Mini-Project in OS Lab Course

“WildHawk”

Designed By:

Jatin Sharma(Y07uc048)

Manoj Alwani(y07uc061)

PROJECT SUMMARY:

"WildHawk "is a real-time wildlife tracking system based on a simple low complexity file system and a dedicated hardware assembly. This system is created with the help of GNU gcc compiler, Top View simulator, AVR Atmega Studio and Proteus emulator. Along with these Atmel's 89S51 Microcontroller, Atmega8 Microcontroller and TX-RX modules are also used to construct its hardware part. The system comprises of a central node situated on a laptop with a dedicated file system for processing and maintaining the wild life data. The list of the important features are-

1. Simple File System- The file system is based on FAT based approach where on a small storage of 10000 bytes a File Allocation Table is created. Each incoming data is first indexed in this FAT table, a memory location is assigned and data is dispatched for that location and finally being stored. During this process unique pointers for FAT table and storage heads are maintained.

2. Operations- WildHawk provides 7 different operations. Which are-

1. Update Records - Lets you update your database (if any)
2. List Existing Records - Lets you see the current database.
3. Modify Existing Records - Lets you modify the records.
4. Delete Records - Lets you delete records one by one.
5. Open Help Manual - Opens this help manual.
6. Open System Architecture - Shows the WildHawk Architecture.
7. EXIT - Exits from the System securing all data.

3. Essential components - These files are the essential components of the file system.

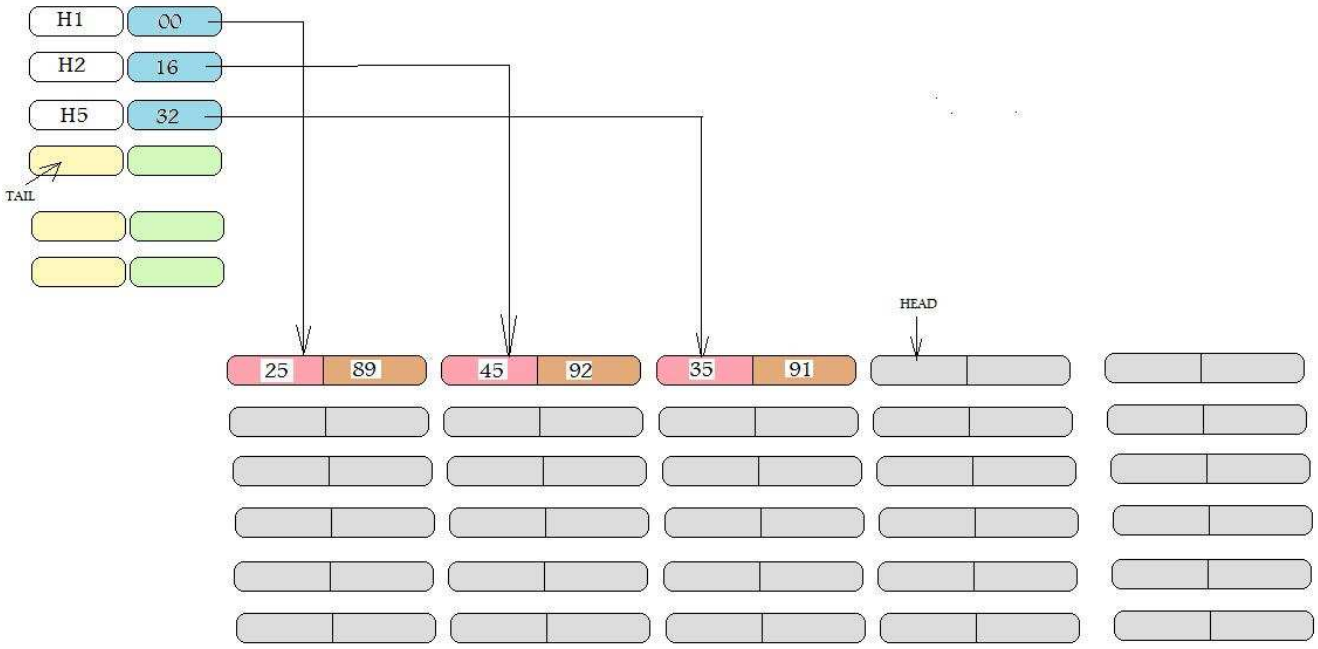
i. wildhawk.c
ii. rx.c
iii. updates.txt
iv. architecture
v. help

4. Automatic creation of File system - While setting up WildHawk system there is no requirement to design the file system. Running the 'wildhawk.c' file automatically creates the file system and configures it.

5. System Architecture- The 10000 Bytes storage is divided into two parts-

(1). File Allocation Table: Here the entries of data is stored and pointers the location of data is being allocated.

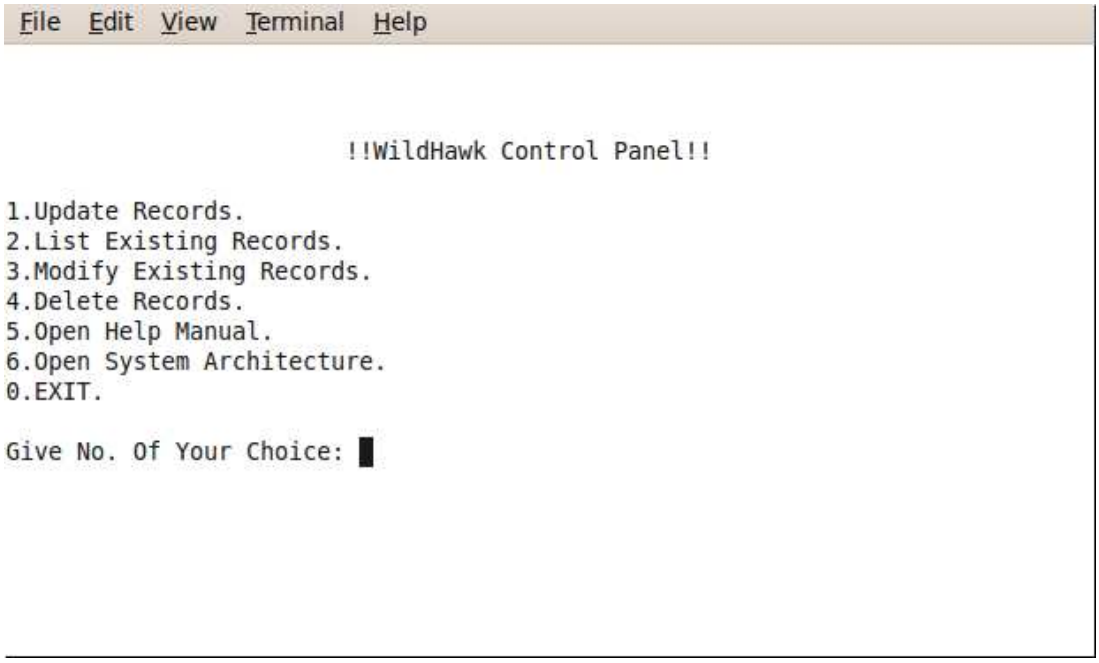
(2). Database: Here actual data is stored.



6. Pointers Consistency-If the execution is stopped the information about data pointers may be lost, to avoid such problem we store the pointers value on some fixed locations in the file system itself, every time we start our system these pointers and headers are restored.

7. Quick Operations – Since we are using FAT based method for any operation of data we have to search only in FAT table rather than going into the actual database so these operations are quick.

6. User Friendly Environment- A user friendly environment is an essential thing for any system. WildHawk provides a very user friendly and help supported system.



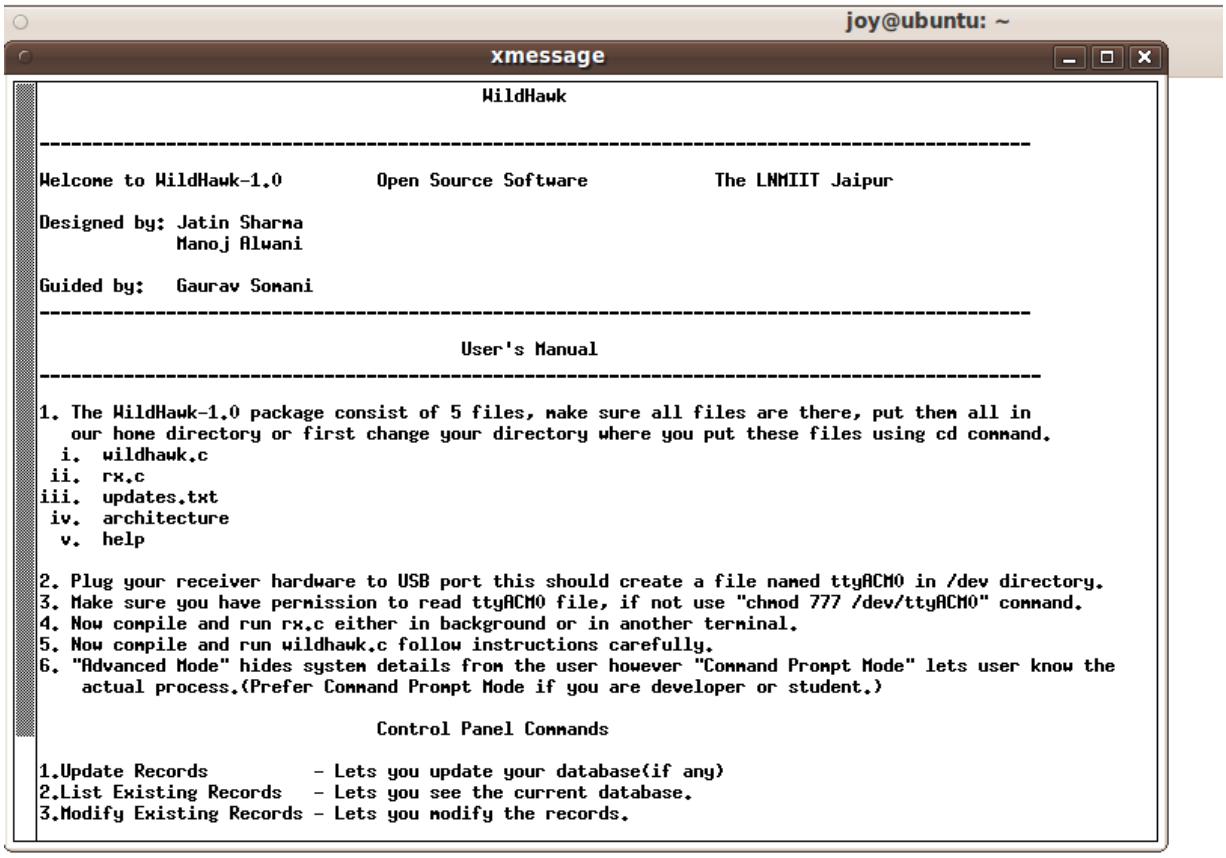
7. Advanced and Command prompt mode- WildHawk provides two different work environments to user. First is advanced mode for novice users which hides all the details from user and uses 'xmessanger' for displaying the records. Second is command prompt mode which shows all the processing on the command prompt only.

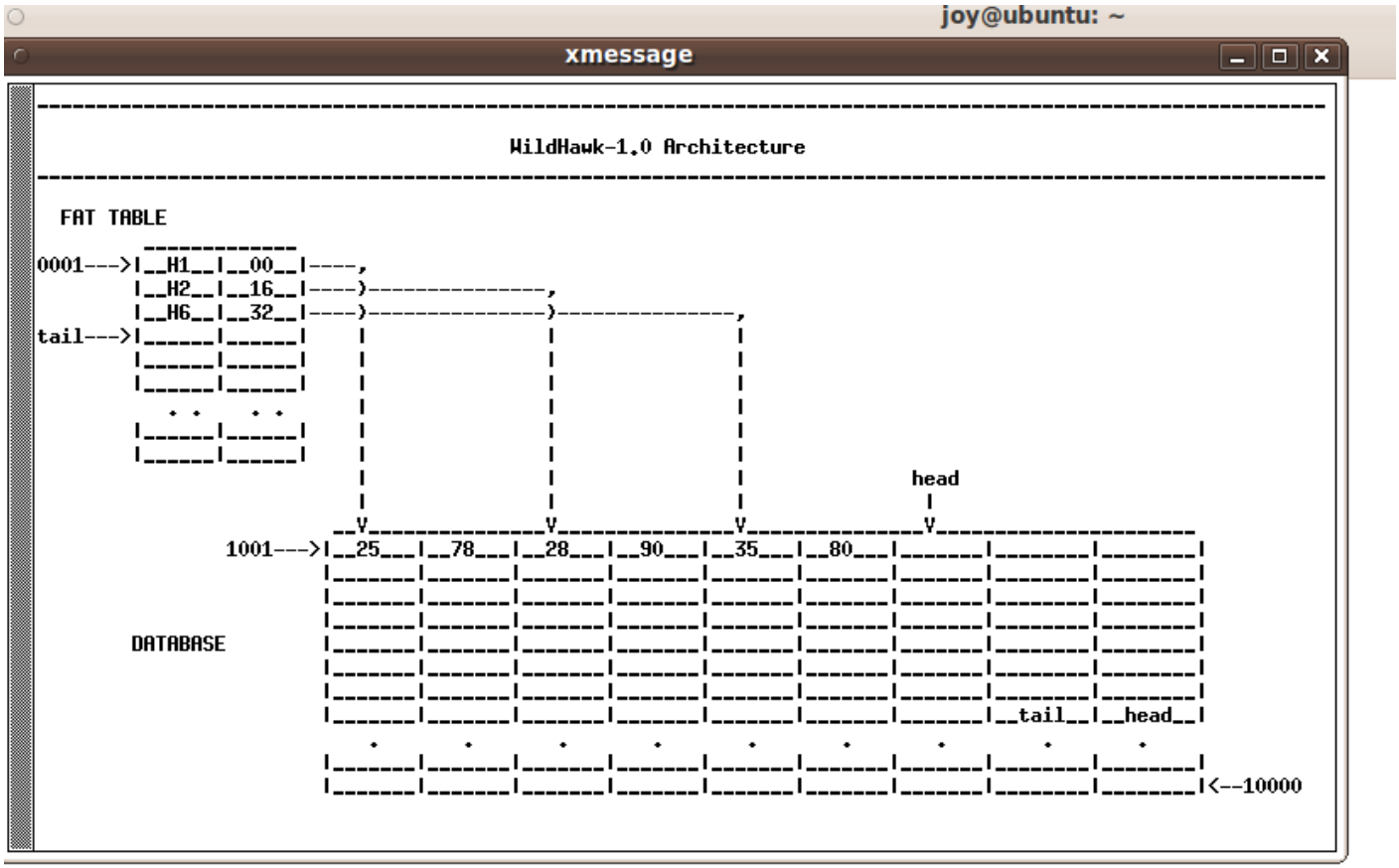
```
File  Edit  View  Terminal  Help
previous head restored: 1080


                        !!Welcome to WildHawk!!
Checks:
1.Plug in your Rx
2.New updates will go to updates.txt.
3.In case you don't have Rx manually write updates in Updates.txt.
4.Run rx.c in background or in another terminal this will enable Rx
  to receive data wirelessly.
5.Advanced Mode hides the details, Command Prompt Mode shows all details.


Advanced Mode/Command Prompt Mode(1/0):
```

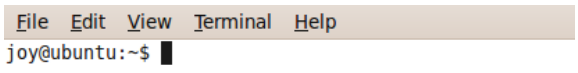
8. Help manual and system architecture- The inbuilt help manual and System architecture facilitate user to understand the actual implementation.



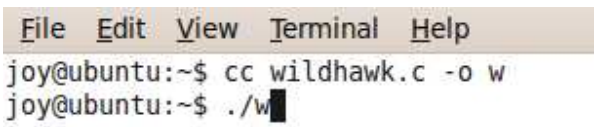


Screen-Shots

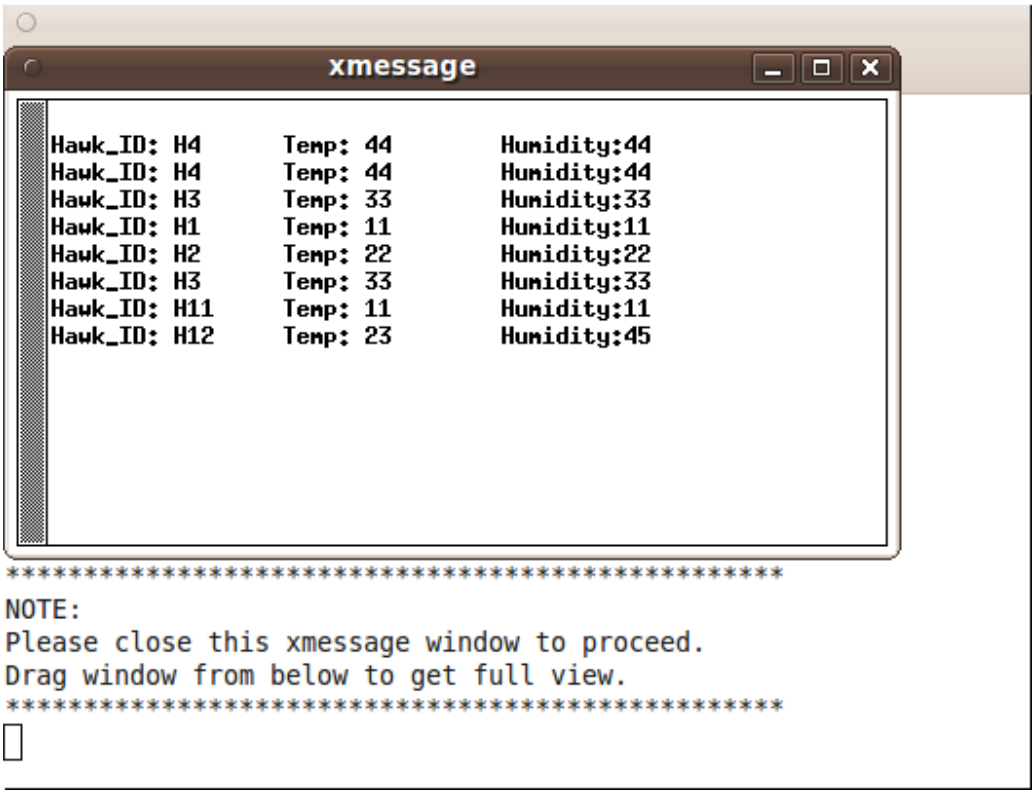
1. Starting on command prompt.



2. Compilation and execution



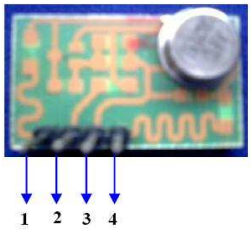
3. Records inspection



Hardware

RF Module:

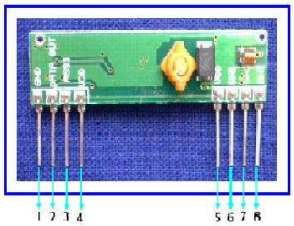
Transmitter



Pin Details:

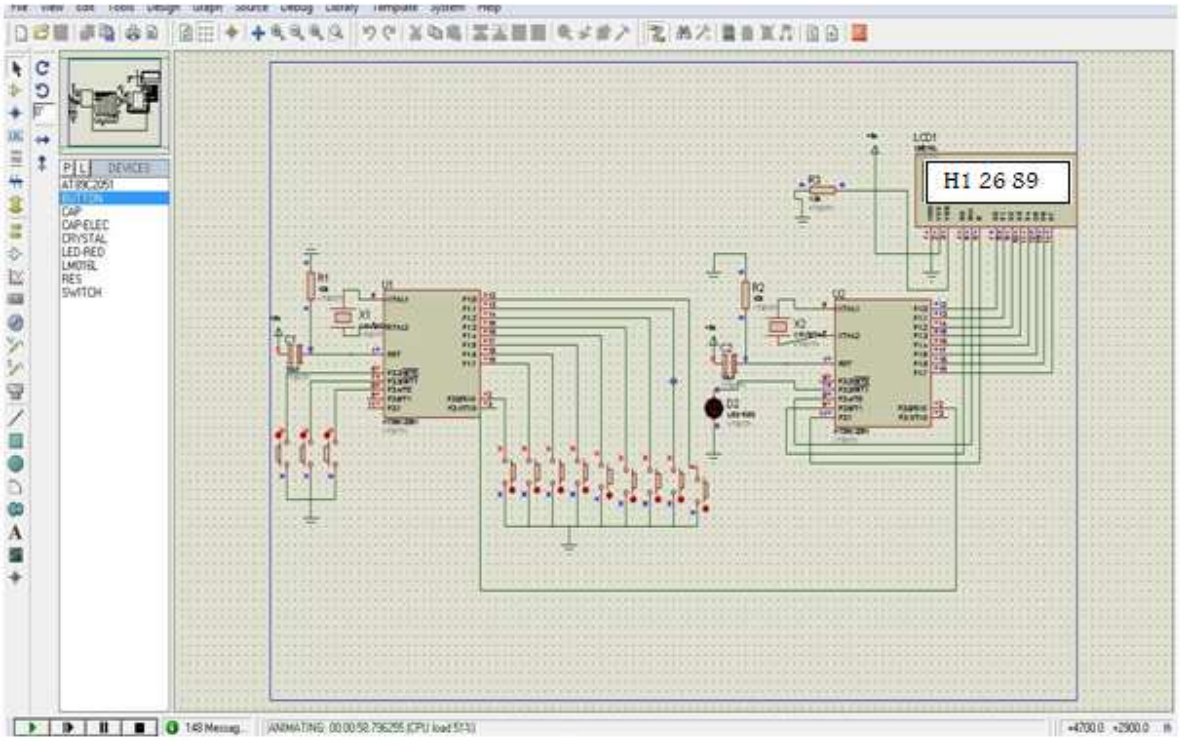
PIN 1	RF OUT
PIN 2	DATA IN
PIN 3	GROUND
PIN 4	VCC

Receiver

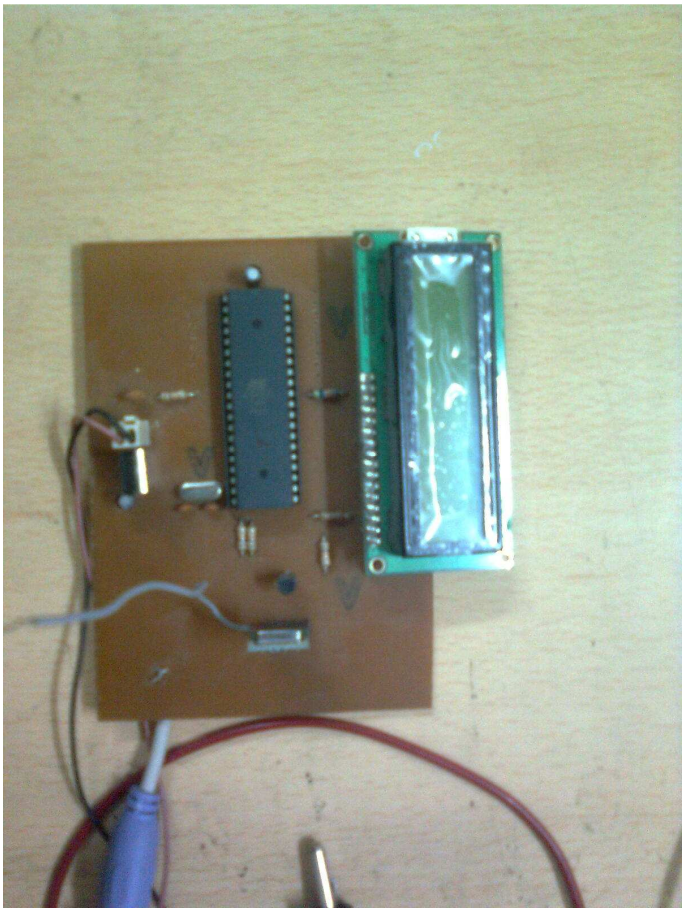


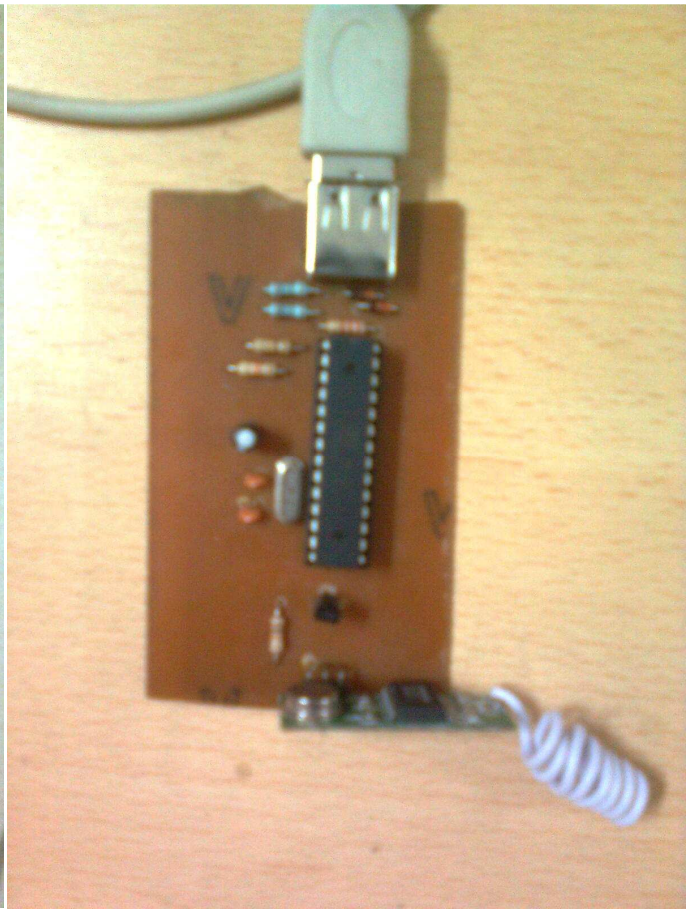
Pin Definition:
PIN 1: GND
PIN 2: DATA OUT
PIN 3: RSSI
PIN 4: VCC (5V DC)
PIN 5: VCC (5V DC)
PIN 6: GND
PIN 7: GND
PIN 8: ANTENNA (About 30 to 35 cm)

Proteus Simulation:



Transmitter and Receiver:





Problems Faced:

USB Interfacing: As laptops do not have any serial or parallel port we were bound to do USB interfacing. It was a difficult job getting the drivers and making the hardware.

Reading Device file: We were unable to read the file /dev/ttyACM0. We had to change the permissions to read this file.

Baud Rate Setting: The transmitter was transmitting at 1200 Baud, so we were required to receive at the same baud rate. We tried a lot when finally we got to know changing the baud rate.

Hardware Problems: During the construction of hardware there were a lot of problems, we tried many ways to implement hardware part including three different receivers and two different transmitters.

Our Experience:

It was really a great experience working in a group of two where both were having different way of looking at the problem and in fact only this empowered us to end-up with such a nice project. Now we find ourselves in a position where we are lot more confident in designing and implementing OS projects in future. It also gave us a direction for our future works and B. Tech Project.

Future Goals:

1. Implementing the solar powered fully fledged sensor network which can be used in practice for wild life tracking system.
2. Implementing a marine life tracking system.
3. Home automation system.
4. Security System for banks and protected zones.

References:

www.cplusplus.com

www.8051.com

'Let us C' – Yashwant Kanitkar

Wikipedia

Advanced Linux Programming

Special Thanks:

Gaurav Somani

LNMIIT, Jaipur