PROJECT REPORT
CS684

<u>SpiderCam</u>

GROUP CODE : 10

SANDIP PAWAR, 123050096
ATUL ROKADE, 123050063
ABHIMANYU KHOSLA, 123050089

Table of Contents

1. Introduction

This project focuses on automating the firebird to monitor a green house. Firebird connected with cables using pulleys and is free to move in different directions both horizontally and vertically. This project will help to monitor a green house remotely.

1.1 Definitions, Acronyms and Abbreviations

Green House: A greenhouse is a building in which plants are grown. These structures range in size from small sheds to very large buildings.

SpiderCam: SpiderCam is a system which enables a camera to move vertically as well as horizontally over a predetermined area.

Control System: A control system is a device to manage, command, direct or regulate the behavior of the SpiderCam.

1.2 References

http://en.wikipedia.org/wiki/Greenhouse
http://en.wikipedia.org/wiki/Spidercam

2. Problem Statement

SpiderCam is a system which enables a camera to move vertically as well as horizontally over a predetermined area. In our project we mount the camera on the bot and the bot is hung with the help of ropes or threads tied in the four corners of a rectangular room. The bot positions itself over a specified area and transmits the video of the trough using a wireless camera. The bot once installed in the Greenhouse should be able to communicate with the user and identify user's requirements to monitor a specific trough in the building or all troughs on the user's request or in a scheduled manner.

3. Requirements

The purpose of this project is to create a SpiderCam which monitors the greenhouse as per the users command, these commands are given by the user through the GUI based application installed on the user`s computer. The main aim of this project is to minimize the human effort and physical presence required at the green house.

3.1 Functional Requirements

The starting point of the bot is always fixed. The bot is free to move in any direction. It captures the image of a particular trough and sends to user as per the users request or as per schedule. The bot calculates the length of each thread from it to the corners of the room (length of the diagonal) and then it calculates the length of the threads at its new position on subtracting the two it gets length which is to be wound or unwound by each motor. The key idea here is to do this in constant time i.e. each motor runs for the same time thus speed of the motors is controlled using PULSE WIDTH MODULATION. Each motor winds a certain length of thread in one revolution which is equal to the circumference of the pulley. Thus controlling the number of revolutions using the Shaft encoder controls the length by which each thread is wound or unwound.

3.2 Non-Functional Requirements

This project has the following Non-Functional Requirements

1. SpiderCam should respond to the user in a predictable time.
2. SpiderCam should be able to localize itself in the Greenhouse.
3. Threads used in the implementation should not expand
4. After completion of every task bot should reposition itself to its home position.
5. Bot should not hit any trough or any wall.
6. In an ideal state bot should be in a mode in which it consumes no or very less energy.

3.3 Hardware Requirements

1. Shaft Encoder
2. Xbee USB Adaptor
3. OnBoard battery support
4. Fire Bird
5. Wireless Camera
6. Pulley Mechanism
7. Customized Shaft attached to the motors
8. Threads

4. Implementation

4.1 C Code Description

An **interrupt handler**, also known as an **interrupt service routine** (**ISR**), is a callback subroutine in microcontroller firmware, operating system or device driver whose execution is triggered by the reception of an interrupt. Interrupt handlers have a multitude of functions, which vary based on the reason the interrupt was generated and the speed at which the interrupt handler completes its task.

An interrupt handler is a low-level counterpart of event handlers. These handlers are initiated by either hardware interrupts or interrupt instructions in software, and are used for servicing hardware devices and transitions between protected modes of operation such as system calls.

Some of the main features of Fire Bird 5 used in our project are

1] ISR for Shaft encoders

2] Timers for Pulse Width Modulation

Details regarding ISR and Timers can be found in Hardware manual mentioned in references section.

Code explains the use of ISR and Timers.

For ISR refer code file ShaftEncoder.c

For Timers refer code file PWM.c

5. Testing Strategy and Data

For testing we have tried rectangular arena and we have found that bot gives more correct output than any other shaped arena. Further this arena is divided into number of troughs. For our naming conventions we assign a particular number to each trough and the co-ordinates of troughs are predefined.
We have also implemented the different modes of controlling bot.

- Automated
- Command

1. Automated

In this mode we have to give number of troughs and trough numbers to cover as a input. Bot will calculate distance to travel to reach next trough from current trough. Then bot will calculate distance in the form of winding and unwinding of four threads.

2. Command

In this mode we have to give direction to move the bot (i.e. forward, backward, left, right). And amount of distance to move. Then bot will calculate distance in the form of winding and unwinding of four threads.

6. System Description

The system has in build pulley, camera mechanism which makes bot itself as autonomous black box. Bot is capable of moving in up, down, forward, back, left, right direction.

Bot takes input from user through Xbee module and operates accordingly.

Working

Bot has two working modes.

1] COMMAND mode

In this mode bot takes input as direction to move(specified by 1 character)

f/F – Move Forward

b/B – Move Backward

r/R – Move Right

l/L – Move Left

u/U – Move Up

d/D- Move Down

s/S - Stop

On the user side command sent through the Xbee module, which is received by other Xbee at bot. After receiving the input bot sends appropriate signals to 4 motors.

As in forward motion front 2 wheels winds the thread while rear 2 wheel unwinds it. To move UP all the 4 motors winds the thread and while coming down all unwinds it.

Forward -> front 2 wind, rear 2 unwind

Backward -> front 2 Unwind, rear 2 wind

Up -> all 4 wind

Down -> all 4 Unwind

Right -> Front Right, Rear Right Wind and Front Left, Rear Left UnWind

Left -> Front Right, Rear Right UnWind and Front Left, Rear Left Wind

2] AUTOMATTED Mode

In Automated mode bot requires 3 main parameters to operate

Command to follow

Distance to cover

Number of troughs

After getting these values bot calculates distance between 2 troughs, after covering calculated distance it stops and captures the images.

It repeats above step till it covers all the troughs.

Things that worked as planned

At the start we planned to make bot operate autonomously. It should have two operating modes in one mode it should follow the users command step by step while in another mode it should get 2 inputs Source and destination trough numbers. After receiving the input it processes it and calculates the displacement required in X and Z direction(Y is fixed at some height).It calculates length of each individual thread to be released or wound to cover X distance bot adjusts speed of motors such that each motor winds or unwinds required length of thread in same amount of time then procedure is repeated to cover Z-distance.

In our implementation the bot reaches the desired co-ordinates but due to the repeated usage of motors the bot looses its height as soon as power is cut off.

Bot takes input form user through Xbee module and operates as per planned. bot can receive input from any Xbee module (provided it is configured with Xbee on bot).

Things that didn't work as planned and their reason

Though the bot takes input from user and properly calculates displacement through Xbee module but it goes way off in traversing the specified distance ,it mainly happened due to the following reasons

1.Speed of motors is dependent on the tension in thread, when tension is low it moves relatively faster as compared with speed when thread are having high tension

2.Tension in the thread depends on the height bot has attained or working on

3.As we are using low torque motors , any random motor starts unwinding thread at random time.

When bot calculates the distance to move in X and Z direction and sets the speed of each motor, because of the above problems it doesn't reach the desired place. Error factor between the distance covered by the bot and the calculated distance is too high because of the height we were working on.We tested the bot at different height and selected the maximum achievable height.

As error factor is too high even at this height we can't use some constant factor to remove the error.

We used some other strategy to resolve the problem, when bot processes the command & calculates the distance to cover, rather traversing the distance at some height bot first moves down by some fixed distance and then moves in Right/Left/forward/Backward direction.After reaching the desired place it adjusts itself to maximum achievable height.

Above solution works only in 1 direction at a time. i.e. It can go Forward , Backward,Right or Left traversing the troughs.

7. Future Work

Our project could not be successfully implemented due to hardware constraints. The firebird has low torque motors which are not suitable for functionality required by SpiderCam. Customize hardware should be build which can integrate high torque like the once used in the power windows in the car. This high torque motors will help in stabilizing bot and also achieve the desired height.

8.Conclusion

This project can be used for automated monitoring of any building or structure. This project has a lot of potential applications which require a monitoring system with a low response time.

As the processor on the bot allows the spider cam to analyze the signals received from the different sensors instantly without the need of transmitting them to another location. The bot can be used as guidance system for other bots which require a hawks eye view of the area and can also alert other bots about different events in the area. Hence SpiderCam can act as eyes for other bots.

The things that seem theoretically possible are not always accomplished as planned especially in an embedded project. SpiderCam seems to be a feasible project but before starting it from scratch again proper Real Time analysis should be done on the hardware that is required to implement it successfully.

9.Refrences

[1] WinAVR User Manual – 20100110 [url : http://dybkowski.net/download/winavr-usermanual.html]
[2] FIREBIRD-V Hardware Manual
[3] FIREBIRD-V Software Manual
[4] X-CTU Refrence Manual

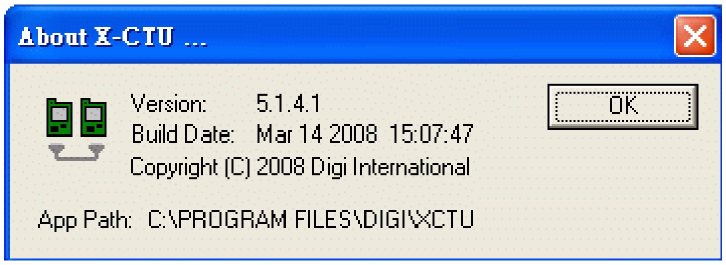
APPENDIX-A

XBee and XBee-PRO 802.15.4 OEM RF modules are embedded solutions providing wireless end-point connectivity to devices. These modules use the IEEE 802.15.4 networking protocol for fast point-to-multipoint or peer-to-peer networking. They are designed for high-throughput applications requiring low latency and predictable communication timing.
While Bluetooth® focuses on connectivity between large packet user devices, such as laptops, phones, and major peripherals, ZigBee is designed to provide highly efficient connectivity between small packet devices. As a result of its simplified operations, which are one to two full orders of magnitude less complex than a comparable Bluetooth® device, pricing for ZigBee devices is extremely competitive, with full nodes available for a fraction of the cost of a Bluetooth node. ZigBee devices are actively limited to a through-rate of 250 Kbps, compared to Bluetooth's much larger pipeline of 1Mbps, operating on the 2.4 GHz ISM band, which is available throughout most of the world.

What is X-CTU?

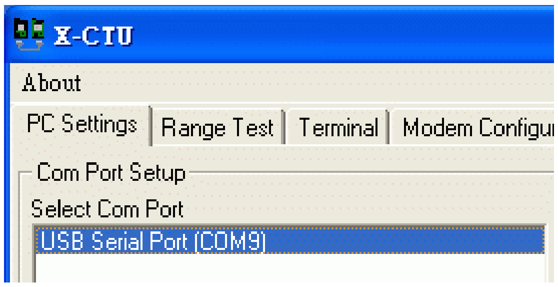
Digi International offers a convenient tool for Xbee module programming - X-CTU. With this software, the user be able to upgrade the firmware, update the parameters, perform communication testing easily. The basic operations are listed below.

How To Use X-CTU?



Version: 5.1.4.1
Build Date: Mar 14 2008 15:07:47
Copyright (C) 2008 Digi International
App Path: C:\PROGRAM FILES\DIGI\XCTU

Before we can talk to an XBee, except USB cable we will need to get an USB adapter for XBee With the USB adapter, we can communicate with Xbee through **"USB Serial Port"**. We may have more than one device on serial port, for example, we want to test the wireless communication and connect two XBee to our PC. So, we will add more devices on serial port. In either case, we need to select the correct one that we want to perform operations.



Com Port Setup
Select Com Port
USB Serial Port (COM9)

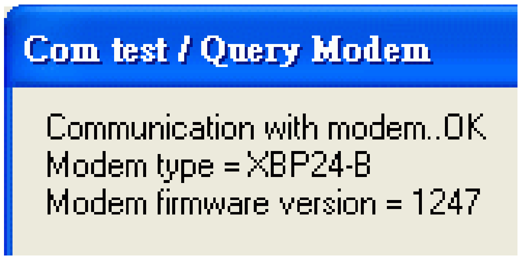
In this case, it only has one Xbee connect to PC and it locates on com port 9 (COM9).

Xbee Query:

This is an easy way to test and check if an XBee is working and configuring properly. After parameter modification and firmware upgrading , we need to do this for checking if everything is ok. we will have a very little chance to get a problem Xbee. If we got problem to Test / Query an XBee , usually it is due to the wrong parameter setting.

For a successful two way communication, the most primary principle is the **"Baud Rate"** should match each other. For a new Xbee module, follow this procedure to query.

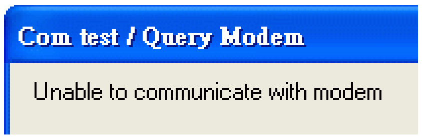
8) Select com port in section : "PC Settings"
9) Baud Rate set to 9600 (only for the new Xbee module , set the value to your case)
10) Flow Control : NONE
11) Data Bits : 8
12) Parity : NONE
13) Stop Bits : 1
14) Enable API : Uncheck
15) Click "Test / Query"
If we had set the Baud Rate to other value, then we should change to it.



```
Communication with modem..OK
Modem type = XBP24-B
Modem firmware version = 1247
```

When everything comes to the right place, this window will appear. If we see any other kind of message window, it means something wrong even we see an OK on it. It is not ok without this message and window. With the same setting, if we only got this occasionally , it means the communication is unstable. We may need to use an XBee adapter with a better quality , or try another XBee.

The modem type and firmware version means the firmware be programmed in this XBee. It is possible to have other types of firmware , depends on your usage for XBee.



Com test / Query Modem

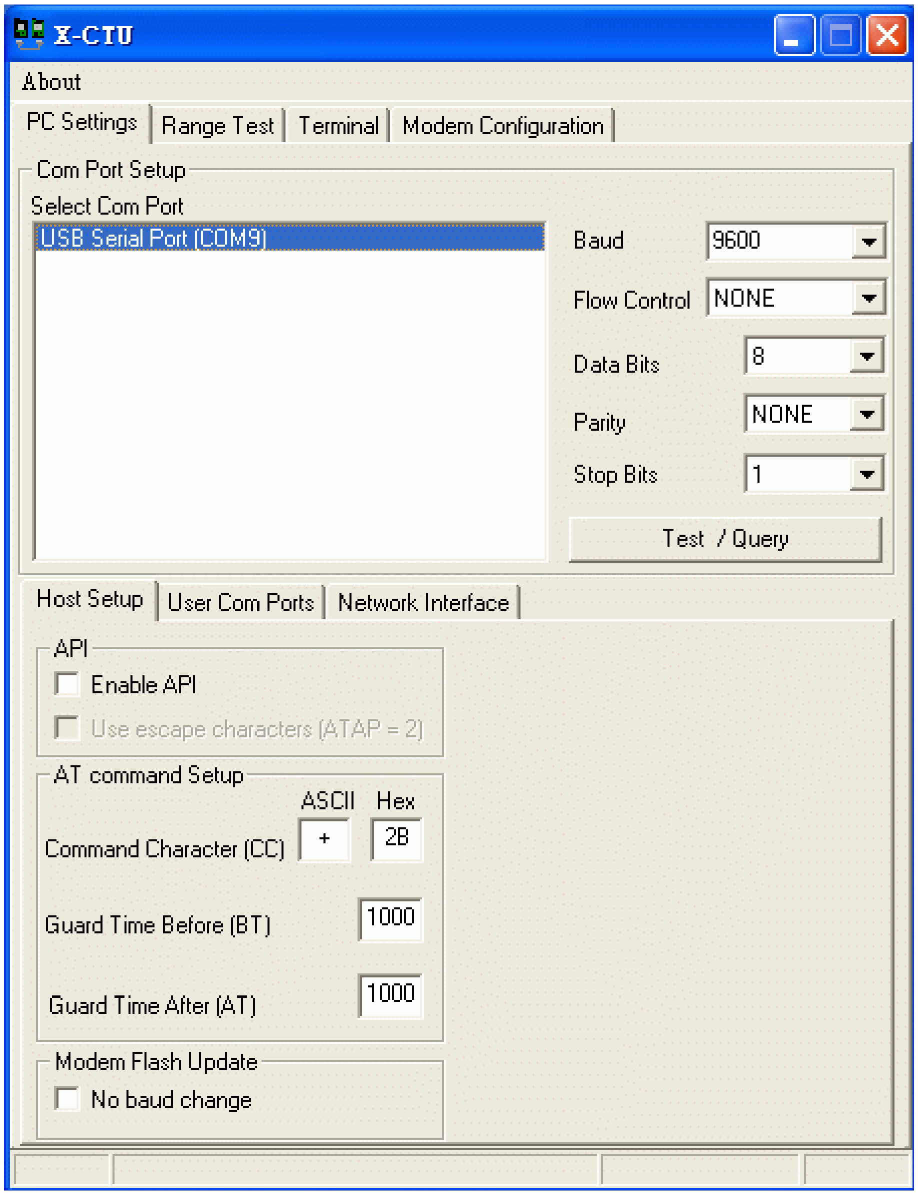
Unable to communicate with modem

If something goes wrong , then we will get this window. For most cases , it can be solved by just changing the Baud Rate and un/check API Mode. We will have a detailed information about this latter.

The wrong firmware in XBee will also result in this kind of message. If the setting and firmware are correct , then the hardware may have problem. In this case , check the adapter first then XBee. A hand made , bad quality Xbee adapter may result in this. Although the specification can not tell this , but XBee Pro apparently has a better tolerance on power source and adapter than normal XBee.

The next section will discuss the four main tabs in X-CTU and information about using XBee with Arduino.

X-CTU User Interface:


X-CTU

About

PC Settings | Range Test | Terminal | Modem Configuration

Com Port Setup

Select Com Port

USB Serial Port (COM9)

Baud: 9600

Flow Control: NONE

Data Bits: 8

Parity: NONE

Stop Bits: 1

Test / Query

Host Setup | User Com Ports | Network Interface

API

[ ] Enable API

[ ] Use escape characters (ATAP = 2)

AT command Setup

	ASCII	Hex
Command Character (CC)	+	2B
Guard Time Before (BT)		1000
Guard Time After (AT)		1000

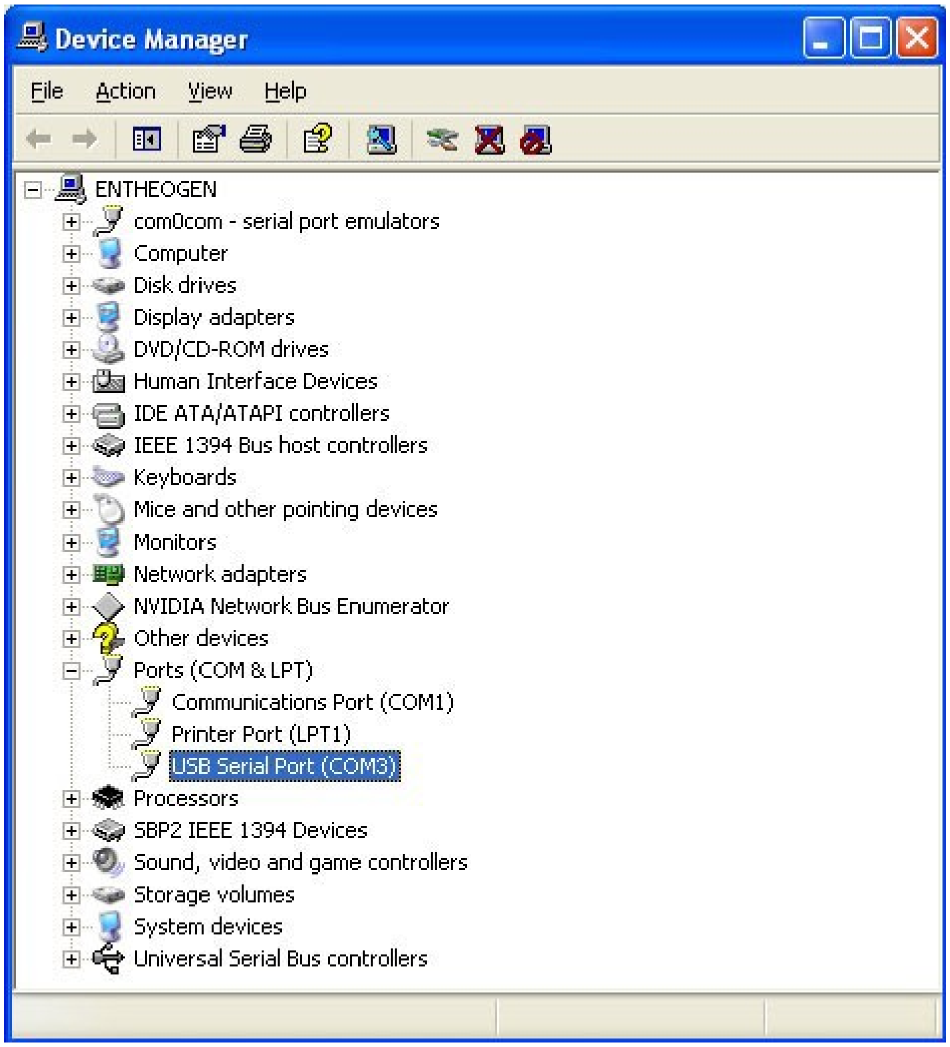
Modem Flash Update

[ ] No baud change

Connecting & Configuring XBee for the project:

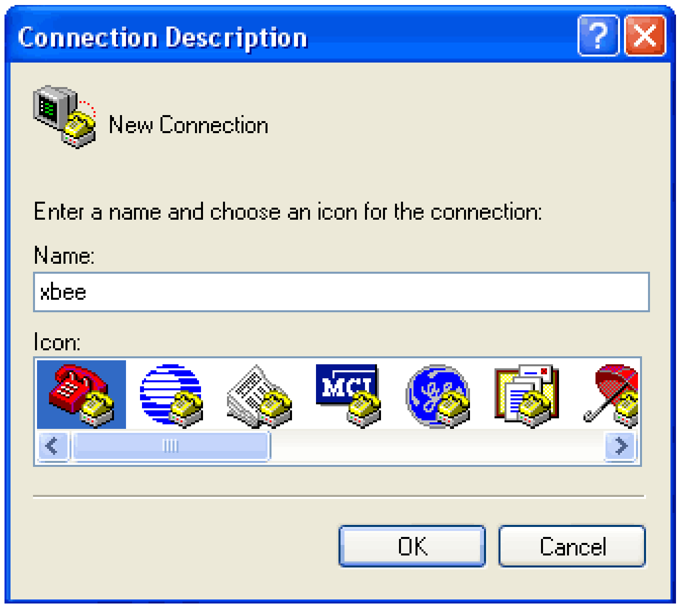
First, insert the xbee module in to the Xbee USB adapter.

Next, we'll need to figure out which serial port (COM) we are using. Plug in the FTDI cable, USB adapter, Arduino, etc. Under windows, check the device manager, look for "USB Serial Port"



ENTHEOGEN
- com0com - serial port emulators
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- IEEE 1394 Bus host controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- NVIDIA Network Bus Enumerator
- Other devices
- Ports (COM & LPT)
  - Communications Port (COM1)
  - Printer Port (LPT1)
  - USB Serial Port (COM3)
- Processors
- SBP2 IEEE 1394 Devices
- Sound, video and game controllers
- Storage volumes
- System devices
- Universal Serial Bus controllers

Next we'll need to open up a terminal program. Windows comes with Hyperterminal, so just use that. Its under **Start->Programs->Accessories->Communications->HyperTerminal**. If we are running a different operating system just use whatever terminal program is available for it, such as ZTerm, minicom, etc.When we open it up, it should ask us for a new connection. Lets name it "xbee"


Connection Description

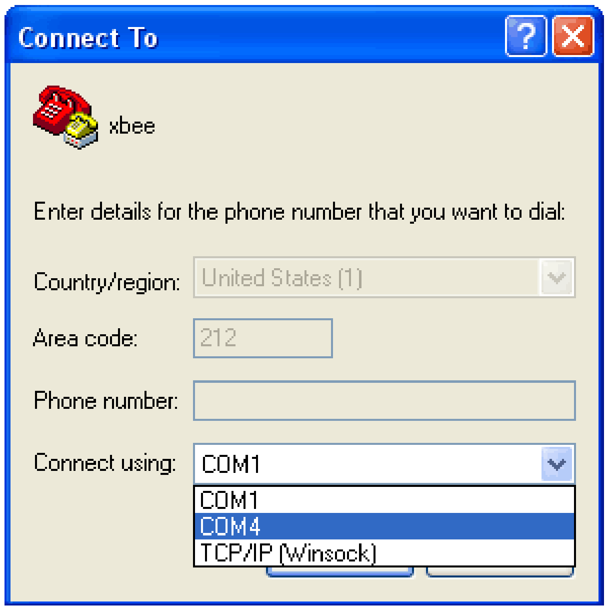
New Connection

Enter a name and choose an icon for the connection:

Name: xbee

Icon:

Next We will select the COM port from the drop down menu, in my case its COM4.


Connect To

xbee

Enter details for the phone number that you want to dial:

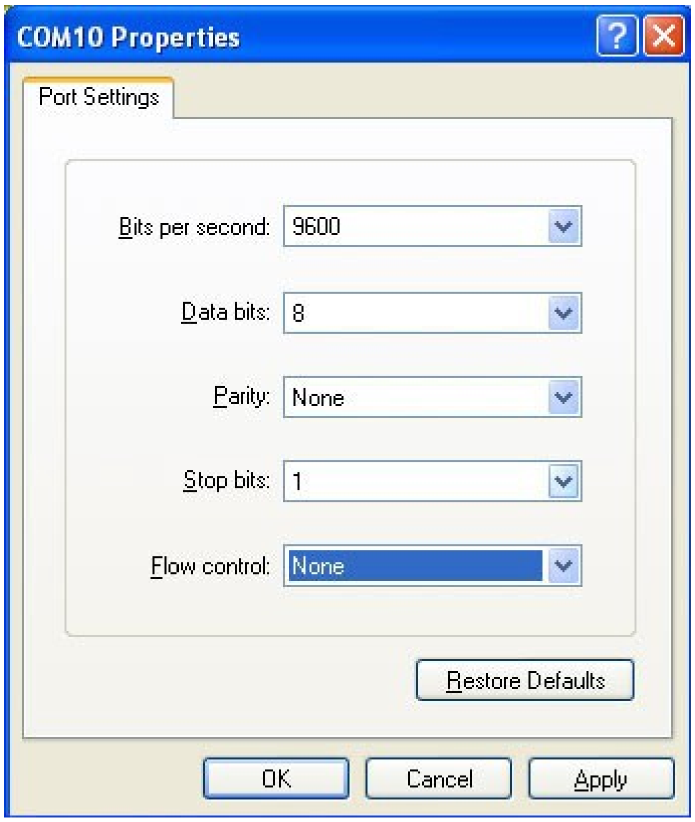
Country/region: United States (1)

Area code: 212

Phone number:

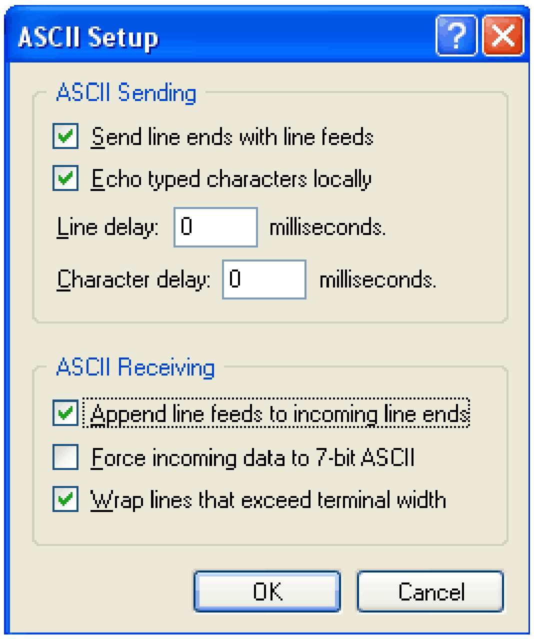
Connect using: COM1

Next, set the properties. We select **9600 bps, 8 bit, No parity, 1 stop bit and no flow control.** Some programs may call this (9600 8N1). If the XBee has been configured for a different baud rate, of course, we should use that.



Bits per second: 9600
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

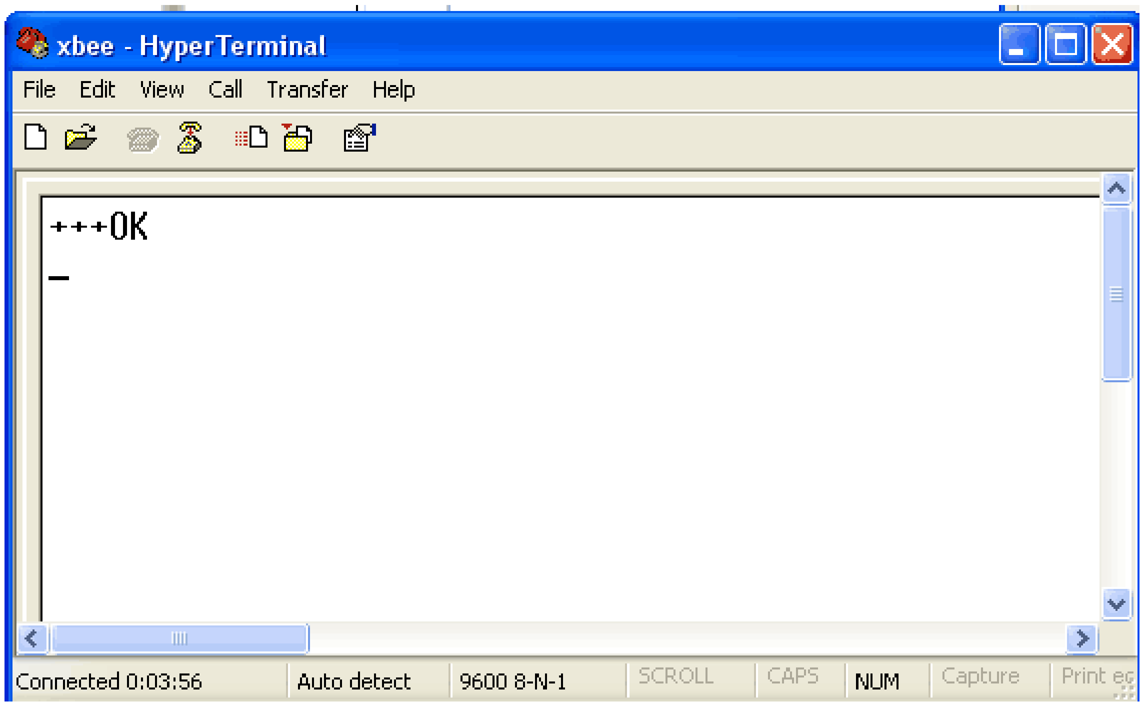
We will get a blank screen that says "Connected" in the bottom left corner. Now, change the setup by selecting **File->Properties** and then going to the **Settings** tab and clicking the **ASCII Setup** button. Make sure we are sending line ends with line feeds and also echoing local Characters



[✓] Send line ends with line feeds
[✓] Echo typed characters locally
Line delay: 0 milliseconds.
Character delay: 0 milliseconds.

[✓] Append line feeds to incoming line ends
[ ] Force incoming data to 7-bit ASCII
[✓] Wrap lines that exceed terminal width

Now type in **+++** (three plus signs) in quick succession. If the XBee is connected up properly we will get an **OK** in response



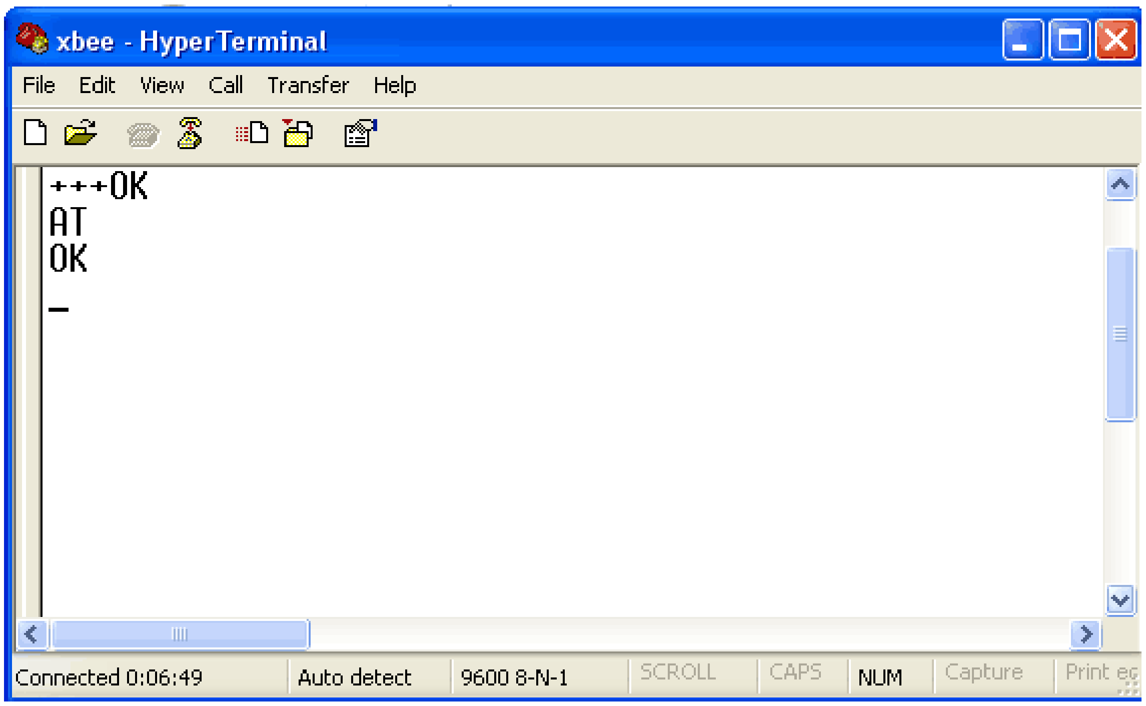
```
+++OK
_
```

Connected 0:03:56 | Auto detect | 9600 8-N-1 | SCROLL | CAPS | NUM | Capture | Print ec

If we got an OK that means the XBee is powered and wired up correctly! If its not working, check:

1. Try again, be sure to wait 10 seconds between each attempt at typing in +++ and type

the +'s quickly
2. Is the module powered? Green LED should be blinking
3. Are RX & TX swapped?
4. Do we have the correct baud rate? By default it should be 9600 baud 8N1 no hardware handshake but if it has been used for something else the baud rate might be different.



```
+++OK
AT
OK
_
```

Connected 0:06:49 | Auto detect | 9600 8-N-1 | SCROLL | CAPS | NUM | Capture | Print echo

Next try typing in +++ (receive **OK**) and then **AT** and press return to get another **OK** This is basically how we can configure the XBee, by sending it AT commands (they all start with AT forATtention). After a while, the XBee times out of configuration mode and goes back to passthrough connection mode. So if we want to get back to config mode, just type in +++ and it will start responding again.

Configuring Xbee for the project:
In Robot, we need to fix the Xbee with the destination address as the Base station xbee. Similarly in all other Swarm Robots all Xbee will contain the base station Xbee’s Serial number as the Destination address.

Base station Xbee can be configured to transmit to anyAddress as we are going to use this communication as the one way i.e swarm robots to base station.