65. Mating the pin headers and pin sockets of the CAM/XBEE-XBEE/CAM interface connector first. The CAM spacers can be fastened tightly. This will result in the final i-CanSat assembly as shown in Figure 3-66.

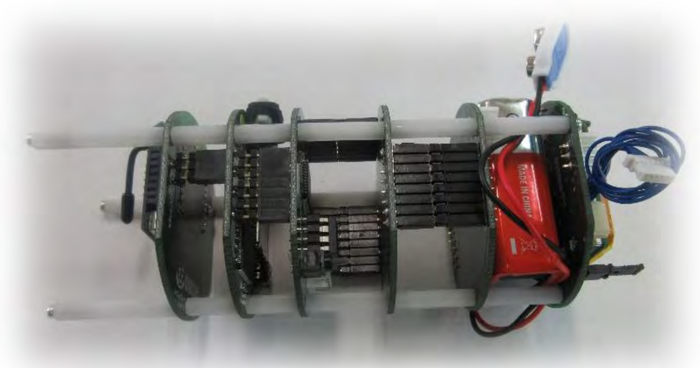


Figure 3-65: Assembly of the XBee board and 3×30 mm spacers to the CAM-OBC-USR-PWR-GPS subassembly: Final i-CanSat Assembly.

Figure 3-66: The final assembly of i-CanSat.

3.8 i-CanSat Operations

In this section, the i-CanSat operations are described. Program writing skills are not required to conduct most of the tests in this section. Firstly, functionality tests of different switches are presented. Secondly, the GPS interface with PC is described and the configuration of the GPS is presented. Thirdly, configuration of the XBee to interfacing the PC and i-CanSat, wirelessly, is presented.

3.8.1 Switches and LEDs

There are four switches installed in the i-CanSat-6. Three switches are installed on the GPS board and one switch on the OBC board, as shown in Figure 3-59. Their functions are as follows:

1. **Power switch**: this switch is used to turn the power on and off.
2. **Read switch**: this switch is used to specify the operation mode to be either Flight (FLT) mode or Read (READ) mode. In FLT mode, the collected data from GPS receiver and the sensors/actuators installed on the USR board will be stored in the EEPROM and can be simultaneously transmitted the data to PC through USB cable, shown in Figure 3-55, inserted in **J13** connector or through XBee wireless communication to the ground station depending on the setting of the TXD_OUT switch. This mode is used when releasing the CanSat from the carrier or for debugging a program in FLT mode uploaded to the OBC. The READ mode is used to read the data from the EEPROM and send it to PC through USB cable, shown in Figure 3-55, inserted in **J13** connector or through XBee wireless communication to the

ground station depending on the setting of the TXD_OUT switch. The READ mode can be used also for debugging a program in READ mode.

3. **GPS_OUT switch**: this switch is used is to send the GPS data to either the OBC or PC through the USB cable, shown in Figure 3-55, inserted in **J13 connector**.
4. **TXD_OUT switch**: this switch is used to select the destination of the data transfer from OBC to either PC, through a USB cable, shown in Figure 3-55, inserted in **J13** connector or XBee wireless communication to the ground station, through the ground station interface board.

Figure 3-67 shows the switch settings for power ON (power switch setting), READ mode (READ switch setting), GPS data transmitted to OBC (GPS_OUT switch setting) and the OBC data is transmitted to the PC (TXD_OUT switch setting) through USB serial cable inserted in **J13** connectors.

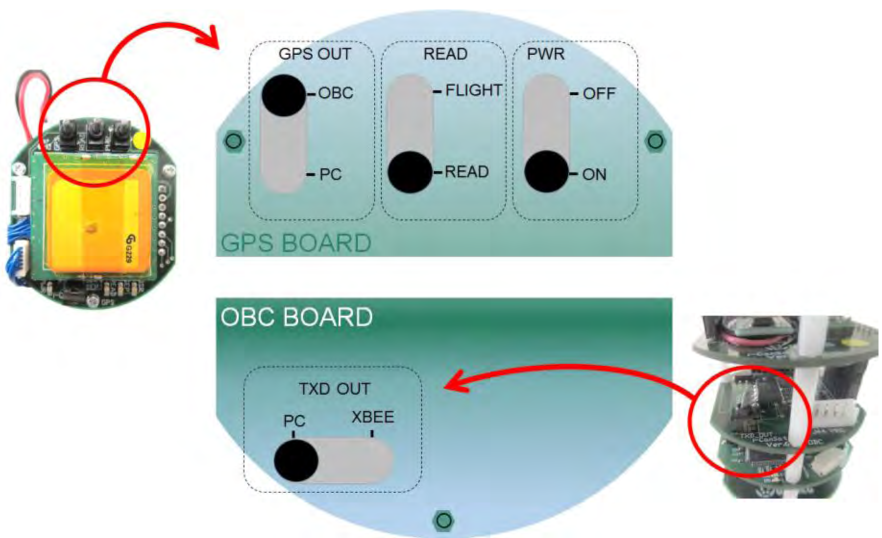
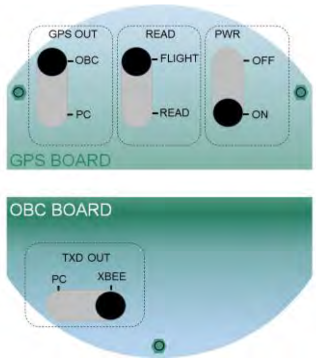


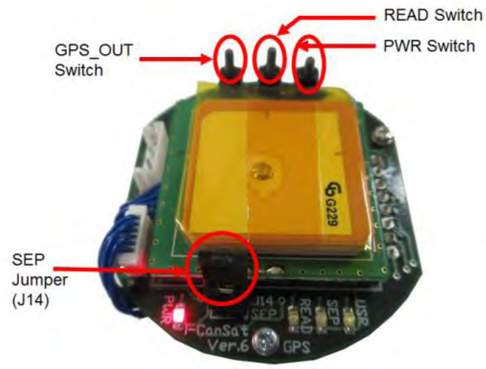
Figure 3-67: Switch settings for PWR on, READ mode, GPS_OUT to OBC, and OBC transmit data to PC.

The following simple tests can be conducted when the battery is installed and the PWR switch is set to ON. These tests will confirm that the hardware is assembled correctly.

1. Test #1: Turn the PWR switch on, set the READ switch to the FLT mode, set the GPS_OUT to OBC, the TXD_OUT switch to XBEE and insert the SEP jumper **(J14)** as shown in Figure 3-68-b. The switch setting is shown in Figure 3-68-a.
   The result: LED of the PWR will light up, as shown in Figure 3-68-b. If the Micro-SD card is not inserted then the LED on the CAM board will blink. If it is inserted then the LED will turned off.



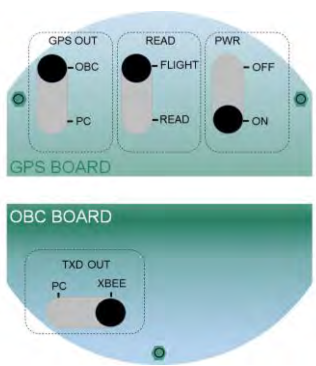
a) Switch setting



b) LED status

Figure 3-68: Switch setting and LED status of test #1.

2. Test #2: Repeat test#1 with the SEP jumper (**J14**) is pulled up.
   The result: The PWR and SEP LEDs will light up, as shown Figure 3-69-b. If the Micro-SD card is not inserted then the LED on the CAM board will blinking. If it is inserted then the LED will turned off.

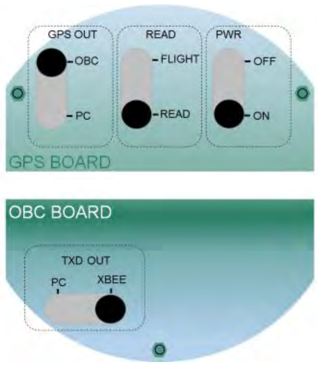


a) Switch setting

b) LED status

Figure 3-69: Switch setting and LED status of test #2.

3. Test #3: Turn the PWR switch on, set the READ switch to the READ mode, set the GPS_OUT to OBC, the TXD_OUT switch to XBEE and the SEP jumper **(J14)** is pulled up as shown in Figure 3-70-b. The switch setting is shown in Figure 3-70-a.
   The result: The PWR, SEP, and READ LEDs will light up, as shown Figure 3-70-b. If the Micro-SD card is not inserted then the LED on the CAM board will blinking. If it is inserted then the LED will turned off.



a) Switch setting

b) LED status

Figure 3-70: Switch setting and LED status of test #3.

Note that if the SEP jumper (**J14**) is inserted, the voltage of the SEP signal will become GND and the SEP LED is turned off. This can also be explained from Figure 3-30. If the jumper is connected to one of the parachute lines and inserted into the SEP terminals (**J14**), the voltage of the SEP signal will become low and the SEP LED is turned off. When i-CanSat is released from the carrier, the parachute will be deployed and the jumper will be pulled out from the SEP terminals (**J14**) by the parachute. Therefore, the voltage of the SEP signal will become high and the SEP LED will light up.

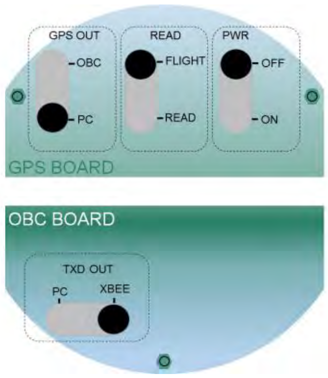
The USR LED is off because there is no program in the OBC at this moment. The USR LED can be set on or off from a program as will be presented in section 3.9.2.

3.8.2 GPS

The GPS employed in the i-CanSat is a standard product which outputs the data based on NMEA 0183 format, and it outputs Global Positioning System Fix Data (GGA), Geographic Position, Latitude and Longitude (GLL), and other factory default outputs.

Usually information about the set of time, latitude and longitude, and the number of satellites is sufficient for the CanSat flight. To do so, the GPS needs to be set to output as GGA data only. In addition, the baud rate for serial communication of GPS with PC should be set to 9,600 bps if the factory setting has different baud rate. These settings can be done from the PC using both the USB serial communication cable, shown in Figure 3-55 and terminal software application. The commonly used terminal software applications are Tera Term and Tuna Term. In this manual, the interface of GPS with PC using the Tuna Term is explained.

Usually the factory setting baud rate of GPS is 4,800 bps but if it is 9600 bps then no need to configure the baud rate of GPS and skip this step. To access the GPS from the PC. Firstly, download and install the Tuna Term from the Internet. Secondly, connect the USB-serial communication cable shown in Figure 3-50 to the **J13** connector of the GPS board as shown in Figure 3-71-b and the other terminal of the cable to the USB port of PC. Then, set the switches as are shown in Figure 3-71-a.



a) Switch setting b) Serial cable connection to **J13**.

Figure 3-71: Switches setting and serial cable connection for GPS configuration.

Open the Tuna Term application and select "Settings" from the top menu then choose "Connection Settings…" from the drop-down menu, as shown in Figure 3-72. In the "Connection Settings." window select the applicable COM port used by USB-serial communication cable, the baud rate of 4800 is assumed, data length is "8" bits, stop bit is "1", parity and flow control are both set to "None". The connection profile for specific hardware can be created so that no need to configure the serial port each time it is accessed. To create the connection profile, click the "Terminal Settings…" in the "Connections Settings" windows, as shown in Figure 3-73. In the "Terminal Setting" window, click "Add New Profile" and then type its name, as shown Figure 3-74 and Figure 3-75.

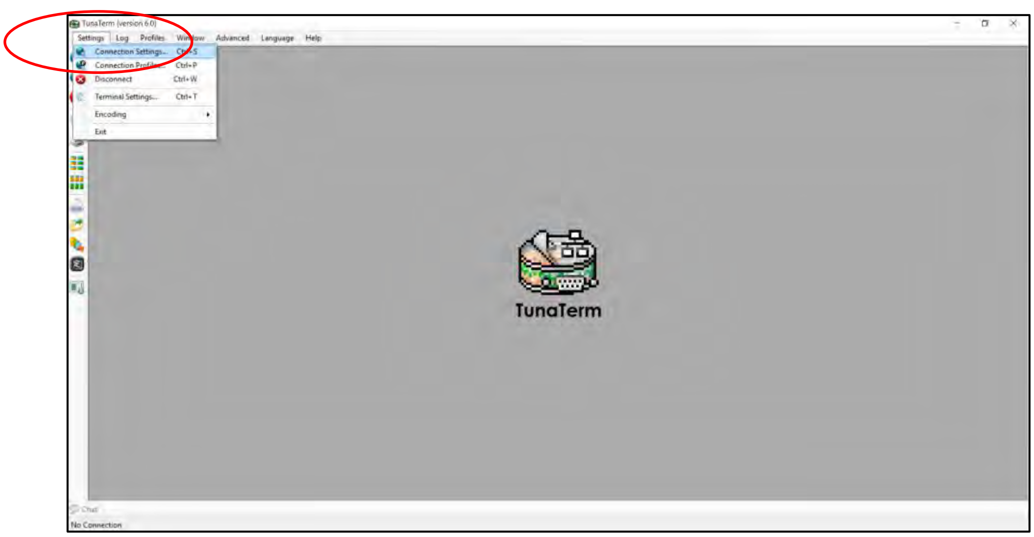


Figure 3-72: The Tuna Term, terminal software application.

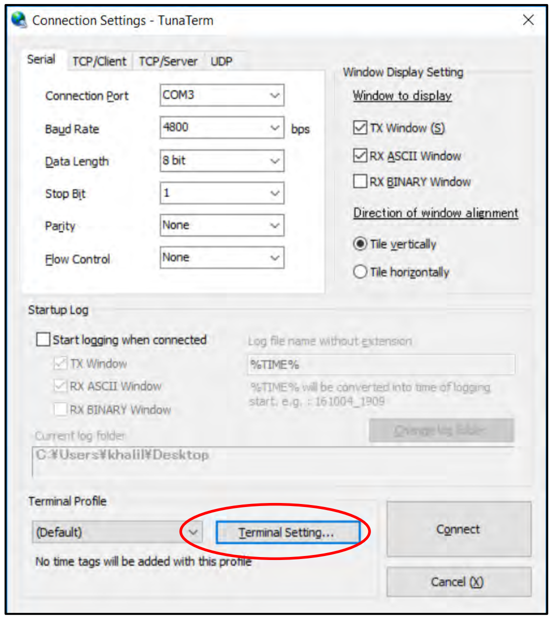


Figure 3-73: Tuna Term connection setting.

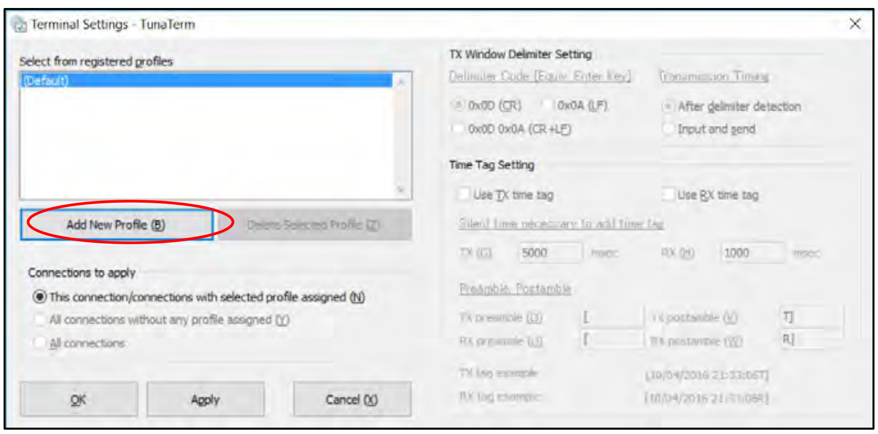


Figure 3-74: Save Tuna Term settings in a profile.

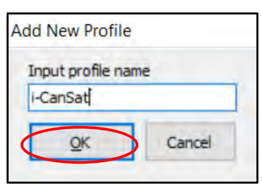


Figure 3-75: Enter the profile name.

The delimiter is a set of character codes used to define the stream of data strings. It can be 0x0D (CR) or 0x0A (LF), or 0x0D 0xoA (CF+LF). In NMEA 0183 format, the delimiter is defined as CR+LF, so choose "0x0D 0x0A (CR+LF) in the TX window Delimiter Setting within the "Terminal Settings", as shown in Figure 3-76. Finally click "Apply" then "OK". The created terminal profile is shown in the "Connection Settings" window. To connect, click "Connect" button, as shown in Figure 3-77.

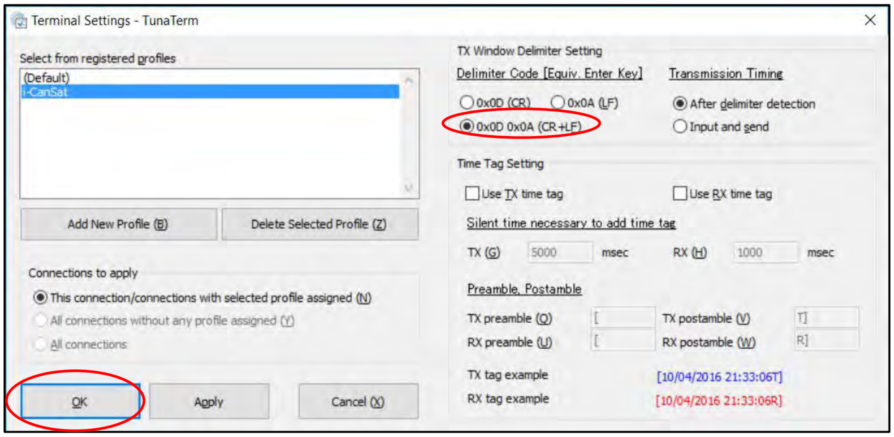


Figure 3-76: Select the delimiter setting.

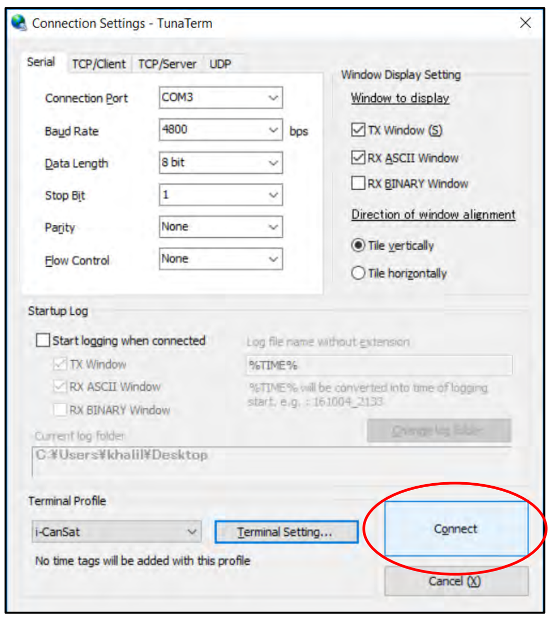


Figure 3-77: Start the connection.

The Tuna Term is connected to the GPS and both the TX window (at the top) and RX window (at the bottom) are displayed as shown in Figure 3-78.

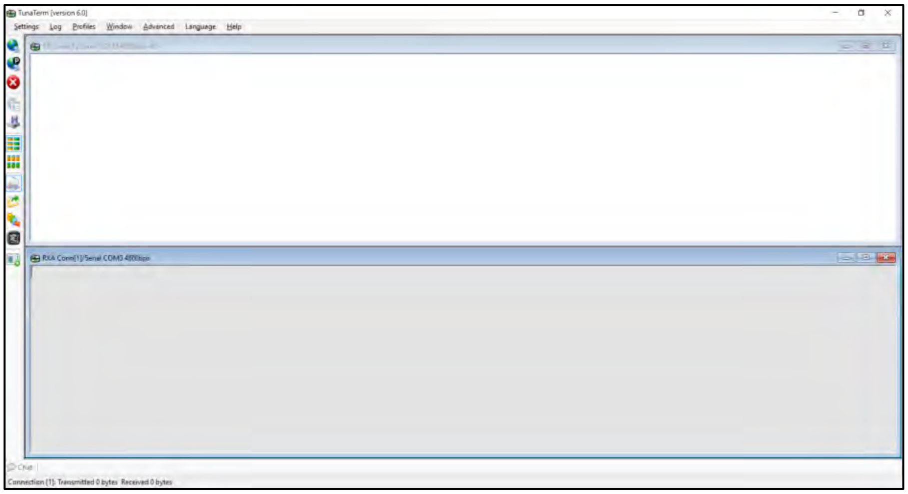


Figure 3-78: Tuna Term connected at 4800 bps without GPS data to display.

Now to start receiving the GPS data by turning the power switch to on. The switches setting in the GPS board are shown in Figure 3-79.

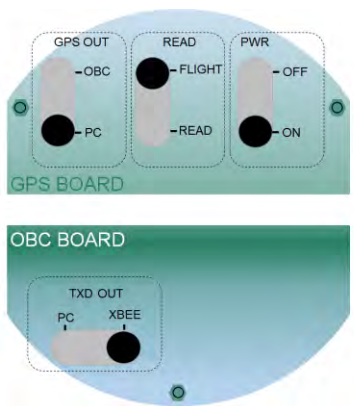


Figure 3-79: Set the switches setting on the GPS to start receiving the GPS data via the USB serial Cable.

The data received under this baud rate value of 4800 bps are shown in Figure 3-80. The data is corrupted. This means that the baud rate set in Tuna Term and that of the GPS are not matched. Disconnect and re-connect with baud rate to 9600 bps with same terminal profile, as shown in Figure 3-77. The output of the GPS using this baud rate is shown Figure 3-81. The data can be shown correctly which means that the default baud rate of the GPS is 9600 bps.

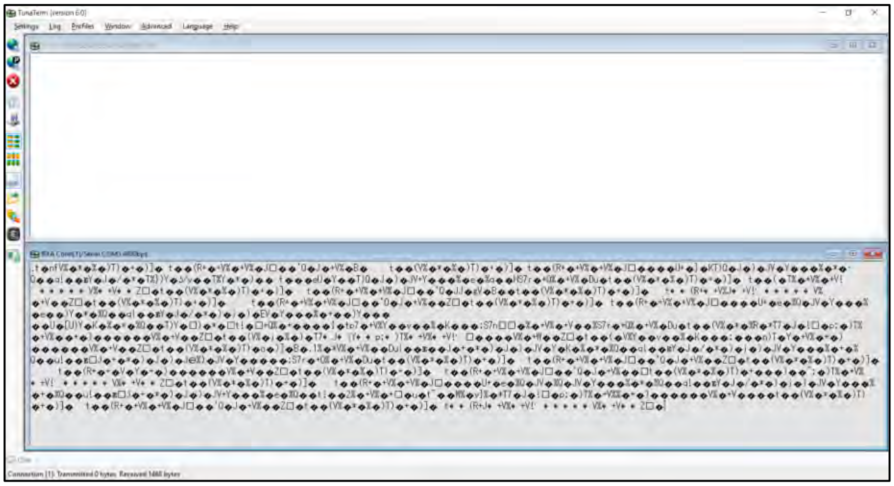


Figure 3-80: The GPS output for baud rate of 4800 bps of the terminal application.

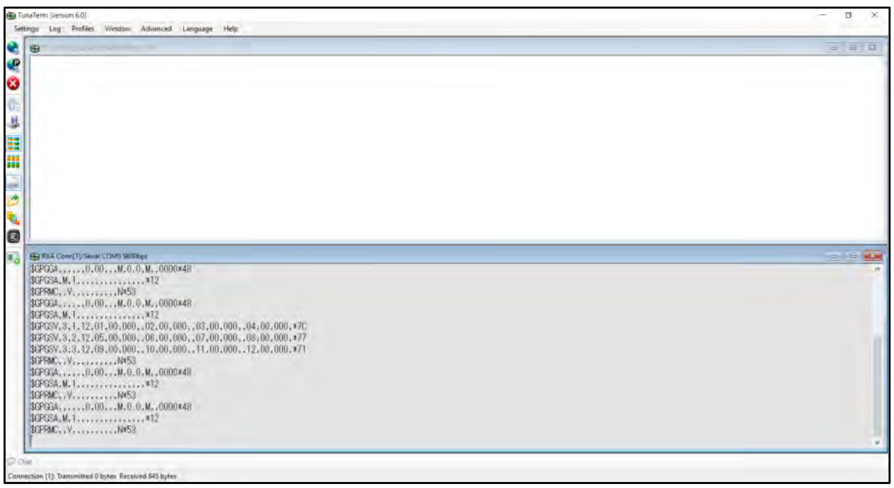


Figure 3-81: The GPS output for baud rate of 9600 bps of the terminal application.

In the sample program(s) which will be described in section 3.9, the serial port is configured to 9600 bps. If the GPS send data with different baud rate from 9600 bps then it must be changed to 9600 bps to run the sample program(s). To change the baud rate of the GPS, the following command should be sent by the Tune Term by typing it in the TX window.

$PSRF100,1,9600,8,1,0*0D <CR><LF>

Note that the command is between the"$" and "D". The <CR> and <LF> are the delimiter send by the program and not typed by the user which is discussed above.

As shown in Figure 3-81, the GPS data contains not only GGA data but also GSA, GSV, and RMC. To stop the GSA, GSV, and RMC data specific commands must by send by TX window, as shown in Figure 3-82.

To stop GSA output, type the following command
$PSRF103,2,0,0,1*26 <CR><CF>

To stop the GSV output type the following command

$PSRF103,3,0,0,1*27 <CR><CF>

To stop the RMC output, type the following command

$PSRF103,4,0,0,1*20 <CR><CF>

The output of the GGA only is shown in Figure 3-82. Note that the settings done in this section for the GPS are not permanent and usually the GPS resets to the factory defaults after a couple of days. Therefore, the GPS baud rate must be changed to 9600 bps before using the i-CanSat.

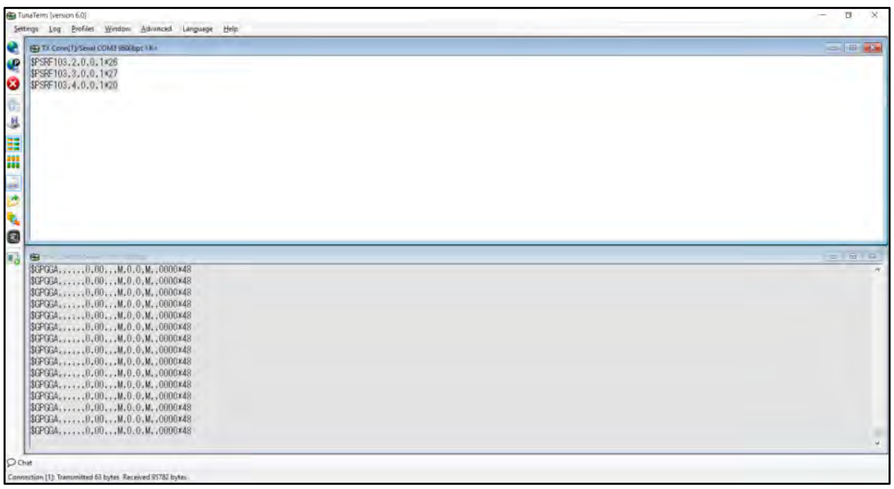


Figure 3-82: The GPS output after executing the commands to send only GGA data.

During the CanSat configuration and operation, it is very convenient to save the displayed data in the RX window or TX window for debugging and analysis purposes. The displayed data can be save in ASCII file called the log file. To start the saving process, click the “Log” option in the top menu and choose the “Open Log…” from the drop-down menu, as shown in Figure 3-83. The “Log Setting” window is then displayed to specify which windows to save its displayed stream of data and to enter the log file name, as shown in Figure 3-84. The default file name is based on the current data and time. To terminate saving of the displayed data, click the “Log” option in the top menu and choose the “Close Log” from the drop-down menu, as shown in Figure 3-85. The log file can be open by any text editor and Excel.

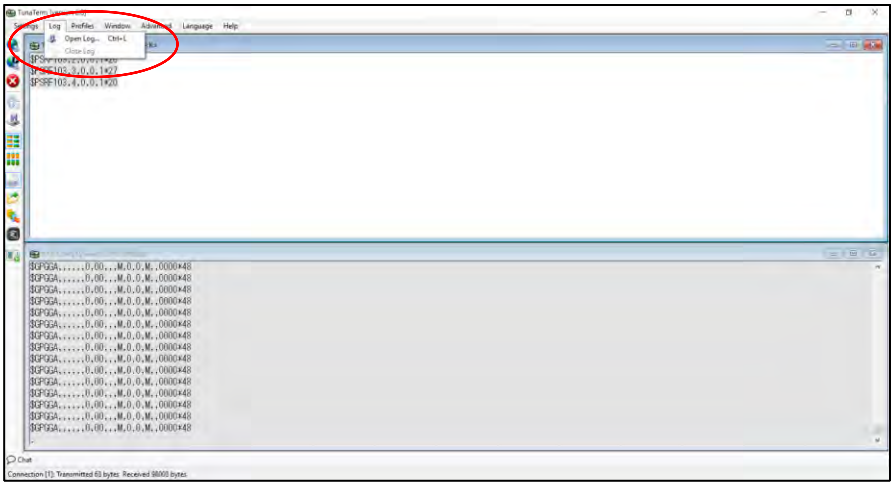


Figure 3-83: Open a log file to save the GPS output.

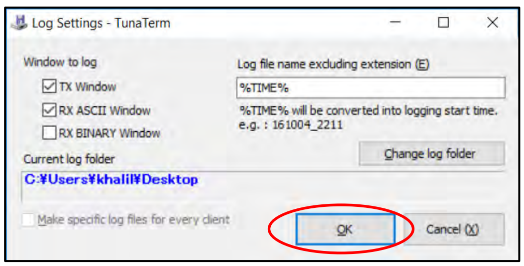


Figure 3-84: The log file settings.

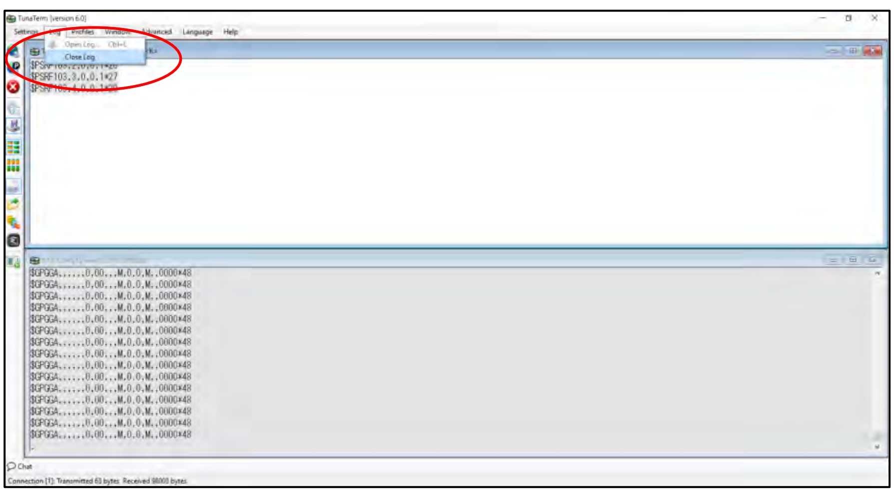


Figure 3-85: Close the log file and terminate data saving.

3.8.3 **XBee Configuration**

In this section, the pairing process of the XBee communication module in the i-CanSat and the Xbee communication module in the Ground Station (GS) interface board are described. At the beginning, the two serial numbers in each XBee communication module must be recorded. The XBee communication module in the i-CanSat is named as "Module **A**" and the XBee communication module in the ground station interface board is named "Module **B**". Figure 3-86 shows the back side of the two XBee modules. The upper serial is abbreviated as "**SH**" and the lower serial is abbreviated as "**SL**". Table 3-1 shows the typical serial numbers of the two XBee modules which are configured in this section. Figure 3-87 shows the ground station interface board with module "**B**" installed and connected to the ground station PC.

Figure 3-86: The serial numbers on the back of the XBee Modules.

Table 3-4: Typical values of XBee modules' serial numbers.

Parameter	Module A (i-CanSat)	Module B (Ground Station)
SH	0013A200	0013A200
SL	40A6F78E	40A6F88E

Figure 3-87: Ground station interface board with XBee communication module, "**B**", installed and

connected to the PC.

The setting procedures for module "**A**" and module "**B**" are almost the same. At the beginning the "X-CTU" configuration and test utility software is downloaded from the Digi International Inc. website, then install the application to the ground station PC. The following instructions in this section are based on version 6.3.1 of the X-CTU.

Plug module "**B**" into the ground station interface board and connect it to the ground station PC via a USB cable. Start the X-CTU and click the "Add" icon as shown in Figure 3-88. Configure the serial port of the ground station interface board as shown in Figure 3-89. To identify the port number associated to the ground station interface board, access the device manager from the control panel of the PC operating system. After setting the parameters, as shown in Figure 3-89, of the serial port then click "**Finish**". The window shown in Figure 3-90 is resulted.

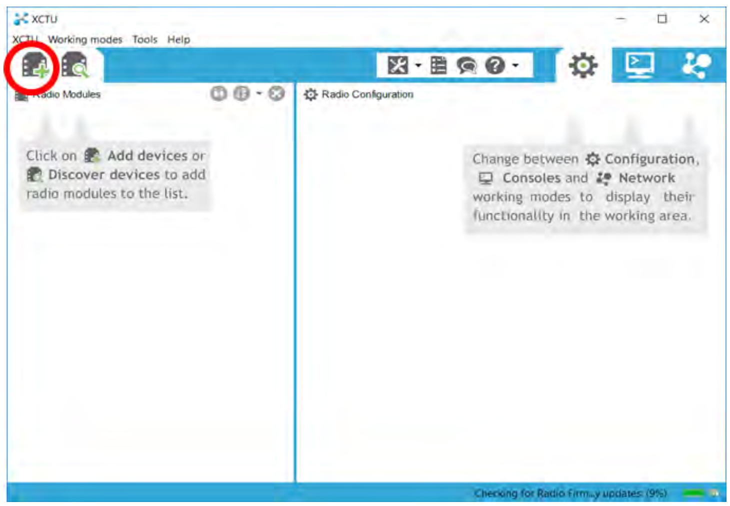


Figure 3-88: X-CTU interface window.

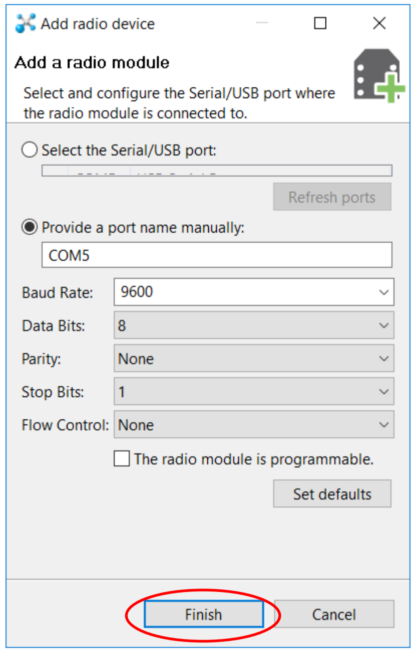


Figure 3-89: Configure the serial communication with the ground station interface board.

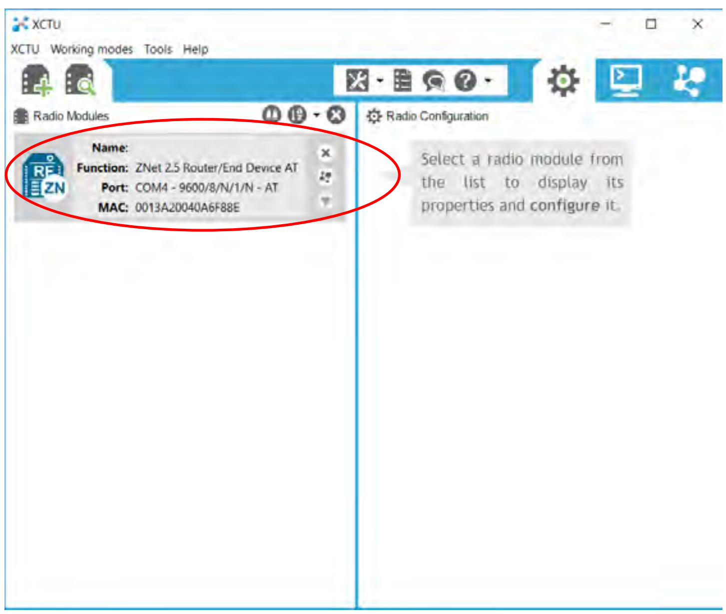


Figure 3-90: X-CTU windows after successful communication with module "**B**".

Double click the discovered radio module which is shown in Figure 3-90. The window presents all the configuration parameters of module "**B**" is shown Figure 3-91.

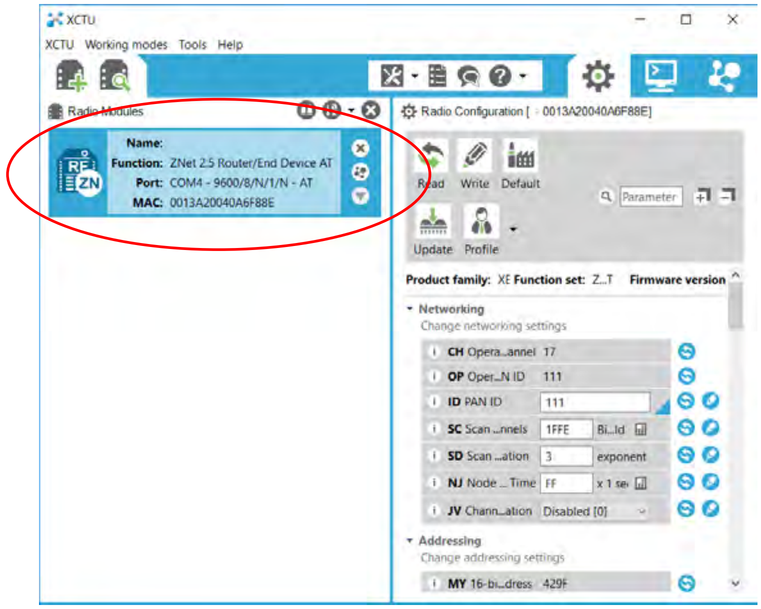


Figure 3-91: Configuration parameters of module “**B**” read by X-CTU.

Looking through all the configuration parameters under the addressing group, the recorded serial numbers, “**SH**” and “**SL**”, of module “**B**” can be seen, as shown in Figure 3-92. The function of module “**B**” must be Router/End Devices, which is the default. Here, the “**SH**” and “**SL**” of module “**A**”, as listed in Table 3-4, are written in the “**DH**” and “**DL**” of module “**B**”, respectively, in the entry boxes, as shown in Figure 3-92. Then click the “**Write**” icon as shown in Figure 3-93. The colored triangles next to the two entry boxes will change color from green, as shown in Figure 3-92, to blue, as shown in Figure 3-93. This confirms the successful writing operation and completion the configuration of module “**B**”. It should be noted here that in the networking group of the configuration parameters of module “**B**”, the PAN ID is “111”. The PAN ID of module “**A**” must be the same. The PAN ID must be changed if there are multiple i-CanSats present in the same location, i.e., each pair must have same PAN ID that is different from others.

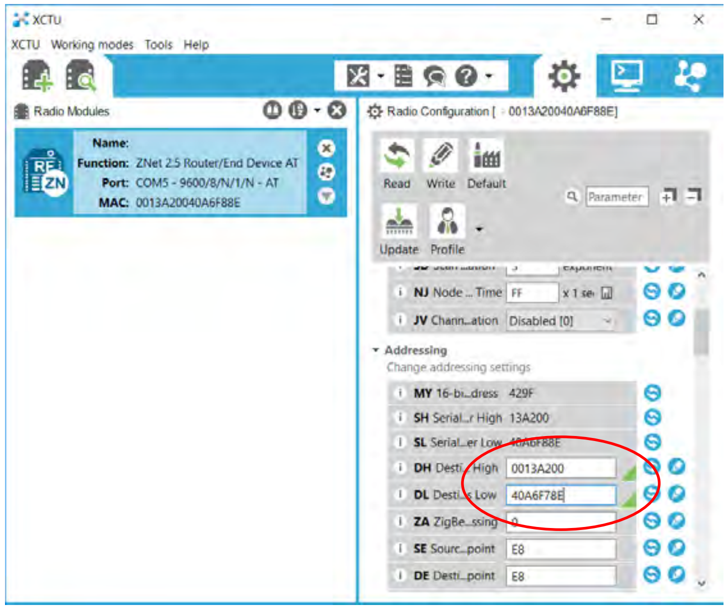


Figure 3-92: Configuring module "**B**".

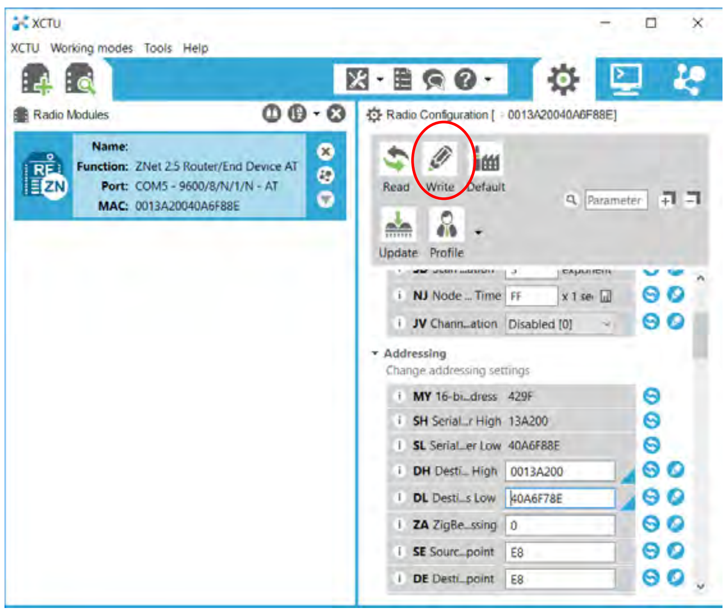


Figure 3-93: Write the new destination addresses for module "**B**".

The same procedures are repeated for module "**A**". Figure 3-94 shows configuration parameters of module "**A**". The "**SH**" and "**SL**" of module "**B**", as listed in Table 3-4, are written as "**DH**" and "**DL**", respectively, as shown in Figure 3-94, then click the "**Write**" icon. The function of module "**A**" must be Coordinator AT. However, the default is Router/End Device, as shown in Figure 3-94. This must be changed. To change the function of module "**A**", click the "**Update**" icon, as shown in Figure 3-95. Then from the "**Update Firmware**" window, select the product family, function set, and firmware version, as shown in Figure 3-96, then click "**Update**" button. After successful firmware update, the windows in Figure 3-97 will appear then click "**OK**". It is observed that the function of module "**A**' is changed from Router/End Device to Coordinator AT, as shown in Figure 3-98. Finally, check if the "**DH**" and "**DL**" addresses of module "**A**" are changed during the update process or not.

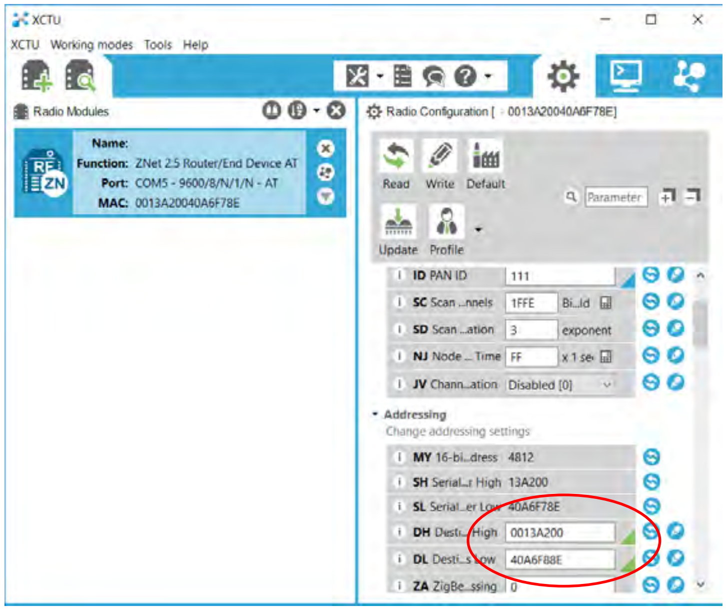


Figure 3-94: Configuring module "**A**".

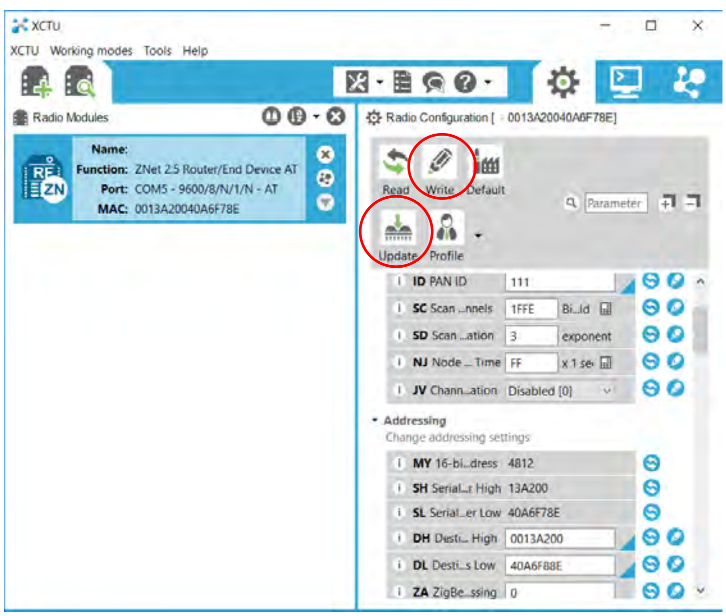


Figure 3-95: Write the new destination addresses for module "**A**".

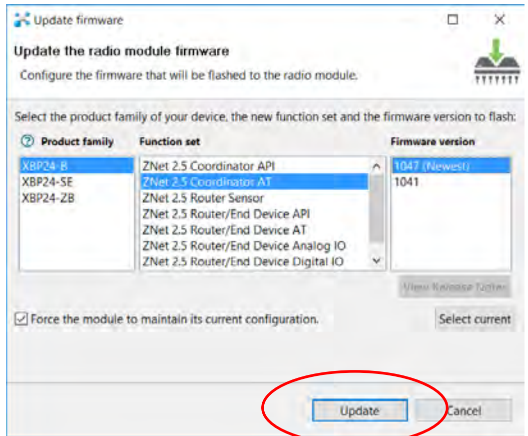


Figure 3-96: Change the function of module “**A**” from Router/End Device to Coordinator.

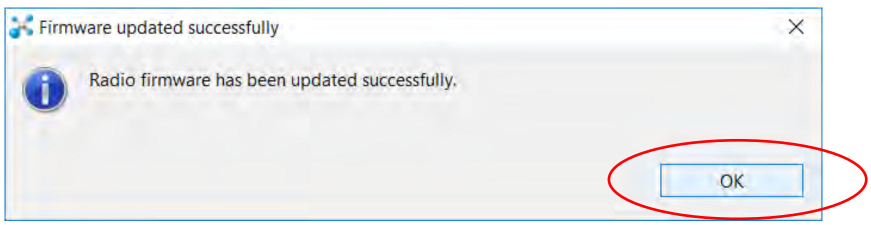


Figure 3-97: Confirmation of the firmware update.

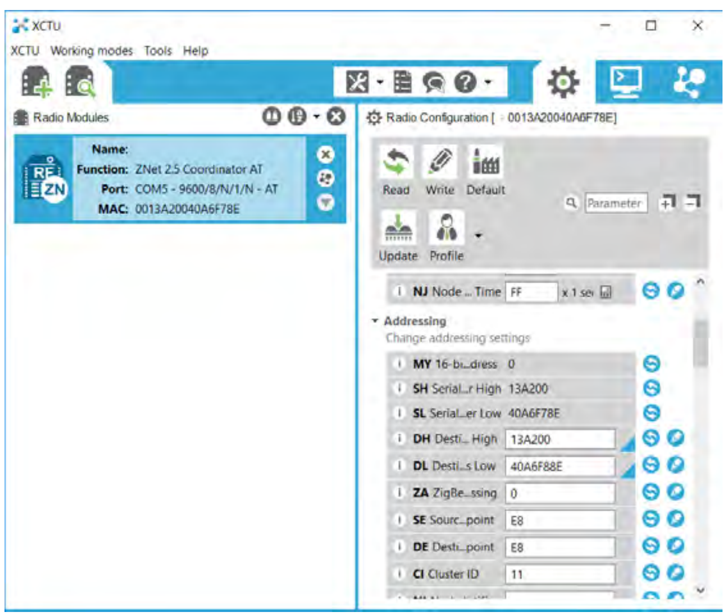
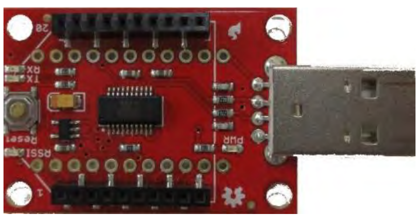


Figure 3-98: Completion of the configuration of module “**A**”.

8.3.1 Confirmation of successful XBee configuration

To confirm the successful configuration of both module "**A**' and module "**B**" and check whether they can communicate to each other, a simple experiment must be conducted before install of module "**A**" in the i-CanSat. The experiment requires two ground station interface boards and one or two PCs with terminal software. In this section, the confirmation using single PC with two ground station interface boards is explained. The ground station interface board used in configuration of module "**A**" and "**B**" is employed and the other board is a small version of interface board called XBee explorer, as shown Figure 3-99.



a) XBee Explorer board.

b) Installation of module "**A**" to PC.

Figure 3-99: connection of module "**A**" to PC using the XBee explorer board.

Each of which is connected to a single PC through a USB cable as shown in Figure 3-100. Each module will be accessed using different terminal software application. Module "**B**" is accessed through the built-in terminal software in X-CTU. After connected module "**B**" to the ground station interface board, open the X-CTU application and choose the associated COM port to module "**B**", as shown Figure 3-101. To start the terminal software application in X-CTU click the "**Consoles**" icon as shown in Figure 3-102. The terminal software application is opened as shown in Figure 3-103. To start sending text from module "**B**" click the "**Open**" icon as shown in Figure 3-103. Module "**A**" can be accessed using Tuna Term as described in section 03.8.2.

Figure 3-100: Connection of both module "**A**" and module "**B**" to a single PC to check the operation after configuration.

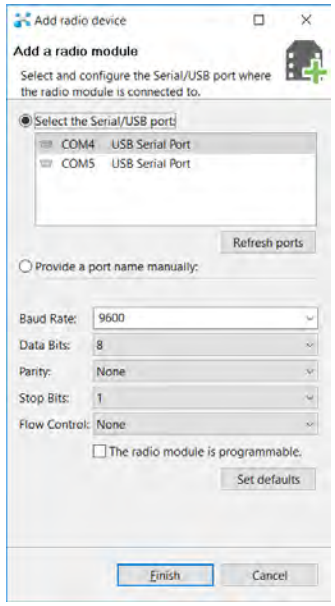


Figure 3-101: Connect module “**B**” to X-CTU configuration software.

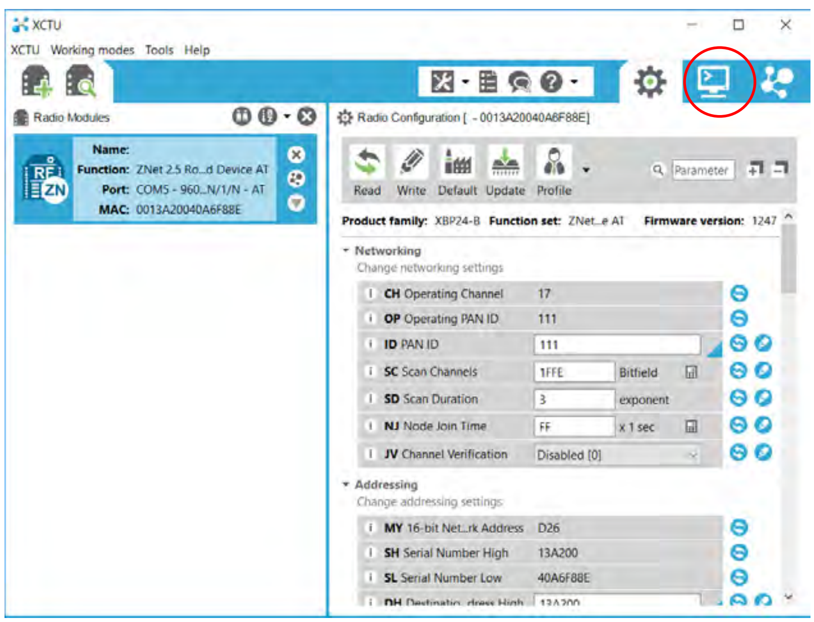


Figure 3-102: Access the terminal application from X-CTU for module “**B**”.

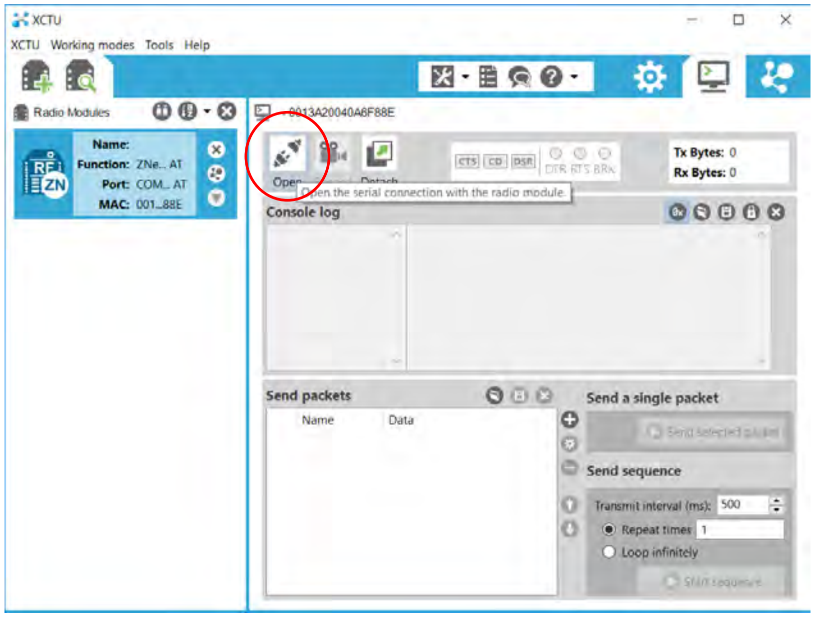


Figure 3-103: Open the terminal application to start sending and receiving by module “**B**”.

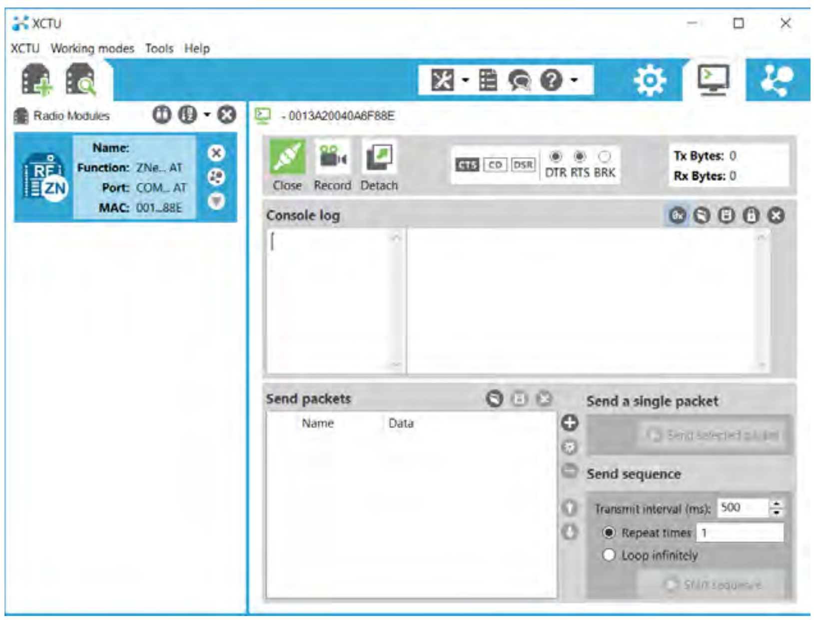


Figure 3-104: Start sending and receiving by module “**B**”.

To start sending data from module “**B**” to module “**A**”, any text can be typed in the console log window of the X-CTU terminal application shown in Figure 3-104. To confirm that module “**B**” can send data to module “**A**”, type any text through X-CTU application console log, as shown in Figure 3-105, and if it appears in the RX window of the Tuna Term application which is connected to module “**A**” then it is confirmed that module “**B**” sends data to module “**A**”, as shown in Figure 3-106. The same experiment can be conducted to confirm that data from module “**A**” can be sent to module “**B**” by typing the data in the TX window of Tuna Term and observe the console log of the X-CTU terminal application, as shown in Figure 3-105 and Figure 3-106. If no data appears in neither the console log window of X-CTU nor the RX window of the Tuna Term then the configuration of the XBee modules is not successful and must be conducted again.

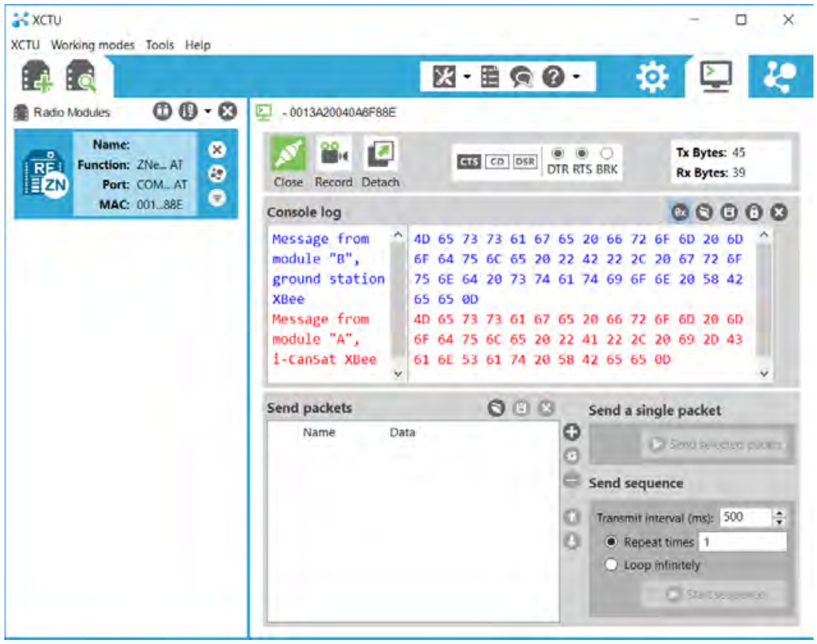


Figure 3-105: Sending data from module “**B**” and receiving data from module “**A**” through the X-CTU console log.

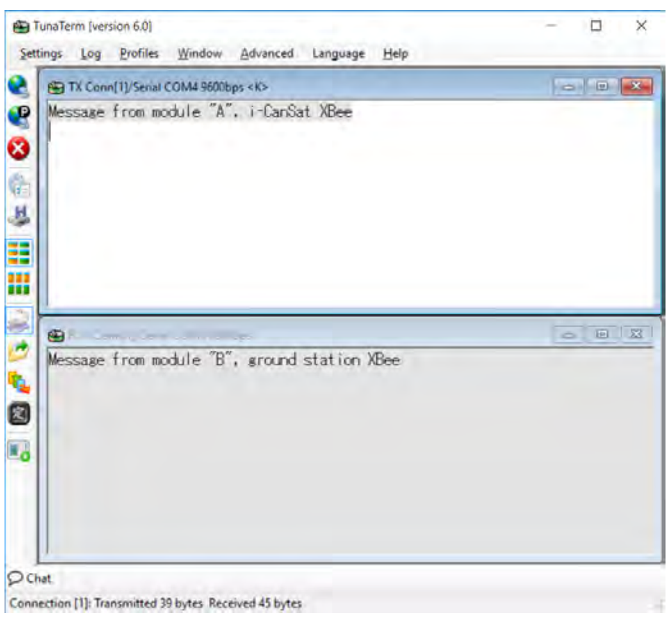


Figure 3-106: Sending data from module “**A**” and receiving data from module “**B**” through the Tuna Term terminal application.

3.9 Writing A Program

3.9.1 **The PICKit3 and MPLAB X IDE**

A program can be written for the i-CanSat to perform a specific mission. The program can be written and debugged using the PICKit3 interface kit shown in Figure 3-107. The PICKit 3 is connected to i-CanSat via the OBC board through the **J44** and to the ground station PC through a USB port, as shown in Figure 3-107. The cable used for connection with the i-CanSat is described in section 3.6.7 and shown in Figure 3-56. To start writing a program, two applications should be installed in the ground station PC. They are as follows:

1. MPLAB X IDE version 3.35.
2. MPLAB XC8 Complier version 1.36.

These applications can be downloaded from the Microchip Technology Inc. website. The MPLAB CX8 complier sometime installs automatically after the installation of MPLAB X IDE. At the time of writing this manual, the two applications must be installed.

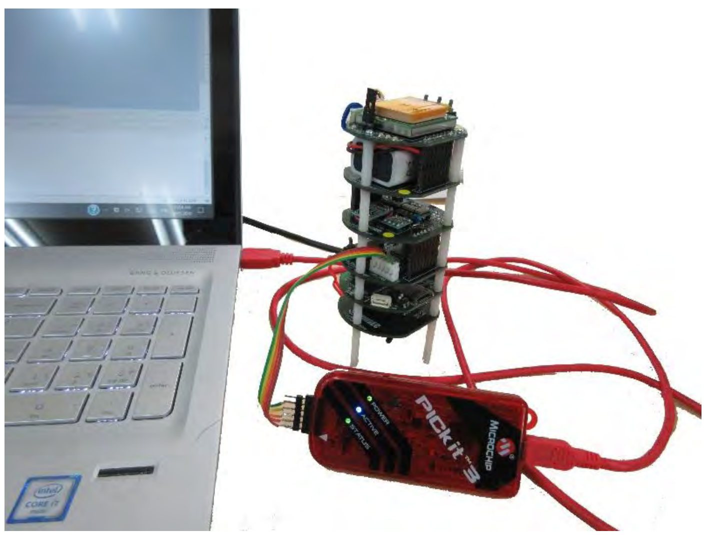


Figure 3-107: Connection of i-CanSat with PC and PICKit3.

3.9.2 **Write and Upload Simple Program**

A simple program that describes the whole compilation and uploading process is presented in this subsection. At the beginning open the **MPLAB IDE**, as shown in Figure 3-108, and from the file menu select "**New Project**", as shown in Figure 3-109. Choose the "**Standalone Project**" then click "**Next**". Specify the family and device. The PIC type in i-CanSat is PIC16LF877A. Specify "**All Families**" for the family and PIC16LF877A for the device, as shown in Figure 3-110, then click "**Next**". Select the "**PICkit3**" as tool for programming as shown in Figure 3-111, then click "**Next**". Select the compiler for this type of PIC which is "**XC8 v.136**". It should be noted that there are two installed versions of XC8 They are version 1.00 and 1.36. Only version 1.36 is active for PIC16LF877A and therefore it is selected. Click "**Next**" after complier selection then the project name and its associated folder must be entered, as shown in Figure 3-113. By clicking "**Finish**" button, the window shown in Figure 3-114 is resulted.

Figure 3-108: Open the MPLAB IDE (Integrated Development Environment).

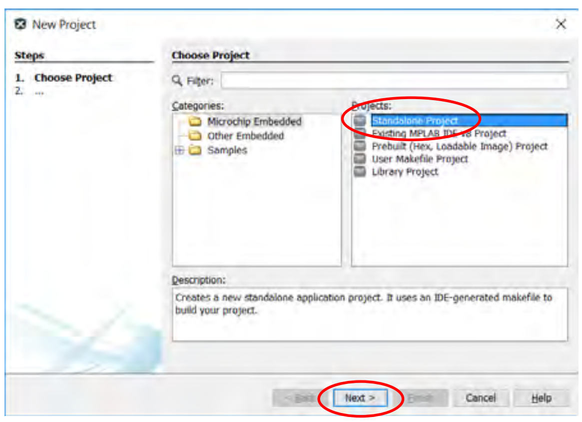


Figure 3-109: Start "New Project" as standalone project.

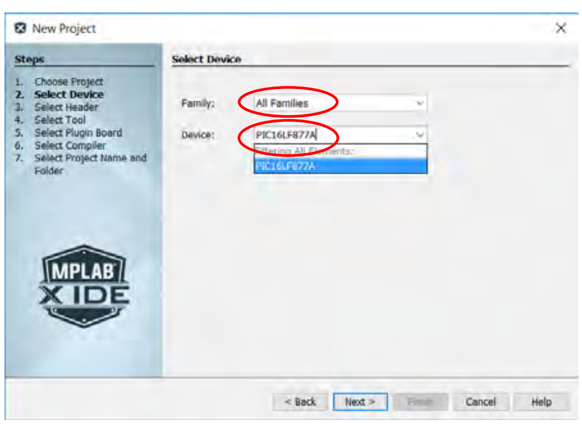


Figure 3-110: Select the family and device.

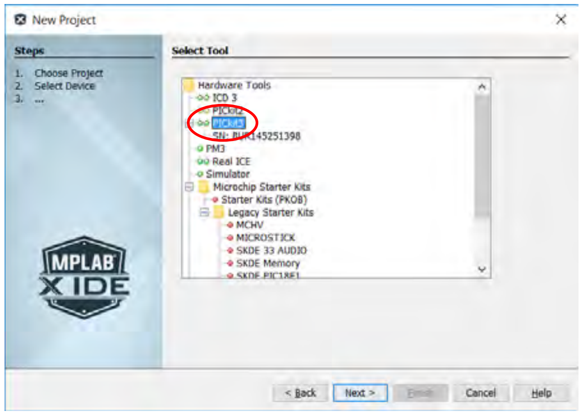


Figure 3-111: Select the tool used for programming, PICkit3.

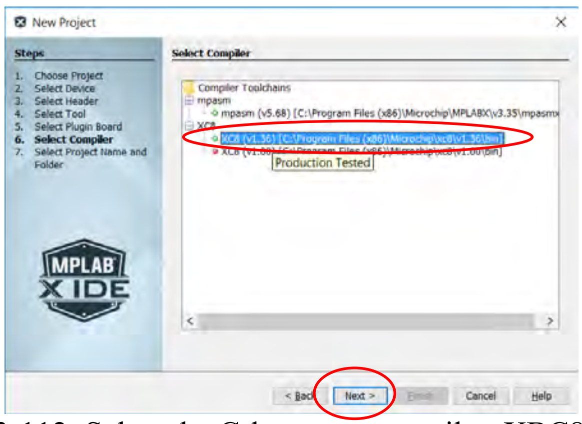


Figure 3-112: Select the C-language compiler, XBC8 V1.36.

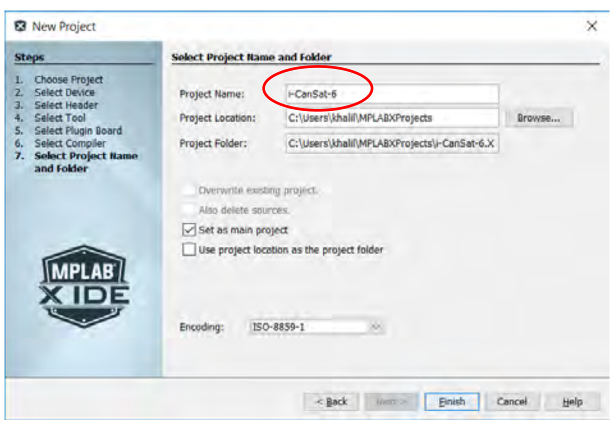


Figure 3-113: Type the project name.

To write the sample program, right click the “**Source**” folder in “**Projects**” window then select “**New**” then “**Others**”, as shown in Figure 3-114. Choose the source file type as “**C Main File**” from the “**New File**” window then click “**Next**”, as shown in Figure 3-115. Enter the source file name in the “**New C Main File**” window then click “**Finish**”, as shown in Figure 3-116.

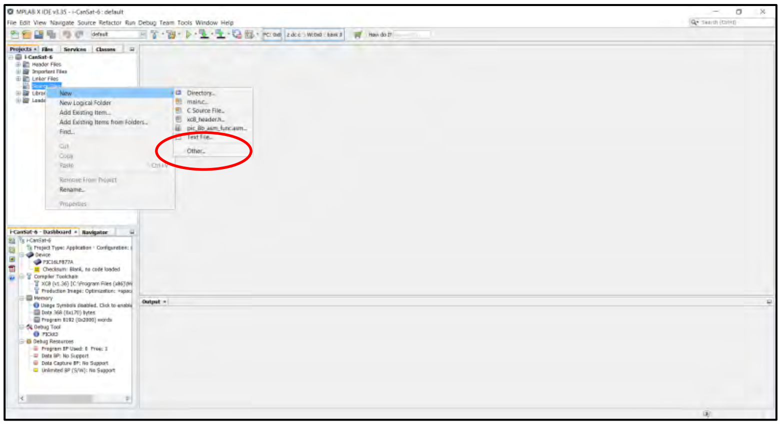


Figure 3-114: Start creating the source file.

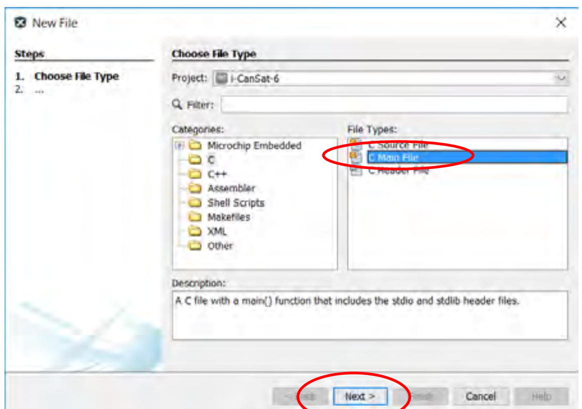


Figure 3-115: Select the type of the source file as "**C Main File**".

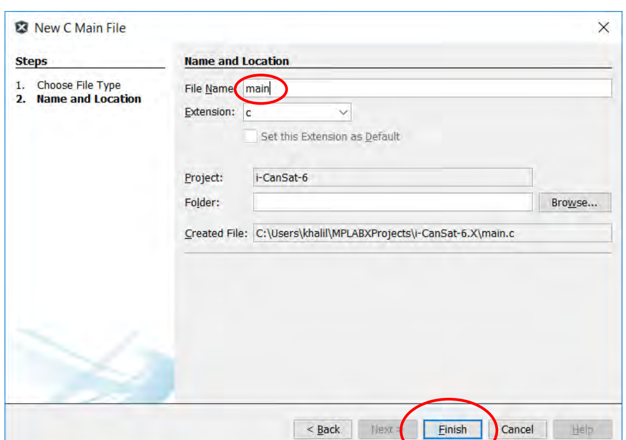


Figure 3-116: Type the name of the source file.

Double click "**main.c**" source file from the "**Source**" folder in the "**Projects**" window and write the following program, as shown in Figure 3-117.

```
#include <stdlib.h>
#include <htc.h>

/**************************************************************/
/*User Global Variable Declaration                              */
/**************************************************************/

__CONFIG(FOSC_HS & WDTE_OFF & PWRTE_OFF & BOREN_OFF & LVP_OFF & CPD_OFF & WRT_OFF &
CP_OFF);
#define _XTAL_FREQ 10000000

/* Define 'value' Register name of port B */
#define LED RB3

#define INPUT 1
#define OUTPUT 0
#define HIGH 1
#define LOW 0

/**************************************************************/
/* Main Program                                                */
/**************************************************************/

void main()
{
TRISB3 = OUTPUT; RB3 = LOW; // I/O as LED (HIGH: Lighted, LOW: Unlit)
  while(1)
  {
    LED = HIGH;
    __delay_ms(5000);

    LED = LOW;
    __delay_ms(5000);

  }
}
```

The program allows the PIC to access single digital "**OUTPUT**" port which is the third port of B-ports. This port is connected to the "**USR**" LED and the program will light up the "**USR**" LED for five seconds and unlit it for another five seconds continuously. To compile and upload the program to the PIC, two steps must be done.

Firstly, set the configuration of the PIC by right click on the project name in the project window and select "**Set Configuration**" then "**Customize…**", as shown in Figure 3-118. From the "**Categories**" select "**PICkit3**" then from the "**Option Categories**" choose the "**Power**" from the drop-down window, as shown in Figure 3-119. Check the "**Power target circuit from PICkit3**" and specify the voltage level, as shown in Figure 3-120. This option will power up the target circuit using "**PICkit3**". The voltage level value depends on the USB specification of the PC. Some USBs can work with 3.5 volts and others work with higher values. Ideal power source must provide the target circuit with 3.3 volts. Therefore, if 3.5 volt is not working try higher level step by step. Excessive high voltage might destroy the target circuit and USB port. This power setting is very important during firmware development because it will save alot of battery's power to power up the target circuit.

Secondly, compile and upload the program by click the "**Make and Program Device Main Project**" icon , as shown in Figure 3-117. Select the programing device as PICKit3, as shown in Figure 3-121. Then a warning message about voltage interface compatibility will be shown as in Figure 3-122, press "**OK**". The progress and successful compilation will be shown in the output window of MPLAB IDE, as shown in Figure 3-123. If there are errors then they will be shown on the same output window of MPLAB IDE. Now, the program is uploaded and run on the OBC of the i-CanSat and observe "**USR**" LED. The program can be hold from the MPLAB IDE by clicking the "**Hold in Reset**" icon  . It can also run again by clicking the "**Release from Reset**" icon . It is highly recommended to click the "**Hold in Reset**" before pulling out the PICkit3 cable.

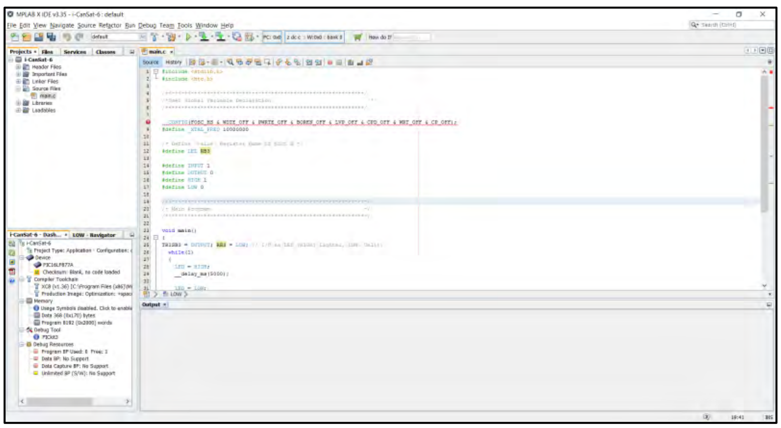


Figure 3-117: Write down the program.

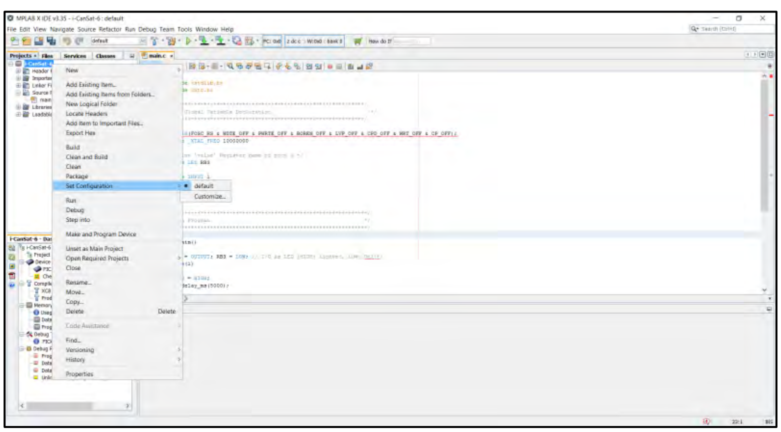


Figure 3-118: Set the project configuration.

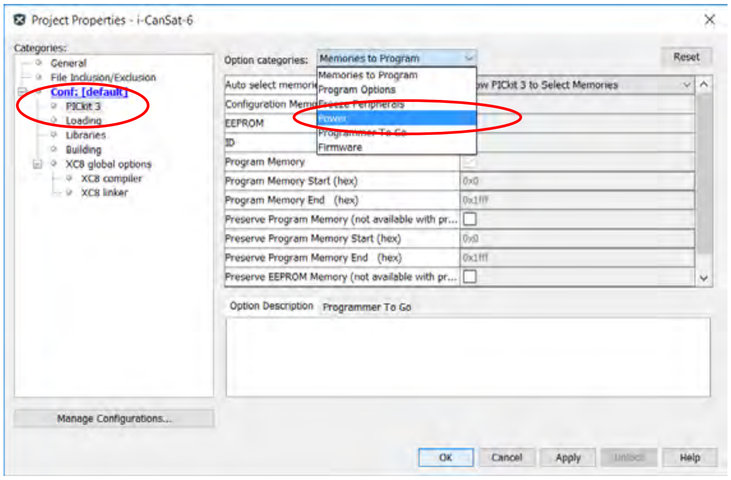


Figure 3-119: Set the PICkit 3 power parameters.

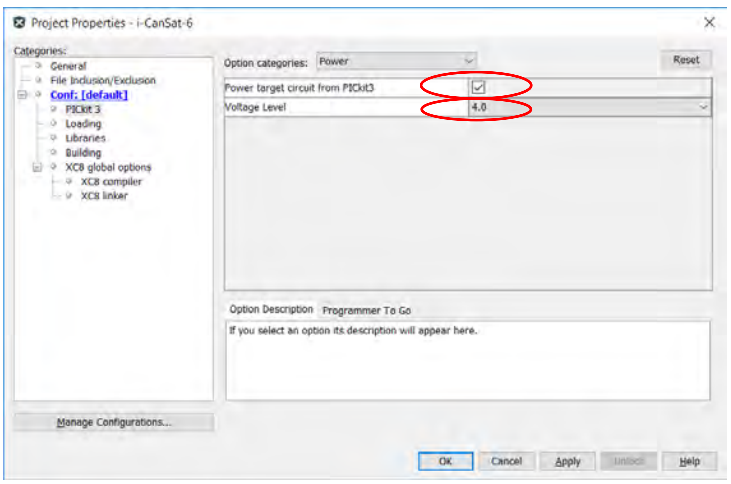


Figure 3-120: Check to power up the i-CanSat from PICkit3 with specific voltage level.

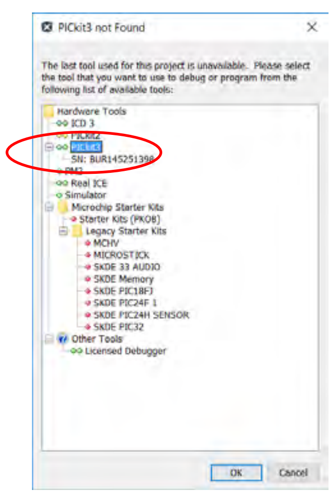


Figure 3-121: Select the programming device as PICkit3.

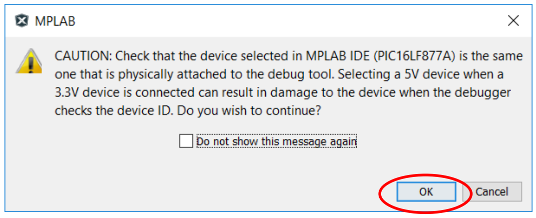


Figure 3-122: Warning message about voltage interface compatibility.

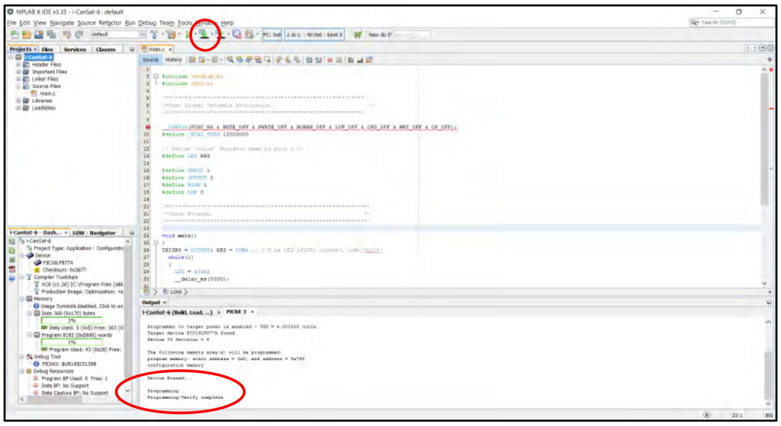


Figure 3-123: The program is successfully complied and uploaded to the device.

Another simple program can be written. This program accesses digital "**OUTPUT**" and "**INPUT**" ports. It keeps the "**USR**" LED lighting up and reads the status of the "**SEP**" (**J14**) jumper. If it is pulled up then the "**USR**" LED will unlit. Write, make the program and observe the LEDs of the "**USR**" and "**SEP**" while inserting and pulling up the "**SEP**" jumper.

```
/******************************************************************/
/* Files to Include                                               */
/******************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <htc.h>

/******************************************************************/
/*User Global Variable Declaration                                   */
/******************************************************************/

__CONFIG(FOSC_HS & WDTE_OFF & PWRTE_OFF & BOREN_OFF & LVP_OFF & CPD_OFF & WRT_OFF &
CP_OFF);
#define _XTAL_FREQ 10000000

/* Define 'value' Register name of port B */
#define LED RB3
#define SEP RB1

#define INPUT 1
#define OUTPUT 0
```

```
#define HIGH 1
#define LOW 0

/**********************************************************/
/* Main Program                                           */
/**********************************************************/

void main()
{
TRISB3 = OUTPUT; RB3 = LOW; // I/O as LED (HIGH:Lighted, LOW:Unlit)
TRISB1 = INPUT; // I/O as SEP (LOW:NotSEP, HIGH:Separated)
  while(1)
  {
    if (SEP == HIGH) LED = LOW;
    if (SEP == LOW) LED = HIGH;
    __delay_ms(10);

  }
}
```

3.9.3 Writing a Typical Program for i-CanSat

A Typical sample program for a simple CanSat mission is depicted by the flow chart shown in Figure 3-124. A typical sample program is described in appendix **B** at the end of this manual. The interruption is not employed for the transmission operation of serial communication. No interruption programming is available when the speed of the main routing loop is much higher than that of the reception of data from the GPS. Accordingly, the interruption should be adopted for serial receiving operation at the very least, otherwise the basis will collapse if the time-consuming processes are added in the loop of the main routine loop. For a more thorough understanding of the sample program, refer to the circuit diagram of i-CanSat, and the specifications of each electronic modules including PIC and electronic modules in **USR** board. In the description of main routine, for cases where A/D conversion is used, refer to the code for data acquisition from the gyro or accelerometer mounted on the **USR** board described in appendix **C**. It is highly recommended to compile and upload the sample program without the USR board related instructions. After conducting the tests described in the following section successfully, the USR board related instructions can be included and tested.

In the following subsections i-CanSat operations are described. These operations are based on the successful compilation and uploading of the typical sample program described in Appendix **B** without any **USR** board instructions.

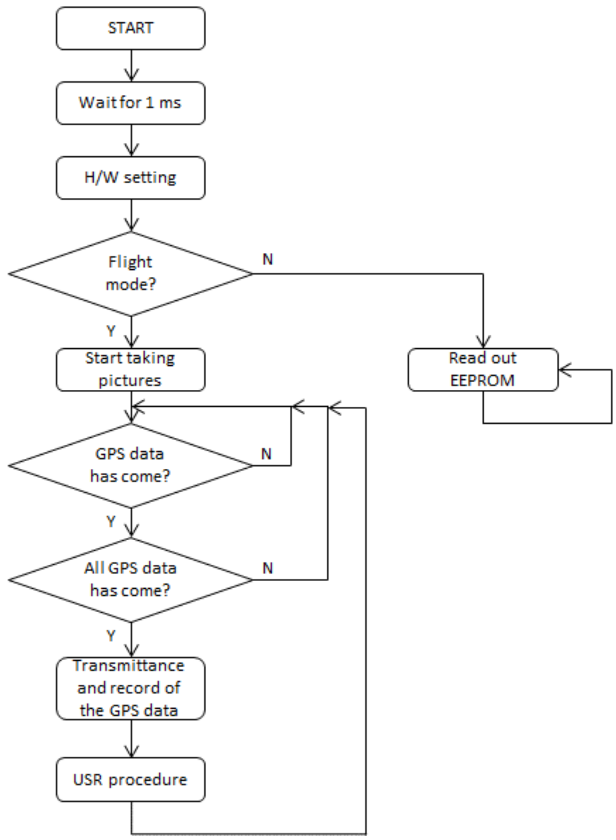


Figure 3-124: Flow chart of the sample program.

9.3.1 Debugging in FLT mode: wired communication to PC

Set the switches on the i-CanSat as shown in Figure 3-125. Connect the i-CanSat to the ground station PC using the USB-serial cable described in section 3.6.7 and shown in Figure 3-55. The i-CanSat can be powered up from the battery or the PICkit3. When the power is turned **ON**, in case of powering up from the battery, the i-CanSat begins the procedures for flight mode (**FLT**) according to the condition, **`if (READ = LOW)`**, in the main function of the sample program. Open the Tuna Term with the proper serial communication port to read from the i-CanSat the data transmitted through the USB-serial cable. Since the **GPS_OUT** switch is set to **OBC,** the GPS data is sent to the OBC. The output data from the OBC is sent to PC via **J13** as the **TXD_OUT** switch is set to **PC**, and the stream of the GPS data is displayed on the Tuna Term window, as shown in Figure 3-126. The GPS data will not be transmitted to the PC unless the **SEP** jumper is pulled out. According to the instructions in the program the data from the OBC will be terminated if the **SEP** jumper is inserted. While sending the data to the terminal application in the PC, the data is written continuously to the EEPROM on the OBC board regardless the **SEP** jumper status.

Further development of the program can make the output data contains not only the GPS data but also values or messages from different sensors mounted on the **USR** board. Debugging may require some trial and error and it is recommended to debug via wired communication to PC as it is simple and fast compared to debugging via the XBee.

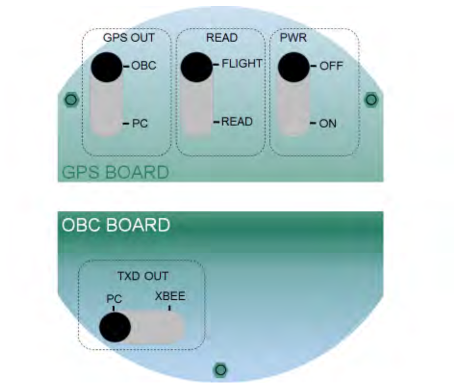


Figure 3-125: Switches setting in **FLT** mode. Data is transmitted via USB-Serial cable to PC.

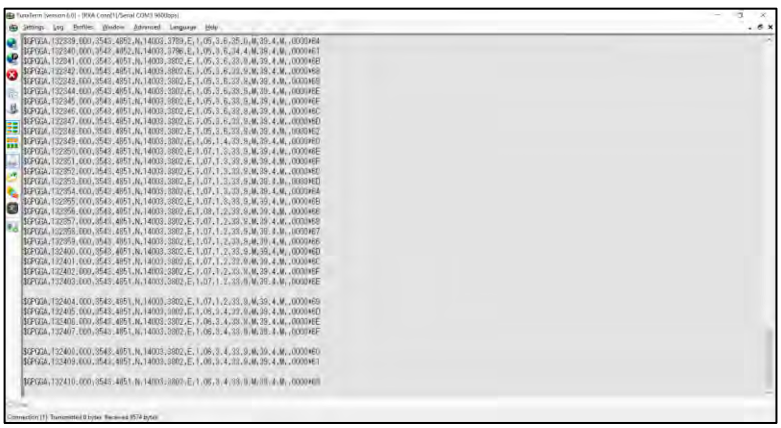


Figure 3-126: i-CanSat output in the terminal application window.

9.3.2 Debugging in FLT mode: wireless communication to PC

To debug using the XBee communication, the i-CanSat switch must set as shown in Figure 3-127. The i-CanSat can be powered up from the battery or the PICkit3. The XBee modules must be installed on their respective position, module "**A**" in the i-CanSat and module "**B**" in the ground station interface board as described in section 3.8. Open the Tuna Term with the proper serial communication port to read from the i-CanSat the data transmitted through the XBee communication modules. By turning the power **ON,** in case of powering up from the battery, i-CanSat begins the instructions for flight mode according to the condition, **`if (READ = LOW)`**, in the main function of the sample program. The output data from the OBC is sent to XBee, as the **TXD_OUT** switch is set to **XBee**, and received by the Ground Station (GS) and displayed on the terminal application. A couple of seconds is required to establish the communications between the two XBee communication modules before it is displayed the data in the terminal application. According to the instructions in the program the data from the OBC will be terminated if the **SEP** jumper is inserted. While sending the data to the terminal application in the **PC**, the data is written continuously to the EEPROM on the OBC board regardless the **SEP** jumper status. This simulates a typical situation when the i-CanSat is deployed from the carrier, a rocket or balloon, by parachute. One of the parachute lines will pull out the **SEP** jumper and i-CanSat will start to transmit the data to the GS PC through the XBee module. When the communication between the i-CanSat's XBee and the GS's XBee is established, the three **LEDs** on the ground station interface board will all turn on.

Further development of the program can make the output data contains not only the GPS data but also values or messages from different sensors mounted on the **USR** board. Debugging may require some trial and error and it is recommended to debug via wired communication to PC as it is simple and fast compared to debugging via the XBee.

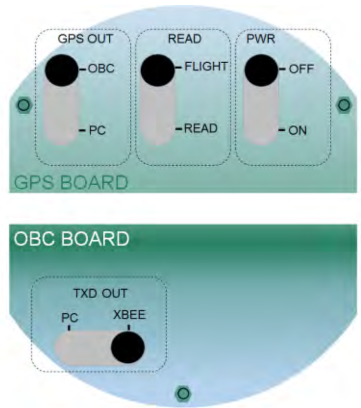


Figure 3-127: Switches setting in **FLT** mode. Data is transmitted via XBee to PC (wireless).

9.3.3 GPS and satellite acquisition

The procedures described in this section required that the i-CanSat is taken to a testing field with portable PC if it is decided to debug using USB-serial cable or take i-CanSat only if it is decided to debug using the XBee communication modules. The maximum separation distance between the i-CanSat and PC in case of wireless communication is about 500 meters line of sight. The switches setting can be any of that described in section 9.3.1 or section 9.3.2 The i-CanSat must see the open sky to acquire the GPS satellite signal. It is hard to acquire the GPS satellite signal indoors.

Since the L1 band frequency from GPS satellites is 1,575 MHz, it is difficult to acquire GPS signals when using a crystal oscillator of submultiple frequencies such as 25 MHz or 12.5 MHz. To avoid this, i-CanSat is driven by a frequency of 10 MHz, and typically acquires GPS satellites within seconds for the first time, or within a couple of minutes at most. This is called a "**cold start**". The GPS data obtained is shown in Figure 3-128.

After initial GPS satellite acquisition, the acquisition time is shortened as information of previously acquired GPS satellites is stored into the GPS. This is called a "**hot start**". From the data shown in Figure 3-128, one minute was required from the point of cold start to acquire five GPS satellites. For example, the GPS data which is displayed as follows:

```
$GPGGA,165800.000,3539.7495,N,13922.0655,E,1,05,6.5,184.4,M,39.2,M,,0000*5F
```

This means the data type is GGA, the second data is 16:58:00 UTC, the latitude is 35 degrees and 39.7495 minutes (=35.6624917 degrees) North, the longitude is 139 degrees and 22.0655 minutes

(=139.3677583 degrees) East, solo positioning, the number of acquired GPS satellites is 5, HDOP=6.5 meters, the height of the antenna is 184.4 meters above the mean sea level, and the height of the mean sea level from the WGS-84 ellipsoid is 39.2 meters. The actual altitude at this location is 114 meters, but is calculated to be 184.4 –39.2 = 145.2 meters from the GPS data. Keep in mind that the error in height calculated from the GPS is very large regardless of the HDOP value. If accurate information about the altitude is required, different sensor must be employed to measure the altitude directly or indirectly. For example, barometric sensor can measure the local atmospheric pressure and from its measurements the altitude can be calculated. This is an example of indirect measurement and it is accurate compared with GPS measurements.

If the i-CanSat can acquire over 4 GPS satellites within a couple of minutes, even in cold-start, no problem will occur. Note that the length of GPS data output is flexible, depends on the status of GPS acquisition. A particular algorithm is required to extract values from a string with flexible length, but it is not implemented in the sample program.

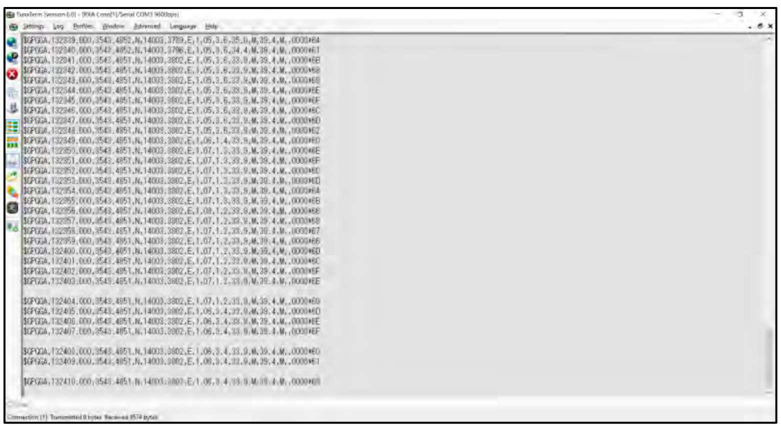


Figure 3-128: GPS output data with satellite acquisition.

3.9.3.3.1 Uploading the GPS track to Google maps.

In this section, the procedures to upload the GPS track data to the Google maps are described. The procedures can be summarized as follows:

1. The i-CanSat must see the open sky to acquire the satellite signals. Then conduct the procedures described in section 9.3.2 .
2. Open log file to start save the acquiring GPS signals.
3. Move with the i-CanSat any distance, for example 200 or 300 meters but not more than 500 meters.
4. Close the log file.
5. The data of the log file can be opened using any text editor, for example notepad as shown in Figure 3-129. It can be also opened in Excel as explained in the next step.
6. Open the log file in Excel and follow the sequence of actions shown in Figure 3-130 and Figure 3-131. This informs the Excel about the used delimiters between the GPS data stream which is a "Comma".

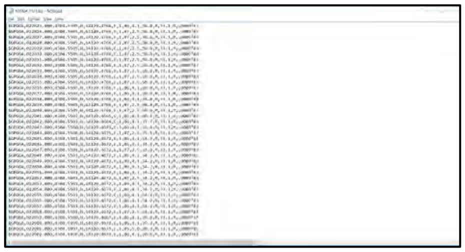


Figure 3-129: Check the log file of the GPS track data using the text editor “notepad”.

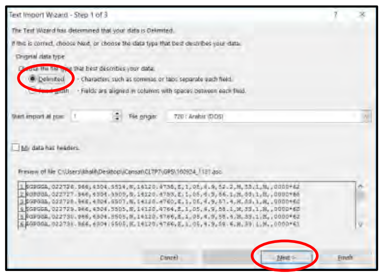


Figure 3-130: Open the log file with Excel.

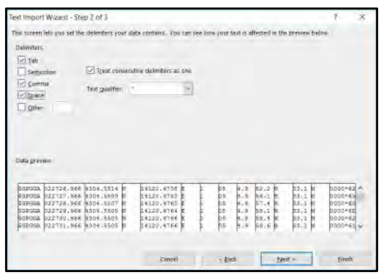


Figure 3-131: Select the delimiters of the log file.

7. The raw GPS data in Excel is shown in Figure 3-132. The raw data of latitude and longitude are listed in column “C” and “E”, respectively. Their interpretation is described in section 9.3.3 .
8. The latitude and longitude data in degrees can be calculated from the raw data of the of latitude and longitude using the following formula:

$$\mathrm{Data}_{\mathrm{deg.}} = \left[\frac{\mathrm{Data}_{\mathrm{raw}} - ROUNDDOWN(\mathrm{Data}_{\mathrm{raw}}, -2)}{60}\right] + \left[\frac{ROUNDDOWN(\mathrm{Data}_{\mathrm{raw}}, -2)}{100}\right] \qquad \text{Equation 3-7}$$

ROUNDDOWN is an Excel function that returns a number rounded down to a specified number of digits. (Always rounds towards 0.)

9. The calculated latitude and longitude data in degrees are listed in column “Q” and “P”, respectively in Figure 3-133.
10. Copy and paste (as values) the calculated latitude and longitude in step 9 into a new Excel file and save it, as shown in Figure 3-134.

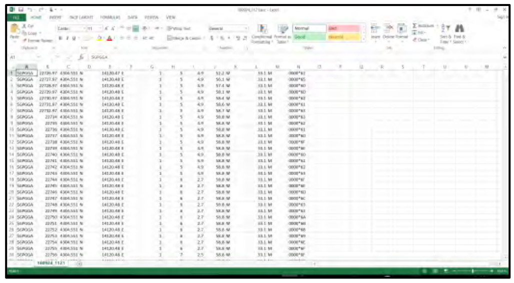


Figure 3-132: The log file opened in Excel.

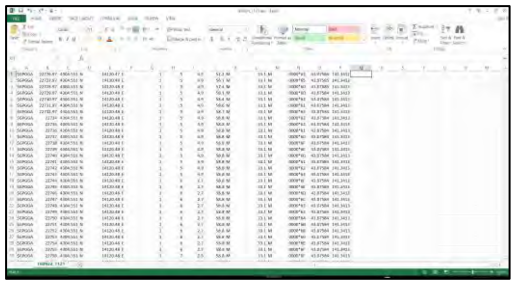


Figure 3-133: Conversion of the raw data of latitude and longitude of the GPS track in column “C” and “E”, respectively to the latitude and longitude in degrees in column “O” and “P”, respectively.

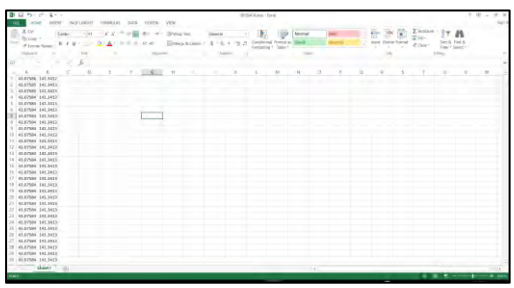


Figure 3-134: Save the calculated latitude and longitude of the GPS track data in degrees in a new Excel file by copy and paste them as “values” in the new Excel file.

11. It is required to have a Google account to conduct the following steps. After logging in to the Google account, open the Google maps and click the Google maps menu as shown in Figure 3-135.
12. In the Google maps menu click “**Your places**”, as shown in Figure 3-136.
13. In the “**Your places**” menu click “**MAPS**”, as shown in Figure 3-137.
14. In the “**MAPS**” menu click “**CREAT MAP**”, as shown in Figure 3-138.
15. In the “**CREAT MAP**” menu click “**Import**”, as show in Figure 3-139.

Figure 3-135: Access the google map from any internet browser.

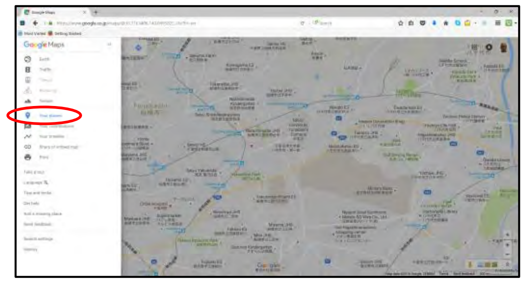


Figure 3-136: Click “Your places” from the google map Manu.

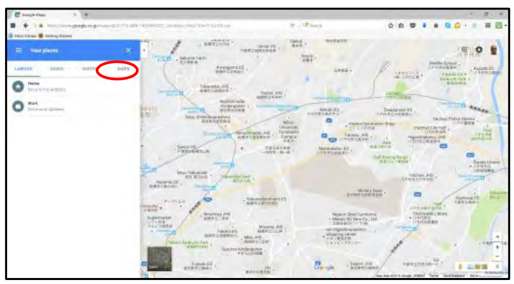


Figure 3-137: Click “Maps” from the options in “Your places”.

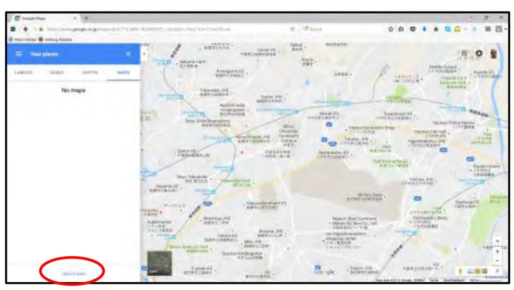


Figure 3-138: Click “CREATE MAP” from the “MAPS” window.

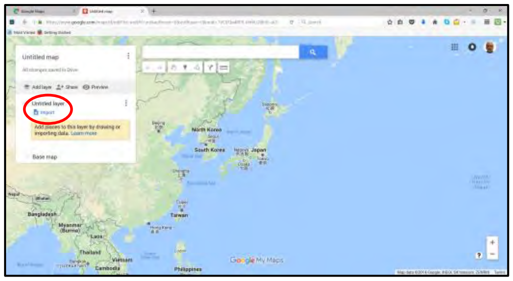


Figure 3-139: import the GPS track data to Google Maps.

16. Choose a file to import which is the Excel file saved in step 10, as shown in Figure 3-140.
17. Choose the columns to position the placemarks, since there are only two columns in the Excel file then select these two columns, as shown in Figure 3-141.
18. Choose the columns that represent the latitude, as shown in Figure 3-142.
19. Choose the column that represent the longitude, as shown in Figure 3-143.
20. Choose the column to title the markers, it can be any column, as shown in Figure 3-144, then click "**Finish**".

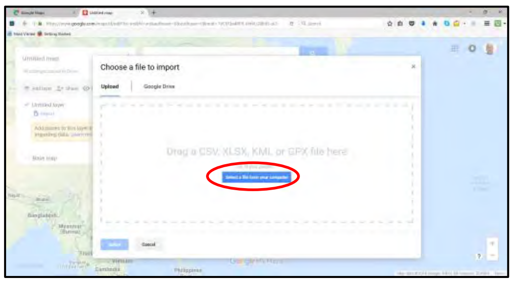


Figure 3-140: Select the saved Excel file of the GPS track data.

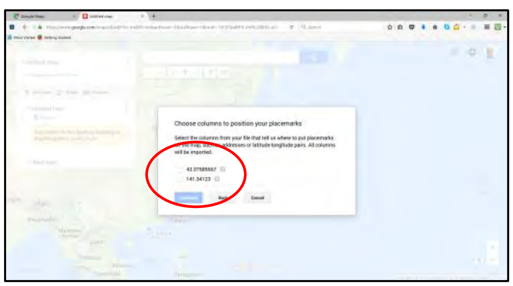


Figure 3-141: Check the latitude and longitude columns.

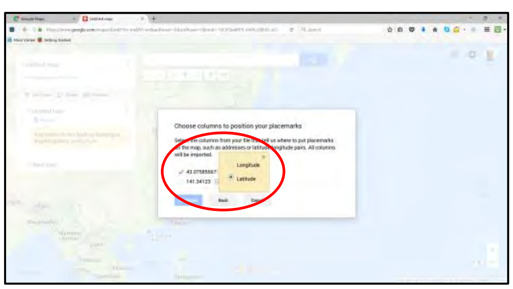


Figure 3-142: Identify the column for latitude.

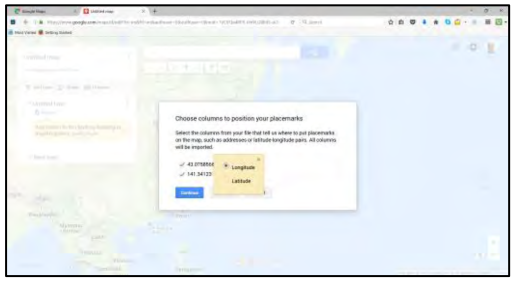


Figure 3-143: Identify the column for longitude.

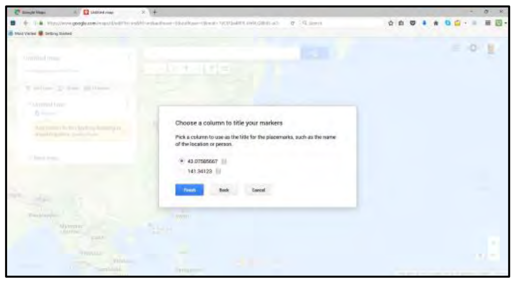


Figure 3-144: Choose any column to title the markers.

21. The track on the Google map is shown in Figure 3-145. To change the base of the map, click "Base map" as shown in Figure 3-145 and select the desired base as shown in Figure 3-146.

Figure 3-145: GPS track mapped in two-dimensional Google maps

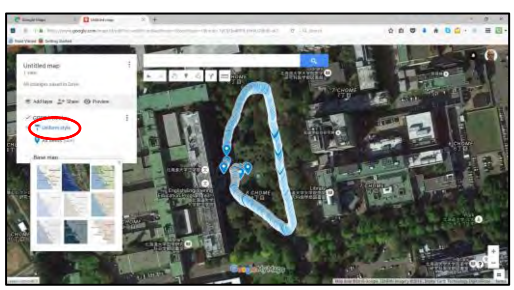


Figure 3-146: GPS track mapped in two-dimensional Google Maps with earth base.

3.9.3.3.2 Uploading the GPS track to Google Earth

For high quality presentation, three-dimensional GPS track need to be shown and projected in Google Earth where the altitude information can be visualized. The following procedures summarize how to construct the three-dimensional GPS track on Google Earth:

1. Rename the GPS log file from *filename.asc* to *filename.nmea*.
2. Convert the NMEA file to KML file using any software application. The GPSbabel is a free application and can be downloaded from the internet.
3. Start the GPSbabel application and specify the input NMEA file and the output KML file as shown in Figure 3-147.
4. Set the options of the NMEA file as shown in Figure 3-148, then click "**O**K".
5. Set the options of the KML file as shown in Figure 3-149, then click "**OK**".
6. Click "**OK**" button, if the conversion is successful a message is printed in the log window as shown in Figure 3-150.
7. Open Google Earth and from the file menu choose "**Open…**" and open the generated KML file in step 6, as shown in Figure 3-151.
8. The three-dimensional GPS track in Google Earth is shown in Figure 3-152.

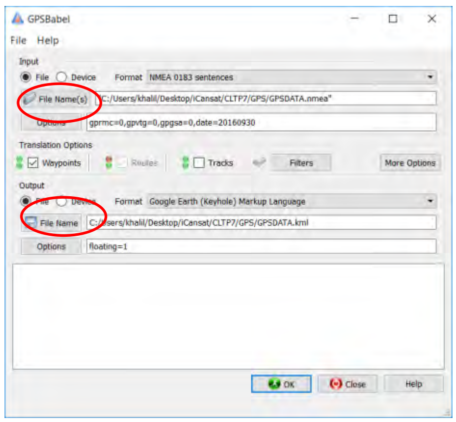


Figure 3-147: specify the input "NMEA" file and the output "KML" file.

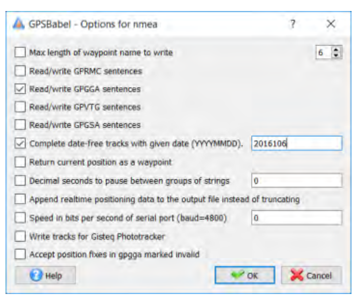


Figure 3-148: Set options for the NMEA file.

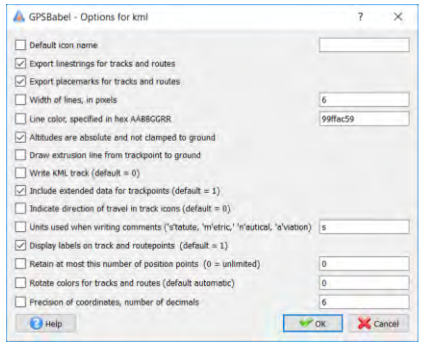


Figure 3-149: Set options for the KML file.

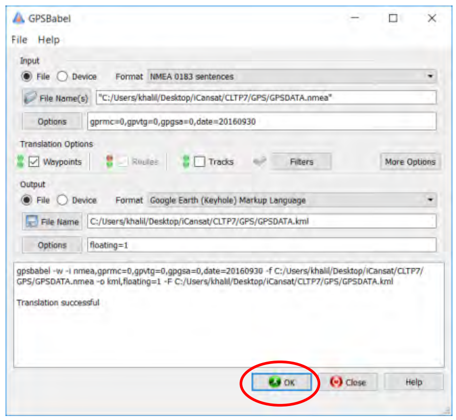


Figure 3-150: Successful completion of converting the NMEA file to KML file.

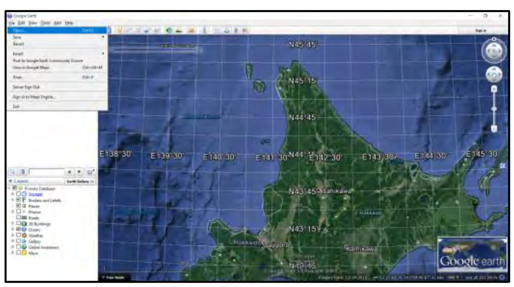


Figure 3-151: Open the KML file from Google Earth.

Figure 3-152: The three-dimensional GPS track in Google Earth.

9.3.4 Debugging in the READ mode

This subsection outlines the operation procedure for **READ** mode via the **PC** or **XBEE**. The Switches setting is shown in Figure 3-153. The **TXD_OUT** switch can be set to **PC** or **XBee** communication modules**.**

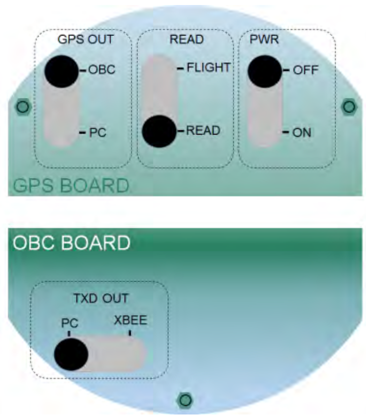


Figure 3-153: Switches setting for debugging in the **READ** mode.

In the READ mode, i-CanSat begins the procedure for **READ** mode according to the condition, **`if (READ == LOW),`** in the main function of the sample program. In **READ** mode, i-CanSat reads the data in EEPROM and outputs it to the terminal application via **PC** or **XBee**. Since the **FLT** mode procedure in the sample program writes the output data to the EEPROM, as described in sections 9.3.1 and 9.3.2 , the read data from EEPROM is the same as the written data. Note that the read data may be displayed on every other line due to the delimiter.

9.3.5 Camera Operation

Set the DIP switches of the CAM board to I^2C mode as follows: SW1=**OFF** and SW2=**ON**, as shown in Figure 3-154. If a Micro-SD card is in the card slot of the CAM board, CANCAM will start taking images one after the other, when a command sent by the OBC via the I^2C is received. When CANCAM receives the I^2C command, the green **LED** on the **CAM** board will light on once, and CANCAM will take an image every few seconds. If a Micro-SD card is not inserted into the card slot, or the wrong I^2C command is received, the red **LED** on the CAM board will blink.

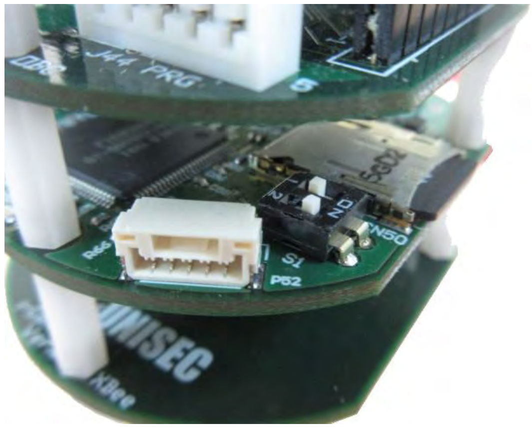


Figure 3-154: DIP switches and Micro SD card slot on the CAM board.

After the i-CanSat flight, the images recorded on the Micro-SD card can be read by the PC using the SD adaptor. Image data is stored as files with extension .DAT. The images can be viewed by a dedicated program called CANCAM Converter. To view the image, drag it into the window of the CANCAM Converter shown in Figure 3-155. Images can be also viewed by CANCAM converter by clicking the “**Convert**” button and select the image from the proper folder. The program converts the .DAT images to .BMP images automatically once it is shown in the view window as shown in Figure 3-156. The image name will have the same name but with BMP extension.

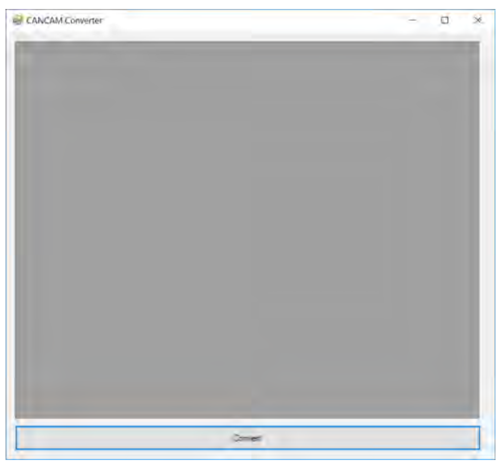


Figure 3-155: CANCAM converter program.

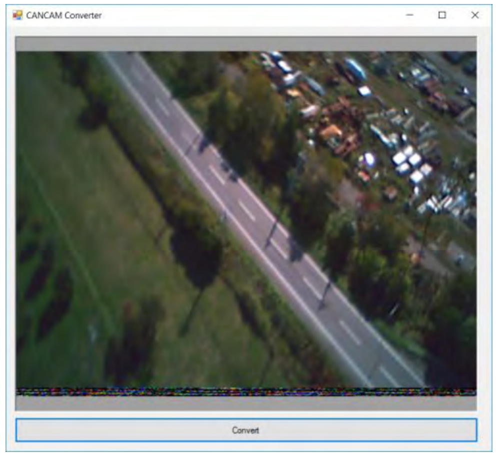


Figure 3-156: Image taken by CANCAM.

3.10 Flight and Data Reading

When it is ready to launch the i-CanSat, refer to the settings and procedures described in section 9.3.2 . If the i-CanSat uses separation detection, do not forget to insert a jumper, connected to one of the parachute lines, to the **SEP** terminal before installing the i-CanSat into the carrier like a rocket or a balloon. Before flight, check all settings and switches on the i-CanSat and ground station.

After the flight, recover your CanSat, and turn power **OFF** and set the READ switch to **READ** to avoid overwriting the precious flight data on the EEPROM, as shown in Figure 3-157. To prevent overwriting, follow the procedure such that the OBC records data on the writing point stored in the final address on the EEPROM. Follow the procedure described in section 9.3.4 when reading out the flight data. Data can be saved in a log file from the terminal application and processed later. The file name of the flight data should be unique, and include the date and time for example.

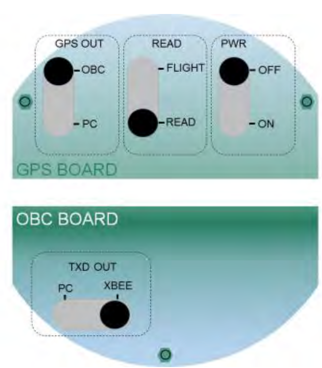


Figure 3-157: Switches setting after flight to protect the data in EEPROM from overwriting.

3.11 Parachute Fabrication

The parachute design is affected by many constraints. These include but not limited to the required decent velocity, the local wind speed during flight, light weight, and robust deployment. i-CanSat usually descents from an altitude of 100-80 meters. To achieve a flight time of about 20 seconds which corresponds to about 5 m/sec decent velocity, three hexagonal parachutes made from thin polythene sheets are fabricated. Most of the shopping bags are made of polythene sheets which make it very affordable. The detailed dimensions of the hexagonal is shown in Figure 3-158. Each parachute is attached to the i-CanSat structure using the three strings each has a length of 500 mm and connected to the parachute as depicted in Figure 3-159. The strings are attached to the i-CanSat through a hole and both strings and parachute must form what is known as cow hitch knot as depicted in Figure 3-160. Figure 3-161 shows a photo of the attachments of three parachutes' strings to the i-CanSat structure.

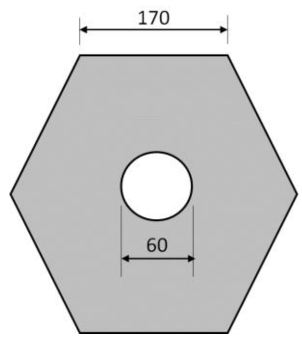


Figure 3-158: Dimensions of the hexagonal parachute.

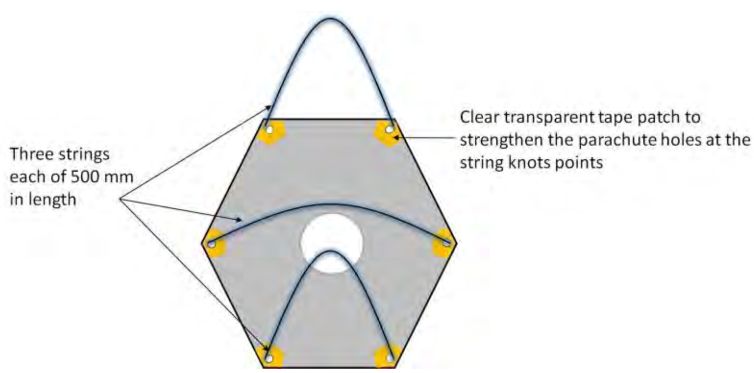


Figure 3-159: The strings attachment to the parachute.

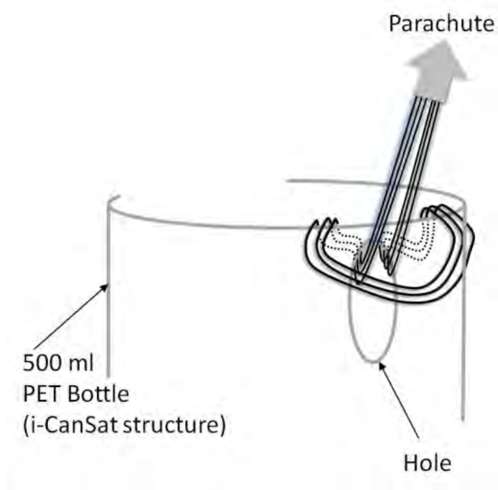


Figure 3-160: The attachment of the parachute and strings to the structure.

Figure 3-161: The attachment of the three parachutes and strings to the i-CanSat.

3.12 The i-CanSat Structure

i-CanSat-6 is designed to be installed into a typical Japanese 500ml PET bottle or can. Note that only cans from Non-Alcoholic beverages can be used, as the diameter of the opening of Alcoholic drink cans is smaller than that required for the i-CanSat. Both the can and plastic bottle have sufficient strength to be used as i-CanSat structures. The procedure to make the structure can be outlined as follows:

- The selected PET bottle must of light weight and has a flat bottom. These conditions usually achieved for water PET bottles.
- Cut 3 holes of about 2 mm in diameter at the bottom of the can or PET bottle. Then fix the i-CanSat with three M2 screws, as shown in Figure 3-163-a.
- For the CANCAM, cut a hole larger than the viewing angle of the CANCAM so pictures can be taken without being obstructed by the structure, as shown in Figure 3-162-b.

- The antenna of XBee communication module should be uncovered, so cut a larger hole at the bottom of can. For PET bottle structures, this hole is unnecessary.
- Cut three holes of about 3 mm in diameter on the upper side of the curved cylindrical surface of the PET bottle, and attach the parachute to these holes as described in section 3.11. In case of the PET bottle, the hole opening weakens the structure therefore kapton tape or thick transparent tape can be used to improve the strength of the structure wall at the hole's location, as shown in Figure 3-161. The parachutes in unfolded condition are shown in Figure 3-162-c and in folded condition are shown in Figure 3-162-d .
- Based on the above steps, the completed i-CanSat using a 500 ml PET bottle is shown in Figure 3-162-c and Figure 3-162-d.

a) Fix the i-CanSat from the bottom.

b) CANCAM hole.

c) Parachute, Unfolded.

d) Parachute, Folded.

Figure 3-162: i-CanSat structure using a 500 ml PET bottle.

3.13 Further Development

The sample program contains only basic features to record data to and read data from the EEPROM, and to receive and transmit positioning data using the GPS and XBee communication module as well as reading from different sensors in the **USR** board, for example accelerometers, gyro, barometric pressure, and temperature. It can also take capture image with CANCAM. Using the i-CanSat, there are many possible experiments that can be done. The following is an example of missions that can be done:

- The current sampling frequency is about 0.5 Hz. This is because the GPS has a sampling frequency of 1 Hz. **USR** board sensors can have a sampling frequency higher than 1 Hz (usually of order 50-100 Hz). The program must be developed to capture the reading from different sensors at high sampling frequency to analyze the launch and flight segments of i-CanSat.

- The OBC of i-CanSat can acquire data from limited number in the **USR** board. Typically three sensors; accelerometer, gyro and barometric pressure sensor or accelerometer, gyro and temperature sensor. This is because the OBC has a limited memory. Firmware development and/or hardware development in the **USR** board are needed to overcome this limitation.
- Develop the i-CanSat to perform come back mission, i.e., fly to a specific point during the descend. This can be done using parafoil which can be controlled by servo actuators controlled by the OBC.

3.14 General Precautions

The i-CanSat is subject to strong shock loads during launching, separation, parachute deployment and touch-down. Therefore, special precautions should be taken to avoid any failure in the i-CanSat mission. These precautions can be summarized as follows:

1. Although the i-CanSat is protected by the outside structure, the shock loads can deform it and change the setting of the **PWR** switche because they have slightly long handle, as shown in Figure 3-163-a. Simple and fast solution to overcome such possibility is to place a shroud around these switches, as shown Figure 3-163-b. The shroud used in Figure 3-163-b is the protected plastic packaging of the EEPROM of the OBC board.

a) **GPS** board switches without shroud.

b) **GPS** board switches with shroud.

Figure 3-163: Shrouded the Switches of **GPS** board.

2. Due to friction, the force generated in the parachute string during parachute deployment might not sufficient to pull-out the **SEP** jumper. Usually the new battery with a single EEPROM can power-up and store data for about 20 minutes which is longer than the typical i-CanSat mission. It is recommended to pull-up the **SEP** jumper manually before launch at the launch site to avoid any possibility of unsuccessful pulling-out by the parachute string.
3. The battery clip snap on connector should be securely and firmly attached to the battery. Thick transparent tape can be used to make sure that it will not detach due to any shock loads.
4. The Micro-SD card should have enough space to store the captured images during the mission.
5. The GPS should be attached to the **GPS** board using double sided tape. To reduce the possibility of detachment from the GPS board due to excessive loads, a think transparent tape can be used to fasten it to the whole i-CanSat.
6. The folding sequence of the parachute must be tested by a simple drop test from a building to make sure that the parachute will open successfully without any jamming. In this test, also the i-CanSat can be powered up and in full operational mode to check if it can survive the shock loads during the parachute deployment and touch down.
7. Sponge can be used at the base of the i-CanSat to act as a damper during touch down and reduce the shock loads. This sponge shouldn't add extra weight and shouldn't increase the diameter of the i-CanSat to fit into the launcher chamber.

4 | CanSat Mission and System

In this Chapter, CanSat's mission concepts, mission examples, examples of instruments to achieve missions, etc. are presented in section 4.1. In the remaining sections, CanSat's subsystem design, development and testing are introduced. Specifically, the Command and Data Handling (C&DH) subsystem is presented in section 4.2, the Electric Power Supply (EPS) subsystem is presented in section 4.3, the communication subsystem is presented in section 4.4, the sensors are presented in section 4.5, the actuators are presented in section 4.6, the structural subsystem is presented in section 4.7, and the ground station subsystem is presented in section 4.8.

4.1 CanSat Mission

4.1.1 Mission and V-diagram

Mission is to determine the significance of the CanSat such as the observations and data acquisition during flight performed by CanSat. For the success of the mission, it is necessary for the equipment installed in the mission subsystem to operate as intended, so it is necessary for the power system, communication system, and other bus systems to perfectly support the mission subsystem. Since CanSat is also a "system" that fulfills one purpose by collaboration of multiple components, the system engineering techniques can be applied in its development.

In the design of CanSat, "mission statement" must be declared briefly expressing CanSat development objective. This is followed by setting up "mission items" to realize the mission and the "success criteria" which is essential for the success evaluation. Based on mission items and success criteria, enumerate "requirements" for realizing mission items and propose verification methods for evaluation of the success/failure. Ideally for each of these requests, "system" is defined by associating "specifications" on a one-to-one basis and grouping the similar specifications, as shown on the left side of Figure 4-1. Alternatively, from the mission item and success criteria, the necessary systems can be listed for this, enumerate the requirements to that system, and make the specifications correspond one-on-one to those requirements, as shown on the right of Figure 4-1. In cases where past CanSats and experiences are the foundation, this is often adopted. These series of processes is called "breakdown".

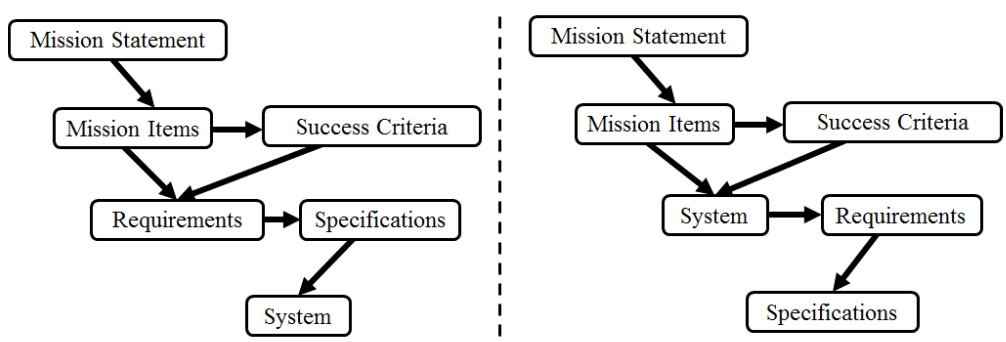


Figure 4-1: Breakdown of CanSat system.

As described above, after clarifying what kind of functions CanSat possesses and what kind of specification it is required, "Integration" which realizes individual specifications and subsystems of CanSat system is performed. “Testing” is performed to confirm that the integration corresponding to each stage of breakdown has been fully fulfilled. In summary, the whole CanSat project can be expressed as "V model", as shown in Figure 4-2.

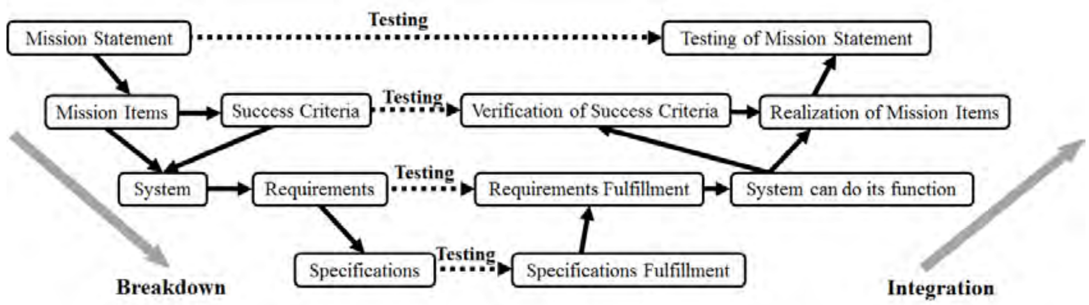
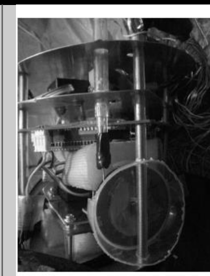


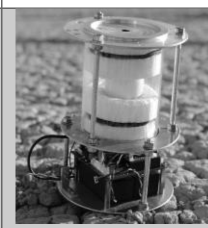
Figure 4-2: V-model of CanSat.

4.1.2 **Examples of CanSat Missions**

As an example of CanSat's mission, some of the participating teams in the A Rocket Launch for International Student Satellites (ARLISS) held in 2012 are presented, as listed in Table 4-1. The mission statement at the time of development, mission items and success criteria are clearly stated. These should clarify what to produce and how it should be produced during the CanSat development.

Table 4-1: Participating teams’ CanSat mission in ARLISS 2012

CanSat Participating Team	Mission Statement	Mission Items	Success Criteria (minimum)	Success Criteria (Full)	Developed CanSat
NASU, University of Tokyo	Develop CanSat with parafoil that can fly back to a prescribed target point on the ground when it is separated in the strong winds layer of the atmosphere.	- Approach the target point with parafoil - After quickly leaving the strong winds with free fall and entering controllable layer of the atmosphere, the parafoils is deployed	- Deploy the Parafoil at a predetermined altitude. - Verify the deployment by telemetry from at the ground station. - Store the flight records and verify that the control is possible	- Do autonomous control by using the parafoil - At altitude of 1 Km, the projected CanSat path, while flying, on the ground will be within 300 m or less from the ground target point . - The CanSat will land within 1 Km or less from the target point.	

SABRO, Tokyo Institute of Technology	Planetary exploration with autonomous control and relay communication with ultra-small spacecraft	- Perform weather observations until landing - After landing, it separates into a master unit and a slave unit, the master unit functions as a ground station, and the slave unit performs autonomous control	- The master unit that performs data communication with the slave unit acquires weather data - The slave unit is separated from the master unit and takes moving pictures	- The master unit transmits the status of the slave unit, including the weather data - The slave performs autonomous movement and feedback control	
TMG, Tokyo Metropolitan University	Bring a fragile object safely to the ground	Put egg in CanSat so that it will not be broken and let it fall to the ground	- Bring eggs to the ground without breaking eggs - Deploy parachute after separation from rocket	- Take eggs to the ground without cracking - Shoot movies of the eggs in flight.	

The first thing that should be considered is what kind of mission can be implemented with CanSat. It is essential practice to conduct a "brainstorming" which gives opinions freely among teams' members at the beginning of development and lists "things you want to do". At the stage of brainstorming, any opinion is accepted without criticism, and it is expected to stimulate the discussion and exploit different ideas. However, at the next stage, “things you want to do” is different from “things you can do”, “things you want to do” is restricted by the technical ability and human resources of each team which define “things you can do”. Equipment, budget, and CanSat loading and size capacities are important inputs to define ‘what you can do”. It is essential to distinguish between “things you want to do?” and “things you can do?". In the same way, it is essential to distinguish between “things you want to do?” and “what you can do?". These three different spaces are depicted in Figure 4-3.

As an imaginary example, if you want to install the CanSat with an anti-gravity propulsion device which will allow the CanSat to approach and land to a target point while flying. Whether the device is considered as “things you want to do” or “things you can do” or ‘what you can do”. The engineering significance of installing this device in the CanSat should be considered.

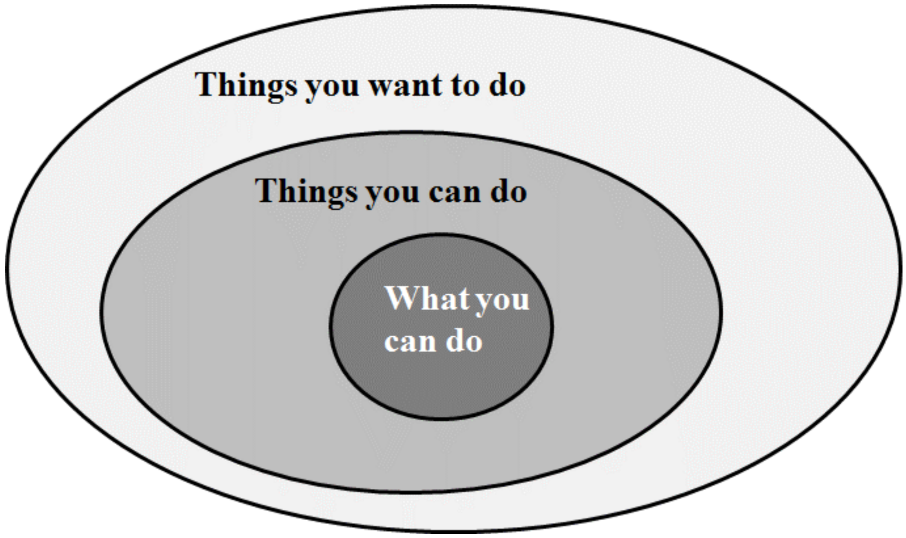


Figure 4-3: Things you want to do, things you can do, and what you can do

4.1.3 **Examples of Breakdown**

Following the procedure described in the previous section, a specific CanSat project is considered. The CanSat project breakdown and design is are discussed in this section. The Tokyo Metropolitan University CanSat project, TMG, which participated in ARLISS 2012, is considered. This CanSat mission, mission items and success criteria are illustrated in Table 4-1.

At that time, the team developed CanSat to perform the mission of putting a fragile object in CanSat, letting it fall and landing it safely to the ground. During the mission, there are many forces that can break the object. Therefore, Tokyo Metropolitan University team discussed the engineering significance of the mission and its relevance to space mission. The team wrote the mission statement and declare it as follows:

Bring a fragile object safely to the ground

Next, mission items should be listed to realize the mission statement. The team used the "fragile objects" as "eggs", " bring safely to the ground " to "let it fall to the ground without breaking", The mission item was written as follows:

Put egg in CanSat, let it fall to the ground without breaking

For this mission item, we define criteria for success evaluation as success criteria. If the verification method itself for evaluating the success or failure is recognized as an engineering result, it can also be included in the success criteria. Success evaluation should be done subjectively for the success criteria. Incidentally, even if it is minimal, those that can be identified as achievement are "minimum success", and those that complete the original mission are called "full success" or "normal success". In addition, something that will result more than initially planned may be defined as "advanced success" or "extra success". It also clarifies the verification methods for how to confirm the success or failure of each item.

Minimum Success

- Launching without breaking eggs
  - [Verification method] Confirm that the egg is not broken after launching
- Deploy the parachute after separation from the rocket
  - [Verification method] By visually inspecting that parachute is deployed without excessive deceleration.

Full Success

- landing without breaking eggs
  - [Verification method] Confirm that the egg is not broken after landing.
- Take a movie of the state of eggs in flight
  - [Verification method] evaluate on presence / absence of moving image data and its recorded content.

List the requirements for realizing the mission items and the success criteria. In the following example, the item number is attached to the request (Requirement), and it is described as R1, R2, and so on. Requests should be thoroughly extracted without ignorance, but how much granularity the request will have will affect subsequent specification decisions. It should also be noted that what is to be listed here is a request and not a specification. For example, if R12 is write as "install a battery to the CanSat for normal operation" instead of “CanSat can supply power for normal operation”, then it is more likely to be a specification not requirement. Because there are several means for power supply and it is decided after considering the specification from the requirement.

- Requirement items of the Tokyo Metropolitan University team.

R1	The mass of CanSat is less than 1050g.
R2	CanSat fits into a carrier with an inner diameter of 146 mm and a height of 240 mm.
R3	CanSat is made from a material that can withstand the load applied when launching a rocket of maximum acceleration (10 G) and random vibration (25 G_{rms})
R4	CanSat has a deceleration mechanism that functions properly during the touch down at the landing point.
R5	CanSat can withstand parachute's load during the deceleration.
R6	Carrier will not affect the performance of the parachute.
R7	Sometime, rockets cannot be launched immediately due to weather condition. Therefore, a method to switch off the power in case of delaying the launch should be installed.
R8	Transmitter should not send any radio waves before separation from the carrier.
R9	To acquire the current position and extract the information necessary for controlling the flight route.
R10	Record the location data of CanSat and conduct the necessary calculations to control the CanSat trajectory.
R11	CanSat downlinks the position information to the ground station.
R12	CanSat can supply power for normal operation.
R13	CanSat can withstand the impact of landing.
R14	The position information should be reliable for successful recovery of the CanSat.
R15	Clear understanding of the position information is essential for successful recovery the CanSat.
R16	Mechanism should be installed to relieve the shock applied to the egg shell.
R17	Having a mechanism which prevents broken pieces from splashing outside when the shell of the egg cracks.
R18	Obtaining the acceleration of the egg over the entire flight.
R19	The target position can be determined easily by the operator.

After extracting the requests, the specifications can be listed one to one correspond to each requirement. In the example of the Tokyo Metropolitan University team, item numbers are added to the specification, S1, S2, and so on which corresponds to the item number of the requirement. Although it might help to decide the specifications carefully but it is not limited to those that have possibility to change in future development (e.g. material). Also, depending on the item there may be almost no difference from the requirement (e.g. R2 / S2). Even so, it is important that the specification was defined from the requirements.

- Specification items of Tokyo Metropolitan University team.

S1	Weight of CanSat including shock absorption mechanism is kept being less than 1,050 g
S2	CanSat's size is smaller than 146 mm in diameter and 240 mm in height.
S3	CanSat can withstand the load applied to the launch vehicle during the launching with a large acceleration of up to 10 G, and the random vibration of up to 25 G_{rms}. The bolts of structure do not untighten during the launch.
S4	Use the parachute which fits into the carrier with the inner diameter of 146 mm and the height of 240 mm as the decelerator performance.
S5	Parachute line should not cut off during rocket launch.
S6	Use the proper wire anchoring the CanSat to the parachute without causing any damage to the CanSat.
S7	Identify the mechanism that should be switched off the power of CanSat until separation from the rocket is detected.
S8	Identify the mechanism that should switch off the radio transmitter until separation from the rocket is detected.
S9	Using a GPS that can receive data every second to acquire current position with high accuracy.
S10	After separation from the rocket and during flight, landing, and other mission sequence, the data that must be recorded is time (6 bytes) and position (14 bytes) and control input (2 bytes) which has a total of 22 bytes each second for 2 hours (total of 2,640 bytes of storage capacity).
S11	To facilitate recovery, downlink the current position to the ground station using a communication device.
S12	Install two batteries for stable power supply.
S13	Designing a structure that does not breaks even if it falls freely and collides with the ground at a speed of 200 km/h
S14	Two operational modes are implemented (Flight Mode and Read Mode), so the recorded data couldn't be erased. The mechanism that switches between the two modes should be identified.
S15	To prevent accidental turning on of the power, the battery can be removed from the CanSat main unit.
S16	Design mechanism that can absorb the shock applied to the egg compartment.
S17	Designing mechanism to cover the whole egg when it breaks.
S18	Install a camera to capture, send and save the egg shell images during the mission.
S19	In the field, prepare a separate file to store the target location like "target.h". The file should be separated from "main.c" of the installed program so that the mission can be done simply by setting the target

Once the specifications are completed, define the system by grouping things of similar specifications. Furthermore, the interface (I/F) of each subsystem is defined, then it is summarized as Interface Control Document (ICD). The ICD describes a hole position, a thread diameter, connector shape, pin assignment, voltage level, current capacity, signal type, and protocol. It can also describe the mechanical, electrical, informational, and thermal aspect of the subsystem. It should be detailed. A brief description is considered in this chapter for sake of simplicity. An effective way to visualize the type of interface among the subsystems is shown in Figure 4-4 and it is commonly used in ICDs.

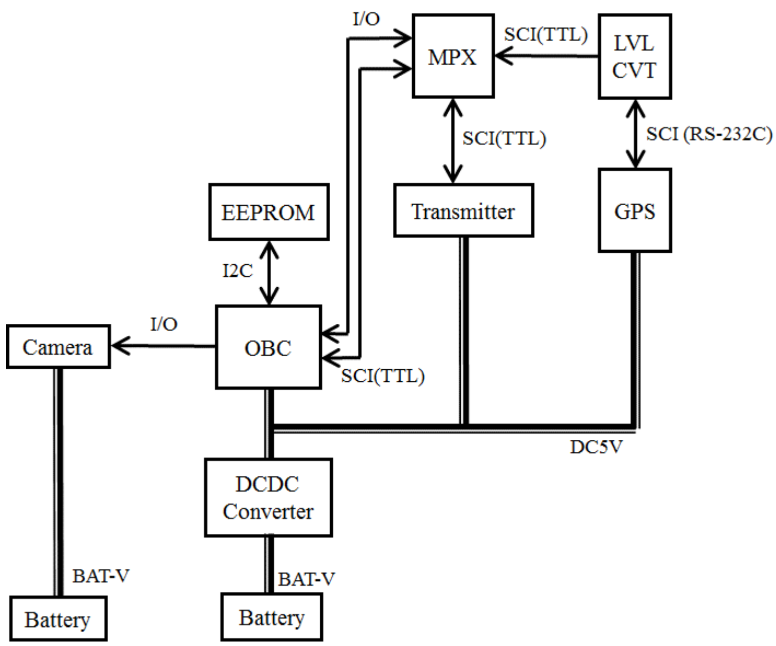


Figure 4-4: The interface (I/F) among the subsystems.

4.1.4 Mission Modules Used in CanSat

In CanSat mission, each team builds their own mission subsystem, but generally there are commercially available sensors that are often used. Few of these sensors that can be used for CanSat mission are introduced in this section.

4.1.4.1 Position Sensor

A positioning sensor is used to acquire the position of CanSat. There are some positioning sensors compatible with American GPS and Russian GLONASS, but GPS is commonly used in Japan. There are various kinds of positioning sensors, such as GH-85 manufactured by Furuno Denki Co., Ltd. and GT-723F manufactured by Canmore Electronics Co., Ltd. The output from the position sensor is compliant with NMEA 0183 format and the manufacturer's own format. The output data from the positioning sensor includes the position information (latitude, longitude, altitude) of CanSat, as well as the speed. Attention should be paid to the difference between the north latitude and the south latitude, the east longitude and the west longitude in the positioning, an appropriate geodetic system is selected and set (typical system is the world geodetic system 1984 (WGS 84)).

4.1.4.2 Camera

When capture images from CanSat, cameras are installed. In the case of acquiring images at short time intervals or acquiring moving images, it is necessary to select and purchase a product that realizes a required high processing speed, or to make a circuit using FPGA etc. by oneself. On the other hand, if one image is taken every several tens of seconds, then it is relatively easy to use serial communication to transmit and save these images.

4.1.4.3 Gyroscope Sensor

Gyroscope sensor measures angular velocity of the flying object like CanSat around anybody axis in degrees per second. By integrating the output data, it is possible to estimate the attitude of CanSat. The S.T.L. Manufacturer in Japan produces gyroscopes. The model number 21 and 41 are commonly used in CanSats because of its low cost. For extremely high-precision measurements, expensive model such as ADIS 16488 manufactured by Analog Devices is a good option. In addition to the three-axial gyroscope, ADIS 16488 includes three axes accelerometer and magnetometer.

4.1.4.4 Accelerometer

Accelerometer measures large acceleration caused by the launch of CanSat, separation from rocket, and deployment of parachute. ACA302/ACB302 manufactured by Star Precision Cooperation, AS-3ACC-3 manufactured by Asakusa Giken Co., Ltd., CXL series manufactured by Silicon Sensing Systems Japan, KXM 52-1050/2050 manufactured by Akizuki Electronics Co., Ltd., and the like are commonly used in CanSats. However, it is necessary to specify the maximum anticipated acceleration during the launch to select the suitable accelerometer because each accelerometer has its maximum limit of acceleration that can be measured.

4.1.4.5 Barometer

Barometer measures the atmospheric pressure during the mission, it is used to estimate the altitude of CanSat from the measured atmospheric pressure. Attitude estimation through barometer measurements is commonly used when deploying parachute or parafoil after reaching certain altitude to avoid high wind regions. SCP1000-D01 module manufactured by Akizuki Electronics Co., Ltd., MPS-2407-015AD manufactured by Metrodyne Microsystem Corp., and the like are commonly used in CanSats. It should be noted here that when measuring the atmospheric pressure, there are two types of measurements. They are the absolute pressure measurement and the gauge pressure measurement, and each type of measurement require the suitable sensor. Absolute pressure sensor measures the pressure with respect to the absolute vacuum. However, gauge pressure sensor measures the pressure relative to some reference pressure which is commonly the atmospheric pressure.

4.1.5 Application to Satellite Development

The topics presented above can be applied directly to actual satellite development. However, CanSat's flight can only operate for few seconds or minutes in the atmosphere, and multiple flights in a short period of time can be carried out. So, continuous improvement and troubleshooting can be conducted based on flight test results. On the other hand, the satellite is launched only one time and it cannot be recovered. Therefore, careful design and testing should be conducted before the launch to ensure that it will operate in space as planned. This is one of the major differences between the satellite and CanSat.

Furthermore, compared with CanSat system diagram, as shown in Figure 4-5, the system architecture scale of the well-known class of Nano-satellites which is the CubeSat is shown in Figure 4-6. It is clearly shown the Cubesat system is more complicated than the CanSat system. Therefore, the number of mission items in the satellite to be extracted and the corresponding specifications are very large. Also, it is said that the number of test items necessary for the entire system to operate normally, or without difficulty is empirically proportional to 2^N, where N is the number of subsystems that constitutes the whole system. This is because the number of test items is the number of combinations of all systems, i.e., ${}_NC_1 + {}_NC_2 + ... + {}_NC_N = 2^N$, for example, the number of test items is 31 and 63 when N is 5 and 6, respectively. The test items are doubled by just adding one subsystem.

In CanSat, it is possible to increase one sensor during the development, but adding only one sensor will not increase the work effort like in satellite system. It is important to master this through the development of CanSat and its experience will be of great importance in developing satellite systems or similar complex system.

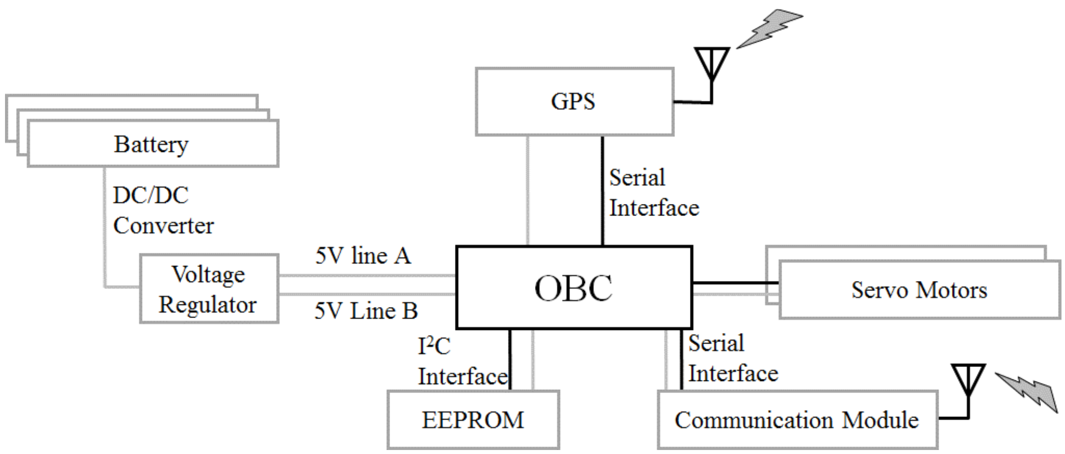


Figure 4-5: CanSat system block diagram.

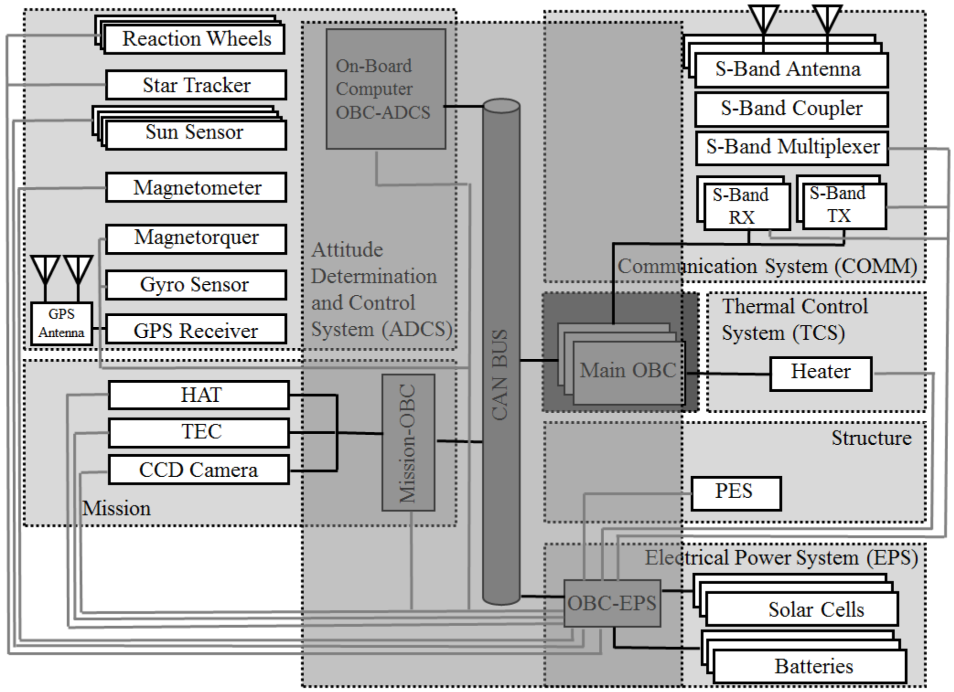


Figure 4-6: Satellite system block diagram.

4.2 Command and Data Handling (C&DH) Subsystem Design

For the satellite to realize the expected mission in orbit, the satellite autonomously needs to control its behavior. There are a need to get the information of various sensors installed on-board the satellite, decide the situation of the satellite, report the status of the satellite from the telemetry information received by the ground station, appropriately receive the command from the ground station and appropriately change the operation status. As the mission becomes more complicated, it is sometimes necessary to execute a mission sequence autonomously or to analyze the acquired data on-board the satellite. A Command and Data Handling subsystem (C & DH) realizes such functions. C&DH subsystem is called the brain of the satellite. The quality of the C&DH subsystem not only affects the capacity of the satellite but also directly affects the reliability and manufacturing ability of the satellite. The C&DH subsystem is realized as a subsystem including both the hardware and software mainly for on-board computers. In this section, the design of the C&DH is presented.

4.2.1 Requirements for Command and Data Handling (C&DH) Subsystem

The C&DH subsystem has electrical interfaces with almost all the on-board components which have variety of functions, but it can be categorized into the following three functions.

4.2.1.1 Equipment Interface Function

Although satellites are made up of several components such as sensors, actuators, communication devices, etc., the C&DH subsystem needs to collect information about these devices and control each device. The state of the Electric Power Supply (EPS) subsystem is very important information for the operation of the satellite, and to maintain the attitude of the satellite, an appropriate command must be given to the actuators based on the information from the attitude sensors. One of the most important functions of the C&DH subsystem is to establish the proper interface with these components.

4.2.1.2 Telemetry and Command Processing Function

Once a satellite is launched into space, its state can only be known by telemetry information send from the satellite to the ground station. Without telemetry information, it is not exaggerated to say that the satellite does not exist. In addition, to operate the satellite, it is often necessary to send instruction from the ground station. These instructions and the data acquired from the on-board components are processed by the C&DH subsystem and different actions can be resulted like change the attitude control mode of the satellite, switching on or off subsystem and so on. Communication subsystem is the physical interface which responsible about transmitting command/telemetry and the interpretation of their contents, for example what kind of instructions and what kind of contents are included in telemetry, what kind of command is prepared, how to change the control state according to the interpretation of these commands, etc. Most of these functions are carried out by the software of the on-board computer which is the function of C&DH subsystem.

4.2.1.3 Autonomous Control Function

If the satellite is in a state where it can communicate directly with the ground station, the remote control by sending a command based on the telemetry information can be implemented. However, the time to communicate directly with the satellite depends on orbit and it is very limited. In general, in the case of operation in Low Earth Orbit (LEO), the communication window is about 10 minutes for one or several times per day. Outside of this window, the communication with the ground station is not possible and the satellite inevitably needs to acquire its own state and control itself. There are many autonomous control modes such as attitude control, electric power control, as well as mission sequence control. The execution of specific autonomous control mode depends on the situations of the satellite.

4.2.2 Subsystem and Device Interface Functions

One of the most important functions of the C&DH subsystem is to realize an electrical interface with the on-board components and there are various kinds of electrical interfaces depending on the components. C&DH subsystem needs to be designed so that these interfaces can be accessed by software via the hardware. Designing the interface function is a task of managing the interfaces of these components and allocating resources without excess or deficiency. In this subsection, the fundamentals of electrical interfaces between components is considered first then how to manage each interface among them is presented.

4.2.2.1 Fundamentals of Electrical Interface (Physical Layer)

Let's say you want to tell set of numbers to a friend who is far away from you so that your voice can not be heard. What would you do if you have two electrical wires between you and a friend? As a simple idea, it is acceptable to give each number a specific voltage between the electrical wires and the friend is asked to measure the voltage. If you have determined the relationship between the voltage and the number in advance, you can easily send numbers by adjusting the voltage between the electrical wires. If you change the voltage in specific sequence, then you can send a waveform. Such form of interface is called Active Analog Interface (AA) in satellite development, as shown in Figure 4-7. To realize such interface in the on-board computer, the voltage is read with analog-digital converter.

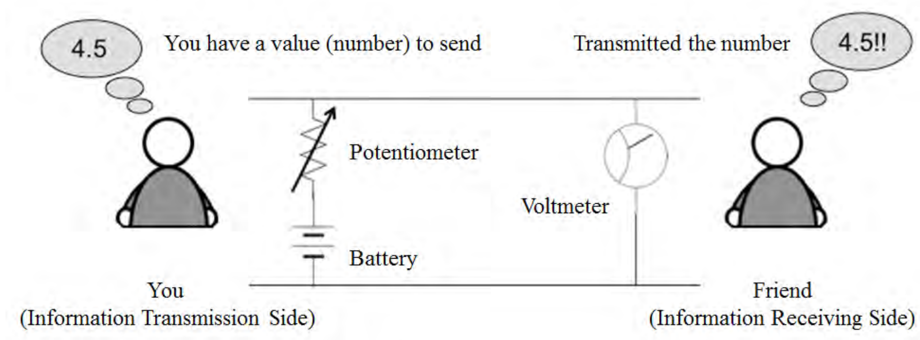


Figure 4-7: Active Analog (AA) interface.

What is important for managing such interface is "the relationship between voltage and number". If this relationship is not accurate, the message cannot be interpreted correctly. Similarly, the range of voltages given is also important. If a voltage is applied beyond the range allowed by the receiving side, the value cannot be read correctly, which in the worst case causes component to failure. On the other hand, if the range is too narrow, the issues of the noise and resolution can be of great concern. Some kind of compatibility between the sending and receiving sides is needed. Also, it is necessary to specify the frequency of sending the values. If the sending side changes so quickly, the numerical values cannot be read correctly by the receiving side who reads at low frequency. In summary, for AA interface, it is necessary to manage at least the range and frequency of change of the voltage.

In the same way, what if you do not have a device to generate voltage on your side? A similar interface to the Active Analog (AA) can be realized if the voltage can be supplied from the side of the friend and you can change the resistance between the two lines. Such a form of interface is called Passive Analog Interface (PA) in satellite systems, as shown in Figure 4-8. An example of Passive Analog (PA) Interface components are the temperature sensors, which are so small that they are not supplied with power and are not capable of generating voltage.

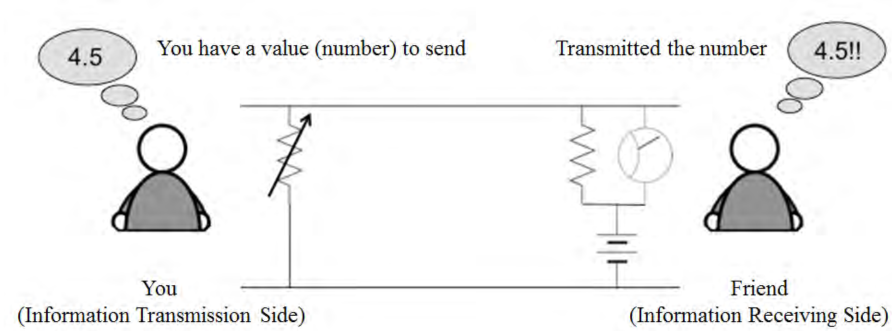


Figure 4-8: Passive Analog (PA) interface.

In PA interface, as well as AA interface, it is necessary to manage the range in which resistance changes and the frequency of this changes, but in the case of PA interface, it is also necessary to be aware of the influence of the connecting lines. Although the sensors and the on-board computer are connected by cables and connectors, resistances are generated not only in the sensors but also in these cables and connectors even though it is very small. Especially when the cable length varies, there is a danger that the amount of resistance will change. Therefore, when finally mounting the PA interface sensors, it is necessary to re-calibrate the sensors to include the effect of cables and connectors resistance.

The analog interface such as AA and PA is very simple and intuitive, but there is a problem in accuracy. For example, when the receiver receives a voltage of 4.9 V, actually it might be 4.91 V or 4.89 V, but it is received 4.9 V.

Therefore, a way of sending accurate numerical value should be developed. Let's assume the information you want to send is of binary nature like ON and OFF which is corresponding to 5 V and 0V, respectively. Then, if a friend receives 4 V, it can judge that this is fairly ON. Even if the cable is long and the voltage drops, the influence can be compensated to some extent. This is the basis of the digital interface. Such a way of interface is called Discrete (DC) interface in satellite systems, and it is used when handling simple binary information such as switches and simple operation statuses, as shown in Figure 4-9.

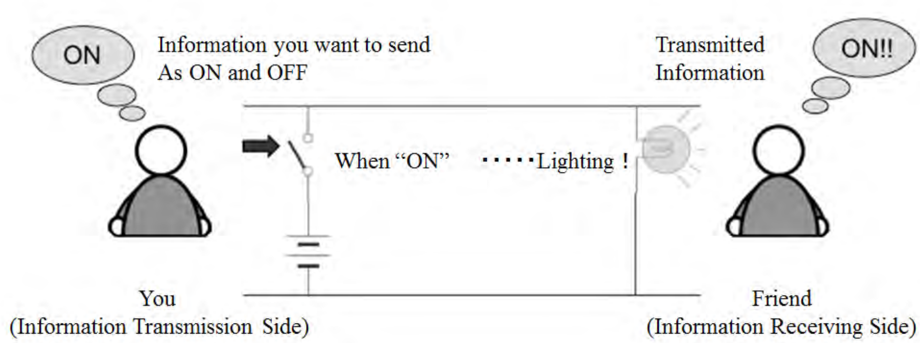


Figure 4-9: Discrete (DC) interface.

In the case of DC interface, OFF condition is represented when the voltage difference between the signal lines are the same, but it is necessary to decide what the voltage difference value that should be used to represent the ON. There are several typical values such as 5 V, 3.3 V, 2.5 V depending on the electronic device, and in Field-Programmable Gate Array (FPGA) devices, it can be changed by the voltage supplied for each channel in some cases. It is necessary to keep consistency between the voltages used on the sending side and the receiving side. If the sender side and the receiver side are different, it is necessary to use level conversion IC or similar component for adjustment. Also, in the Discrete (DC) or Digital interface, the input impedance on the receiver side is generally set high enough when handling the signal voltage, but it is necessary to pay attention to match the out of phase characteristics at the interface point.

Although DC can accurately transmit information, DC can only pass two pieces of information (called 1 bit), ON and OFF. How can we exchange more information in the same way? As an obvious idea, increasing the number of lines will increase the information that can be sent at the same time. Assuming that N sets of similar signal lines are prepared, each signal line can send two pieces of information, ON and OFF, so that 2^N kinds of combinations can be made. Here, if a clock signal indicating the timing of reading data can be added, not only a steady value but also a changed value, a numeric string, etc. can be sent. The idea of sending collectively many data by using a set of signal lines is called a parallel interface. Examples of the parallel interface include General Purpose Interface Bus (GPIB) used for measuring instruments, Small Computer System Interface (SCSI) used in older hard disks, Peripheral Component Interconnect (PCI) bus and the like are well known. Since it requires a lot of lines it is a delicate structure to the timing and impedance among the signal lines. Therefore, it is not used in computer boards as bus architecture.

As a related concept, let's discuss the differential interface and the single-end interface. As explained, N signal lines are required in the parallel interface, but there are two ways of interpretation in their usage. One is that the voltage difference of two wires originally represented one signal, so it is a concept to prepare N pairs of signal lines, that is, 2N signal lines in total. Such interface is referred to as a differential interface because it reads a signal with the voltage difference in each pair. On the other hand, if only one signal line (referred to as a ground line) representing a certain reference voltage is assigned and the other signal lines are compared with the reference voltage, the number of the signal line is equal to the number of the electrical wires. Then it is enough to have N signal lines. Such interface is called a single-end interface. These concepts of the differential interface and the single-end interface are also common to the serial interface which is described later.

Although the single-end interface can reduce the number of signal lines, it also can reject the external noise as in the differential interface. It is suitable for high speed because the voltage difference can be set small. There are many merits such as not being affected even if the level of the reference line fluctuates, and it is often used properly depending on the application. There are standards called RS 422 and Low-Voltage Differential Signaling (LVDS) which are differential interface. In satellite systems, a differential interface tends to be used to separate the reference line and the signal line. When managing the electrical interface, it is necessary to identify whether the digital interface is differential or single-end, together with the reference voltage.

Although the parallel interface can send multiple data at once at the cost of increasing the number of lines, it is delicate and difficult to handle with the timing and impedance between lines. So, what have to be done in case a lot of message to be sent? In the explanation of the parallel interface, it explained that values can be sent sequentially by adding a clock signal indicating the timing to read. Using this fact, two lines can be used to transmit a message, one for the signal and the other for the clock, then send a serial of numbers one bit at a time with one set of signal lines. In this way, a method of sending serially numeric strings with one signal line is called a serial interface. In particular, the method of serial transmitting data using the clock signal is called a synchronous serial interface, as shown in Figure 4-10. The Inter-Integrated Circuit (I^2C) interface and Serial Peripheral Interface (SPI) are typical examples of synchronous serial interfaces, and these interfaces are often used for satellites systems and the like. In the synchronous serial interface, one of the devices must generate the clock signal. It is necessary for devices to share the roles. A device that generates a clock signal (not necessarily a device on the transmitting side) is called a master, and the device that operates in response to the clock signal is called a slave.

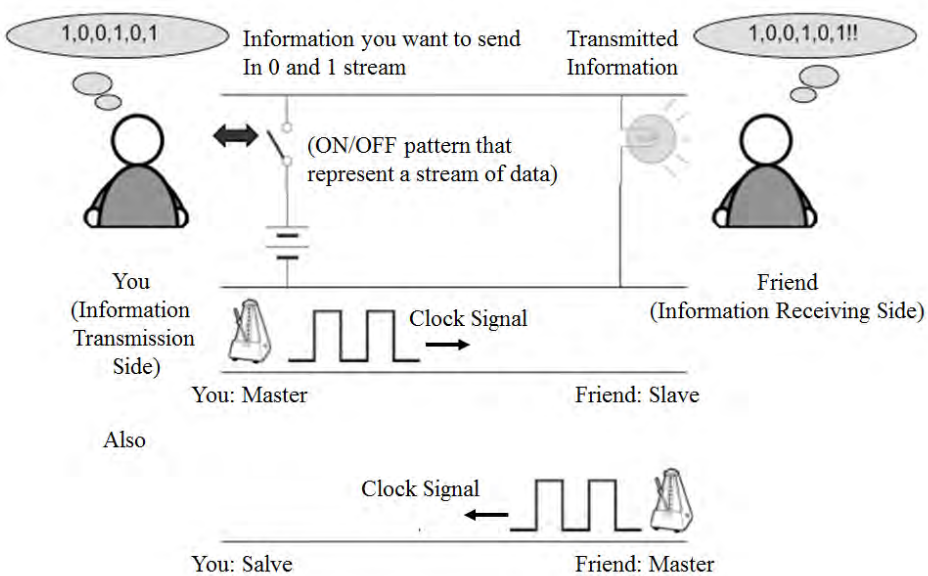


Figure 4-10: Synchronous serial interface.

When managing the synchronous serial interface, it is necessary not only to grasp the communication standards such as I^2C and SPI but also to manage the distinction between master and slave. Also, in the synchronous serial interface, data is sent in accordance with the clock signal, but there is a danger that it cannot be received if data is sent at very high speed. It is also important to check whether the communication speed can be matched among the synchronous serial interfaces. In the synchronous serial interface, data is transmitted and received one bit at a time according to the timing of the clock signal. To obtain the correct data several issues should be considered, specifically; the reading is carried out at the rising timing of the clock signal or at the falling timing and the data is sent from the Most Significant Bit (MSB) or Least Significant Bit (LSB). Particularly in SPI, these combinations are allowed, so there is a danger of getting incorrect values if these combinations are not set correctly.

In the synchronous serial interface, it is necessary to use the clock signal, but what if such signal is not available? A Universal Asynchronous Serial Interface (UART), which is widely used in on-board equipment, is a typical method of realizing serial communication without using a clock signal, as shown in Figure 4-11. In the UART, a data starts with what is known as a starting bit to be sent when communication is established with specific speed. The number of bits transmitted per second, bps or baud rate, is determined. The receiving side monitors the voltage of the signal line at a frequency of 1/16 of a time interval of 1 bit, recognizes that the communication is started when receiving a signal corresponding to the start bit, and thereafter decide the time interval. By reading data serially, it is possible to receive data even without a clock signal.

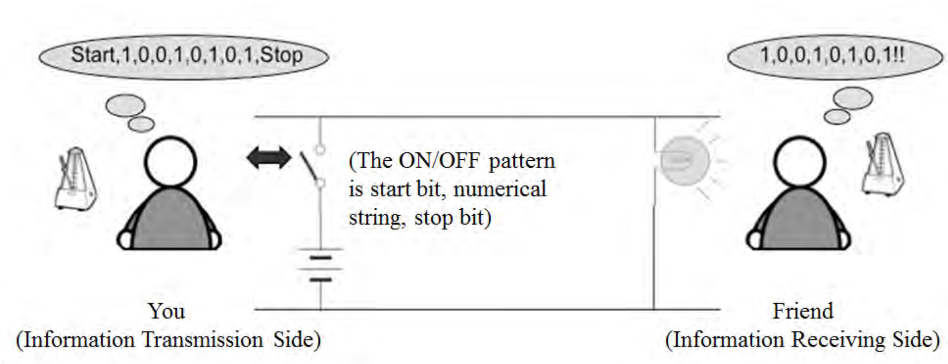


Figure 4-11: Asynchronous serial interface.

In this way, it is necessary to adjust the baud rate between the transmitting side and the receiving side in the UART interface. Also, unlike the synchronous serial interface, role assignment such as master/slave is unnecessary, but there is a distinction between the sender and recipient. Like a transceiver, it is also possible to switch between transmission and reception. For reception, it is necessary to monitor the signal line, and there is a danger that the signal will happen to coincide.

4.2.2.2 Design of Electrical Interface

Since there are various kinds of interfaces of the on-board devices and in order to realize the electrical interface with the on-board devices, the information about the electrical interfaces of the on-board devices should be appropriately arranged. It important in the design to cover, all the required interfaces with the on-board computers. For that purpose, it is essential to create an interface management table that summarizes the interfaces with the satellite's on-board devices, as shown in Figure 4-12 and Figure 4-13. In these examples, the following items need special attention:

○ Interface type: UART, SPI, AA, PA etc.
○ Voltage level and voltage range
○ Single-end and differential interfaces
○ Communication speed

Electrical Interface of the Attitude Control OBC

Component	Abbreviation	Interface				
		Type	Level (V)	Input (I)/Output (O)	Number of Channel	Route
Gyro	GYR	UART	RS422	IO	1	
		DC	3.3	O	1	
Star Tracker	STT	SPI	3.3	IO	1	
		DC	3.3	O	4	
		DC	3.3	IO	1	
Sun Sensor - 1	NSAS1	UART	RS422	I	1	
		DC	5	O	1	
Sun Sensor - 2	NSAS2	UART	RS422	I	1	
		DC	5	O	1	
Sun Sensor - 3	NSAS3	UART	RS422	I	1	
		DC	5	O	1	
magnetometer	GAS	AA		I	1	
GPS Receiver	GPS-R	UART	TTL	IO	1	4.8kbps
		DC		I	1	
Reaction Wheel - 1	RW1	UART	RS422	IO	1	
		DC	3.3	I	3	
Reaction Wheel - 2	RW2	UART	RS422	IO	1	
		DC	3.3	I	3	
Reaction Wheel - 3	RW3	UART	RS422	IO	1	
		DC	3.3	I	3	
Reaction Wheel - 4	RW4	UART	RS422	IO	1	
		DC	3.3	I	3	
Power distribution units (including magnetorquer)	PDU(MTQ)	AA	5	I	4	
		DC	5	O	6	
Mission OBC	MOBC	SpW		IO		
		UART	RS422	IO	1	

Electric Interface Table for the Mission OBC

Component	Abbreviation	Interface				
		Type	Level (V)	Input (I)/Output (O)	Number of Channel	Route
Attitude Control OBC	AOBC	SpW		IO	1	
		UART	RS422	IO	1	
Power Control Unit	PCU	UART	RS422	IO	1	
Power Distribution Unit	PDU	UART	RS422	IO	1	
S-Band Receiver	S-RX	UART	3.3	IO	1	
		DC	RS422	I	3	
S-Band Transimitter	S-TX	DC	RS422	O	2	
X-Band Transmitter	X-TX	AA		I	2	
Temperature Sensor	THERM	PA		I	8	
Mission Equipment	MISSION	UART	RS422	IO	1	

Integrated Interface Table

Type	AOBC	MOBC	Integration	Line	Connector
RS422	10	1	10	50	MDM-25×2
SPI	1	1	3	12	MDM-25
DC IN	13	3	16	32	MDM-37
DC OUT	14		16	32	MDM-25
AA	8		8	16	MDM-37
PA		8	8	16	

Figure 4-12: Example of interface management tables.

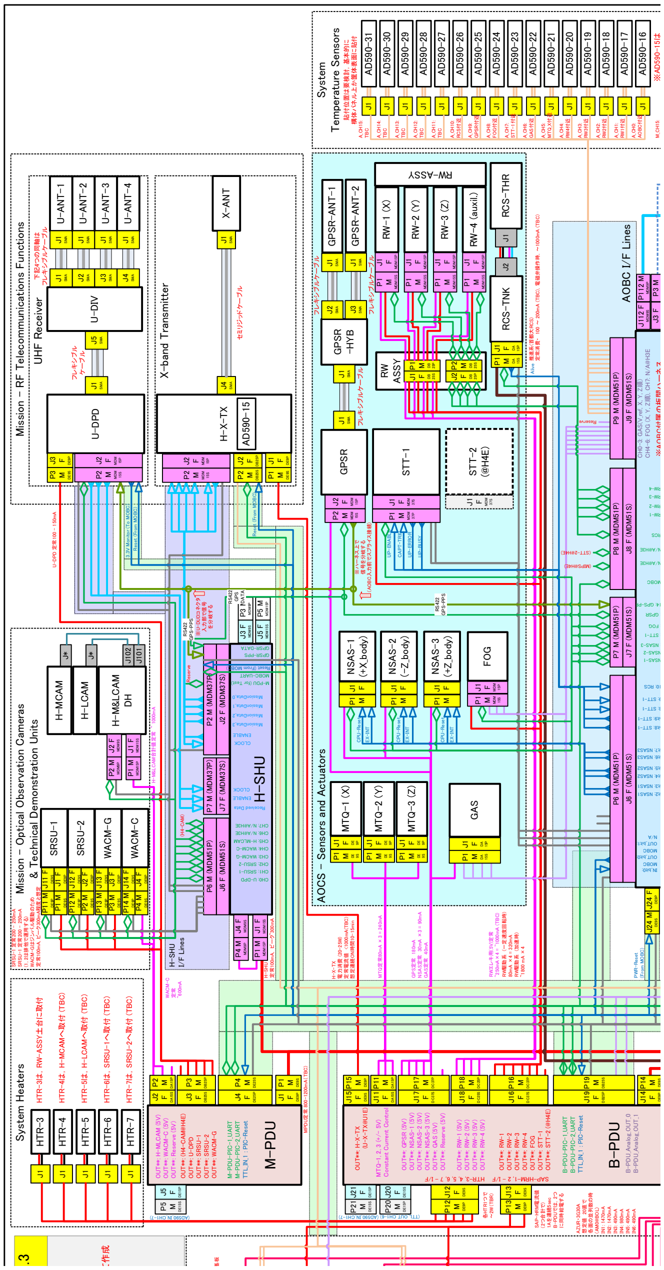


Figure 4-13: Example of interface diagram.

4.2.2.3 Protocol Compliance

In addition to the electrical interface of the physical layer which is described above, it is also important to correctly understand the structure of the data delivered via the communication path, mainly for software development. Installed components exchange information in a predetermined format, as shown in Figure 4-14. For example, a certain device sends information corresponding to sending a specific command or when a device turns on the power, information is continuously sent automatically in a predetermined format. There are several formats. The content of the sent data and the format of the data are defined according to standard format of the device. Such standards concerning data transmission and reception are referred to as protocols. To extract the necessary information from the data to be transmitted and received, it is necessary to develop software to implement such a protocol. Since such protocols are generally described in datasheets and Interface Control Document (ICD), it is desirable that they are integrated with the above physical interface.

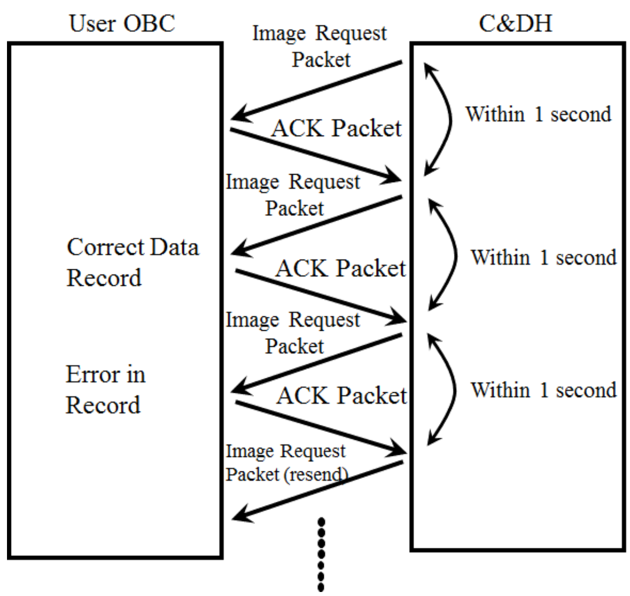


Figure 4-14: Image transfer protocol.

4.2.3 Telemetry and Command Processing Functions

Once the satellite is launched, the telemetry information becomes the only way to know the status of the satellite. Instructions to the satellite can only be done through commands from the ground station. On the other hand, the bandwidth of the telemetry communication from the satellite to the ground station is limited, and the information that can be sent from the satellite is limited, and there are commonly limited kinds of commands that the on-board computer can process. Therefore, extracting necessary and sufficient information from the satellite and putting it as telemetry information has a great influence not only on the function and operation of the satellite but also on the survival of the satellite.

4.2.3.1 Telemetry and Command Basics

Telemetry and commands require the content and arrangement of data according to standard protocols between ground station and satellite. However, the data format in this protocol can be freely decided by the developer according to design and operation of the satellite. How data are structured in a specific format?

First, to acquire a reliable telemetry information and its associate time stamp, it is necessary to know the reception timing. If the telemetry frame is started with a specific symbol (start code), the telemetry frame can be seen as soon as this symbol is detected. To avoid encountering similar symbol in the content of the data by chance, these start codes are often given as two consecutive characters with 0X80, 0X80 or 0X10, 0X02, etc. instead of one character. In some cases, it is desirable to use a code that is easy to implement in the ground station, therefore the specifications of the ground station are important.

Assuming that the starting point of telemetry could be identified, the next problem would be to know the length of one received frame. The telemetry and command presented in this context are of the certain length. It is easy to talk about receiving data of a certain length at any time. For telemetry, it is often the case that a frame using the full communication band is usually transmitted, so that it is often a fixed frame (fixed length frame). On the other hand, as for the command, the necessary parameters are often different depending on the type of the command, so that the length of the frame is often different depending on the command (free length frame) in many cases. In such a case, it is necessary to know the length of the frame or the end of the frame. As a method to do this, the length of the frame is typed as a code within the frame or the length can be identified by a terminating string at the end of the frame. Once the frame is successfully received, the next question is whether it is correct or reliable or not? Are there errors resulted from communication? If errors occur in sending the commands, there is a danger that the satellite will malfunction. How can errors be detected?

The simplest way is to include a specific exit code at the end of the frame. If the end of the received frame is not included an exit code, serious errors occurs in the transmission and the received frame is not reliable. When an error is included in the middle of the data, a more effective way to detect it is to put an error identification code calculated by a specific method in a frame in advance. For example, Check Sum calculated by adding data contents is an example of a simple error identification code. When receiving a frame, the error identification code is calculated the same way it is calculated before sending. If it differs from the sent Check Sum, there is a risk that errors occur in the transmitted data, so that frame is not reliable. There is a Cyclic Redundancy Check (CRC) as a similar but stronger error identification code.

CRC is a specific number called a characteristic polynomial, and uses the remainder when dividing the target character string as an error identification code. In the case of Check Sum, as a simple example, when a character in the data is replaced by error in communication the same Check Sum is obtained at the receiving side and the error is not detected, but such errors can be detected using CRC. CRC is widely used because it is relatively easy to calculate without high computation cost, but since there are so many kinds of characteristic polynomials, it is necessary to always confirm which characteristic polynomial is used. As described above, although the telemetry command frame can be designed freely to a certain extent, if it is structured to have at least the start code, the frame length information, the error identification code, the termination code, etc., as shown in Figure 4-15. This is effective for improving the reliability of the command/telemetry process.

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	....	N-4	N-3	N-2	N-1
	STX	STX	Length	TLM_CMD_DISCRIMINATION	TLM_ID	TLM_COUNT	Time	CMD_CODE	CMD_STS	CMD_ERROR_STS	CMD_COUNT	TLM packet ID	FROM_ID	From_ID_Sub	TO_ID	TLM PARAM		CRC1	CRC2	ETX	ETX
Bits	8	8	8	8	8	8	32	8	8	8	8	8	8	8	8			8	8	8	8
Content				Discriminated TLM or CMD	TLM ID	Count up for TLM	Time	Received CMD code	CMD status	CMD error status	Count up for CMD							Cyclic Redundancy Check Highbyte	Cyclic Redundancy Check Lowbyte		

CMD Frame

Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	....	N-4	N-3	N-2	N-1
	STX	STX	Length	TLM_CMD_DISCRIMINATION	FROM_ID	TO_ID	CMD_EXE_TYPE_ID	Time	CMD_TYPE_ID	CMD_COUNT	CMD_ID	TO_ID_SUB	CMD_PARAM					CRC1	CRC2	ETX	ETX
Bits	8	8	8	8	8	8	8	32	8	8	8	8						8	8	8	8
Content				Discriminated TLM or CMD	CMD sender ID	CMD receiver ID	CMD execution type ID	Time	CMD type ID	Count up for CMD	CMD ID	CMD receiver ID SUB						Cyclic Redundancy Check Highbyte	Cyclic Redundancy Check Lowbyte		

Figure 4-15: Example of telemetry and command frame.

4.2.3.2 Types of Telemetry and Commands

The most basic function of the telemetry and command is to connect the satellite and the ground station, send the current state of the satellite as telemetry, and control the current state of the satellite by command. However, satellites are not always able to communicate with the ground station. When operating satellites with Low Earth Orbit (LEO) from one ground station, it is difficult to communicate with the ground station for most of the time. For the satellite to realize the desired mission, it becomes necessary to know information about the duration when satellite cannot communicate with the ground station, and to control and operate the satellite in this duration.

Therefore, most of the satellites have the function of storing satellite information during the duration when telemetry information cannot be communicated with the ground station. The telemetry accumulated in this way is called stored telemetry. In distinction from this, telemetry which directly downlinks the current state of satellite is called real time telemetry. Although stored telemetry is generally stored in the memory of the On-Board Computer (OBC), since the capacity of the memory is limited and the time window for transmitting accumulated telemetry to the ground station is also limited, software should be written so that carefully selected parameters with specific storage are accumulated.

On the other hand, a command executed within the satellite automatically at a specified time is called a stored command. For telemetry, a command to execute a transmitted command at specific point is called a real-time command. Since the function of interpreting the accumulated stored commands is almost the same for real-time commands, it is possible to manage the types of commands, but it is necessary to secure a space where commands can be stored in the memory of the On-Board Computer (OBC).

In particular, when using stored telemetry or stored commands, time management becomes very important. How to manage the time on each device on the satellite, how to correlate with the time acquired by the ground station, and how to associate the time with the position on the orbit. It can be said that this is a very important issue in the operation of satellites. There are various methods such as configuring the time on the satellite by commands from the ground station and another method of periodically obtaining time correction information from the satellite by downlinking the time information from the satellite on a real-time telemetry etc. It is necessary to consider each method and its effect on considering the mission of the satellite and the required accuracy.

Some satellites also have a function to execute a reset command sequence with a single command. Such function is called macro command, and it is convenient to use a specific operation sequence repeatedly. When using the macro command, it is necessary to consider time management of the macro command, how to deal with interference between the real command and the stored command and the command execution equivalent to the contents of the macro, and so on.

Furthermore, implementing the function to upgrade the program of On-Board Computer (OBC) on satellite makes it useful for various application including bug fixation. Such a command is called a reprogram command. Since it is difficult to know in advance how to upgrade the program of the On-Board Computer (OBC), it is often implemented by transmitting binary data corresponding to the program differences from the ground station and replacing part of the program. Although it is convenient to upgrade the program, there is a risk that the program might be erased. Therefore, it is extremely important to conduct such upgrade process with extreme care and downlink the uplinked data for comparison and error checking. Usually the error identification codes discussed above are not applicable for binary data. In addition to program upgrade, it is also possible in principle to replace the boot program itself written in the nonvolatile memory.

4.2.3.3 Management of Telemetry and Command

As described above, the role in designing and managing the telemetry and command functions as the control/information processing system is to design the function of the corresponding telemetry/ command such as design of the frame structure, the data contents, storage, macro etc. The functional design of the command and its capacity setting, maintenance and management of these information, etc. are diverse. The process of exploring the consistency of various requests among the finite resources is a typical system design practice. It is necessary to proceed while looking at the mission request, system configuration, and resource allocation.

In addition to the above design requirements, additional telemetry items might be considered. A telemetry counter that counts every time a telemetry is transmitted and a command counter that counts the received commands can be implemented easily. This item is very effective for acquiring telemetry and command transmission/reception status. Telemetry items expressing the execution state of commands are also important. It is better to include these as basic items when defining the frame structure.

Although the operation/analysis process becomes slightly complicated, if parametric telemetry expression is realized by changing the contents of a specific place of the telemetry frame according to the parameter condition, the limited resources can be used effectively. Especially when the expression of the telemetry corresponds to a command, if a unique return value for the command is prepared in the telemetry frame, it is possible to flexibly deal with extension of the command etc. Figure 4-16 shows and example of telemetry command management table.

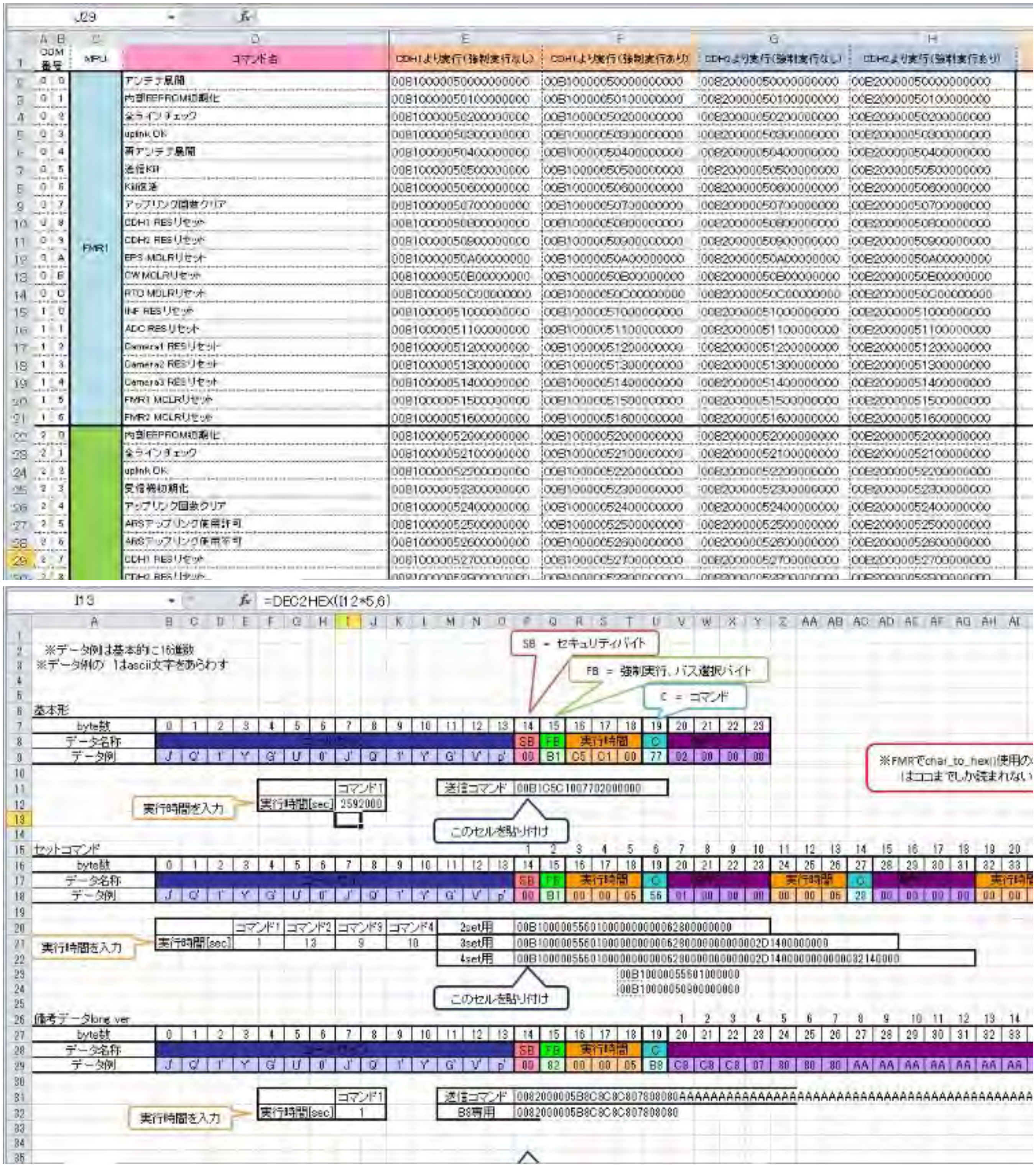


Figure 4-16: Example of telemetry command management table.

4.2.4 **Autonomous Control Functions**

Once the satellite is launched, it is necessary to autonomously manage its own state. During the period in which communication with the ground is impossible, the C&DH subsystem keeps the stability of the satellite itself by acquiring the situation of each device of the satellite and giving appropriate instructions to them according to the situation (Housekeeping). The implementation of autonomous mission needs to be realized. Such autonomous control functions can roughly be divided into the following three categories:

4.2.4.1 Real Time Control

Like the attitude control, the C&DH subsystem needs to establish a real-time feedback loop by connecting the sensor and the actuator, as shown in Figure 4-17. Such control is generally referred to as real-time control. In real-time control, it is important to execute the control program regularly on a reliable basis. Fluctuation or irregular execution interval, freezing in execution, or the like may cause a fatal problem. To implement, it is common to realize a mechanism that executes a specific program simultaneously by utilizing a time interrupt in a real-time operating system. However, the processing at each step should be completed within the time slot. It is required to eliminate the risk of stacking during the processing.

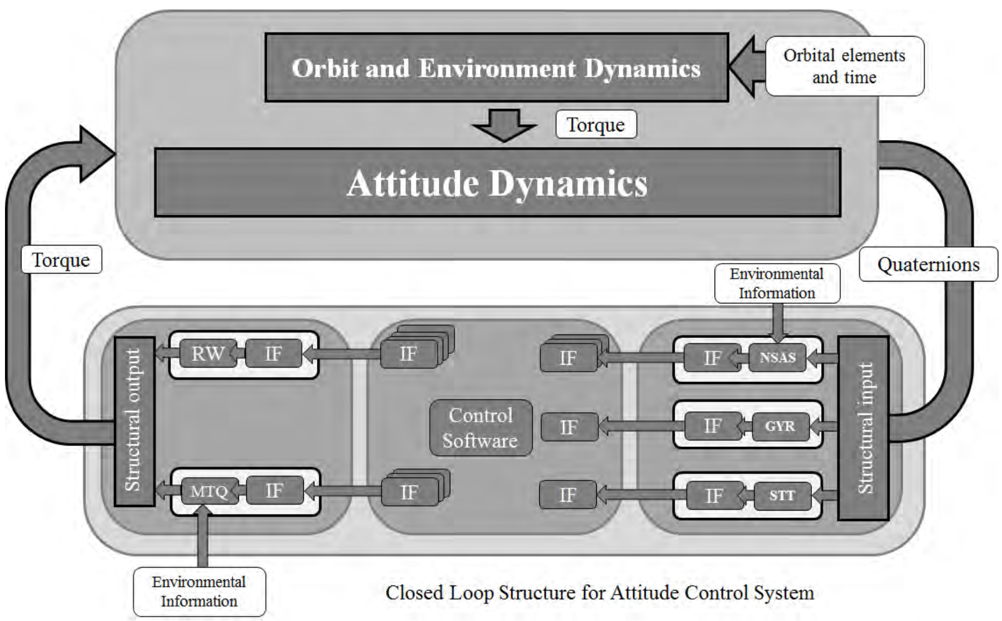


Figure 4-17: Closed loop control of spacecraft attitude.

4.2.4.2 Sequence Control

Satellites may be required to perform some sequences automatically. Especially from the time the satellite is placed into orbit, until the satellite can establish communication with the ground station, the satellite must execute the initial sequence and ensure its survival. In steady state operation, it may be required to automatically execute a sequence necessary for implementing the mission. It is necessary for the C&DH subsystem to reliably execute the sequence during this time. In automatic execution of such a sequence, it is essential to design the program with the foremost objective is to secure the survival of the satellite. In this case, it is desirable that the sequence be made as simple as possible and not complex for decision and execution conditions. If an execution condition is set, there is a danger that the execution condition may not be unintentionally satisfied due to malfunction or other similar condition, or the verification process before launch may become complicated. For such sequences, it can be said that a simple implementation is usually desirable.

4.2.4.3 Abnormal Event Control (Failure Detection, Isolation, and Recovery: FDIR)

Sometimes satellites may encounter partial malfunctions or abnormal events, such as Under-Voltage Control (UVC). It is one of the most important functions of the C&DH subsystem to reliably detect such an event and realize an appropriate countermeasure. Such functions are generally called Failure Detection, Isolation, and Recovery (FDIR). To realize such FDIR control, it is necessary to first analyze possible fault phenomena of satellites by making use of techniques such as Failure Modes and Effects Analysis (FMEA) and Fault Tree Analysis (FTA). Since these analyses cannot be conducted by the C&DH subsystem alone, it is necessary to coordinate with the whole system from the global point of view. As a result, it is necessary for the C&DH subsystem to realize the implementation with responsibility, but like in the case of sequence execution, it is vital to avoid imposing complicated decision condition in this case and to implement a simple condition. Particularly in the case of FDIR, it is concerned with the survival of the satellite, so it is necessary to set the detection condition and biasing the safer one.

4.2.5 The Characteristics of Orbital Conditions

The core of the C&DH subsystem is the On-Board Computer (OBC), which realizes the interface with each component on-board the satellite and is designed to realize the memory and computing capacity requirements. In the fabrication of the On-Board Computer (OBC), space parts are extremely expensive with limited capabilities, so in recent years, Commercial-Off-the-Shelf (COTS) has been widely utilized. The On-Board Computer (OBC) is commonly COTS computer. The orbital environment is different from the ground in some aspects, and it is necessary to design with care in quality with respect to these aspects. Let's think about what are the main differences between the on-orbit environment and the ground environment. As for the electronic components, the orbit is different from the ground in mainly two aspects, it is high vacuum and radiation environment.

4.2.5.1 Thermal Vacuum Environment

There is no air present on orbit. it is natural that the heat of a device is naturally cooled by convection due to the presence of air on the ground. On the other hand, since there is no air on orbit, heat is transferred only by conduction or radiation. Since almost all the electric energy consumed in electronic devices is converted into heat, it is necessary to consider how to release the generated heat. Also, the thermal environment of the satellite as a whole varies greatly on orbit. It is necessary to select a device that can adapt to such thermal environment, and it is necessary to control actively thermal conditions with a heater or the like. Also, when lead-free solder is left in a high vacuum environment, protrusions called whiskers are generated, and there is a danger of causing short-circuit. It is necessary to take measures such as avoiding the use of lead-free solder and coating.

4.2.5.2 Radiation Environment

On the ground, protection against radiation emitted from the sun is provided by atmosphere and the magnetosphere. Radiation environments is the main distinction from the ground when using electronic devices on orbit.

The influence of radiation on electronic devices is roughly divided into two types. One is a Total Ionizing Dose (TID) that causes degradation of electronic devices by continuous irradiation of radiation for a long period of time. Operation failure occurs when irradiating a certain amount or more, and in general the deterioration is irreversible and cumulative. The magnitude of the influence differs depending on the kind of the device and the manufacturing process, and although it can be alleviated to some extent by shielding. In general, it is necessary to conduct radiation test such as gamma ray irradiation during the operation of the device of interest.

The other type is a Single Event Effect (SEE) in which a defect is stochastically caused by a beam of particles. A Single Event Upset (SEU) in which change of state is caused by one single ionizing particle (ions, electrons, photons...) striking a sensitive node in a micro-electronic device, such as in on-board computers, memory, or power transistors. The state change results from the free charge created by ionization in the electronic device causes a short circuit due to the influence of the beam of particles and causes what is known as Single Event Latch-up (SEL). In general, it can be restored by reset, and for SEU, operation can be continued without trouble by repairing the memory contents. Since it is a probabilistic phenomenon caused by beam of particles, the occurrence probability is estimated by using particle accelerator facility, and if the mission is affected by SEU, the error correction protection logic in the memory, overcurrent detection reset circuit, triple majority voting logic, etc. are implemented.

4.2.6 Software Implementation

The C&DH subsystem is composed mainly of On-Board Computer (OBC), and software to operate with OBC is indispensable as well as other hardware to conduct various functions. Therefore, the reliability of software largely affects the reliability of the C&DH subsystem, and the reliability and robustness of the entire satellite.

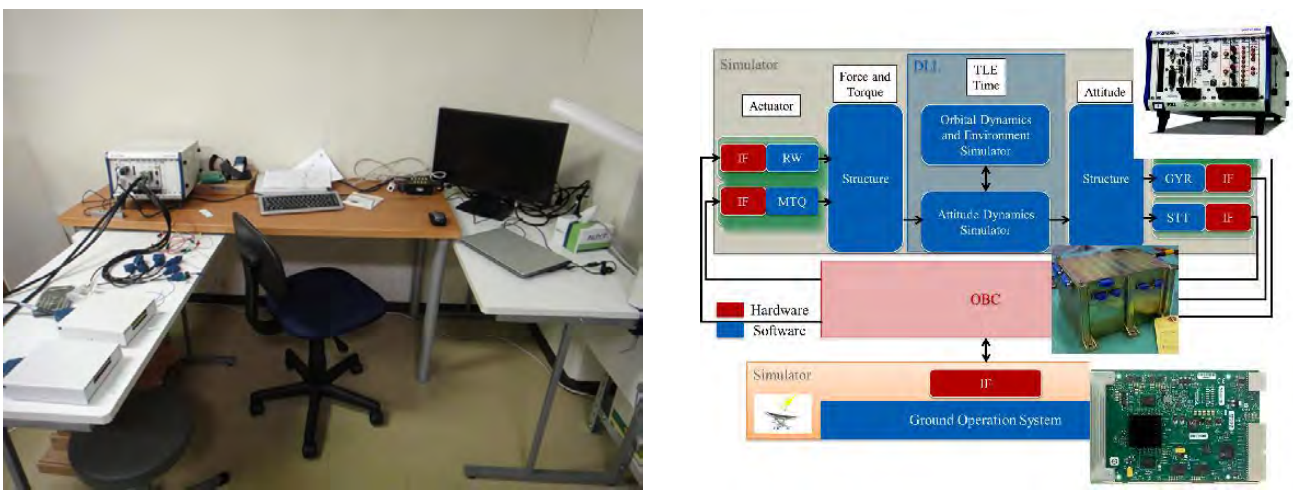
4.2.6.1 Notes on Software Implementation

The program lists the tasks to be performed sequentially, but when software programs are programmed, the simpler and shorter the software structure, the easier it is to understand. Appropriate understanding of the structure of the program is extremely important in avoiding unintentional troubles or bugs. It is especially important when software development is shared by multiple developers.

Decomposing complex tasks into its subtasks can be described as a collection of relatively simple tasks. In system engineering, the Work Breakdown Structure (WBS) which hierarchically decomposes tasks to be performed is well known. WBS can be applied also to the software by dividing the work contents, it is important to properly understand the structure of the program. Furthermore, if division of this work is done with awareness of common points with other work, elemental work can be commonly used by a set of programs, not only is it efficient but also systematically and it also helps to improve the reliability. In this way, in program development, the task of dividing into common functional units is called modularization and performing appropriate modularization is the basis of software development. In addition, most software development environments in recent years can realize software development by combining a set of files. So, if files are divided and managed for each functional group of modules, when a problem occurs, it can be easily identified because it is only concentrated in one file. This increases the reusability.

4.2.6.2 Importance of Software Verification

Although the reliability of software can be secured only after verification of various situations. This verification process is extremely important especially for on-board software development which is required to operate properly under various situations. Also, when software is migrated from a simulator environment to a real satellite, there is a risk of problems such as programming bugs and timing issues due to differences in platforms. Therefore, it is desirable to verify the software by using the real satellite or in a situation as close as possible to the real satellite. Although it is difficult in principle to verify "unexpected event", it is difficult to minimize the "unexpected event scope" by verifying at least all possible situations. It is extremely difficult to cover all such combinations manually. Also, in the case of verification using a real satellite, it may be difficult to simulate various situations on the ground. Therefore, it is desirable to construct a system that realizes efficient verification by freely generating situations, which are difficult to realize on the ground with the real satellite, of the On-Board Computer (OBC) and some or all of the peripheral devices. Such a system is called Hardware In the Loop Simulator (HILS), as shown in Figure 4-18.



a) Experimental setup b) System configuration

Figure 4-18: Example of Hardware in the Loop Simulator (HILS).

4.2.6.3 Utilization of Common Framework

Software is repeatedly used, and by being verified in various environments, reliability is improved. On the other hand, with nano-satellites and CanSat, each team often develops software individually using different platforms, it is difficult to share mutual software programs. If a software framework that can mutually share software programs within the community can be realized, mutual use of software programs as a whole community will be promoted, which will reduce the labor and cost of software development and lead to improved reliability, as shown in Figure 4-19. From this viewpoint, a commonly usable software framework "Hodoyoshi-SDK" for the ultra-small satellite community was developed. "Hodoyoshi-SDK" was originally developed as a prerequisite for specifications on high-performance computers mounted on satellites, but it is possible to use reduced version SDK, and to have complete compatibility in terms of software. The low-cost, ultra-compact, educational computer board "BoCCHAN-1" that can also be used in CanSat was launched, and a CanSat was developed based on using reduced version SDK, as shown in Figure 4-20. It is also an option to utilize such a common software framework. By using the software framework, existing software programs can be utilized, and complicated and advanced programs can be relatively easily realized. In addition, by feedbacking awareness items etc., it leads to improvement of the software reliability of the whole system.

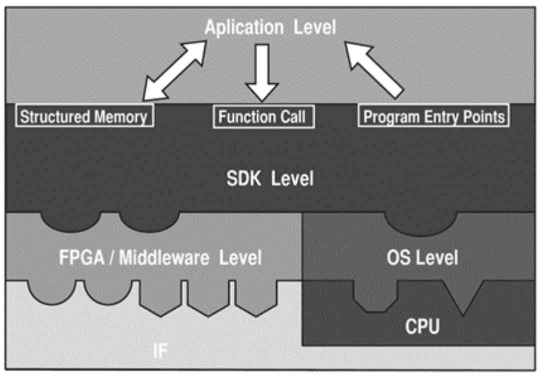


Figure 4-19: Software framework.

Figure 4-20: Hodoyoshi-SDK computer (BoCCHAN–1).

4.2.7 **Command and Data Handling Subsystem in Satellite Development**

The above is a summary of the functions and design considerations that should be provided by the C&DH subsystem. Based on these facts, in summarizing the roles that the C&DH team should take in developing real satellite are as follows:

4.2.7.1 Hardware Design and Development

The team responsible for C&DH subsystem needs to design and develop the hardware that constitutes the C&DH subsystem, including On-Board Computer (OBC). As described above, the On-

Board Computer (OBC) must establish an interface with almost all of the on-board components, and on the top of that, it needs to have resources such as computational power and memory capacity to realize autonomous control. Therefore, to properly develop On-Board Computer (OBC), it is necessary to understand the composition and mission sequence of the satellite. Furthermore, it is necessary to arrange the necessary resources and to proceed with the design while paying attention to environmental factors such as radiation and thermal vacuum environment.

4.2.7.2 Command and Telemetry Management

As mentioned above, the design of the frame structure of the command and telemetry, the contents of the telemetry and the command type are closely related to the software of the On-Board Computer (OBC), so the teams of the ground operation system and the C&DH subsystem should cooperate with each other. In this case, it is necessary to realize the sufficient resource allocation while understanding various resource constraints such as communication resources and memory resources of On-Board Computers (OBCs).

4.2.7.3 Software Implementation

The hardware design of the On-Board Computer (OBC) and the implementation of software operating are one of the most important roles of the C&DH team. Reliability of software greatly influences the reliability of the C&DH subsystem, and thus the reliability and robustness of the entire satellite. First, based on a hardware interface with each component, it is necessary to properly acquire device information and realize device control by supporting the protocol in software. In addition, it is necessary to implement autonomous control functions such as attitude control, FDIR, mission sequence, and other realized functions of the satellite system. Even more important is to carry out the verification process. Functional verification under various conditions is indispensable for the satellite to operate correctly in orbit. From functional verification of each component to the whole satellite, it is necessary to verify in a stepwise and hierarchical manner.

4.2.8 What Can be Learned About C&DH subsystem from CanSat

CanSat is an excellent educational material which is realizing the basic structure of the satellite in a compact manner. Through the development of CanSat, a lot of basic processes about the design and development of C&DH subsystem can be learned.

First, CanSat is composed of several sensors and actuators, and it is required that the On-Board Computer (OBC) to establish interfaces with the mounted components. The training is almost the same as the interface architecture design in satellite development, it is necessary to organize the hardware and software interfaces of each component and establish the interface. With this kind of training, the basics of interface architecture design in satellite development can practically be acquired.

Most of CanSats don't have the capability of remote operation by commands, but in addition to transmitting the CanSat state in real time by telemetry, data stored in the on-board storage media can be accumulated and can be acquired after the CanSat recovery. With the limitation of memory capacity, it is important to understand what kind of data is to be transmitted in real time and what type of data is to be stored in the on-board storage components, how to design the frame structure, etc. The process of basic ideas concerning telemetry design in satellite development can be mastered in the CanSat training.

In addition, CanSat needs to be autonomously controlled after its release, and can conduct experiments on the implementation of the autonomous control function that realizes the mission set by the developer. The CanSat come-back competition addresses such technological challenge of autonomous control. In the process of software development, it is possible to practically acquire a training equivalent to real satellite software implementation, such as implementation of a program that operates in real time, implementation of the verification. Unlike simulation, by using real sensors/actuators, etc. countermeasures about unexpected events and FDIR, etc. can be inevitably experienced.

If CanSat is developed with full awareness of its role and limitations, it can be considered as indispensable educational tool that provide the new satellite developers with the basic skills and techniques to develop a real satellite.

4.3 Electrical Power Subsystem (EPS) Design

The Electric Power Supply (EPS) subsystem plays an important role in keeping the satellite functional. The subsystem is the only source of electric power for the satellite, controls, stores, and distributes energy according to the operational mode of the satellite. After a satellite is launched, its power-on and wake-up process are expected to be controlled by the power subsystem. In this section, the function and design of the EPS subsystem and what can be learnt by using a CANSAT on the EPS subsystem are presented.

4.3.1 The Role of the EPS Subsystem

The role of the EPS subsystem can be summarized by the four functions described in this section. Because every equipment and function of a satellite depends on the electric power generated and maintained by the EPS subsystem, each of its function is critical for the successful operation of the satellite and its survival.

4.3.1.1 Power Generation and Storage

Electric power is generated by the EPS subsystem. The amount of electric power generated by the electric power resources onboard the satellite determines the performance of the equipment of the satellite in a qualitative and quantitative manner. In general, electric power is generated by the solar cells present on the solar panel. The size of solar panels depends on the size of the satellite. Therefore, the size of the power subsystem is highly dependent on the satellite size.

The energy conversion efficiency of a solar cell depends on its power output. Hence, the Peak-Power Tracking (PPT) method is utilized to optimize energy conversion in the solar panel. Such control on energy generation is exercised by the EPS subsystem.

The power storage function is important not only for the satellite to survive when electric power cannot be generated—for instance, when the satellite is in the shadow of the earth—but also when power consumption exceeds power generation. The power storage media, i.e., the batteries, are among the most important components of the EPS subsystem. The charging and drawing of electric power from the battery is also handled by the EPS subsystem. Owing to the fact that the batteries are comparatively heavier in size than the other components of a satellite, battery size directly affects the weight of the satellite. Figure 4-21 shows a system diagram of an example of EPS subsystem.

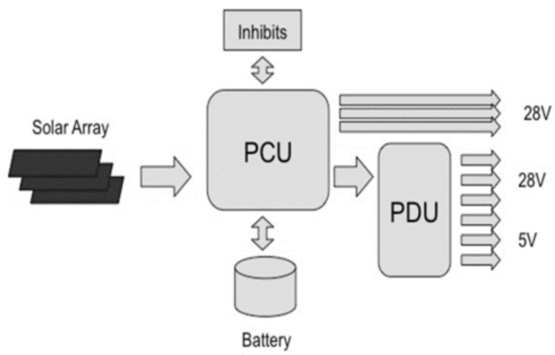


Figure 4-21: Example of Electrical Power Supply (EPS) subsystem configuration.

4.3.1.2 Voltage Conversion, Stabilization, and Distribution

The electric power generated by the solar panel is at a high voltage, and, unstable; therefore, the generated voltage needs to be stabilized and down-converted in order to be used by onboard components. The functions of down conversion, stabilization, and distribution to each component is also carried out by the EPS subsystem. The power supply for each component is controlled depending on its operational mode. Thus, power on/off control is coupled with the power distribution function. In general, the decision to turn the power on or off is made by the C&DH subsystem and the EPS subsystem executes the commands from the C&DH subsystem.

4.3.1.3 Monitoring Power Conditions and Providing Support during Emergencies

The power condition is so critical for satellites. Power conditions such as the level of power generated by the solar panel and the battery charge level should be monitored constantly. Such critical parameters are monitored by the EPS subsystem because these parameters can only be monitored during the power generation and battery charging processes. As the anomalies in power conditions critically affect the satellite survivability, once anomalies in power conditions are observed, they need to be mitigated as soon as possible. The anomaly mitigation process depends on the power condition and is generally implemented in the EPS subsystem.

4.3.1.4 Separation Switch and System Start-Up Control

To meet safety requirements, a satellite is required to turn all power off before it is launched and separated from the launch vehicle. After it is separated, the satellite is expected to turn its power on. Switching between perfect power-off before separation and perfect power-on after separation is performed by using a separation switch (in other words the "kill switch"). The separation switch is usually a part of the EPS subsystem, which controls the power conditions in the satellite. The separation switch is required to satisfy the three-inhibit requirement to ensure safety.

4.3.2 Power Subsystem Design (The Role of Power Subsystem Design Team)

The designing process of the EPS subsystem is closely related to satellite functionality and reliability. Hence, the EPS subsystem must be designed in parallel with the satellite design process. Power subsystem design can be divided into four stages: (1) sizing, (2) resource allocation, (3) function designing and contingency analysis, and (4) hardware designing and development.

4.3.2.1 Sizing Process

The power generation capability affects the satellite's functions directly. The amount of power available decides the possible onboard equipment and architecture of the bus system as well as of mission equipment. In communication subsystem, the affordable power directly affects the communication bandwidth. In the case of the satellite, as solar power is the only power source, the surface size of the satellite limits the amount of generated power. Utilizing the satellite's body surface to mount solar panels is simple and highly reliable, but we can utilize deployable structures to increase the surface area of the solar panels. Thus, there is a tradeoff between reliability and power capacity.

Another critical sizing issue is the sizing of power storage components. Stored power is used by the satellite to function during times when electric power cannot be generated. Stored power also enhances the power capacity when the satellite requires more power than that generated. High-energy consumption missions can be performed by the combined use of generated power and stored power. Sizing of power storage components affects the satellite's weight because the battery is among the heaviest components in satellites. Therefore, we need to balance the satellite's weight and power storage capability requirements. As discussed above, the sizing of power generation and storage components is essential to achieve optimum balance between satellite's power capacity and power requirement. Thus, the first step of the design of the EPS subsystem, i.e., the sizing of power generation components, is very important in the early stage of satellite design to determine the size, structure, and capacity of the satellite. The first important task that needs to be undertaken by the EPS subsystem design team is to estimate the approximate power demand of each subsystem and mission to determine the overall power demand. This estimated power demand should be summarized according to operational phases and modes. Constructing and utilizing a power budget table is a standard practice. In coordination with the structure subsystem design team, based on the summarized power demand, the EPS subsystem design team should decide the approximate size of power generation and storage components.

4.3.2.2 Power Resource Allocation and Power Interface Designing

The power requirements vary depending on the voltage and current specifications of each component. An important task is to allocate adequate power to the components depending on the operating mode and to design the power interface to ensure that the power sources meet the power demands of components. For designing the interface, it is a good practice to revise the power budget table by considering the exact power demands of components. In addition to the required power, the voltage and current information of the components are also important in the design of the power interface. If the demanded voltage cannot be supplied by the satellite power source, the EPS subsystem needs to convert the voltage. Depending on the current demand of components, current resources in the power interface should be allocated. To manage the power interface, it is important to consider not only the steady-state currents but also the transient currents of each component. If one component requires a high transient current for a certain operation such as start-up, it may seriously affect other components using the same channel. To reduce very high transient currents, filters or slow-start circuits are introduced. Interface design is an important part in the design of the power distribution function of the EPS subsystem.

The energy supply for each component is controlled depending on the operational mode. For example, in the safe mode almost all components that are not essential to the satellite's survivability should be switched off, whereas mission equipment should be powered on during mission operations. Because transmitters of the communication subsystem tend to consume large amounts of power, the power to transmitters is enabled only during telemetry downlink. Such power on/off control is implemented along with the power distribution function.

4.3.2.3 Function Designing and Contingency Analysis

To start hardware design, it is quite important to define the functions that the EPS subsystem should perform. As discussed above, the power interface design includes the voltage conversion function and the power distribution function, which includes power on/off control function. It is a good idea to summarize and map interface conditions into requirements for the power distribution function of the EPS subsystem. Figure 4-22 shows and example of the EPS system diagram.

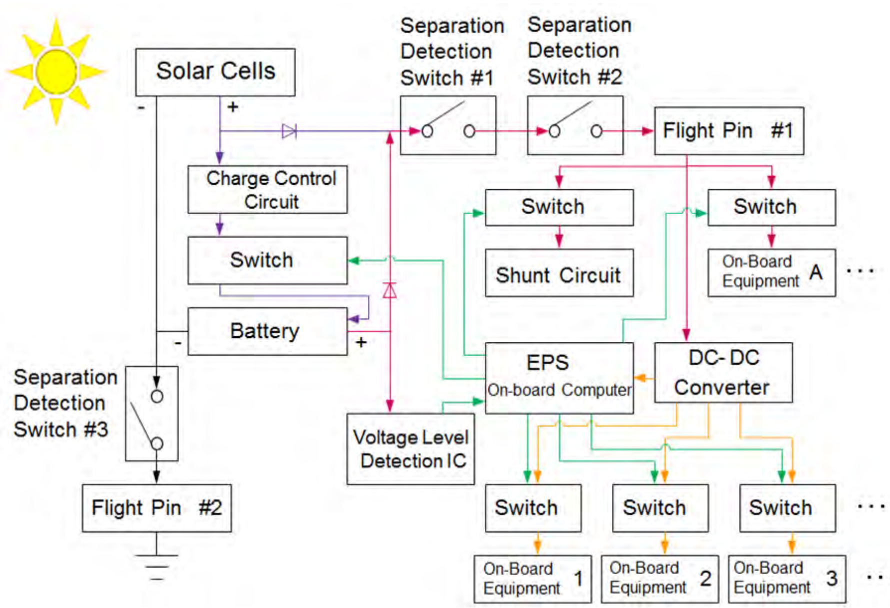


Figure 4-22: Example of EPS system diagram.

Refining the specifications of the power generation and storage functions based on the results of sizing is a basic task. Not only qualitative specifications, but also quantitative specifications such as the number and the voltage of solar cells, and the size and voltage of the batteries should be considered. For achieving basic power generation and storage functions, appropriate power generation and storage algorithms such as the Peak-Power Tracking (PPT) algorithm, battery charge methods, and overcharge inhibit strategies should be considered and adopted.

The power condition is very critical for satellites and conditions such as power generation levels of the solar panel and battery charge levels should be monitored regularly. The monitoring function is also defined in the functional design. The parameters to be monitored need to be defined and the frequency of monitoring should be set.

Because the anomalies in power conditions may critically affect satellite survivability, anomalies found in power conditions must be mitigated as soon as possible. Because the power subsystem monitors critical power-related parameters, the power subsystem can mitigate anomalies such as Under Voltage Control (UVC) without waiting for any command from the ground or C&DH subsystems. Therefore, the mitigation process of power condition anomalies is generally implemented in the EPS subsystem. Possible anomalies should be considered in contingency analysis, and the anomalies mitigation function should also be defined. Because the anomalies mitigation strategy is a very critical factor that affects satellite survivability, the contingency analysis and decision process should be shared with the entire satellite development team.

4.3.2.4 Hardware Design, Development, and Verification

Based on the specifications defined in the functional design process, the EPS subsystem equipment, including power generators (solar panel), power storage (battery), power control units (control and monitor of power generation and storage), and power distributing units (voltage conversion and distribution), is designed.

As for the hardware, the use of processor with software is generally avoided to ensure high reliability. Control using software is flexible and convenient but anomalies may be caused not only by software coding errors but also by hardware-related issues such as single-event effects. Hence, a hard-wired analog control system is preferred for use in the EPS subsystem.

As in the case of development, verification is also important for hardware implementation. Each component of the power interface must be verified because it is critical for the satellite's operation. By measuring transient currents using a current probe, it must be ensured that transient currents meet specifications because transient phenomena are highly dependent on interface conditions.

Another critical and important element to be developed by the EPS subsystem team is the separation switch. The design and development of the separation switch should ensure high reliability by considering relevant safety requirements as well as development aspects of the satellite. The separation switch includes three redundant switches to meet the stringent safety requirements, and its development is critical.

The reliability of the separation switch must be guaranteed not only by analysis, but also by experimental demonstration. Such experimental demonstration will be performed as part of vibration tests; therefore, the EPS subsystem team needs to develop a verification tool that proves that the separation switch never turns on during the satellite launching process.

4.3.3 What Can Be Learned About EPS Subsystem from CanSat

Even though a CanSat does not have power generation and storage functions because its mission duration is quite short, designing the CanSat includes the basic EPS subsystem design process and it can be said that CanSat is a good tool to learn the basic design of the EPS subsystem.

In the design of CanSat, sizing of battery is the most critical process that determines the mission's success and the capability of the CanSat. Thus, CanSat design serves as a good practice for the sizing process in satellite design. Sizing of battery involves summarizing the estimation of demand and determining the necessary and sufficient battery size. If the CanSat includes a few components such as a sensor, actuator, and communication unit, power interface design and power on-off control will be required to utilize the components effectively. The CanSat is expected to turn off its power before launch, and reliably turn it on right after separation. Therefore, the CanSat design process helps in learning how to design a separation switch and how to achieve high reliability. Even if the mission duration of the CanSat is too short to test the anomaly mitigation process, it is quite important to specify various possible anomalies during development.

As shown above, the CanSat is a very good tool that helps designers to learn the basic design and development process of a satellite, especially that of its EPS subsystem. The CanSat can be more effectively utilized by understanding how the processes related to the CanSat correspond to that in a satellite.

4.4 Communication Subsystem Design

The communication subsystem of a satellite receives commands uplinked from the ground station and downlinks house-keeping and mission data to the ground station. Command data is sent to change satellite operating status, on/off switching of on-board equipment and for selection of satellite operational modes such as power saving, attitude controlling, data downlink, etc. House-keeping data is information about the satellite health status, such as, equipment temperature, battery voltage, and whether parts are functioning or not, etc. This housekeeping data is used to make sure everything is working properly.

In general, the communication subsystem consists of two segments, as shown in Figure 4-23. The receiving segment consists of receiving module with receiving antenna, receiver, modem, and decoding equipment such as microcomputer. The transmission segment consists of transmission module with transmitting antenna, transmitter, modem, and encoding equipment.

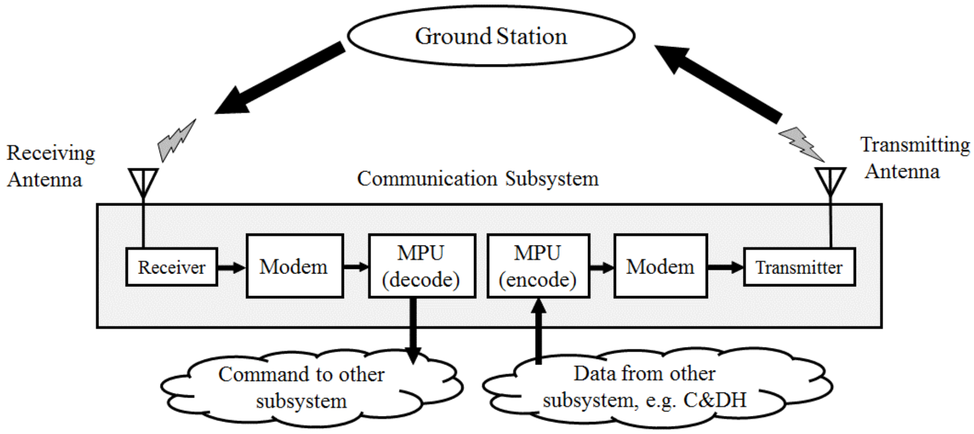


Figure 4-23: Concept Diagram of Communication Subsystem of Satellite.

Understanding the function of these components requires a basic knowledge of communication and antenna engineering. It is recommended that the reader reads an introduction about a radio and antenna engineering. In this section, the minimum information that the reader should know about the communication subsystem of satellite will be presented.

4.4.1 **Basics of Wireless Communication of CanSat**

Typical communications between the CanSat and the ground station are uplink of commands and downlink of house-keeping and mission data. Commands uplinked from the ground station are received via the receiving antenna as analog signals, digitized by the modem, and then forwarded to the microcomputer. The microcomputer interprets the command (decoding) to identify the data transmitted from the right ground station which enables it to process tasks or relay commands to the rest of the subsystems.

When it functions to downlink data, it receives digital data from the C&DH subsystem and other subsystems, and then encodes it in accordance with a pre-specified protocol. This signal is relayed to the modem for modulation before the data is converted into a radio signal for transmission via the transmitting antenna. Thus, communications between CanSat and the ground station, such as command uplink to CanSat and data downlink to the ground station, will be enabled in accordance with pre-determined protocol and radio signal carriers.

4.4.2 **Communication Protocol**

what is a protocol? It is simply defined as “accepted communication rules”. The simplest protocol applicable for satellite communication is the Morse protocol or Morse signal communication. However, it needs much time to transmit radio signals which are coded with variably time-dependent data frames. Thus, it is difficult to transmit a large data stream. However, in comparison with the packet data communication protocol for transmitting a unit of data frame, the Morse protocol has advantages. Firstly, in case transmission is temporarily disrupted during data communication due to difficulties such as interference caused by illegal radio signals and noise-rich radio signal, only the data during the disrupted time frame will be lost but the rest of the data stream can be correctly received. Secondly, it enables data communication with less power consumption, and superiorly even with narrow band. Low power consumption is a good advantage, and therefore Morse signal communication is commonly applied for small satellites, except for specific missions.

Packet communication, which transmits data by frames, is much more efficient than Morse signal transmission. Figure 4-24 illustrates an example of this communication protocol. It is structured as a unit frame of packet data by start code and end code, which identifies the beginning and end of the frame respectively. The data frame contains segments such as frame length, data, and check-sum. Check-sum is an error checking code which is a methodology to detect error incidence in the total integrated data to identify receiving all data without errors.

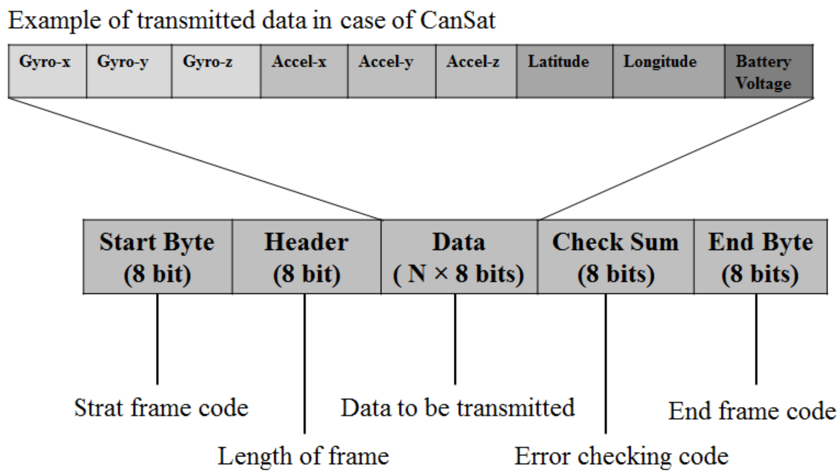


Figure 4-24: Example of Packet Communication Protocol.

In radio communication, the unit of communication rate is bps (bits per second), which shows the data amount transmitted per second. However, a large amount of data in a frame may cause loss of the whole packet data in case data communication is disrupted due to unfavorable radio jamming and interference. The communication rate is subject to a condition fixed to a certain packet data size (D) and its number of transmission per second (N), and beside that condition, the actual data rate may be lower than that in the free packet size. Therefore, in designing a communication subsystem, just keep the specification as reference for the data rate of the radio device, but importantly set the packet size of data transmitted (N×D), and then verify the rate with the real performance of the communication subsystem under development.

4.4.3 **Communication Devices (Transceiver, Modem, Antenna)**

Communication between satellite and the ground station uses radio signals. Radio signals have a unique frequency, which is strictly regulated by governmental laws for every utilization of radio signal, for example, the frequency band 3 kHz – 30 kHz used for communication for earth observation satellite and the band 300 MHz – 3 GHz used for mobile phone communication. There are provisions for licensing radio in accordance with certain power strength (Watt, unit of power). It is also noted that the government legally allows some radio frequency bands for free use without a radio license.

In summary, radio equipment for satellite communication must comply with the governmental regulations in the country where it is used. “Specified low-power radio” would be the easiest for CanSat radio communication devices. One of the well-known products in the specified low-power radio device is XBee communication modules made by Digi International Inc. and shown in Figure 4-25. Xbee is a very useful all-in-one single unit with multi-functions of receiving/transmitting modem for radio modulation/demodulation, receiver/transmitter and each antenna can perform dual communication. Though the communication distance is not long and the radio signal is not powerful enough, the product is a useful radio equipment for training to learn the CanSat system.

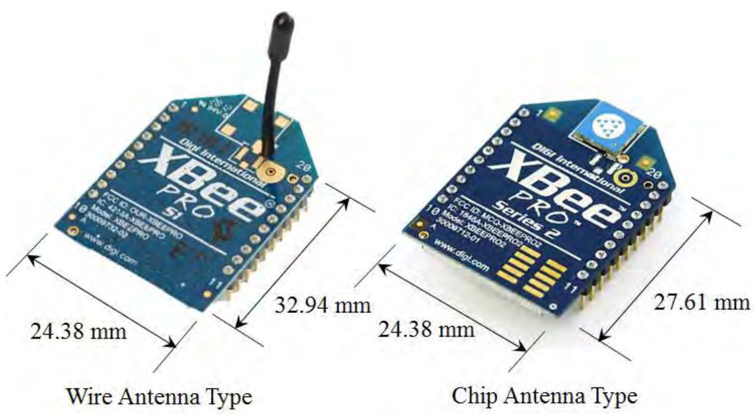


Figure 4-25: Example of Antenna.

The knowledge of radio equipment could be necessarily somewhat expanded into not only frequency and communication protocol, but also modulation/demodulation. Except for unique protocol like Continuous Wave (CW) communication, in general, a carrier radio wave, the frequency of which represents the frequency of radio equipment, will carry signals, which will transmit the data. The function of superimposing data signals to relay onto a carrier wave is defined as modulation. Conversely, when receiving radio signals, the signals featuring data carried by modulated radio will be extracted, and this is defined as demodulation. The modem (modulator-demodulator) is a device that modulates transmitted signals and demodulates received signals. Figure 4-26 is the schematic diagram of the carrier wave and data signal during the modulation and demodulation.

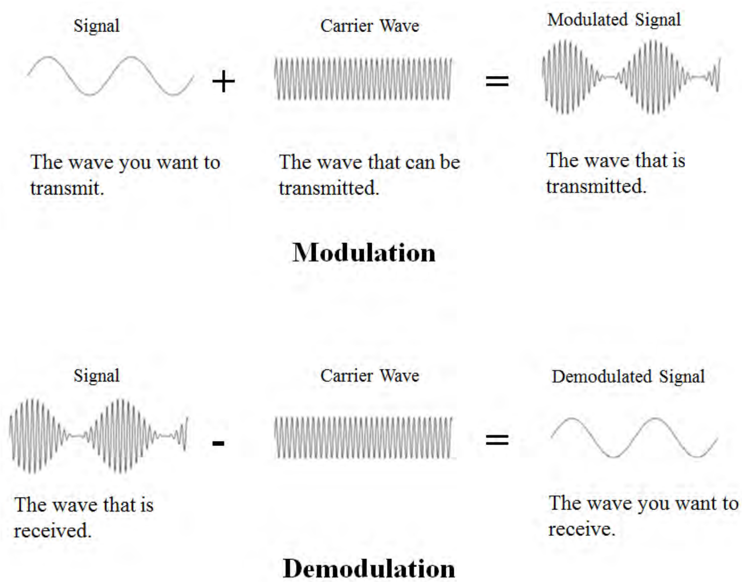


Figure 4-26: Amplitude Modulation (AM) and Demodulation Diagram.

There are variations in the modulation and demodulation scheme. Amplitude Modulation (AM) or Frequency Modulation (FM) are ways of broadcasting radio signals. Both transmit the information in the form of electromagnetic waves. AM works by modulating (varying) the amplitude of the signal or the carrier to be transmitted according to the information being sent, while the frequency remains constant, as shown in Figure 4-27. This differs from FM technology in which information is modulated by varying the frequency of the wave and the amplitude is kept constant Figure 4-27.

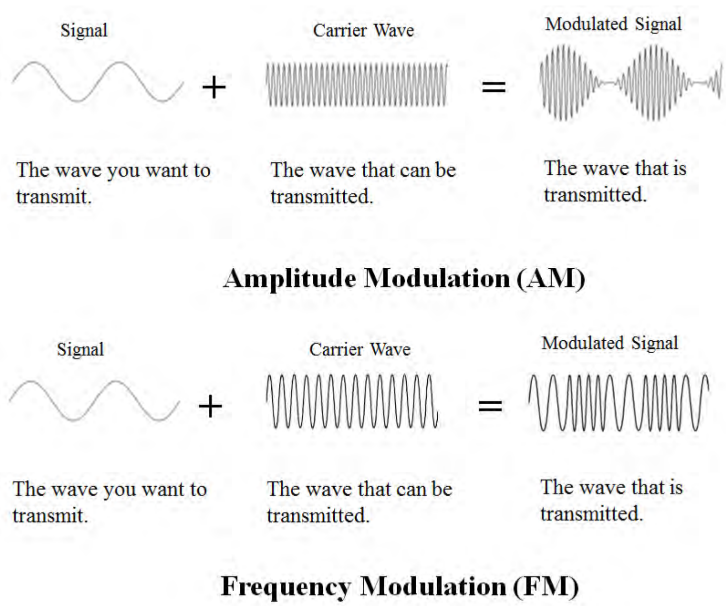


Figure 4-27: Amplitude modulation and frequency modulation.

Digital data is also conducted by modulation and demodulation. Figure 4-28 is the schematic diagram of frequency modulation of digital data.

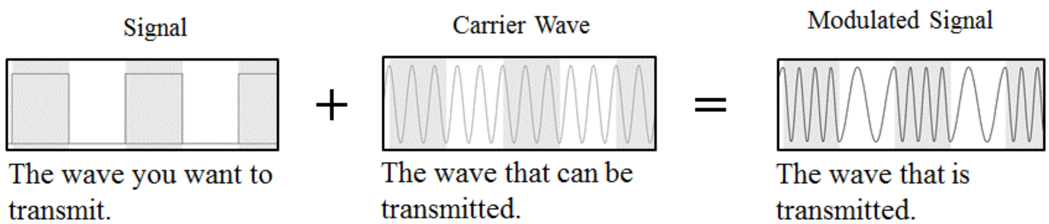


Figure 4-28: Frequency modulation of digital data.

As seen above, a modem functions as a modulator/demodulator using the features such as amplitude, frequency, and phase of a carrier wave. When receiving, a modulated radio wave is sent to the modem to be demodulated and the data will be the output of the modem. The data will be transmitted to the decoding and processing unit which is microcomputer. The opposite happens when transmitting. Since the decoding/encoding method differs by modem types, data sheet of the modem needs to be checked, and design the decoding/encoding processing unit or develop a program for the microcomputer in coordination with the modem.

Antenna is needed to communicate by radio. The antenna is a device which amplifies the radio wave. A coated coaxial cable is robust over noise and widely used to connect the radio transceiver and antenna. For actual use in a satellite such as CanSat, there are several types of antennas available. The Antenna types are monopole, dipole, parabolic, and patch antennas. They can to be specified and procured or more precisely designed and developed. Since expert knowledge is necessary to deals with radio waves, legal procedures should be followed such as getting a radio communication license. In CanSat, the legal aspect is somehow relaxed by using the commercial available XBEE communication module which has an embedded antenna and it doesn't need license for operation. Figure 4-25 shows the XBEE communication module with the commonly used antennas.

4.4.4 **Radio Link Design**

During the design of satellites, a series of tasks is mandatory for the development of hardware and software. These tasks for communication subsystem design include the power budget analysis when the radio signal from the transmitter is attenuated during travelling through air and reaches the receiver, circuit trade-off design to verify the feasibility of end-to-end communication line, and set up design parameters for the feasible design like the output voltage of transmitter, communication data rate, antenna gain, etc.

In the radio link design, the C/No ratio between the carrier power, C, and noise power density, No, when it is received at the receiver side. The determined C/No ratio is for assessing whether it is more than the theoretical required value. Often it is required that the link provides more than 2 times the theoretical value. In fact, the larger the amount of C/No the better the communication. Since C/No is originally the proportion of needed signal to the noise, a signal amplified enough not to be lost due to noise would be sufficient. Usually in wireless communication, as the distance between the transmitter and receiver increases, the noise increases. Generally, the electric power of the radio wave output from the transmitter goes through different amplification and attenuation processes, as shown in Table 4-2. The signal finally reaches the decoding and processing equipment.

Table 4-2: The amplification and attenuation processes of the electric power of the radio wave.

(a) Transmitter ⇒ Transmitting antenna (power consumption at wiring port)
(b) Transmitting antenna (antenna gain, pointing loss (the difference between transmitting antenna direction and receiving antenna direction)
(c) Transmitting antenna ⇒ Receiving antenna (free-space loss, atmospheric absorption loss, polarization coupling loss, rain loss)
(d) Receiving antenna (antenna gain, pointing loss (the difference between receiving antenna direction and transmitting antenna direction)
(e) Receiving antenna ⇒ Receiving station (power consumption at wiring port)
(f) Receiving station processing (receiving signal processing loss)

In the communication between CanSat and the ground station, C/No is large based on the calculation of these gain and loss factors.

The theoretical required value of C/No should be specified. To specify the required value, determination of the losses when demodulating the received signal and the internal loss of the receiving station, L, should be conducted. Then set up the Eb/No, which is the ratio of signal energy per bit, Eb, to noise power density, No, per unit interval of frequency, B, by setting the tolerable error rate of received data. The sum of communication rate, R, and B will decide the ideal C/No. By tacking on the loss, L, to this C/No, the required value is specified. Finally, operability of the line is established by setting a margin from those values.

In the case of CanSat, it is not necessary to design the communication link so rigorously. In fact, except for the loss between transmitting antenna and receiving antenna, as shown in (c) row of Table 4-2, most of other losses and amplifications will not be clear until they are measured, and the measurements are difficult. Therefore, the best way to design the communication link is to read the transmitter's output power, the gain of transmitting/receiving antenna from each data sheet, only calculate the loss between transmitting and receiving antenna, and work out with the margin. Buy a set of transmitter and antenna and check the possible transmitting distance from the data sheet. Set a possible communication distance with appropriate margin, and make a mission possible within that distance. If the assumed communication distance is verified by testing, it is acceptable.

4.5 Use of Sensors

A sensor is an apparatus converting a change occurring in natural or artificial environment, such as light, temperature, pressure, speed, acceleration, angular rate, electric voltage or electric current, into a comprehensive signal by physical law or through a chemical process. The role of sensors in a satellite is to detect the satellite's electric voltage, electric current, temperature and attitude. These are the essential parameters for the satellite to survive autonomously in space. Moreover, a sensor itself can be a special mission apparatus in the case of astronomical observation, weather observation and earth observation. Sensor's function and composition for CanSat are presented and explained in this section.

4.5.1 **Function and Composition of Sensors**

The role of sensors within CanSat, mostly the same as in a satellite, is to detect CanSat's status such as electric voltage, temperature of main devices, and attitude. In some missions, a sensor also acts as a mission payload such as a camera. The measurement by the sensor proceeds as follows: 1) Conversion of the change in the parameter to be measured to electrical signal, 2) Amplifying and filtering the electrical signal as necessary, digitalize by A/D (Analog/Digital) conversion, and 3) OBC (On-Board Computer) reads the digitized data and stores or sends downlink to the ground station. The block diagram of the sensor measurement process is shown Figure 4-29.

Figure 4-29: The block diagram of the sensor measurement process.

As shown in Figure 4-29, the sensor doesn't operate alone. It generally works with an amplifier to amplify its small output voltage, a filter to eliminate noise superimposed on the output, A/D converter to change the analog input quantity into a digital output quantity, OBC controlling data, and a memory to storing the data. Sensor data is not only stored but also used to control an actuator or downlinked to the ground station.

To perform functional capabilities as a sensor system of the satellite, it is required not only specification requirements (instrumental range and precision) of the sensor itself, but also specification requirements for data memory (size and read/write access speed), communication subsystem to downlink data to ground station (downlink data size and format), and positioning precision of the structure holding the sensor to a specific orientation. Thus, sensor design must be coordinated with other subsystems subject to a total balance of management and control for the whole satellite system.

4.5.2 **Types of Sensors**

Sensors are roughly categorized into three types: 1) sensors measuring a reference object or quantity, example of this category are solar sensor which determines the sun direction by measuring the incident direction of sunlight, and magnetic sensor which measures the three-components vector of the Earth's magnetic field, 2) sensors measuring its own operating condition, example of this category are temperature sensor, electric voltage/current sensor, gyro sensor measuring angular rate or acceleration, and acceleration sensor, and 3) sensors detecting its own location, example of this category are GPS sensor which detects location or ultrasonic sensor which can track the positional relationship between objects.

Consider the fly-back CanSat mission profile shown in Figure 4-30. The combination of sensors enables a lot of operations such as an acceleration sensor to measure how much load the CanSat is subjected to at the moment of launching, a GPS and barometric sensor to measure the altitude and for release height control, GPS or magnetic sensor for direction control, and an infrared or ultrasonic sensor for maneuvering around obstacles.

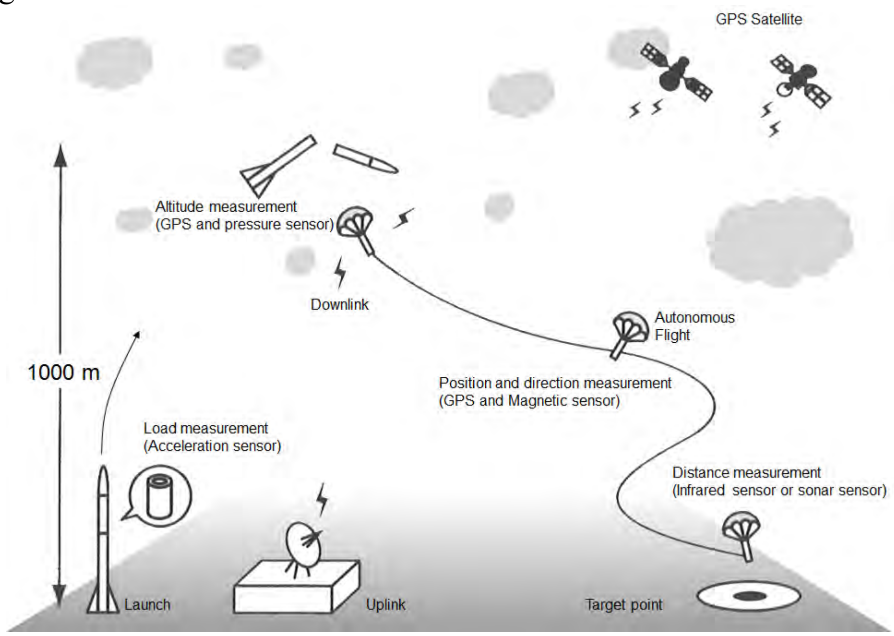


Figure 4-30: Mission profile of a fly-back CanSat.

4.5.3 **Digital and Analog Sensors**

Sensors can be divided into analog or digital types according to their signal format. The measured value must be converted to digital for the OBC readouts. If the sensor outputs a digital signal as shown on the right of Figure 4-31, it communicates directly with the OBC as shown in Figure 4-32. When the sensor outputs an analog signal as shown on the left of Figure 4-31, the signal must be converted to digital by the Analog to Digital (A/D) converter, and then communicates with the OBC as shown in Figure 4-33. An operational amplifier or filtering circuit may also be needed when the sensor output is too small or if sensor output noise need to be removed, respectively. There are sensor types packaged with amplifying circuit, filter and A/D converter, while some sensors are digital, detecting 1 or 0 as switching on/off.

In the analog signal, several deteriorations such as noise contamination tend to occur. Even a sensor which outputs digital signals with an A/D converter packaged needs extra care because it contains analog components inside. For instance, noise on the sensor should be observed when a motor or radio communication device is located near the sensor. Digital sensor only detects data exceeding a certain threshold, while analog sensor measures even a very small voltage change, data once converted to digital is relatively resistant to noise.

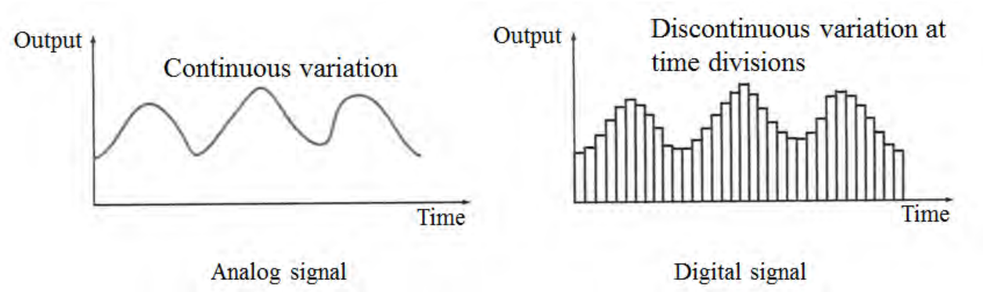


Figure 4-31: Analog and digital signals.

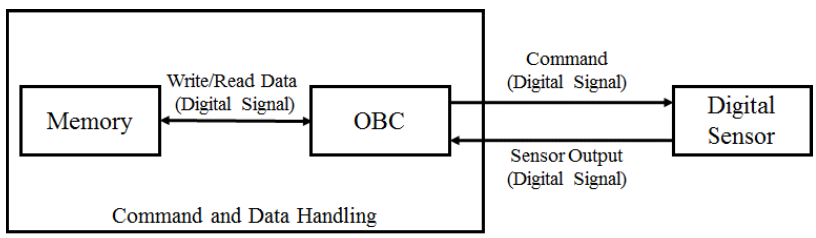


Figure 4-32: Configuration of digital sensor and OBC.

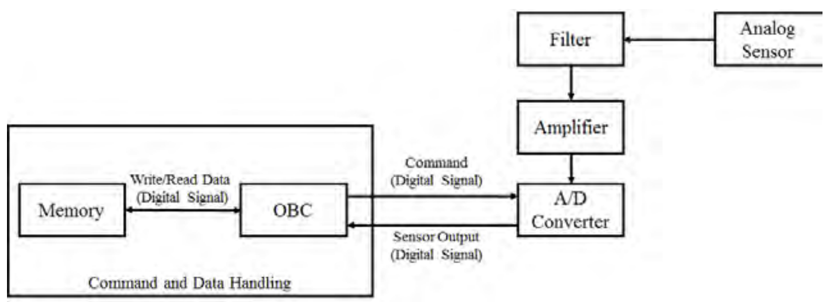


Figure 4-33: Configuration of analog sensor and OBC.

4.5.4 **Sensor Measuring Range and Accuracy**

As each sensor has a predetermined measuring range, a sensor whose maximum measured value on CanSat will not exceed this range should be selected. Sensor type, range, sensitivity, maximum sampling frequency, communication protocol, etc. are written in the sensor's specification sheet. The sensor's sensitivity is defined as the amount of physical quantity change is to be detected as a voltage shift by the sensor. For example, consider the gyro sensor for angular rate measurement. If its measuring range is from 0 - 100 deg/s, sensitivity is 20 mV/(deg/s), and the zero point (gyro sensor output at rest condition) is 2.5 V, the sensor output signal should range 2.5± 2 V and the output at 0 deg/s would be 2.5 V. A unit change in the angular rate results in 0.02-volt change. However, it should be kept in mind that even if the sensor is highly sensitive, maximum utilization of sensor sensitivity cannot be made without enough resolution in the A/D conversion.

The A/D convertor, which converts an analog signal to digital, divides the analog quantity in a phased manner. How small the value can be detected is expressed as resolution performance of the A/D converter and is measured in bits. Each bit will be displayed in only two values, 0 or 1, whichever applies. In a 10 bit A/D convertor, analog electric voltage sent from the sensor is expressed as a 10 character string of “0” and “1”. Therefore, if the electric voltage accepted by the A/D converter ranges from 0-5 V, 0 V would be shown as “0000000000” and 5 V as “1111111111”. 3 V would be expressed as “1001100110”. That is, the range from 0 to 5 V would be described in 10 bits, or 2^{10}=1024 steps. So if the input is scaled to 5 V, the minimum voltage change detectable by the 10-bit A/D convertor would be 5 V/ 1023 = 4.89 mV.

There are types of OBCs which have built-in A/D convertors, and there are types that need to install externally A/D converters to achieve the desired function. For the OBC to read the sensor value correctly, the sensor’s output voltage has to be within the OBC’s operating voltage range, and the output signal voltage should be amplified when the sensor’s output range is too small. When the A/D convertor converts 0-5 V to 10 bit digital signal for example, its resolution capability is not exercised completely if the sensor’s output range is much smaller than 0-5 V. In such situations, the solution is to amplify the sensor’s output up to 0-5 V and utilize the full resolution performance. An operational amplifier circuit is generally used to amplify the sensor’s output. The operational amplifier can amplify voltage, current, or power, as well as perform addition, subtraction, integration and differentiation.

It should be kept in mind that the sensor data may differ from its real value because of several noises, such as noise emitted by the fluctuation caused by launching or separation, noise of the sensor itself, or noise from communication device or actuator. These noises are added to the real data. Figure 4-34 shows an example of a communication device located near a magnetic sensor. It is clearly shown that there are noises added as high-frequency waves when the communication device is on. When allocating equipment such as a communication device or actuator within a confined space like CanSat, it is important to consider the layout and operating manner for the effect they may have on the sensor measurements. For example, when measuring the sensor, beware not to use the communication device or actuator.

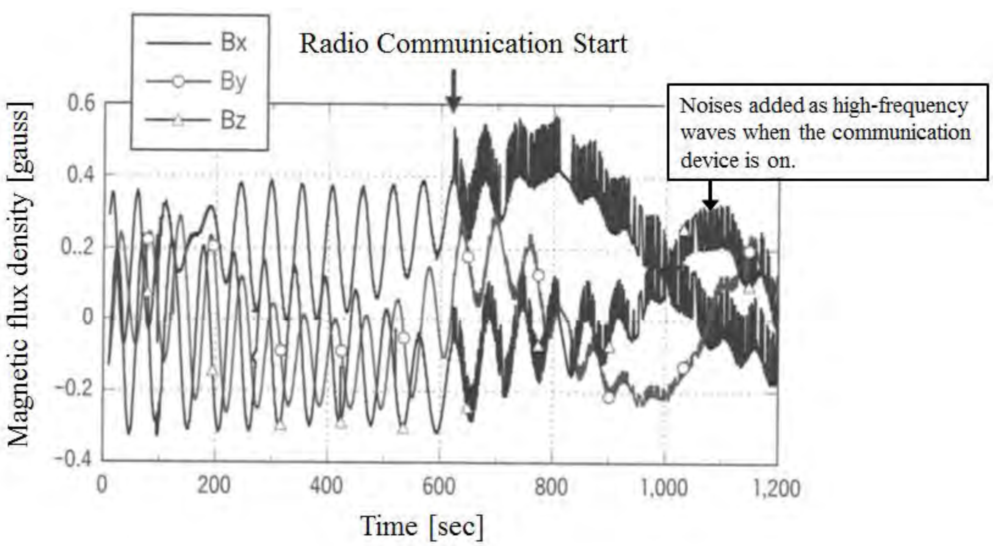


Figure 4-34: Interference between magnetic sensor and communication device.

Noise can be suppressed with filtering devices. Analog sensor signals go through filtering circuits. digital signal processing is performed for digital signals just after A/D conversion. Low-pass filters pass low-frequency signals and attenuate (reduce the amplitude of) signals with frequencies higher than the cut-off frequency. A high-pass filter is the opposite of a low-pass filter. Careful selection of those filters is needed to comply with the characteristics of sensor data and noise.

Figure 4-35 shows the experimental data for each angular rate, when three gyros attached to three axes are rotated around the x-axis at a constant angular rate. Only one gyro is supposed to detect the angular rate, indicated by increased output voltage, while the other two outputs provide non-rotating values. The left graph in Figure 4-35 is the data before low-pass filtering, and the right shows the data after filtering. It is clearly shown that that the high-frequency noise decreases when the low-pass filter is used.

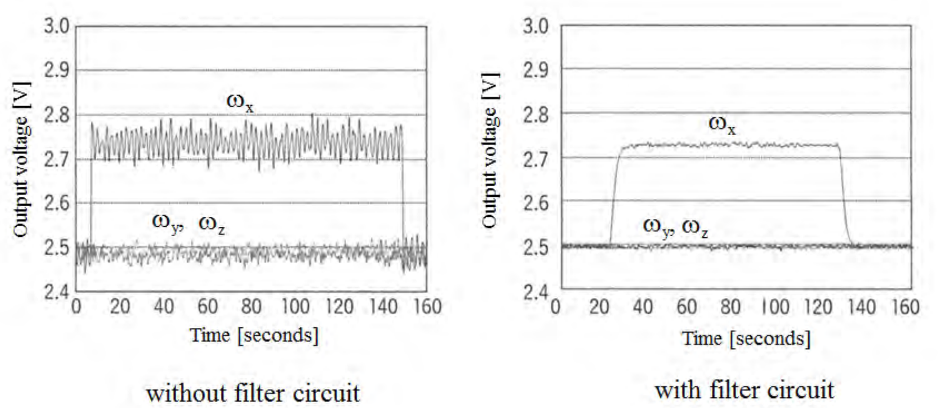


Figure 4-35: Effect of low-pass filter.

4.5.5 **Communication with the Sensor**

The sensor data converted to digital output is transmitted to the OBC. There is a serial or parallel communication in the sensor signal path. In serial communication, data is transmitted/received continuously 1 bit at a time, as shown in the left of Figure 4-36. Since there are few data lines, the OBC can be used efficiently. Some serial communication interface needs a clock line which provides data transmitting timing. In parallel communication, each bit of data is connected with its own signal line, as shown in the right of Figure 4-36. In this type of communication, a one-character signal can be transmitted/received at the same time.

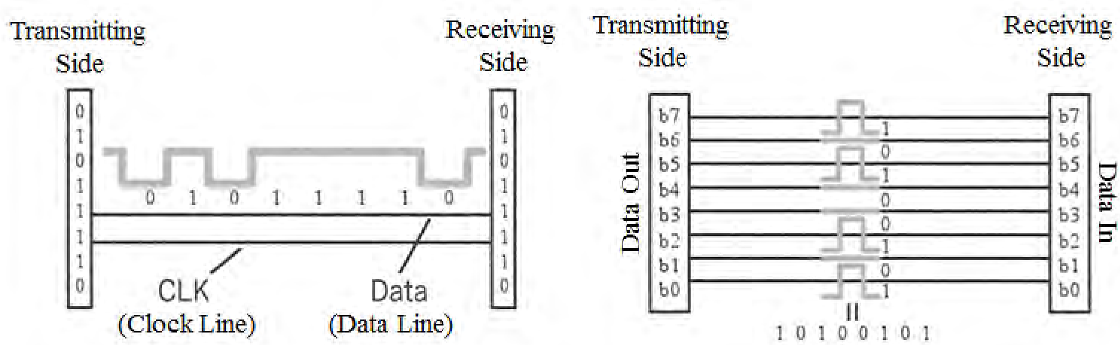


Figure 4-36: Data communication, Left: serial communication, and Right: parallel communication.

4.5.5.1 Sampling Frequency and Amount of Data

The OBC acquires the sensor data converted to digital signals at a constant sampling frequency. This sampling frequency is determined according to the mission that need to be conducted. OBC's processing time is an important factor to achieve the intended sampling frequency. For example, the sampling frequency can be improved by reducing the processing time of OBC for acquiring sensor data, by cutting down the time for A/D conversion, time to write into the memory, and time to send data to the communication or control subsystem. In this regard, the Shannon sampling theorem provides the criteria for reproducing the original signal by the sampling interval. The theorem stated that the sampling frequency must be more than twice the maximum frequency of the original signal's frequency content, in order to reproduce the frequency content of the original signal correctly.

Since the sampling frequency represents the intervals in acquiring sensor data, there is a close relationship between the sampling frequency and amount of data. In case the sensor data is stored in the memory, the memory size must be selected based on the total time necessary in CanSat for realizing the sampling frequency and acquiring the sensor data.

4.5.6 Sensor Selection and Configuration Example

When developing a CanSat, it is important to consider the whole mission and all possible events that occur during the mission, how to realize the mission, and how the systems such as data communication and sensors interact with each other. The sensor configuration in CanSat is like that of real satellite. So, the sensor system development for the CanSat can be applied to real satellite sensor system development. The following section explains the sensors system development for CanSat using a real example of a mission profile.

Selection of the sensors starts from making clear the sensor's requirements from the predetermined specifications such as weight or size, past testing data (vibration, acceleration, temperature and wind), maximum flight time and assumed mission profile. Consider the necessary sensors that can realize the fly-back CanSat mission example shown in Figure 4-37.

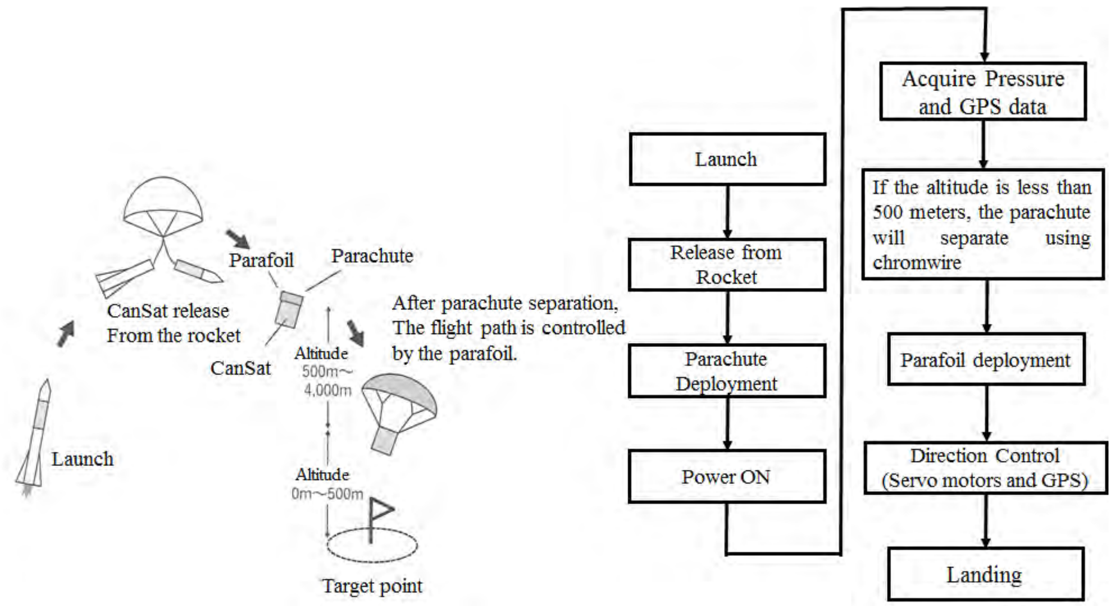


Figure 4-37: Example of a fly-back CanSat mission profile.

The mission profile proceeds as follows: 1) Install CanSat into the rocket, 2) Complete the rocket preparation and assume a few minutes to an hour of standby time, 3) Launch of rocket and separation of CanSat (CanSat is exposed to vibration and high acceleration during the launching and separation), 4) CanSat's mission begins upon detecting separation from the rocket, 5) Acquire, save and downlink of mission data, and finally 6) Landing. In line with this procedure, the sensors and their characteristics (accuracy, sampling frequency, environmental characteristics, transit characteristics, power consumption, data format, size, weight, operating voltage, etc.) are selected based on their availability and simplicity to install into the CanSat. In this mission, after separation from the rocket, CanSat falls by parachute from 4000 m to 500 m in order to avoid the wind effect. When the altitude becomes less than 500 m, it separates from the parachute and deploys the parafoil. Then the CanSat lands to the ground while controlling the parafoil. The main roles of the sensors are to detect of parachute separation and parafoil deployment altitude, and detection of CanSat's location information to control the parafoil, as shown Figure 4-37. For example, these detections are possible by using a barometric sensor for altitude detection and GPS for location detection.

Sensors are selected based on their offset/gain features, repeatability of sensor data, transit characteristics, environmental properties, relations between sensor output and measured quantity, data formats, and resolution performance. However, stand-alone evaluation tests should be conducted by the developer, because these features often differ from the standard values shown in the specification sheets. Figure 4-38 is an example of a CanSat system diagram which realizes the mission profile in Figure 4-37.

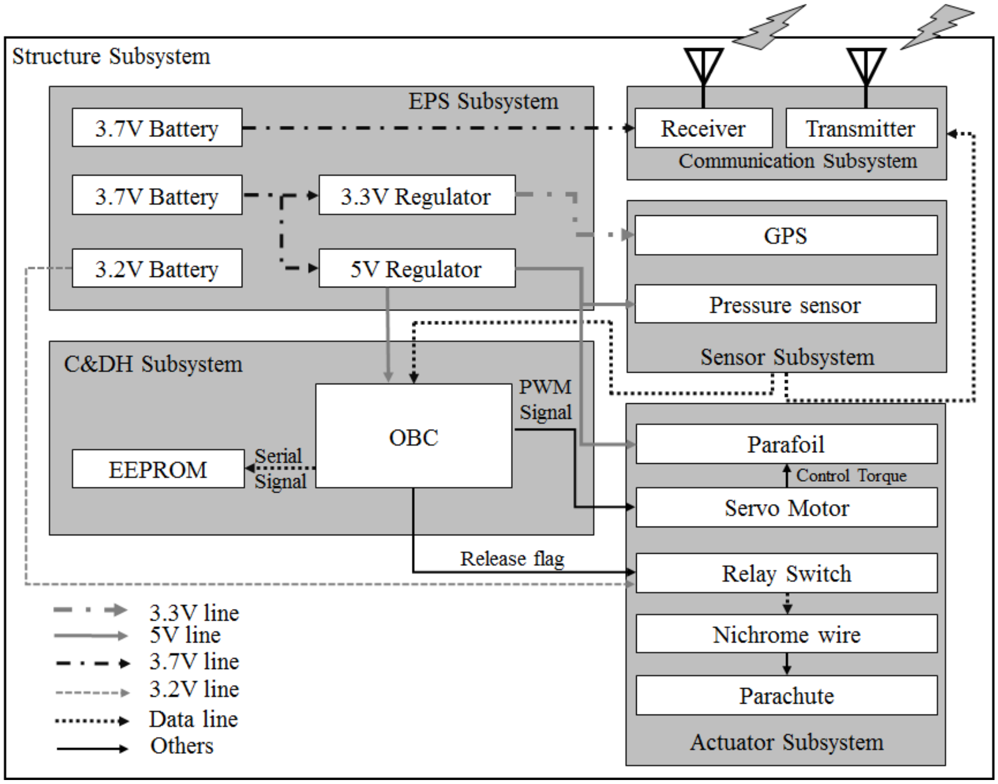


Figure 4-38: Example of fly-back CanSat system diagram.

When checking stand-alone operation, performance guarantee by several tests should be conducted (electric power test, interference test, performance test, communication test, temperature test, vibration test, calibration test, etc.). Because experiments under actual conditions cannot be conducted, it is important to construct a reliable system based on the results of several tests and experiments under restricted circumstances.

4.6 Use of Actuators

An actuator is a device which converts input electrical energy into another form of energy like mechanical motion. The main function of an actuator within a satellite is to position a deployable structure such as a communication antenna or solar array panel, and to apply force or torque for orbit and attitude control. The actuator is not only for basic device deployment but also plays an important role as a mission unit that enables the satellite to operate autonomously in space.

In this section, an example of a CanSat system that contains an actuator and how it functions is presented and explained. In addition, an example of verification method is described for the case when combining the actuator with other subsystems

4.6.1 **Function of Actuator**

Actuators are used in various situations of the CanSat mission. A fly-back CanSat mission profile can be one example, as shown in Figure 4-30. A parafoil control using GPS and servo motor can be considered as a directional control to a destination. If it is required to switch to a rover type system which moves to the destination after landing, another system need to be developed. For example, a system is needed to measure the distance from CanSat to the ground by ultrasonic sensor, and thermally cuts the wires between parachute and CanSat using a heated Nichrome wire, when the distance becomes below a certain threshold. This distance or altitude must be set where CanSat doesn't break down after parachute separation and the free fall. After landing, the GPS receiver data is needed and a DC motor to rotate the wheels. Figure 4-39 depicts the block diagram of CanSat's directional control using actuator. The OBC receives data from the sensor, and sends a control signal to the actuator using "direction control logic". As presented above, a CanSat's actuator doesn't run alone but is closely related to other subsystems such as the OBC and sensor.

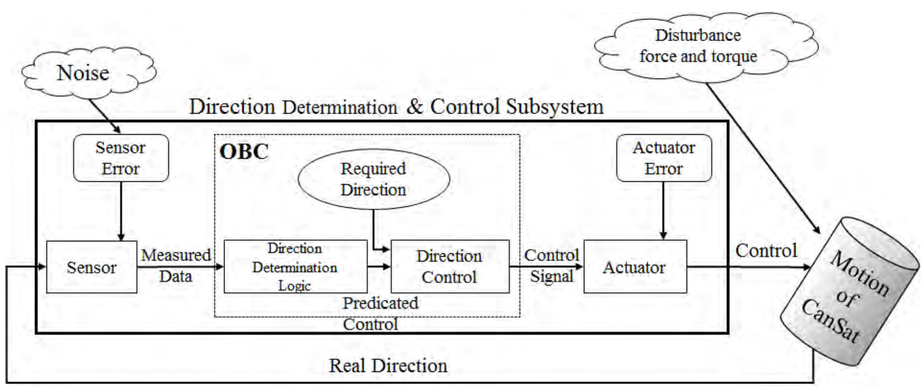


Figure 4-39: Block diagram of CanSat's directional control using actuator.

4.6.2 **Types of Actuators and Control Method**

In this subsection, a brief explanation about the types of actuators and control method used in CanSat are presented. During the selection of an actuator, it is extremely important to carefully consider the weight, size, power consumption, operating voltage and performance (such as transit characteristics, output power, and so on), whether they are sufficient to employ in a CanSat to accomplish the mission. In an initial phase of CanSat development, evaluation of basic specifications is needed using specification sheets and unit testing. In a later phase, actuator response to CanSat's own conditions due to the interference among subsystems need to be carefully tested. For instance, a servomotor which draws a parafoil control line to control flight trajectory and change CanSat's path, or DC motor generating torque for wheel drive, works as an actuator, as shown in Figure 4-40.

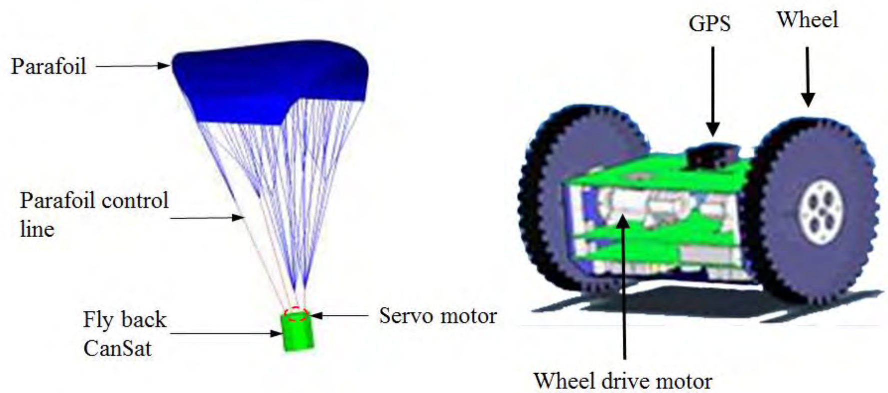


Figure 4-40: Fly-back CanSat and rover-back CanSat.

A servomotor can control the angular position to an arbitrary angle between 0° and 180° according to voltage input, while a DC motor controls rotation speed according to voltage input. Control of the input voltage is necessary to activate the servomotor or DC motor. In CanSat, input voltage is controlled by a technique called Pulse Width Modulation (PWM). As shown in Figure 4-41, the input voltage can be controlled by changing the pulse width. Changing the input voltage to a servomotor causes a change in the angular position of the servomotor's shaft, as shown in Figure 4-41.

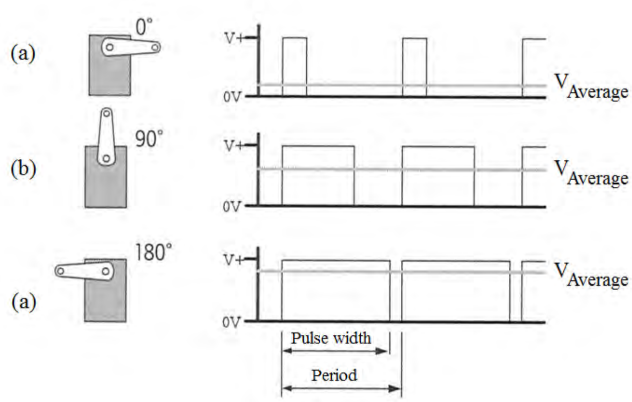


Figure 4-41: Operation example of servo and Pulse Width Modulation (PWM).

As shown in Figure 4-41, If P_T is the pulse period, H_T is the pulse width, and $L_T = P_T - H_T$. the duty ratio, D, can be calculated as follows:

$$D = \frac{H_T}{P_T} = \frac{H_T}{H_T + L_T} \qquad \text{Equation 4-1}$$

The duty ratio can take the value of 1 when $H_T = P_T$ which means the voltage is V+ during the whole pulse period. The duty ratio takes the value of 0 when $H_T = 0$ or $L_T = P_T$ which means the voltage is 0V during the whole pulse period. To control the motor by PWM, the OBC must determine this duty ratio. A servo can maintain the duty ratio neutral as shown Figure 4-41(b), or control the angular position to ±90^o turns as shown in Figure 4-41 (a) and (c). Some OBC may have a built-in PWM ports, but even if there are no such ports, PWM can be realized using OBC by generating pulse wave through repeating high and low using a delay function.

Figure 4-42 is an example of a servomotor control program with a pulse period P_T of 18000 μsec, using a PIC16F877 microcomputer. The servomotor is controlled by PWM whose H_T is 1500 μsec, from Pin number B3. In line 11 of Figure 4-42, B3 PIN is set to output V+. In line 12, H_T is set by delay function. In line 13, B3 PIN is set to output 0V. In line 14, L_T is set by delay function.

```
1. #include<16f877.h>
2. #fuses HS, NOWDT, NOPROTECT, PUT, BROWNOUT, NOLVP
3. #use delay (CLOCK = 10000000)
4. void main()
5. {
6.          int i;
7.          while(1)
8.          {
9.                   for(i=0;i<10;i++)
10.                  {
11.                           output_high(pin_b3);
12.                           delay_us(1500);
13.                           output_low(pin_b3);
14.                           delay_us(16500);
15.                  }
16.         }
17. }
```

Figure 4-42: Example of a program to control a servomotor by PWM.

The motor's switching control or angular rate control is possible by PWM. Since wheel drive control by DC motor must generate positive rotation to advance as well as negative rotation to move backward, an IC called motor driver is commonly needed. The motor driver IC has a built-in negative/positive changeover circuit. So, combined with PWM, it can produce clockwise rotation, counterclockwise rotation, stop, and brake. As listed in Table 4-3, a general motor driver has two input pins (IN1, IN2) to select the mode, two output pins (OUT1, OUT2) to generate the motor rotation, and a pin to control voltage supply to the motor. The OBC controls IN1, IN2 and the voltage supply pin, while OUT1, OUT2 realizes the needed control by connection to the motor

Table 4-3: Example of a motor driver's input-output setting modes.

IN1	IN2	OUT1	OUT2	Mode
0	0	OFF	OFF	STOP
1	0	H	L	Clockwise
0	1	L	H	Counterclockwise
1	1	L	L	Brake

Some CanSats need a deployment system, such as parafoil, after the altitude is below a certain threshold. For this purpose, Nichrome wire is an extremely simple actuator as a deployment actuator. It can be used to thermally cut a Nylon cord, which has restrained the deployable structure. Figure 4-43 shows an example of a circuit which produces an increase in temperature by applying voltage to a Nichrome wire. The circuit consists of a relay switch, diode, and transistor (PNP). This circuit operates as follows: 1) Command signal line from OBC, relay switch, and power source line are connected to the transistor base, collector, and emitter, respectively. Transistor base is kept high until it receives the command signal from OBC, 2) OBC sends the Low signal to the transistor base, 3) Amplified power flows via the collector to the relay switch's coil which switches on the relay switch, 4) The relay switch connects the Nichrome wire to the battery, and 5) Heated Nichrome wire cuts off the Nylon cord holding the deployable structure. This type of Nichrome wire fusing system is used not only in CanSat but also in many other deployable structure releasing systems of Nano-satellites. For example, Nihon University's SEEDS also uses this system where heated Nichrome wire melts the nylon cord holding the antenna, when OBC's timer count exceeds a certain threshold after releasing from the rocket.

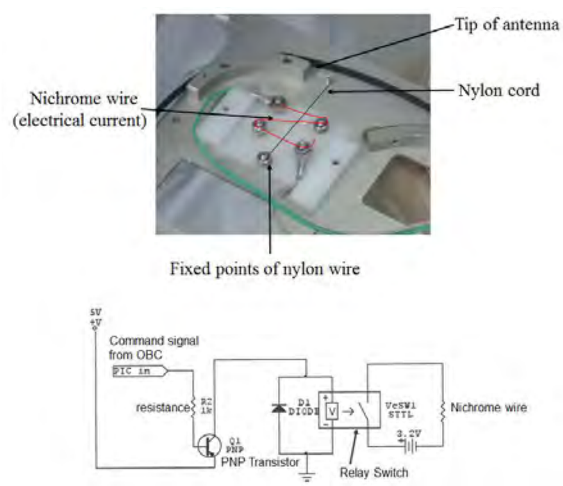


Figure 4-43: Nichrome wire fusing setup and its circuit.

Some CanSats need to control an actuator based on acquired and stored sensor data. In this case, it is very important to acquire and store sensor data to accomplish the actuator control for the desired mission.

4.6.3 System Architecture and Verification Method

This subsection explains the composition of an actuator-equipped CanSat system and its verification experiment. Consider a fly-back CanSat mission which uses a sensor and actuator for guidance control to the destination. The mission profile is shown in Figure 4-37. In this mission, after separation from the rocket, CanSat falls by parachute from 4,000 m to 500 m altitude to avoid the strong wind effect. Below 500 m altitude, it releases the parachute and deploys the parafoil. Then the CanSat lands to the ground while controlling the parafoil.

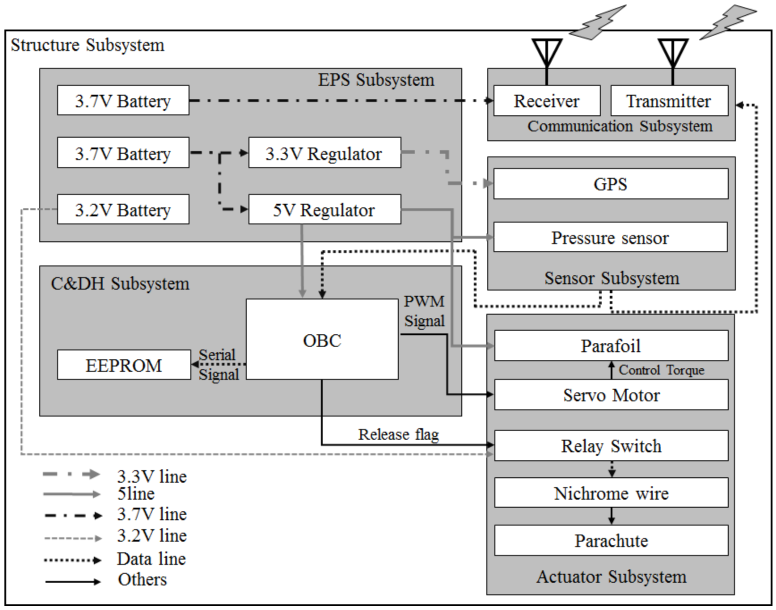
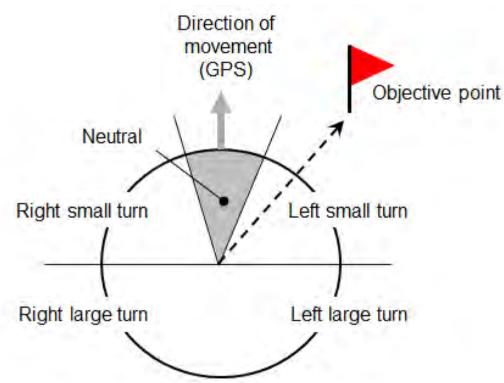


Figure 4-44: Example of system diagram for a fly-back CanSat's mission.

Figure 4-44 shows a system diagram example of a CanSat mounted with an actuator and sensor realizing the mission profile shown in Figure 4-37. The CanSat operates as follows: 1) The barometric sensor detects the deploying altitude and the OBC receives that data from the sensor, 2) The OBC monitors the CanSat altitude and it reaches the target altitude, the Nichrome wire fusing apparatus separates the parachute and the parafoil deploys, 3) After switching to parafoil control, the GPS sensor detects the CanSat's location, 4) The OBC receives location information from the GPS sensor, it determines the control signal that minimizes error margins to the destination point, and 5) The servomotor is controlled by signal sent from the OBC to steer the parafoil and minimize the position error.

To make the CanSat's actuator work as planned, it should be considered not only its elemental function but also if it works successfully or not in combination with other subsystems. For example, the EPS subsystem supply the actuator with operating voltage and power capacity. C&DH subsystem, sensor, and communication subsystems are interacting with actuator on sampling and control interval, and error in control signal resulting from noise is added to the sensor by communication subsystem interference. In addition, the structural subsystem is affecting the actuator by control force and torque error occurred by the any alignment error in the mounting of control actuator and sensor. CanSat should be design taking these circumstances into consideration. It is also essential to conduct a verification test in a simulated environment, to sort out any problems overlooked.

Figure 4-45 shows an example of a controlling direction algorithm, in line with the relation between the location information acquired from GPS sensor and the location information of destination, and its verification test. As shown in this algorithm, the parafoil is controlled by servomotor. Sending the control signal to steer the CanSat to the destination location can have five cases, specifically; Neutral, Right small turn, Left small turn, Right large turn, and Left large turn. The present verification test checks the functionality of actuators combined with other subsystems such as sensors and OBC. The test also judges if the CanSat moves towards destination according to the actuator's control defined by the algorithm's code.



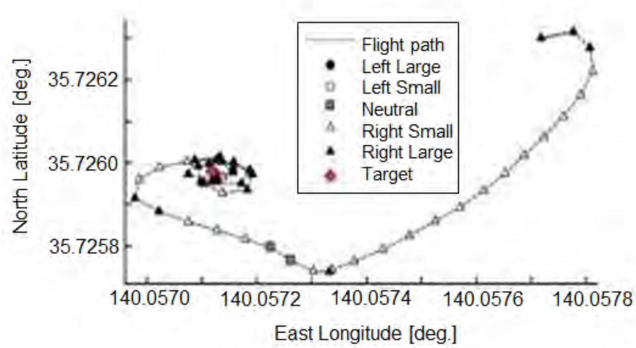


Figure 4-45: Control algorithm and its steering performance test result.

Figure 4-46 shows a verification test setup and results to confirm if power is successfully supplied during mission. The mission profile in Figure 4-37 consists of the following phases: 1) Launch standby phase, 2) Parafoil control phase by servomotor, and 3) Communication phase for informing its location to the ground station after landing. Therefore, in second phase, it is required to check the actual power consumption by putting load on the servomotor, and confirm that the electric power balance is maintained during the whole mission.

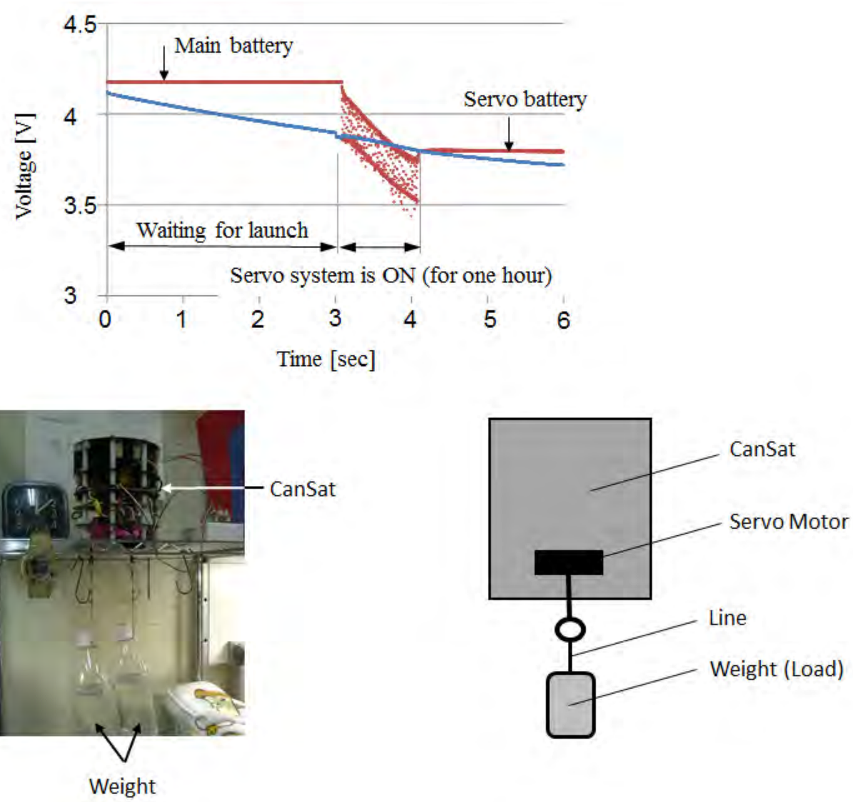


Figure 4-46: Control actuator and its power performance test setup and results.

Figure 4-47 shows an example of a parafoil performance confirmation test. During the flight trajectory controlled by parafoil, adjustment in the drive angular position that draws the control line of the parafoil is an important parameter. Therefore, efficient control is impossible without a thorough understanding of parafoil features such as its turning characteristic. Wind tunnel test, gliding test, and balloon test are vital for performance verification.

Figure 4-47: Parafoil performance confirmation by wind tunnel test.

4.7 Design of Structure Systems

The structure system of a CanSat consists of body structures that hold on-board components, and deployable structures such as the parachute and parafoil. This section describes some important aspects in the design of these structure subsystems.

4.7.1 Functions of Structure Subsystems

The structure subystem of a CanSat perform the function of physically holding the on-board components during the entire lifecycle of the CanSat, such as the design, fabrication, verification (test and analysis), design change, repair, pre-flight operation, mission, and post-flight phases. The factors that may damage a structure subsystem include a static acceleration load, vibrational load, and shock load. If the structure cannot resist these loads, part of the CanSat may deform and/or break. Hence, the structures are required to be designed to have sufficient strength for resisting these loads. As the size of structures becomes smaller, the design for strength becomes easier. Nevertheless, if the locations of load concentration are not properly considered, the mission may fail. In addition, the deployable structures, such as parachutes, are stowed compactly in a rocket, and they must be deployed reliably after their release from a rocket. If the deployment fails, the CanSat will drop to the ground and will not be able to conduct missions; furthermore, the CanSat may be destroyed.

Figure 4-48 shows a mission sequence of a CanSat called "Space Crawler," developed by students at the Tokyo Institute of Technology, Japan, and launched during ARLISS 2009 annual event. This CanSat was aimed at realizing the following mission. First, the CanSat conducts flight control using a parafoil during the descent phase and passes through a prescribed target domain in the air. It then separates the parafoil for landing and then it runs toward the target point on the ground as a crawler-type rover. After reaching the target point, it runs continuously along an 8-shape path. In the following subsection, the design of CanSat structures and the design process are explained using "Space Crawler" as an example.

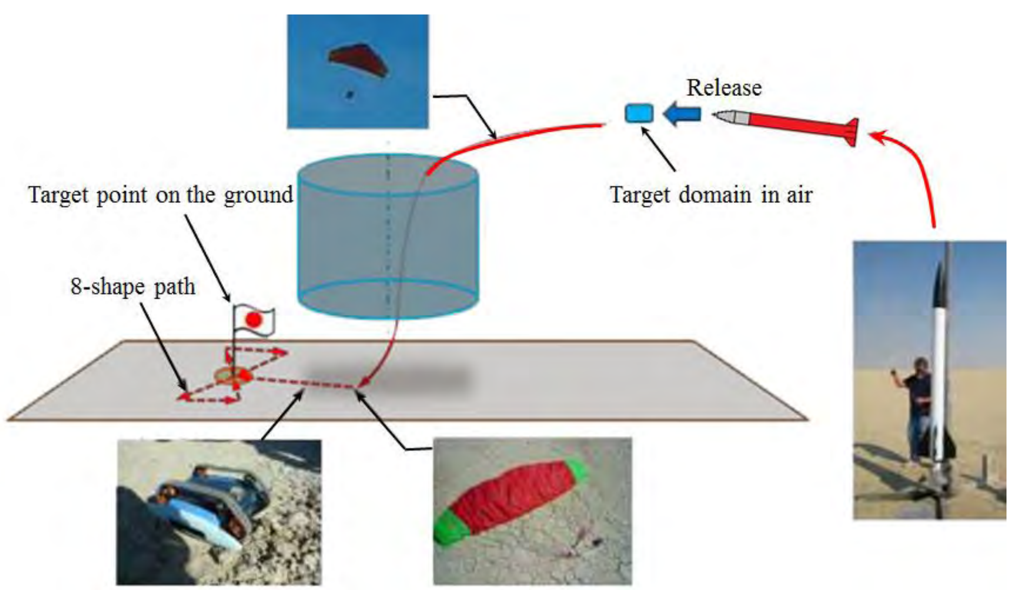


Figure 4-48: Example a mission sequence of CanSat.

4.7.2 Requirements and Design Process

Like the design of all subsystems in the CanSat, the first step of designing a structure system is the consideration of the mission sequence. Once the mission sequence is specified, all the requirements in each mission sequence need to be written down as quantitatively as possible. For example, in the case of “Space Crawler”, the mission sequence and the requirements in each mission sequence are listed in Table 4-4.

Table 4-4: Example of mission sequence and requirements for structures in each mission sequence.

Mission Sequence	Requirements
Launch and Ascent	· Volume and mass must be less than the regulations · It must resist static acceleration load and vibrational load · Parafoil must be properly stowed
Release and Deployment of Parafoil	· It must resist the shock load of release · Parafoil must be successfully deployed
Descent and Flight Control	· Parafoil must maintain its shape during flight
Landing	· It must resist the shock load applied on the ground
Separation of Parafoil	· Parafoil must be separated from CanSat’s body · It must not tangle with parafoil
Ground Locomotion	· It must run on desert without stacking

When the requirements are identified, structure subsystems that satisfy all the requirements are conceived. It is important to place the on-board components within the restricted volume (configuration design). The mechanical and electrical interferences among the components, i.e., interference among subsystems, should be considered. Note that the placement of the components should facilitate maintenance, such as situations when screws are fastened by a driver.

In fact, this is not an easy task at all. Sometimes, during the development stage, the configuration may be updated inevitably, and the design of structures should be modified accordingly. To prevent this situation, it is important to write down all the requirements in a wider lifecycle of the CanSat system than in just the mission itself. The requirements for development, design change, test, repair, and pre-flight operation should also be considered. Once the requirements are noted, the designs of structures are gradually determined by following the process chart shown in Figure 4-49.

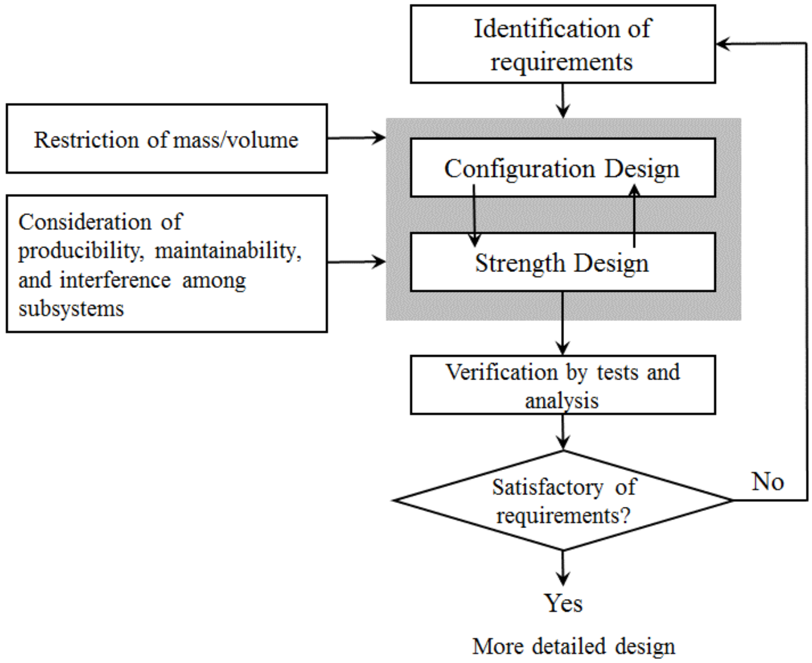


Figure 4-49: Design process of structure subsystem.

4.7.3 **Fundamentals for the Strength Design of Structures**

4.7.3.1 Stress and Strain

Before conducting strength design, it is important to know how structural elements are broken or deformed permanently, such as being bent and not returning to their original shape. When a force P is applied axially on a structural member with a cross-sectional area A, the force per unit area, P/A, is called stress. When the stress σ is larger than a certain value, which is known as the yielding stress, the member is deformed permanently. The yielding stress varies from material to material. Therefore, when a large force P is applied, the structure should be designed such that the stress in the member does not exceed the yielding stress. For safety, a stress value smaller than the yield stress is often used as an allowed value, which is called allowable stress. The fundamental concept of strength design is to establish the stress such that it does not exceed the allowable stress, by increasing the cross-sectional area A and using materials with a larger yielding stress.

Assume the length of a structural member to be L and its elongation when force is applied to be ΔL; then, the ratio ΔL/L is called strain.

The relation between the strain ε and the stress σ varies depending on the kind of material. Some materials have a larger strain for the same stress; in the other words, some materials are easily elongated, whereas others are not. It should be noted that materials such as glass, which does not elongate easily, sometimes break suddenly once the allowable stress is exceeded. These materials are called brittle materials. On the other hand, some materials such as aluminum elongate gradually even after the allowable stress is exceeded, and they break when the stress becomes larger than a certain value, called the breaking stress or ultimate stress. These materials are called ductile materials. It is important to choose appropriate materials based on the requirements.

4.7.3.2 Buckling

Thin columns or thin plates (thin shells) are often used to reduce the structural mass; then, when a structural member is in compression by an axial force, the member sometimes exhibits a large deflection even if the stress does not exceed the allowable stress. This phenomenon is called buckling. The large deflection often causes large stress at a part of the member, which may exceed the allowable stress. Hence, in thin shell structures, consideration of buckling strength is often more important than consideration of strength for axial elongation and contraction.

It is convenient to know the following equation of the Euler buckling load, to design structures in which buckling can be prevented. This equation shows the buckling load for a thin column with a uniform cross-section. When a compression force P is applied axially to a thin column, buckling occurs if P becomes larger than the Euler buckling load P_{cr}.

$$P_{cr} = n\frac{\pi^2 EI}{L^2} \qquad \text{Equation 4-2}$$

Where E is the Young's modulus of the material, I is the second moment of area of the column, and L is the length of the column. The second moment of area, I, is determined by the shape of the cross-section. For example, in the case of a rectangular cross-section, $I = ht^3/12$, where h is the width of the column and t is its thickness. Furthermore, n depends on the boundary conditions at the top and bottom edges of the column. The coefficient n takes a value between 0.25 and 4.

The Euler buckling equation is based on the assumption of a thin column with a uniform cross-section. However, the knowledge obtained from the equation can be applied to the design of any thin shell structures. The parameters that affect the design and the buckling loads are as follows: First, the critical load for buckling is affected by the square of the length of shell structures. For example, if the length of the column doubles, it buckles at 1/4 of same load. Second, an effective way to prevent buckling is to increase EI, which is called the bending stiffness. For example, if the thickness is doubled, the bending stiffness can increase 8-fold. Thus, the allowable load against buckling also increases 8-fold. It should be noted that if the thickness is doubled uniformly, the mass doubles as well. However, if the cross-section is made I-shaped, for example, EI may be increased without increasing the mass. Finally, depending on the boundary conditions, n varies from 0.25 to 4, which corresponds to a 16-fold difference in the buckling strength. Thus, the way in which the structural member is fixed is rather important.

For the CanSat, typically, the length L of structural members is small. Thus, buckling will be effectively prevented by ensuring that EI is not too small.

4.7.3.3 Resonance

When the CanSat is launched by an amateur rocket, a vibrational load, in addition to a static acceleration load, is applied on the CanSat. Structures possess a property called "resonance", by which the structure deforms considerably owing to a vibrational load at a certain frequency called natural frequency. Therefore, structures should be designed such that the CanSat's lowest natural frequency is higher than the frequency range of the rocket's vibration. Alternatively, structures should have enough strength or high damping so that they do not exhibit a stress larger than the allowable stress even during resonance.

Determination of the values of the natural frequency or stress during resonance requires special knowledge. Roughly speaking, the following are the factors that increase the natural frequencies: reducing the mass of the structure; reducing the size of the structure; or increasing the stiffness of the structure by using materials with a higher Young's modulus, larger cross-sectional area, and/or larger bending stiffness. Furthermore, structural members and on-board components should be fixed using screws, instead of using tapes or adhesives to have fixed boundary conditions. Since a typical CanSat is small and lightweight, its natural frequencies are typically high. Nevertheless, natural frequencies can reduce significantly if the components are not fixed properly.

4.7.4 Verification of Strength Design

In general, during the strength design of a real satellite, numerical models are constructed on a computer and the values of stress in the structures are predicted by a numerical method termed the Finite Element Method (FEM). However, the CanSat is, as presented above, typically strong against buckling and resonance because of its small size. Furthermore, its structure is typically much simpler than that of a real satellite. Therefore, computation by the FEM, which requires the use of special knowledge, will not be necessary for the strength design of the CanSat; instead, the following steps will be sufficient:

1. Fundamental structural theories, such as those explained above, are used to estimate stress values through simple calculations such as hand calculations. Based on these simple calculations, the materials and dimensions of the structural members are chosen. Then, a prototype of the CanSat is designed and fabricated.
2. The prototype is tested by applying loads larger than that in an actual mission, typically 1.25–1.5 times the expected loads, to verify that there are no broken or permanently deformed parts. If any malfunction is detected, the part is redesigned to make it stronger and another prototype is fabricated and re-tested. This process verifies that "the design is right."
3. A flight model is fabricated, whose design is identical to that of the final prototype. Then, the flight model is tested to verify that "the fabrication is right".

The following are the three typical mission phases where large loads are exerted on the CanSat: 1) Launch by an amateur rocket (static acceleration and vibrational loads), 2) Release of the satellite and parachute opening (parachute-opening shock load), and 3) Landing (landing shock load). In the following, test methods are described that correspond to each of these three phases. In each test, structures are examined to determine whether there are any broken or permanently deformed parts.

During this examination, not only the global structural members but also the following local parts should be carefully examined to check for malfunctions such as fractures.

- Jointed parts, such as those fastened by bolts
- Solders
- Cables and connectors
- Motors and gear interiors

4.7.4.1 Vibration Test

When an amateur rocket is launched at ARLISS event, the CanSat on the rocket may experience larger vibrational accelerations than would an actual satellite on a space rocket. Therefore, it is important to conduct a vibration test using a vibration exciter, as shown in Figure 4-50.

Vibration tests are typically performed by using sinusoidal excitation and random excitation. First, a sinusoidal acceleration at a certain frequency with a small uniform amplitude is applied, and the frequency is gradually varied within a range between 5 - 100 Hz, which corresponds to the property of the launching rocket. Then, it is observed which resonance modes appear and at which frequencies. This process is termed a modal survey. Then, the sinusoidal excitation is applied again with higher amplitudes, which correspond to the rocket properties, and random excitation is provided by applying input acceleration signals with various frequencies and various phases simultaneously. The CanSat should resist vibrational loads at the time of launch. If any malfunction is detected during these tests, the design should be revised accordingly, and the CanSat should be tested repeatedly until no malfunction is detected.

Furthermore, during the rocket launch, the CanSat is subjected to a static acceleration load. Tests for static loads are often substituted by vibration tests described above, by using sinusoidal excitation with a very low frequency. Some participating teams at ARLISS test their CanSat by using an alternative method which is described as follows. A string is connected to a bucket and rotated with a CanSat inside the bucket. If the length of the string is r meters in length and the rotation is realized with a uniform frequency f in Hz, then the CanSat experiences the acceleration of $r(2\pi f)^2$ in m/s^2.

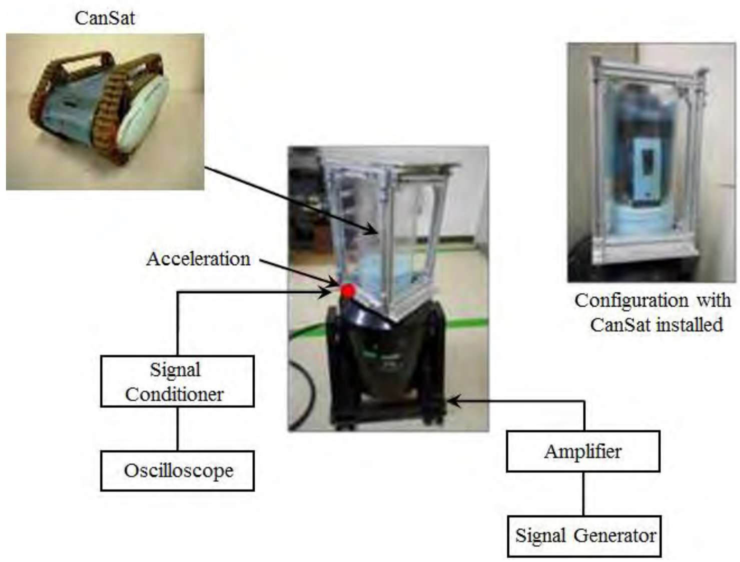


Figure 4-50: Example of CanSat's vibration test performed using magnetic exciter.

4.7.4.2 Satellite Release and Parachute-Opening Shock Load Test

One of the most common failures at ARLISS is that which occurs during the phase of release of the CanSat from an amateur rocket and the deployment of a parachute or another speed reducer device. The failures are caused mostly by fractures or permanent deformation of the CanSat structures resulting from shock loads applied by a parachute. To prevent this malfunction, first, the shock load value should be roughly estimated during the design phase. Then, structural members that pass the load are identified, and the members are designed to be strong enough to resist the load. Finally, shock load tests should be conducted using fabricated hardware.

The easiest method for performing shock load tests is to attach an additional mass to the CanSat with an appropriate factor of safety. The CanSat is suspended by its parachute's cables, with the parachute closed and fixed. Then, the CanSat is dropped from an appropriate height. It should be noted that the results of this test depend on the order through which the cables apply a load, in the case that the parachute connects to the CanSat via multiple cables. Hence, it is important to conduct tests under such conditions.

4.7.4.3 Landing Test

Like rover-type CanSats, some CanSats continue the mission after landing on the ground. In this case, it should be ascertained that the CanSats survived the landing shock load. An example of an effective test method is to land a CanSat from somewhere high in the air, such as from a balloon or a high raise building, by using a parachute or other speed reducers like its landing in its actual mission. If only the structural strength need to be verified, a simpler method is to estimate the velocity of landing with an appropriate factor of safety. Then, the CanSat is dropped from an appropriate height that satisfy the landing velocity without deploying a parachute. It should be noted that the path and amplitude of the shock loads depend on which part lands first or how hard the land is. Hence, it is important to conduct tests under various conditions by controlling the values of parameters.

4.7.5 **Configuration Design**

Since CanSat structures are small, satisfying the requirements for strength design is not very difficult task, if the design and verification are in line with some important basics explained in the previous subsections. However, owing to the small size of CanSat structures, the following tasks become quite challenging: 1) To keep its volume within the restricted volume, 2) To install the components while facilitating production and maintenance, 3) To keep its mass within the restricted value, and 4) To consider the interference among subsystems and components. These four tasks are explained in greater detail below in the following subsections.

4.7.5.1 Management of Stowage volume

Based on the requirements during the "launch and ascent" phase, the CanSat should be stowed within a restricted volume, as shown in Figure 4-51.

Figure 4-51: Example of CanSat's stowed configuration.

To design a CanSat that can be stowed in a restricted volume, the following design process is generally adopted.

- Consider the component configuration based on hand-drawn sketches.
- Fabricate simple prototypes from cardboard, styrene foam, plastic board, wood, sheet metal, or other materials and write down notice and feedbacks for considerations in the detailed design.
- Conduct detailed design by drawing using, for example, 3D CAD software.

Figure 4-52 shows examples of prototypes constructed during the development of "Space Crawler". In fact, quicker and shabbier prototypes were constructed from cardboard and styrene foam, although they are not shown in the figure. Then, the prototype rovers made from wood, shown in the bottom left of the figure, were constructed to determine the detailed designs gradually. In the past, a CanSat team constructed over 20 prototypes before establishing a final structure design, to accomplish a successful mission at ARLISS.

3D CAD will be extremely useful in the case that there are many onboard components or structural members. As shown at the bottom right of Figure 4-52, the shapes can be checked in three dimensions before constructing any hardware. In addition, the maintenance process, which is discussed in the following subsection, can be verified. Thus, this will enable determination of the components' configuration without having all the hardware components available. Various software programs for 3D CAD are available, ranging from free ones to commercially expensive ones. The commercial software programs SolidWorks, Creo Elements/Pro, and CATIA are commonly used for this purpose. Additionally, based on a 3D CAD model, a Finite Element Analysis (FEA) enables computations for strength design and it is commonly used for the development of real satellites.

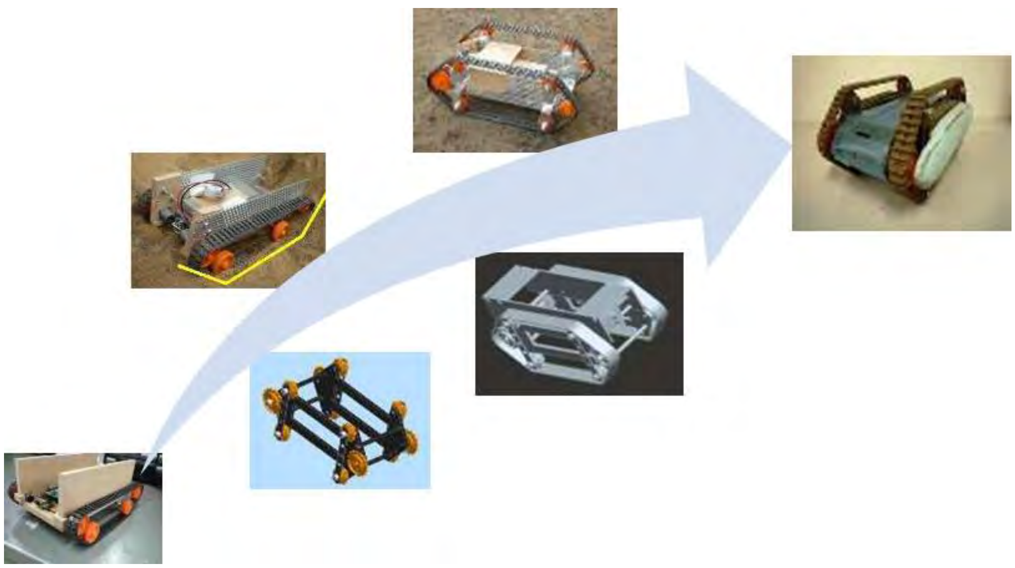


Figure 4-52: Examples of CanSat prototypes.

4.7.5.2 Producibility and Maintainability

In the CanSat lifecycle, the "production" phase requires the CanSat to be at least producible, and its design should facilitate production as much as possible. The following tips will be useful for achieving better producibility.

- Search for machine tools that are accessible to the team, such as a sawing machine, drilling machine, milling machine, lathe, and other tools. Then, create the structure design to be fabricated by using these tools as much as possible. If hand fabrication is not feasible, the team should know at the early development stage who can be requested to perform the fabrication and how.
- Design a structure as a combination of simple plates and columns, and assemble these simple elements, instead of employing many machining processes.
- Use materials that are easily available and easy to machine. Generally, aluminum alloy has good availability and machinability.

- Carbon Fiber Reinforced Plastic (CFRP) is lightweight and has high strength/stiffness. However, it is difficult to machine, and thus, a team needs to carefully discuss the advantages and disadvantages of its use.

Next, in the CanSat lifecycle, the “design change”, “test”, “repair”, and “pre-flight operation” phases require maintainability of the CanSat. For example, the location of switches must be easily accessible. The Light Emitting Diode (LED) used for monitoring the condition of C&DH subsystem should be visible from outside. If a design parameter is uncertain, the design should be flexible enough to permit its change after some tests. When a part is detected to be broken during a test, it requires to be repaired as rapidly as possible. Hence, a structural engineer must understand methods of maintenance for all subsystems during the entire lifecycle and design a structural configuration that enhances maintainability.

4.7.5.3 Management of Mass

The “launch and ascent” phase requires that not only the packaging volume but also the mass of the CanSat should be less than restricted values. The management of mass is very important since it significantly affects the system reliability.

An effective method to keep the mass within the restricted value is required to continuously estimate and control the mass, starting at the earliest preliminary design stage. Furthermore, it is effective to keep a mass margin of approximately 20% at the beginning. Then, as the design becomes more detailed, the mass margin is decreased gradually, and the mass converges to a final value within the restricted value. If the estimated mass is larger than the restricted value at the beginning, as indicated in Figure 4-53 by circles at the top of the restricted mass value line, the thickness of structural elements will be reduced at end. As a result, even if various tests are conducted, the strength of the CanSat will be reduced after the tests. Such a CanSat will fail in its mission.

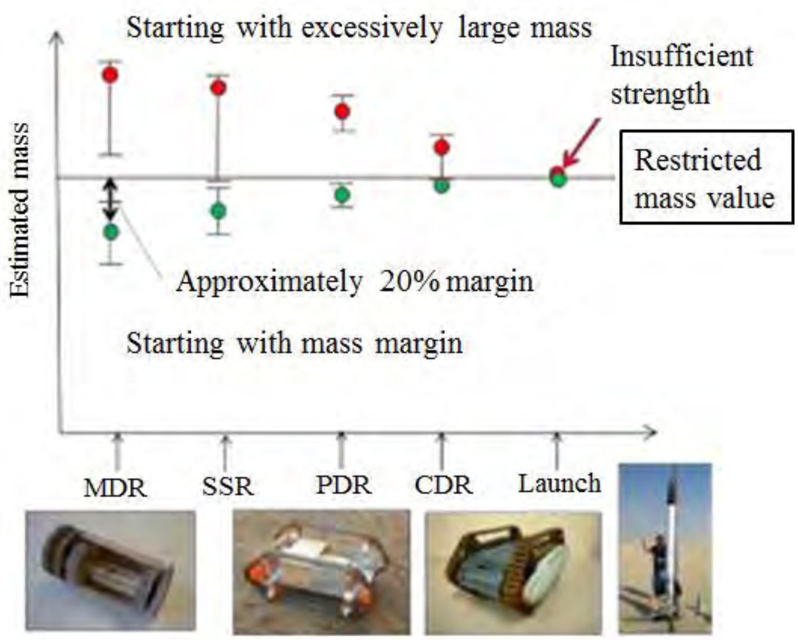


Figure 4-53: Development process with sufficient/insufficient mass margin.

It is important to estimate the mass as accurately as possible, even at the conceptual design stage. 3D CAD is very useful for accurate estimation of mass. Note that joint parts such as bolts, screws, and nuts should not be excluded during the mass estimation. These joint parts generally have a significant effect on the CanSat's mass. Joint parts made from plastic are available and are sometimes useful for reducing the mass. However, their strength is lower than that of metal parts, and therefore, careful strength tests become necessary in such a case.

4.7.5.4 Interference Among Other Subsystems

In the configuration design, the placement of components is determined by considering various interferences among subsystems during the mission. Not only do the components interfere physically with each other but also the following requirements typically exist regarding the interferences among the subsystems of the CanSat.

- Structures should not interfere with the reception of GPS signals.
- Structures should not interfere with communication units.
- Electromagnetic interference should be prevented between a quartz oscillator unit and a GPS receiver.
- There should be no interference between motors and the magnetic compass.

The best way to prevent the occurrence of these undesirable interferences would be to actually conduct experiments by using prototypes, without relying on all the existing information. Interference among subsystems often causes mission failures. Thus, it is important to verify the feasibility of the CanSat mission in an end-to-end test as early as possible, which simulates the beginning to the end of an actual mission. Once any malfunctions are detected, their causes should be examined carefully, and appropriate countermeasures need to be conceived.

4.7.6 **Design of Deployable Structures**

This section discusses "deployable structures", which are stowed compactly during launch and deployed when the CanSat is released from a rocket. Occasionally, antennas, solar panels, and other CanSat structures are deployed, but herein, only a parachute is discussed as it is the most typical deployable CanSat structure. The following are the five aspects of deployable structures that need consideration.

a) Determination of size and shape based on requirements during the "descent" phase
b) Design and fabrication
c) Stowage
d) Deployment method
e) Strength design against deployment shock load

First, in terms of the aspect of the size and shape, the descending velocity is determined based on the requirements in the "descent" phase. The size of a parachute with a terminal velocity V can be estimated using the following equation:

$$V = \sqrt{\frac{2mg}{\rho A C_d}} \quad \text{Equation 4-3}$$

Where m is the total mass, g is the gravitational acceleration and equals to 9.81 m/s^2, ρ is the atmospheric density, A is the area of the parachute, and C_d is the drag coefficient of the parachute. The drag coefficient C_d is a nondimensional number, which depends on the parachute shape. An estimated value, reported in some literatures, will be substituted during the early design stage.

Second, in terms of the aspect of design and fabrication, probable designs are converged to one design by iterating prototyping and testing, while considering how many parachute cables, i.e., number of polygon sides, are used, whether a hole is made in the parachute (a hole stabilizes swinging in the case of reduced altitude and decrease the C_d), and other problems. Ripstop Nylon is commonly used as parachute fabrics.

Third, the stowage method, which refers to the folding pattern, has a strong effect with the reliability of deployment. Deployment tests should be conducted until a folding pattern with sufficient deployment reliability is found. In addition, it is important to repeat the same folding pattern accurately once a reliable folding pattern is found. Effective methods include recording movies of the folding process and making a checklist of the folding process so that the process can be verified by multiple persons. To ensure high repeatability, wrinkles on the parachute should be removed to the best possible extent, and the stowed configuration of each parachute cable should also be controlled.

Fourth, two points need to be discussed pertaining to the deployment method. First, the deployment of a parachute is often used to pull the flight pin commonly known as kill switch to turn on the main power of the CanSat. Figure 4-54 shows examples of used flight pins of the CanSat. These flight pins should not be pulled out during the stowage process or the ascent phase. Instead, they should be reliably pulled out when the parachute is deployed. To guarantee these functions, careful tests are required.

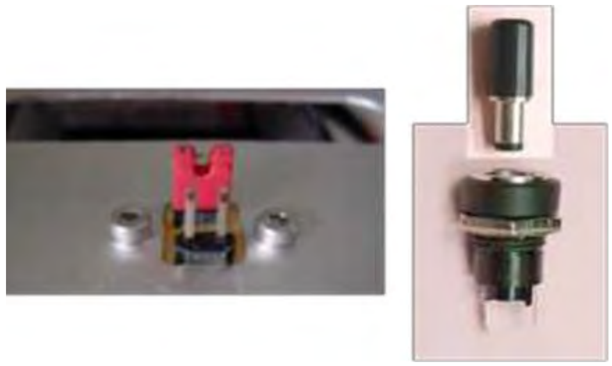


Figure 4-54: Examples of flight pins of CanSat.

The second point, if a deployable structure, such as a parafoil, is deployed not at the time of release from a rocket but at some point of time in the air with a time lag from the release, then a deployment mechanism is necessary. For example, Figure 4-55 shows bands that hold a folded parafoil. The parafoil is folded during descent at high altitudes where winds are strong. Then, these bands are separated at a lower altitude by fusing fishing wires with a Nichrome wire. This kind of fusing mechanism with Nichrome wires is relatively reliable; thus, many CanSats are commonly used such mechanisms.

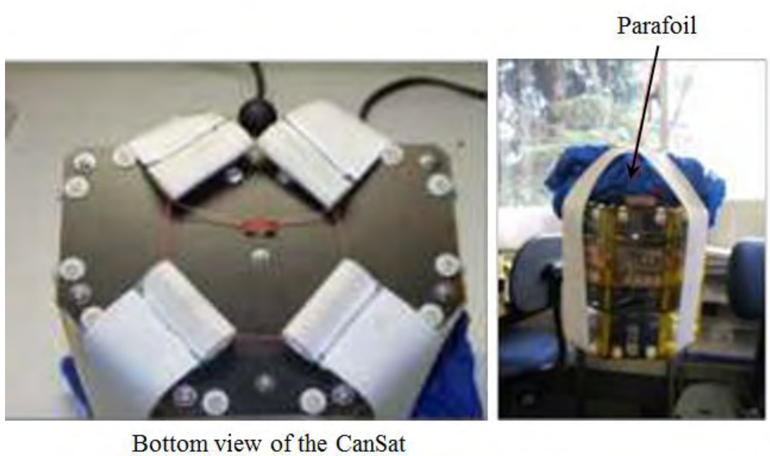


Figure 4-55: Example of parafoil deployment mechanism using fishing and Nichrome wires.

Fifth, pertaining to the final aspect of the strength design against parachute deployment shock load, deployment tests should be conducted carefully, as described above.

4.7.7 **Examples of Test and Design Changes**

Two examples of tests and design changes of "Space Crawler" are shown in Figure 4-56. In this CanSat, the quantification of requirements for the crawler mechanism was difficult. Students had to think how much roughness of the terrain should be conquered and then plan locomotion tests as quantitatively as possible. Based on the photographs of the desert at Black Rock desert in Nevada, a dummy terrain was constructed in a room and the CanSat was tested on it. As a result, two failure modes shown in Figure 4-55 were identified. A deadlock on a bump and an overturn of the CanSat. Therefore, its design was changed to countermeasure each of these possible malfunctions by narrowing the distance between two wheels and by adding foam parts on the side of the crawler to prevent overturns.

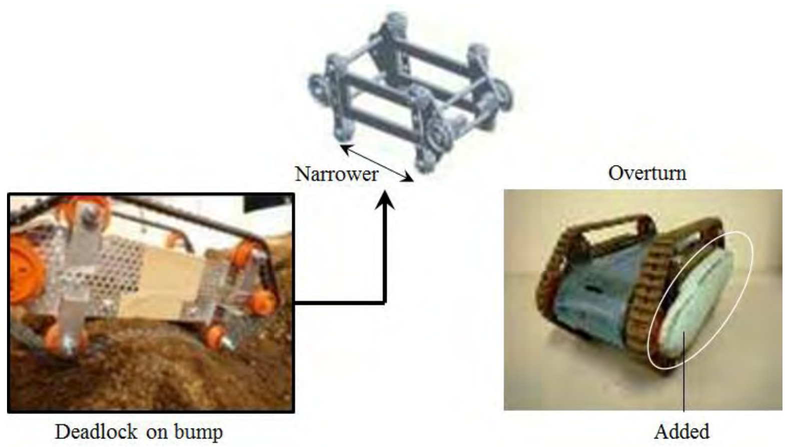


Figure 4-56: Examples of failure mode identification and design changes through tests.

In the design of the CanSat, it is critically important that there are existing methods to verify before the flight and ensure that the design satisfies all requirements by using tests and/or analyses (verification). Designs that cannot be verified before a flight should not be used. It is important to always remember Murphy's Law: "Anything that can go wrong, will go wrong." Like all subsystems of the CanSat, thorough verification is the key to successful structure design.

4.8 Development of the Ground Station

The role of the ground station is to track CanSat, downlink telemetry information to maintain and manage the CanSat status, uplink commands to CanSat, and downlink mission data. Therefore, between CanSat and the ground station, a system that receives/transmits data according to a predetermined protocol need to be adopted. As shown in Figure 4-57, the ground station consists of receiving antenna, receiver, modem performing modulation and demodulation, downlink software for analyzing data via a predetermined protocol, and uplink software that creates commands.

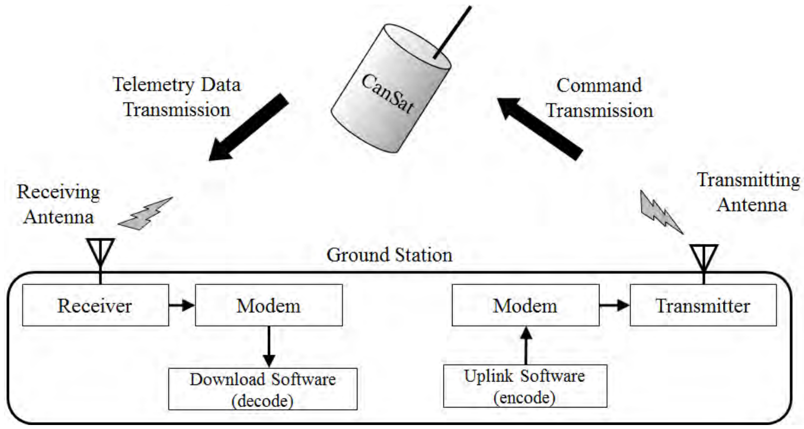


Figure 4-57: Conceptual diagram of ground station.

After the CanSat release from rocket, CanSat performs some sort of mission and communicates with a ground station connected to PC, until falling to the ground over a period of few seconds to 30 minutes depending on the CanSat initial altitude and parachute performance. For example, if CanSat is the one that performs amateur radio communication, it is advisable to prepare Yagi antenna, Terminal Node Controller (TNC) built-in radio to perform modulation and demodulation, and PC installed with ground station software, as shown Figure 4-58. If there is no TNC built-in radio, it can be built by microcomputer and modem. Ground station can be configured by that TNC, antenna, radio, and PC. Amateur radio ground station is commonly adopted for long distance communications, i.e., above 1000 meters which is a typical CanSat mission in ARLISS.

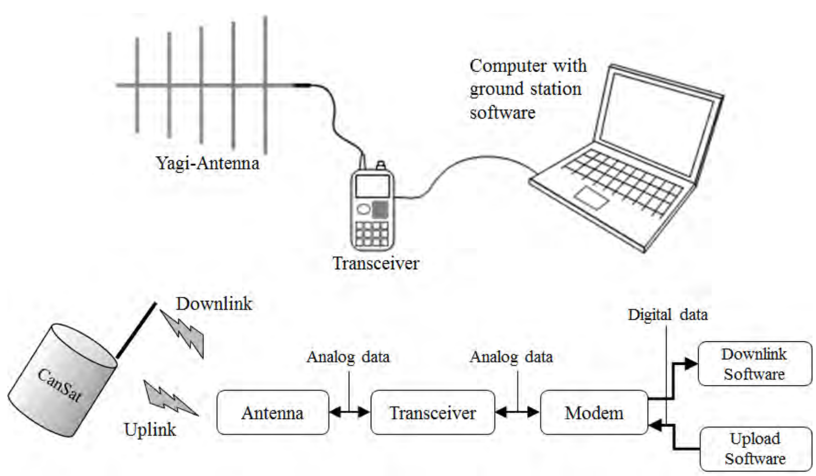


Figure 4-58: Ground station using amateur radio.

If XBee radio, in which functions of antenna, radio and modem are integrated in one package, is used to communicate between CanSat and ground station, a laptop and XBee are the only things needed to create a ground station as shown in Figure 4-59. If the mission requirement of CanSat's communications distance is around hundreds of meters, it is easier to assemble a communication system between CanSat and ground station by using XBee.

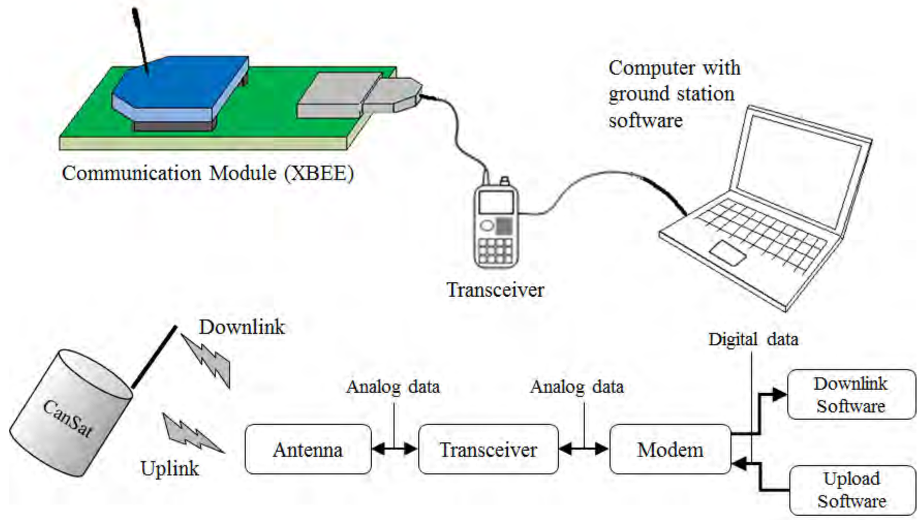


Figure 4-59: Ground station using XBee radio.

To uplink commands to the CanSat, digital data based on predetermined protocol and commands along with their format need to be created, using the uplink software installed on PC. This digital data is modulated by the modem, then converted into analog data to be transmitted to the CanSat by radio through the antenna. On the other hand, when downlink, the radio receives the analog data sent from CanSat through the antenna and converts the analog data into digital data by a demodulating modem. The converted digital data will be analyzed by downlink software installed on the PC of the ground station.

In order to appropriately receive the communication data from CanSat, the operator of the ground station would need to direct the antenna to the CanSat by visual management or by using the data from the CanSat. This is because each antenna has a directional characteristic which represents the sensitivity characteristics of the antenna with respect to its direction. Antennas are roughly divided into two types by their directive property. One is a non-directional antenna or omnidirectional antenna which indicates the same sensitivity in all direction, and the other is a directional antenna which shows a variable level of sensitivity depending on the direction of the antenna. Figure 4-60 shows the sensitivity pattern of the omnidirectional antenna indicating same sensitivity to any direction and the directional antenna having a high sensitivity around the 0° direction.

If omnidirectional antenna shown in Figure 4-60 (a) is used, it is not necessary to turn the antenna to the reception target, since the radio waves are equally received all around. This is effective in the case when position of the CanSat cannot be identified visually or by other data, for example GPS data sent from CanSat. However, the reception intensity of a directional antenna is stronger than omnidirectional antenna. As shown in Figure 4-60 (b), if a directional antenna such as a Yagi antenna is used, efficient communication by receiving only necessary radio waves can be done. In addition, it is preferable to note that the directivity exists for both horizontal and vertical direction of the antenna.

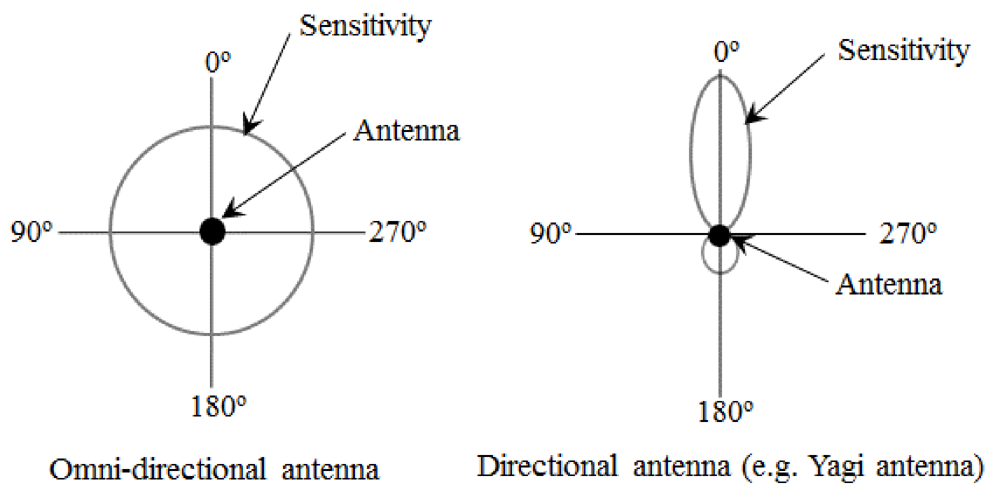


Figure 4-60: Example of an antenna's directional characteristic.

In general, CanSat ground station, radio transceiver and PC are connected through some communication interfaces. Therefore, it is possible to display the raw data sent from the CanSat, by a terminal software application such as Tera Term or Tuna Term installed on your PC. This is commonly done during development phases of ground station software. Usually, ground station software with a user interface that only receives and transmits is not enough. It must be designed as operator friendly. For instance, when sending a command based on the CanSat's status or its location information, data must be displayed in an easily understandable format so that the operator can immediately understand. Figure 4-61 indicates an example of the ground station software. In the example shown in Figure 4-61, the ground station's positional information is saved in the ground station software in advance. This enables the operator to learn which direction to point the antenna to, by graphically demonstrated positional relationship between the ground station and CanSat, based on GPS data sent from CanSat.

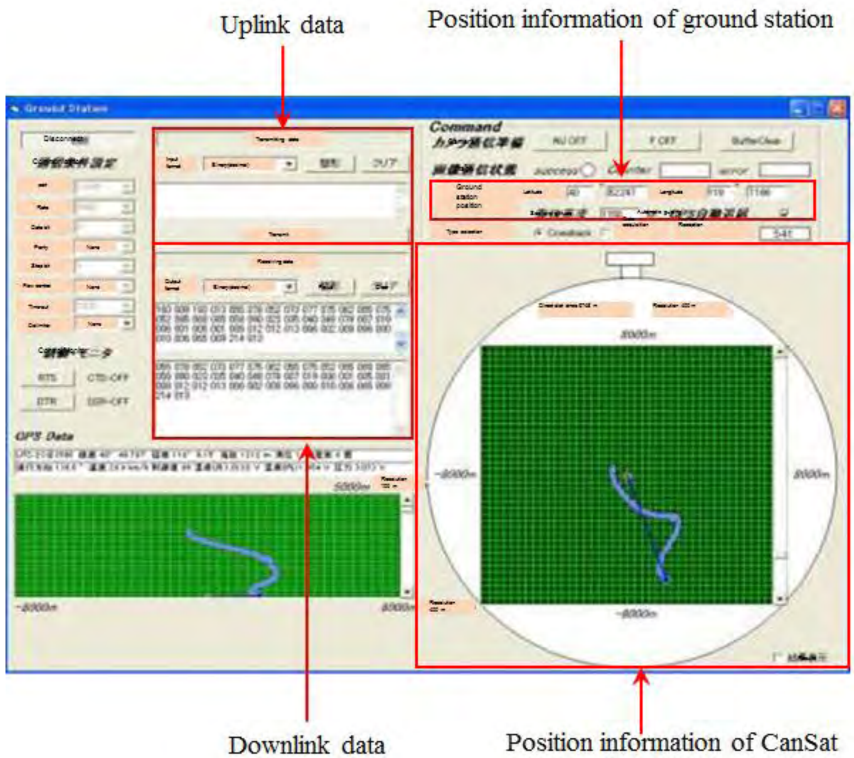


Figure 4-61: Example of ground station software.

The ground station for satellites such as CubeSat, as the case with CanSat, has the functions of track satellite, downlink telemetry information, uplink commands, and downlink mission data. Figure 4-62 indicates a general CubeSat ground station system diagram.

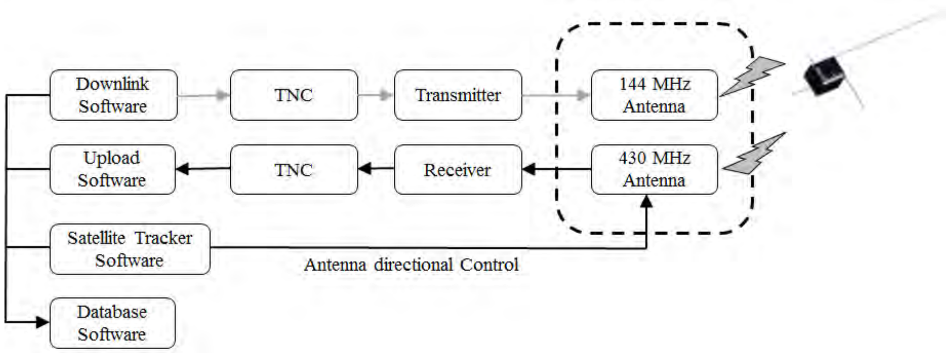


Figure 4-62: Satellite ground station system diagram.

The satellite ground station consists of transmitting and receiving antenna, transmitting TNC, receiving TNC, transmitter, receiver, satellite tracking software, i.e., predicting the orbit of the satellite and directing the antenna towards the satellite, up-link software, downlink software, and database software which stores information such as uplinked and analyzed data.

Thus, in case of a satellite, a system that enables the antenna to automatically track the satellite, based on orbit prediction and Doppler shift calculation, as well as a system to compile large amount of data so that information are available where and when needed, are essential. In addition, in the case of a Low Earth Orbit (LEO) satellite, the satellite can communicate with the ground station only 10 min or so at one or six times per day. The most important thing is to develop a concrete operational plan beforehand, set up a ground station based on communication speed and communication intensity, and assure the effectiveness by experiments under similar environment. Since CanSat receives data which determine the mission success within a limited amount of time, it is a good training for real LEO satellite operation.

5 | Advance Verification

Usually, the satellite ground test is conducted after development of the Engineering Model (EM) in accordance with the basic design. Based on the results of the ground test, the basic design is modified, and a detailed design is then prepared. Next, the Flight Model (FM) is developed, and the ground test is conducted again for final confirmation. Thus, advance verification refers to the work to confirm whether the details of the design meet the given requirements and whether the design provides the functions necessary for carrying out the mission. Final modifications are accordingly made.

In the case of CanSat, the preference is to follow the same procedure as the satellite (basic design, EM development, verification, detailed design, FM development, verification, and modification) as closely as possible. At the very least, the step of FM verification and modification should not be omitted. If the verification and modification is not performed properly at this stage, a failure is likely to occur in actual flight and it will be difficult to determine the cause and implement countermeasures.

5.1 Purpose of Advance Verification

The two main purposes of advance verification are as follows:

1) To verify that the design and development of the satellite are acceptable. To confirm that the satellite operates as expected.
2) To assure the performance and reliability as well as theoretical analysis of the satellite system to a convincing level. In other words, to decrease the risk of mission failure to an acceptable level under the various uncertainties of the launch environment and operational environment and under the restrictions of cost and the schedule which is commonly known as "As Low as Reasonably Practical (ALRP)".

Generally, performance and reliability, schedule, and the cost of the satellite frequently result in conflicting requirements, as shown in Figure 5-1. It is important to make a trade-off among the factors while properly conducting the verification to a level that provides satisfaction and acceptance from the standpoint of the satellite user.

After the design is completed and before actual development is started, all necessary verification items should be listed. Using this list as a starting point, items recognized at the development stage are added. The list allows the verification work to be performed as soon as development is completed. If there is no list and the verification items are considered after development, which may mean that some items might be missed.

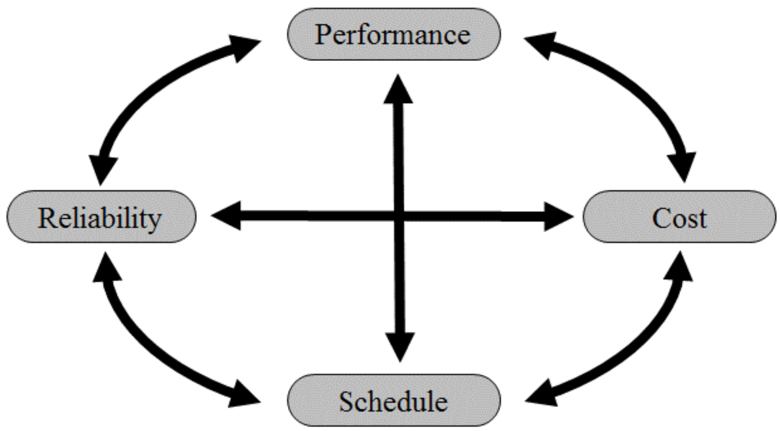


Figure 5-1: Trade-off in advance verification.

5.2 Content and Method of Advance Verification

The content of advance verification is determined in consideration of launch environment and operational environment. Obviously, the method of advance verification depends on the environment. In the CanSat field experiment, for example, the method of advance verification when the release is performed after being launched to an altitude of several kilometers via a model rocket is usually different from when the release is performed from an altitude of approximately 100 to 150 m via a tethered balloon. However, if CanSat is regarded as training for satellite development, then there is an opinion that the launch environment and operational environment for satellites are considered, such as training in advance verification of the satellite although the verification is not necessary at that time. In any case, it is important to conduct the advance verification with a picture of the assumed environment as much as possible.

The content and method of advance verification is described in this chapter for using the release via rocket launch and tethered balloon. The main verification items are listed in Table 5-1. The basic pattern is that the design is based on the results from a theoretical or numerical analysis and then the design is verified in the test.

The test may require facilities, and there are many cases where the execution of the verification requires some amount of costs, such as a rental charge for the facilities. There is also a case where test facilities are unable to be ensured within a defined development period. From the viewpoint of cost effectiveness, it is necessary to consider and determine the verification through an analysis without the test in some cases. This holds true for satellites and CanSat. Especially in the case of the satellite, the test often cannot be conducted and the verification relies on the analysis because it is impossible to completely simulate the space environment on the ground. In the case of CanSat, intrinsically the mark "○" in Table 5-1 means that the verification method shall be conducted, but the mark "△" means that the verification method may be conducted.

Table 5-1: Main verification items of CanSat flights.

Verification item		Flight method		Verification method	
Classification	**Verification content**	**Rocket**	**Balloon**	**Analysis**	**Test**
Durability to launch environment	Durability to launch acceleration (Plastic deformation, breaking, and buckling shall not occur.)	○	-	△	○
	Durability to launch vibration (Plastic deformation, breaking, and buckling shall not occur; fastening sections such as screws shall not be loosened; the system shall operate without problems after the vibration starts.)	○	-	△	○
	Durability to separation impact (Plastic deformation and breaking shall not occur, mechanical and fastening sections shall not come unfastened, and the system shall operate without problem after impact.)	○	△	△	○
	The power shall be turned on at the appropriate time (in the case of a cold launch).	○	○		○
Durability to operational environment	The parachute shall open smoothly without becoming entangled.	○	○		○
	Falling with a stable condition	○	○		○
	Falling at the appropriate speed	○	○		○
Reliability of electrical power supply	Necessary electrical power shall be ensured from power on to the end of the mission.	○	○		○
Reliability of communication system	Communication lines shall be ensured for the assumed flight environment.	○	○		○
Simulated operation	The system shall operate in the designed sequence from power on to the end of the mission.	○	○		○

The concrete method of advance verification is described below. It is understood that the test needs to be conducted safely in the verification method. For example, a test that verifies falling velocity or the stability of the falling attitude of CanSat with a parachute mounted often employs methods of throwing CanSat from the rooftop of a building. In this case, it is necessary to consider the safety of testers and to exercise ingenuity in preventing CanSat from causing any damaging effect.

Such considerations and ingenuity may be insufficient because of the simplicity of the test or lack of preparation time. However, to succeed in CanSat development, it is important to properly perform everything within the possible capability instead of feeling that this might be enough. The test must be conducted while bearing in mind that a failure can occur if the test is not conducted.

5.2.1 **Durability to Launch Environment**

Table 5-1 lists four items as verification of the durability to the launch environment. Those items are listed for CanSat, and the other items, such as durability to rapid decompression, are required for satellites. Moving upward with a rocket, a satellite will be exposed to the vacuum state within a few moments from the atmospheric pressure on Earth. If the satellite has a place where air is likely to build up, the air cannot escape within a short time and expands to apply pressure on the peripheral devices, which affects those devices. Moreover, some other items, such as fatigue of materials and the temperature environment in fairing (cover of rocket tipping where a satellite is incorporated inside) should be considered. Generally, those items are often presented by the rocket development side. For example, in the case of the H-IIA piggyback, Japan Aerospace Exploration Agency (JAXA) defines the items in the user's manual for piggyback satellite developers. Satellite developers shall execute the design in consideration of them and shall conduct verification before a flight. In the case of CanSat, the items are determined with a picture of the flight, but at least four verification items listed in Table 5-1 should be considered. Those four verification items are explained in the following subsection.

5.2.1.1 Acceleration Load

In the verification of durability to launch acceleration, it is common to use the method for applying the acceleration load to a specimen by utilizing centrifugal force with the specimen mounted on the tip of the rotational arm. The same method is also used for satellites. In CanSat, for example, in the case of CanSat developed for ARLISS event, it is defined that the design shall be executed in expectation of an acceleration of approximately 10 G based on past launch data. It is ideal to conduct the acceleration test, for example, with a rotational device. However, the easiest way would be to add a weight equivalent to the assumed acceleration on the top face (rocket tipping side) of CanSat because making a device entails costs. Although this method of simulating the acceleration load, which is a distributed load in reality, on the top face of CanSat as a concentrated load will apply excessive load on the top face, it is worth performing because the load path is checked.

The acceleration should not present a problem in the FM, except for some extraordinary circumstances in which the stress applied to members is an allowable value or less and the whole system is designed to receive less than buckling load. CanSat is compact and if each member is not extremely slim or thin or if a material with a low allowable stress is not in use, an acceleration of approximately 10 G does not become a load that causes buckling or plastic deformation. In fact, if the materials and dimensions are defined in consideration of the strength of each member, a sufficient margin is ensured, and the FEM analysis of the overall system would not need to be conducted. The relationship between size and strength can be understood in formulas from the theory of material mechanics, elastic mechanics, and structural mechanics. The real conduct of experiments would bring actual feeling of relationship between formulas and hardware.

5.2.1.2 Vibration Load

Because acceleration caused by the upward movement and rocket random vibration are transmitted at the time of the rocket launch, the durability to vibration shall be verified. It is common to use the verification method for random vibration tests with a vibration exciter. Figure 5-2 shows an example of the vibration exciter. There are various ranges from large in which relatively large table is vibrated to small, and the small one is enough for CanSat. However, because a jig to mount CanSat onto the vibration exciter is necessary, the specifications of the vibration exciter and the jig should be determined. The small vibration exciters are usually available in engineering colleges.

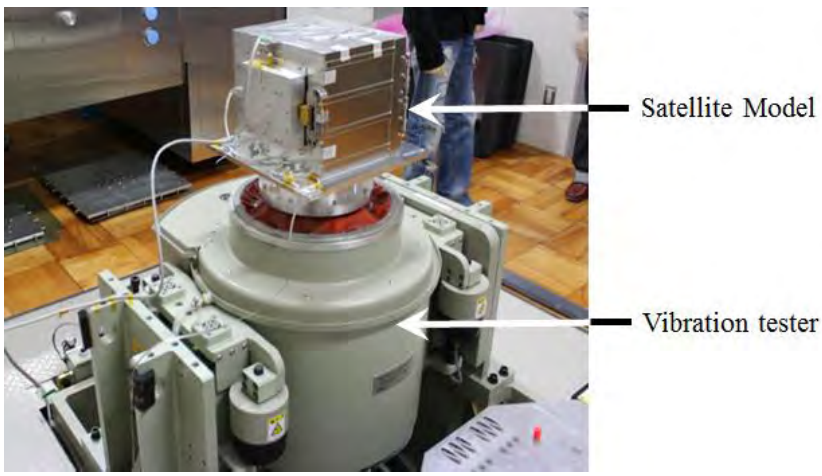


Figure 5-2: Example of vibration exciter.

The test is actually conducted in accordance with specific values of acceleration density depending on frequency, duration of action, effective values which are related to random vibrations required from the satellite's launcher. The required values are different between the EM and FM, and the EM requires durability at the Qualification Test (QT) level, and FM requires durability at the Acceptance Test (AT) level. The QT level is often more severe than the actual launch environment. Figure 5-3 shows an example of a random vibration test level and Figure 5-4 is an example of the result of an actual satellite test.

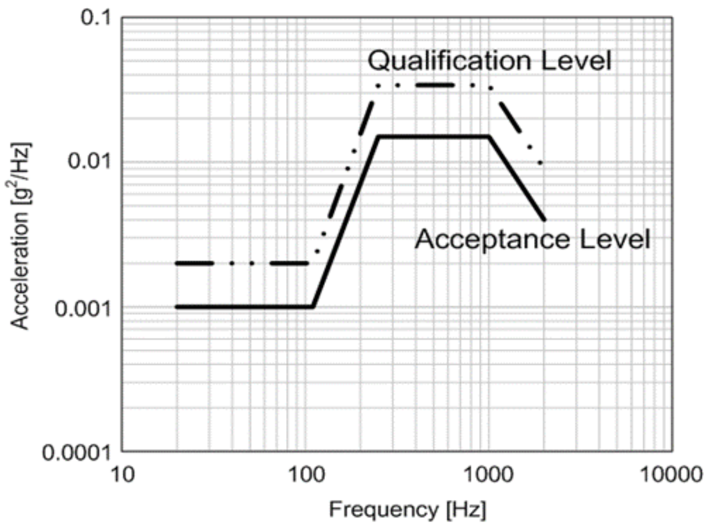


Figure 5-3: Input level to vibration exciter.

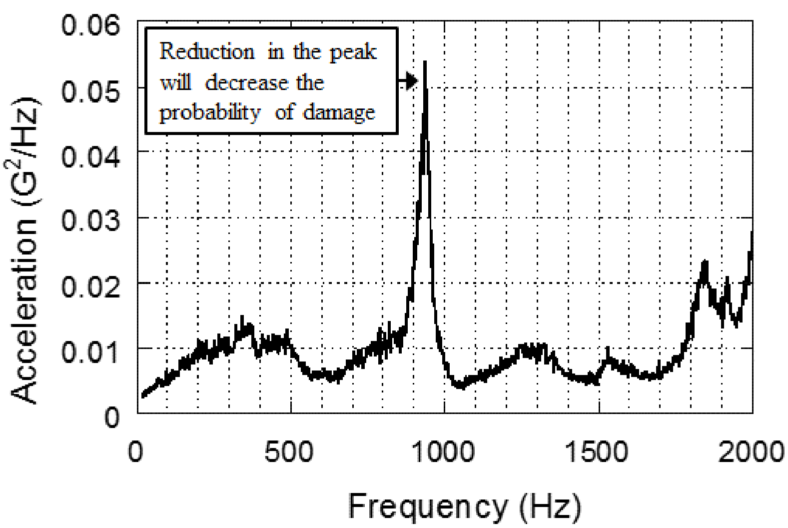


Figure 5-4: Vibration test results of a satellite.

In the case of CanSat, for example, it is defined that CanSat developed for ARLISS shall withstand random vibrations with an effective value of 25 G (25 G_{rms}). The following three points shall be checked in the verification of the CanSat vibration load.

1) Vibrations shall not cause plastic deformation and breaking.
2) Fastening sections such as screws shall not be loosened.
3) The system shall operate properly even after vibrations are applied.

If it is difficult to arrange the vibration exciter, an idea might be that the vibration test is not conducted for (1) and only a suitable equivalent static load, refer to section 4.7, and the corresponding static acceleration load test are conducted. As for (2), if the vibration time is short (at longest approximately 10 seconds even for ARLISS launch), it may be determined that the fastening sections are not loosened by being thread ably mounted with a sufficient fastening force via spring washers attached to it. However, if the assembly is performed, it is often the case that minor levels of mistakes are found, such as a section more sensitive to vibration than expected, a loosened screw, or excessive vibration of a harness owing to failure of the fastening. A part with the design slightly changed from the previous CanSat's design may cause a problem in an actual flight, especially only when it was decided not to conduct the test because of a design like the previous CanSat with the previous results accepted without question. It is strongly recommended to conduct the vibration test on a priority basis.

As for (3), identifying a defect portion when the system does not operate properly becomes a problem. Possible causes include soldering defect on electronic substrate, contact failure of connector, and breaking of the crimping portion of harness. Generally, the causes of problems are listed in advance and are checked before proceeding to the vibration test. Because these three points are unable to be discovered in the check before the test, failure to carefully check them will result in a lot of trouble identifying a defect after the test. In fact, these three points do not cause a problem in operation, and often cause problem in the vibration test. For example, even if there is a soldering defect or a contact failure, there is an electrical continuity barely before the test, and the test often causes the solder to float or causes the contact portion of the connector to disconnect, which cuts off the continuity. In satellites, if a problem occurs in orbit, Fault Tree Analysis (FTA) is conducted to identify the cause of the problem. In the case of CanSat, it is also recommended to conduct FTA as training in the stage of advance verification. For basic theories relating to vibration analysis and vibration test, it might be better to refer to special reference about FTA.

5.2.1.3 Separation Impact

Generally, when something separates, an impact occurs. For example, in the case of a satellite, when the rocket fairing opens, an impact occurs and affects the satellite via the interface section between the rocket and satellite. Also, when the satellite separates from the rocket, an impact obviously occurs to the satellite.

In the same way for CanSat, when released from a rocket or the carrier of the balloon, an impact occurs. The impact is a negligibly small force for the balloon while it is an unexpectedly major impact for the ARLISS model rocket. The problem of strength to withstand the impact is the bracket for mounting the parachute line to CanSat, as shown in Figure 5-5. If the strength of the bracket is insufficient, the bracket may be plastically deformed, which may unfasten the parachute line. Alternatively, if the strength of the outer plate on the CanSat body side is insufficient, the outer plate may break, which may unfasten the parachute line together with the bracket. Furthermore, some sort of mechanism, such as a gear, may be unfastened. It is necessary to conduct the impact test to check that no problem occurs at these parts.

In the impact test, the simple and reliable method would be to have an impact on CanSat by suspending CanSat, fixing a weight equivalent to the impact on the underneath of CanSat, and then losing hold of the weight. In the impact test, check that no problem occurs at the bracket for mounting the parachute line and the outer plate on the CanSat body, no plastic deformation or damage occurs anywhere including other portions, and the system operates without problem after the impact test.

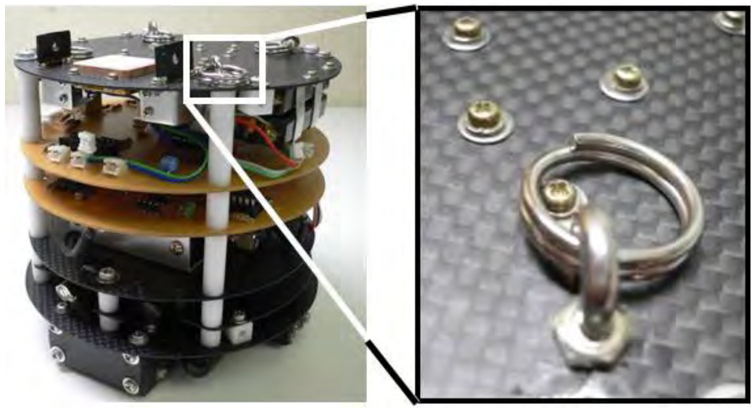


Figure 5-5: Parachute mounting section.

5.2.1.4 Power-up

In ARLISS, to avoid interference with the rocket radio waves, it is defined that CanSat shall not emit radio waves until released from the rocket. To meet this requirement, select one of the following three methods.

1) The power of CanSat is switched off until the release from the rocket (cold launch).
2) The power of the radio is switched off until the release from the rocket.
3) With the power switched on, the radio starts transmission when the release from the rocket is detected.

Because (1) takes long time to obtain the GPS data, it is recommended to select (2) or (3). In each case, the function to detect when CanSat is released from the rocket is necessary. A simple example of the detection function includes a method for using a switch or a flight pin as described in Chapter 4 | . For example, a switch that becomes electrically continuous when the pin is pulled out is inserted between the radio and the power supply so that the radio can be automatically turned on when the pin is pulled out. Alternatively, the same switch is inserted between the power line and the OBC signal input pin so that the OBC input pin becomes at high level (H) when the pin is pulled out, which is detected by the OBC, and the OBC starts sending the data to the radio.

There are other possible arrangements for pulling out the flight pin. For example, a short string is attached to one end of the pin and the other end of it is coupled to the upper side of the parachute line (proximate region of parachute). When the parachute is stored, the line is folded and is very close to the CanSat body. And when the parachute releases from the rocket, the parachute opens and the line becomes straight, which increases the distance to the CanSat body. At this point, the string attached to the parachute line pulls out the pin from the CanSat body. Thus, the release from the rocket can be detected. This is an example, and other various methods would be considered. The detection function that is suitable to your experimental environment and is easily applied onto CanSat and should be considered.

5.2.2 **Durability to Operational Environment**

For satellites, the durability to the space environment (e.g., durability to radiation, durability to thermal environment, durability to vacuum environment) shall be verified in a test or analysis. In the case of CanSat, it would not necessary to be anxious about this because it is not especially exposed to a severe environment when falling with the parachute from a certain height.

In the case of CanSat, the parachute must open properly when released from the rocket or the balloon carrier. If the parachute is folded improperly, it may not fully open or may not open at all. Also, in the case of CanSat that performs fly-back, a parafoil is often used. The parafoil must be folded with greater care than the parachute, as shown in Figure 5-6. If ingenuity is not exercised in the folding method, the parafoil will not open properly, which often causes free fall or flight in a deformed shape, and will fail to provide the expected performance.

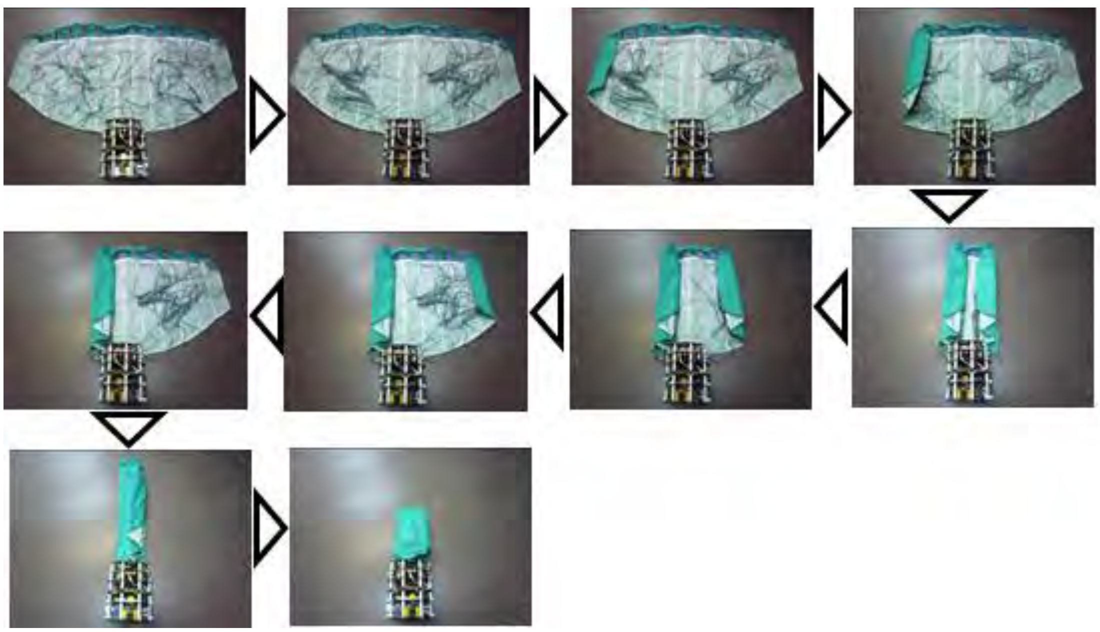


Figure 5-6: Example of folding of parafoil.

Because the placement of the line coupling the CanSat body and parachute should be designed so that the dropping velocity may not increase certain value from a safety standpoint, it is necessary to check in the test whether it drops at the expected velocity, as shown in Figure 5-7. When using a parafoil, the flight performance itself becomes important. It is recommended to conduct the drop test in a gymnastic hall and others and check the dropping velocity and flight performance under calm conditions to check the validity of your design before making experiments outdoors with wind.

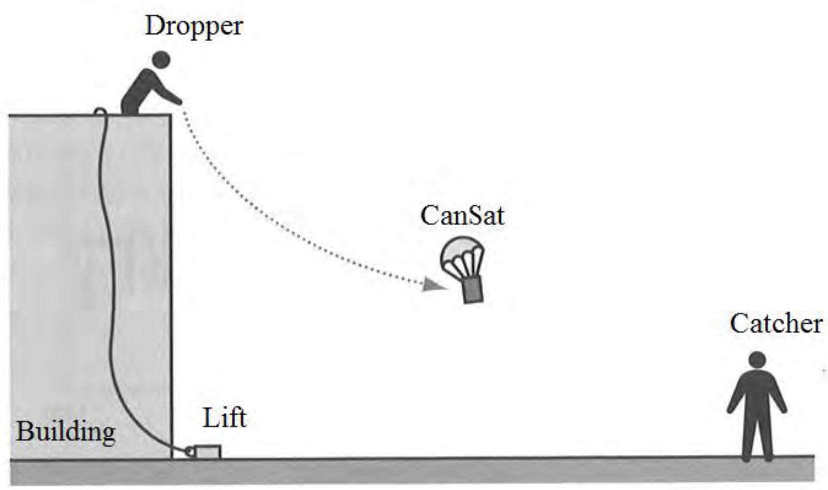


Figure 5-7: Example of parachute velocity test.

5.2.3 **Ensuring of Electrical Power**

Typically, a satellite is not equipped with sufficient batteries and solar cells because of its limited mass and volume. Therefore, whether the electrical power is sufficient shall be considered properly from the development stage. After the satellite is developed, the power consumption and remaining battery level are measured via the operation test with a simulating the operation mode to verify whether their measurements are the designed values. This power test is insufficient if it is conducted at normal temperature, and shall be conducted at the lower temperature limit and the upper temperature limit if possible. This is because the performance of the battery drastically varies depending on the temperature. In the case of the satellite, the satellite temperature differs between in sunlight and in shade. Therefore, the relationship among temperature, power consumption, and remaining battery level must be well understood in the test.

In the case of CanSat, the test must be conducted only at normal temperature. This is because the battery power barely meets the necessary level if the power consumption of the mission device is large or the number of batteries is kept to the minimum owing to the restricted mass value.

Figure 5-8 shows an example of power test results for CanSat which indicates the measurement results of power consumption that changes every moment, both the battery voltage and current are measured. Figure 5-9 indicates the measurement results of battery voltage. In these figures, the CanSat performed the 30-minute mission which decreased the power consumption in 30 minutes. Because the relationship between the battery voltage and the remaining battery level is described in the battery data sheet, the remaining battery level can be evaluated.

There is one caution in the power test which is the operation time. In both cases of rocket and balloon, the operation does not start as soon as the battery is loaded onto CanSat. In the case of a rocket, there is a launch standby time between when CanSat is loaded onto the rocket and when the rocket is launched. It may take approximately 2 hours for in ARLISS. In the case of a balloon, it may take approximately 20 minutes between loading onto the carrier and releasing after moving upward in the air.

In the case of a mission where the power must be on after loading, it is necessary to conduct the power test in consideration of the time between the loading and the release.

Figure 5-10 shows an example of the power test for CanSat that performs the fly-back mission. There are two systems for the battery in this example, one is to operate the bus system and the other is for the servomotor to control the parafoil. The graph shows the measurement results of the transition of battery voltage under the conditions of 3 hours for standby before launch, 1 hour for the mission, and the transmitter operates for about 2 hours after the mission starts. During the test, CanSat is turned upside down, and the servomotor undergoes the load with a weight instead of the parafoil connected. In this way, the test shall be conducted on the assumption of an actual flight and in consideration of a certain margin. Checking that the power consumption can be ensured under the similar flight conditions allows the flight to proceed safely.

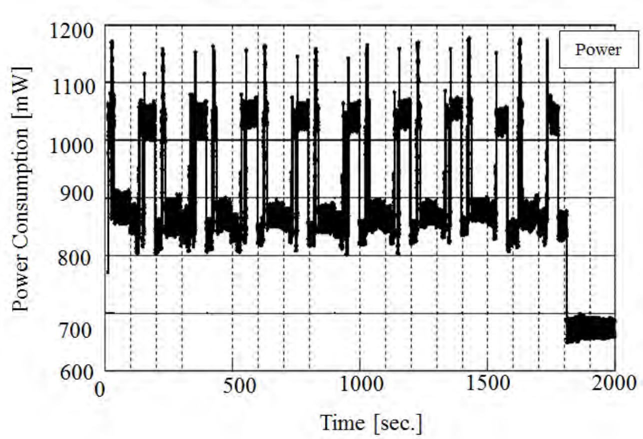


Figure 5-8: Temporal variation in power consumption.

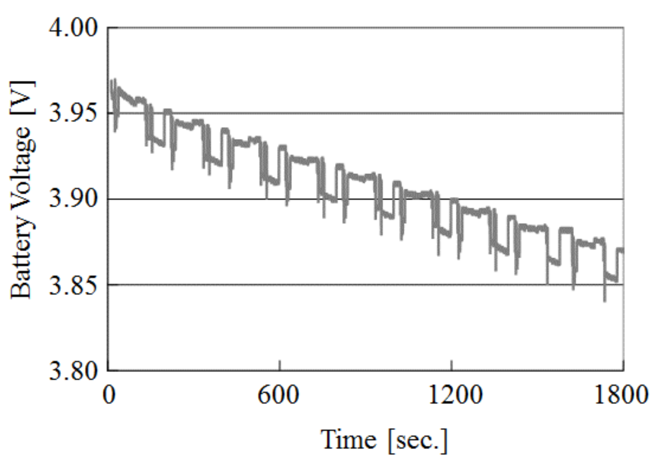


Figure 5-9: Temporal variation in battery voltage.

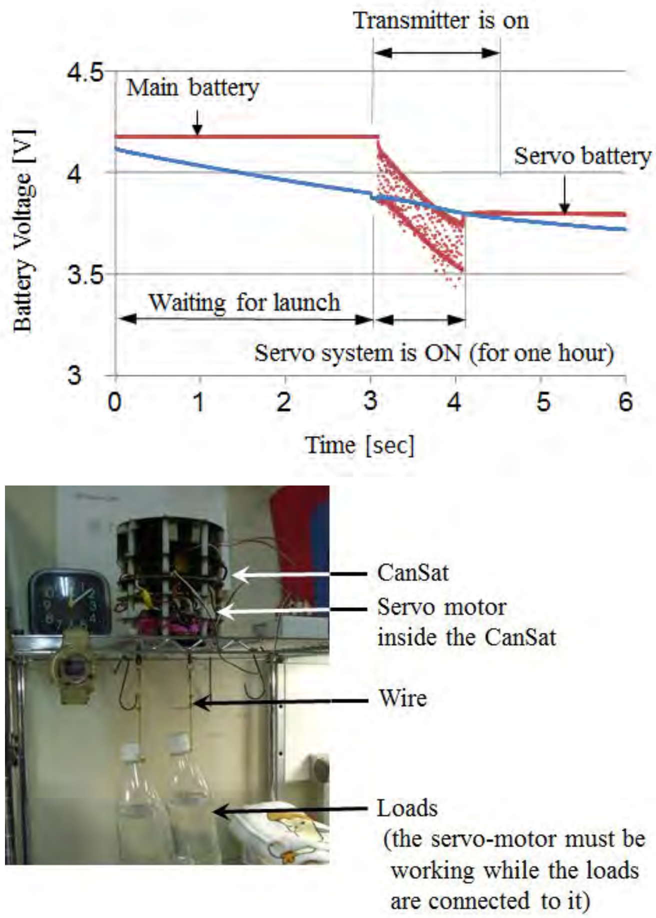


Figure 5-10: Example of CanSat electric power test in which fly-back mission is performed.

5.2.4 **Ensuring of Communication**

It is easily comprehensible to conduct the long-range communication test in the verification of communication subsystem. In the case of a satellite, the wireless license is applied for the satellite station and ground station. When the provisional license is obtained, the satellite is moved to a place away from the ground station, and communication is performed with the ground station to calculate the acquisition rate or decoding rate. Slightly less acquisition rate, for example 98%, can be accepted in consideration of an influence of buildings and whether the calculated value is the measured or more can also be checked.

In the case of CanSat, specific small power radio requiring no radio license is often used. The communication test should be conducted in a place of where the line-of-sight length is several hundreds of meters to 1 km, within the radio reachable distance. In that case, it is desired to place CanSat above the ground station to suppress the influence of radio reflection from the ground.

When the communication is conducted with the PC for CanSat or ground station held by human hands, the radio wave is more greatly weakened than that on the ground or in the air because both PC and the ground station are at approximately 1 meter from the ground. As a rule of thumb, the reachable distance would be reduced by half. The problem is that if the reachable distance is shorter than the design value, and identification is performed whether its cause is the influence of ground or mistake in design or development of communication subsystem. It is recommended to conduct the communication test under the distance that is short at first and then is lengthened gradually, and to check how the acquisition rate decreases as well as to find a mistake in design and development. In fact, if there is no radio interference from the surroundings, the acquisition rate at a short distance of 100 meters or 200 meters should be 100%. If it is not 100%, there may be mistake in design or development.

5.2.5 Simulated Operation

The most important thing in the advance verification is the operation test in accordance with the mission sequence. This holds true with both the satellite and the CanSat. The purpose of the operation test is to verify that the system has the functions to fulfill the mission. However, it is difficult to simulate the operation completely because simulating an actual flight environment is impossible. For example, in the case of CanSat with the mission of fly-back, it is difficult to verify the fly-back function unless an actual flight is performed. Therefore, it is important in the operation test to properly consider what shall be verified to what extent under what condition.

For example, just performing the operation test in an experimental laboratory allows problems in the electric system, for example circuits and software, to be discovered and allows defects, inadequacies, items to be improved and to be identified, which greatly facilitates the flight. In that sense, the operation test would be the most important advance verification.

The important thing in the operation test is not to overlook any minor matters. In any cases not limited to the operation test, determining the procedure and checking items before test and creating a test plan document will avoid getting confused in the actual test. Because there are many checking items especially in the operation test, discussions should be conducted among members and checking items should be listed before the test.

6 | **Field Test**

When the development and advance verification of CanSat are complete, finally, the next phase is to conduct the actual field test. Although this is an experiment conducted on the ground. It will be introduced as a complete single experiment. The field test itself should be prepared properly and the entire experiment should be succeeded safely. This chapter mainly explains the plan and preparation of the field test.

6.1 Overview of Field Test

Provided that the purpose of CanSat project is regarded as the simulation of an artificial satellite, the field test requires the following seven conditions:

1) Capable of simulating the rocket launch environment (acceleration, vibration, and impact environment).
2) Capable of simulating the cold launch and separation detection (simulation of piggyback satellite).
3) Capable of simulating the remote control via data uplink and downlink.
4) Capable of simulating the microgravity environment.
5) Capable of simulating the vacuum environment.
6) Capable of simulating the temperature environment.
7) Capable of simulating the other space environments (e.g., radiation environment, variation of shade and sunlight).

It is not easy to ensure the experiment methods that meet all these conditions while assuring safety. It might be extremely difficult to achieve (5) to (7). Regarding (4), it may be possible to simulate it to some degree if a free fall is performed from a certain height. However, it is necessary to incorporate the system that allows the parachute to be reliably opened immediately before arrival on the ground in a safe landing so that CanSat will not break due to the impact when dropping to the ground. Considering such damaging situation, the opening of the parachute requires very high reliability, and this method is practically not recommended for microgravity experiments. Considering such limitations, the following two methods have been formulated as CanSat field tests.

6.1.1 **Rocket Launch**

CanSat is launched by a small rocket, such as a model rocket and hybrid rocket, and is released in the upper air. CanSat automatically opens a parachute, drops at a constant rate, and performs the mission during the time of landing to the ground. CanSat can continue the mission after landing on the ground such as in the case of rover-type CanSat.

6.1.2 **Balloon Launch**

After the carrier is installed onto the balloon and CanSat is loaded, the balloon is flown to an appropriate height and tethered, and then the lid of the carrier is opened. This allows CanSat to drop and automatically open the parachute. Subsequently, in the same way as the rocket launch, it performs the mission during dropping or after landing.

The experiment with a drone instead of a rocket and balloon is being considered. The drone is relatively expensive and requires some degree of skills for a safe flight. In recent days, however, a relatively inexpensive drone that can be equipped with a payload of approximately 1.2 kg has been developed, which is likely to be used for the CanSat field test.

In the case of a rocket launch, it is necessary to appropriately set the rocket maximum height and the CanSat separation height depending on the size of the experiment site. If the rocket or CanSat is moved up too high and is flown out of the site, it cannot be said that such an experiment is safe even if it drops slowly with a parachute. Similarly, in the case of the use of a balloon or drone, extreme care should be taken for the CanSat separation height. If safety is not assured, the field test should not be conducted. Should a serious accident occur when the field test is conducted without considering safety, the perception becomes one where the CanSat field test is perceived as dangerous. This may result in the creation of an environment where it is hard for other people to conduct CanSat field tests. It should be kept in mind that safe field tests are conducted in the sense that the CanSat community should be respected.

The following sections detail a field test using a small rocket and a field test using a tethered balloon, which are conducted in large numbers at the present time. The small rocket experiment allows a simulation at nearly the same level as the artificial satellite launching as shown in Table 6-1. The item of microgravity environment is shown as “△” in Table 6-1, which means it is possible to some degree if the rocket altitude is held at a sufficient height and free-fall time is set. In contrast, the balloon experiment can simulate only two items in the launch and operation environments. If this assumed to be insufficient, other experiment methods for the rocket experiment might be considered. In the author opinion, the balloon experiment is regarded as an advance experiment for the rocket experiment or as an experimental method for beginners. If a full-scale field test is desired, the recommendation is to conduct the rocket experiment in which a sufficient height as ARLISS (which means a sufficient experiment duration) is provided.

Table 6-1: Simulation of launch and operation with balloon experiment.

No.	Item Description	Rocket experiment	Balloon experiment
1	Capable of simulating the rocket launch environment (acceleration, vibration, and impact environment)	○	×
2	Capable of simulating the cold launch and separation detection in simulating a piggyback satellite	○	○
3	Capable of simulating the remote control via data downlink and uplink over the radio	○	○
4	Capable of simulating the microgravity environment	△	×
5	Capable of simulating the vacuum environment	×	×
6	Capable of simulating the temperature environment	×	×
7	Capable of simulating the other space environments (e.g., radiation environment, variation of shade and sunlight)	×	×

6.2 Rocket Experiments

This section introduces field experiments using rockets. Rocket can be developed or take the opportunity to participate in rocket launch experiments offered by other organizations. The description below is mainly about the available vehicles and opportunities. No details about rocket technologies are given.

6.2.1 Model Rocket (Solid Propellant)

Model rockets were originally developed in the U.S. and are widely used as a tool in space education aimed for young students. The Japan Association of Rocketry (JAR) began licensing model rockets in 1995, to ensure their safe operation. No accidents resulting in either injury or death have occurred with model rockets in their 50-year history, making them an extremely safe educational tool. Model rocket engines are classified according to the total impulse [N.s], which is the amount of energy produced by their engine. The total impulse produced by a Class-A engine is between 1.26 N.s and 2.50 N.s, while a Class-B produces about twice as much energy, at between 2.51 N.s and 5.00 N.s. Table 6-2 lists the model rocket engine classes and their total impulses.

Table 6-2: Classes and total impulses of model rocket engines which can be used in Japan.

<table>
<tr><th>Class</th><th>Total Impulse [N.s.]</th><th colspan="4">License</th></tr>
<tr><td>A</td><td>1.26 ~ 2.50</td><td rowspan="3">Class 4</td><td rowspan="7">Class 3</td><td rowspan="8">Class 2</td><td rowspan="10">Class 1</td></tr>
<tr><td>B</td><td>2.51 ~ 5.00</td></tr>
<tr><td>C</td><td>5.01~ 10.00</td></tr>
<tr><td>D</td><td>10.01 ~ 20.00</td><td></td></tr>
<tr><td>E</td><td>20.01 ~ 40.00</td><td></td></tr>
<tr><td>F</td><td>40.01 ~ 80.00</td><td></td></tr>
<tr><td>G</td><td>80.01 ~ 160.00</td><td></td></tr>
<tr><td>H</td><td>160.01 ~ 320.00</td><td></td><td></td></tr>
<tr><td>I</td><td>320.01 ~ 640.00</td><td></td><td></td><td></td></tr>
<tr><td>J</td><td>640.01 ~ 1280.00</td><td></td><td></td><td></td></tr>
</table>

A license is required to operate a model rocket, with the classification of the license being determined by the engine class. A Class-4 license allows the license holder to operate a rocket of up to Class C, with up to 20 g of grain. A Class-3 license allows the holder to operate up to a Class-G rocket. However, given that the Class-G rocket uses more than 20 g of grain, under Japanese law it is necessary to seek approval from the local authority prior to purchase and operation. At least a Class-D rocket is required to safely launch a CanSat with a weight of about 500 g. Therefore, CanSat launches using a model rocket require a Class-3 license. The JAR website explains how to acquire a license, as well as the license fees (http://www.ja-r.net/).

The model rocket engine uses three types of grain. The first is the solid propellant which generates the thrust. The second is a delay charge which generates tracking smoke to allow observers to follow the rocket's trajectory. The third type is an ejection charge for ejecting the recovery system, such as a parachute, from inside the body of the rocket. A mechanical recovery system can be fitted to engines larger than Class H. However, rockets smaller than Class G would become excessively heavy if a mechanical system is to be installed. Therefore, it is preferable to use a simpler system such as an ejection charge. Figure 6-1 shows the SCR-002, a model rocket capable of carrying a CanSat and equipped with a simple ejection system, designed by the student rocket committee of UNISEC.

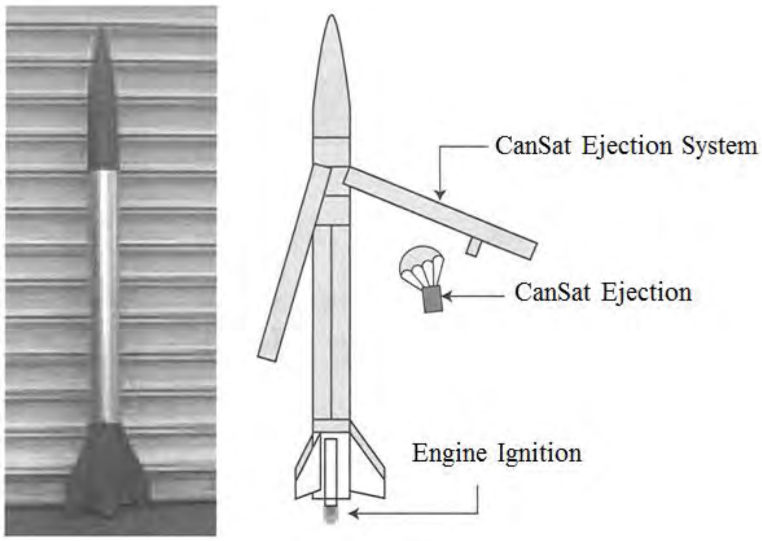


Figure 6-1: SCR-002 rocket developed by student rocket committee of UNISEC.

The SCR-002 rocket uses a G80-4T engine, which can be operated with a Class-3 license. This rocket can carry a CanSat weighing up to 250 g, to a height of 120 m. The ejection delay ignites about 4 s after the end of propellant combustion, and burns through a string which fixes part of the rocket body in position, thus triggering the CanSat ejection system. The system opens the rocket body such that the CanSat is ejected at the apogee altitude.

Figure 6-2 shows a schematic view of a model rocket powered by a Class-H engine, developed by JAR. The rocket is 1.3 m in length, and can carry a CanSat weighing 1 kg to an altitude of 300 m to 350 m. The ejection delay activates a piston inside the rocket body, which pushes on a hook which unlocks the ejection door. The door opens and the CanSat is ejected. The rocket uses an H220-M Class-H engine, and therefore requires a Class-2 license to operate.

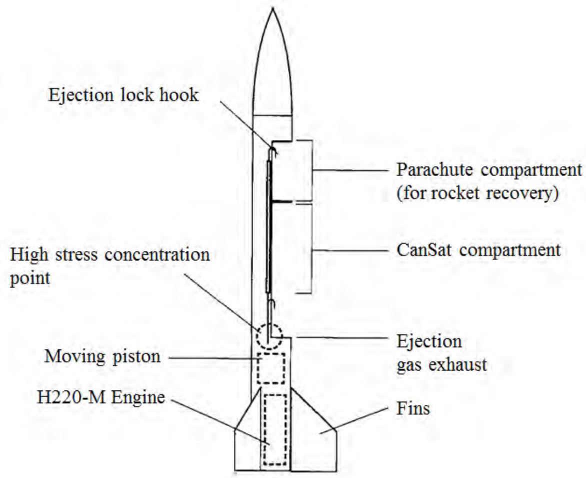


Figure 6-2: JAR-developed model rocket for CanSat launches, powered by Class-H engine.

The SCR-005 is a larger model rocket, developed by the student rocket committee of UNISEC, can launch an open-class CanSat weighing 1.5 kg to a height of 3 to 4 km, as shown in Figure 6-3. The SCR-005 has been successfully launched by ARLISS (refer to section 6.3.2). The rocket uses a K-700W engine which is imported from U.S. The rocket body is subjected to huge stresses when the rocket is launched, because the Class-K engine generates a very large thrust. This makes it difficult to design an ejection system whereby the body divides, or which features a large ejection door. Therefore, the CanSat and recovery parachute are ejected from inside the rocket by a piston operated by gas pressure after the ejection delay.

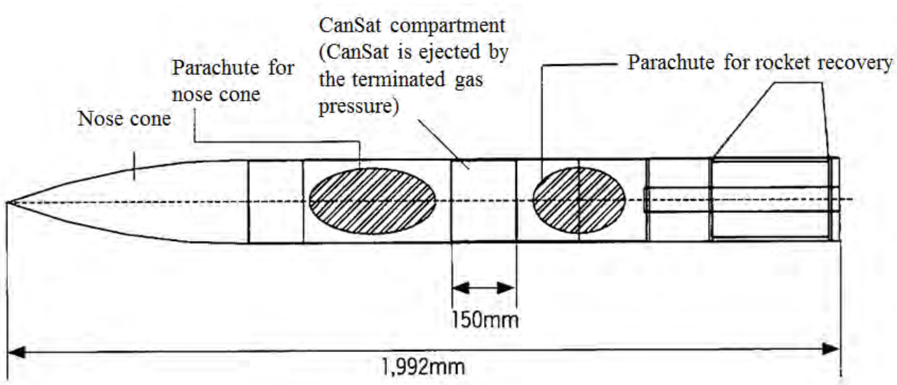


Figure 6-3: Schematic view of SRC-005 rocket.

It is completely feasible to launch a CanSat using a model rocket like that described above. The creation of such a large-scale rocket, however, requires not only a knowledge of rocketry, but also proficiency with machine design and manufacturing. Rather, beginners are recommended to start with the design of a small rocket. Launch experience is very important for ensuring safe rocket development and launching. Moreover, the student rocket committee of UNISEC has produced the "Open-Class

CanSat Launch Rocket Manual." The manual describes the design and detailed manufacturing method of the SCR- 005 large-scale model rocket. Although this manual has not been published, any developer wishing to read it is advised to contact the UNISEC secretariat.

6.2.2 **Hybrid Rocket (HyperTEK)**

Hybrid rockets are a type of chemical rocket propulsion system; they usually employ a liquid or gas oxidizer and a solid fuel. A hybrid rocket offers several benefits: throttlability, environmental friendliness, simplicity, and low cost. Some of these benefits would allow a hybrid rocket to be an excellent educational tool. Launch experiments with a hybrid rocket have been conducted at some universities in Japan. Many of these university hybrid rockets have used HyperTEK engines, Shown in Figure 6-4 made by Cesaroni Technology Inc. of Canada. The HyperTEK engine uses Nitrous Oxide (N_2O) as an oxidizer and ABS resin as fuel. The oxidizer is supplied by being blown down by the high steam pressure generated by the N_2O, such that the HyperTEK engine does not need any kind of pressurization mechanism. A spark wire is used for ignition, rather than explosives. Thus, the HyperTEK engine can produce a large thrust comparatively safely, making it ideally suited to space education and university research activities.

Figure 6-4: HyperTEK hybrid rocket engine.

The different types of HyperTEK engine are classified by their total impulse, in the same way as the model rocket engines described above. The HyperTEK engine classes run from Type I to Type M. Details of the HyperTEK engine can be found on the Cesaroni Technology homepage (http://www.hypertekhybrids.com/). A HyperTEK manual can also be accessed from the company's homepage. Almost every university that has launched a hybrid rocket is a member of UNISEC. UNISEC is responsible for setting and applying safety standards. Therefore, a considerable amount of information related to the safety of rocket launching is shared among the members of UNISEC.

6.2.3 **CAMUI-Type Hybrid Rockets**

The Laboratory of Space Systems at Hokkaido University has been developing CAMUI-type hybrid rockets for over a decade. CAMUI-type hybrid rockets employ a combination of Polyethylene and liquid oxygen as propellants. The key concept of these rockets is to enhance heat transfer to the solid fuel which is Polyethylene by using impinging jets of hot combustion gas on multiple burning surfaces. Figure 6-5 shows a schematic view of a CAMUI-type fuel grain. It consists of several cylindrical fuel blocks. Each fuel block has two ports in the axial direction. The staggered orientation of axial ports produces impinging jets on each upstream end face, resulting in enhanced heat transfer and high combustion efficiency.

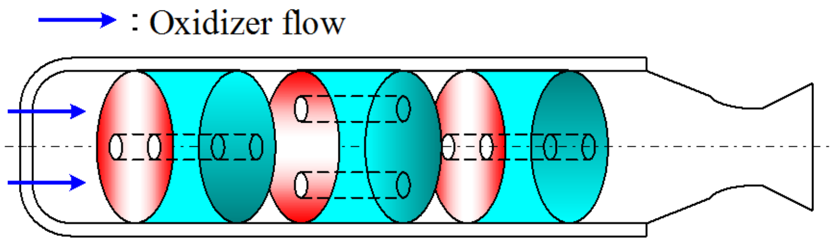


Figure 6-5: Schematic View of a CAMUI-type fuel grain.

The first launch experiment of a CAMUI type hybrid rocket was in March 2002. This was also the first launch of a hybrid rocket using a liquid oxidizer in Japan. Tokyo Metropolitan Institute of Technology, now known as Tokyo Metropolitan University, had launched a hybrid rocket using gas oxygen in March 2001. The laboratory began a close partnership with Uematsu Electric Co., Ltd. At Akabira in Hokkaido, Japan in 2004 to develop larger motors. The largest motor ever launched was 5000 N thrust class. This motor boosted a vehicle to a speed of Mach 1.4 and apogee of 7.5 km in March 2012. CAMUI-type rockets are the only hybrid rockets to have broken the sound barrier as of July 2013. In the 53 launches so far, several vehicles carried a CanSat as a payload. The first CAMUI-type rocket, launched in March 2002, carried a CanSat developed by Intelligent Space Systems Laboratory (ISSL) of the University of Tokyo. This CanSat successfully acquired flight data using GPS and acceleration sensors. Figure 6-6 shows the equipment configuration of the first CAMUI-type rocket. The total length and weight of the vehicle were 1.6 m and 10.5 kg, respectively. The 500 N thrust motor carried the vehicle to an apogee of 1 km.

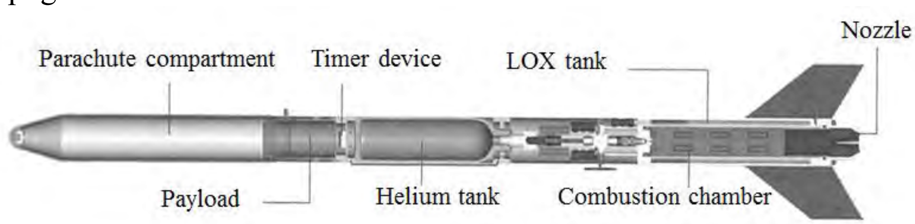


Figure 6-6: Equipment configuration of the first CAMUI-type rocket.

In parallel with efforts to increase size, development activities to simplify the rocket system by employing gas oxygen instead of liquid oxygen are also underway. In 2012, a student group from Hokkaido University conducted 13 launches with a Gas-CAMUI rocket. Three of these launches were for a local high school CanSat championship called CanSat Koshien (a competition about CanSat technology geared towards high school students). These 13 launches set a record in Japan for the number of launches in one year by a student group, all the while demonstrating the practicality of the rocket design. Figure 6-7 shows the appearance of a 2012 model-year Gas-CAMUI rocket. The total length and weight of the vehicle are 1.7 m and 6.4 kg, respectively. The maximum apogee is 400 m. It uses a CFRP oxygen cylinder for medical use without any modification.

Figure 6-7: Appearance of a 2012 model-year Gas-CAMUI rocket..

In the early years of development, this motor was called “Mini-CAMUI”, according to its role in launching small vehicles to altitudes below 250 m. The target altitude, below 250 m, comes from the altitude restriction in the Civil Aeronautics Act. Since the total length of the first Mini-CAMUI rocket exceeded the total length of the first CAMUI rocket launched in 2002, doubts about the name “Mini” arose. For this reason, the motor is now called "Gas-CAMUI," with regards to the phase of the oxidizer not liquid but gas.

In 2013, Uematsu Electric Co., Ltd. developed a low-cost version of Gas-CAMUI, employing a fuselage made of Vinyl Chloride tube. This version launched CanSats in the 2013 during the CanSat Koshien regional tournaments, as well as the finals held later that year. Gas-CAMUI motors surpass HyperTEK motors in reliability and operability.

6.3 Opportunities to Conduct Rocket Launch Experiments

6.3.1 **Noshiro Space Event**

CanSat launch rockets often have large bodies and can reach high altitude, so it is rather difficult to find suitable sites to launch these rockets in Japan. Therefore, since 2005, the “Noshiro Space Event” has been organized to provide an opportunity to launch large rockets. This annual summer event is held in the Noshiro Space Park in Noshiro-city, Akita, Japan. The events give university students from around the country the chance to undertake rocket launch experiments and participate in a CanSat fly-back competition. The park is large enough to allow the launch of rockets to an altitude of up to 500 m. As of 2013, the event has featured another launching site, from which rockets can be launched to an altitude of up to 10 km. This second launch site is called the “Ochiai Beach” launch site. About 3000 people visit the event on the public open days, and the event has come to be recognized also as a local festival in Noshiro. A photograph of the event site is shown in Figure 6-8 and Figure 6-9.

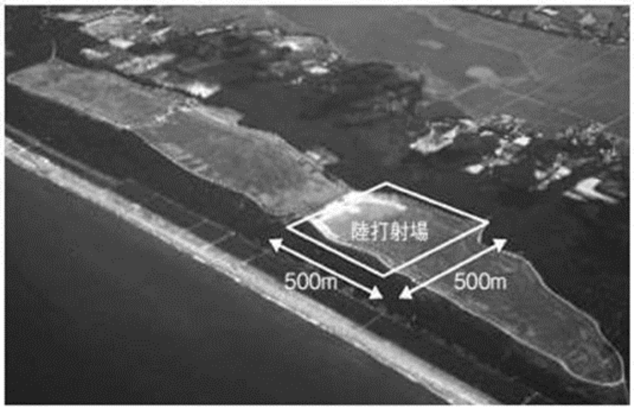


Figure 6-8: Aerial photograph of Noshiro Space Event site (Noshiro Space Park).

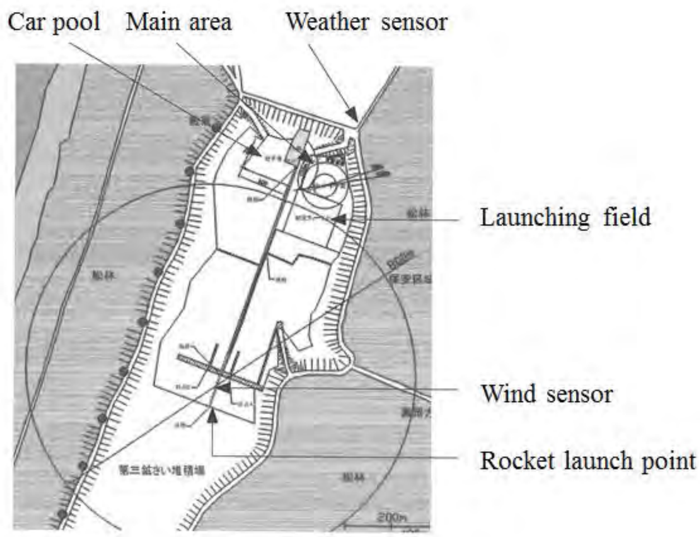
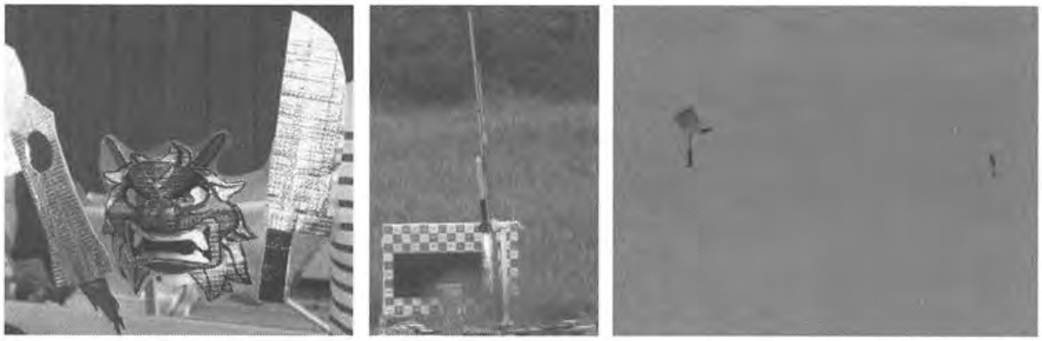


Figure 6-9: Map of Noshiro Space Event site.

One of the trials that has come to characterize the Noshiro Space Event is the “CanSat-Rocket Collaboration Project”. The rocket teams and CanSat teams collaborate to develop a rocket and CanSat as a single project. The project gives the CanSat developers an excellent opportunity to perform a launch experiment. A photograph of a launch experiment performed jointly by Akita University and Keio University is shown in Figure 6-10.



CanSat with a demon image

Launching of CanSat using model rocket

CanSat during decent using parachute

Figure 6-10: Collaborative project between Akita University and Keio University.

6.3.2 **ARLISS**

ARLISS is a rocket launching event held in the Nevada Black Rock desert in September every year. As part of the ARLISS event, CanSats are launched by large-size rockets that have been developed by amateur rocket groups in the U.S. These rockets reach an altitude of about 3 km to 4 km. Of course, these rockets can launch an open-class CanSat. Comeback competitions and mission competitions are held every year. The comeback competitions can consist of either a “fly-back” competition featuring CanSats equipped with paragliders etc., or a “come-back” competition featuring rover-type CanSats. The result of the mission competition is decided by self-assessment. These competitions provide each of the teams with an excellent opportunity to try out the design, development, and technology of their satellites. The event runs for five days every year, with the launch experiments being done on three of those days. On the first day, a technology exchange meeting is held. A breakfast meeting is held on the final day, giving all the teams a chance to report their results.

Amateur rocket groups develop all the rockets launched at ARLISS, based on the same specifications. A successful launch and CanSat ejection demands a high level of technical ability from the participants. A CanSat developer pays only the cost of the engine used for the launch. A schematic view of an ARLISS rocket and an image of a launch are shown in Figure 6-11. The rocket is about 2.4 m long, about 156 mm in diameter, and weighs about 11 kg. The rockets are powered by an M-type engine (M1419W) manufactured by AeroTech.

The rocket separates in two steps to guard against the tangling of the rocket and CanSat parachutes. First, the rocket parachute opens at the apogee altitude. Then, the CanSat is ejected. This two-step separation is controlled by a flight computer mounted inside the rocket.

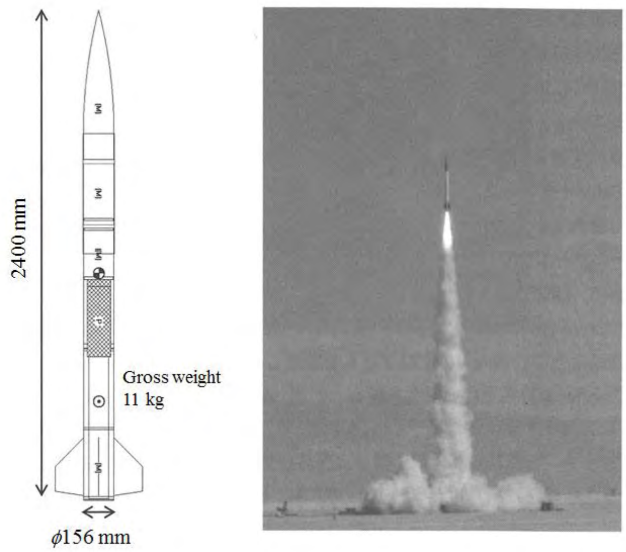


Figure 6-11: ARLISS rocket and launch.

The apogee altitude, maximum acceleration, maximum speed, burn time, altitude upon the completion of combustion, etc. are all recorded on the rocket's flight computer. Typical values are listed in Table 6-3. A CanSat developer can obtain these values from the rocket developers, and record and cross-check the values for the CanSat.

Table 6-3: Example for data from rocket flight computer.

Parameter	Flight #1	Flight #2
Altitude	3735	3726
Maximum Acceleration [G]	8.9	10.1
Maximum Velocity [m/s]	333.833	324.3
Apogee time [s]	28.06	21.3
Burn time [s]	5.38	4.69
Altitude at burn out [m]	1075	928

6.3.3 Collaboration Among University Laboratories

Many of the universities which participated in UNISEC, including Hokkaido University, Akita University, Tokai University, and Chiba Institute of Technology, are developing their own small rockets. Cooperation among these institutions could lead to the successful launch of a CanSat. As an example of such collaboration in which a small rocket is launched, a CanSat is carried as the payload and the flight data is measured. Establishing a connection with a university capable of launching a rocket will greatly increase the chances of launching a CanSat (refer to the UNISEC homepage http://www.unisec.jp/member/university.html).

6.4 Rocket Launching Sites in Japan

6.4.1 **Taiki-Town (Hokkaido)**

Taiki-town is located south of Obihiro City, and lies along Hokkaido's southeastern pacific coast. From the very beginning of the Hokkaido Aerospace Industry Station Project starting in 1988, Taiki-town has been one of the most enthusiastic supporters of aerospace-related activities. Even now, they continue to promote "town development based on space" project. In 1993, they started contracting "Taiki Multipurpose Aerospace Park," consisting of a 1 km runway as an aerospace-related experiment field and operation of the park began in 1995. By 1997, Taiki-town concluded an agreement with the National Aerospace Laboratory (NAL at that time) concerning use of the park. In 2003, the park became the base of the stratospheric platform airship demonstration experiments led by Japan Aerospace Exploration Agency (JAXA) and National Institute of Information and Communications Technology (NICT). Cooperation between Taiki-town and JAXA led to establishment of the Partnership Cooperation Agreement in 2008, in which JAXA-owned facilities in the park, including the large facility for the stratospheric platform airship demonstration experiments, were reorganized to form Taiki Aerospace Experimental Field. On this occasion, JAXA constructed a control and operation tower, as well as a launching apparatus for stratospheric balloons. Various research organizations, including JAXA and many universities, use this field for aerospace-related experiments throughout the year. In Taiki-town, aerospace-related research activities are deeply accepted by both the town office and residents. Figure 6-12 shows an aerial photo of Taiki Multipurpose Aerospace Park.

Figure 6-12: An aerial photo of Taiki Multipurpose Aerospace Park.

When a rocket experiment is conducted in Taiki-town, a launcher is set near but outside of the Multipurpose Aerospace Park. The park is surrounded by grass farms, and rocket experiments are possible there when these farms are covered with snow. Accordingly, rocket experiments at this location are possible only in winter. As a safety requirement, apogee altitude is limited below 1 km, corresponding to the distance from the launch site to a national road. When a higher altitude is necessary, a launch from the coast to the sea is possible. Fishing boats recover the vehicle from the sea in this case. Up to now, four launch experiment campaigns to the sea have been conducted, in August 2007, July 2011, July 2012, and August 2013. The highest altitude ever recorded from the coastal launch site was 7.5 km by a CAMUI-type rocket.

An approach line to Obihiro airport passes through the airspace above Taiki-town. Consequentially, the area above 1.8 km is under the control of Kushiro-airport; which is in charge of regular flights. The area below 1.8 km is used as training airspace, and the Obihiro branch campus of Civil Aviation College is the primary user. Before a launch experiment, negotiations with air traffic control offices are necessary to close the training airspace. Additionally, launch experiments are limited to time slots that not include the landing and takeoff periods at Obihiro airport.

6.4.2 **Noshiro-City (Akita)**

The first Noshiro Space Event was held in August 2005; in commemoration of the 50th anniversary of the first Japanese rocket experiment, and subsequent experiments conducted by Professor Itokawa, revered as the father of space development in Japan. Now the event has grown to be one of the largest space-related events in Japan. The launch site prepared for the event in 2005 is continuously used. The previous subsection 6.3.1 describes the details of this launch site.

6.4.3 **Izu-Oshima (Tokyo)**

A feasibility study in 2010 revealed that a launch site in Izu-Oshima would allow for high altitude ground recovery launch experiments. The first launch experiments were planned to take place in March 2011, but were postponed due to the Great East Japan Earthquake. Eventually, the first launches were conducted in June 2011 by student groups from Wakayama University, Tokyo Institute of Technology, and an inter-college circle

The launch site in Izu-Oshima is in a field near the crater of Mountain Asama, commonly referred to as “Back Desert”. Since the crater and immediate surroundings are officially designated as a special environmental protection area, for which public access is not allowed, the launch site is located in a nearby type-1 preservation district. Compared to other domestic launch sites, in which maximum available safety distance is below 1 km, a longer safety distance is possible in Izu-Oshima. Accordingly, this launch site has the potential to allow for apogee altitude launch experiments up to 2-3 km with ground recovery.

6.4.4 **Cosmo Park Kada (Wakayama)**

The Institute for Education on Space at Wakayama University started using Cosmo Park Kada as a local launch site in February 2009. The first launch experiment here was conducted by a student group from Wakayama University. This area was a soil mining zone for the landfill construction of Kansai International Airport. Currently, a redevelopment project for an industrial complex by Wakayama Land Development Corporation is underway. There are vast flat vacant parcels sufficiently far from neighborhood residents to be considered appropriate for launch experiments. The Wakayama prefectural government manages this area, and only a portion of the land is used. As of 2013, they have provided a parcel for use in launch experiments. Launches up to 400 m apogee altitude are possible here. Figure 6-13 shows the landscape of the parcel.

Figure 6-13: Landscape of a parcel provided for launch experiments in Cosmo Park Kada..

6.5 Field Test Using Tethered Balloon

Figure 6-14 shows a conceptual diagram of the entire system of the balloon experiment. The balloon is tethered with several cables usually 3 cables as shown in the figure. The balloon is coupled to the reels on the ground and is moved up by releasing the reels. In the experiment, the reel rotation is locked to hold the cable at a constant altitude. The lid is on the bottom face of the carrier, and CanSat is released by gravity when the lid opens. Subsequently, when released from the carrier, the parachute opens and then drops slowly. After the experiment, the reels are wound up to pull the balloon down to the ground safely.

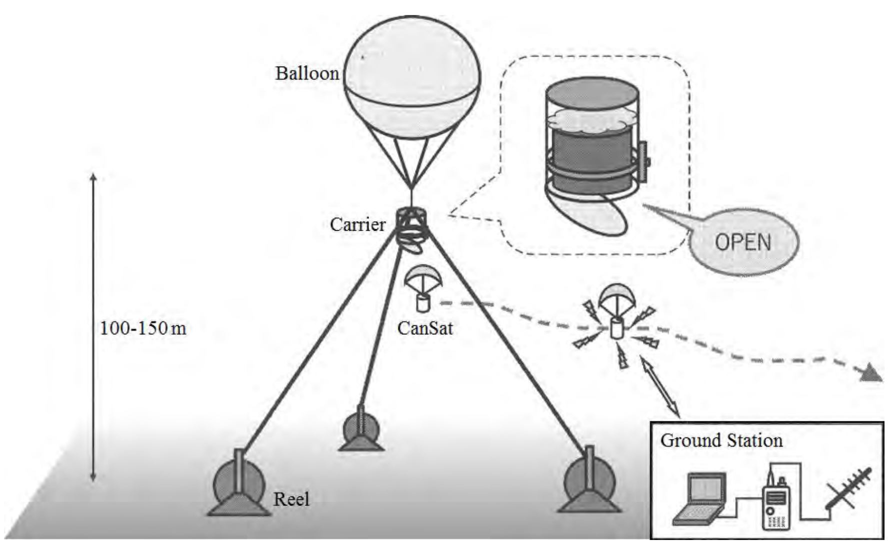


Figure 6-14: Conceptual diagram of balloon experiment system.

Although the details are described later, the infrastructure necessary for the balloon experiment is shown in Figure 6-15. Six elements are required, specifically; a) Tethered balloon, b) Carrier, c) Reel, d) Clasp coupling the balloon to the cable, e) Weight (e.g., plastic container of water) and chain that allows the balloon to not be moved up but be tethered on the ground, and (f) PC for ground station to control the carrier.

a) Tethered balloon

b) Carrier for CanSat

c) Reel

d) Clasp coupling the balloon to the cable

e) Weight for balloon standby

f) Ground station for carrier

Figure 6-15: Infrastructure necessary for the balloon experiment.

The procedure for the balloon experiment is listed as follows:

a) Perform the setting of the tethered balloon.
b) Perform the setting of the carrier.
c) Store CanSat in the carrier.
d) Move up the balloon at a predefined height and tether it.
e) Open the lid of the carrier to release CanSat by gravity from the carrier.
f) The air enters into the folded parachute to open it, and CanSat drops at a predefined speed.
g) During dropping, CanSat performs the mission. Meanwhile, at the simple ground station consisting of PC and radio, the experimenter receives data from CanSat. The experimenter also uplinks a command to CanSat.
h) After landing, the mission is continued in some cases.
i) After the mission is completed, CanSat send data to the experimenter about its position using a method such as beacon or transmission of GPS data through radio waves.
j) The experimenter collects CanSat, and then retrieves the data stored in the memory and compare it with the data at the simple ground station.

When conducting the balloon experiment, the experiment shall be planned in careful consideration of the above experiment sequence. The above experiment procedure is detailed below. Note that this is an example, and the actual procedure shall be figured out in consideration of the experiment site and other conditions.

6.5.1 **Setting of Tethered Balloon**

The recommendation is to conduct the balloon experiment during a time when the wind is weak, for example, early morning. Although the wind may be weak during daytime, the good weather is likely to cause an upward current in the upper air, which may fly CanSat out of the site. In addition, even if the wind is weak, it is better not to set the tethering height too high.

The tethering height is determined so that CanSat may land inside the experiment site. In the early morning with weak wind and not so high temperatures, for example, the tethering height might be up to approximately 100 to 150 m for the site of approximately 400 to 600 m^2. In addition, when the site is in the control zone of the Self Defense Force or an airport, it is necessary to obtain permission from the Self Defense Force or a corresponding local authority if the tethering height is a constant value or higher. Although it is no problem if the prescribed procedure is followed, it takes some time. It is better to assume at least one month. If the experiment is lightly performed, it is safer to set the height lower.

The most important point in handling with adeptness is a stable tethering of the balloon, which is understood via an actual balloon experiment. There is no problem if the experiment is conducted when the wind is weak, but it may have to be conducted when the wind blows to some degree. In that case, it is very important to determine whether the experiment is conducted or canceled. If it is predicted that it will have rain and strong wind according to earlier weather forecast, it is required to make an appropriate decision, such as cancellation on the previous day. Because the criterion for judging which extent the experiment should be conducted or canceled depends on the area of the experiment site, the wind environment, and capabilities and experiences of experimenters, it cannot be generalized here. However, if at least the situation where the tethering of the balloon is destabilized is understood, an appropriate determination may be made to some degree. In many cases, destabilization occurs in the following situations:

- The balloon is flown by the wind to a lower height and is pulled back by the tethering cable when being flown to a place. This sequence of operations is repeated, and the balloon position is not stabilized in one place.
- The tension on only one or two of tethering cables, and other cables are slack and are not functional.

This is simply because the distance between the reel installation positions is too narrow. For simplicity, assume tethering of two cables of the same length in two dimensions. In this case, the balloon subjects to a force of subtracting the gravity force, mg, from lift force, L, in the vertical direction, and subjects to a drag force, D, in the horizontal direction by the wind. The resultant vector F of these forces is balanced with a resultant force of tensional forces, T, by two tethering cables, and T is always applied between the two cables. This is because not a compression force but only a tensional force works on the cable.

At this point, as shown in Figure 6-16, if the cable tilt angle θ is large, F is not applied between the cables as shown in the left of the figure. Therefore, no balance with T will move the balloon. Specifically, the balloon will move down until F's angle α is matched to the angle θ_2 of the cable 2, as shown in the left of the figure. Accordingly, only cable 2 supports the balloon, and cable 1 is slack and is not functional. As the change of the wind speed varies the drag force D, F's angle α also varies, and the balloon also moves until α is matched to the angle θ_2 of cable 2. Therefore, a slight change in the wind speed will move the balloon, which destabilizes the balloon position. F's angle becomes smaller as D becomes larger, which will move down the balloon.

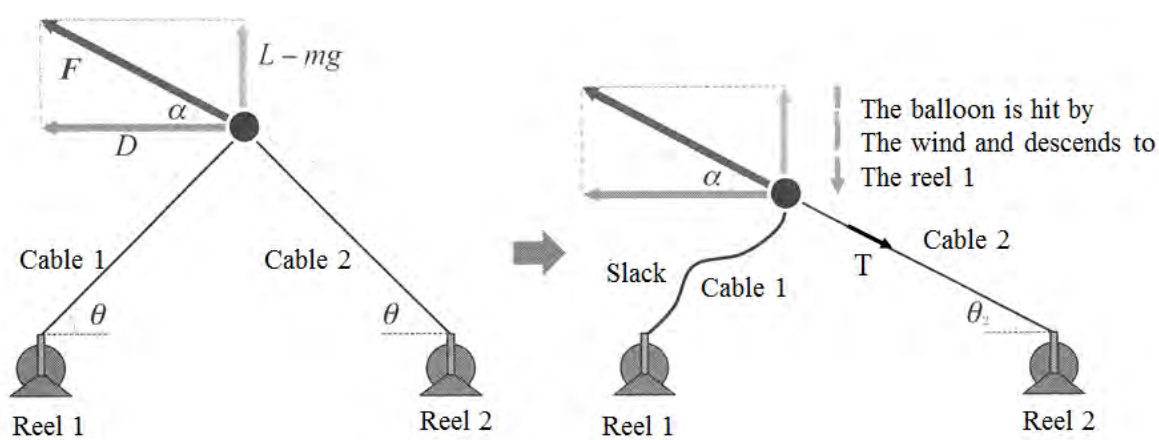


Figure 6-16: Unstable tethering of balloon.

In the case shown of Figure 6-16, if the distance between reels is too short, it is useless to wind up the cable with reel 2 for stretching cable 2. Because α does not become less than θ_2 even for the maximum peak wind unless the reel is wound up a great deal, and even if it is wound up so, the height of the balloon is too low to be meaningful for the experiment. To avoid this, even for the maximum peak wind as shown in Figure 6-17, F must be applied between two cables. This requires the distant between reels to be long enough for the maximum assumed peak wind, in other words, the reel installation position shall be estimated. In such case, even if the wind varies, the simple tensile forces of two cable changes and the balloon position do not change. In fact, the cable wobbles slightly because the drag force working on the cable changes, but the wobbling does not become so large if a tensile force is applied. Of course, when the lift force, L, is increased, even if D increases, F's angle, α, does not become so small, which allows the balloon position to be stabilized. However, depending on the size and strength of the balloon, the lift force, L, is limited. In fact, it would be a practical solution to lengthen the distance between reels.

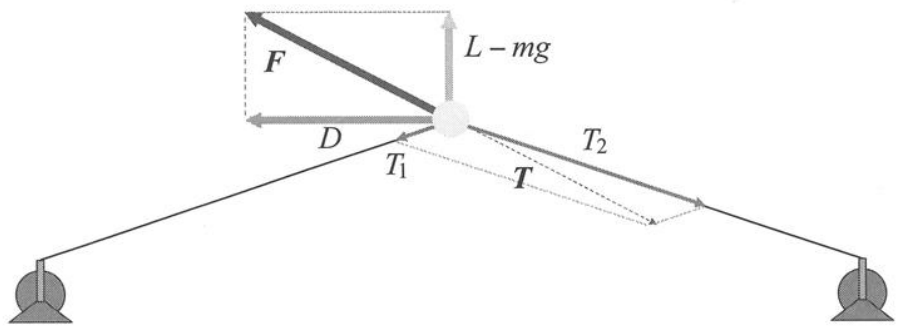


Figure 6-17: Stable tethering of balloon.

The above is a case where cables are light and the lift force of the balloon is sufficient. Figure 6-16 or Figure 6-17 shows each cable drawn in direct line, but in fact, it appears to be catenary instead of a direct line under its own weight to be exact. Although the cable is regarded to be an almost direct line if it is light, the cable is extremely slack if a heavy one such as a rope is used as a tethering cable. In this case, stabilizing the balloon in a fixed position should be given up because the angle of the cable becomes high as shown in Figure 6-18. This issue can be addressed through a method in which, for example, a person moves the balloon depending on the wind by operating it like a kite and opens the lid to drop CanSat at the moment when the wind dies.

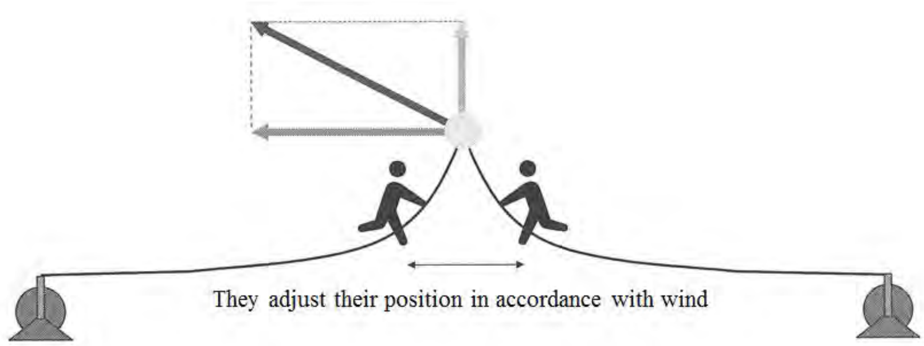


Figure 6-18: Tethering method when lift force is insufficient.

This procedure recommends the method for stably tethering the balloon shown in Figure 6-17. This means it is important not only to inject helium into the balloon, which provides a sufficient lifting force, but also to use a tethering cable of lightweight one and high strength, such as a tuna fishing wires.

6.5.2 **Carrier Setting**

Extreme care shall be taken for the carrier setting, in other words, the operational confirmation of functions required for the carrier. Cutting corners in the above work will result in problems, including a case where the carrier lid does not open at the release of CanSat. It is common, especially for beginners, that too much attention is focused on the development of the CanSat body and insufficient attention is focused on the carrier setting, which results in failure at the release. Figure 6-19 shows an example of the carrier. The functions required for the carrier are as follows:

- The carrier shall be able to store CanSat. The carrier is preferably compatible with the rocket experiment. Specifically, the envelope area of the storage section shall be 146 mm in diameter and 240 mm in height.
- The carrier shall have a function to stably mount itself to the balloon. For example, the carrier is three -point supported, and three cables are connected to the cable directly below the balloon.
- The lid shall be able to be opened remotely. The carrier shall have the timer function, and after the balloon is moved up with the timer ON and is tethered at a predefined height, the lid shall automatically open or shall be ready to open via the uplink command from the ground station. For the timer, the time between when the balloon is moved up and when the balloon is tethered shall be checked beforehand, and the value of timer time determined by adding a margin to the above time shall be set.
- The carrier shall be able to acquire the sensor data in which the flight environment can be checked, such as wind speed and temperature (optional).
- The carrier shall not have a protrusion in which the parachute of CanSat may be caught.

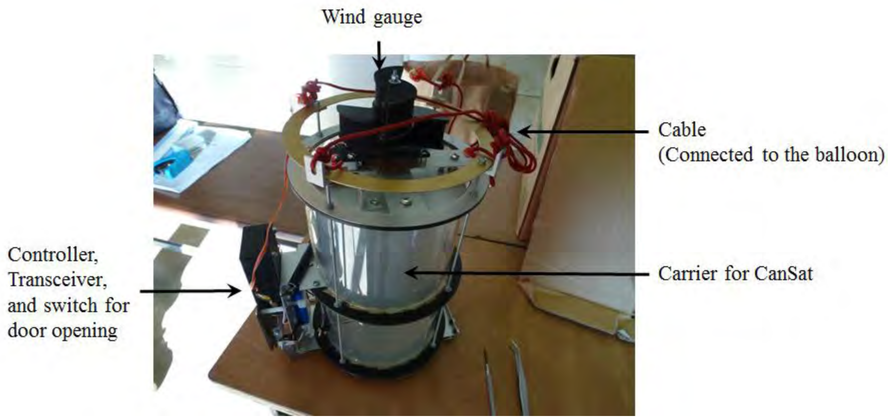


Figure 6-19: Example of carrier.

6.5.3 **Storage of CanSat**

Pay close attention when storing CanSat in the carrier to prevent the parachute from being caught in the carrier upon release and to ensure that the parachute opens. Both points depend on how to fold the parachute and how to store it after folding. It is necessary to store the parachute with its size smaller than the internal diameter of the carrier and to exercise ingenuity in folding it so that it will open easily. Also, the verification shall be performed thoroughly (as described in Chapter 5 | for advance verification).

Figure 6-20 shows an example of CanSat stored in the carrier. The carrier can accommodate the 350 ml CanSat or the open class CanSat, and the partition plate is inserted in the carrier's cylinder for the 350 ml CanSat. Figure 6-20 (a) shows CanSat accommodating two 350 ml CanSats with the partition plate inserted. The lid is divided into two parts, and the first CanSat is dropped when only one lid is opened, and the second CanSat is dropped when the other lid is opened.

a) 350 ml CanSat   b) Open class CanSat

Figure 6-20: CanSat stored in the carrier.

6.5.4 **Moving up, Tethering of Balloon and Release of CanSat**

After the carrier is installed, the moving up and tethering of the balloon is performed by safely elongating the tethering cable from the reel if the balloon setting is properly performed. It is preferable that an indicator stands under the balloon and instructs a person in charge of each reel to elongate the cables while maintaining balance.

There are some methods for the trigger to release CanSat; for example, the recommendation is to use a method in which the uplink command is prepared in each of timer and radio. Usually, the controller is programmed so that the release is performed when the uplink is received. It is practical that the release time of the timer is set to a slightly long time so that the release is performed by a timer if the uplink is disabled due to wireless failure. However, using only the timer disables the release depending on the wind conditions. In addition, setting the release time too short will release the carrier before the balloon is tethered, and setting it too long time will take long for the release to be performed.

6.5.5 **Collection of CanSat and Subsequent Work**

There should be no problem between when CanSat is released and when it is collected as long as advance verification is conducted properly. However, because it is likely that the wind is stronger than expected in the upper air and CanSat is blown a considerable distance, collectors shall wait in the outermost area of the site or outside the site, and then contact the leader of the collectors to perform the collection work. The cooperation among collectors is required for efficient execution of the experiment.

It is probably required to implement in CanSat and the simple ground station system the function to check the data at the experiment site. There is also a method for checking the data in the laboratory after the experiment is completed, but it is difficult to deal with a problem in the data in the laboratory. Care should be taken especially when performing flights many times. If a problem in data acquisition is confirmed on-site for the first flight, the field correction is made on-site for any subsequent flights. If this cannot be done, the flight is performed many times as is, which ends up finding that everything failed in the laboratory. This “it's too late” situation should be avoided.

7 | Field Work and Project Self-Evaluation after Field Test

7.1 Work Contents and Data Analysis

The field test is a very intensive task, the feeling of accomplishment and pleasure when it went well or the disappointment and depression when it did not and move on to the next task cannot be done. The project is completed only after analyzing and judging the success of the mission, and make a report or presentation that can be reflected provide experience for the next CanSat project. It is an extremely important process. This chapter will present the post field test activities.

7.1.1 Visualization and Analysis the Data

It is very important to visualize and analyze the acquired data during the mission. Data visualization must be done in a clear way using the proper graphs. It should be kept in mind that such visualization is aiming to help analyzing the data and extract hidden information relevant to the mission and its sequence. Sometimes the data is outputted as non-dimensional quantities of the A/D conversion or voltages and needs conversion to the proper physical quantities. It is required also sensors calibration in advance

In the comeback competition, it is requested that the control history be stored for assessment and analysis. This also will assess whether CanSat achievement is due to the control actions or the environmental factors like the wind effect or luck. This means that, for example, the camback CanSat acquire the relationship between the current position, the speed vector, and the target position, and sets an appropriate control action based on these information and indicates that the correct control signal is being sent to the actuator. Since this kind of analysis should be carried out immediately after the flight of the camback CanSat. Therefore, a template need to be prepared that can show the above information as soon as data is obtained. Such preparation is vital in making the presentation and appealing to the judging committee and the available limited time after the flight.

7.1.2 Confirmation of Operation Status and Failure Analysis of each Component

Check the operational status of each designed and manufactured component, for example, microcomputer and sensors, and check what kind of failure was observed in case of normal operation or abnormal operation in a clear and organized way. For failure, it is important to know its cause firmly and its effect. The cause of failure should be made clear.

7.1.3 Determination of Success Level

Since the levels of mission success criteria were decided at the very beginning of CanSat development, post flight results will help to identify what the achieved level of success whether it is minimum, full or advanced

7.1.4 **Impact on the Project and Future challenges**

Describe in detail the future challenges, if the mission goes well then it is fine to highlight any future challenges that the team want to pursue. In case of mission failure, the needed improvements to successfully achieve the mission next time should be presented and discussed.

7.1.5 **Report Preparation and Final Presentation**

Final report to sponsors and those who cooperated is essential. In addition to preparing the report, it is desirable to have a presentation meeting. The people involved and interested in the teams' CanSat project should be invited to the presentation meeting. At the meeting, the whole project from the early design phase, manufacture and testing of the subsystems to realize the mission as announced at the Preliminarily Design Review (PDR), verification results, flight results, cause of failure and finally a general comment and suggestions for the future challenges must be presented. Questions and answer session is typically followed the presentation so it is a good chance to collect feedbacks and comments. It is highly recommended to prepare your presentation in a comprehensive way. Also, when many universities conduct projects in parallel like in ARLISS, the joint presentation meeting with all of them is also effective.

7.2 Future challenges

As mentioned above, the future challenges must be highlighted. Sometimes it is difficult but it should be done because it provides the team members with a chance to assess themselves. The following are examples of future challenges that might be considered and discussed in a CanSat project.

7.2.1 **Evaluation of Adopted Parts / Equipment and Design**

- Whether the used technology was reasonable or not. For example, whether the sensor adopted to measure one parameter was properly specified, whether the microcomputer was an appropriate choice from the viewpoint of capability and development efforts, whether the designed circuit was properly designed, parts or equipment, etc. These evaluations can be carried out.
- Evaluate the development process including the length of development time, difficulty, issues of reproducibility of data at the time of the verification, even if eventually everything worked well. Propose methods for development that is more reliably and more easily.

7.2.2 **Evaluation on Advance Ground Test Methods**

- Whether the conducted tests are enough to verify system reliability and performance.
- It is also important to evaluate not only the content of the test but also whether it was possible to conduct a sufficient number of tests to ensure reliability and reproducibility. For critical operation, for example, if 10 trials were done, how often failure in repeatability was observed and how this was overcome.

- Regarding the verification, evaluate whether it was precisely defined in advance what to test for verification, and whether it was the suitable test that confirmed the requirement. Verification test should be clear and unambiguous.
- At what level the flight model was used? Is it used too much? Generally, the failure rate is said to change over time in the same way as shown in Figure 7-1. The flight model must be tested enough to drop the initial failure period and don't reach the fatigue failure period. The flight model testing should be highlighted.

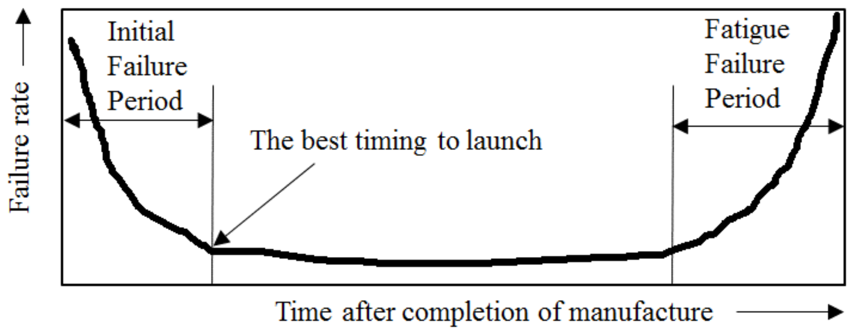


Figure 7-1:.The failure rate curve.

7.2.3 **Mission objectives Evaluation**

- Evaluate the mission objectives and whether they were reasonable set for given budget, period and ability. People with little experience have a strong tendency to set difficult goals beyond their abilities and this should be avoided.
- In CanSat , the results of past projects performed by seniors by each university are available in advance. These projects usually provide good motivation and guidelines. Sometimes it is hard to advanced mission for beginner. The mission objectives should be in line with the capability of the team as beginner and the available resources and experience.
- Have the success criteria been set reasonably enough to be objectively evaluated? For example, the success criteria that "the fly-back CanSat flies stably" is not valid as a criterion because the definition of "to fly stable" is not clear. It was necessary to set concrete measurable goals such as "Horizontal stationary flight with height sink rate of 5 m/s or less can be continued for 20 seconds". The acquisition of latitude, longitude, and altitude information by GPS should be done for several minutes and evaluate the measurement and its precision and whether it is enough or not. Details should be specified. Otherwise objective evaluation cannot be done. While analyzing the results, re-evaluate the success criteria that was set earlier.
- To realize the mission goal, "Requirement Analysis" to derive the breakdown requirements should be done carefully otherwise it will be a very vague development. Focus on existing status and what should be developed with priority and sharpness such as "This must be realized reliably" or "It will consider this later if others can be realized". Evaluate whether the requirement analysis was conducted carefully.

7.2.4 **Evaluation Methods for Realizing the Objectives**

- o Even if a goal was set that was not within the current capability, there are cases where it will be achieved if it involves external knowledge in the project by external person and looking for guidance from that person. Even if one of components did not developed locally, it might be considered a purchased component. Evaluate whether such measures so that the realization of goal was appropriate.

7.2.5 **Evaluation on How to Set and Manage the Schedule**

- o Schedule creation is a very important step. This requires an accurate estimate of how much time is required for each required work in the development before implementation. This breakdown is called Work Breakdown Structure (WBS). However, in the case of a beginner who has not actually developed a CanSat before, it is often unknown how much time it will take for which work, and as a result, it tends to be a very "optimistic" schedule as a result. The countermeasure against that is to put a sufficient margin, but in many cases even beginners do not know how much minimum margin is necessary. Therefore, it is important to seek adequate guidance from experienced people and instructors and make a schedule with plenty of time, and if there is a time limit, lower the target so that schedule will have a margin.
- o Also, when breaking down the goal to several development items, it is important to prioritize and set the priority such as "This is developed reliably first" and "Develop if you can afford it".
- o As the development progresses, the schedule gradually shifts from the initial estimate. It is important to recognize the deviation always and to take actions for tasks with high priority or to change the placement of personnel. And if you make a serious decision, such as having to delay the milestone date, for example, integration test, review the adequacy of system requirements and design, or the validity of how to proceed. To manage the schedule precisely, it is necessary to visualize the schedule so that multiple people can discuss the schedule.

7.2.6 **Other Evaluation**

Many of the points mentioned above are related to project management, but also evaluate the following points in addition to those mentioned above.

- Were team divisions and roles shared appropriately?
- Have project managers properly demonstrated leadership roles?
- Has the team's morale maintained until the end?
- Has there been any problem with sharing information within the team?
- How was the time required for each work different from the predicted and the actual?
- Was allocation and management of the budget appropriate?
- Has the document management been done properly?
- Has the information disseminated to the outside necessarily and sufficiently?
- Was enough consideration for those who took care of sponsors?
- Others

7.3 Lessons Learned from Past CanSat Projects

At the end, some important considerations based on failed cases in the past CanSat project are presented as follows

7.3.1 Failure Factors and Countermeasures of Parachute and Parafoil

The parachute and the parafoil are "single point of failure" because it is vital in CanSat experiment. It is a critical element for safe recovery and is not normally used as a redundant system. Nevertheless, it seems that there is not enough time to design and analyze it. It is necessary to pay sufficient attention to the following points.

- Is the structure is designed to withstand the impact of the parachute opening? It is necessary that the structure of the joint of the parachute and the CanSat body have sufficient strength. Calculate the parachute shock from the speed of CanSat release (which is expected to be considerably faster when released from the rocket with ejection charge). The force spreads to the joint of the CanSat body with the string without breaking it. It is necessary to calculate strength and design the structure as to whether it can withstand such strength sufficiently.
- After the parachute is folded and stored, it is necessary to consider whether it can deploy properly when falling. Since how to fold has know-how, it is a short cut to learn from experienced people and to master such know-how.
- Parafoil is very difficult to deploy, accidents often of the kind of falling down due to halfway deployment, and many strings becoming tangled before deployment. For deployment, it is essential to examine the way of folding so that the wind surely enters between the two layers of the parafoil, and consider the material of appropriate rigidity. For the problem of cord tangling, the number of cords should be reduced as much as possible, and a cylindrical cord storage part like a hose cut at the top of CanSat is provided, and a cord is drawn out as the parafoil expands Such an ingenuity is effective.
- Size the parachute assuming the required descend speed is necessary. Calculate the descend speed from altitude data and the necessary experimental time and derive the parachute area that realizes it. It should be noted that the larger the area, the greater the horizontal distance will be.

7.3.2 Definition of Environmental Conditions and Measures

- For example, in the case of came-back CanSat, it is necessary to define the boundary conditions related to the environment such as how much wind speeds have the ability to move toward the target point, and consider this in the design. In the case of a fly-back CanSat using parafoil, it will not be possible to handle any kind of wind speeds. However, if you are looking at the came-back CanSat so far, first develop it without defining speed limits, flight in any circumstances, eventually you cannot lose due to the wind effect and cannot produce good results. Defining and developing as "designing condition of this wind speed" as a design condition, in the field we constantly monitor the speed of the wind and it is necessary to address that "to make it fly only at this wind speed or less".
- It is necessary for defining the boundary condition that it can cope with the Rover-back type CanSat to, "How much depth of rut" can be dealt with. If you go to the site and have more ruts, let's give it up if it cannot be helped. By taking statistical data on how much the depth of rut is often in the past,

it is important to set boundary conditions by considering that "if we can get over to this depth, we can cope with this probability." It is an engineering approach.

- In the case of software verification, it is effective to simulate actual environmental conditions by giving dummy input data to the microcomputer. In Japan, it is impossible to imitate the vast environment of ARLISS, failures such as bugs in processing when during flight, incorrect switching between east and west, can be avoided by this method.

7.3.3 **On the Defect of Electronic Components**

- Among the problems of the electronic system, many are initial mistakes such as disconnection, contact failure, peeling of solder, …etc. Especially, even though it is done properly at the time of preliminary experiments, such defects are often created at the time of the final integration in the field such as the desert. Such a field is the worst as a place to integrate, so it is required to reduce work in such a place. Even more, you must avoid thoroughly the situation that you must solder on site. For that purpose, it is desirable to reduce work on site (just switch on), even if you have to work, It is necessary to thoroughly implement a method with high reproducibility.
- In the case of ARLISS, the rocket gives a large acceleration (8 G or more) and intense random vibration (25 G_{rms} or more), so the above-mentioned troubles are often caused by it. Although it is difficult to test for acceleration, vibration test with environmental conditions of this rocket should be done in advance.
- In the case of ARLISS, the temperature inside the rocket is high and there are some electronic systems in which troubles occur at this temperature. Parts with a wide allowable temperature range are desirable. This can also be said to be a problem of definition of the environmental condition in the previous subsection 7.3.2.
- When purchasing electronic parts, communication equipment, sensors, etc., it is necessary not to trust the specification sheets but to always test and to confirm that the performance as per specification sheet is obtained. There are many cases that failed because of the lack of the confirmation of specification.

7.3.4 **Failure and Measures of the GPS Receiver**

Sometimes, GPS cannot be locked in came-back CanSat. This is often caused by the following reasons, and appropriate countermeasures can be taken once the cause is investigated.

- When the multiple of the clock of the CPU, overlaps the GPS frequency (L-band), there is a possibility that noises enter through the GPS antenna and cannot be locked. Change the frequency of the clock, or shielding it by covering the CPU and oscillation circuit with aluminum foil.
- When the posture of the came-back CanSat is unstable and the orientation of the antenna is not stable, the GPS satellite locked at one time and cannot be locked at the next timing, and so on. Respond by broadening the field of view of the antenna and by stabilizing the direction of the antenna as much as possible.
- GPS takes time to lock up at cold start (tens of minutes in many cases). Therefore, at the very least, turn off the switch with the measurement being done before the launch, or respond by keeping the GPS receiver running only when it is in the rocket or balloon.

- In some cases, locking does not take place due to antenna gain problem. In that case, it is necessary to change to a somewhat higher gain antenna rather than a small simple antenna, or to check whether the coaxial cable of the antenna is broken at a small turning radius.

7.3.5 Note on Come-back Mission

- In the case of a fly-back type CanSat, if you must lift up to the sky, many end-to-end experiments in advance cannot be done easily, which has led to poor grades. On the other hand, rover-type CanSat experiments can be conducted as much as making a traveling environment simulating the desert, so in recent years it has been making outstandingly good results. It will be necessary to devise contrivance such as jointly creating preliminary experiment opportunities many times.
- On the other hand, the failure often occurs in the case of the rover-type CanSat falls on the rover that the parachute first landed, and the case where the rover and the parachute are entangled with each other is resulted. Again, it is necessary to ensure the reproducibility to the extent that the design can be avoided certainly and the simple drop test is repeated and at least 10 consecutive successes are attained.
- In the integration test, ensuring such reproducibility (always achieving the same state of success as always) is very important. It is difficult to succeed if it comes to field examination saying that it succeeded one or two times.

Appendix A: Part List of i-CanSat kit

Table I: i-CanSat Part list

No.	Description	Quantity	Kit code
GPS Board			
1	GPS PCB	1	-
2	GPS receiver	1	GPS-17
3	Toggle switch	4	GPS-13
4	EH-Connector, 3 circuits, shrouded header (top entry type)	1	GPS-7
5	EH-Connector, 3 circuits, housing	1	GPS-18
6	Pin header, 2 circuits	1	GPS-8
7	2 pin jumper	1	GPS-19
PWR Board			
8	PWR PCB	1	-
9	EH-Connector, 2 circuits, shrouded header (top entry type)	2	PWR-1
10	EH-Connector, 2 circuits, housing	2	PWR-2
11	9V Alkaline battery	1	PWR-3
12	9V Battery clip snap on connector with 220 mm lead cable	1	PWR-4
USR Board			
13	USR PCB	1	-
14	Temperature sensor module I^2C - 16 bits (ADT7410)	1	USR-1
15	Miniature I^2C digital barometer sensor module (MPL115A2)	1	USR-2
16	Two-axes gyroscope sensor module (ENC-03R)	1	USR-3
17	3-axes accelerometer module (ADXL335)	1	USR-4
18	1 µF, 50V, Multilayer ceramic capacitor	10	USR-5
OBC Board			
19	OBC PCB	1	-
20	EH-Connector, 5 circuits, shrouded header (top entry type)	1	OBC-5
21	EH-Connector, 5 circuits, housing	1	OBC-15
22	Surface mount pin socket, 5 circuits	1	OBC-4
23	EEPROM (24LC1025)	1	OBC-13

No.	Description	Quantity	Kit code
CAM Board			
24	CAM PCB	1	-
25	CANCAM CMOS sensor	1	CAM-17
26	CANCAM sensor housing	1	CAM-18
27	2 GB Micro SD card with adapter	1	CAM-3
28	M2×6 plastic screw	4	CAM-20
29	M2 plastic nuts	4	CAM-19
X-Bee Board			
30	XBee PCB	1	-
31	XBee Pro module	1	XBEE-3
32	Surface mount pin header, 3 circuit	1	XBEE-4
33	Xbee pin socket	2	XBEE-5
Ground Station Unit			
34	XBee Pro Module	1	GS-1
35	USB cable (1 meter)	1	GS-3
36	XBee-USB interface board	1	GS-2
Structure and Connectors			
37	13 mm Male-Female spacer	6	STR-4
38	20 mm Male-Female space	9	STR-3
39	30 mm Female-Female spacer	3	STR-5
40	M2×10 metal screw	6	STR-6
41	Pin header (40 pins)	2	STR-2
42	Pin sockets (40 pins)	2	STR-1
43	Contacts (terminals)	≈33	STR-7
Development Kit and Cables			
44	Microship Pickit 3	1	DEV-2
45	USB-TTL RS232 Cable	1	DEV-1
46	Color flat cable, 5 wires, 125 mm	1	DEV-4

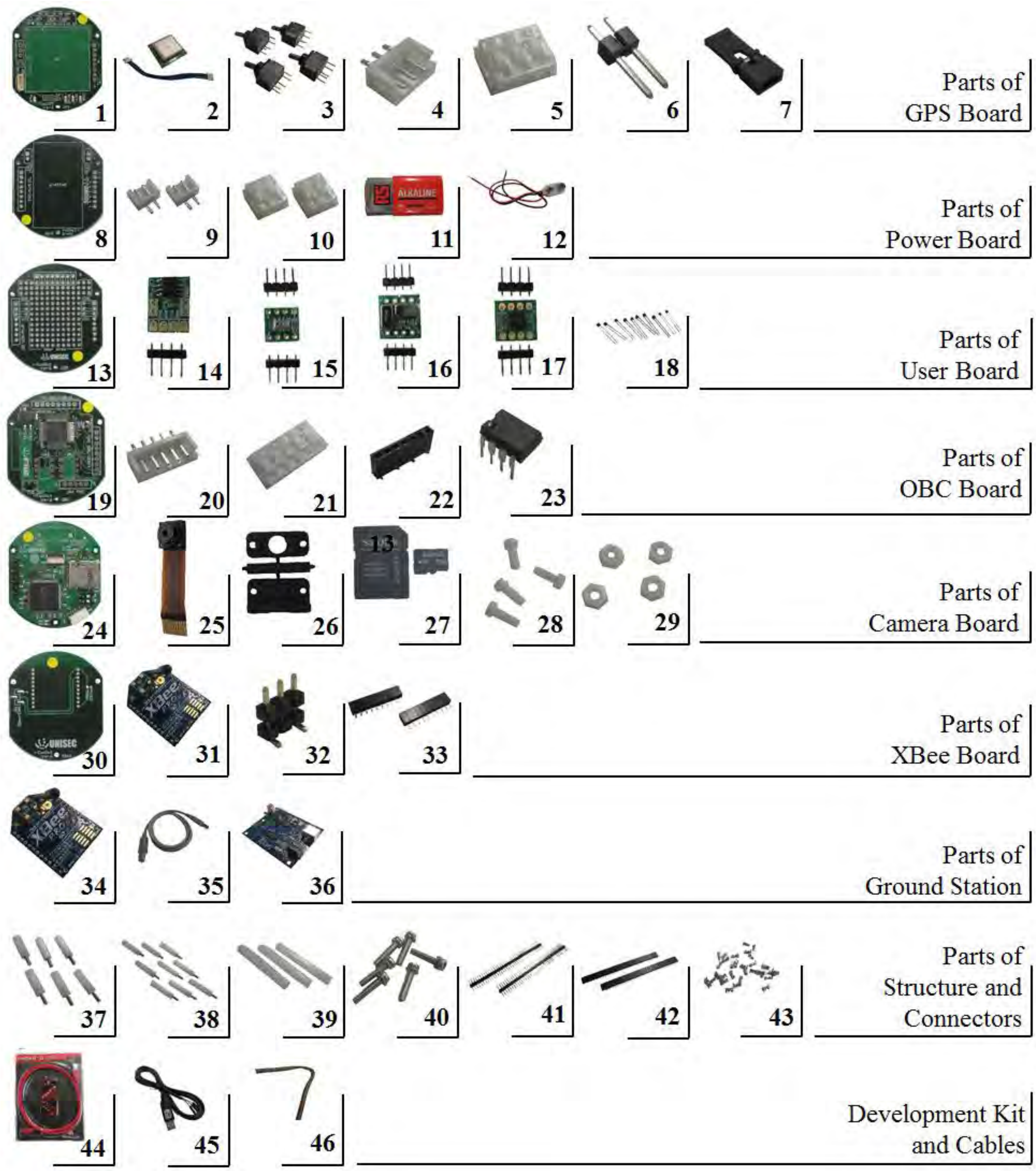


Figure I: Different parts i-CanSat kit.

Appendix B: Sample Program

```
/*
 * File:    main.c
 * Author:  H. Sahara @ TMU
 * Created on 2015/06/06
 */  //Changed register setting of ADCON1(ln133), PIE1(ln172), PIR1(ln184)

#include <stdio.h>
#include <stdlib.h>
#include <xc.h> //  For using the XC8 complier

// PIC16LF877A Configuration Bit Settings
__CONFIG(FOSC_HS  &  WDTE_OFF  &  PWRTE_OFF  &  BOREN_OFF  &  LVP_OFF  &  CPD_OFF  &  WRT_OFF  &
CP_OFF);  // PIC16LF877A Configuration
#define _XTAL_FREQ  10000000   // Crystal Frequency 10 MHz and its is loacted on the OBC
board.

// I/O port
#define READ    RB0
#define SEP     RB1
#define LED     RB3
#define CAM_IF1 RB4
#define CAM_IF2 RB5

#define INPUT   1  // USER Board Parameters
#define OUTPUT  0
#define HIGH    1
#define LOW     0
#define ACK     0

//#define RXBUF_LENGTH            16
#define GPS_LENGTH              76              //stores GPS data
#define EEPROM_INITIAL_ADDRESS  0x000000        //Defines where GPS data starts
#define EEPROM_FINAL_ADDRESS     0x01ffff          // when single EEPROM used. This line
must be commented when using 2 EEPROMs and it depends on the size of EEPROM
//#define EEPROM_FINAL_ADDRESS     0x03ffff       // when double EEPROMs used.(comment out
either one of them)

//volatile char rxbuf[RXBUF_LENGTH];
//volatile char rxbuf_wp, rxbuf_rp;

void hw_init(void)  //hardware PIC initialization
{
    /*--------------------------------------------------*/
    /* You may revise settings of the followings        */
    /*--------------------------------------------------*/
    // Port A (IO Analog Ports)
    TRISA0  = INPUT;                // AS YOU LIKE as IO0 or AN0. Set as AN0
    TRISA1  = INPUT;                // AS YOU LIKE as IO1 or AN1. Set as AN1
    TRISA5  = INPUT;                // AS YOU LIKE as IO2 or AN4. Set as AN4
    // Port E (IO Analog Ports)
    TRISE0  = INPUT;                // AS YOU LIKE as IO3 or AN5. Set as AN5
    TRISE1  = INPUT;                // AS YOU LIKE as IO4 or AN6. Set as AN6
    TRISE2  = INPUT;                // AS YOU LIKE as IO5 or AN7. Set as AN7
    /*--------------------------------------------------*/

    /* DO NOT REVISE ALL OF THE FOLLOWINS */

    /*-----*/
    /* I/O */
    /*--------------------------------------------------*/
    // Port A
    TRISA2  = OUTPUT;   RA2 = LOW;  // N/A in i-CanSat. Setting Only
```

```
    TRISA3  = OUTPUT;   RA3 = LOW;  // N/A in i-CanSat. Setting Only
    TRISA4  = OUTPUT;   RA4 = LOW;  // N/A in i-CanSat. Setting Only
    /* RA6, RA7 not equipped */

    // Port B
    TRISB0  = INPUT;                // I/O as READ (LOW:Flight, HIGH:Read)
    TRISB1  = INPUT;                // I/O as SEP (LOW:NotSEP, HIGH:Separated)
    TRISB2  = OUTPUT;   RB2 = LOW;  // N/A in i-CanSat. Setting Only
    TRISB3  = OUTPUT;   RB3 = LOW;  // I/O as LED (HIGH:Lighted, LOW:Unlit)
    TRISB4  = OUTPUT;   RB4 = LOW;  // I/O as CAM_IF1
    TRISB5  = OUTPUT;   RB5 = LOW;  // I/O as CAM_IF2
    TRISB6  = OUTPUT;   RB6 = LOW;  // N/A in i-CanSat. Setting Only
    TRISB7  = OUTPUT;   RB7 = LOW;  // N/A in i-CanSat. Setting Only

    // Port C
    TRISC0  = OUTPUT;   RC0 = LOW;  // N/A in i-CanSat. Setting Only
    TRISC1  = OUTPUT;   RC1 = LOW;  // N/A in i-CanSat. Setting Only
    TRISC2  = OUTPUT;   RC2 = LOW;  // N/A in i-CanSat. Setting Only
    TRISC3  = INPUT;                // I2C, SCL
    TRISC4  = INPUT;                // I2C, SDA
    TRISC5  = OUTPUT;   RC5 = LOW;  // N/A in i-CanSat. Setting Only
    TRISC6  = INPUT;                // USART TX
    TRISC7  = INPUT;                // USART RX

    // Port D
    TRISD0  = OUTPUT;   RD0 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD1  = OUTPUT;   RD1 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD2  = OUTPUT;   RD2 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD3  = OUTPUT;   RD3 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD4  = OUTPUT;   RD4 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD5  = OUTPUT;   RD5 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD6  = OUTPUT;   RD6 = LOW;  // N/A in i-CanSat. Setting Only
    TRISD7  = OUTPUT;   RD7 = LOW;  // N/A in i-CanSat. Setting Only

    // Port E
    /* RE3-RE7 not equipped */
    /*--------------------------------------------------*/

    /*-------*/
    /* USART */
    /*--------------------------------------------------*/
    // TXSSTA
    TXSTAbits.CSRC  = 0;    // Don't care
    TXSTAbits.TX9   = 0;    // 8-bit TXD
    TXSTAbits.TXEN  = 1;    // Enable TXD
    TXSTAbits.SYNC  = 0;    // Asynchronous mode
    TXSTAbits.BRGH  = 1;    // Low speed
    TXSTAbits.TRMT  = 0;    // TSR FULL
    TXSTAbits.TX9D  = 0;    // 0

    // RCSTA
    RCSTAbits.SPEN  = 1;    // Enable Serial Port
    RCSTAbits.RX9   = 0;    // 8-bit RXD
    RCSTAbits.SREN  = 0;    // Don't care
    RCSTAbits.CREN  = 1;    // Enable continuous RXD
    RCSTAbits.ADDEN = 0;    // Disable Address Detection
    RCSTAbits.FERR  = 0;    // Clear Flaming Error Bit
    RCSTAbits.OERR  = 0;    // Clear Overrun Error Bit
    RCSTAbits.RX9D  = 0;    // 0

    // Baud Rate
    SPBRG   = 64;           // 9600 bps = FOSC/(16*(X+1)) @ BRGH=1
                            //based on OBC's clock emitter; FOSC = Frequency of the
Crystal; X = SPBRG; BRGH is written in the PIC data sheet and it is equal to 1 for
PIC16LF877A
    /*--------------------------------------------------*/

```

```
    /*---------------------------------------------------*/
    /* ADC                                               */  //Analog / Digital Converter
    /*---------------------------------------------------*/
    // ADCON0
    ADCON0bits.ADCS1    = 1;        // FOSC/32
    ADCON0bits.ADCS0    = 0;
    ADCON0bits.CHS      = 0;        // CH0 (AN0), temporary
    ADCON0bits.GO       = 0;        // ADC STOP
    ADCON0bits.ADON     = 0;        // ADC OFF

    //ADCON1 //changed
    //ADCON1bits.ADFM     = 1;        // ADRESH:0000 00xx, ADRESL:xxxx xxxx
    //ADCON1bits.PCFG     = 0b0000;   //  AN0-AN7 available
    ADCON1 = 0b10000000;  //added
    /*---------------------------------------------------*/

    /*---------------------------------------------------*/
    /* I2C                                               */
    /*---------------------------------------------------*/
    // PIE1
    PIE1bits.SSPIE      = 0;    // SSP Interrupt

    // SSTPSTAT
    SSPSTATbits.SMP     = 0;    // 400kHz
    SSPSTATbits.CKE     = 0;    // for I2C

    // SSPCON
    SSPCONbits.SSPEN    = 1;        // RC3, RC4 as I2C
    SSPCONbits.SSPM     = 0b1000;   // I2C Master Mode

    // SSPADD
    SSPADD              = 12;       // 10MHz/400kHz/4-1 in PIC data sheet

    __delay_ms(10);
    /*---------------------------------------------------*/

    /*---------------------------------------------------*/
    /* Interrupt                                         */
    /*---------------------------------------------------*/
    // INTCON
    INTCONbits.GIE  = 0;    // Disable Global Interrupt
    INTCONbits.PEIE = 0;    // Disable peripheral Interrupt
    INTCONbits.T0IE = 0;    // Disable TMR0 Overflow Interrupt
    INTCONbits.INTE = 0;    // Disable RB0/INT External Interrupt
    INTCONbits.RBIE = 0;    // Disable RB Port Interrupt
    INTCONbits.T0IF = 0;    // Clear TMR0 Overflow Interrupt Bit
    INTCONbits.INTF = 0;    // Clear RB0/INT External Interrupt Bit
    INTCONbits.RBIF = 0;    // Clear RB Port Interrpt Bit

    // PIE1 //changed
    /*PIE1bits.PSPIE  = 0;    // Disable PSP R/W Interrupt
    PIE1bits.ADIE   = 0;    // Disable ADC Interrupt
    PIE1bits.RCIE   = 0;    // Disable USART RX Interrupt
    PIE1bits.TXIE   = 0;    // Disable USART TX Interrupt
    PIE1bits.SSPIE  = 0;    // Disable SSP Interrupt
    PIE1bits.CCP1IE = 0;    // Disable CCP1 Interrupt
    PIE1bits.TMR2IE = 0;    // Disable TMR2-PR2 Interrupt
    PIE1bits.TMR1IE = 0;    // Disable TMR1 Overflow Interrupt
     */
    PIE1 = 0b00000000; //added

    // PIR1 //changed
    /*PIR1bits.PSPIF  = 0;    // Clear PSP R/W Interrupt Bit
    PIR1bits.ADIF   = 0;    // Clear ADC Interrupt Bit
    PIR1bits.RCIF   = 0;    // Clear USART RX Buffer Full
    PIR1bits.TXIF   = 1;    // Stop USART TX
    PIR1bits.SSPIF  = 0;    // Clear SSP Interrupt Bit
```

```
    PIR1bits.CCP1IF = 0;    // Clear CCP1 Interrupt Bit
    PIR1bits.TMR2IF = 0;    // Clear TMR2 Interrupt Bit
    PIR1bits.TMR1IF = 0;    // Clear TMR1 Interrupt Bit
     */
    PIR1 = 0b00010000; //added

    // PIE2
    PIE2bits.EEIE   = 0;    // Disable EEPROM Write Interrupt
    PIE2bits.BCLIE  = 0;    // Disable Bus Collision Interrupt
    PIE2bits.CCP2IE = 0;    // Disable CCP2 Interrupt

    // PIR2
    PIR2bits.EEIF   = 0;    // Clear EEPROM Write Interrupt Bit
    PIR2bits.BCLIF  = 0;    // Clear Bus Collision Interrupt Bit
    PIR2bits.CCP2IF = 0;    // Clear CCP2 Interrupt Bit
    /*---------------------------------------------------*/
}

// 1 character TX
void put_char(unsigned char dum)
{
    while(!PIR1bits.TXIF);

    TXREG   = dum;
}

// String TX, uses put_char inside
void put_string(const unsigned char *dum)
{
    int cnt = 0;

    while(dum[cnt] != '\0')
        put_char(dum[cnt++]);
}

// I2C Send Start Condition, has to be used before I2C
char I2C_StartCondition(void)
{
    SSPCON2bits.SEN = 1;
    while(1) {
        if (PIR1bits.SSPIF == 1) {
            PIR1bits.SSPIF  = 0;
            return(1);
        }
        if (PIR2bits.BCLIF == 1) {
            PIR2bits.BCLIF  = 0;
            return(-1);
        }
    }
}

// I2C Send Stop Condition, has to be used after I2C
char I2C_StopCondition(void)
{
    SSPCON2bits.PEN = 1;
    while(1) {
        if (PIR1bits.SSPIF == 1) {
            PIR1bits.SSPIF  = 0;
            return(1);
        }
        if (PIR2bits.BCLIF == 1) {
            PIR2bits.BCLIF  = 0;
            return(-1);
        }
    }
}
```

```
// I2C Send 1 Byte Data
char I2C_Write(char dat)
{
    SSPBUF  = dat;

    while(!PIR1bits.SSPIF);
    PIR1bits.SSPIF  = 0;

    return((SSPCON2bits.ACKSTAT == 1)?  -1:1);
}

// I2C Get 1 Byte Data
char I2C_Read(char ack)
{
    SSPCON2bits.RCEN    = 1;

    while(!PIR1bits.SSPIF);
    PIR1bits.SSPIF  = 0;

    SSPCON2bits.ACKDT   = ack;
    SSPCON2bits.ACKEN   = 1;

  while(!PIR1bits.SSPIF);
    PIR1bits.SSPIF  = 0;

    return(SSPBUF);
}

// EEPROM Write
void EEPROM_Write(char str[])
{
    static long addr  = EEPROM_INITIAL_ADDRESS;
    long addr_eep;
    char ctrl, ret;
    char i  = 0;
    NOP();

    /*
     i-CanSat Ver.5 can possess 2 eeprom of 1024 kbits,
     * the first one has 00 (A1=0/A0=0) of Slave ID (Chip Select)
     * and the second one has 01 (A1=0/A0=1) of Slave ID (Chip Select)
     and each eeprom has 2 blocks of 512 kbits,
     * the former block has 0 B0=0 of Block Select
     * and latter block has 1 B0=1 of Block select
     That is,
     * the maximum address is 0x3ffff with 2 EEPROMs
     * 0x00000 - 0x0ffff is A1=0/A0=0 and B0=0
     * 0x10000 - 0x1ffff is A1=0/A0=0 and B1=1
     * 0x20000 - 0x2ffff is A1=0/A0=1 and B0=0
     * 0x30000 - 0x3ffff is A1=0/A0=1 and B1=1
     so that Block Select Bit (B0) and Chip Select Bits (A1/A0)
     should be set corresponding to the specified EEPROM address.
     *
     BUT,
     * i-CanSat Ver.4 can possess only 1 eeprom of 1024 kbits,
     * so that the maximum address is 0x1ffff with the eeprom
     * 0x00000 - 0x0ffff is A1=0/A0=0 and B0=0
     * 0x10000 - 0x1ffff is A1=0/A0=0 and B1=1
     BE CAREFUL!!!
    */

    while(str[i] != '\0') {
        if (addr <= EEPROM_FINAL_ADDRESS) {
            // Chip and Block Selections adjustment compared with
            // 0b0000 0011 0000 0000 0000 0000 = 0x030000;
            switch (addr & 0x030000) {

```

```
                // B0=0, A1/A0=00
                case 0x000000:
                    ctrl = 0b10100000;  addr_eep = addr;            break;
                // B0=1, A1/A0=00
                case 0x010000:
                    ctrl = 0b10101000;  addr_eep = addr - 0x010000; break;
/*
                // B0=0, A1/A0=01 //unncomment out if using two EEPROM
                case 0x020000:
                    ctrl = 0b10100010;  addr_eep = addr - 0x020000; break;
                // B0=1, A1/A0=01 //unncomment out if using two EEPROM
                case 0x030000:
                    ctrl = 0b10101010;  addr_eep = addr - 0x030000; break;
*/
                // Invalid Address
                default:
                    put_string("Invalid EEPROM Address\n");
            }

            START_EEPROM_WRITE:

            // Send Start Condition
            ret = I2C_StartCondition();
            if (ret == -1)  goto START_EEPROM_WRITE;

            // Send Control Byte
            ret = I2C_Write(ctrl);
            if (ret == -1)  goto START_EEPROM_WRITE;

            // Send Address High Byte
            ret = I2C_Write((char)((addr_eep & 0x00ff00) >> 8));
            if (ret == -1)  goto START_EEPROM_WRITE;

            // Send Address Low Byte
            ret = I2C_Write((char)(addr_eep & 0x0000ff));
            if (ret == -1)  goto START_EEPROM_WRITE;

            // Send Data
            ret = I2C_Write(str[i]);
            if (ret == -1)  goto START_EEPROM_WRITE;

            // Send Stop Condition
            ret = I2C_StopCondition();
            if (ret == -1)  goto START_EEPROM_WRITE;
            else {
                // in the case of success to write the data
                i++;
                addr++;
            }

            __delay_ms(5);
        }
    }
}


// EEPROM Read
char EEPROM_Read(void)
{
    long addr, addr_eep;
    char ctrl, data;
    char ret;

    __delay_ms(1000);

    for(addr=EEPROM_INITIAL_ADDRESS; addr<=EEPROM_FINAL_ADDRESS; addr++) {

```

```
        START_EEPROM_READ:

        LED = ~LED;

        switch (addr & 0x030000) {

            // B0=0, A1/A0=00
            case 0x000000:
                ctrl = 0b10100000;  addr_eep = addr;            break;
            // B0=1, A1/A0=00
            case 0x010000:
                ctrl = 0b10101000;  addr_eep = addr - 0x010000; break;
/*
             // B0=0, A1/A0=01
            case 0x020000:
                ctrl = 0b10100010;  addr_eep = addr - 0x020000; break;
            // B0=1, A1/A0=01
            case 0x030000:
                ctrl = 0b10101010;  addr_eep = addr - 0x030000; break;
*/
             // Invalid Address
            default:
                put_string("Invalid EEPROM Address\n");
        }

        // Send Start Condition
        ret = I2C_StartCondition();
        if (ret == -1)  goto START_EEPROM_READ;

        // Send Control Byte
        ret = I2C_Write(ctrl);
        if (ret == -1)  goto START_EEPROM_READ;

        // Send Address High Byte
        ret = I2C_Write((char)((addr_eep & 0x00ff00) >> 8));
        if (ret == -1)  goto START_EEPROM_READ;

        // Send Address Low Byte
        ret = I2C_Write((char)(addr_eep & 0x0000ff));
        if (ret == -1)  goto START_EEPROM_READ;

        // Send Stop Condition
        I2C_StopCondition();

        switch (addr & 0x030000) {

            // B0=0, A1/A0=00
            case 0x000000:
                ctrl = 0b10100001;  break;
            // B0=1, A1/A0=00
            case 0x010000:
                ctrl = 0b10101001;  break;
/*
            // B0=0, A1/A0=01
            case 0x020000:
                ctrl = 0b10100011;  break;
            // B0=1, A1/A0=01
            case 0x030000:
                ctrl = 0b10101011;  break;
*/
             default:
                put_string("Invalid EEPROM Address\n");
        }

        // Send Start Condition
        ret = I2C_StartCondition();
        if (ret == -1)  goto START_EEPROM_READ;
```

```
        // Send Control Byte
        ret = I2C_Write(ctrl);
        if (ret == -1)  goto START_EEPROM_READ;

        // Read Data
        data    = I2C_Read(1);    // NCK
        NOP();

        // Send Stop Condition
        ret = I2C_StopCondition();
        if (ret == -1)  goto START_EEPROM_READ;

        // Send the data to USART
        put_char(data);
        __delay_ms(10);
    }

}


/*
// Interrupt
void interrupt play(void)
{
    INTCONbits.GIE  = 0;    // Disable Global Interrupt
    LED = HIGH;

    // USART RX
    if (PIR1bits.RCIF) {
        PIR1bits.RCIF   = 0;    // Clear USART RX Interrupt Bit

        // Error Clear
        if (RCSTAbits.FERR || RCSTAbits.OERR) {
            RCSTAbits.CREN  = 0;
            __delay_us(10);
            RCSTAbits.CREN  = 1;
            rxbuf[rxbuf_wp] = '?';
        }
        else
            rxbuf[rxbuf_wp] = RCREG;

        rxbuf_wp++;
        rxbuf_wp %= RXBUF_LENGTH;
    }

    LED = LOW;
    INTCONbits.GIE  = 1;    // Enable Global Interrupt
}
*/

float c2f(int c, int sbits, int nbits, int fbits, int zpad)  //for pressure sensor, data
comes in 16bit, fractional part is different, so it has to be converted
{
    int cd;
    cd = (c >> 16 - nbits) / (1 << fbits + zpad); // ">>" shifts right, "<<" shifts
left(increases bit size)
    return((float)cd);
}

void main(void)  //
{
    char cnt;
    unsigned char ret;
    char GPS[GPS_LENGTH], num_sat;  //creates array to store GPS data
    char AD[8];
    char f_GPS  = 0;  //flag of GPS
```

```
    char gps_wp = 0;
    char rxbuf;
    int tdataint;
    float tmp;
    int padc;
    int tadc;
    int cd[8];
    float a0;
    float b1;
    float b2;
    float c12;
    float pcomp = 0;
    float pre;
    /*----------------------------------------------------*/
    /* H/W Initialization                                 */
    /*----------------------------------------------------*/
    __delay_ms(10);  // Wait for stable power supply
    hw_init();
    ADCON0bits.ADON       = 1;           // ADC ON   //turns on AD convertor, has to be commented out for your first trial without option
    /*----------------------------------------------------*/

    /*----------------------------------------------------*/
    /* Parameters Initialization                          */
    /*----------------------------------------------------*/
//    for(cnt=0; cnt<RXBUF_LENGTH; cnt++) rxbuf[cnt]  = 0;
    for(cnt=0; cnt<GPS_LENGTH;  cnt++)  GPS[cnt]    = 0;
//    rxbuf_wp    = 0;
//    rxbuf_rp    = 0;
    /*----------------------------------------------------*/

    /*----------------------------------------------------*/
    /* Welcome Message                                    */
    /*----------------------------------------------------*/
    //put_string("Welcome to i-CanSat!!\n");
    /*----------------------------------------------------*/

        LED = HIGH;
        __delay_ms(500);
        LED = LOW;


    /*----------------------------------------------------*/
    /* Flight Mode                                        */
    /*----------------------------------------------------*/
    // SEP?????????????????????????????????????????????????
    if (READ == LOW) {

/*
        // Enable Interrupt
        PIE1bits.RCIE   = 1;    // Enable USART RX Interrupt
        INTCONbits.PEIE = 1;    // Enable Peripheral Interrupt (USART RX)
        INTCONbits.GIE  = 1;    // Enable Global Interrupt (ALL INTERRUPT)
*/
        // CANCAM, Start Continuous Image Capture
        ret = I2C_StartCondition();     // Start Condition
        ret = I2C_Write(0b00010000);    // ID=8 (in the upper 7 bits)
        ret = I2C_Write(0x73);          // ADDRESS
        ret = I2C_Write(0x00);              // DATA, Dummy   //placed to have connection finished properly.
        ret = I2C_StopCondition();      // Stop Condition
        __delay_ms(10);

        // Start I2C Sensor initialization.
        //--------------------------------
        // Temperature Sensor
        ret = I2C_StartCondition();     // Start Condition
```

```c
        ret = I2C_Write(0b10010000);    // ADDRESS
        ret = I2C_Write(0x03);          // Configuration register
        ret = I2C_Write(0b10000000);    // Normal mode 16bit
        ret = I2C_StopCondition();      // Stop Condition
        __delay_ms(10);


        // Pressure Sensor
        ret = I2C_StartCondition();  // Start Condition
        ret = I2C_Write(0xc0);
        ret = I2C_Write(0x04);
        ret = I2C_StopCondition();
        __delay_ms(10);

        ret = I2C_StartCondition();  // Start Condition
        ret = I2C_Write(0xc1);       // ADDRESS + Read mode
        __delay_ms(10);
        cd[1] = I2C_Read(0x00) << 8;
        cd[1] |= I2C_Read(0x00) ;
        cd[2] = I2C_Read(0x00) << 8;
        cd[2] |= I2C_Read(0x00) ;
        cd[3] = I2C_Read(0x00) << 8;
        cd[3] |= I2C_Read(0x00) ;
        cd[4] = I2C_Read(0x00) << 8;
        cd[4] |= I2C_Read(0x01) ;
        ret = I2C_StopCondition();
        __delay_ms(10);

        //a0 = c2f(cd[1], 1, 16, 3, 0);
        //b1 = c2f(cd[2], 1, 16, 13, 0);
        //b2 = c2f(cd[3], 1, 16, 14, 0);
        //c12 = c2f(cd[4], 1, 14, 13, 9);
        a0 = (float) cd[1] / (1 << 16 -16 + 3);
        b1 = (float) cd[2] / (1 << 16 -16 + 13);
        b2 = (float) cd[3] / (1 << 16 -16 + 14);
        c12 = (float) cd[4] / 16777216.0; //(1 << 16-14 + 13 + 9)
        __delay_ms(10);

        // Complete I2C Sensor Initialization
        // Main Loop
        while(1) {

            // USART RX
            // GPS????????????
            if (PIR1bits.RCIF) {
                PIR1bits.RCIF   = 0;    // Clear USART RX Interrupt Bit
                // Error Clear
                if (RCSTAbits.FERR || RCSTAbits.OERR) {
                    RCSTAbits.CREN  = 0;
                    __delay_us(10);
                    RCSTAbits.CREN  = 1;
                    rxbuf   = '?';
                }
                else {
                    rxbuf   = RCREG;
                }

                if (rxbuf == '?') {
                    LED = HIGH;
                    __delay_ms(50);
                    LED = LOW;
                    __delay_ms(50);
                }

                // Find the start of GPS data
                // GPS??????????????????????
                if (rxbuf == '$') { // if the first Character is "$"
```

```
                    gps_wp  = 0;
                    f_GPS   = 1;
                }

                // Find the terminal of GPS data
                // GPS??????????????????????????
                if (f_GPS == 1) {
                    GPS[gps_wp] = rxbuf;  //saves GPS data into an array
                    if (GPS[gps_wp] == 0x0a) {
                        GPS[gps_wp+1]   = '\0'; //return
                        f_GPS   = 2;
                    }
                    else {
                        gps_wp++;
                    }
                }
            }

            // GPS Data Available
            // This routine is activated in each 1 seconds (1Hz)
            // because GPS data can be available in 1Hz.
            // GPS???????????????????
            if (f_GPS == 2) {       // write to the EEPROM if f_GPS = 2
                // Count the number of GPS satellites with LED
                // 0          1          2          3          4          5          6
7
                //
012345678901234567890123456789012345678901234567890123456789012345678901
                //
$GPGGA,115946.000,0000.0000,N,00000.0000,E,0,00,0.0,0.0,M,0.0,M,,0000*63
                num_sat = (GPS[45] - 0x30) * 10 + (GPS[46] - 0x30);
                // For example, the ASCII of '1' is 0x31,
                // so that the number of 1 can be expressed by ('1' - 0x30).
                if (num_sat > 4) {
                    num_sat = 4;
                    LED = HIGH;
                }

                // Output the GPS data
                // GPS????EEPROM?XBee??????
                EEPROM_Write(GPS);  //writes data in to the EEPROM
                if (SEP == HIGH)  //if it is in flight mode
                    put_string(GPS);  //the cansat sends data
                f_GPS = 0;

 // --------------------  Sensors readings ------------------------------------
                // ADC
                LED = HIGH;

                //IO0=AN0
                /* ADCON0bits.CHS = 7;  ADCON0bits.GO = 1;  while(ADCON0bits.GO); //CHS...
designates where the data is stored
                sprintf(AD, "AD0:%d,", ADRESH*256+ADRESL);  //"AD0:xxx" will be stored in
AD
                if (SEP == HIGH)    put_string(AD);  //AD will be displayed
                EEPROM_Write(AD); __delay_ms(10);  //AD will be shown */

                //IO4=AN6 gyro 2
                ADCON0bits.CHS = 5;  ADCON0bits.GO = 1;  while(ADCON0bits.GO);
                sprintf(AD, "%d,", ADRESH*256+ADRESL);
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD); __delay_ms(10);

                // IO3=AN5 gyro 1
                ADCON0bits.CHS = 6;  ADCON0bits.GO = 1;  while(ADCON0bits.GO);
                sprintf(AD, "%d,", ADRESH*256+ADRESL);
                if (SEP == HIGH)    put_string(AD);
```

```
                EEPROM_Write(AD); __delay_ms(10);


                // IO0=AN0 accel x
                ADCON0bits.CHS = 0;  ADCON0bits.GO = 1;  while(ADCON0bits.GO);
                sprintf(AD, "%d,", ADRESH*256+ADRESL);
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD); __delay_ms(10);

                // IO1=AN1 accel y
                ADCON0bits.CHS = 1;  ADCON0bits.GO = 1;  while(ADCON0bits.GO);
                sprintf(AD, "%d,", ADRESH*256+ADRESL);
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD); __delay_ms(10);

                // IO2=AN4 accel z
                ADCON0bits.CHS = 4;  ADCON0bits.GO = 1;  while(ADCON0bits.GO);
                sprintf(AD, "%d,", ADRESH*256+ADRESL);
                //sprintf(AD, "AD7:%d%c%c", ADRESH*256+ADRESL, 0x0d, 0x0a); // last two letters are for return
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD); __delay_ms(10);



                // Temperature Sensor
                ret = I2C_StartCondition();      // Start Condition
                ret = I2C_Write(0b10010001);     //ADSRESS READ mode 13bit
                //ret = I2C_Write(0x48+1);       // ADDRESS Read mode
                tdataint = I2C_Read(0x00) << 8;  // Read upper
                tdataint |= I2C_Read(0x01);      // Read low
                ret = I2C_StopCondition();       // Stop Condition

                tdataint >>=3;
                if(tdataint & (0x8000 >> 3)) {
                    tdataint = tdataint - 8192;
                }

                tmp = (float)tdataint/16.0;
                sprintf(AD, "T=%f%c%c", tmp, 0x0d, 0x0a);
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD);
                __delay_ms(10);


                //Pressure sensor
                ret = I2C_StartCondition();
                ret = I2C_Write(0xc0);       // ADDRESS + Write mode
                ret = I2C_Write(0x12);       // Start Conversions
                ret = I2C_Write(0x00);       // Send dummy data
                ret = I2C_StopCondition();
                __delay_ms(10);

                ret = I2C_StartCondition();  // Start Condition
                ret = I2C_Write(0xc0);
                ret = I2C_Write(0x00);
                ret = I2C_StopCondition();
                __delay_ms(10);

                ret = I2C_StartCondition();  // Start Condition
                ret = I2C_Write(0xc1);       // ADDRESS + Read mode
                __delay_ms(10);
                padc = I2C_Read(0x00) << 2;
                padc |= I2C_Read(0x00) >> 6;
                tadc = I2C_Read(0x00) << 2;
                tadc |= I2C_Read(0x01) >> 6;
                ret = I2C_StopCondition();
```

```
                __delay_ms(10);

                pcomp = a0 + (b1 + c12 *tadc) * (float)padc + b2 * (float)tadc;
                __delay_ms(10);
                pre  = pcomp * 650 / 1023 + 500;
                __delay_ms(10);
                sprintf(AD, "%f%c%c", pre, 0x0d, 0x0a);
                if (SEP == HIGH)    put_string(AD);
                EEPROM_Write(AD);
                __delay_ms(10);


                LED = LOW;
            // ---------  Compete Sensor Readings -----------------------------
             }
        }
    }
    /*--------------------------------------------------*/

    /*--------------------------------------------------*/
    /* Read Mode                                        */
    /*--------------------------------------------------*/
    // SEP????????????????????????????????????????????????
    // EEPROM???????????????????XBee??????????????????????
    else {
        while(1) {
            put_string("Start EEPROM Read\n");
            EEPROM_Read();
            put_string("End EEPROM Read\n");
        }
    }
    /*--------------------------------------------------*/

}
```

Appendix C: USR Board

User defined sensor modules can be installed in the **USR** board. In this appendix the circuit diagram and the assembly of four different sensors modules are described. They are as follows:

1. 3-axes Accelerometer module, ADXL-335, its circuit diagram is shown in Figure C-1.
2. 2-axes gyroscope sensor module, ENC-03R, its circuit diagram is shown in Figure C-2
3. Temperature sensor module, ADT7410, its circuit diagram is shown in Figure C-3.
4. Barometric sensor module, MPL115A2, its circuit diagram is shown in Figure C-4.

All the sensors modules can be powered-up from the i-CanSat power line which is 3.3 volt. Due the memory limitations of the OBC only three sensors modules can work simultaneously. The reader must refer to the sample program described in appendix A and the sensor module specification sheets form more information about acquiring data from these sensor.

Figure C-5 shows the circuit diagram of the assembled **USR** board. Photo of the soldered **USR** board is shown in Figure C-6.

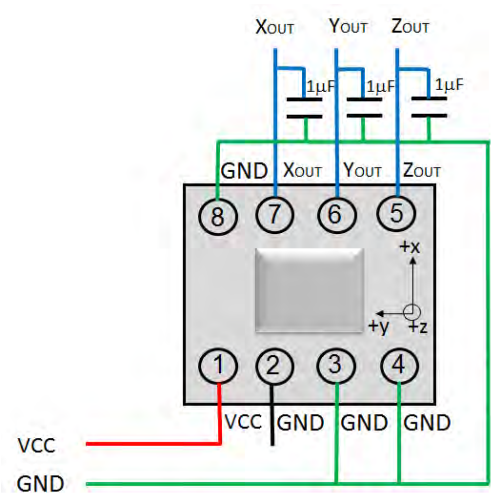


Figure C-1: Circuit diagram of 3-axes accelerometer sensor module, ADXL-335

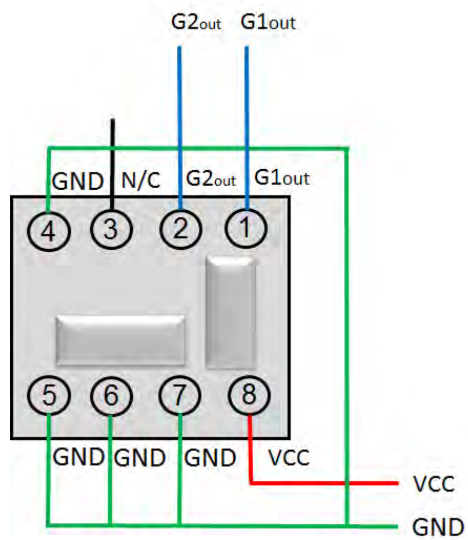


Figure C-2: Circuit diagram of 2-axes gyro sensor module, ENC-03R.

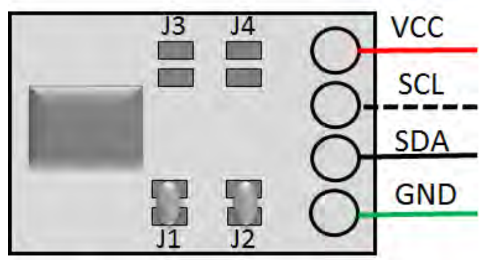


Figure C-3: Circuit diagram of temperature sensor module, ADT7410.

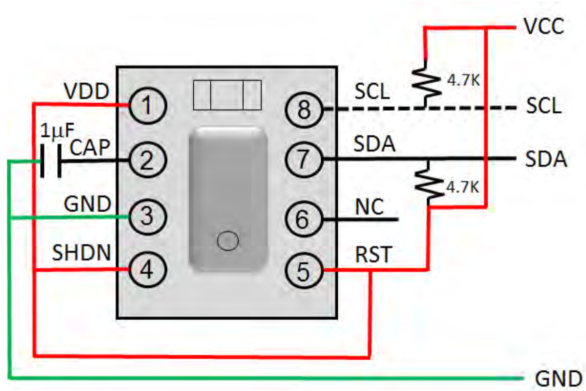


Figure C-4: Circuit diagram of barometric sensor module, MPL115A2.

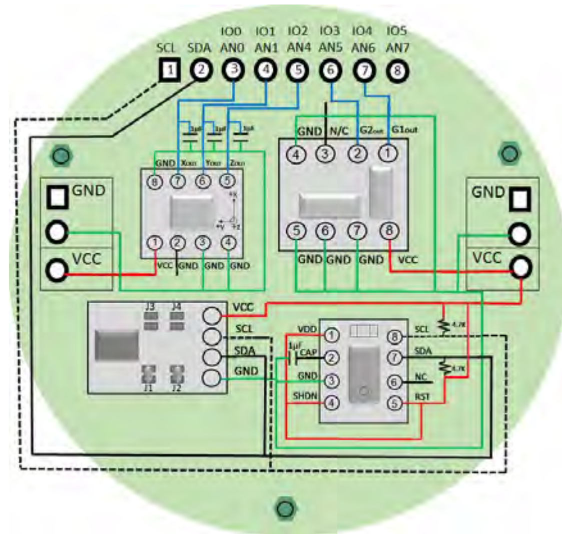
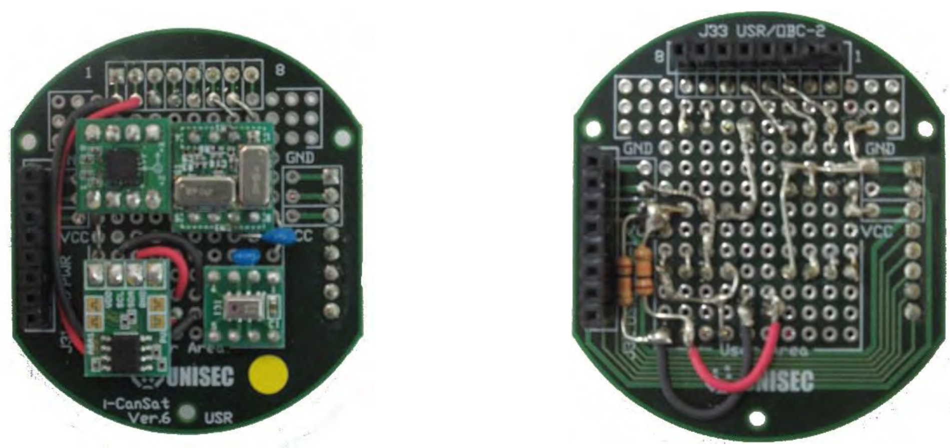


Figure C-5: Circuit diagram of the **USR** board include the accelerometer, gyro, temperature and barometric sensor modules.



a) Upper surface b) Lower surface

Figure C-6: Complete the installation and soldering works of the **USR** board.