UNIVERSITAS INDONESIA

PERANCANGAN *HARDWARE* & *SOFTWARE ON-BOARD DATA HANDLING* UI-SAT DENGAN PROSESOR LPC1768 ARM CORTEX M-3

SKRIPSI

MOHAMMAD GAVIN RENALDI RIPHARBOWO

0806455332

FAKULTAS TEKNIK UNIVERSITAS INDONESIA

DEPARTEMEN TEKNIK ELEKTRO

DEPOK

JUNI 2012

UNIVERSITAS INDONESIA

PERANCANGAN *HARDWARE* & *SOFTWARE ON-BOARD DATA HANDLING* UI-SAT DENGAN PROSESOR LPC1768 ARM CORTEX M-3

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana

MOHAMMAD GAVIN RENALDI RIPHARBOWO

0806455332

FAKULTAS TEKNIK UNIVERSITAS INDONESIA

DEPARTEMEN TEKNIK ELEKTRO

DEPOK

JUNI 2012

HALAMAN PERNYATAAN ORISINALITAS

Seminar ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan benar.

Nama	:	Mohammad Gavin Renaldi Ripharbowo
NPM	:	0806455832
Tanda Tangan	:	
Tanggal	:	7 Juli 2012

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama : Mohammad Gavin Renaldi Ripharbowo

NPM : 0806455332

Program Studi : Teknik Elektro

Judul Skripsi : Perancangan Hardware & Software On-Board data Handling UI-SAT dengan Prosesor LPC1768 ARM Cortex M-3

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar sarjana pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Ir. Muhammad Asvial M. Eng. PhD. ( )
NIP : 196804061994031001

Penguji 1 : Ir. Gunawan Wibisono M.Sc., Ph.D. ( )
NIP : 196602221991031003

Penguji 2 : Dr. Ir. Arman D. Diponegoro ( )
NIP : 194811131985031001

Ditetapkan di : Depok

Tanggal : 6 Juli 2012

UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan atas rahmat dan hidayah Allah SWT dari awal saya memasuki dunia kuliah sampai semester ini berjalan sehinggal terselesaikannya seminar ini. Semua hal ini juga tidak lepas dari bantuan yang diberikan oleh berbagai pihak di sekitar saya. Oleh karena itu, penulis mengucapkan terima kasih sebesar-besarnya kepada :

(1) Bapak Dr. Ir. Muhammad Asvial M,Eng, selaku dosen pembimbing yang telah membantu judul seminar ini dari proyek nano satelit di Universitas-universitas terkemuka di Indonesia

(2) Bapak Dr. Ir. Purnomo Sidi Priambodo, MSEE, yang telah memberikan kemudahan untuk mengguinakan fasilitas Laboratorium Elektronika Departemen Teknik Elektro Universitas Indonesia, khususnya sarana computer dan perangkat lunakn untuk melakukan simulasi.

(3) Merry Oerip dan Haryanto Rahardjo, selaku orang Tua saya dalam menyemangati dan mendukung saya sepenuhnya untuk menyelesaikan seminar ini.

(4) Kepada 3 saudari kandung saya: Yardian Wensdi. Nadya Meidianti, dan Dania Adela.

(5) Syifa Aulia, Galih Dewandaru, Nanda Gustiyanto, Wahyu Kuncoro Adhi, Tb. Tidra Barezna, Prayudo K.Wardhanan, Mohammad Wahyu Santoso, Zesyara, Naqib, Steward Augusto, Muhammad Idham Habibie, Muhammad Iqbal, Harland F. Amin, Asisten Lab Telekomunikasi, dan teman satu angkatan Elkom 2008 yang turut membantu dan bersama-sama kuliah dalam 3 tahun ini.

Akhir kata, penulis berharap bahwa Allah SWT akan membalas semua kebaikan yang telah diberikan oleh pihak-pihak di sekitar saya.

Depok, 4 Juli 2012

Mohammad Gavin Renaldi Ripharbowo

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Indonesia, saya bertanda tangan di bawah ini :

Nama : Mohammad Gavin Renaldi Ripharbowo

NPM : 080645533 2

Program Studi : Teknik Elektro

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exlusive Royalty-Free Right)*** atas karya ilmiah saya yang berjudul :

PERANCANGAN *HARDWARE & SOFTWARE ON-BOARD DATA HANDLING* UI-SAT DENGAN MPROSESOR LPC1768 ARM CORTEX M-3

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non Ekslusif ini Universitas Indonesia berhak menyimpan, mengalih media/format-kan, mengelola dalam bentuk pangkalan data (database), merawat, dan mempublikasikan seminar saya selama tetap mencamtumkan nama saya sebagai penulis/pencipta sebagai pemegang Hak Cipta

Demikian pernyataan ini saya buat dengan sebenar-benarnya

Dibuat di : Depok

Pada tanggal : 12 Juli 2012

Yang Menyatakan

(Mohammad Gavin Renaldi Ripharbowo)

ABSTRAK

Nama : Mohammad Gavin Renaldi Ripharbowo

Program Studi : Teknik Elektro

Judul : Perancangan *Hardware* dan *Software On-Board Data Handling* UI-SAT dengan Prosesor LPC1768 ARM Cortex M-3

Telekomunikasi pada abad ke-21 di dunia sudah sangat penting untuk komunikasi antarorang ataupun antara gadget. Salah satu teknologi tersebut adalah satelit. Salah satu komponen penting pada nanosatelit adalah OBDH/ On-Board Data Handling berfungsi sebagai media yang berfungsi untuk fungsi perintah dan pengolahan data. Namun, diperlukan adanya TTC (Telemtery, Tracking, and Command)yang berguna untuk menghubungkan antara ground station dengan nanosatelit. Adapun, pada penelitian kali ini, penelitian ini akan memfokuskan pada perancangan Hardware dan Software *On-Board Data Handling* dan hubungannya dengan TTC khususnya TNC(*Terminal Node Controller*)

Kata Kunci : OBDH, Nanosatelit, UI-SAT, TNC, IiNUSAT

ABSTRACT

Name : Mohammad Gavin Renaldi Ripharbowo
Study Program : Teknik Elektro
Title : Design of Hardware and Software Onboard Data Handling UI-SAT-Based with LPC1768 Arm Cortex M-3

Telecommunications in the 21st century the world has been very important for communication between individuals or between gadgets. One such technology is the satellite. One important component is the nanosatelit OBDH / On-Board Data Handling serves as a medium that serves for the command and data processing functions. However, it is necessary to TTC (Telemtery, Tracking, and Command) is useful for connecting between the ground station with nanosatelit. Meanwhile, in the present study, the research will focus on designing hardware and software On-Board Data Handling and its relationship with the TTC, especially TNC (Terminal Node Controller)

Keyword : OBDH, Nanosatelit, UI-SAT, TNC, IiNUSAT

DAFTAR ISI

HALAMAN JUDUL ..... **ii**
HALAMAN PERNYATAAN ORISINALITAS ... Error! Bookmark not defined.
HALAMAN PENGESAHAN ..... Error! Bookmark not defined.
UCAPAN TERIMA KASIH ..... **iv**
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS. Error! Bookmark not defined.
ABSTRAK ..... **vii**
ABSTRACT ..... **ix**
DAFTAR ISI ..... **x**
DAFTAR GAMBAR ..... **xiii**
DAFTAR TABEL ..... **xv**
BAB I. PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Tujuan 4
1.3 Batasan Masalah 4
1.4 Metodologi Penelitian 4
1.5 Sistematika Penulisan 4
BAB II. ON-BOARD DATA HANDLING UI-SAT DAN TERMINAL NODE CONTROLLER PADA TELEMETRY, TRACKING, AND COMMAND IiNUSAT **6**
2.1 NanoSatelit 6
2.2 Orbit Satelit 7
2.3 Frekuensi Satelit 8
2.4 Sistem Satelit 9
2.5 On Board Data Handling 10
2.5.1 Mikroprosesor ARM Cortex M3 12
2.5.2 Bus UARTs (Universal Asynchronous Receiver Transmitter) 14
2.5.3 DB9 (F/M) 14
2.5.4 RS-232 16
2.6 Telemetry, Tracking, And Command (TTC) 16
2.6.1 *Radio Frequency* 18
2.6.2 *Terminal Node Controller* (TNC) 18

2.7 Deskripsi Sub Sistem Nano Satelit Lainnya yang Saling Berintegrasi dengan OBDH UI-SAT 18
2.7.1 *Payload Communication* 18
2.7.2 *Electrical Power System* (EPS) 19
2.7.3 *Altitude Determination and Control System (ADCS)* 20
BAB III. INTERKONEKSI ON-BOARD DATA HANDLING UI-SAT DENGAN TERMINAL NODE CONTROLLER IiNUSAT 22
3.1 Tahap Perancangan On Board Data Handling UI-Sat 22
3.1.1 Perancangan LPC1768 pada OBDH UI-SAT 22
3.1.2 Perancangan MAX3232EUE pada OBDH UI-SAT 27
3.1.3 Perancangan ISP(*In-System Programming*) dan *Reset Schematic* pada OBDH UI-SAT 30
3.1.4 Perancangan Power Supply 3.3V 31
3.1.5 Perancangan JTAG (*Joint Test Action Group*) 32
3.2 Simulasi Rangkaian pada Software Simulasi Proteus 33
3.3 Interface OBDH UI-SAT dengan Subsistem TTC Iinusat 37
3.3 Interface OBDH UI-SAT dengan Subsistem NanoSatelit Lainnya 37
3.4 Algoritma Pemograman On-Board Data Handling dengan TTC IInusat. 38
3.4.1 Algoritma Mode Ambil Data dari TTC 38
3.4.2 Algoritma Kirim Data ke TTC 39
BAB IV. ANALISIS KINERJA INTERKONEKSI OBDH UI-SAT DENGAN TTC IiNUSAT 40
4.1 Fitur-Fitur Mikroprosesor LPC1768 yang Digunakan pada OBDH UI-SAT 40
4.1.1 Register yang Digunakan Pada LPC1768 40
4.1.2 BootCode, Startup Code, Main Code, dan Header Files 41
4.2 Fitur-fitur Mikroprosesor ATMEGA1280 yang Digunakan pada TNC TTC Iinusat 42
4.2.1 Register yang Digunakan Pada ATMEGA1280 42
4.2.2 BootCode, Startup Code, Main Code, dan Header Files 43
4.3 Analisis Command and Response OBDH UI-SAT dari Perintah Ambil Data Dari Terminal Node Controller pada Tracking, Telemetry and Command Iinusat ke OBDH 43
4.4 Analisis Command and Response OBDH UI-SAT dari Perintah Kirim Data ke Terminal Node Controller pada Tracking, Telemetry, and Command Iinusat 47

4.5 Analisis Kesalahan Informasi Terminal Node Controller Tracking, Telemertry, and Command IiNUSAT dengan OBDH UI-SAT.....50
4.6 Analisis Interkoneksi TNC dengan Modem MO-96.....53
BAB V. 55
KESIMPULAN 55
DAFTAR REFERENSI 56
LAMPIRAN.....57

DAFTAR GAMBAR

Gambar 2. 1 CubeSat....7
Gambar 2. 2 Orbit Satelit....8
Gambar 2. 3 Alokasi Frekeunsi....8
Gambar 2. 4 Sistem Satelit....10
Gambar 2. 5 Struktur Konfigurasi Pin LPC176x untuk Packaging FBD100....14
Gambar 2. 6 Konektor DB9-Female....15
Gambar 2. 7 Koneksi RS-232 *Null-Modem Cables*....16
Gambar 2. 8 Sistem payload IiNUSAT-1....19
Gambar 2. 9 RF Front End IiNUSAT-1....19
Gambar 2. 10 Sistem Electrical Power System secara umum....20
Gambar 2. 11 Desain Sistem Altitude Determination and Control System....21
Gambar 3.1 Perancangan LPC1768FBD100 pada *software* Altium Bagian 1.26
Gambar 3.2 Perancangan LPC1768FBD100 pada *software* Altium Bagian 2.27
Gambar 3.3 Perancangan LPC1768FBD100 pada *software* Altium Bagian 3.27
Gambar 3.4 Perancangan MAX.3232EUE pada Interface DB9/F dengan LPC1768FBD100 pada ADCS dan TTC....28
Gambar 3.5 Perancangan MAX3232EUE pada Interface DB9/F dengan LPC1768FBD100 pada EPS dan Payload....28
Gambar 3.6 Perancangan Rangkaian ISP dan INT0....30
Gambar 3.7 Perancangan Rangkaian Reset....31
Gambar 3 8 Perancangan Power Supply 3.3 V....32
Gambar 3 9 Perancangan Rangkaian Joint Test Action Group....32
Gambar 3 10 Simulasi Rangkaian ISP dan INT0....33
Gambar 3 11 Simulasi Rangkaian ISP dan INT0 pada Saat W1 dan W2 Keadaan Hubung Buka....34
Gambar 3 12 S.imulasi Rangkaian ISP dan INT0 pada Saat Jumper W1 dan W2 pada Keadaan Hubung Singkat....34
Gambar 3 13 Simulasi Rangkaian ISP dan INT0 pada saat Jumper W1 Hubung Buka dan W2 Hubung Singkat....34
Gambar 3 14 Simulasi Rangkaian ISP dan INT0 pada saat Jumper W1 Hubung Singkat dan W2 Hubung Buka....35
Gambar 3 15 Simulasi Rangkaian Reset....35
Gambar 3 16 Simulasi Rangkaian Reset pada Jumper S1 dan W1 Keadaan Hubung Buka....36
Gambar 3 17 Simulasi Rangkaian Reset pada Jumper S1 dan W1 Keadaan Hubung Singkat....36
Gambar 3 18 Simulasi Rangkaian Reset pada Jumper S1 Hubung Buka dan W1 Keadaan Hubung Singkat....36
Gambar 3 19 Simulasi Rangkaian Reset pada Jumper S1 Keadaan Hubung Singkat dan Jumper W1 Keadaan Hubung Buka....37
Gambar 4. 1 FlowChart Ambil Data RF....45
Gambar 4. 2 Hasil UjiCoba pada PC1 pada saat perintah Ambil Data....46
Gambar 4. 3 Hasil Ujicoba pada PC2 pada saat perintah Ambil Data....46
Gambar 4. 4 FlowChart Algoritma Kirim Data RF....48
Gambar 4. 5 Hasil UjiCoba PC1 pada saat perintah Kirim Data RF....49
Gambar 4. 6 Hasil UjiCoba PC2 pada saat perintah Kirim Data RF....49

Gambar 4. 7 Hasil Uji Coba Perintah Kirim Data Dengan Baud Rate 4800 bps.. 51
Gambar 4. 8 Hasil UjiCoba Perintah Kirim Data dengan Baud Rate 57.600 bps. 52
Gambar 4. 9 Hasil Uji Coba Perintah Kirim Data dengan baud Rate 115.200 bps 52
Gambar 4. 10 Hasil Uji Coba Perintah Kirim Data dengan Baud Rate sebesar 203.400 bps 53
Gambar 4. 11 Konversi antara Sinyal Digital dan Sinyal Analog 54
Gambar 4. 12 Perubahan Level Tegangan 54

DAFTAR TABEL

Tabel 2.1 Konfigurasi konektor DB9 Female.....15
Tabel 2.2 Konfigurasi Konektor DB9/F 16
Tabel 3.1 Perancangan Pin LPC1768 22
Tabel 4. 1 Tabel Kombinasi Baud Rate ATMEGA1280 dan LPC1768.....50

BAB I.
PENDAHULUAN

1.1 Latar Belakang

Telekomunikasi pada abad ke-21 di dunia sudah sangat penting untuk komunikasi antarorang ataupun antargadget. Teknologi telekomunikasi pun sudah sangat beragam, baik teknologi telekomunikasi yang menggunakan kabel yang menawarkan peredaman noise yang minimal atau seluler yang menawarkan fleksibilitas dan kemudahan untuk berhubungan dengan jarak jauh tanpa memikirkan masalah kabel.

Tidak terkecuali yang terjadi di Indonesia. Perkembangan telekomunikasi di Indonesia mengalami kemajuan yang sangat pesat dalam pemakaian teknologi telekomunikasi seluler. Sayangnya, perkembangan pemakaian teknologi ini tidak diikuti dengan semakin banyaknya pulau-pulau kecil yang tersentuh dengan komunikasi. Menurut survey dan verifikasi yang dilakukan oleh Kementrian Kelautan dan Perikanan (KKP), jumlau pulau yang ada di wilayah Indonesia yang tersebar dari Sabang sampai Merauke berjumlah kurang lebih 13.000 pulau. Sebagian besar dari pulau-pulau tersebut jauh dari pulau-pulau yang besar di Indonesia. Hal tersebut yang menjadikan pulau tersebut menjadi terisolir dan dibutuhkan teknologi yang tepat untukdapat terhubung dengan luar pulau.

Salah satu teknologi yang tepat adalah komunikai satelit. Satelit mampu untuk melakukan komunikasi yang sangat jauh selama daerah tersebut masih dijangkau oleh satelit. Selain fungsi satelit sebagai komunikasi, satelit juga berfungsi untuk sistem navigasi di bumi pengintaian, prakiraan cuaca, dan lain-lain. Fungsi pengintaian dan prakiraan cuaca sangat berguna untuk keamanan transportasi dan keamanan negara dari serangan negara lain. Adapun, satelit merupakan alat elektronik yang mengorbit di angkasa yang dapat dikendalikan oleh stasiun pengendali yang ada di bumi.

Satelit terbagi menjadi beberapa jenis. Seperti contohnya, satelit palapa dan satelit Ilnusat. Satelit palapa merupakan satelit yang berukuran satelit besar, dan Ilnusat merupakan satelit yang berukuran nano satelit. Keduannya mempunyai banyak perbedaan, terutama dari segi ukuran dan kemampuannya. Adapun,

pengembangan nano satelit lebih menguntungkan daripada satelit konvenional dikarenakan waktu pengembangan nano satelit yang relatif singkat serta biaya pengembangan yang lebih murah. Keuntungan lainnya adalah tetap bisa mempelajari sistem konvensional, walau dengan sistem yang lebih kecil.

Tantangan dalam pembuatan nanosatelit juga tidak berbeda jauh dengan satelit konvensional. Dengan kondisi lingkungan yang ekstrim seperti contohnya panas dan radiasi magnetik matahari yang sangat besar membuat perancangan nanosattelite harus bisa beradaptasi dengan kondisi ruang angkasa. Adaptasi nanosatelit bisa berupa penambahan komponen yang berguna untuk menghilangkan gangguan pada nano satelit, atau modifikasi dari segi software yang dapat digunakan untuk meminimalisir gangguan radiasi magnetik yang dapat mempengaruhi data yang dikirimkan dari stasiun pengendali yang ada di bumi atau pengiriman data dari nanosatelit.

Sebuah nanosatelit memiliki beberapa sistem yang saling terintegrasi satu sama lain. Sistem-sistem tersebut terdiri dari EPS (*Electrical Power Supply*), ADCS (*Altitude Determination Control System*), OBDH (*On-Board Data Handling*), serta Payload. OBDH memiliki fungsi untuk mengatur kerja, menyimpan data, dan memproses data sub-sistem yang ada pada nanosatelit. Selain itu, OBDH juga berfungsi untuk mendapatkan data *housekeeping*.

Pada OBDH (On-Board Data Handling), terdapat sisi *hardware* dan *software* yang perlu dirancang agar OBDH dapat berfungsi dengan baik. Untuk sisi hardware, *hardware* yang dirancang perlu diperhatikan apakah OBDH yang dirancang apakah dapat berfungsi secara optimal tanpa kekurangan *resources* yang dibutuhkan. Dari sisi *software*, *software* harus dirancang agar OBDH dapat terkoneksi, saling berkomunikasi, hingga menunjang kehidupan sub-sistem nanosatelit tersebut. Persyaratan *software* tersebut juga harus handal dan berfungsi secara optimal dalam menangani penugasan yang dibebankan kepada OBDH.

Beberapa riset tentang software OBDH pada nanosatelit atau satelit telah dilakukan sebelumnya. Beberapa perancangannya tersebut adalah:

- Berdasarkan pada jurnal [4] oleh J.Perez Diestro, *software* OBDH MiniSat terbagi atas 3, yaitu Low Level Function, Mission Control Function, dan Attitude Control System. Fungsi Low-Level terdiri dari BootCode, ISR (*Interrupt Service Routines*), dan I/O Drivers. BootCode berfungsi untuk menginisialisasi dan memeriksa *hardware* (prosesor, coprosesor, RAM(*Random Access Memory*), EDAC, EEPROM, dan Memori PROM, (timers, watchdog…), untuk melaporkan hasil pada FCM dan memberikan kendali pada FCM. *Mission Control Function* adalah software utama yang digunakan untuk memonitor dan mengendalikan subsistem, management data housekeeping, dan management data payload. *Attitude control system* adalah software yang digunakan untuk mengendalikan ketinggian satelit.
- Berdasarkan pada laporan akhir [2] oleh ESA (European Space Agency), hardware OBDH ini dibuat dari mikroprosesor Intel 8052. Adapun, hardware yang diujicobakan pada laporan tersebut adalah 3 timer, UART, *Interupt Controller*, dan port I/O. Dari laporan tersebut, telah dilakukan program berupa penerimaan paket CCSDS pada RS-232, memverifikasi dan memechakan paket ke perintah Telemetry and Command, melakukan perintah Telemetry and Command (counters, resetting, start and stop dan pelaporan hasil)
- Berdasarkan pada Jurnal dengan Judul Analisis dan Perancangan Toleransi Kesalahan pada On Board Data Handling Satelit Lapan A2 oleh Abdul Kariri dan Gunawan. S.Prabowo, apabila protokol sampai TTC, maka TTC akan melanjutkan perintah ke OBDH dengan membuang sebuah header dan menyisakan 4 byte protokl untuk dilanjutkan ke OBDH. Error detection yang digunakan dalam hal ini adalah memantulkan (echo) atau mengembalikan tiap byte dari 4 byte. Jika hasil echo tidak sama maka perintah dianggap akan mengalami gangguan. Interfacing dengan setiap subsistem OBDH harus dalam protokol 4 byte. Semua perintah akan sampai ke subsistem OBDH dalam protokol 4 byte/

1.2 Tujuan

Tujuan skripsi ini, yaitu merancang *hardware* dan *software* untuk OBDH (On-Board Data Handling) dengan menggunakan prosesor LPC1768 Cortex M-3. Software ini terdiri dari *Boot Code*, *System Code*, interkoneksi OBDH dengan TTC, sedangkan hardware nya mempunyai prosesor Arm Cortex M3 yang memiliki performa yang tinggi dan hemat energi sehingga cocok untuk aplikasi luar angkasa.

1.3 Batasan Masalah

Skripsi ini membatasi pada perancangan *hardware* dan *software* OBDH yang meliputi perancangan dari hardware dan software pada OBDH (*On Board Data Handling*) UI-SAT Nano Satelite dengan melakukan interkoneksi dengan sub-sistem TTC pada nanosatelit IinuSat.

1.4 Metodologi Penelitian

Metode penelitian yang digunakan adalah studi pustaka dari beberapa jurnal, tesis, dan buku yang berkaitan tentang satelit atau nanosatelit. Selain studi pustaka, metode penelitian yang dipakai adalah metode perancangan On Board Data Handling baik hardware dan software nya agar dapat terkoneksi dengan Terminal Node Controller Telemetry, Tracking, and Command IiNUSAT.

1.5 Sistematika Penulisan

Bab-bab yang terdapat pada skrispsi ini adalah :

BAB 1 : Pendahuluan.

Bab 1 berisi latar belakang, perumusan masalah, tujuan penelitian, batasan masalah, dan sistematika penulisan. Pada bab ini, juga dijelaskan beberapa penelitian yang telah dilakukan pada *On-Board Data Handling* dan *Terminal Node Controller* TTC yang telah dilakukan.

BAB 2: On Board Data Handling UI-SAT dan Terminal Node Controller TTC Iinusat

Bab 2 berisi tentang Satelit dan Jenis Satelit, Definisi OBDH (*On Board Data Handling*) secara umum, TNC (*Terminal Node Controller*), dan Interkoneksi OBDH dengan subsistem lainnya.

BAB 3: Perancangan Hardware dan Algoritma OBDH Interkoneksi dengan TTC untuk OBDH UI-SAT

Bab 3 berisi tentang perancangan hardware dan algoritma pemograman *software* interkoneksi OBDH dengan TTC. Pada bab ini, dijelaskan tentang perancangan hardware untuk OBDH beserta karakteristik dari OBDH, sedangkan untuk software, terdiri dari *boot code, system code,* dan algoritma interkoneksi OBDH UI-SAT dengan TNC Iinusat.

BAB 4: Analisis Kinerja Interkoneksi Obdh Ui-Sat Dengan TTc Iinus**at**

Bab 4 berisi tentang Analisis Hardware OBDH UI-SAT, Frame Pengiriman, Uji-Coba Pengiriman Data OBDH ke TNC Iinusat, Analisa Baud Rate, dan Analisa Pemograman,

BAB 5: Kesimpulan

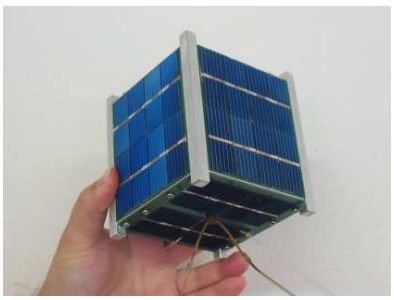
Bab 5 berisi tentang kesimpulan dari perancangan hardware dan software, interkoneksi dengan TTC IInusat, dan kesimpulan kinerja yang dibebankan kepada OBDH.

BAB II.
ON-BOARD DATA HANDLING UI-SAT DAN TERMINAL NODE CONTROLLER PADA TELEMETRY, TRACKING, AND COMMAND IiNUSAT

2.1 NanoSatelit

Umumnya, kegunaan dari sebuah satelit biasanya dilihat dari payload yang dipasangkan pada satelit tersebut. Salah satu dari sekian banyak fungsi dari satelit adalah satelit komunikasi. Satelit komunikasi adalah satelit buatan yang mengorbit di bumi dan biasanya berfungsi untuk repeater untuk kepentingan telekomunikasi. NanoSatelit. Satelit komunikasi ini menerima sinyal dari stasiun bumi, menguatkanm, memproses, dan mengirimkan sinyal kembali ke stasiun bumi. Adapun, satelit komunikasi merupakan salah satu infrastruktur pada telekomunikasi dan bisa jadi alternatif komunikasi apabila komunikasi pada jaringan terestrial pada bumi berada dalam kondisi rusak.

Satelit terbagi atas beberapa jenis, salah satunya adalah nanosatelit. Nanosateit umumnya memiliki berat antara 1-10kg, Gambar dari sebuah nanosatelit dapat dilihat pada gambar 2.1. Dengan ukuran yang kecil, sebuah nanosatelit memiliki beberapa keterbatasan, terutama pada daya listrik dan ukuran. Dengan daya yang terbatas menyebabkan terbatasnya komponen dan fungsi sebuah dari nanosatelit, sedangkan keuntungannya adalah dapat dikembangkan dalam waktu yang lebih cepat dan biaya yang lebih murah dibandingkan dengan satelit konvensional.



Gambar 2. 1 CubeSat

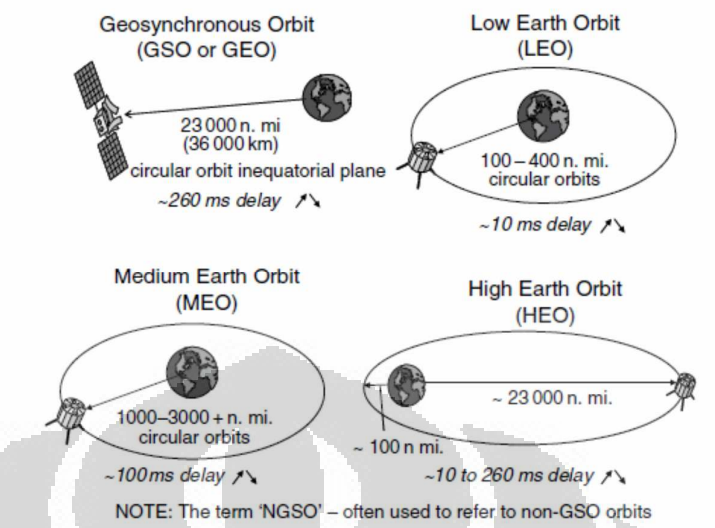
2.2 Orbit Satelit

Orbit mempunyai peran yang penting dalam menentukan jangkauan dan sistem operasional dari sebuah satelit. Terdapat empat macam orbit yang biasanya digunakan pada satelit. Orbit-orbit rersebut adalah

- *Geosynchronous Orbit*
- *Low Earth Orbit*
- *Medium Earth Orbit*
- *High Earth Orbit*

Perbedaan antara empat macam orbit tersebut adalah bentuk orbit, jarak antara satelit dengan bumi, dan delay dari pengiriman sinyal. Semakin jauh jarak dari bumi maka orbit akan meningkat dan waktu akses stasiun bumi akan ebih lama. Begitu pula sebaliknya, semakin dekat jarak dari bumi, maka waktu *pathloss* dan waktu *delay* akan semakin kecil. Jarak LEO dari permukaan bumi yang lebih dekat daripada GEO akan menghasilkan *delay* yang semakin kecil dan *pathloss* yang kecil sehingga daya yang dibutuhkan ldan sistem antenna yang lebih kecil. Adapun, kekurangan dari orbit LEO adalah satelitnya yang posisinya tidak menetap pada angkasa namun akan terus bergerak sehingga stasiun pada bumi harus mengikuti pergerakan satelit dan hanya dapat diakses oleh stasiun bimi selama 8-10 menit dari lokasi tetap di bumi.

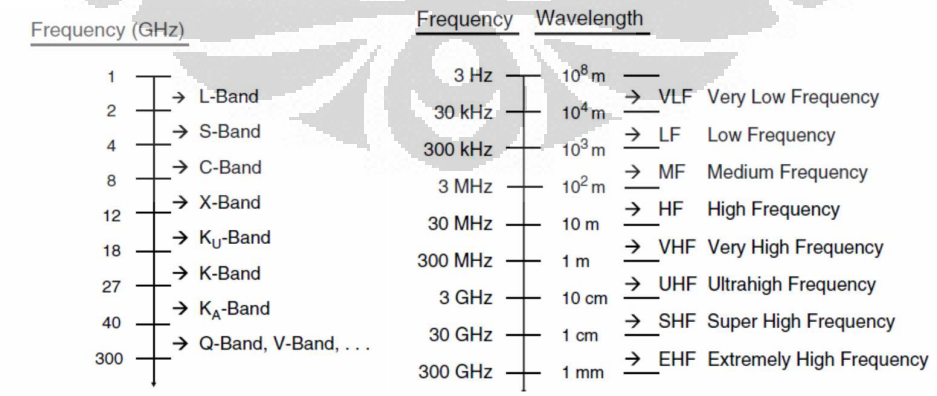
Jenis orbit LEO, yaitu sun sychronous orbit (SSO) dimana satelit berputar dengan inklinasi dan ketinggian tertentu sehingga perputaran satelit terjadi dari kutub utara menuju kutub selatan bersamaan dengan bumi yang berputar di bawahnya. Pada SSO, orbit melewati bidang equator dengan waktu lokal yang bersamaan setiap harinya. Hal tersebut didapatkan karena orbit berputar ke arah timur sebesar 1 derajat setiap perputarannya untuk menyeimbangkan kecepatan revolusi bumi terhadap matahari.



Gambar 2. 2 Orbit Satelit

2.3 Frekuensi Satelit

Frekuensi kerja satelit merupakan suatu faktor yang merupakan suatu faktor yang menentukan dalam rancangan dan performa komunikasi satelit. Panjang gelombang sinyal yang bergerak dalam udara juga merupakan parameter yang menentukan efek dari interaksi sinyal dan atmosfer serta degradasi sinyal akibat adanya noise. Penentuan frekuensi sangat penting bagi perancangan payload komunikasi dari satellit karena berhubungan dnegan ukuran perangkat, *link budget*, daya yang dibutuhkan, dll. Terdapat duia macam penamaan dalam alokasi frekuensi yang ditunjukkan pada gambar 2.3



Gambar 2. 3 Alokasi Frekeunsi

2.4 Sistem Satelit

Sistem sederhana satelit terdiri dari satelit komunikasi yang saling bertukar informasi dengan stasiun bumi. Sistem satelit terdiri dari 3 segmen, yaitu space segment, control segment, dan ground segment yang ditunjukkan gambar 2.4.

- *Space Segment*

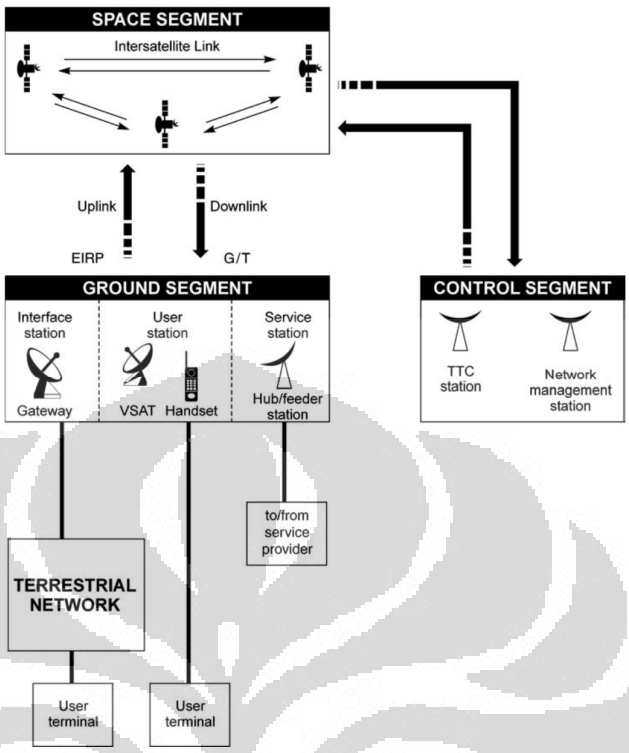
  Terdiri dari satu atau lebih satelit komunikasi yang mengorbit dan diatur sebagaia salah satu konstelasi (kesatuan).

- *Control Segment*

  Terdiri dari semua periferal di bumi yang berfungsi untuk mengatur dan memonitor satelit yang biasa disebut dengan stasiun tracking, tracking, telemetry, command dan monitoring (TTC&M) dan pengelola traffic dan sumber daya yang berhubungan yang terdapat di satelit. Fungsi dari TTC&M ini adalah mengontrol satelit agar satelit ini tetrap bekerja pada orbitnya, Hubungan antara TTC&M dan stasiun bumi biasanya terpisah dari hubungan komunikasi pengguna. TTC&M merupakan stasiun bumi yang mempunyai kemampuan untuk pengendalian satelit.

- *Ground Segment*

  Ground Segment merupakan terminal yang ada di permukaan bumi yang menggunakan kemampuan komunikasi dari satelit. Ground segment terdiri dari berbagai ukuran tergantung dari layanan dari satelit tersebut. Stasiun bumi atau ground segment terdiri dari tiga kelas, yaitu user station, contohnya *mobile station* dan *very small apparture antenna*.



Gambar 2. 4 Sistem Satelit

2.5 On Board Data Handling

OBDH merupakan kepanjangan dari *On Board Data Handling*. OBDH juga merupakan sistem yang lebih kecil dan dijadikan menjadi satu kesatuan dari *Command and Data Handling* pada satelit konvensional. OBDH adalah sebuah mikrokontroller yang mempunyai input, output, dan sensor yang berguna untuk menunjang fungsi OBDH. OBDH juga bisa dikatakan sebagai sebuah subsistem dari satelit yang berfungsi sebagai amtarmuka/penghubung antara subsistem nano staelit lainnya, seperti komunikasi, *payload*, EPS, dan ADCS. Selain dari pengubung dari sub sistem-sistem, OBDH juga berperan sebagai pendesain dari simpan dan transmit sebuah data. Pada OBDH juga terdapat RAM *(Random Access Memory)* dan ROM *(Read Only Memory)* yang digunakan untuk menyimpan data dan memproses memori yang dipakai oleh mikroprosesor.

Fungsi dari OBDH memiliki dua fungsi utama. Fungsi pertama adalah menerima, memvalidasi, mendekode, dan menyalurkan perintah ke sub-sistem nano satelit lainnya. Fungsi kedua adalah mengumpulkan, memproses dan

mengatur dari kerja dan status dari OBDH. Selain itu, OBDH pada umumnya memiliki tambahan fungsi, seperti ketepatan waktu, *monitoring* (*watchdog*), dan antarmuka keamanan.

Ukuran dari subsistem OBDH biasanya berbanding lurus terhadap komplektisitas serta fungsi nanosatelit yang ingin dibuat. Semakin banyak sistem dan fungsi nano satelit yang ingin dimiliki dan fungsi nano satelit,dibutuhkan semakin banyak komponen yang dibutuhkan pada OBDH. Selain itu, untuk membuat sub sistem OBDH semakin reliable, ukuran OBDH bisa diperbesar dan ditambahkan beberapa periferal yang dapat mendukung stabilitas OBDH.

Beberapa standart perintah dari sistem *Command* dan *Data Handling* sudah ditetapkan. Seperti yang sudah ditetapkan pada *Consultative Committee for Space Data Systems* (CCSDS) pada tahun 1987. Pada umumnya, perintah terdiri kode sinkronisasi, bit alamat nano satelit, bit perintah, dan bit *error check*. Perintah yang diterima divalidasi tergantung dari eksekusinya. Validasi terdiri dari bit penerima sinkronisasi, bit panjang message, dan bit error pada sebuah kode polynomial pemeriksa error. Ketika *decoder* memvalidasi sebuah perintah, maka perintah tersebut akan menambah sebuah counter untuk mengingat banyak perintah yang telah dijalankan. Kemudian, bit pesan tersebut akan disampaikan ke decoder untuk dijalankan. Dekoder perintah akan menolak perintah yang tidak sesuai dengan kriteria validasi dan akan meningkatkan *counter* perintah tolak. Sistem penanganan data tersebut akan membaca *counter* terima dan tolak dan memasukkan nya pada data *downlink* untuk dapat menjalankan operasi umpan balik.

Decoder perintah menentukan tipe sebuah perintah output dan kanal antar muka yang spesifik. Sistem OBDH biasanya menyediakan dua tipe dari output, diskrit dan serial. Perintah diskrit mempunyai ciri-ciri yaitu amplitudo yang tetap setiap perintahnya dan durasi pulsa yang tetap dan terdisiri dari dua jenis :

- *High-Level Discrete Command* : merupakan serial yang terdiri dari tegangan +28V, dari 10 sampai dengan 100ms.

- *Low-Level Discrete Command :* merupakan 5V pulsa yang biasanya di antar muka dengan logic digital.

Sedangkan, perintah output serial adalah *interface* data yang menggunakan tiga sinyal atau pin yang terdiri dari *clock, data serial*, dan *data enabled.* Perintah serial lainnya terdiri dari *data receiver, data transmitter*, dan *ground.*

2.5.1 Mikroprosesor ARM Cortex M3

Prosesor ARM Cortex M3 mempunyai arsitektur ARMv7-M. Pada arsitektur ini mengintegrasikan inti prosesor utama dengan keunggulan periferal sistem yang memungkinkan untuk kendali *interupt*, perlindungan memori, dan debug sistem dan trace. Periferal sistem bisa diatur sedemikian rupa sehingga sistem ini bisa disesuaikan dengan persyaratan aplikasi sebuah sistem. Prosesor Cortex-M3 merupakan prosesor 32-bit dengan panjang data 32 bit, register 32 bit dan antar muka memori.

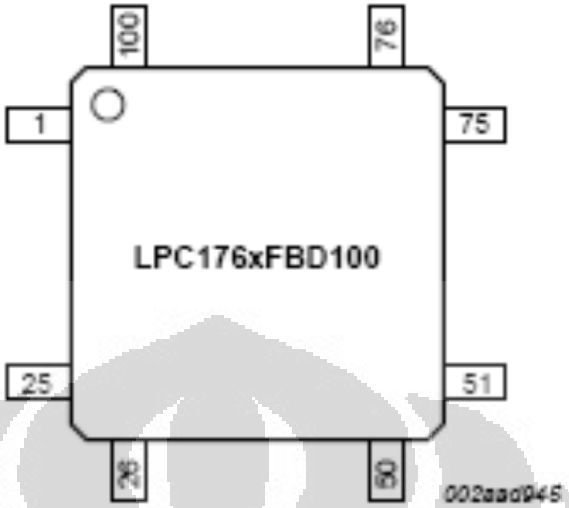
Salah satu dari sekian banyak pabrikan yang memproduksi Prosesor ARM Cortex M3 adalah NXP. Varian dari NXP untuk ARM Cortex M3 adalah LPC1768/67/66/65/64/63. Perbedaan dari varian tersebut adalah perbedaan besar memori flash, dan beberapa fitur yang tidak terdapat pada beberapa varian tersebut, seperti contohnya LPC1767 yang tidak memiliki koneksi USB. Adapun, prosesor ini mendukung frekuensi CPU(Central Processing Unit) sampai dengan 100 MHz. Dengan ukuran prosesor yang ukurannya relatif kecil, prosesor ini tidak membutuhkan daya listrik yang besar. Hal inilah yang membuat mikroprosesor ARM Cortex M3 sangat cocok untuk aplikasi luar angkasa terutama nanosatelit.

Fitur Utama yang ada pada Prosesor LPC1768 ini adalah :

1. Prosesor yang mempunyai struktur data dan register sebesar 16/32 bit.
2. Mempunyai besar Flash Memory sebesar 512 k, dengan maksimum frekuensi CPU sebesar 100 MHz.
3. *Programming in-System/In-Application Programming (ISP/IAP)* dengan software bootloader.

4. Memungkinkan untuk men`debug dari proses yang sedang berlangsung.
5. Mempunyai 8 kanal GPDMA (General Purpose Direct Memory Access) yang bisa digunakan pada port SSP, bus I^2S, UART, periferal Analog to Digital, periferal Digital To Analog, dan Transfer memory to memory.
6. Mempunyai 4 UART dengan Baud Rate yang terpisah, dan dukungan *Direct Memory Acess*.
7. Mempunyai 4 mode daya : Sleep, Deep-sleep, Power-Down, dan Deep Power-Down.
8. Mempunyai PMU (Power Management Unit) yang sudah terintegrasi pada prosesor tersebut sehingga dapat secara otomatis mengatur regulator internal untuk mengurangi konsumsi daya pada kondisi *Sleep*, *Deep-Sleep*, *Power-Down*, dan *Deep Power-Down*.
9. WIC (Wake-up Interrupt Controller) dapat mengatur CPU sehingga CPU dapat kembali hidup dari kondisi deep sleep, Power-down, dan mode Deep power-down.
10. Power-On Reset.
11. Penggunaan Oscilator Kristal dengan jarak frekuensi dari 1 MHz sampai dengan 25 MHz.
12. PLL dapat mengatur frekuensi operasi CPU sampai dengan maksimum Prosesor yaitu sampai dengan 100 MHz tanpa perlu Kristal frekeunsi tinggi. PLL dapat diatur baik dari oscillator utama, oscillator frekuensi internal, atau dari oscillator RTC (*Real Time Clock*)
13. Oscilator Terintegrasi beroperasi dengan Kristal dari 1 MHz sampai dengan 30 MHz dan dengan eksternal oscillator sampai dengan 50 MHz.
14. Memungkinkan fungsi *peripheral* mikroprosesor untuk dimatikan dan penurunan *clock* untuk optimalisasi penggunaan daya.
15. *Chip power supply* dengan rangkaian POR dan BOD :

    CPU beroperasi dari jarak tegangan 3,0 V sampai dengan 3,6 V (3,3V±10%) dengan 5V tolerant I/O



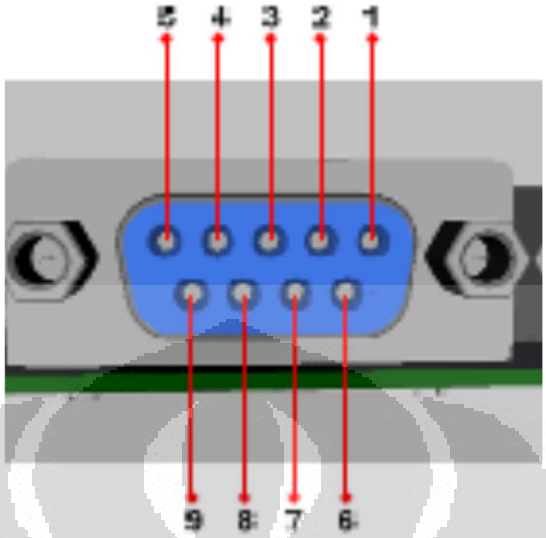
Gambar 2. 5 Struktur Konfigurasi Pin LPC176x untuk Packaging FBD100

2.5.2 Bus UARTs (Universal Asynchronous Receiver Transmitter)

Arsitektur bus UARTs berfungsi sebagai penghubung antara OBDH dan subsistem nano satellite lainnya. Arsitektur ini merupakan serial data bus, sama halnya dengan I2C dan SPI/SSP, namun memilik perbedaan cara kerja. Bus UARTs mempunyai 3 jalur utama yang digunakan untuk koneksi OBDH, yaitu Transmitter(TX), Receiver(RX), dan Ground(GND). Pada bus UARTs ini memungkinkan koneksi secara *duplex*, yaitu memungkinkan koneksi menjadi dua arah, baik masuk atau keluar.

2.5.3 DB9 (F/M)

DB9 adalah konektor yang sesuai dengan koneksi serial dan RS-232. DB9 juga menandakan bahwa ada 4 buah pin konektor yang tersedia pada DB9. Pada DB9, ada dua jenis: *male* dan *female.*



Gambar 2. 6 Konektor DB9-Female

Tabel 2.1 Konfigurasi konektor DB9 Female

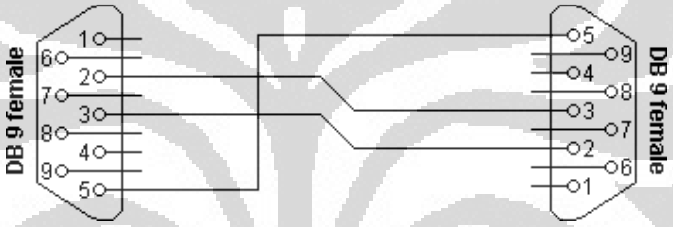
Pin Number	Fungsi
1	Tidak digunakan
2	Pin Data Receiver (RxD)
3	Pin Data Transmitter
4	Tidak digunakan
5	Ground
6	Tidak digunakan
7	Tidak digunakan
8	Tidak digunakan
9	Tidak digunakan

Konfigurasi pada konektor DB9 baik *Female* dan *Male* memiliki perbedaan, terutama pada pin untuk Data Penerima (RxD) dan Data Pengirim (TxD). Untuk interkoneksi konektor DB9 Male dengan *Female*, pin dari transmitter *Female* harus bertemu dengan pin *receiver*.

2.5.4 RS-232

RS-232 merupakan *standart* serial yang digunakan sebagai antarmuka dari peringkat *Data Terminal Equipment* (DTE) dengan perangkat komunikasi data *Data Communication Equipment* (DCE). Awalnya, komunikasi RS-232 pertama kali ditemukan pada tahun 1962, dan kemudian ada pengembangannya pada tahun 1997.

Adapun, konfigurasi pada RS-232 pada kali ini hanya penggunaan berjeniskan untuk *null-modem cables* saja. Jalur yang masuk dalam line cable ini hanya 3 jalur saja, pin *transmitter*, pin *receiver*, dan pin *ground*.



Gambar 2. 7 Koneksi RS-232 *Null-Modem Cables*

Tabel 2.2 Konfigurasi Konektor DB9/F

Connector 1	Connector 2	Function
2	3	Rx ← Tx
3	2	Tx → Rx
5	5	Signal ground

2.6 Telemetry, Tracking, And Command (TTC)

TTC atau merupakan subsistem komunikasi merupakan antarmuka antara nanosatelit dengan stasiun bumi. Data dari sebuah payload dan pengendalian sebuah nanosatelit dikendalikan dari subsistem ini. Adapun, data houskeeping juga diberikan. Perintah ataupun data yang disampaikan dari stasiun bumi atau nanosatelit melewati subsistem ini. Berdasarkan Morgan dan Gordon [1989], subsistem dari TTC memiliki fungsi sebagai berikut :

- Penerimaan Perintah dan Deteksi Perintah

- Modulasi Telemetri dan Transmisi (Penerimaan Data, Pengolahan Data, dan Menyampaikannya Dari Sisem Angkasa)
- Jarak (Penerimaan, Pemrosesan, dan Pentransmittan Sinyal untuk menentukan posisi satelit)
- Operasi Subsistem (Pengolahan Data Subsystem, Pemeliharaan kondisi dan status TTC, penunjukkan antenna, dan pendeteksian kesalahan)

Secara detail, TTC memiliki fungsi sebagai :

- *Carrier Tracking*
  - Komunikasi Dua jalur koheren (*Full Duplex*)
  - Komunikasi Dua jalur non koheren (*Half Duplex*)
  - Komunikasi Satu jalur (*Simplex*)
- Penerimaan Perintah dan Deteksi
  - Mengolah carrier *uplink*
  - Demodulasi *carrier* dan *subcarrier*
  - Pendeteksian *bit timing* dan *data bit*
  - Melewatkan data perintah, clock, dan indikator untuk *Command* dan *Data Handling* ( )
- Modulasi Telemetri dan Transmisi
  - Mendapatkan data telemetri dari *Command* and *Data Handling* atau *On-Board Data Handling* atau Subistem penyimpanan
  - Modulasi *Subcarrier downlink* dan carrier dengan telemetri misi atau telemetri ilmiah
  - Menyampaikan tranmsisi sinyal ke stasiun bumi atau satelit relay
- Ranging
  - Mendapatkan perintah dari subsistem untuk Command and Data Handling C&DH
  - Mengirimkan kondisi dan status telemetri ke subsistem C&DH
  - Melakukan penunjukkan antena (*antenna pointing*)
  - Melakukan urutan operasi misi antara urutan pemograman

- Secara langsung memilih antenna-omni ketika ketinggian nanosatelit tidak diketahui
- Secara langsung dapat mendeteksi kesalahan dan memperbaiki komunikasi dengan menggunakan urutan pemograman yang ada

Adapun, terdapat dua blok penting pada TTC, yaitu *Radio Frequency* dan *Terminal Node Controller.*

2.6.1 *Radio Frequency*

Radio Frequency sangat dibutuhkan untuk melakukan hubungan komunikasi dengan nanosatelit dengan stasiun bumi. Adapun, perancangan pada bagian RF harus dipehatikan agar dapat terkoneksi dengan antenna yang selanjutnya meradiasikan informasi dari sebuah nanosatelit.

2.6.2 *Terminal Node Controller* (TNC)

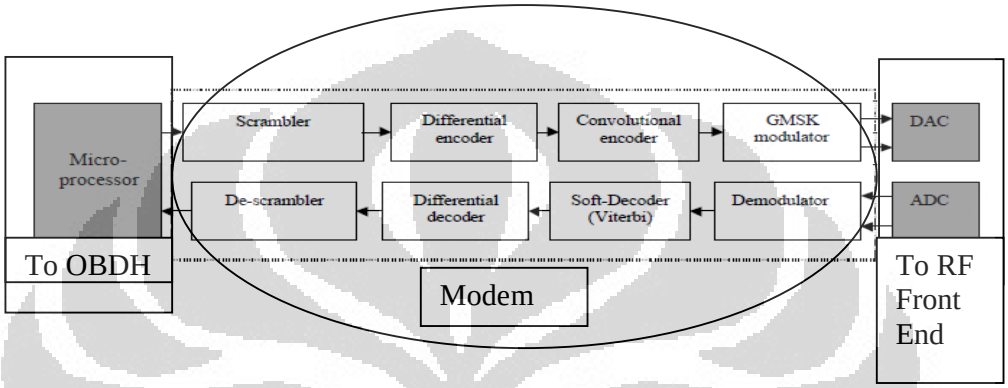
Terminal Node Controller disini berfungsi sebagai penerima perintah dari OBDH, pemaketan data, enkoder dan dekoder dari blok Radio Frequency, serta penerima dan pentransmitan data dari *ground station*. Pada TNC juga terdapat enkoder dan dekoder perintah, apabila terjadi kegagalan atau kesalahan pengiriman antara OBDH dengan TNC

2.7 Deskripsi Sub Sistem Nano Satelit Lainnya yang Saling Berintegrasi dengan OBDH UI-SAT

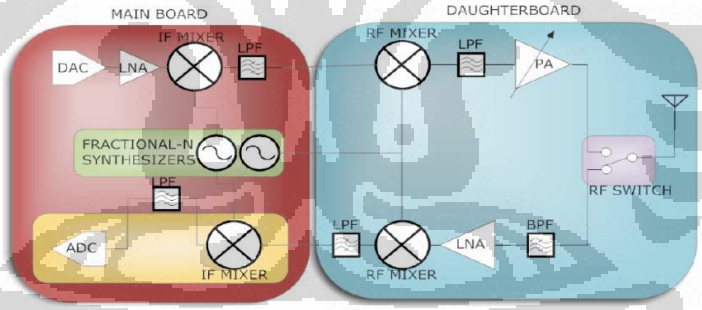
2.7.1 *Payload Communication*

Subsistem *payload* merupakan subsistem nano satelit yang menangani sambungan atau repeater komunikasi antara terminal yang satu ke terminal lain yang ada di bumi. Seperti contoh pada *payload* nanosatelit IiNUSAT, ada 3 bagian penting pada sistem payload. Bagian pertama adalah *RF Front End* yang menangani proses sinyal pada frekuensi radio sebelum masuk ke modem. Bagian kedua adalah modem yang merupakan bagian yang menangani perubahan sinyal informasi dari sinyal dijital ke sinyal analog agar dapat diransmisikan melalui berbagai media transmisi ataupun sebaliknya. Bagian ketiga merupakan

microcontroller yang mengatur proses coding pada sinyal sebelum ataupun sesudah melewati modem. Pada bagian ketiga dari payload inilah yang tersambung pada OBDH IiNUSAT-1, yang selanjutnya OBDH ini dapat memerintah atau memperoleh data dari payload komunikasi IiNUSAT-1.



Gambar 2. 8 Sistem payload IiNUSAT-1

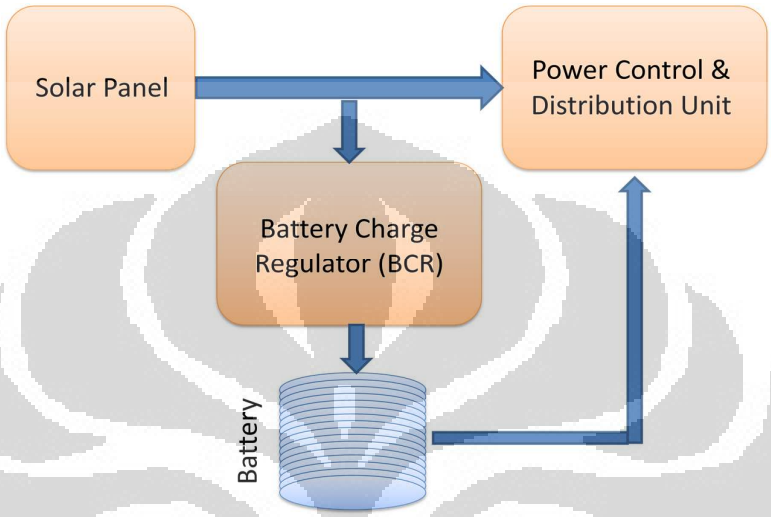


Gambar 2. 9 RF Front End IiNUSAT-1

2.7.2 *Electrical Power System* (EPS)

Subsistem ini digunakan untuk mengatur, mendistribusikan dan menyuplai kelistrikan yang ada pada seluruh subsistem yang berada pada nano satelit. Secara umum, EPS terdiri dari tiga bagian utama yaitu *power source*, *energy storage system*, dan *power control and distribution system*. *Power source* adalah bagian yang dapat menghasilkan energy listrik seperti *solar cell*, *static power source*, *dynamic power source*, dan *fuel cell*. *Energy storage system* berupa baterai yang

berfungsi untuk menyimpan kelebihan energy listrik dari *solar cell*. *Power control distribution system* adalah sistem yang bertugas untuk mengubah tegangan bus menjadi tegangan tertentu sesuai dengan kebutuhan subsistem dan mengontrol *charge discharge* baterai.



Gambar 2. 10 Sistem Electrical Power System secara umum

Seluruh block dari dari subsistem Electrical Power System baik itu block *Charge/Dischare Controlle*r ataupun *block power distribution* dikendalikan sepenuhnya oleh block *Power Control* yang akan terintegrasi dengan komunikasi serial yang terhubung ke OBDH. *Block Power Control* ini berfungsi sebagai pengiriman info baterai, seperti tegangan, arus, status baterai menuju OBDH. *Block Power Control* juga bisa menerima perintah dari OBDH.

2.7.3 *Altitude Determination and Control System (ADCS)*

Selama fase awal peluncuran, satelit akan mengalami rotasi pergerakan secara acak di luar angkasa. Untuk mengantisipasi hal tersebut, maka diperlukan ADCS untuk menstabilkan pergerakan dari satelit tersebut. Subsistem ADCS ini befungsi untuk menentukan posisi arah satelit yang arahnya relatif pada titik tertentu, dalam hal ini bumi. Kendali sikap pada nano satelit adalah proses mengarahkan sikap satelit ke arah yang ingin dikehendaki. Kendali sikap satelit mencakup dua bidang, yaitu :

- Stabilisasi sikap *(attitude stabilization*)

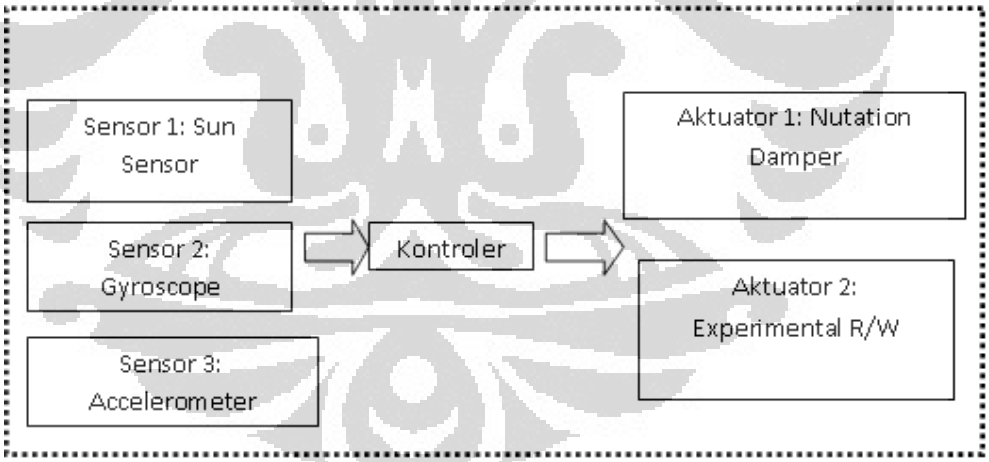
- Kendali maneuver sikap (*attitude maneuver control*), atau proses pengendalian arah satelit dari satu sikap ke sikap lainnya.

Pada sistem kendali dan sikap satelit, secara umum terdiri dari tiga elemen yaitu sensor, actuator, dan sistem kendali.

Metode Pengendalian satelitnya digolongkan menjadi :

- Metode pengendalian Pasif (*Gravity Gradient, Momentum Wheel, Passive Magnetic,* dan *Spin Stabilization*)
- Metode pengendalian Aktif (*Three axis stabilized : Thruster, Reaction Wheel, Control Momment Gyro* (CMG)).

Pada ADCS (*Altitude Determination and Control System*), dibutuhkan adanya sensor, actuator, dan sistem kendali. Sensor dibutuhkan untuk mengetahui kondisi lingkungan, dan untuk menentukan posisi dan ketinggian nano satelit dibutuhkan minimal dua atau tiga sensor dengan keluaran yang berbeda. Sensor tersebut adalah *sun sensor, gyroscope,* dan *accelerometer.*



Gambar 2. 11 Desain Sistem Altitude Determination and Control System

Bagian controller yang akan terhubung dengan OBDH pada nanosatelit UI-SAT.

BAB III.
INTERKONEKSI ON-BOARD DATA HANDLING UI-SAT DENGAN TERMINAL NODE CONTROLLER IiNUSAT

3.1 Tahap Perancangan On Board Data Handling UI-Sat

Perancangan On Board Data Handling harus memerhatikan proses perancangan pin dan persyaratan tegangan dan arus yang dibutuhkan baik untuk mikroprosesor atau periferal lainnya yang dibutuhkan untuk memenuhi fungsi dari OBDH. Selain dengan pemenuhan fungsi, beberapa rangkaian elektronika dibutuhkan untuk On-Board Data Handling untuk rangkaian *clock*. Rangkaian clock ini dibutuhkan untuk rangkaian pemberi clock pada mikroprosesor agar dapat berjalan dan sesuai dengan kecepatan yang diberikan oleh mikroprosesor. Selain dengan pemberian clock, penggunaan rangkaian RS-232 juga dibutuhkan untuk dapat menjembatani antara sub-system dengan On Board Data Handling.

3.1.1 Perancangan LPC1768 pada OBDH UI-SAT

Total pin yang tersedia pada LPC1768 sebanyak 100 pin. Diantara 100 pin tersebut, terdapat pin yang digunakan komunikasi serial, ADC (*Analog to Digital Converter*), atau pin untuk sumber listrik LPC1768. Adapun, penggunaan pin LPC1768 untuk OBDH UI-SAT adalah sebagai berikut :

Tabel 3.1 Perancangan Pin LPC1768

Simbol/Pin	Tipe	Penjelasan
P0.0/RD1/TXD3	O	Pin ini dirancang untuk pin output serial dari LPC 1768 menuju MAX3232EUE untuk UART3
P0.1/TD1/RXD3	I	Pin ini dirancang untuk pin input dari MAX3232EUE menuju LPC1768 untuk UART3
P0.2/TXD0/AD0[7]	O	Pin ini dirancang untuk pin output serial dari LPC1768 menuju MAX3232EUE untuk UART0

P0.3/RXD0/AD0[6]	I	Pin ini dirancang untuk pin input dari MAX3232EUE menuju LPC1768 untuk UART0
P0.10/TXD2/SDA2/MAT3[0]	O	Pin ini dirancang untuk pin output dari LPC1768 ke MAX3232EUE untuk UART2
P0.11/RXD2/SCl2/MAT3[1]	I	Pin ini dirancang untuk pin input dari MAX3232EUE ke LPC1768 untuk UART2
P0.15/TXD1/SCK0/SCK	O	Pin ini dirancang untuk pin output dari LPC1768 ke MAX3232EUE ke LPC1768 untuk UART1
P0.16/RXD1/SSEL0/SSEL	I	Pin ini dirancang untuk pin input dari MAX3232EUE ke LPC1768 untuk UART1
P2[10]/EINT0/NMI	I/O	Pin ini digunakan untuk konfigurasi pin apabila diberi logic LOW maka mikrokontroller ini akan berada pada kondisi RESET pada perintah ISP
VDD(3V3)		Pin ini digunakan sebagai pin tegangan symber untuk Tegangan Suplai *Mikrokontroller*
TDI	I	Test Data pada Antarmuka JTAG (*Joint Test Action Group)*
TD0/SW0	O	Test Data Out untuk Antarmuka JTAG (*Joint Test Action Group*)
TCK/SWDCLK	I	Test Clock untuk Antarmuka JTAG (*Joint Test Action Group*)
TMS/SWDIO	I	Test Mode Select untuk Antarmuka JTAG (*Joint Test Action Group*)
RTCK	O	Signal Pengendali antarmuka JTAG
RESET	I	Signal LOW pada pin ini akan mereset

		prot dan periferal pada kondsi awal dan eksekusi prosesor akan dimulai dari alamat 0
XTAL1	I	Input ke rangkaian *oscillator* dan rangkaian generator *clock internal*
XTAL2	O	Output dari penguat *oscillator*
RTCX1	I	Input pada rangkaian *oscillator* Real Time Clock
RTCX2	O	Output dari rangkaian *oscillator* Real Time Clock
VSS, VSSA	I	Pin ini dirancang secara untuk tegangan referensi 0 Volt
VDD, VREF, VBAT	I	Pin ini secara berturut-turut dirancang untuk tegangan referensi analog, ADC dan DAC

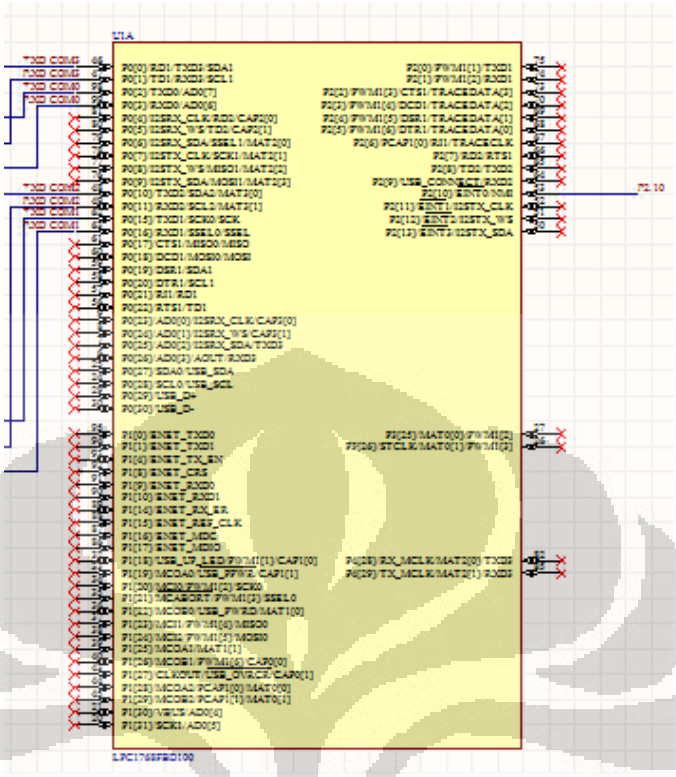
Karakteristik dan Parameter Listrik dan Suhu untuk LPC1768 adalah sebagai berikut :

1. $V_{DD(3v3)}$ merupakan nilai tegangan suplai untuk sumber listrik LPC1768. Batas minimal pada LPC1768 adalah 2.4 Volt dengan batas maksimalnya adalah 3.6 Volt.
2. $V_{DD(REG)(3v3)}$ merupakan besaran tegangan Regulator suplai untuk sumber listrik yang sudah di regulasi. Batas minimal pada LPC1768 2.4 Volt dengan batas maksimalnya adalah 3.6 Volt.
3. V_{DDA} merupakan besaran tegangan suplai untuk suplai tegangan analog. Batas minimal V_{DDA} adalah -0.5 Volt dengan batas maksimalnya adalah 4.6 Volt.
4. $V_{I(VBAT)}$ merupakan besaran tegangan input pada pin V_{BAT} untuk suplai tegangan untuk Real Time Clock. Batas minimal V_{DDA} adalah -0.5 Volt dengan batas maksimalnya adalah 4.6 Volt.
5. Vi merupakan besaran tegangan input pada pin Vi dengan toleransi tegangan 5Volt sampai dengan 0 Volt. Tegangan ini digunakan sebagai

tegangan referensi untuk pin Input dan Output. Batas minimal Vi adalah -0.5 Volt dengan batas maksimalnya adalah 5.5 Volt.

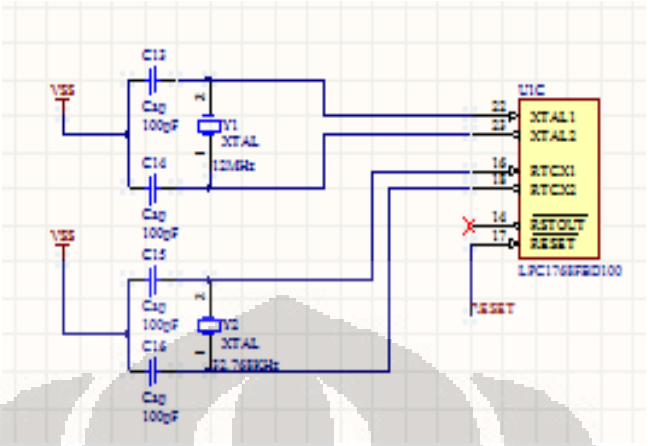
6. I_{DD} merupakan besaran arus yang teralirkan pada pin Regulator Suplai. Nilai yang tersalurkan adalah 42mA pada saat frekuensi Core Clock mencapai 100 MHz.
7. I_L merupakan besaran arus yang mengalir ke LPC1768 pada saat logic LOW yaitu sebesar , dengan Vi adalah 0 Volt, sedangkan I_H merupakan besaran arus pada saat logic HIGH, dengan Vi adalah 3.3 Volt pada pin I/O.
8. I_{IL} merupakan besaran arus yang masuk pada LPC1768 dalam kondisi logic LOW pada pin RESET dengan batas arus 0nA sampai dengan10nA, sedangkan I_{IH} merupakan besaran arus yang masuk pada LPC1768 dalam kondisi logic HIGH pada pin RESET dengan batas arus 0nA sampai dengan 10nA.
9. V_{IH} merupakan besaran tegangan input pada pin reset LPC1768. Batas minimum tegangan V_{IH} adalah 2.33 Volt agar dapat terdeteksi sebagai logic HIGH. V_{IL} merupakan besaran tegangan input pada pin Reset LPC1768. Batas maksimum V_{IL} adalah 0.99 Volt agar dapat terdeteksi sebagai logic LOW. Pemisahan level tegangan ini bertujuan agar dapat memisahkan antara logic LOW dan HIGH.
10. Karakteristik termal yang diperbolehkan pada LPC1768 adalah sebesar 125^{o}C. Hal ini menandakan pada proses fabrikasi, penyolderan pada LPC1768 tidak diperbolehkan terlalu panas karena bisa merusak prosesor tersebut.

Oleh karena itu, untuk memenuhi kriteria tersebut dibutuhkan pembuatan rangkaian pendukung seperti Rangkaian Reset dan ISP. Rancangan pendukung tersebut harus tidak melewati parameter yang ada.

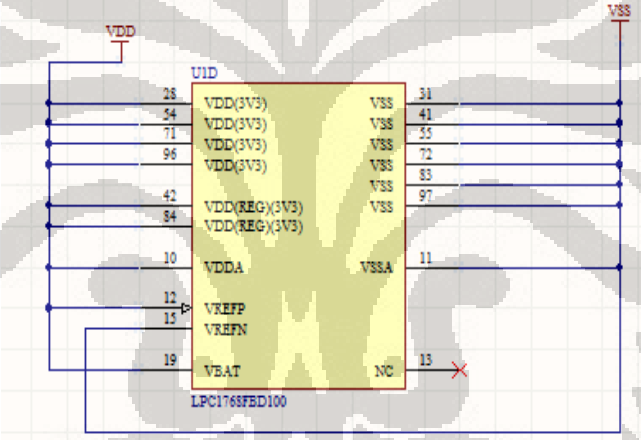


Gambar 3.1 Perancangan LPC1768FBD100 pada *software* Altium Bagian 1

Perancangan Clock dengan frekuensi kristal dengan frekuensi sebesar 12 MHz dan 32.768 KHz bertujuan untuk menghasilkan clock pada LPC1768. Pemberian clock disini sangat diperlukan untuk mengatur seberapa cepat prosesor tersebut berjalan untuk setiap instrukti yang diberikan . Dari range yang diperbolehkan LPC1768 dari 1 MHz sampai dengan 20 MHz, maka pemakaian kapasitor kristal sebesar 12 MHz bertujuan untuk menghasilkan clock yang mudah diatur pada LPC1768.



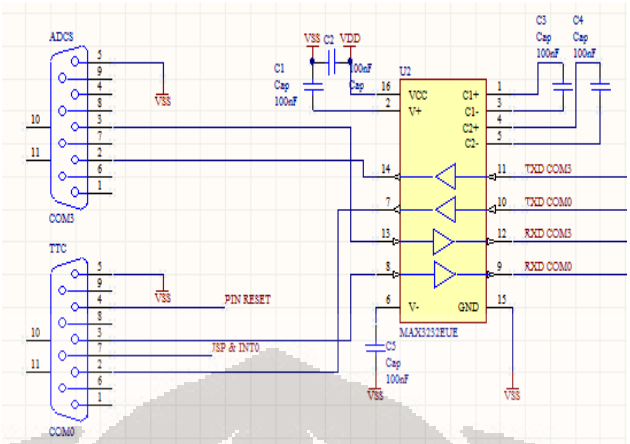
Gambar 3.2 Perancangan LPC1768FBD100 pada *software* Altium Bagian 2



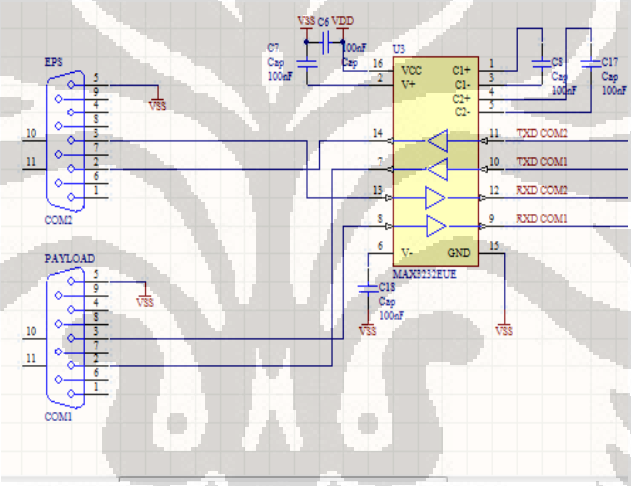
Gambar 3.3 Perancangan LPC1768FBD100 pada *software* Altium Bagian 3

3.1.2 Perancangan MAX3232EUE pada OBDH UI-SAT

Penggunaan MAX3232 digunakan sebagai antarmuka RS-232 antara sub-sistem OBDH dengan subsistem lainnya. Adapun, konektor yang digunakan untuk RS-232 adalah DB9/F. Perancangan MAX.232 pada OBDH UI-SAT adalah sebagai berikut :



Gambar 3.4 Perancangan MAX.3232EUE pada Interface DB9/F dengan LPC1768FBD100 pada ADCS dan TTC



Gambar 3.5 Perancangan MAX3232EUE pada Interface DB9/F dengan LPC1768FBD100 pada EPS dan Payload

Pada rancangan ini, untuk membuat 4 keluaran dan masukan RS-232, maka dapat digunakan 2 transceiver yang menopang koneksi hubungan RS-232 dengan TTL pada mikroprosesor. Kapasitor pada MAX3232EUE tersebut berguna untuk *flying capacitor dan reservoir capacitor*. *Flying capacitor* berfungsi untuk menaikkan atau menurunkan tegangan DC apabila tegangan belum sesuai dengan persyaratan, sedangkan *reservoir capacitor* berguna untuk memperbaiki sinyal keluaran tegangan DC.

Karakteristik Parameter Listrik dan Suhu dari MAX3232 ini adalah

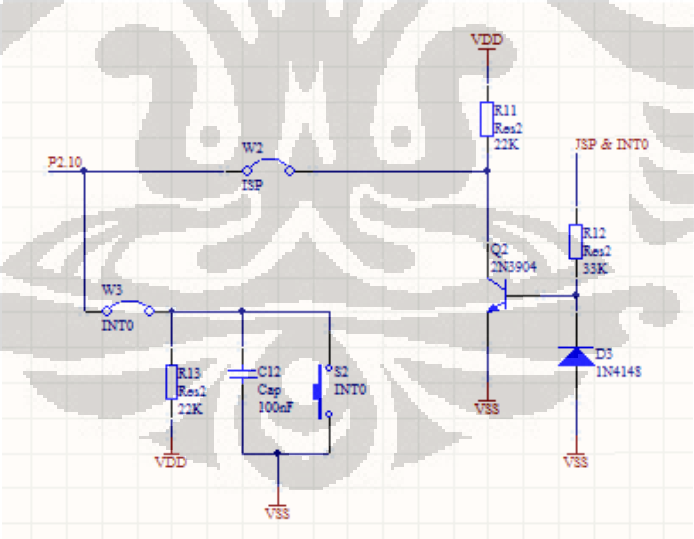
1. Besar VCC yang diperbolehkan untuk LPC1768 adalah 3.3Volt Volt sampai dengan 5 Volt. VCC ini bertujuan sebagai sumber tegangan yang dibutuhkan untuk MAX.3232EUE yang diperlukan untuk tegangan referensi dan sumber listrik
2. Besar V+ merupakan besaran tegangan yang diperlukan untuk MAX3232EUE untuk menghasilkan tegangan dasar yang diperlukan untuk menghasilkan level tegangan positif yang dibutuhkan untuk mencapai level tegangan RS-232. V- merupakan besar tegangan yang diperlukan MAX3232EUE untuk menghasilkan tegangan dasar yang diperlukan untuk menghasilkan level tegangan negatif yang dibutuhkan untuk mencapai level tegangan RS-232. Tegangan maksimal yang diperbolehkan untuk V+ adalah -0.3 Volt sampai dengan 7 Volt, sedangkan untuk V- adalah +0.3 sampai dengan -7 Volt.
3. Tegangan Input maksimal untuk Transmitter MAX3232 adalah -0.3Volt sampai dengan 6Volt, sedangkan tegangan untuk *Receiver Input* adalah ±25Volt. Untuk perancangan MAX3232, *Transmitter Input* dirancang untuk pin receiver(RXD) pada LPC1768 dan *Receiver Input* dirancang ke DB9 *Female.*
4. Tegangan Output maksumal untuk Transmitter MAX3232 adalah ±13.2Volt, sedangkan tegangan untuk Receiver Output adalah -0.3Volt sampai dengan 3.6 Volt. Untuk perancangan MAX3232, *Transmitter Output* dirancang untuk DB9 *Female,* sedangkan *Receiver Output* dirancang untuk pin Transmitter(TXD) pada LPC1768.

Dari karakteristik MAX3232EUE ini, maka perancangan MAX3232 ini dapat dirancang atau diantarmuka dengan LPC1768 karena karakteristik yang sesuai antara LPC1768 dengan MAX3232. Hal ini dapat dilihat dari Tegangan Receiver Output MAX3232 sesuai dengan Level Tegangan Input dan Output pada

LPC1768 dan Tegangan Transmitter Input juga sesuai dengan Level Tegangan Input/Output LPC1768.

3.1.3 Perancangan ISP(*In-System Programming*) dan *Reset Schematic* pada OBDH UI-SAT

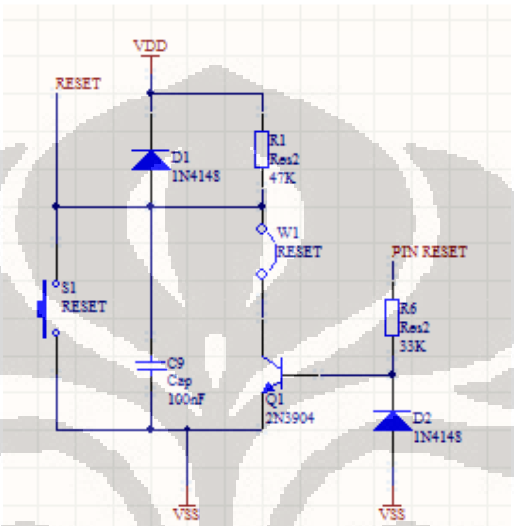
Pada perancangan *ISP* dan *Reset Schematic*, perancangan ini diupayakan agar pin 2.10 tidak berada dalam kondisi LOW terus menerus. Perancangan ini harus didesain sedemikian rupa agar input pin 2.10 berada dalam kondisi LOW apabila memang diinginkan. Selain itu, pemasangan jumper dan *push button* berguna untuk memilih opsi yang memang diinginkan. Selain dengan jumper, penambahan opsi dengan pin ISP&INT0 dapat digunakan pada DB9 *Female* pertama atau COM0. 2N3904 disini merupakan BJT tipe NPN yang merupakan komponen elektronika yang berguna sebagai switching untuk posisi LOW atau HIGH pada pin P2.10. Dioda yang digunakan pada Rangkaian ISP dan INT0 adalah 1N4148 yang merupakan dioda yang berguna sebagai *High Switching Diode*.



Gambar 3.6 Perancangan Rangkaian ISP dan INT0

Begitu pula pada rangkaian *reset*, perancangan ini dirancang agar pin RESET tidak terus berada pada kondisi LOW. Apabila pin Reset terus berada dalam kondisi Low, hal yang akan terjadi adalah LPC1768 akan terjadi reset terus

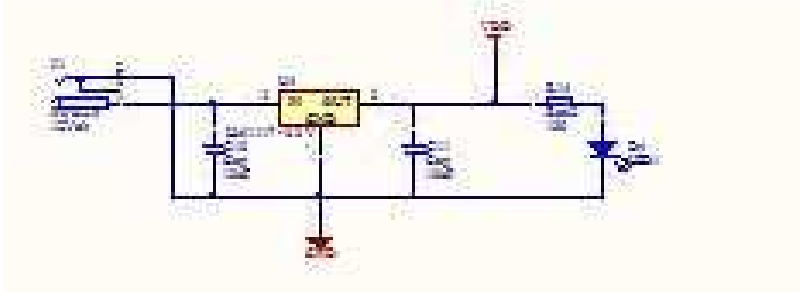
menerus dan tidak dapat meneruskan instruksi yang diberikan. Diode 1N4148 berfungsi sebagai *High Switching Diode* yang berguna blocking tegangan DC dengan pulse yang cepat, dan switiching yang cepat dari *push button* dan BJT 2N3904 berguna sebagai switching yang menentukan pin RESET berada dalam kondisi LOW atau HIGH.



Gambar 3.7 Perancangan Rangkaian Reset

3.1.4 Perancangan Power Supply 3.3V

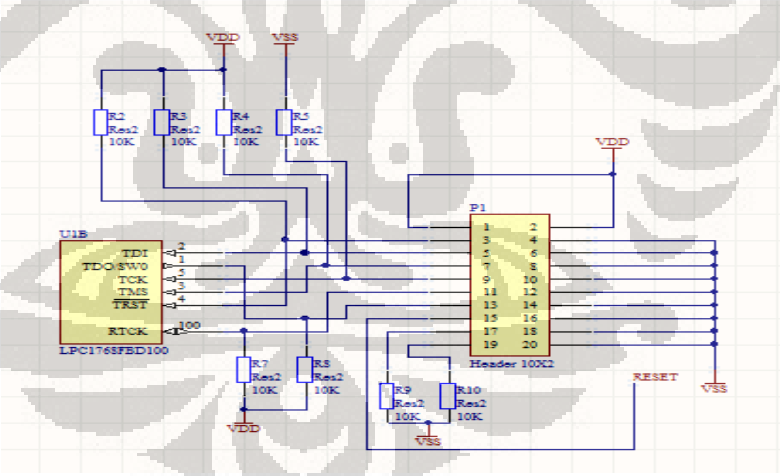
Perancangan bagian power supply ini bertujuan sebagai sumber listrik untuk menyalakan *On Board Data Handling*, beserta semua komponen yang ada didalamnya. Diasumsikan bahwa tegangan yang didapat dari *Electric Power System* sebesar 9 Volt. Oleh karena itu perancangan power supply dibutuhkan voltage regulator yang berfungsi untuk mengatur tegangan dari 9 Volt menjadi 3.3 Volt.Oleh karena itu, digunakan IC LM1117 dengan tegangan input maksimal 15 volt dengan *fixed* tegangan output sebesar 3.3 Volt. Penggunaan kapasitor berguna untuk menstabilkan tegangan DC agar tidak terjadi Tegangan DC berdenyut yang dapat merusakkan komponen LM1117 atau komponen lainnya.



Gambar 3 8 Perancangan Power Supply 3.3 V

3.1.5 Perancangan JTAG (*Joint Test Action Group*)

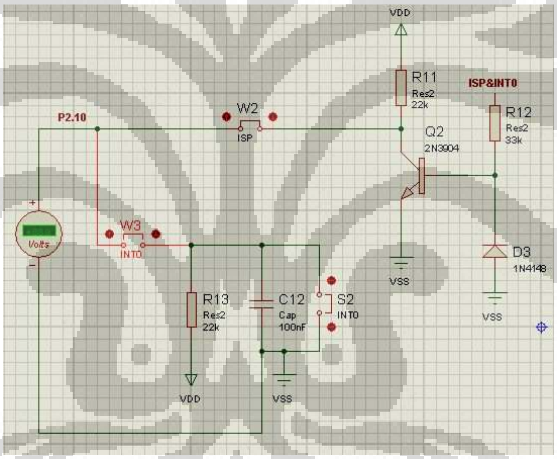
Perancangan JTAG ini bertujuan untuk memprogram mikroprosesor yang ada pada LPC1768. Perancangan JTAG yang baik sangat dibutuhkan untuk menghindari adanya byte-byte pemograman yang hilang. Oleh karena itu, untuk mencegah terjadinya byte-byte yang hilang tersebut diperlukan perancangan hardware yang baik. Resistor *pull-up* dan *pull-down* berguna untuk menghindari terjadinya drop nya tegangan logic 0 dan 1.



Gambar 3 9 Perancangan Rangkaian Joint Test Action Group

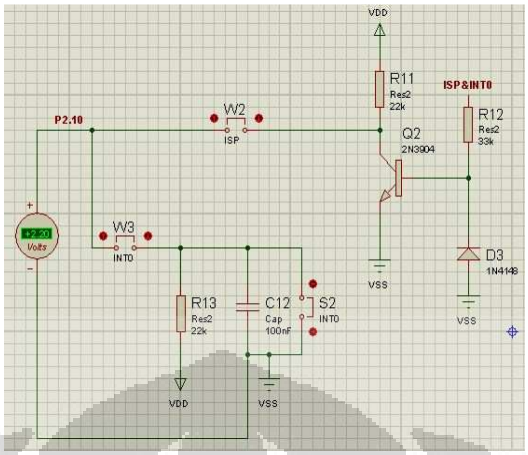
3.2 Simulasi Rangkaian pada Software Simulasi Proteus

Dari hasil Simulasi Rangkaian ISP dan INT0, maka terdapat empat kondisi yang memungkinkan pada rangkaian ISP dan INT0 ini. Dengan asumsi pin ISP&INT0 bernilai logic LOW atau dengan kata lain tidak ada tegangan yang masuk pada sambungan DB9 Female atau COM0. Dari hasil simulasi, maka dapat dilihat pada pin P2.10 akan didapatkan level tegangan sebesar 2.2 Volt pada saat Jumper W2 dan W3 dalam keadaan hubung buka, level tegangan P2.10 sebesar 3.3 Volt pada saat Jumper W2 hubung singkat dan W3 hubung buka, level tegangan P2.10 sebesar 3.3Volt W2 hubung buka dan W3 hubung singkat, dan level tegangan P2.10 sebesar 3.3 Bolt pada saat jumper W2 dan W3 pada keadaan hubung singkat.

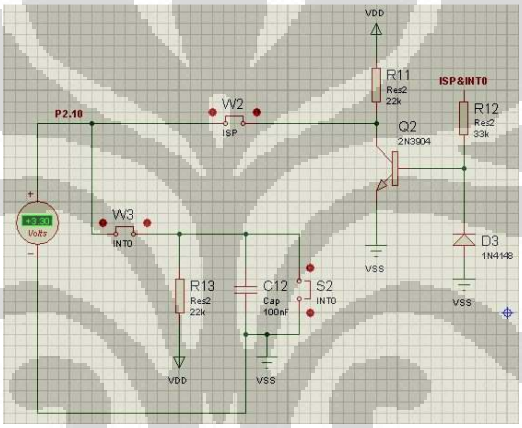


Gambar 3 10 Simulasi Rangkaian ISP dan INT0

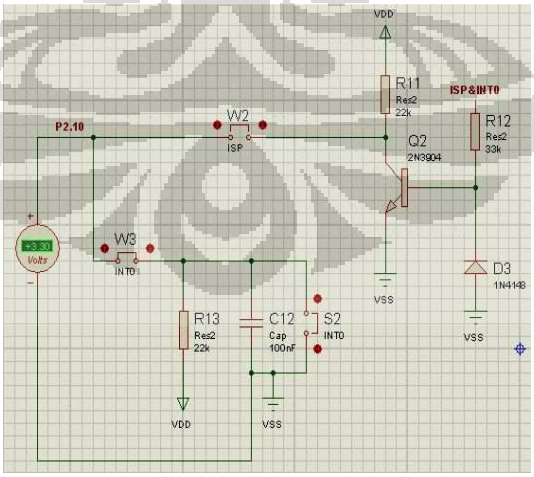
Analisis dari rangkaian ini adalah apabila tidak ada tegangan yang hadir pada kaki ISP&INT0, maka tegangan P2.10 tidak akan LOW. Hal ini sesuai dengan perancangan karena pemograman pada LPC1768 tidak menggunakan ISP sebagai programmer LPC1768



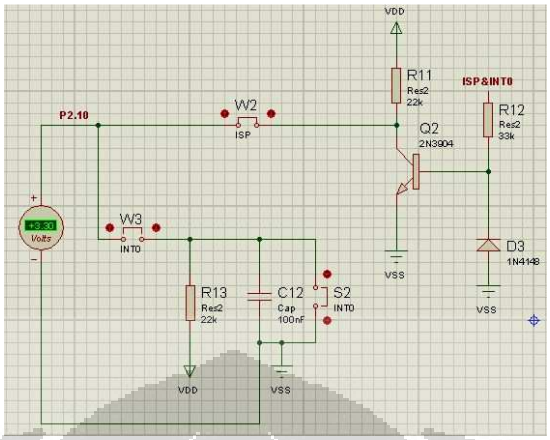
Gambar 3 11 Simulasi Rangkaian ISP dan INT0 pada Saat W1 dan W2 Keadaan Hubung Buka



Gambar 3 12 Simulasi Rangkaian ISP dan INT0 pada Saat Jumper W1 dan W2 pada Keadaan Hubung Singkat

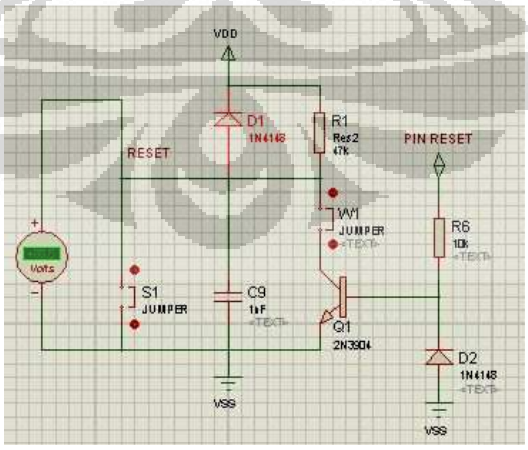


Gambar 3 13 Simulasi Rangkaian ISP dan INT0 pada saat Jumper W1 Hubung Buka dan W2 Hubung Singkat

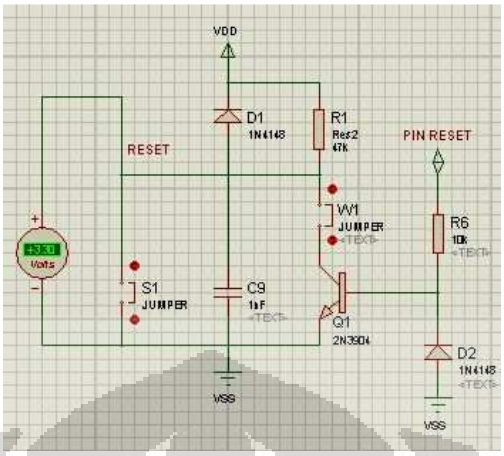


Gambar 3 14 Simulasi Rangkaian ISP dan INT0 pada saat Jumper W1 Hubung Singkat dan W2 Hubung Buka

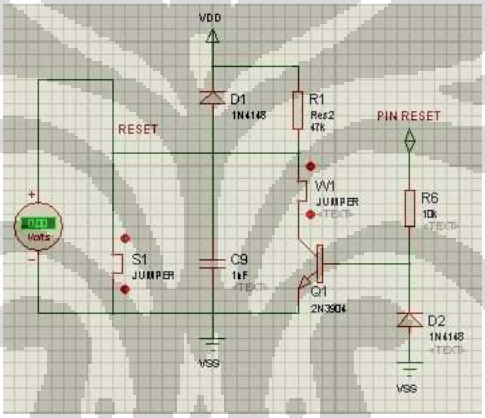
Dari hasil simulasi Rangkaian Reset, terdapat empat kondisi yang dapat terjadi pada Rangkaian Reset ini. Rangkaian Reset ini akan menghasilkan level tegangan RESET sebesar 3.3 Volt apabila jumper S1 dan W1 berada pada kondisi hubung buka, level tegangan RESET sebesar 0 Volt apabila jumper S1 dan W1 pada kondisi hubung singkat, level Tegangan RESET sebesar 3.3 Volt apabila jumper S1 berada pada kondisi hubung singkat dan W1 dalam keadaan hubung buka, dan level Tegangan RESET sebesar 3.3Volt dengan S1 pada kondisi hubung buka dan W1 dalam keadaan hubung singkat.



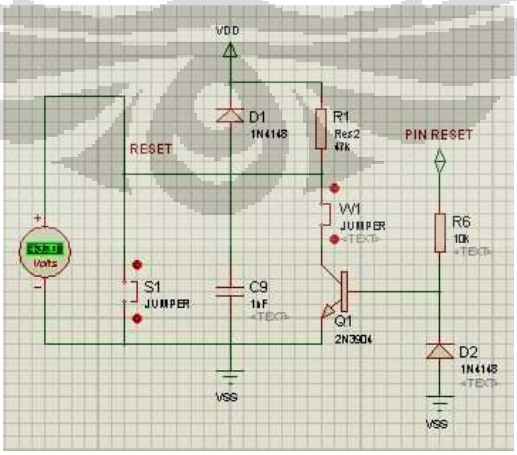
Gambar 3 15 Simulasi Rangkaian Reset



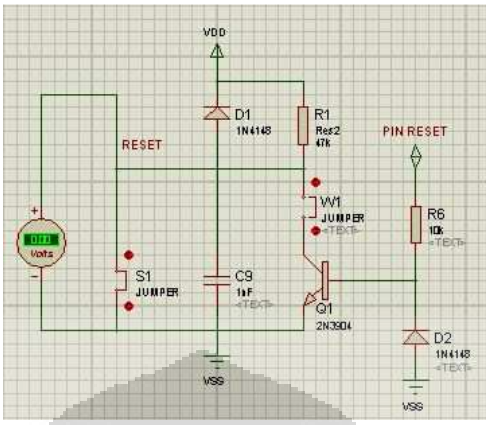
Gambar 3 16 Simulasi Rangkaian Reset pada Jumper S1 dan W1 Keadaan Hubung Buka



Gambar 3 17 Simulasi Rangkaian Reset pada Jumper S1 dan W1 Keadaan Hubung Singkat



Gambar 3 18 Simulasi Rangkaian Reset pada Jumper S1 Hubung Buka dan W1 Keadaan Hubung Singkat



Gambar 3 19 Simulasi Rangkaian Reset pada Jumper S1 Keadaan Hubung Singkat dan Jumper W1 Keadaan Hubung Buka

Analisis dari Rangkaian Reset ini akan menghasilkan sebuah RESET yang sesuai dengan level RESET yang ada pada LPC1768. Logic Level sebesar 0 Volt akan menghasilkan RESET pada LPC1768 dengan mengembalikan posisi memory menjadi awal, dan Logic Level sebesar 3.3 Volt tidak menghasilkan posisi Reset untuk LPC1768.

3.3 Interface OBDH UI-SAT dengan Subsistem TTC Iinusat

Untuk dapat terhubung dengan TTC, kedua subsistem tersebut harus menggunakan RS-232, pasangan DB9 antara Male dan Female, dan dengan Tegangan Referensi yang Sama. Selain itu, pada TTC harus diberi pemograman sehingga dapat merespon fungsi perintah yang ada pada On-Board Data Handling. Selain itu, pada TTC juga diperlukan pemograman yang agar dapat mengirimkan data menuju OBDH. Karena hubungan yang sudah melewati digital, maka tidak diperlukan lagi antar muka antara analog dan digital.

3.3 Interface OBDH UI-SAT dengan Subsistem NanoSatelit Lainnya

Untuk bisa terhubung dengan OBDH, subsistem nanosatelit harus menggunakan RS-232, pasangan DB9 antara Male dan Female, dan dengan tegangan referensi yang sama. Selain itu, untuk dapat terhubung konektor DB9/F, pada subsistem harus diberi pasangan DB9/M. Selain itu, pada sub sistem juga diberi sistem dekoder dan enkoder yang perannya selain berfungsi untuk mengatur kerja dari subsistem juga melakukan hubungan antara mikrokontroller dengan

mikrokontroler. Selain itu, karena perancangan interkoneksi sistem menggunakan serial, maka oleh karena itu pengiriman data ataupun perintah dari OBDH menuju sub-sistem atapun sebaliknya menggunakan tipe pengiriman serial.

3.4 Algoritma Pemograman On-Board Data Handling dengan TTC IInusat.

Untuk dapat terhubung antara OBDH dengan TTC, diperlukan adanya suatu perjanjian antara TTC dan OBDH tentang frame yang akan dikirimkan nantinya. Frame yang dikirimkan dan struktur pengiriman antara OBDH dengan TTC Iinusat harus sesuai. Sehingga, apabila terjadi kesalahan pengiriman atau terjadinya noise pada saluran penghubung antara OBDH dengan TTC, diperlukan adanya algoritma yang bisa mendeteksi kesalahan tersebut dan dapat mengurangi kesalahan kerja pada sistem nanosatelit.

3.4.1 Algoritma Mode Ambil Data dari TTC

Mode perintah OBDH yang dikirimkan ke TTC dimana data yang didapatkan dari TTC untuk dikirimkan ke OBDH (On-Board Data Handling). Pada TTC, terdapat TNC yang digunakan untuk mengatur fungsi pada TTC, baik untuk *receive* dan *transmit* dari *Ground Station* atau OBDH. Adapun struktur pengiriman dari OBDH untuk mengambil data dari TNC adalah $ + 0x2E (.)+ 0x2F (/) + Info (4 byte)+ 0x21 (!)+ CR + LF. Dengan $ adalah 0x24 merupakan hexadecimal dari tanda dollar, ditambah dengan identitas awal pengirim, yaitu id OBDH (0x2E), selanjutnya ditambah dengan identitas pengiriman perintah Ambil Data dari TNC, 0x21 merupakan byte untuk checking kesalahan data, CR(*Carriage Return*) merupakan karakter yang menandakan berakhirnya suatu perintah yang sudah ditentukan dengan 0x0D, dan LF (*Line Feed*) merupakan karakter yang menandakan baris baru yang sudah ditentukan dengan nilai hexadecimal 0x0A. Frame ini berjumlah 10 byte, dengan 1 byte sebagai kode singkronisasi, 1 byte sebagai kode alamat asal, 1 byte sebagai identitas perintah/ command, 3 byte sebagai informasi yang masuk dalam paket tersebut, 1 byte sebagai error checking, 2 byte sebagai penanda akhir dari sebuah frame. Respon yang diberikan oleh TNC kepada OBDH apabila fungsinya berjalan adalah frame paket pengirman dari TTC.

3.4.2 Algoritma Kirim Data ke TTC

Mode kirim OBDH yang dikirimkan ke TNC dimana data yang didapatkan dari OBDH dikirimkan ke TNC untuk dapat disampaikan ke bagian RF. Pada TTC, terdapat TNC yang digunakan untuk mengatur fungsi pada TTC, baik untuk receive dan transmit dari *Ground Station* atau OBDH. Adapun struktur pengiriman dari OBDH untuk mengambil data dari TNC adalah $ + 0x2E (.) + 0x2B (+) + Info Kosong (4 byte) + 0x21 (!) + CR + LF. Dengan $ adalah 0x24 merupakan *hexadecimal* dari tanda dollar, ditambah dengan identitas awal pengirim, yaitu id OBDH (0x2E), selanjutnya ditambah dengan identitas pengiriman perintah Kirim Data ke TNC, 0x21 merupakan byte untuk checking kesalahan data, CR (Carriage Return) merupakan karakter yang menandakan berakhirnya suatu perintah yang sudah ditentukan dengan 0x0D, dan LF (Line Feed) merupakan karakter yang menandakan baris baru yang sudah ditentukan dengan nilai hexadecimal 0x0A. Respon yang diberikan oleh TNC kepada OBDH apabila fungsinya berjalan adalah respon $ + 0x2E (.) + 0x2B (+) + Info Isi (4 Byte + 0x21 (!) + CR + LF)

BAB IV.
ANALISIS KINERJA INTERKONEKSI OBDH UI-SAT DENGAN TTC IiNUSAT

4.1 Fitur-Fitur Mikroprosesor LPC1768 yang Digunakan pada OBDH UI-SAT

Pada mikroprosesor LPC1768 yang telah dirancang, adapun fitur-fitu yang digunakan pada On-Board Data Handling adalah UART0 dan UART1. Adapun, UART0 dihubungkan dengan TNC Iinusat dengan menggunakan frame pengiriman serial, dan UART1 dihubungkan dengan PC1 (*Personal Computer*) sebagai pemberi perintah *dummy* kepada TNC dan penyaji informasi dan data dari OBDH. Pemograman mikroprosesor LPC1768 menggunakan Uvision sebagai IDE *Compiler* dan *Programmer*, dan bahasa pemograman yang digunakan pada pemograman OBDH UI-SAT adalah bahasa C. Clock frequency yang digunakan pada LPC1768 adalah 100 MHz dengan external clock kapasitor kristal sebesar 12 MHz. Adapun, kapasitor kristal ini diletakkan sedekat mungkin agar terhindarkan loop yang tinggi akibat karakteristik kapasitor kristal dan efek kapasitan yang akan berubah apabila jalur PCB yang dibuat terlalu panjang.

4.1.1 Register yang Digunakan Pada LPC1768

Register yang digunakan pada LPC1768 berbeda register yang digunakan, untuk tiap kanal komunikasinya, baik UART0 dan UART1. UART0 dan UART1 ini menggunakan register sebagai berikut

- RBR (*Receiver Buffer Register*) (DLAB=0). Register yang berisi karakter yang akan dibaca (Akses : *Read Only*)
- THR (*Transmit Holding Register*) (DLAB = 0). Register yang berisi karakter yang akan disampaikan (Akses : *Write Only*)
- IIR (*Interrupt ID Register*). Register yang berisi status interrupt yang tertunda (Akses : Read Only)
- DLL (*Divisisor Latch LSB*(*Most Significant Byte*)). Register ini digunakan untuk menghasilkan pembaguan baudrate untuk transfer data (Akses : *Read and Write*)

- DMM (*Divisor Latch MSB* (*Most Significant Byte*)). Register ini digunakan untuk menghasilkan baudrate untuk transfer data.
- IER (*Interrupt Enable Register*). Register ini berisi digunakan untuk mennyalakan interrupt UART
- FCR (*FIFO Control Register*). Register ini bertujuan untuk mengendalikan mode penggunaan FIFO dan Mode
- LCR (*Line Control Register*). Register ini bertujuan untuk Format Frame
- LSR (*Line Status Register*). Register ini berguna untuk tanda status sedang menerima atau menyampai kan data, termasuk kesalahan kawat.

4.1.2 BootCode, Startup Code, Main Code, dan Header Files

Bootcode digunakan pada LPC1768 untuk menyalakan beberapa periferal,, pemberian clock, pemberian PLLClock dan penerapan multiplier dalam pemberian clock pada LPC1768. Pemberian clock pada LPC1768 menentukan seberapa cepat mikroprosesor tersebut berjalan. Code ini secara otomatis terhasilkan pada pemograman pada software *uvision*.

Sedangkan pada startup code LPC1768, code ini digunakan untuk penamaan periferal, menyalakan beberapa intterupt yang ada pada sistem, dan penamaan register sehingga bisa diakses pada main code.

Main code LPC1768, code ini digunakan untuk main program yang menjalani dan mengakses fungi perintah dan penyaji data. Main code ini yang akan berjalan setelah semua BootCode dan Startup Code telah berhasil dijalankan oleh mikroprosesor. Adapun, isi dari main code ini adalah bagaimana dapat mengirimkan sebuah karakter atau string berupa ASCII Code yang digunakan sebagai representasi dari fungsi perintah dari

Header Code LPC1768, code ini digunakan untuk penamaan fungsi dan definisi pemograman yang telah digunakan sebelumnya. Pada header code juga berisi tentang pernyataan tipe dan jenis variabel. Selain itu. Pada header file juga

terdapat IRQ yang digunakan untuk menjalankan fungsi *background* pada LPC1768.

4.2 Fitur-fitur Mikroprosesor ATMEGA1280 yang Digunakan pada TNC TTC Iinusat

Pada mikroprosesor ATMEGA1280, adapun fitur-fitu yang digunakan pada Terminal Node Controller adalah UART0 dan UART1. Adapun, UART0 dihubungkan dengan OBDH, sedangkan UART1 dihubungkan dengan RF Iinusat, dan UART1 dihubungkan dengan PC2 (*Personal Computer*) sebagai pemberi data kepada TNC dan penyaji informasi dan data dari TNC. Pemograman mikroprosesor ATMEGA1280 menggunakan CodeAvr IDE *Compiler* dan *Programmer*, dan bahasa pemograman yang digunakan pada pemograman untuk TNC IiNUSAT adalah bahasa C. Dengan menggunakan external clock kapasitor kristal sebesar 12 MHz, maka bisa didapatkan baud rate dengan jarak antara 2400bps sampai dengan 0.5Mbps.

4.2.1 Register yang Digunakan Pada ATMEGA1280

Pada mikroprosesor ATMEGA1280, register yang digunakan untuk pemrosesan UART0 dan UART1 adalah

- USART Data Register (UDR). Register UDR ini digunakan untuk penyimpanan data yang akan dikirimkan . Register ini juga digunakan untuk register pentrasmitan data. Register ini hanya akan dikirimkan ketika UCSRA bernilai logic 1. Selanjtunya data yang masuk pada register ini, baik berupa karakter atau string, akan dikirimkan secara serial melalui pin TXD.
- USART Control and Status Register (UCSRA). Register ini berfungsi untuk mengatur kerja komunikasi serial. Register ini bergungsi untuk status data dan pendeteksian status register
- USART Control and Status Register (UCSRB) . Register ini berfungsi untuk menatur kerja komunikasi serial dan berkerjanya berkaitan dengan register UCSRA dan UCSRC

- USART Control and Status Register C (UCSRC). Register ini berfungsi sebagai pengaturan kerja komunikasi serial dan berkerjanya berkaitan dengan register UCSRA dan UCSRB. Register ini mempunyai lokasi I/O yang sama sebagai register UBRRH.

4.2.2 BootCode, Startup Code, Main Code, dan Header Files

Bootcode digunakan pada ATMEGA1280 untuk menyalakan beberapa periferal,, pemberian clock, pada ATMEGA1280. Pemberian clock pada ATMEGA1280 menentukan seberapa cepat mikroprosesor tersebut berjalan. Code ini secara otomatis terhasilkan pada pemograman pada software *uvision*.

Main code pada ATMEGA1280 ini digunakan untuk pendeteksian kesalahan pada perintah yang didapatkan dari OBDH, pendeteksian apakah perintah atau data yang akan dikrimkan ke Ground Station, dan menjalankan fungsi nya sebagai encoder dan decoder perintah yang akan dikirimkan oleh OBDH.

Header Code ATMEGA1280, kode ini digunakan untuk penamaan fungsi dan definisi pemograman yang telah digunakan sebelumnya. Selain itu, juga terdapat header-header yang berisi kode dasar yang digunakan untuk pemograman C, seperti STDIO.H dan beberapa fungsi lainnya yang digunakan untuk penulisan karakter ASCII atau perbandingan karakter dengan karakter lainnya, seperti contohnya sebagai perbandingan apakah perintah atau data yang dikirimkan agar tidak terjadi kesalahan atau sebagai sinkronisasi data yang akan dikirimkan.

4.3 Analisis Command and Response OBDH UI-SAT dari Perintah Ambil Data Dari Terminal Node Controller pada Tracking, Telemetry and Command Iinusat ke OBDH

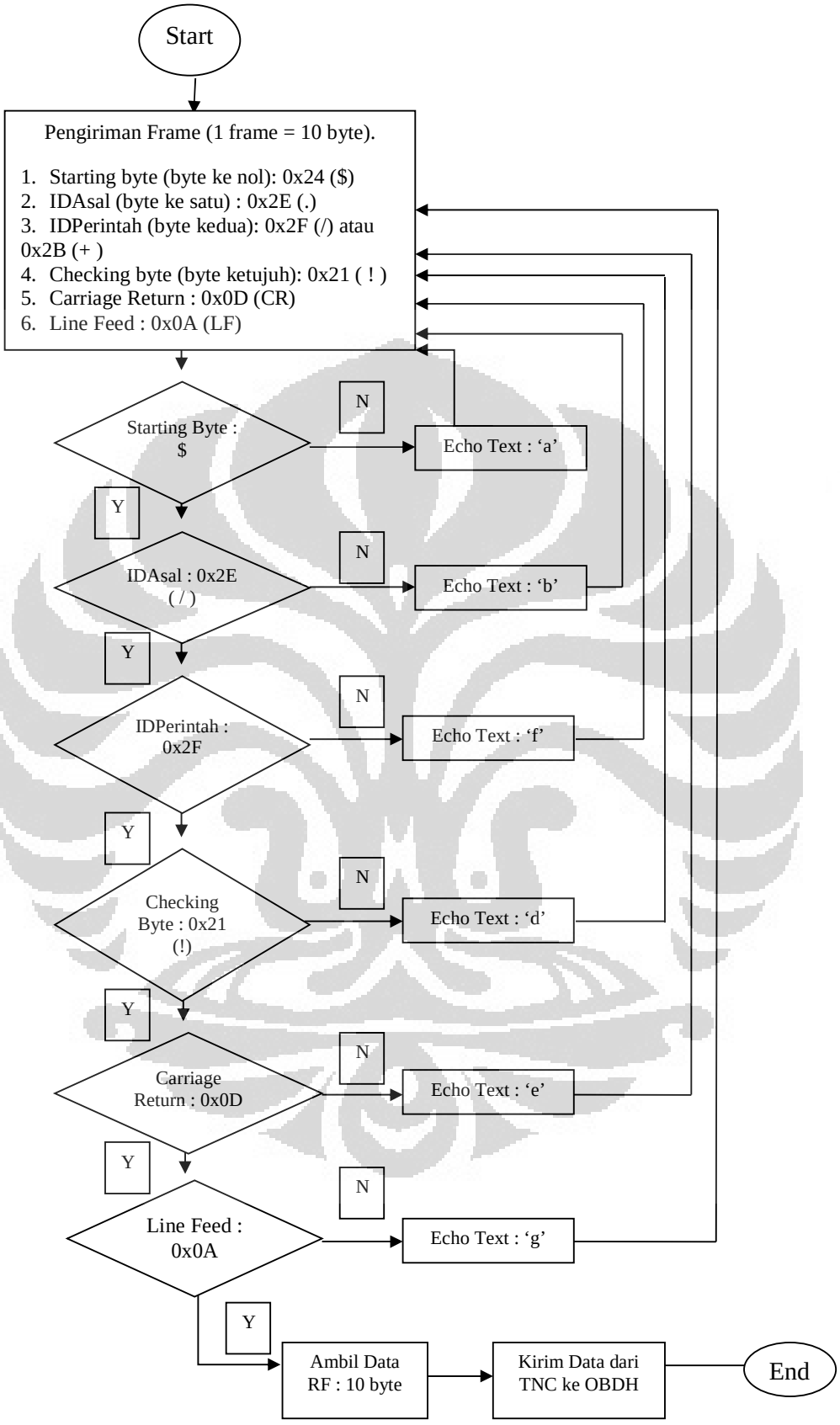
Dari pemograman yan telah dibuat, pemberian Perintah Ambil Data Dari Terminal Node Controller dibuat dari UART1. Frame yang diberikan adalah $ + 0x2E (.) + 0x2F (/) + Info Isi (4 Byte + 0x21 (!) + CR + LF). Adapun, metode yang digunakan apabila terjadi error dalam proses pengiriman data atau perintah

ke TNC atau OBDH yang membandingkan byte antara OBDH dengan TNC , TNC akan memberikan suatu respon jika terdapat kesalahan byte. Respon yang diberikan TNC adalah membalikkan byte yang salah kepada OBDH, sehingga dapat diketahui byte mana yang salah. Selanjutnya, apabila perintah telah berhasi dikirimkan dengan baik, maka TNC akan merespon perintah tersebut dengan data yang didapatkan dari TNC atau RF.

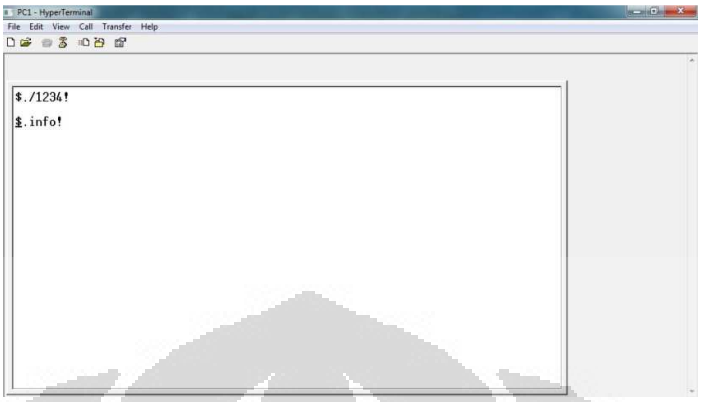
Dari hasil pemograman yang didapat, didapatkan beberapa analisis yang bisa didapatkan. Dari TNC, pemograman yang belum terlalu efisien. Hal ini dikarenakan pada pemograman, TNC menunggu perintah dari On Board Data Handling, dan selanjutnya menunggu input yang masuk dari RF. Hal ini membuat TNC masih berada pada kondisi menyala. Hal ini menjadi tidak efektif , apabila tidak ada inputan dari Radio Frequency sehingga fungsi TNC akan menjadi tertunda dan kehilangan fungsi lain yang bisa dikerjakan.

Adapun, untuk proses pengujian, pada bagian RF dari TTC diganti dengan RF virtual berupa *personal computer* yang digunakan untuk menginput suatu informasi atau data yang berguna sebagai *virtual* masuk ke dalam TNC. Karena, saluran masuk melalui kanal yang minim interferensi, maka data yang didapatkan dari TNC akan mendapatkan hasil yang sama. Pada pengujian, dipakai clock TNC sebesar 8 MHz dengan baudrate sebesar 9600bps, sedangkan pada OBDH digunakan clock sebesar 12MHz yang dikalikan sebesar 12 kali untuk mendapatkan PLL0 (Phase Locked Loop) . Setelah diperbesar 12 kali, selanjutnya dibagi empat untuk mendapatkan CPU Clocknya. Adapun, untuk mendapatkan peripheral clock nya dapat diatur dengan membagi *CPU Clock* nya dengan divider sebesar empat.

Adapun, algoritma pendeteksian kesalahan pada TNC pada perintah ambil data yaitu sebagai berikut :



Gambar 4. 1 FlowChart Ambil Data RF



Gambar 4. 2 Hasil UjiCoba pada PC1 pada saat perintah Ambil Data

Prosedur pengujian perintah ambil data adalah pengiriman perintah ambil data dari PC1 yang melewati On-Board Data Handling lalu dikirimkan ke TNC. Pada gambar 4.2 menunjukkan proses pengiriman perintah kirim data dari PC1 yang melewati On-Board Data Handling, lalu seperti yang terlihat pada gambar 4.3 menunjukkan proses pengambilan data dari PC2, lalu data yang didapat tersebut dikirimkan lagi menuju OBDH. Pada Gambar 4.2 terlihat terdapat pada baris 3 merupakan data yang didapat dari PC2. Baud Rate yang digunakan untuk perintah ambil data ini adalah 9600 bps.

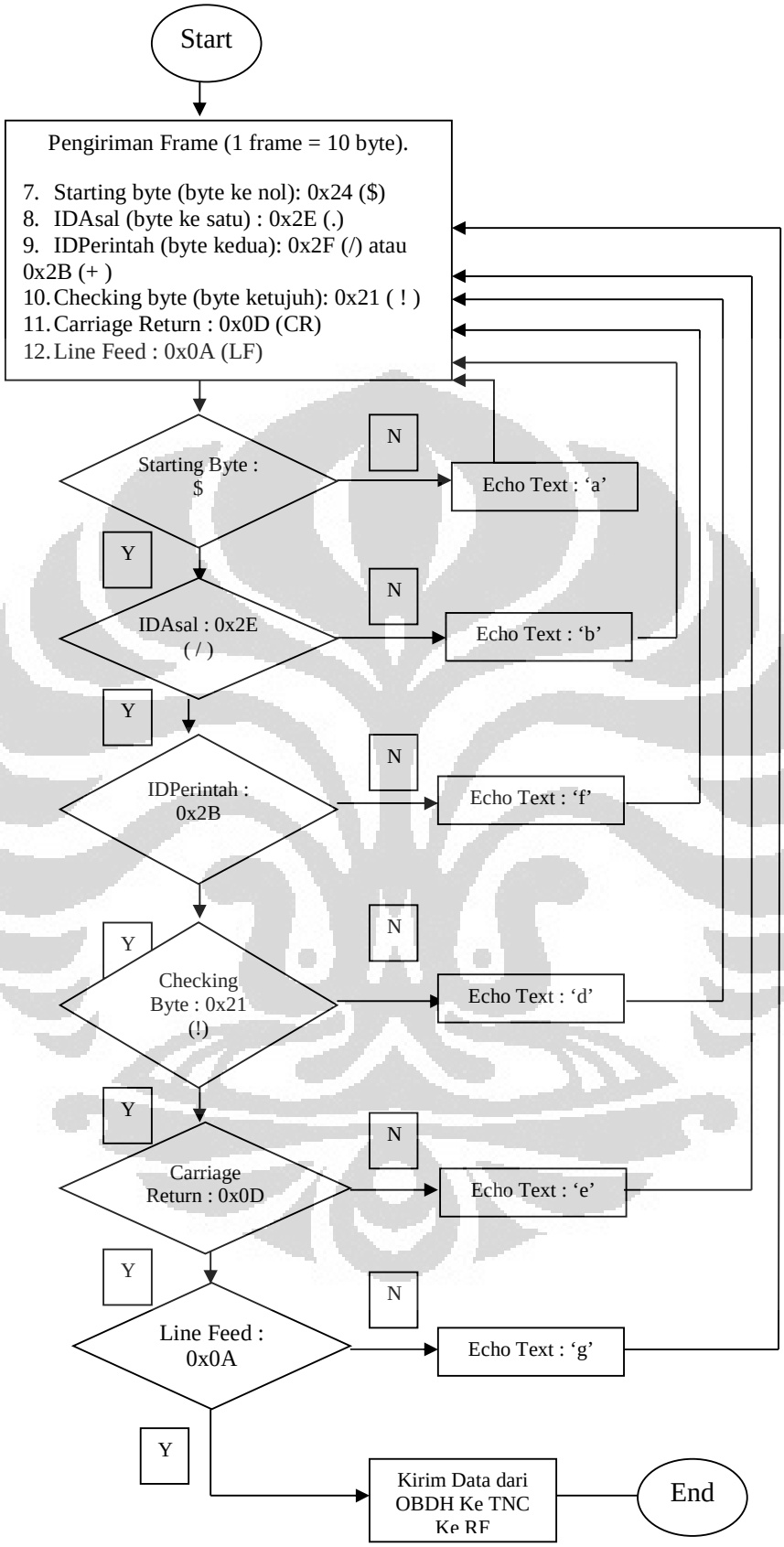
Gambar 4. 3 Hasil Ujicoba pada PC2 pada saat perintah Ambil Data

4.4 Analisis Command and Response OBDH UI-SAT dari Perintah Kirim Data ke Terminal Node Controller pada Tracking, Telemetry, and Command Iinusat

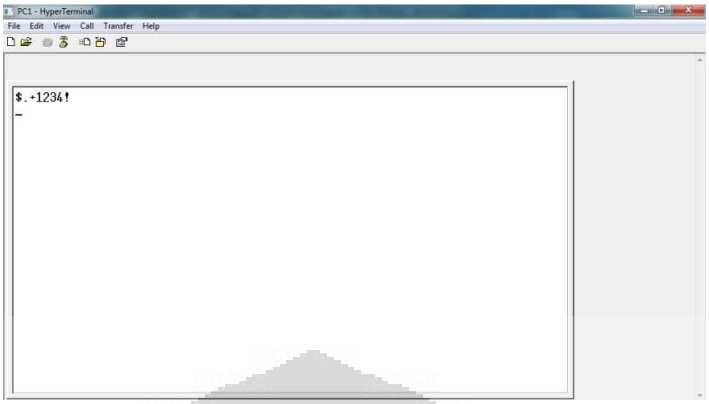
Dari pemograman yan telah dibuat, pemberian Perintah kirim Data ke Terminal Node Controller berasal dari UART0 pada OBDH. Frame yang diberikan adalah $ + 0x2E (.) + 0x2B (+) + Info Isi (4 Byte + 0x21 (!) + CR + LF). Adapun, metode yang digunakan apabila terjadi error dalam proses pengiriman data atau perintah ke TNC atau OBDH yang membandingkan byte antara OBDH dengan TNC , TNC akan memberikan suatu respon jika terdapat kesalahan byte. Respon yang diberikan TNC adalah membalikkan byte yang salah kepada OBDH, sehingga dapat diketahui byte mana yang salah. Selanjutnya, apabila perintah telah berhasi dikirimkan dengan baik, maka TNC akan merespon dengan mengembalikan/memantulkan (echo) semua data yang dikirimkan dan selanjutnya akan meneruskannnya ke bagian RF

Dari hasil pemograman yang didapat, didapatkan beberapa analisis yang bisa didapatkan. Dari TNC, pemograman yang dilakukan sudah efisien. Hal ini dapat dilihat bahwa TNC memang menunggu perintah dari On Board Data Handling, dan selanjutnya apabila decoding perintah dari On Board Data Handling telah berhasil diterjemahkan dengan baik, maka selanjutnya TNC akan mengirimkannya pada bagian *Radio Frequency*.

Adapun, untuk proses pengujian, pada bagian RF dari TTC diganti dengan RF virtual berupa *personal computer* yang digunakan untuk menginput suatu informasi atau data yang berguna sebagai *virtual* masuk ke dalam TNC. Karena, saluran masuk melalui kanal yang minim interferensi, maka data yang didapatkan dari TNC akan mendapatkan hasil yang sama.

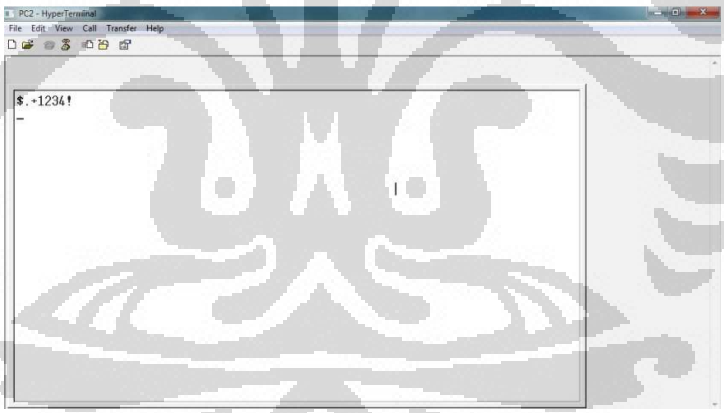


Gambar 4. 4 FlowChart Algoritma Kirim Data RF



Gambar 4. 5 Hasil UjiCoba PC1 pada saat perintah Kirim Data RF

Prosedur pengujian perintah kirim data adalah pengiriman data dari PC1 yang melewati On-Board Data Handling lalu dikirimkan ke TNC. Pada gambar 4.5 menunjukkan proses pengiriman data dari PC1 yang melewati On-Board Data Handling, lalu pada PC2 menunjukkan hasil pengiriman data tersebut dapat disampaikan data dari OBDH yang diproses di TNC. Baud Rate yang digunakan untuk perintah kirim data ini adalah 9600 bps.



Gambar 4. 6 Hasil UjiCoba PC2 pada saat perintah Kirim Data RF

4.5 Analisis Kesalahan Informasi Terminal Node Controller Tracking, Telemertry, and Command IiNUSAT dengan OBDH UI-SAT

Untuk menganalisis kesalahan informasi, maka hubungan antara TNC dengan OBDH, dan Dengan PC menggunakan Baud Rate yang berbeda-beda. Baud Rate yang semakin tinggi akan menghasilkan tranfer rate yang lebih besar pula. Oleh karena itu, dengan memasukkan Perintah Kirim Data, maka didapatkan data kesalahan informasi sebagai berikut :

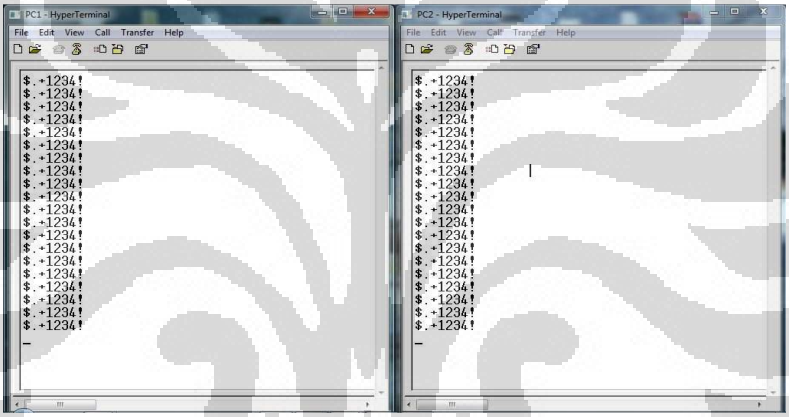
Tabel 4. 1 Tabel Kombinasi Baud Rate ATMEGA1280 dan LPC1768

Standart Baud Rate	UBRR	DLL	DLM	Peripheral Clock	Fosc	% Error
2400 bps	207	234	1	18 MHz	8 MHz	0 %
4800 bps	103	234	0	18 MHz	8 MHz	0%
9600 bps	51	117	0	18 MHz	8 MHz	0%
19,2 kbps	25	59	0	18 MHz	8 MHz	0%
38,4 kbps	12	30	0	18 MHz	8 MHz	0%
57.6 kbps	8	20	0	18 MHz	8 MHz	100%
115,2 kbps	3	10	0	18 MHz	8 MHz	100%
230,4 kbps	1	5	0	18 MHz	8 MHz	100%

Dari hasil uji coba, maka dapat dilihat terdapat persen error 100 % pada baud rate 57.600, 115.200 bps, dan 230.400 bps. Hal ini terjadi karena proses pembagian untuk mendapatkan frekuensi baud rate dari TNC atau OBDH yang

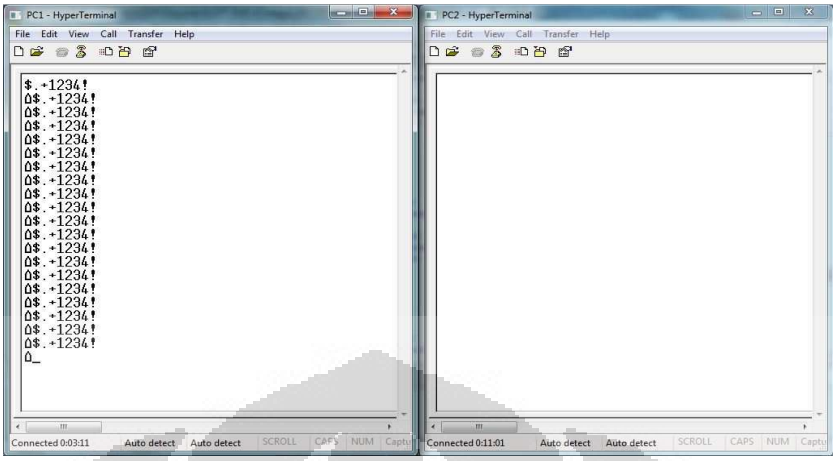
tidak optimal. 100 % Error menunjukkan pada tidak dilakukannya perintah kirim data dari OBDH menuju TNC. Selain tidak dilakukannya perintah, data yang didapat TNC juga salah sehingga fungsi perintah TNC juga tidak bisa dilakukan.

Pada hasil uji coba, baud rate dengan nilai 2400 bps, 4800 bps, 9600 bps, 19.200 bps, 38.400 bps dapat mengirimkan data yang sesuai dengan yang dikirimkan dari PC1 dan dapat diterima pada PC2.Pengujian pada uji coba ini mengirimkan frame sebanyak 20 paket, atau 20 kali percobaan. Seperti hasil uji coba, pada gambar 4.7 terlihat terdapat ada pengiriman framse sebanyak 20 kali, dan semua frame tersebut dapat dikirimkan secara sempurna ke TNC dapat menghasilkan perintah kirim data.



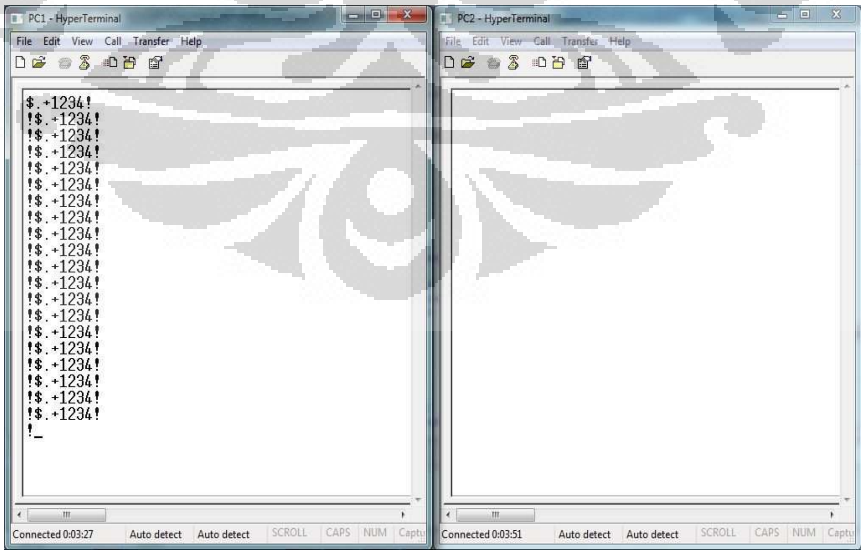
Gambar 4. 7 Hasil Uji Coba Perintah Kirim Data Dengan Baud Rate 4800 bps

Pada hasil uji coba perintah kirim data dengan baud rate 57.600 bps, maka dapat dilihat bahwa tidak ada perintah kirim data yang dapat dikirimkan dan berhasil untuk dikirimkan. Perintah tersebut memiliki kesalahan pada byte yang ada didalam frame tersebut sehinggga tidak ada perintah yang berhasil dikirimkan. Pendeteksian frame tersebut tidak dapat diketahui karena ascii yang dikirimkan ke TNC tidak sesuai dengan algoritma yang ada. Hal ini menunjukkan terdapat kesalahan perhitungan baud rate yang tidak bisa diperbaiki baik pada LPC1768 dan TNC ATMEGA1280.



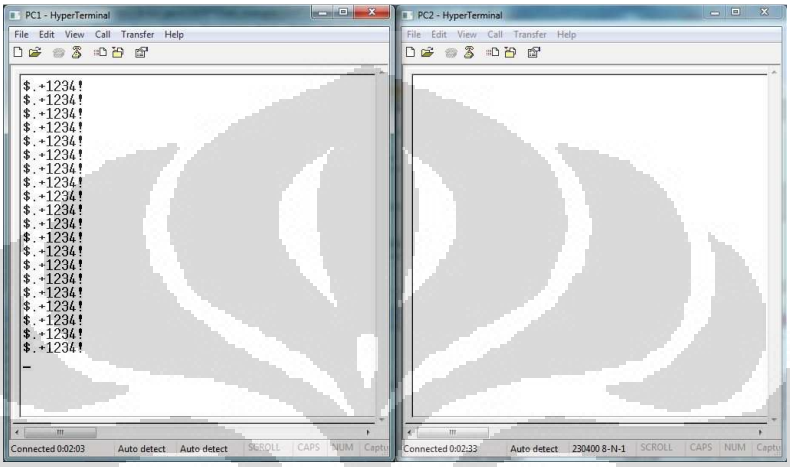
Gambar 4. 8 Hasil UjiCoba Perintah Kirim Data dengan Baud Rate 57.600 bps

Pada hasil uji coba perintah kirim data dengan baud rate 115.200 bps, maka dapat dilihat bahwa tidak ada perintah kirim data yang dapat dikirimkan dan berhasil untuk dikirimkan. Perintah tersebut memiliki kesalahan pada byte yang terdapat pada frame tersebut. Sama halnya dengan pengiriman data dengan baud rate 57600. Pendeteksian error ini tidak sesuai karena data yang dikirimkan kembali ke OBDH memiliki text yang berbeda dengan algoritma. Hal ini menunjukkan terdapat kesalahan perhitungan baud rate yang tidak bisa diperbaiki baik pada LPC1768 dan TNC ATMEGA1280.



Gambar 4. 9 Hasil Uji Coba Perintah Kirim Data dengan baud Rate 115.200 bps

Sedangkan pada hasil uji coba perintah kirim data dengan baud rate 203.400 bps, maka dapat dilihat bahwa pengiriman frame ini sama sekali tidak bisa terbaca baik LPC1768 ataupun ATMEGA1280. Hal ini dapat dilhat bahwa tidak ada respon text yang dikirimkan apabila terdapat kesalahan frame. Hal ini menunjukkan tidak ada interkoneksi dengan TNC dengan OBDH.

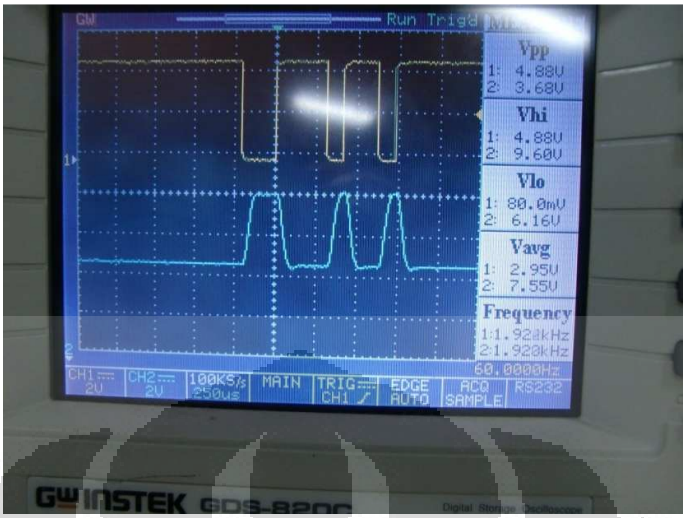


Gambar 4. 10 Hasil Uji Coba Perintah Kirim Data dengan Baud Rate sebesar 203.400 bps.

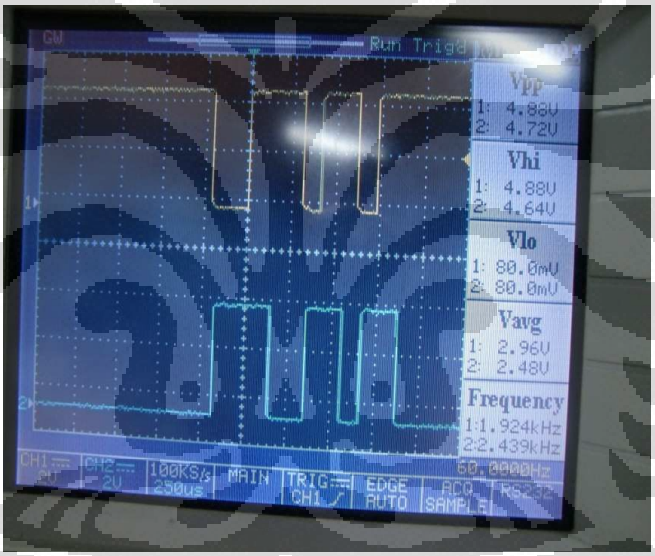
Dari hasil uji coba, maka baud rate optimal yang bisa diterapkan pada OBDH dengan TNC adalah 2400 bps, 4800 bps, 9600 bps, 14.400 bps, dan 38.400 bps dengan persen error yang mencapai hasil 0 %.

4.6 Analisis Interkoneksi TNC dengan Modem MO-96

Untuk percobaan antara TNC dengan modem, digunakan alat tambahan yaitu RS-232 to TTL. Alat ini diperlukan untuk mengubah inputan dari RS-232 to TTL. Modem ini digunakan untuk merubah dari sinyal digital menjadi sinyal analog sehinggga dapat ditransmisikan dengan RF. Adapun, dari TNC mengirimkan data ASCII ke modem MO-96secara simultan dengan jangka waktu 50ms. Adapun data ASCII yang dikirimkan adalah n.



Gambar 4. 11 Konversi antara Sinyal Digital dan Sinyal Analog



Gambar 4. 12 Perubahan Level Tegangan

Pada modem ini, terdapat dua proses utama pengubahan sinyal digital menuju analog. Langkah-langkah yang dilakukan tersebut adalah melakukan proses invers terhadap tegangan masukan. Proses iniver tersebut memiliki delay terhadap pengubahan tersebut, dan selanjutna terdapat perubahan dari sinyal analog.

BAB V.

KESIMPULAN

1. OBDH UI-SAT yang berbasiskan LPC1768 Arm Cortex M-3 harus memiliki proses dan fungsi terkait untuk melakukan suatu pemrosesan suatu data seperti fungsi UART, interrupt, dan fungsi Perintah dan Pemrosesan Data

2. Perancangan OBDH UI-SAT harus memenuhi tujuan dan hasil yang ingin dicapai oleh misi yang diembat oleh suatu nanosatelit dan diperlukan suatu perjanjian seperti kondisi frame, pendeteksian kesalahan, enkoder dan dekoder perintah, dan melakukan hubungan antara TTC dengan Ground Station.

3. Frame yang dikirimkan dari OBDH pada saat mengambil data dari RF adalah $ + 0x2E + 0x2F + Data Kosong (4 byte) + 0x21 + 0x0D + 0x0A dengan frame dari Virtual RF adalah $ + 0x2E + 0x2F + Data Isi (4 byte) + 0x21 + 0x0D + 0x0A

4. Frame yang dikirimkan dari OBDH pada saat pengiriman data dari OBDH adalah $ + 0x2E + 0x2B + Data Isi (4 byte) + 0x21 + 0x0D + 0x0A dengan frame yang masuk ke Virtual RF adalah $ + 0x2E + 0x2F + Data Isi (4 byte) + 0x21 + 0x0D + 0x0A.

5 Baud rate yang paling optimal pada interkoneksi TNC dengan OBDH adalah 2400 bps, 4800 bps, 9600 bps, 19.200, dan 38.400 bos.

DAFTAR REFERENSI

[1] Agfianto Eko Putra,Ph.D, *On-Board Satellite Controller using Arm-Based Microcontroller*, diakses pada tanggal 25 November 2011, dan diakses di http://inspire.or.id/web/index.php/2010/10/on-board-satellite-controller-using-arm-based-microcontroller/#more-53

[2] European Space Agency, 2001, *Microcontroller for On-Board Data Handling, Ramonville Saint Agne*

[3] INSPIRE – IiNUSAT, 2010, *"LAPORAN AKHIR, ANALISIS MISI dan DESAIN AWAL Indonesian Inter University Satellite – 01", INDONESIAN INTER UNIVERSITY SATELLITE- 01*

[4] J. Pérez Diestro, F. Cerezo Martínez,,*SPAIN´S MINISAT01 ON-BOARD DATA HANDLING SUBSYSTEM AND ON-BOARD SOFTWARE,* Instituto Nacional de Técnica Aerospacial, I.N.T.A : Spain

[5] Larson WJ, Wertz WJ, 1999, *Space Mission Analysis and Design: Third edition*, Microcosm, Inc: Torrance, California.

[6] *Mohit Garg,* Jennifer Sembera, 2003 ,*Cubesat-Final Report*, University of Texas : Austin

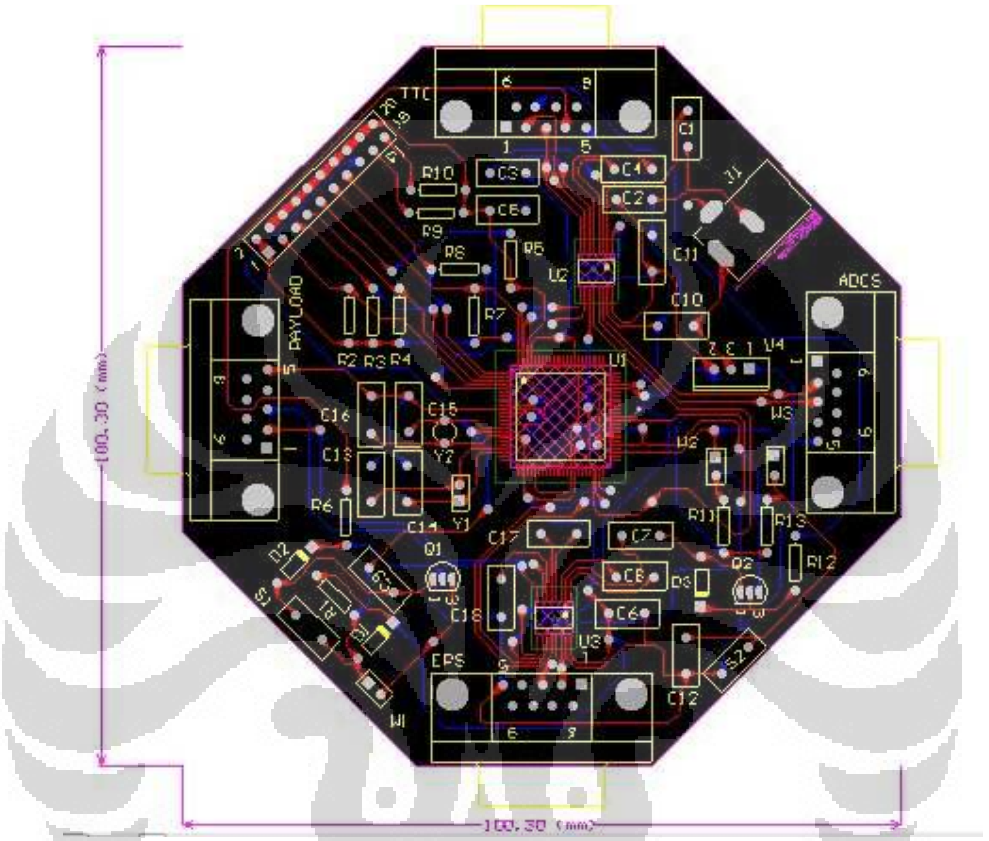
[7] NXP Semiconductors, 2010, *LPC176x Single-chip 16/32-bit microcontrollers; 32/64/128/256/512 kB ISP/IAP flash with 10-bit ADC and DAC,*

[8] Texas Instruments, Incorporated [SLLS047,L], 2004, *MAX3232, MAX232I (Rev. L),*

[9] Tri Kuntoro PRIYAMBODO, Muhammad ASVIAL, 2010, *IiNUSAT-1: The 1st Indonesian Inter-University Nano-Satellite for Research and Education,*

LAMPIRAN

LAMPIRAN 1 (Gambar OBDH UI-SAT)



LAMPIRAN 2 (Coding TNC IInusat)

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.03.9 Evaluation
Automatic Program Generator
© Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : TNC (Terminal Node Controller)
Version :
Date    : 6/2/2012
Author  : Freeware, for evaluation and non-commercial use only
Company : Mohammad Gavin Renaldi Ripharbowo
Comments:


Chip type               : ATmega1280
Program type            : Application
AVR Core Clock frequency: 8.000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 2048
*****************************************************/

#include <mega1280.h>
#include <delay.h>
```

```
#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// Get a character from the USART1 Receiver
#pragma used+
char getchar1(void)
{
char status,data;
while (1)
    {
    while (((status=UCSR1A) & RX_COMPLETE)==0);
    data=UDR1;
    if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
      return data;
    };
}
#pragma used-
```

```
// Write a character to the USART1 Transmitter
#pragma used+
void putchar1(char c)
{
while ((UCSR1A & DATA_REGISTER_EMPTY)==0);
UDR1=c;
}
#pragma used-

// Get a character from the USART2 Receiver
#pragma used+
char getchar2(void)
{
char status,data;
while (1)
    {
    while (((status=UCSR2A) & RX_COMPLETE)==0);
    data=UDR2;
    if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
       return data;
    };
}
#pragma used-

// Write a character to the USART2 Transmitter
#pragma used+
void putchar2(char c)
```

```
{
while ((UCSR2A & DATA_REGISTER_EMPTY)==0);
UDR2=c;
}
#pragma used-

// Get a character from the USART3 Receiver
#pragma used+
char getchar3(void)
{
char status,data;
while (1)
    {
    while (((status=UCSR3A) & RX_COMPLETE)==0);
    data=UDR3;
    if ((status & (FRAMING_ERROR | PARITY_ERROR |
DATA_OVERRUN))==0)
      return data;
    };
}
#pragma used-

// Write a character to the USART3 Transmitter
#pragma used+
void putchar3(char c)
{
while ((UCSR3A & DATA_REGISTER_EMPTY)==0);
UDR3=c;
}
```

```
#pragma used-

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Crystal Oscillator division factor: 1
#pragma optsize-
CLKPR=0x80;
CLKPR=0x00;
#ifdef _OPTIMIZE_SIZE_
#pragma optsize+
#endif

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTB=0x00;

DDRB=0x00;


// Port C initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTC=0x00;

DDRC=0x00;


// Port D initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTD=0x00;

DDRD=0x00;


// Port E initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In

// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTE=0x00;

DDRE=0x00;


// Port F initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
```

```
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTF=0x00;
DDRF=0x00;

// Port G initialization
// Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State5=T State4=T State3=T State2=T State1=T State0=T
PORTG=0x00;
DDRG=0x00;

// Port H initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTH=0x00;
DDRH=0x00;

// Port J initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTJ=0x00;
DDRJ=0x00;

// Port K initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTK=0x00;
```

```
DDRK=0x00;


// Port L initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTL=0x00;
DDRL=0x00;


// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0A output: Disconnected
// OC0B output: Disconnected
TCCR0A=0x00;
TCCR0B=0x00;
TCNT0=0x00;
OCR0A=0x00;
OCR0B=0x00;


// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// OC1C output: Discon.
// Noise Canceler: Off
```

```
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
OCR1CH=0x00;
OCR1CL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2A output: Disconnected
// OC2B output: Disconnected
ASSR=0x00;
TCCR2A=0x00;
TCCR2B=0x00;
```

```
TCNT2=0x00;
OCR2A=0x00;
OCR2B=0x00;

// Timer/Counter 3 initialization
// Clock source: System Clock
// Clock value: Timer 3 Stopped
// Mode: Normal top=FFFFh
// OC3A output: Discon.
// OC3B output: Discon.
// OC3C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 3 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR3A=0x00;
TCCR3B=0x00;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;
```

```
OCR3CH=0x00;
OCR3CL=0x00;

// Timer/Counter 4 initialization
// Clock source: System Clock
// Clock value: Timer 4 Stopped
// Mode: Normal top=FFFFh
// OC4A output: Discon.
// OC4B output: Discon.
// OC4C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 4 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR4A=0x00;
TCCR4B=0x00;
TCNT4H=0x00;
TCNT4L=0x00;
ICR4H=0x00;
ICR4L=0x00;
OCR4AH=0x00;
OCR4AL=0x00;
OCR4BH=0x00;
OCR4BL=0x00;
OCR4CH=0x00;
```

```
OCR4CL=0x00;

// Timer/Counter 5 initialization
// Clock source: System Clock
// Clock value: Timer 5 Stopped
// Mode: Normal top=FFFFh
// OC5A output: Discon.
// OC5B output: Discon.
// OC5C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 5 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR5A=0x00;
TCCR5B=0x00;
TCNT5H=0x00;
TCNT5L=0x00;
ICR5H=0x00;
ICR5L=0x00;
OCR5AH=0x00;
OCR5AL=0x00;
OCR5BH=0x00;
OCR5BL=0x00;
OCR5CH=0x00;
OCR5CL=0x00;
```

```
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
// INT3: Off
// INT4: Off
// INT5: Off
// INT6: Off
// INT7: Off
EICRA=0x00;
EICRB=0x00;
EIMSK=0x00;
// PCINT0 interrupt: Off
// PCINT1 interrupt: Off
// PCINT2 interrupt: Off
// PCINT3 interrupt: Off
// PCINT4 interrupt: Off
// PCINT5 interrupt: Off
// PCINT6 interrupt: Off
// PCINT7 interrupt: Off
// PCINT8 interrupt: Off
// PCINT9 interrupt: Off
// PCINT10 interrupt: Off
// PCINT11 interrupt: Off
// PCINT12 interrupt: Off
// PCINT13 interrupt: Off
// PCINT14 interrupt: Off
```

```
// PCINT15 interrupt: Off
// PCINT16 interrupt: Off
// PCINT17 interrupt: Off
// PCINT18 interrupt: Off
// PCINT19 interrupt: Off
// PCINT20 interrupt: Off
// PCINT21 interrupt: Off
// PCINT22 interrupt: Off
// PCINT23 interrupt: Off
PCMSK0=0x00;
PCMSK1=0x00;
PCMSK2=0x00;
PCICR=0x00;

// Timer/Counter 0 Interrupt(s) initialization
TIMSK0=0x00;
// Timer/Counter 1 Interrupt(s) initialization
TIMSK1=0x00;
// Timer/Counter 2 Interrupt(s) initialization
TIMSK2=0x00;
// Timer/Counter 3 Interrupt(s) initialization
TIMSK3=0x00;
// Timer/Counter 4 Interrupt(s) initialization
TIMSK4=0x00;
// Timer/Counter 5 Interrupt(s) initialization
TIMSK5=0x00;

// USART0 initialization
```

```
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud Rate: 9600
UCSR0A=0x00;
UCSR0B=0x18;
UCSR0C=0x06;
UBRR0H=0x00;
UBRR0L=0x33;

// USART1 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART1 Receiver: On
// USART1 Transmitter: On
// USART1 Mode: Asynchronous
// USART1 Baud Rate: 9600
UCSR1A=0x00;
UCSR1B=0x18;
UCSR1C=0x06;
UBRR1H=0x00;
UBRR1L=0x33;

// USART2 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART2 Receiver: On
// USART2 Transmitter: On
// USART2 Mode: Asynchronous
```

```
// USART2 Baud Rate: 9600
UCSR2A=0x00;
UCSR2B=0x18;
UCSR2C=0x06;
UBRR2H=0x00;
UBRR2L=0x33;

// USART3 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART3 Receiver: On
// USART3 Transmitter: On
// USART3 Mode: Asynchronous
// USART3 Baud Rate: 9600
UCSR3A=0x00;
UCSR3B=0x18;
UCSR3C=0x06;
UBRR3H=0x00;
UBRR3L=0x33;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
ADCSRB=0x00;

IDData=0x24;//Inisialisasi Id Starting Data ($) Untuk Transmisi Awal TTC - Ground Station - OBDH
IDTujuan=0x2E;//Inisialisasi Id Tujuan (0x01) Untuk Pengiriman TTC - Ground Station - OBDH
```

```
IDTerima=0x2F;//Inisialisasi Id Command Terima (0x02) Untuk Pengiriman Ground Station -> TNC -> OBDH

IDKirim=0x2B;//Inisialiasi Id Command Kirim (0x2B) Untuk Pengiriman OBDH -> TNC -> Ground Station

IDCS=0x21;//Inisialisasi Id Checksum (0xxx) Untuk Pengiriman TTC - Ground Station - OBDH

IDCR=0x0D;//Inisialisasi Id Carriage Return (0xxx ) Untuk Pengiriman TTC - Ground Station - OBDH

IDLF=0x0A   ;//Inisialisasi Id Line Feed (0xxx) Untuk Pengiriman TTC- Ground Station - OBDH



while (1)// Non-Stop Repeating Code

{

        for(buffer1=0; buffer1<10; buffer1++)//Setting Besar Buffer Untuk Komunikasi TTC - OBDH

        {

        obdh[buffer1]=getchar();

        }


        hai=obdh[0];//Pemisahan antara byte-dengan byte pada satu frame

        hai1=obdh[1];

        hai2=obdh[2];

        hai3=obdh[3];

        hai4=obdh[4];

        hai5=obdh[5];

        hai6=obdh[6];

        hai7=obdh[7];

        hai8=obdh[8];

        hai9=obdh[9];
```

```
if(hai!=IDData)//Penyeleksian Frame ID ($) Pada TNC - OBDH
{
 putchar('a');
}

else if(hai1!=IDTujuan)//Penyeleksian Frame ID (0x01) Pada TNC - OBDH Untuk Tujuan
{
 putchar('b');
}

else if(hai7!=IDCS)//Penyeleksian Checksum Parity Pada TNC - OBDH
{
 putchar('d');
}

else if(hai8!=IDCR)//Penyeleksian Carriage Return Pada TNC-OBDH
{
 putchar('e');
}

else if(hai9==IDLF)//Penyeleksian Line Feed Pada TNC-OBDH
{
 if(hai2==IDTerima)//Peyeleksian Command Decoder untuk Receiver dari RF to OBDH
 {
 bumi();
 }
```

```
        else if(hai2==IDKirim)//Penyeleksian Command Decoder untuk
Transmitter dari OBDH To RF
        {
        satelit();
        }

        else
        putchar('f');
        }

        else
        {
        putchar('g');
        }
};
}
void bumi(void)//Fungsi Dari RF System -> TNC -> OBDH
{
      for(buffer=0; buffer<10; buffer++)// Setting Besar Buffer untuk Komunikasi
TTC - Virtual Ground Station
        {
        ground[buffer]=getchar1();
        }

        halo=ground[0];//Pemisahan byte dengan antara satu frame
        halo1=ground[1];
        halo2=ground[2];
        halo3=ground[3];
        halo4=ground[4];
```

```
        halo5=ground[5];
        halo6=ground[6];
        halo7=ground[7];
        halo8=ground[8];
        halo9=ground[9];

    putchar(halo);
    putchar(halo1);
    putchar(halo2);
    putchar(halo3);
    putchar(halo4);
    putchar(halo5);
    putchar(halo6);
    putchar(halo7);
    putchar(halo8);
    putchar(halo9);
}

void satelit(void)//Fungsi Dari OBDH -> TNC -> Ground Station
{
    putchar1(hai);
    putchar1(hai1);
    putchar1(hai2);
    putchar1(hai3);
    putchar1(hai4);
    putchar1(hai5);
    putchar1(hai6);
    putchar1(hai7);
```

```
        putchar1(hai8);
        putchar1(hai9);
}
```

LAMPIRAN 3 (Coding OBDH UI-SAT)

```
/*****************************************************************************
**********

*****************************************************************/
#include "lpc17xx.h"
#include "type.h"
#include "uart.h"

extern volatile uint32_t UART0Count;
extern volatile uint8_t UART0Buffer[BUFSIZE];
extern volatile uint32_t UART1Count;
extern volatile uint8_t UART1Buffer[BUFSIZE];

/*****************************************************************************
************
**   Main Function  main()
*****************************************************************************
**********/
int main (void)
{
 /* SystemClockUpdate() updates the SystemFrequency variable */
 SystemClockUpdate();

 UARTInit(0, 9600);  /* baud rate setting */
 UARTInit(1, 9600);  /* baud rate setting */

 while (1)
 {                                  /* Loop forever */
       if ( UART0Count != 0 )
```

```
        {
          LPC_UART0->IER = IER_THRE | IER_RLS;                    /*
Disable RBR */
          UARTSend( 1, (uint8_t *)UART0Buffer, UART0Count );
          UART0Count = 0;
          LPC_UART0->IER = IER_THRE | IER_RLS | IER_RBR;          /* Re-
enable RBR */
        }
        if ( UART1Count != 0 )
        {
          LPC_UART1->IER = IER_THRE | IER_RLS;                    /*
Disable RBR */
          UARTSend( 0, (uint8_t *)UART1Buffer, UART1Count );
          UART1Count = 0;
          LPC_UART1->IER = IER_THRE | IER_RLS | IER_RBR;          /* Re-
enable RBR */
        }
 }
}

/******************************************************************************
**                            End Of File
******************************************************************************/
```