HARDWARE DESIGN OF CDMS FOR IITMSAT

A THESIS

to be submitted by

Sanak N K

EE11M059

for the award of the degree

MASTER OF TECHNOLOGY

under the guidance of

Prof. Nitin Chandrachoodan

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS

CHENNAI-600036

CERTIFICATE

This is to certify that the thesis titled **"HARDWARE DESIGN OF CDMS FOR IITMSAT,** submitted by Mr. Sanak N K, to the Indian Institute of Technology Madras, Chennai for the award of the degree of Bachelor of Technology and Master of Technology, is bonafide record of research work done by him under my supervision. The contents of the this thesis, in full or parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Nitin Chandrachoodan

Project Guide

Assistant Professor

Dept. of Electrical Engineering

IIT-Madras, Chennai-600036

Place: Chennai

Date: 23^{th} May 2013

ACKNOWLEDGEMENT

I would like to take this opportunity to express my sincere gratitude to my project guide **Dr. Nitin Chandrachoodan** Assistant Professor Department of Electrical Engineering Department. He put a consider amount of time and effort to explain the concepts related to project and helped me to understand the project clearly. His guidance has broadened my Knowledge and thinking level. Interaction with him made me realise the values of diligence, patience and dedication.

I would like to extend my thanks to Akshay Gulati , Ankith and Satish for devoting their time in discussing ideas with me and giving their invaluable feedback. I also thank Amit , Chaithanya, Karthikeyan , seetal , Syed and Vikas for maintaining a healthy atmosphere in lab.

I thank Anoop M K, Lino A , Sreejith P Das and Sujitha M for the pleasant company I have had during my stay at IIT Madras.

I owe very much to my grand mother , brother and sister for their constant support and motivation throughout my stay away from home.

Sanak N K

Abstract

IITM student Satellite is a distributed structure of Micro controllers. The Command and Data Management functions of IITMSAT is performed by CDMS subsystem which is also called On Board Computer. A topology connecting all subsytems has chosen and implemented such that inter subsystem communications and memory storage is done in an efficient way.

For the design of CDMS block, bus protocols , microcontroller ,external memory and other hard ware components were chosen. Other subsystems and external memory has been interfaced with CDMS. For the reliability improvement of the inter communication bus, protection circuits has been implemented and tested. Analysis of back up functionality that can be provided by the CDMS block is also an objective that has been achieved in this project.

Contents

List of Tables

List of Figures

ABBREVIATION

ADCS	Attitude Determination and Control System
CAN	Controller Area Network
CDMS	Controll and Data Management System
COM	Communication System
CS	Chip Select
DMA	Direct Memory Access
DSO	Digital Storage Oscilloscope
EFSL	Embedded File Systems Library
EPS	Electrical Power Subsystem
FAT	File Allocation Table
FSK	Frequency Shift Keying
GMSK	Gaussian Minimum Shift Keying
GS	Ground Station
HEPD	Fractional Frequency Reuse
HK	House Keeping
I2C	Inter Integrated Bus
ISI	Inter Symbol Interference
MISO	Master In Slave Out
MOSI	Master Out Slave In
OBC	On Board Computer
RAM	Random Access Memory
SC	Science
SCL	Serial Clock Line
SD	Secure Digital
SPI	Serial Peripheral Interface
USCI	Universal Serial Communication Interface
USI	Universal Serial Interface

Chapter 1

INTRODUCTION

1.1 IITM Student Satellite Mission

The ionosphere of earth is a shell of charged particles which contains energy particles such as protons ,electrons,other charged molecules etc. It extends from 50 km from earth's surface to approximately 1000km. This shell of charge particles are responsible for several earthly phenomena. So a detailed study about the changes in particle concentration can help in earthquake predictions,weather predictions etc. For accurate measurements, a system which measures on going changes in space radiation has to be built. For this purpose a satellite which measures particle energy spectrum has to be built and launched into ionosphere.

The mission of the IITMSAT project is to launch a nano satellite of size 30 x 30 x 30 cm3. It will have an on board High Energy Particle Detector and will be launched in to Low Earth Orbit of earth. HEPD will count the number of protons and electrons in the ionosphere and will calculate energy spectrum of particles[1]. Satellite will have two types of data. Science Data which is the data regarding energy spectrum calculated over wide range of positions of IITMSAT. The other data is House Keeping data which comprises of details about health of the satellite measured through different sensors. When satellite establishes a link with GS , it sends both SC and HK data to ground.

1.2 IITMSAT Architecture

IITMSAT is a distributed system with five subsystem. Each subsystem will have their own micro controller to perform functions assigned to them.

- CDMS/OBC

It responsible for collecting all HK data from various subsystems and storing it in on board memory. Depending upon commands from GS it performs mode transitions of satellite by issuing commands to various subsystems.

- HEPD

HEPD will collect all SC data and store in the on board memory.

- COM

COM subsystem performs up link and down link data transfers.It is responsible for reception of commands from GS and their decoding. It sends SC and HK data to GS.

- ADCS

Attitude control of satellite is done by this Subsystem. Micro controller of this subsystem will collect data from different sensors and actuators and process them for accurate attitude control of satellite.

- EPS

Power generation and distribution to other subsystems are manged by EPS. It also runs the software for beacon transmit or .

1.3 Problem Statement

The goal of the project is to design and develop CDMS subsystem for IITMSAT, which includes the following objectives

- Development of an efficient topology connecting all sub systems such that data transfer and memory management is done in an efficient manner.
- Selection of hardware components for CDMS subsystem ,such as micro controllers ,memory device etc..
- Communication with other subsystem micro controllers need to be implemented such that CDMS can collect HK data and can issue commands to them. A suitable multipoint bus protocol need to be chosen and implemented.
- Interfacing of memory device with CDMS micro controller. Suitable drivers need to be developed.
- Development of Hardware structures to protect the Bus from single node failures, and their testing.
- The entire CDMS structure need to be implemented and tested on a PCB .

1.4 Document Outline

The document outline is given below

- Chapter 2:

Discusses about various topology considered ,selection of bus protocol and selection of suitable micro controller for CDMS structure.

- Chapter 3:

Describes about implementation and testing of bus protocol chosen for CDMS to communicate with other subsystems.

- Chapter 4:

provides detail description of external memory interface and its driver development.

- Chapter 5:

goes through implementation and testing of protection circuits.

- Chapter 6:

Talks about implementing some extra functions to CDMS structure such that it can act as back up for other subsystem.

- Chapter 7:

Conclusion and future work.

Chapter 2

IITMSAT BUS ARCHITECTURE

Payload will continuously collect data in every 3.6 us. They are then converted in to a frame called fine frame. This is called Science Data (SC) .It has to be stored in a Memory card. Health data from various subsystems need to be collected at regular interval which is called House Keeping (HK) data. CDMS is collecting HK data and storing it in Memory Card. When Satellite is over Ground Station (GS) this stored data is transferred to COM and From COM it will be transmitted to GS. During the mode while the satellite is sending data to the Ground station, the data rate is limited by the capacity of the COM subsystem, which is about 19.2[kbps].

A suitable bus topology need to be implemented ,so that an efficient way of data storage in memory card and transfer of data to COM subsystem is provided.Various bus topologies were considered.

2.1 Bus topologies Considered

2.1.1 Topology 1

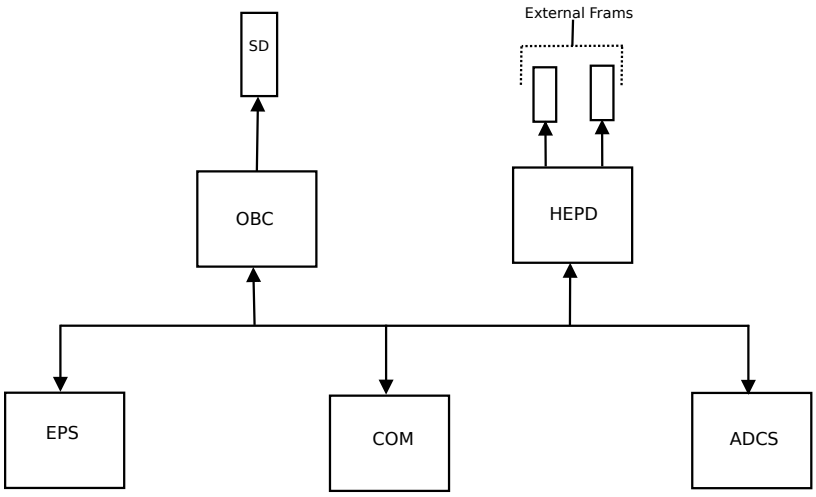

Figure 2.1: Topology 1

Memory card is connected to the OBC and payload sends SC data over inter communication bus to OBC. HEPD is connected to external FRAMS. FRAMs are external memories that have very high access times, comparable to that of RAMs. They are non-volatile. The two memory architecture helps to read from one FRAM while the HEPD writes to the other FRAM.

Advantage

- Main memory, SD card is accessed by only one Micro controller.

Disadvantages

- Main bus will be always busy as OBC will be collecting SC data from PAYLOAD
- Time for PAYLOAD to perform other actions such as histogram creation will be limited by size of FRAM

- During transmission of data to Ground Station (GS) Memory card will be accessed by OBC. Some arrangements will be needed for storing of SC data in to SD card during that period.

2.1.2 Topology 2

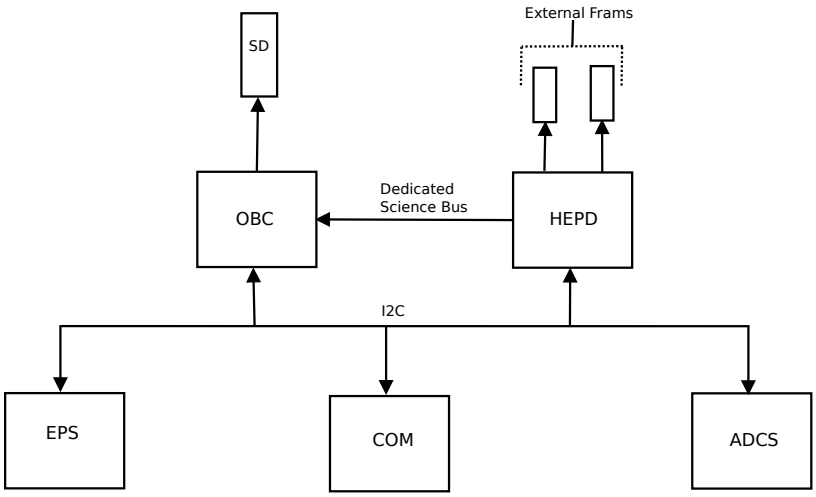

Figure 2.2: Topology 2

In this topology there is a dedicated connection between PAYLOAD and OBC. So SC data can be send over this. In this topology Main Bus will be free for HK data.

Disadvantage

- Dedicated connection need to be a high speed connection so that all science data will be written to SD card. This will result in higher power consumption.

2.1.3 Topology 3

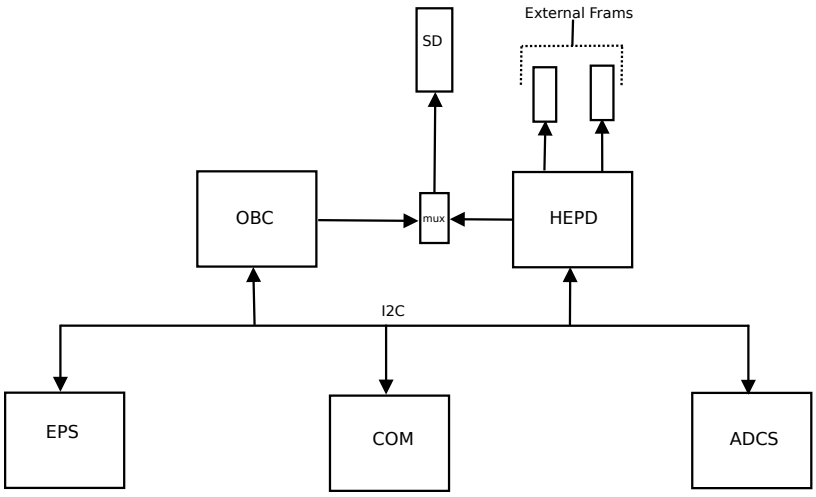

Figure 2.3: Topology 3

In topology 3, memory card is shared between payload and OBC. There is mux which controls the interface with Memory card and the two subsystems. OBC sends control signal to mux to decide which subsystem will do data transfer between memory card at particular time. There are Micro controllers with large internal memories so external FRAMs can be avoided if possible for PAY LOAD .

Disadvantage

- In this topology data to COM is transferred through main bus .So speed of data transfer will depend on MAIN bus speed which is going to be relatively slow.

2.1.4 Topology 4

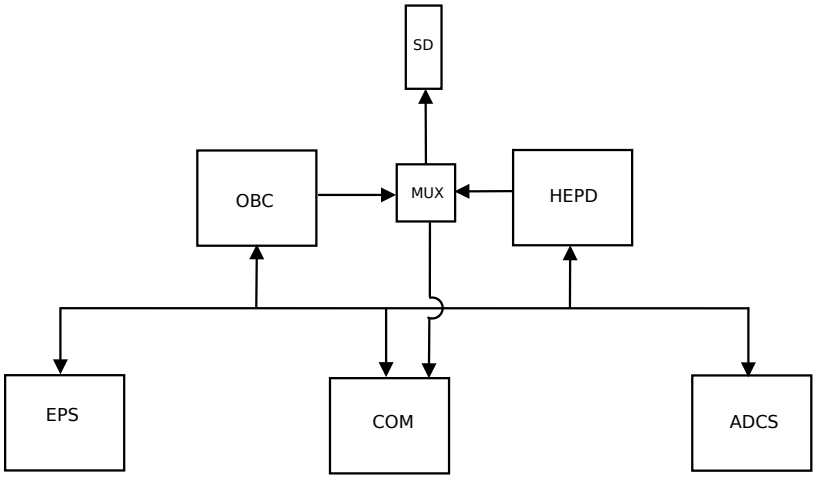

Figure 2.4: Topology 4

In this Topology Memory card is shared between OBC , PAYLOAD and COM. OBC will write HK data to SD card . When satellite is over GS, COMM will directly take data from memory card.

Disadvantage

- Sharing of Memory card between 3 micro controllers is a complex issue. Micro controller's access to SD card need to be managed well.

2.1.5 Topology 5

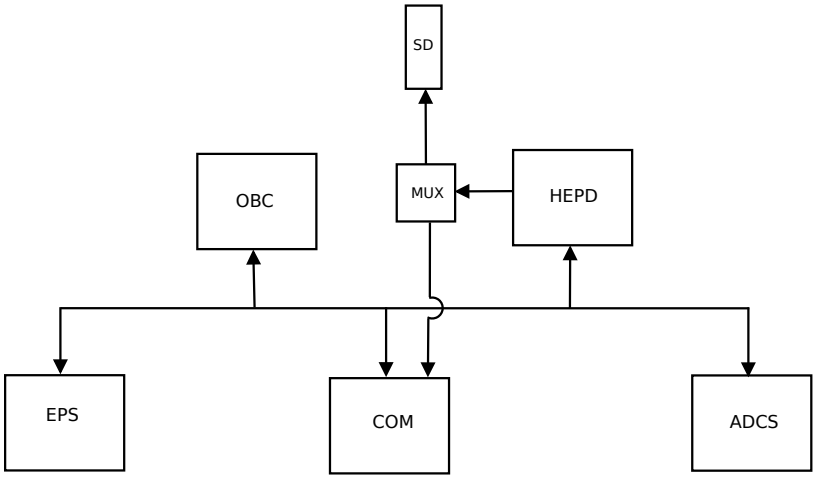

Figure 2.5: Topology 5

In this topology OBC does not have direct connection between memory card. HK data will be send to COM which will store it in Memory card. In this topology COM is the only subsystem which reads from memory card .Once functionalities of COM and PAYLOAD are fixed it is possible to merge OBC with COM so that number of micro controllers can be reduced.

Before selecting a topology actual data rates needed to be measured such as memory read / write speeds of three sub systems.Main bus transfer rate ,what kind of science bus can be used whether it is feasible to implement etc.

2.2 Main Bus Selection

IITM student satellite is a distributed system with each subsystem having their own micro controller . In order to transmit data between them a bus needs to be implemented between them. The main bus will provide communication between the subsystems. OBC has to collect health data from each subsystem through this bus. And the protocol for the bus need to be multipoint as it is connected to five subsystems.

There are two methods of data transfer parallel and serial. As parallel communication suffers from Inter Symbol Interference (ISI) and high bus capacitance due to high number of wires , serial communication was chosen.

for the selection of bus protocol different criterion's were analyzed.Reliability ,immunity towards radiation, data rates,power consumption , support to multipoint communication support to all the micro controllers used in IITM student satellite were considered.

2.2.1 SPI

SPI interface is four wire serial interface used by many micro controller peripherals.The SPI bus, which operates at full duplex (means, signals carrying data can go in both directions simultaneously), is a synchronous type data link setup with a Master / Slave interface. It can support up to 1 mega baud or 10Mbps of speed. Both single-master and multi-master protocols are possible in SPI. But the multi-master bus is rarely used .

Disadvantages

- Reliability is less compared to other two.
- Not commonly used in space for multipoint communication.

2.2.2 CAN

The Controller Area Network (CAN) is a two wire differential protocol.It is commonly used for communication between micro controller and other peripherals within a vehicle. CAN bus can support up to 1mbits/s of data rate and wire lengths less than 40 meters.

Disadvantage

- MSP430 a low power micro controller which is used for some of the subsystem cannot support CAN protocol
- Higher power consumption compared to I2c

2.2.3 I2C

I2C is a two wire interface used for low speed peripherals.I2C is half duplex as it uses only two wires serial clock line (SCL) and serial data line(SDA). It supports single master and multi-master protocols. In multislave operation each slave is given its own unique address.

Considering all the facts it was found that i2c communication is the best choice for main bus protocol.

2.3 Choice Of Micro Controller for OBC

A suitable micro controller need to be chosen for performing CDMS functions. Several Micro controllers were considered. Selection is done based on several criteria. These are the following requirements need to be satisfied by the OBC micro controller.

- Power consumption

As satellite has a few watts of power available for various subsystems it is important that the micro controllers should be low in power consumption. In certain modes of operation, OBC will be put in sleep mode. So stand by power consumption should also be low.

- Computational performance

Micro controll should have enough resources to perform all functions to be performed by OBC.

- I/O interfaces

Communication with other sub systems is very important constraint for OBC. There should be communication modules for storage of data to memory device. Micro controller should have in good built peripherals for handling all communication related tasks.

- Temperature Range

The temperature will vary significantly depending on the location of the satellite. The temperature range has been set to 0°C to 40°C for normal operation. But in special case the temperature might exceed this range in both directions. So Micro controller should have good temperature tolerance.

- Already used in space

It desirable to select a micro controller which has worked in previous space applications.

Following Micro controllers were considered for OBC subsystem.

2.3.1 ARM Micro controller

ARM micro controllers has good in built support for complex calculations. It is suitable for applications which needs lot of computational work. It is used in space applications by other universities. But the power consumption of ARM when put in stand by mode is high. As OBC need not want to perform any complex calculations and will be in stand by mode most of the time ARM processor will not be suitable for OBC.

2.3.2 ATMEL Avr Micro controllers

Atmel AVr micro controllers have good i/o support.They are less power consuming compared to ARM micro controller. But these micro controllers are not usually used in space applications as they are not resistant to radiation effects.

2.3.3 MSP430 micro controllers

MSP430 is low power consuming micro controller. In stand by mode it consumes few μwatts of power. It has several i/o peripherals which supports various types of communication protocols. It is used in space applications by several other universities. Compared to other micro controllers it offers high radiation resistance. From initial analysis it is found that MSP430 offers best performance for our application.

2.4 Back Ground on MSP430

MSP430 is used in low power embedded applications. In sleep mode it draws current less than 1 μA. It has six different low power modes. It can disable clocks and CPU when they are not used. Its wake up time is less than 1 μs which also helps in reduction of power consumption. Maximum CPU clock frequency supported by MSP430 is 25 MHz. The device has several configurations and comes with several peripherals[5].

General Purpose I/O

As is standard on micro controllers, most pins connect to a more specialized peripheral, but if that peripheral is not needed, the pin may be used for general-purpose I/O. The pins are divided in to groups of eight and each of it is called a port. They can be controlled through 8 bit registers.

Timers

- Basic timer (BT)

The BT has two independent 8-bit timers that can be cascaded to form a 16-bit timer/counter. Timers can be accessed for read and write through software. BT can also support RTC. An internal calendar compensates for months with less than 31 days and includes leap-year correction.

- Real-Time Clock

RTC_A/B are 32-bit hardware counter modules that provide clock counters with a calendar, a flexible programmable alarm, and calibration. RTC has a back up power system which enables it to work even if main supply fails.

- 16-bit timers

Timer_A, Timer_B and Timer_D are asynchronous 16-bit timers/counters with up to seven capture/compare registers and various operating modes. The timers support multiple cap-

ture/compares, PWM outputs, and interval timing. They also have extensive interrupt capabilities.

- Watchdog (WDT+)

The WDT+ can be used for controlled system restart in case of system software failures. If the selected time interval expires, a system reset is generated. In order to avoid unwanted system rebooting, software should reset watch dog timer before its time interval expires. WDT+ also can be used as normal timers.

Communication Peripherals

Different families of MSP430 offers various peripherals such as USI,USCI,USART, USB etc. This peripherals can support communication protocols such as SPI ,I2C, UART etc. A single peripheral will have different channels and it can support more than one type of communication. Some of these modules are extremely useful for our application such as USCIB. It is described in detail in next chapter.

Direct Memory Access (DMA) Controller

The DMA controller transfers data from one address to another across the entire address range without CPU intervention. The DMA increases the throughput of peripheral modules and reduces system power consumption. This module has different channels. DMA will be use full for transfer of data between Memory devices and OBC in our application.

SUMMARY

This chapter discusses about different bus topologies considered and selection of micro controller for OBC. After gaining proper knowledge about data rates and other hard ware implementation limitations, one of the topology need to be chosen. Depending upon topology a MSP430 micro controller will be chosen for OBC.

Chapter 3

I2C PROTOCOL

I2C bus was introduced by Philips semiconductors . It uses only two wires

- SCL (Serial Clock Line)
- SDA (Serial Data Line)

Transfer occur between master and slave.Each slave will have unique address.Usually address will be 7 bits long. Master starts data transmission and provides clock . Master addresses slaves, manages data transfer and terminates it.

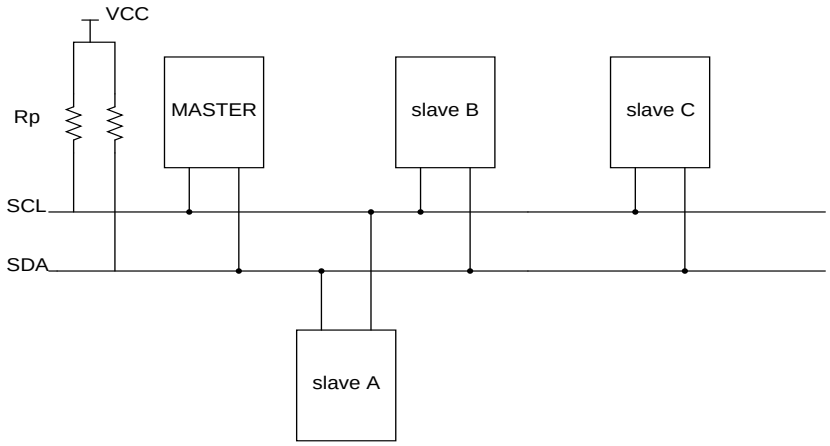

Figure 3.1: I2C bus connection diagram

Figure shows single master multislave arrangement. I2C supports multimaster also. In multimaster mode collision detection and arbitration are there to prevent data corruption if

two masters try to transfer data at same time.

Data rates of i2c can be

- up to 100 kbits/s in standard mode
- up to 400 kbits/s in fast mode
- up to 1 mbits/s in fast plus mode
- up to 3.4 mbits/s high speed mode

MSP430 can only support first two modes.Since transmission rate to ground station is going to be at 19.2 kbits/s ,low data rates such as 100 kbits/s for main bus of IITMSAT is found acceptable in the first analysis.

3.1 I2C PROTOCOL

3.1.1 SCL AND SDA LINES

Due to the variety of different technology devices (CMOS, NMOS, bipolar) that can be connected to the I2C-bus, the levels of the logical ‘0’ (LOW) and ‘1’ (HIGH) are not fixed and depend on the associated level of VDD. Input reference levels are set as 30 % and 70 % of VDD; VIL is 0.3VDD and VIH is 0.7VDD[2].

3.1.2 DATA VALIDITY

For each bit transfered there will be one clock generated by the master.Data line should remeain at high or low when clock signal is high.Data can change its state from ’HIGH to LOW’ or ’LOW to HIGH’ only when clock line is low.

3.1.3 START AND STOP CONDITION

Master initiates communication by giving out a start condition.Communication is terminated when master gives a stop condition. A HIGH to LOW transition on the SDA line

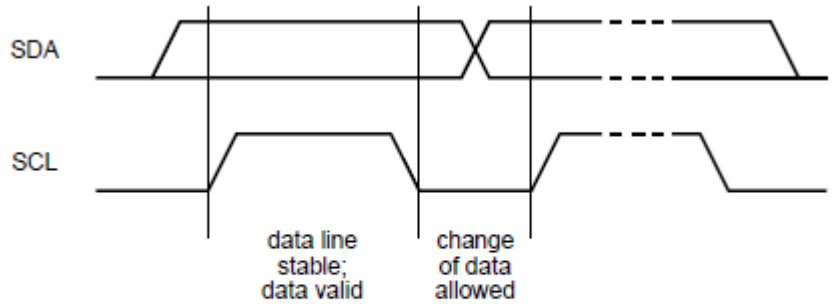

Figure 3.2: Bit transfer on I2C bus[2]

while SCL is HIGH defines a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition.

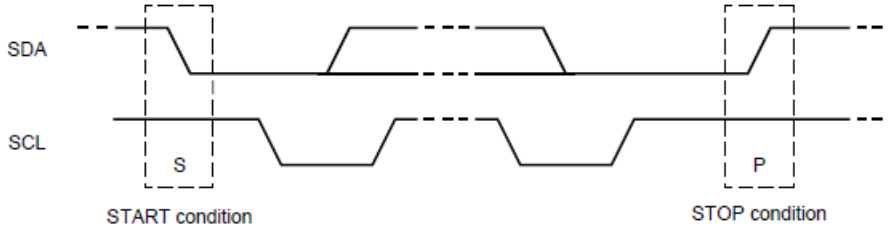

Figure 3.3: START and STOP condition[2]

BUS will be busy after START condition. It will become free some time after STOP condition. BUS will be busy if a repeated start condition occures.Slaves connected to bus need to detect START and STOP condition. Some slave devices will have integrated hardware to detect START and STOP conditions. In other devices with no such interface have to sample the SDA line at least twice per clock period to sense the transition.

3.1.4 DATA FORMAT

Data is placed in data line in the form of 8 bits bytes.For each bit clock is provided on SCL line.There should be acknowledge after each byte transferred.The number of bytes that can be transferred between START and STOP condition is unrestricted.Data is transferred with the Most Significant Bit (MSB) first . If a slave has to do some actions such as interrupt routine in between two byte data transfer ,it can hold the clock line low. Then Master goes in to wait state. Data transfer then continues when the slave is ready for another byte of data and releases clock line SCL.

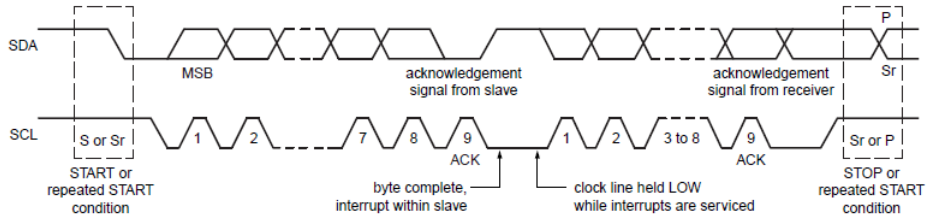

Figure 3.4: Data transfer[2]

3.1.5 SLAVE ADDRESSING

After START condition master gives slave address on data line which is 7bits long and one bit (R/W) which determines direction of data transfer.If R/W bit is zero it indicates master will transmit data(WRITE) . If the R/W bit is one master will receive data(READ) from slave . A data transfer is always terminated by a STOP condition (P) generated by the master. However, if a master still wishes to communicate on the bus, it can generate a repeated START condition (Sr) and address another slave without first generating a STOP condition.

3.1.6 ACKNOWLEDGE (ACK) AND NOT ACKNOWLEDGE (NACK)

After each byte transfer there will be Acknowledge in I2c communication. After each data transmission ,receiver should give ACK indicating data has been received success-

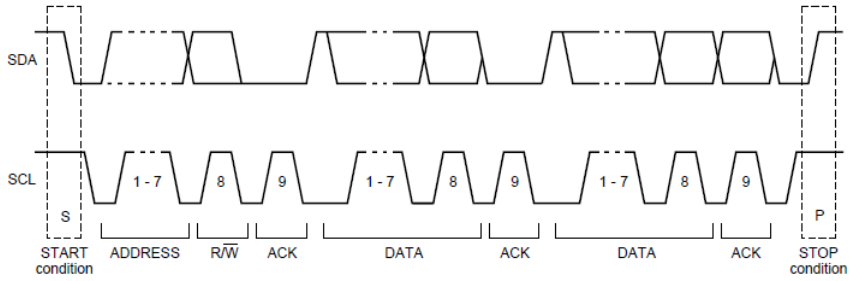

Figure 3.5: I2C addressing[2]

fully.Transmitter releases SDA line during ACK clock pulse. Then if data was received success fully receiver will keep SDA line low for the high period of clock.

There are five conditions which result in NACK condition

- No slave is there in the bus with address given by master.
- As receiver is performing some actions it is unable to respond to transmission call given by MASTER.
- During the transfer, the receiver gets data or commands that it does not understand.
- During the transfer, the receiver cannot receive any more data bytes.
- A master-receiver must signal the end of the transfer to the slave transmitter.

3.2 I2C HARDWARE

Hardware diagram of I2C master and slaves is shown in figure 3.6 . In I2C active devices pull down the lines instead of pulling up. If active devices were pulling up lines,then when two devices try to drive the bus to two different values, there will be problems. If none of the devices are active, pull up resistance Rp keep lines at VCC. The devices therefore need to have open drain output. There will be only one n-channel transistor in between ground and output.

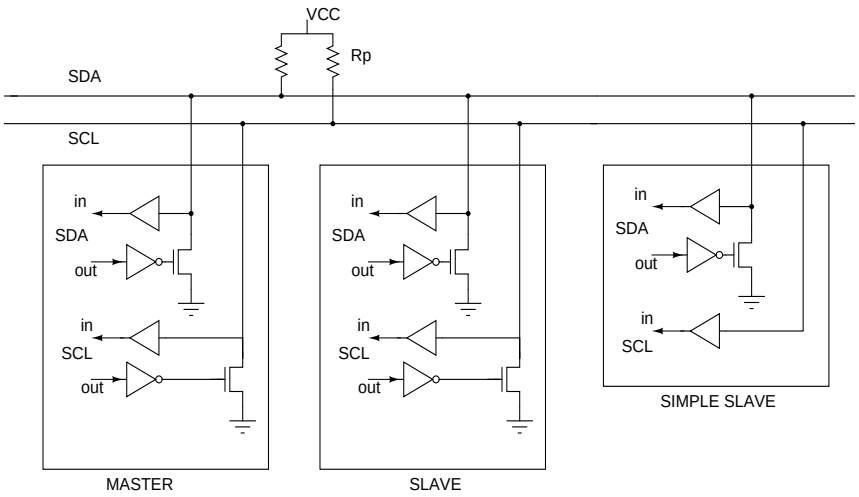

Figure 3.6: Electronic interface of the I2C bus

The pull-up resistor Rp holds the line at VCC when there is no activity, so both the clock and data idle high. When a single device is on, NMOS of that device is turned on which pulls the line to VSS. Current flows to ground through the pull-up resistor . Nothing changes if a second n-MOSFET on the same line on a different device is turned on because it has the same effect as the first one. There are no short circuits between VCC and VSS because there are no transistors connected between the outputs and VCC.

Contention arises if one device tries to write a 1 to the bus and a second writes a 0. The second device “wins” because its output actively pulls the line down, while the device with an output of 1 does nothing itself, but relies on the pull-up resistor to provide the voltage. The value of 1 is therefore called recessive while 0 is dominant. The arrangement is also called wired and because the value on a line is 1 only if all the devices write a 1 to it. The line is pulled down to 0 if one or more devices writes a 0.

A device that writes to the bus when there is any possibility of contention must monitor the value on the bus. It has lost control and must relinquish the bus if it detects any difference between the value on the bus and the value that it writes. This happens only if more than one master attempts to start a transfer at the same time and the procedure is called arbitration. It ensures that only one master is left in control.

Due to wired AND behavior of the i2c bus it avoids contention but there are two disad-

vantages.One is wastage of current through pull up resistance ,hence it needs to be as large as possible. Secondly speed of the bus depends on resistance value Rp. Every output and input on the bus will introduce capacitance on the bus ,there will be capacitance between two wires also. When line goes to high ,current flows through Rp to charge bus capacitance Cb to VCC. Then the wave form will be an exponential curve with rise time~RpCb.

In standard mode clock frequency maximum is 100khz. So time period is 10μs . For a rise time of 1μs and bus capacitance of 100pf, pull up resistance Rp < (rise time)/(Cb) = 10k. Often the capacitance is higher and typical values of Rp are 1–10 k.When Rp is 1k current drawn is 3mA for a VCC of 3v, which is close to the limit of MSP430. So when Rp decreases rise time improves but current drawn increases.It is much faster to pull the bus down to VSS because the resistance of the n-MOSFET when it is turned on is usually much smaller than Rp.

3.3 I2C COMMUNICATION PERIPHERALS OF MSP430

Three different types of communication peripherals are offered by different families of MSP430.

3.3.1 Universal Serial Interface (USI)

USI is a light weight module included in smaller devices such as MSP430G2231 which comes with launchpad kit provided by Ti. It includes a clock generator ,shift register,bit counter and some other hardware to support i2c communication. As it does not have good inbuilt hardware support for I2C, it needs good software support. So i2c using USI will be complicated and code size also will be high .

3.3.2 Universal Serial Communication Interface(USCI)

Some of the large MSP430 micro controllers has one or more USCI module, for example MSP430G2553 which is supported by launchpad kit of Ti has one USCI module.It has good

inbuilt hardware support for I2C.Software only needs to handle interrupts and data.So the code size foe I2C communication will be less compared to USI. Each USCI contains two channels, A and B. These are largely independent but share a few registers and interrupt vectors:USCI B is the synchronous channel which support I2C.It contains a full state machine to run I²C communications in compliance with the specification from NXP (formerly Philips). This includes multiple masters and clock stretching.

3.3.3 Universal Synchronous/Asynchronous Receiver/Transmitter (USART)

The universal synchronous/asynchronous receiver/transmitter (USART) is an older module, which has been superseded by the USCI. In most cases it provides only asynchronous communication and SPI but it also handles I²C in a few devices.

3.4 I2C EXPERIMENTS

First Implementation of I2C was done using USI module of MSP430. One MSP430g2231 was used as master and another micro controller of the same type as slave. Later these micro controllers were replaced by MSP430g2553 which has USCI module and it has more I2C support hardware compared to USI module.

For the first experiment with USCI two launchpad kits with MSP430G2553 were used .One Launchpad was made to act as I2C master and the other as slave. Pull up resistance used was 10k.

3.4.1 Interrupts In Master

For read from slave receive interrupt bit UCRXIE is set in I2C interrupt enable register UCB0IE . Similarly for write to slave UCTXIE bit should be set in UCB0IE. UCNACKIE bit also can be set for not acknowledge interrupt. Whenever master receives data in UCB0RXBUF a flag is set and interrupt routine for recieve data is executed .

3.4.2 Interrupts in Slave

UCSTTIE bit and UCSSTPIE are set in I2C interrupt enable register for start and stop condition interrupts.Whenever master gives a start condition, UCSTTIFG flag is set. An interrupt routine is executed inside which slave sends data to master.

3.4.3 Single slave operation

A msp430g2553 which is acting as master, was made to receive data from another msp430g2553 slave.Using rand() function random numbers were generated in slave and will send that to master. In code composer studio, code was executed step by step ,and receive register UCB0RXBUF was observed on each reception of bytes. To verify the correctness of data transfer without step by step execution, check sum was calculated on both master and slave ,after all data is send slave will send this check sum to master. Master compares this with check sum on it's side ,if both are equal an led is turned on.

In another experiment an array of data was stored inside slave. This stored array of bytes is then send to Master. Master will store this data in a buffer inside it. After executing the code this contents of this buffer was observed in code composer studio. In this way number of errors in received bytes can be counted.

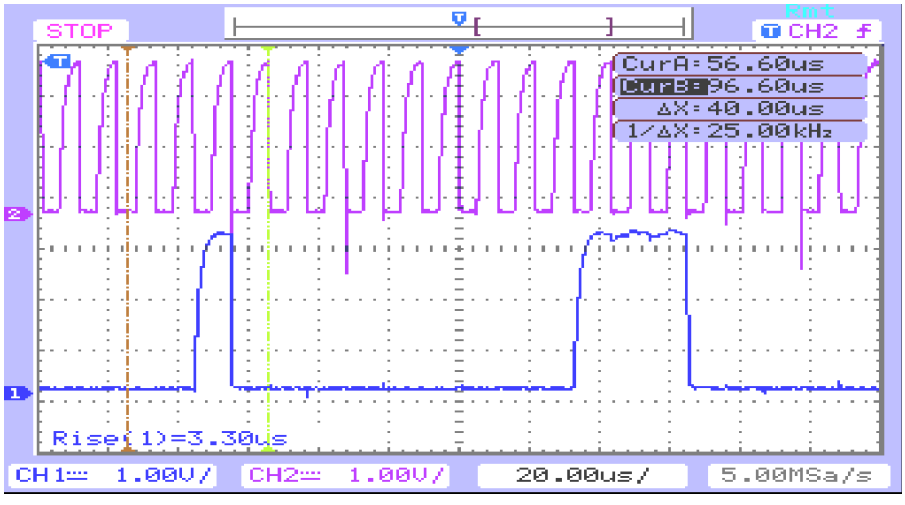

Figure 3.7: I2C communication observed on a Digital Oscilloscope,top waveform is clock bottom one is data

The transmission of data from master was also done in the same way. Correctness of data transfer was verified using check sum. Different wire lengths ranging from 10cm to 30 cm were used for communication ,in all cases error rate of less than .01% was found .

Total time taken for transfer of 16 bytes at 100khz was 1.53 ms . Before beginning the transfer 8 bit address is also send. For every byte transferred there is a overhead of one acknowledge bit .So total time for completion of on transfer of x number of bytes at I2C clock frequency f_{clk}= $\frac{((x+1)*9)}{f_{clk}}$ s.

3.4.4 Multislave operation

In multislave operation two slaves which are msp430g2553 was connected to I2C bus. Both were given different address. Master Addresses slaves one at a time ,does communication with it. Once one communication is finished Master addresses the another one. Each slave adds capacitance to the bus,so as number slaves are increased timing issues will occur.Pull up resistor is needed to be decreased if more number of slaves need to be added to the bus.

3.5 SCL rise time and Bus capacitance

As explained in 3.2 on page 20 with pull up resistance Rp and bus capacitance Cb, rise time of SCL changes.Experiments were done with different values of Rp with single msp430g2553 master and slave. For this experiment frequency of clock was set at 100khz as this is going to be the data rate for satellite main bus.

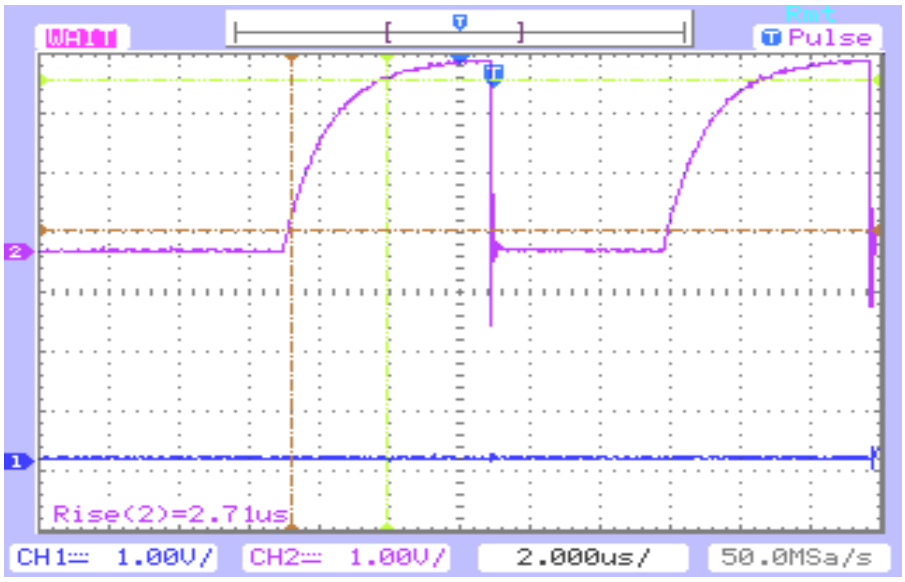

Figure 3.8: Rise time of clock with 10 k pull up and one slave, observed on a DSO

With the same slave Rp pull up resistance was varied and rise time was measured using Digital Oscilloscope. Current drawn on SCL line was also measured using Digital Multimeter.

Pull up Resistance :Rp	10k	6.2k	4.7k
Rise time :tr	2.71μs	1.75μs	1.26μs
Bus capacitance: Cb	271 pf	280pf	269pf
Current drawn :iclk	.32mA	.52mA	.702mA

Table 3.1: Rise time and current drawn on SCL pin

As shown above bus capacitance due to two MSP430's and capacitance due to wire is found to be 270pf.

Then for a pull up resistance of 6.2k different number of slaves were connected and the rise time was observed.

Number of slaves	1	2	3
Rise time :tr	1.75μs	2.01μs	2.20μs

Table 3.2: Rise time for different slaves

From table 2 it is found that each MSP430 slave adds about a capacitance of 20 pf to the Bus.

SUMMARY

This chapter deals with i2c communication with MSP430 micro controllers. Single slave and multiple slave I2C communication was successfully done using MSP430 micro controllers. USCI module of MSP430 was used for this purpose, mainly because of reduction in code size. No errors were found in any of the operation. Behavior of I2C bus for different values of pull up resistance and different number of slaves were analyzed. Depending upon number of nodes going to be connected in main bus pull up resistance need to be chosen.

Chapter 4

SD CARD INTERFACE WITH MSP430

The data from payload need to be stored in an external memory . House keeping data from OBC is also stored in this same external memory. The external memory size should have sufficient enough store all the on board data . So this memory was chosen to be SD card because it offers storage capacity of Giga Bytes .

4.1 SD card

SD card is a non volatile kind of memory storage device. They are small in size and relatively simple and less in power consumption. The Secure Digital card is available in 3 different form factors. The three form factors are the original size,mini size and micro size.

pin diagram of SD card is shown below.

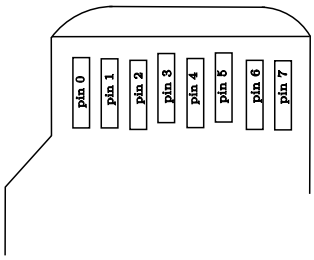

Figure 4.1: Micro-SD card pin diagram

Many SD card pins are used different way depending upon the protocol it uses for interfacing. A table is given below detailing the pin assignments.

pin	Name	Function(SD Mode)	Function(SPI Mode)
1	DAT3 /-	Data line 3	Not used
2	CD/CS	Card detect	Chip select
3	CMD/DI	Command Line	Data in
4	VDD	Supply voltage	Supply voltage
5	CLK/SCLK	Clock	Clock
6	VSS	Ground	Ground
7	DAT0/DO	Data line 0	Data Out
8	DAT1/-	Data Line 1	Not used

Table 4.1: Pin assignments of micro-SD card

There are two modes of interfacing for SD card SD mode and SPI mode.

- SD mode

The SD 1-bit protocol is a synchronous serial protocol. It uses three lines .One data line for data transfer in bulk.One clock line for synchronisation,And one command line over which command frames are send. SD 1 bit protocol supports bus sharing .SD 4 bit mode is slightly different from SD 1 bit . In SD 4 bit mode it uses four wires for to transfer data parallely. So the speed of data transfer it can achieve will be quadruple of SD 1 bit protocol. For bulk data transfers a crc protection is required in SD protocol.

- SPI mode

This mode is different from SD mode ,it works with the generic SPI protocol which is largely used by many micro controller peripherals. MSP430 also supports SPI communication which has dedicated peripherals to implement SPI. Compared to SD mode SPI speed will be less .

Interface with Micro- sd card was done using SPI as it is simpler to implement compared to SD mode and well supported by MSP430.

4.2 SPI Protocol

SPI interface was introduced by Motorola . It does not have fixed standard ,it has different variations. There should be at least two devices. One will be master and the other slave. Master provides clock .If there are more than one slave devices , each slave is selected through chip select CS. SPI is full duplex as it can transfer data in both directions simultaneously between both devices. The two data lines are Master in slave out (MISO) and Master out Slave in (MOSI).

Each device has one shift register. They are connected together to form a loop. These registers are usually 8 bit long. On positive edge of clock ,each device output their msb bit on the bus. On negative edge each device will read data on bus and store it in the lsb of shift register. So in eight clock cycles new data will be transferred between devices. In order to receive it is necessary to send something.

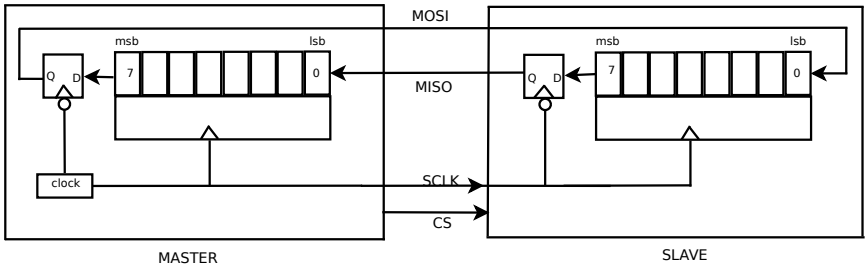

Figure 4.2: SPI interface between master and slave

Multislaves are possible in SPI communication . There are two methods to do that. In first method as shown in the figure below ,Different chip select is provided for each slave.

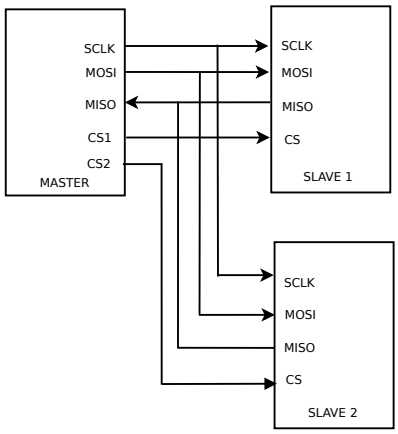

Figure 4.3: SPI multislaves

All the MOSI pins are connected together, as are the MISO pins. Slaves must ignore data on MOSI when their CS pin is idle, even if clock signal is there, and must leave their MISO pins in a high-impedance state .

The next method is to connect them in a daisy chain. In this method MISO pin of a slave is connected to MOSI pin of another slave.MOSI of the final slave is connected to MISO of master.Effectively all the shift registers inside the device forms a loop. So Master shift register should contain total number of bits in shift registers in all the slaves. But this is usually done in software.

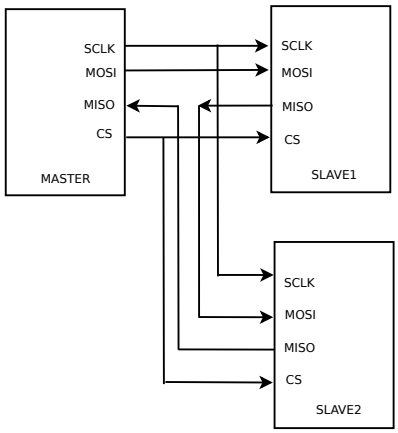

Figure 4.4: SPI Multislaves in daisy chain

4.3 Micro-SD card SPI interface

SPI interface connections of Micro-SD card with MSP430 is shown below. MSP430G2553 was used which is supported by the launchpad kit. It has one USCI module ,which has two channels USCIA and USCIB. Both can be used for spi communication. But USCIB was used for I2C communication,USCIA module was used for memory interface. A GPIO pin of MSP430 was used as chip select pin. Supply voltage of Micro SD was given as 3.3 volts.

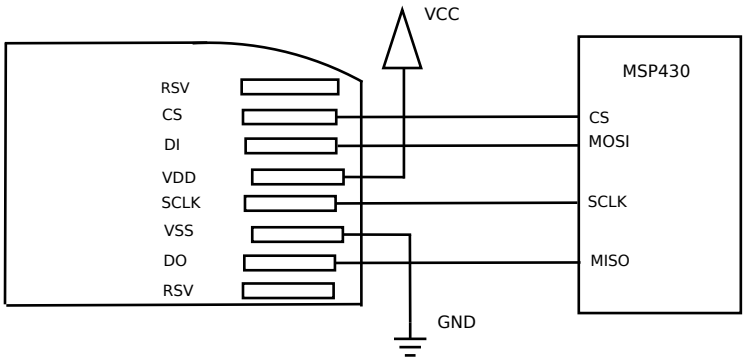

Figure 4.5: SPI interface with MicroSD card

4.4 File System

General file system used in SD card is FAT file system. There are many open source implementation of FAT file system.

- EFSL
- Net labs FAT32
- FATFS and Petite FATFS [4]

RAM memory of msp430g2553 is 256 bytes so it cannot support a file system which consumes lot of RAM space. So petit file system was chosen.

4.4.1 Petit file system

The features of Petit file system is given below

- Small RAM consumption.
- Supports FAT 32
- File write has some restrictions

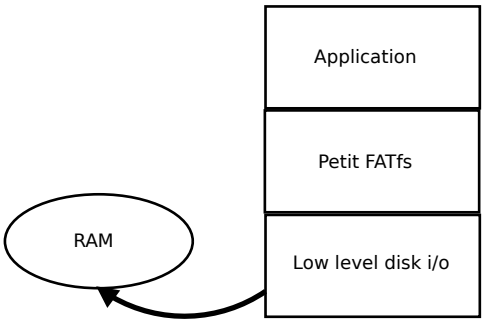

Figure 4.6: Petit File system modular structure

Petit file system application interface provides following functionality to the user

- Pf_mount :mount a volume in Micro-SD
- pf_open :Open a File
- pf_read :Read a File
- pf_write : Write File
- pf_lseek : Move read/write Pointer
- pf_opendir :Open a Directory
- pf_readdir :Read a Directory Item

Petit FatFs is completely separated from disk I/O layer. To access the disk it needs some functions in the lower layer. Depending upon the device this functions varies . This should be provided by the user.Those functions are

- disk_initialize :Initialize disk drive
- disk_readp : Read partial sector
- disk_writep : Write partial sector

4.4.2 Experimental Setup

In order to connect Micro sd with launchpad a breakout board was used. It can hold micro sd and can provide pin connections for SPI interface. A uart connection with computer was made through which commands were given to micro controller. Termite 2.6 was used as hyper terminal. Baud rate was set at 9.6 kbps.

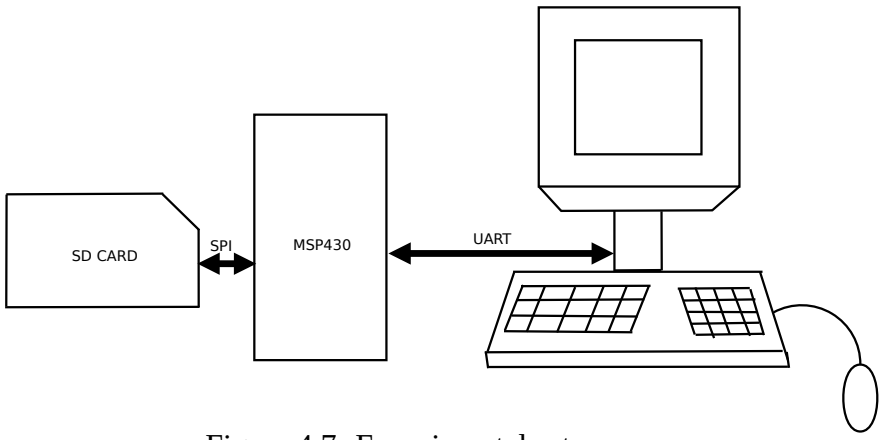

Figure 4.7: Experimental setup

After power up Micro sd card need to be initialized. This involves issuing certain commands to sd card which check for the voltage compatibility,version etc of SD card. Next step is initialization of file system. Pf-mount is called for doing file system initialize. For reading data from a file ,that file need to be opened first ,Pf_open function of petit

file system will take care of this. When a file is opened data in that file is transferred to micro controller through pf_read function. A buffer of certain length was set inside micro controller. The received data is stored inside this buffer and send to computer over uart for display, in this way it is possible to verify whether data read from Micro-sd is correct or not.

RESULTS

After successfully reading data from Micro-SD ,profiling of read and write speed was done. Time required for disk initialize and file system initialize was also measured. A DSO was used for this purpose. Probes were connected to clock line and data line of SPI interface. Burst width was measured using DSO.

File size(bytes)	File open time(ms)	File dump time(ms)
1	145.33	2.44
2	140.44	2.88
4	134.88	2.55
8	92.77	2.61
16	107.2	2.61
32	113.55	2.44
64	118.66	2.44
128	124	2.44
1KB	98.81	20.2
10KB	77.81	211.11
100KB	85.61	2001.12

Table 4.2: Read profiling of Micro sd Card with SPI clock 4 mhz

As shown in the table file open time was almost constant and took about 100 ms. So file read operation take s approximately 50 kbytes/s approximately. Disk initialize took 20 ms to complete. File system initialize took 39 ms. Write speed also was measured in the same way as read and found to be approximately 35 kbytes /s.

To improve the transfer rates clock speed of SPI was increased to 8mhz. Same profiling was done but it was found that read speed was improved to 60Kbytes/s only. The idle time

between each 8 bit transfer remains the same in both the cases . At 16 Mhz clock Micro SD stopped responding.

High overhead time of file open and low data rates showed that it is not feasible to implement file system for our application. Same results were obtained for ARM board interface with Micro sd which is used for PAYLOAD by IITMSAT team. So it was decided to remove file system and use SD card in its raw form.

4.5 Micro SD interface in Raw format

In raw format read and write is performed in blocks. Block length is 512 bytes. Before each action command is issued to SD card which specifies action to be performed by SD card. If command was received successfully it returns a token as response. Then required action can be performed.

4.5.1 Block Read

In Read Block function argument it specifies the location of block. Command number needed to be send is 17. After this command is accepted a read operation is initialized. If a valid data token is received by the micro controller ,then data bytes will be transferred to micro controller. A two bytes of CRC data is also received by micro controller.

4.5.2 Block Write

Write command is issued to SD card for the start of Block Write. Command number is 24. If a valid token is received by the micro controller as response from SD card ,it will immediately start sending the data bytes. When a data packet has been sent, the card responds a Data Response immediately following the data packet. After the data response

a busy flag is set by the Micro sd card in order to do the processing of received data. During this time chip select of SD card can be deasserted such that other slave devices in the bus can be enabled.

Results

After Reading data from Micro SD card it is stored in a buffer inside micro controller. The buffer size was set to be 512 bytes. Contends of this buffer was observed after read operation . No errors were found. To verify write operation, the written sector was read back to microcontroller and viewed. Total time taken for read of 512 bytes block, and write of 512 bytes were measured.

- SPI clock 4mhz

  Read time for 512 byte block - 2.42 ms

  Write time for 512 byte block - 3.44 ms

- SPI clock 8 mhz

  Read time for 512 byte block - 1.2ms

  Write time for 512 byte block - 1.78ms

4.5.3 DMA for data transfer

In DMA operation data in one resource is transferred to another resource without CPU intervention. This can help reducing power consumption. CPU can also process any interrupt unrelated to memory DMA is using ,during this time. MSP430g2553 does not have DMA. MSP430F5436 has 3 DMA channels. With this micro controller for SD interface interface DMA was utilized.

For block read operation two DMA channels are required. To receive anything in recieve buffer in msp430 it need to send something. So for one DMA channel the source address was given a variable which holds value 0XFF,and destination address as spi transmit buffer. And source and destination address remains same for the whole transfer. Block length was set to be 512 bytes. For the another channel source address was SPI recieve buffer of MSP430 and a destination address was given as start address of buffer in which read data bytes are stored. Destination address will get incremented after each byte transfer.

For block write only one DMA channel was used. Source address is start address of buffer in which data to be transferred is stored. Destination address is spi transmit buffer. Destination address is incremented after each byte transfer.

4.6 Memory sharing

This SD card is shared between PAYLOAD micro controller and COM micro controller in topology 5. Both Micro controllers are using SPI interface for data transfer. If both micro controllers try to access SD card at the same time it may cause conflicts in the bus . It may result in failure of SD card which is not desirable.

This SD card is shared between PAYLOAD micro controller and COM micro controller. Both Micro controllers are using SPI interface for data transfer. If both micro controllers try to access SD card at the same time it may cause conflicts in the bus . It may result in failure of SD card which is not desirable.

So some hardware arrangements were needed to make sure that both micro controllers will not access memory card at the same time. When one micro controller is accessing the memory card , in order to avoid conflicts other micro controller SPI pins should be put in tri state condition.3 State octal buffer 74hc541 was used for the purpose.It has input capacitance of 3.5 pf and has a delay of 10 ns.

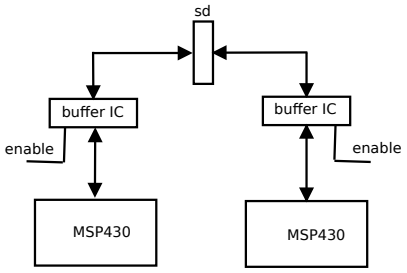

Figure 4.8: Memory sharing between two MSP430s

As shown in figure, this arrangements was tested between two MSP430s sharing a common SD card. The 74hc541 is enabled if micro controller connected to it want to access SD card. The other buffer is disabled so that the other micro controller is disconnected from SD card. In this way Memory sharing was success fully tested.

SUMMARY

This chapter discusses Micro SD interface with MSP430. It was found that using file system data rates are not satisfactory for our application. With SD card in raw format data rate of up to 500 kbytes/s was achieved. DMA controller of MSP430 was utilized for Micro SD interface as it helps in power reduction. Suitable Micro-SD card need to be chosen which is industrial grade, as normal SD cards may fail in space conditions.

Chapter 5

I2C BUS NODE ARCHITECTURE

All the health data communication is done through I2C bus, and controll of subsystems based on the commands from Ground station is also done through main bus. As explained earlier main bus is using i2c protocol. Even though I2C provides reliable way of communication, for space applications extra reliability need to be added. So extra Hardware is added to main bus to make it more reliable[3].

Extra hardware should be implemented in such a way that even if any of the connected nodes failes total bus should not fail. It should be single point failure free.

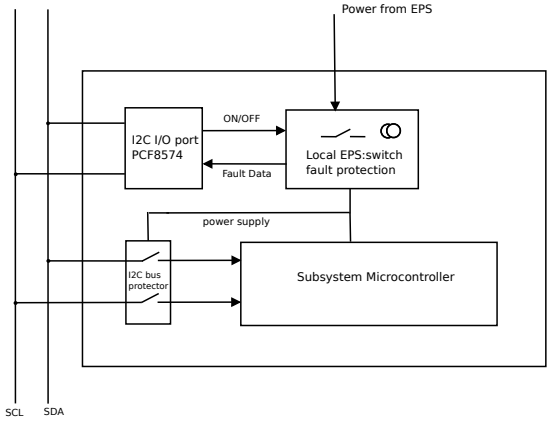

Figure 5.1: Bus node architecture

As shown in figure for each sub system there is a local eps switch which is controlled through an i/o port, and a protection circuit.

5.1 I2C Protection Circuit

For synchronisation , slaves such as micro controllers are allowed to pull the clock line low till some internal operations within slave are completed. Once this line is released then only master can control the clock line . In case of some erroneous events occurring inside slave which causes clock line to be pulled low , it can result in global failure of the bus. Protector circuit is implemented to prevent this kind of global bus failures[3].

Protector circuit is basically a timer which monitors both SDA and SCL lines. If any of the lines connected to any of the nodes stays low for a predetermined time it electrically removes the subsystem which is causing problems, from bus .

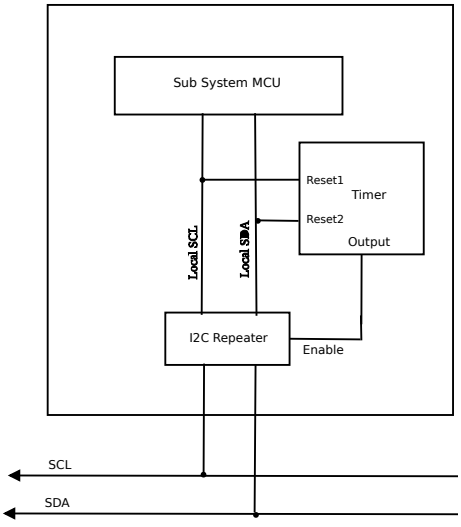

Figure 5.2: I2C protector block diagram

Timer is implemented through RC circuits. P82B96 chip which is a dual bi directional buffer . This adds a minimal loading to the Bus. So number of nodes that can be connected to the bus increases.

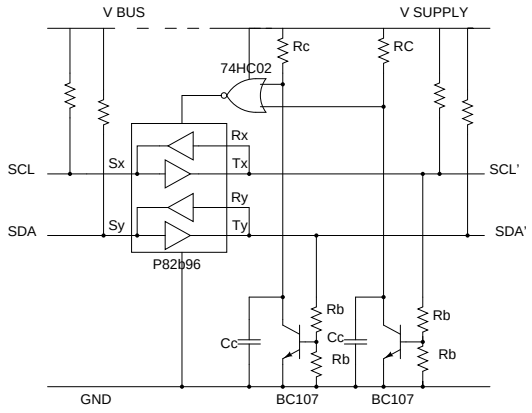

Figure 5.3: I2C protector implementation

As shown in figure if any of the the line goes low the capacitor Cc will begin to charge to Vcc through Rc since the transistor is turned off. If the voltage across capacitor goes above VIH of the NOR gate 74HC02, its output will become zero. This is given as power supply of buffer ic. Since Buffer's power supply is cut , eventually the subsystem which pulled downed the bus line is removed from master side of the bus. Once erroneous slave device works again the bus lines on its side will go high and capacitors will get discharged through the ON transistor. NOR gate output will go to logic one. This results in enabling the buffer which in turn connects the sub system to main bus.

5.1.1 Protection circuit Testing

In order to understand the need of protection circuit ,Multislaves i2c communication was done without protection circuit. Two MSP430 slaves were connected to the bus. Another MSP430 was made master. In one slave, inside the start condition interrupt a while loop was inserted. If this slave is addressed by master , start interrupt routine of this slave will be executed. So it will pull down the clock line low,as it will be inside while(1) loop.

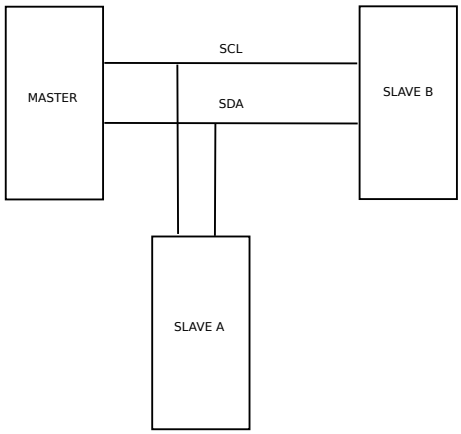

Figure 5.4: Test without protection circuit

Slave A and B were made to be addressed by the master at regular intervals of time. Slave B will start executing infinite loop once addressed by the master. After addressing slave B, it was found that master was not able to address slave A . So without protection circuits single node failure can result in global hanging of the bus.

In the next test After this protection circuit was implemented for slave B. Timer inside protection circuit was set at 1ms by setting proper value of Rc and Cc.

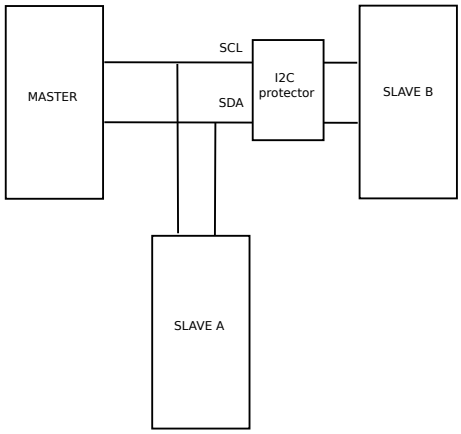

Figure 5.5: Test 2 with I2C protector

When clock line was pulled down by slave B protection circuit removed slave B from Bus , so master was able to address slave A and data transmission was success full between the two.

In another test certain delay of period less than timer value of protector,was set in slaveB. This was to make sure that protector circuit does not causes any problems to normal i2c communication. In this case it was observed that during the delay period, capacitor Cc begins to charge to Vcc,but as set delay is such a way that before it reaches VIH of nor gate , clock line is released.Capacitor discharges to zero. So I2C buffer will be on all the time, communication was un interrupted and no erroneous behavior was observed.

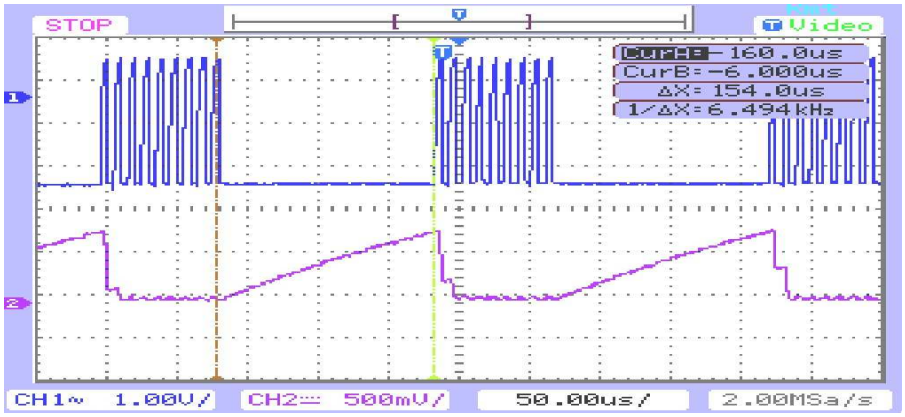

Figure 5.6: Capacitor charging in I2C protector

As shown in figure when clock line is pulled low during the delay period capacitor Cc charges, but once clock line is released it is discharged to zero.

5.2 I2C input/output Port

I2C i/o port is used to controll local EPS in each subsystem. PCF8574 which is a common I2C expander ,was chosen to serve this purpose. It has three address pins ,so total of eight such devices can be connected to main bus.

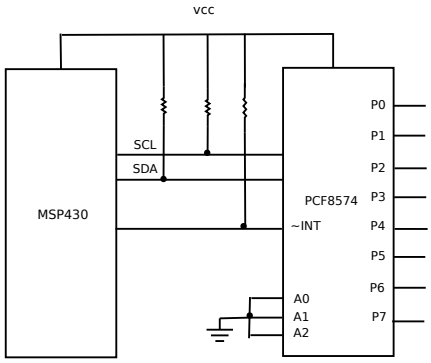

Figure 5.7: I2C i/o port typical connection with MSP430

PCF8574 has low stand by current consumption of 10 μA ,and can work at 100khz I2C clock frequency. The ports P0 to P7 of PCF8574 can be directly controlled by sending I2C commands. In typical applications these pins can be connected to temperature sensors,push buttons,leds etc. There is an Interrupt line ~INT which can be connected to interrupt logic of the micro controller. Whenever there is an incoming data in any of the pins which are set to be inputs, it interrupts the micro controller without having communicate through I2C. INT line goes low when there is an high to low or low to high transition in any of the port pin. This goes to high again soon after this port is read through I2C by master micro controller.

Fault data input from Local EPS is given to one of the inputs of PCF8574. On/Off control of local eps is given to another pin of PCF8574 which set to be output. When a fault data is given by the local EPS switch ,PCF8574 interrupts OBC micro controller. OBC will command PCF8574 to switch off local EPS switch through I2C. So whichever sub system drains more power than limits of EPS can be immediately shut down.

5.3 Local EPS switch

Local eps switch present in each subsystem distribute power from EPS to subsystem micro controller and protection circuit of that micro controller. If fault happens in that subsystem ,local eps should report it to OBC micro controller. TP202X family power distribution

switches from Ti were chosen for this purpose. These devices are N-channel mosfet power switches.

When the output load exceeds the current-limit threshold or a short is present in connected subsystem, the TPS202x limits the output current to a safe level by switching into a constant-current mode. It makes the over current (OC) logic output low. This output is connected to PCF8574 which interrupts OBC micro controller. Then OBC can turn off the switch by giving logic zero to the enable pin of TPS202X. When the over current condition is resolved it automatically makes the OC out put high. Then OBC can turn ON the switch .

When continuous heavy overloads and short circuits increase the power dissipation in the switch the junction temperature of the switch rises, a thermal protection circuit shuts off the switch to prevent damage. It is then automatically recovered from thermal shut down after temperature has gone down sufficiently.

Summary

The need of I2C protection circuit was justified. It was found that protection circuit can successfully prevent global hanging of the bus due to single node faults. The timer setting for each subsystem need to be analyzed. The I/O port PCF8574 was tested. I2C communication with this device was successfully established. Depending upon current consumption of each subsystem suitable device of TPS202X family need to be chosen. Testing of these switches need to be done.

Chapter 6

BACKUP FUNCTIONS OF OBC

OBC collects House Keeping data from other subsystem at regular intervals of time ,which is not frequent. Rest of the time it is put in idle state. So OBC can be made to act as back up for any of the subsystem. PAYLOAD and ADCS are using ARM micro controllers , it is difficult for OBC to do their functions . As drivers and hardware interfaces Will be different for them. COM subsystem which is responsible for managing communication between GS and IITMSAT is using MSP430 micro controller.

COM is an important subsystem, if COM fails, control over satellite will be lost from GS. So it was decided to make OBC to act as back up for COM. If COM can manage to do OBC functions OBC can be put in idle state. OBC then act as a redundant system for COM. If COM fails it comes to active state. Testing was needed to check whether COM could do House Keeping also while it is in most active stage ie when satellite is over GS.

The level of redundancy needed to be implemented was also considered. A simple method would be replicating the COM subsystem and sharing hardware resources between COM and OBC.

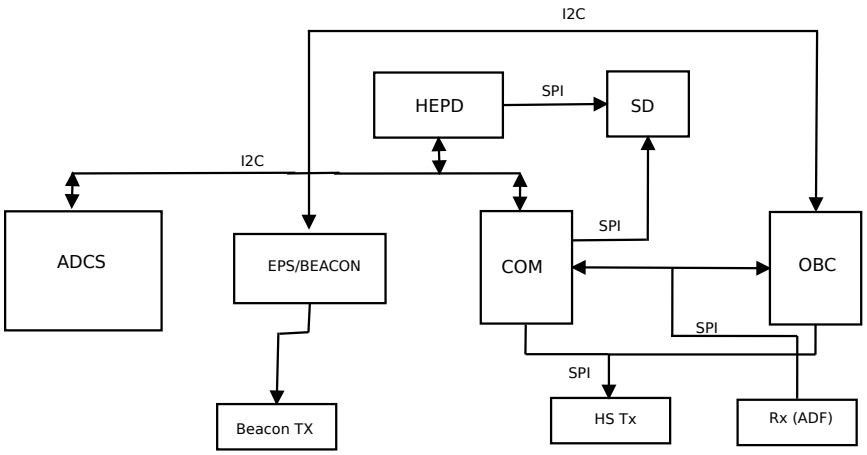

Figure 6.1: OBC as back up for COM

As shown in figure Transmit and receive Hardware of COM can be connected to OBC. During normal operations as OBC is put in sleep mode so it does not affect COM functionality.

To test the OBC for COM functions some level of understanding about COM was needed.

6.1 COM subsystem

COM micro controller which is MSP430 is connected to a receiver hardware and transmiter hardware.

6.1.1 FSK Receiver (ADF7020-1)

ADF7020-1 is a FSK/ASK transceiver manufactured by Analog Devices. When satellite is over GS the receiver will receiving data from GS. As soon as SYNC word which is stored inside a register of ADF is received it interrupts MSP430 micro controller. ADF7020 uses SPI interface to transfer received bits to MSP430. After the Synch interrupt MSP430 receives data from ADF7020 for 9 clock cycles. SPI frequency is 1.2 kHz.

6.1.2 GMSK base band modulator (CMX7164)

CMX7164 is using c-bus protocol which is same as SPI. CMX 7164 has two internal buffers CMD FIFO and command buffer. From MSP430 15 bytes of data is send to CMX7164. An interrupt is generated when CMD FIFO is empty. Down link transmission is occurring at 19.2 kbps. So interrupts will occur in $\frac{15 \times 8}{19200}$ = 6.25 ms. At 2 Mhz speed of SPI, MSP430 takes $\frac{17 \times 8}{2 \times 10^6}$ = 0.68 μs to fill the buffer.

6.1.3 Test Setup

The MSP430 which is to be used for OBC now needs 3 spi (one for ADF ,one for CMX7164 and one for memory card) and one i2c peripherals. So 4 USCI channels are needed. MSP430f5436 has four USCI modules,so it was chosen.

To simulate ADF7020 and CMX7164 two launch pad kits with MSP430g2553 micro controllers were used. They were connected to MSP430F5436 through SPI interface. Memory card was also interfaced through SPI. Another Launchpad was connected to MSP430F5436 through I2C.

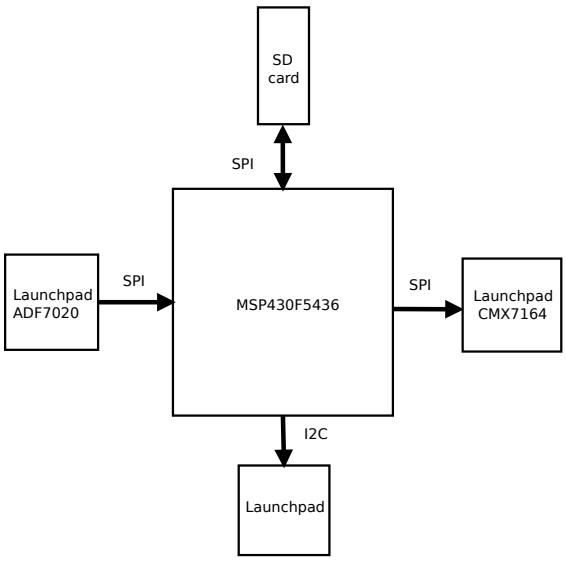

Figure 6.2: Test setup for OBC as back up

As receiver has higher priority it was connected to USCIA2 which has higher priority.

USCIA1 which has next higher priority was connected to Transmitor launch pad. For I2C communication USCI B3 was used.

As from Tx it will receive interrupt at intervals of time 6.25 ms from function generator a square wave of period 6.25 ms was given to port 2.0 of MSP430. And port 2 interrupts were enabled. So MSP430 will get interrupted in every 6.25 ms . Then it will send 15 bytes of data to Tx launchpad.

As memory read and write is done through DMA it won't be affected by various interrupts. I2C communication done in between this interrupts.

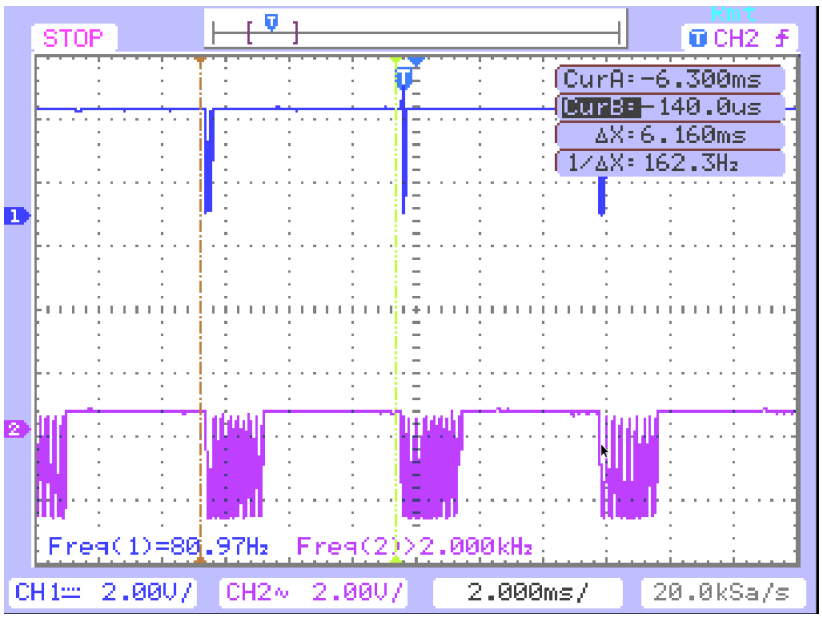

Figure 6.3: Figure showing SPI communication at intervals of 6.25 ms and i2c communication of 16 bytes in between it.

For I2C communication of 16 bytes it takes about 1.5 ms to complete. So it is possible for the COM to do the House keeping also as it has lot of free time in between each 6.25 ms.

6.2 Flow diagram for COM with CDMS function

COM need to have two internal buffers read buffer and HK buffer. The data to be transfered to CMX7164 is stored inside read buffer. House keeping data to be stored in SD card is

saved inside HK buffer. Both buffers are of 512 byte size because memory can be accessed only in blocks of 512 bytes.

As SD card is accessed by payload also, there need to be synchronisation in between them. When PAYLOAD writing data to SD card it gives a signal ,memenp to COM indicating COM to hold its memory transfers. Similiarly if COM is accessing SD card it gives a signal,memenc to PAYLOAD indicating PAYLOAD to hold. House Keeping can be done at regular intervals of time by keeping a timer.

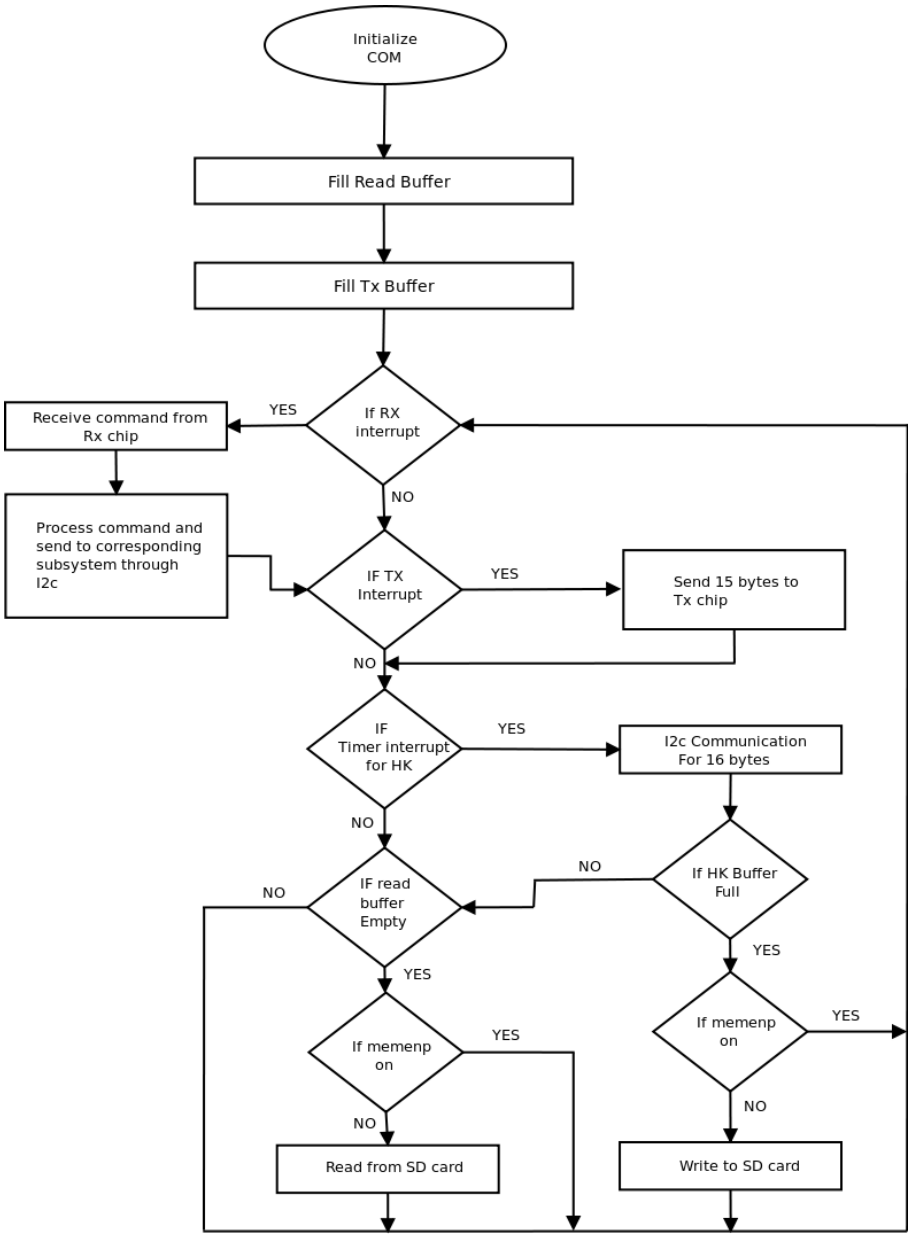

Figure 6.4: Suggested flow chart for COM

Both buffers read buffer and HK buffer are two in number. While one read buffer is getting filled from memory card other can be used for sending to Tx chip. Similiarly when one HK buffer is full ,and while it is getting written to SD card the House keeping data can be stored in other HK buffer.

6.3 Disadvantages in this topology approach

In this topology back up is provided for communication micro controller. Any of the receive or transmit hardware is also prone to failures. If Rx chip fails even if Back up of COM is there the controll of satellite over GS will be lost. Similiarly if Tx fails science data cannot be transferred to GS. But it is not of high importance. The priority is for Rx. Similiarly as we are connecting same hard ware to the OBC ,it is possible that the reason which caused COM to fail will cause problems for OBC also.

The memory sharing is done through buffers. It is addition of extra hardware in the system. If any of the buffer fails or synchronisation between PAYLOAD and COM is lost it will result in damaging the memory card. And the buffers used are not proven in space conditions.

6.4 Backup in terms of Rx Hardware

Instead of giving back up for COM micro controller it was decided to provide a extra Rx hardware as back up. In order to reduce number of micro controllers CDMS was given EPS functions also.

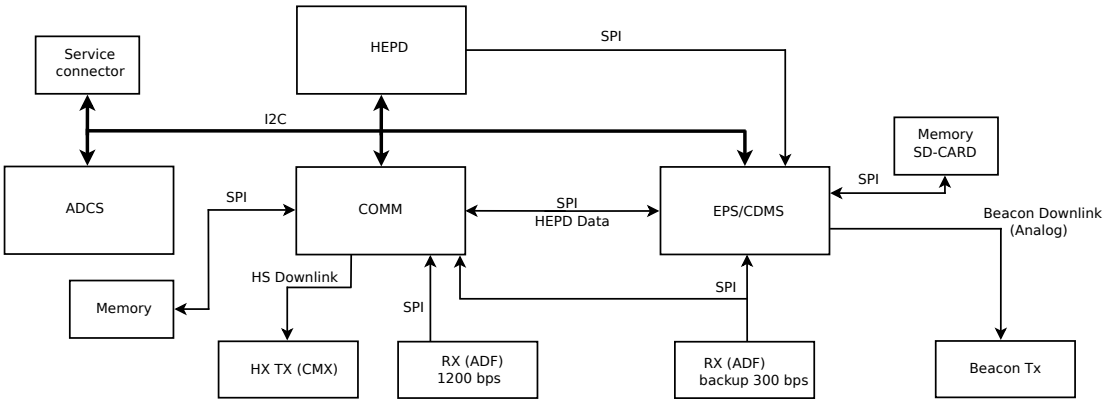

Figure 6.5: Topology with back up RX ADF

In this topology storing of both SC and HK data is done by CDMS. PAYLOAD will collect SC data and send to CDMS in every .1ms. The interface used is SPI and this bus can be called SC1 bus. CDMS will keep a buffer of 512 bytes and store it to SD card attached to it. DMA can be utilized for this purpose. And when COM needs data on which framing needs to be applied , CDMS read data from memory and sends it to COM. This interface with COM and CDMS is also SPI,and this bus is called SC2 bus. COM stores data from OBC in a external memory attached to it. For SPI interface between COM and OBC ,COM is the master. So whenever it needs data from CDMS,it can initiate data transfers. For the SPI bus between OBC and PAYLOAD,PAYLOAD is decided to be the master.

RX ADF (backup) interface is also SPI. It is the master for the communication. This back up RX can receive from GS at a speed of 300 bits/s. So if COM micro controller fails RX ADF (back up) attached to OBC can be turned on and can receive commands from GS.So some level of control of satellite can be maintained. If RX ADF fails ,as COM also is connected to RX ADF(backup) it can receive commands through it.

There will be interrupts from various subsystems and hardware components for the OBC micro controller. So the priority of them needs to assigned properly. There is four SPI connection and one I2C connection for OBC micro controller. As maximum number of USCI module available for MSP430 micro controller is four, same module need to be used for two buses. Data transfer to SD card is done through DMA and it does-not have

any interrupts. And I2C communication is done intervals of few seconds for HK data. So same USCI module can be used for them. USCI A for memory and USCI B for the SD card. Even though USCIA and USCIB interrupts are shared ,there wont be any interrupts from USCIA channel as it is used for SD card.

In MSP430 different USCI module has their priority order. So higher priority one is connected to RX ADF (back up). Next priority is given for PAYLOAD SPI interface. COM SPI has the next priority. And I2C bus is given least priority. There are two GP IO interrupts for the OBC. Whenever a SYNCH word is detected by RX ADF (backup) it gives interrupt to the OBC micro controller. This GPIO interrupt is given higher priority. And When COM receives a valid command frame which need to be given to any of the sub system it interrupts OBC micro controller. OBC then initiates an I2C communication with COM receives this command. OBC addresses the subsystem for which the command need to be issued and transfers it through I2C.

Summary

This chapter discusses about back up that can be provided to IITMSAT system. Various methods were discussed and back up functionality that can be provided by OBC sub system was analyzed. A new topology was chosen and its testing need to be done.

Chapter 7

Conclusion And Future work

7.1 Conclusion

Different topologies connecting all subsystem, were tested and a topology was chosen for efficiently store and transfer SC and HK data. Suitable bus protocols, micro controller ,external memory and other hardware components has been selected. Communication of CDMS with different subsystem has been established. Interface of external memory card with CDMS was implemented and suitable drivers were developed. Reliability of the bus was improved by providing extra hardware at each subsystem nodes ,and protecting the bus from single node failures. Analysis of level of back up that can be provided by the CDMS was also done.

7.2 Future Work

The chosen topology need to be implemented in a PCB and tested. In the new topology CDMS is doing EPS functions also. So testing need to be done to verify the feasibility of this implementation. Different sensor modules such as temperature sensor ,current sensor and voltage sensor need to be interfaced with CDMS subsystem. Future work also includes fault tree analysis of developed system.

Appendix A

I2C communication Register Settings

A.0.1 Master code USCI register settings

- Control register 0 ,UCBOCTL0 :

Master bit should be set in this register , sync bit for synchronisation. I2C mode need to be set in this register .

UCB0CTL0 = UCMST + UCMODE_3 + UCSYNC; // I2C Master, synchronous mode

- Control register 1,UCB0CTL1:

Clock for I2C can be choosen from different sources by changing UCSSEL bit. For this experiment SMCLK was chosen.UCSWRST should be held high untill the settings of module is completed.Default SMCLK will be 1 mhz ,this was set at 8mhz.

UCB0CTL1 = UCSSEL_2 + UCSWRST; // Use SMCLK, keep SW reset

- Baud rate control register, UCB0BR0 and UCB0BR1:

The 16-bit value of UCBRx in registers UCBxBR1 and UCBxBR0 is the division factor of the USCI clock source, BRCLK. The maximum bit clock that can be used in single master mode is fBRCLK/4. The 16-bit value of (UCxxBR0 + UCxxBR1 × 256) forms the prescaler value UCBRx.

UCB0BR0 = 0x48; // fSCL = SMCLK/70 = ~100kHz

UCB0BR1 = 0;

- Own address register, UCB0I2COA:

For master there is no need to set this as this will be zero by default.

- Slave Address register,UCB0I2CSA:

Address of the slave need to be stored in this register.

UCB1I2CSA = 0x46;

A.0.2 Slave Code USCI register settings

- Control register 0 ,UCBOCTL0 :

No need to set Master bit in UCB0CTL0 for slave

UCB0CTL0 = UCMODE_3 + UCSYNC; // I2C Slave, synchronous mode

- Control register 1,UCB0CTL1:

Keep the module in reset untill all settings are done

UCB0CTL1 |= UCSWRST;

- Own address register, UCB0I2COA:

The address of the slave is kept in this register

UCB0I2COA = 0x48; // Own Address is 048h

Appendix B

SD CARD INTERFACE

The read and write speeds of SD card with petite files , were measured using DSO ,with probes connected to clock line and data line. Burst width was measured to calculate the data rates for different file sizes.

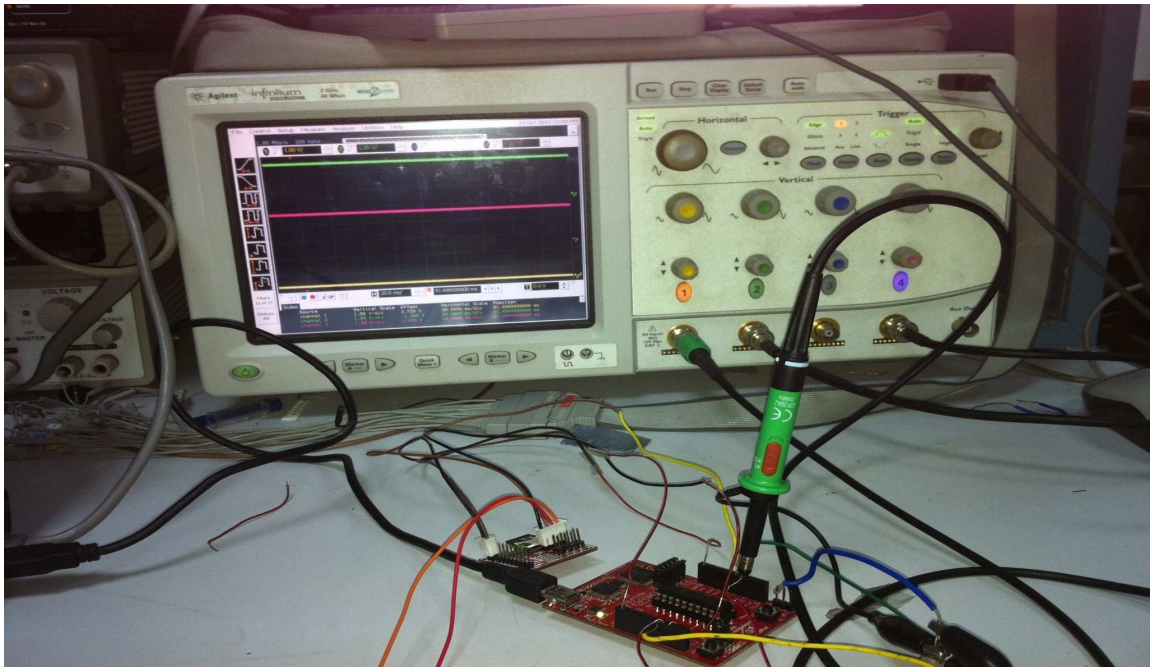

Figure B.1: Test Setup to measure Data rates

To measure read and write speeds of Micro SD in raw format, a GPIO was turned ON and OFF in between each block read and block write. The half period of the GPIO output will give the read and write speeds. The below code is run for 100 times in a loop.

P1OUT |= BIT0; //turn on pin 0

SD_readSingleBlock(0);// read block 0

P1OUT &= ~BIT0;//turn off pin 0

SD_readSingleBlock(0);//read block 0

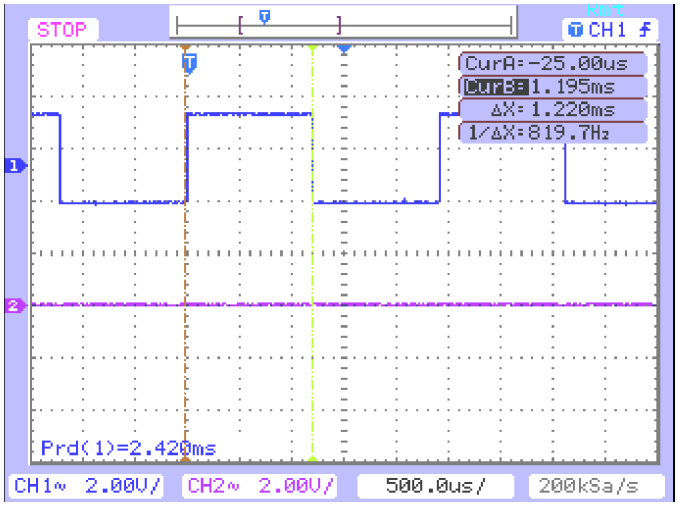

Figure B.2: GPIO output for read at 8 MHZ SPI clock

Read time for one block = 1.2ms

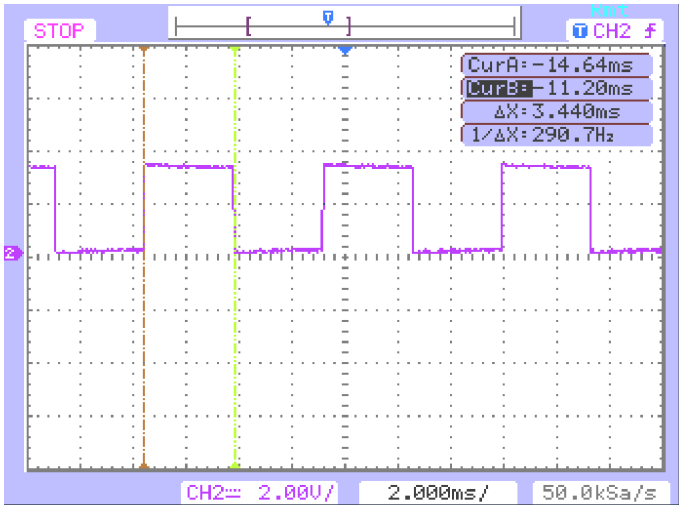

Figure B.3: Write time GPIO output at SPI clock 4Mhz

Write time for one block= 3.44ms

Appendix C

PCB DESIGN

C.1 Board 1

A PCB was designed for testing I2C communication and SD card interface. Protection circuit and I2C i/o port also was included. MSP430G2553 micro controllers were used as Master and Slave.Debug interface used was SPi BY ware.

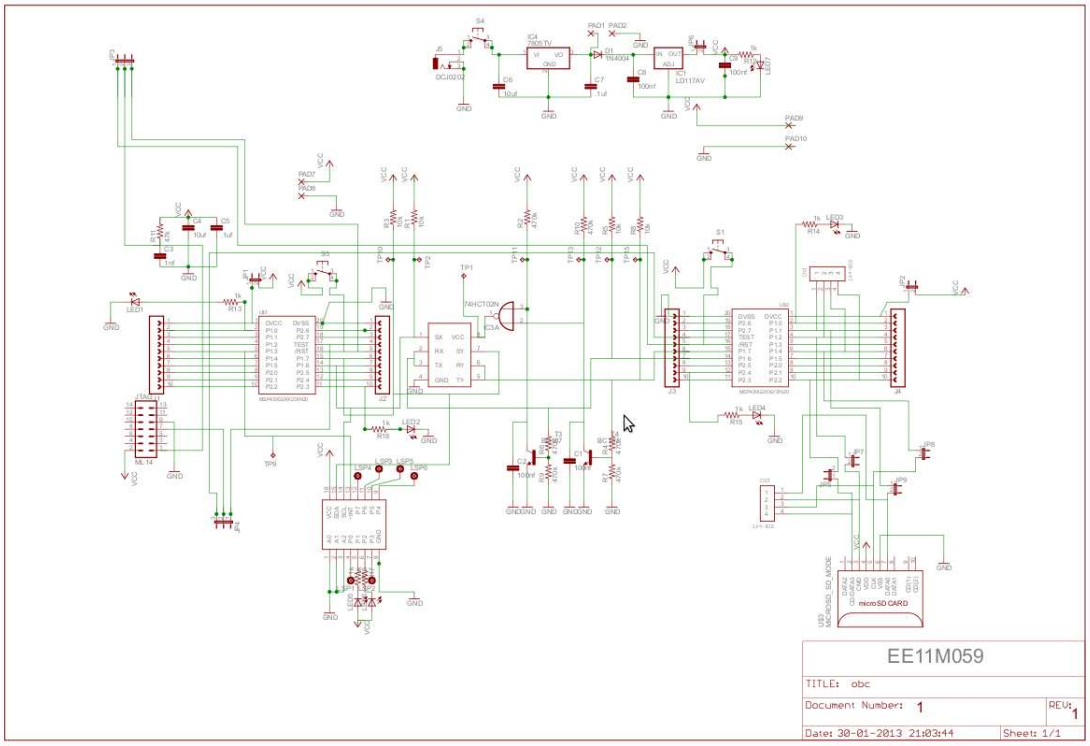

Figure C.1: PCB schematic for board 1

C.2 Board 2

Second PCB was designed with MSP430F5436 for testing the backup functionality of OBC. Debug interface used was J-TAG.

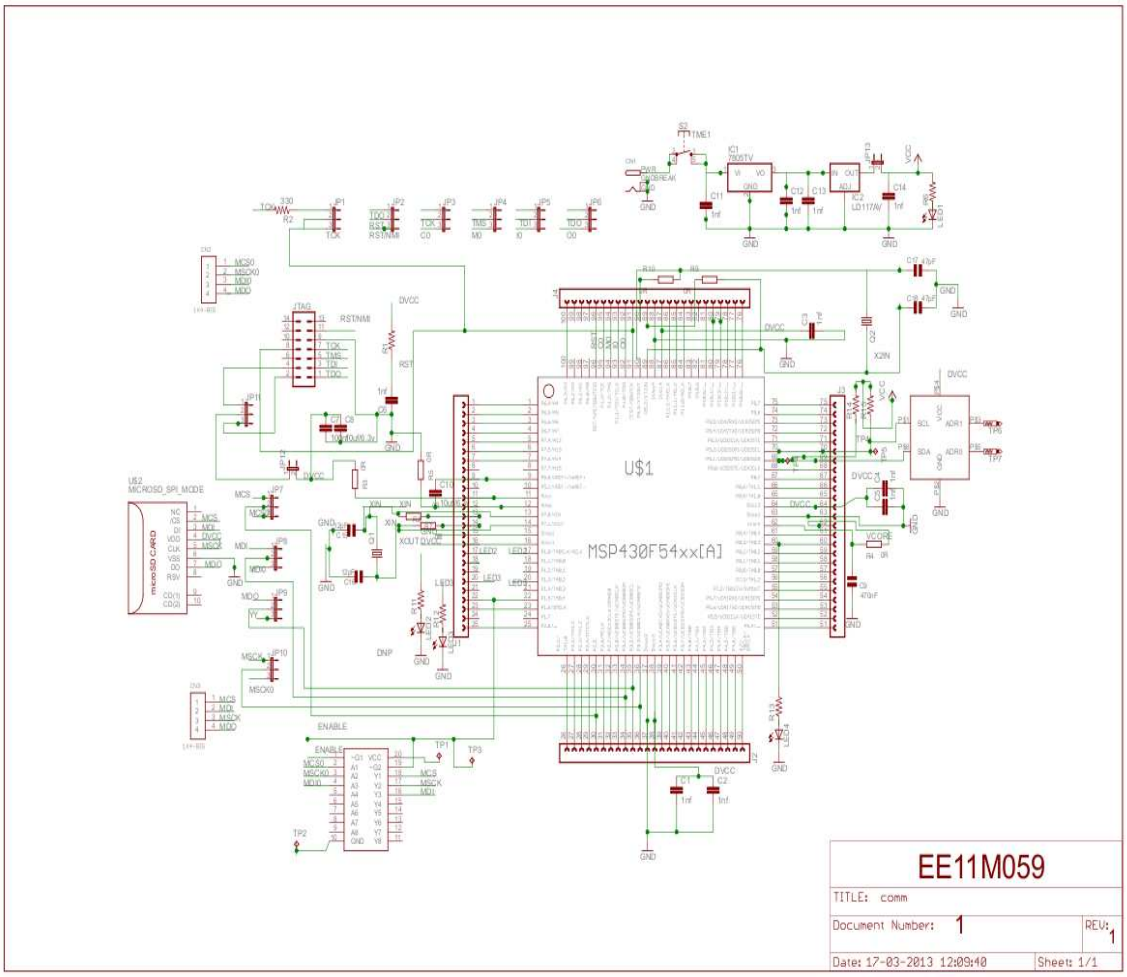

Figure C.2: Board 2 schematic

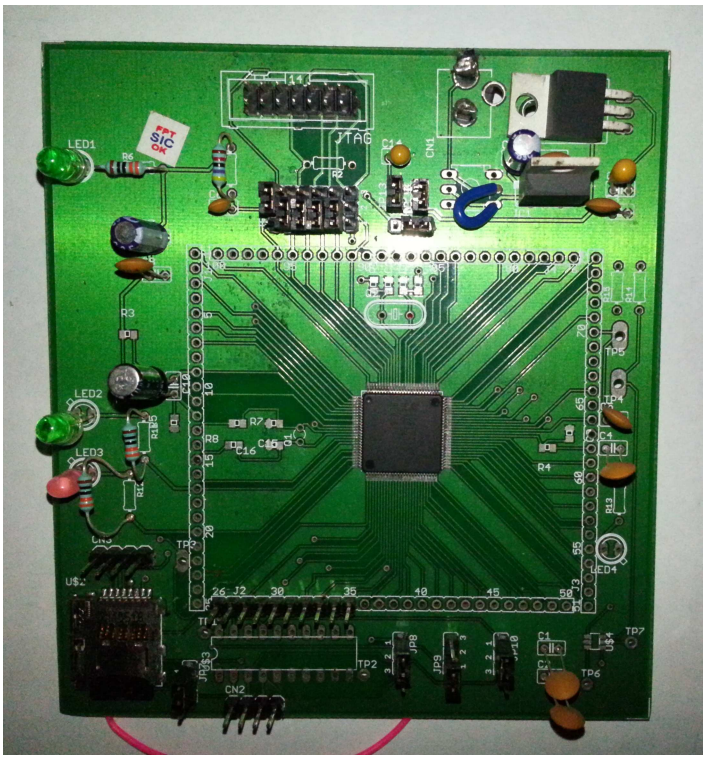

Figure C.4: Image of board 2

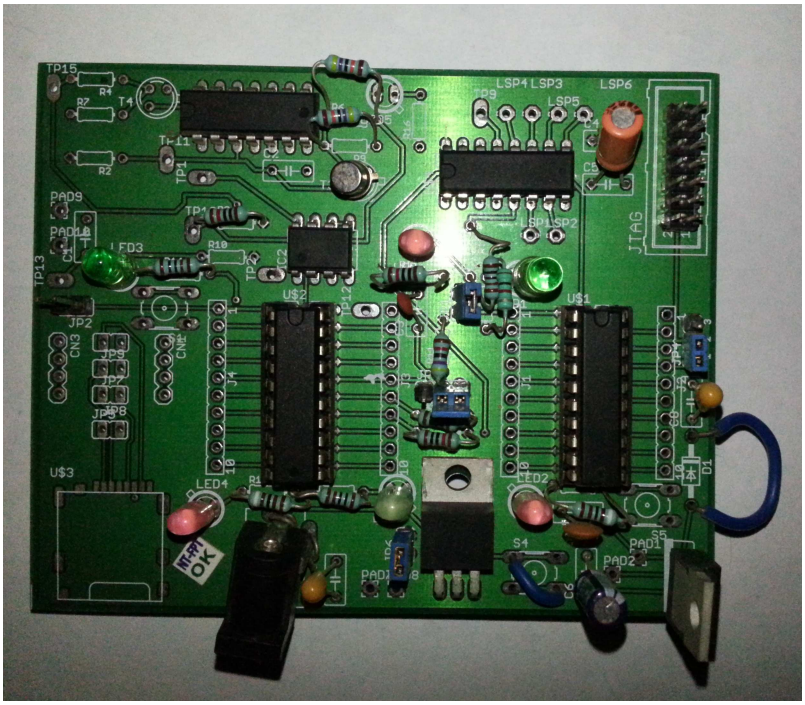

Figure C.3: image of Board 1

Bibliography

[1] C.-P. I. f. I. Principal Investigator, Co-Principal Investigator from IITM, “High Energy Particle Detector, Project Research Proposal to ISRO-IITM Space Technology Cell,” June 2012.

[2] UM10204. I2C-bus specification and user manual. Rev. 5 — 9 October 2012

[3] Implementation of a Reliable Data Bus for the Delfi Nanosatellite Programme N.E. Cornejo, J. Bouwmeester, G.N. Gaydadjiev Proceedings of the 7th IAA Symposium on Small Satellites for Earth Observation, Berlin, Germany, 2009

[4] Elm Chan ,http://elm-chan.org/fsw/ff/00index_p.html,n.d

[5] Texas Instruments,MSP430x5xx/MSP430x6xx Family. User’s Guide. Literature Number: SLAU208G. June 2008