Zigbee Mesh Network

By

Peter Fyon

Andrew Kusz

Bong Geon (John) Koh

Supervisor: Dr. Aitken, Prof. Eatherley

A report submitted in partial fulfillment of the requirements

of SYSC 4907 Engineering Project

Department of Systems and Computer Engineering

Faculty of Engineering

Carleton University

March 22, 2010

Abstract

Acknowledgements

- Dr. Aitken, Prof. Eatherley, Jacob Hammer (for box construction).

Table of Contents

–Description of how we took health and safety into account in the design of our system.

–Low power RF, low powered vehicles = no significant danger to health or property

1.2 Engineering Professionalism

–"Using their course experience of ECOR 4995 Professional Practice, students should demonstrate how their professional responsibilities were met by the goals of their project and/or during the performance of their project."

–kept logs

1.3 Project Management (peter)

In all phases of the project, we used a number of tools to facilitate the management of the project. Weekly meetings were attended to facilitate group work and collaboration, and to ensure that every group member's work was progressing as expected. In addition, log books were kept by each individual and every major research component and design decision was documented. Finally, an off-site Subversion repository was used to store all versions of the code, images,

presentation slides, and documentation in a publicly available, web-accessible location to ensure no documentation was lost and provide a dated history of changes made to the above documents.

1.4 Individual Contributions

1.4.1 Project Contributions

–Peter: project management, initial handyboard example code, majority of the coding of the drive software on the car, modification of the original 1 byte protocol to include NAKless reliable data transfer

1.4.2 Report Contributions

–Peter: initial skeleton outline and bullet points

2.0 Introduction

2.1 Problem Background (andrew)

Typical wireless communications consist of a point-to-point setup; a transmitter transmits a signal to the receiver. Cellphones, wireless routers, and remote control vehicles are some examples of real world applications of point-to-point communications. While point-to-point works in the majority of cases, there are some major limitations that make it less than ideal in some situations.

Over large distances, point-to-point communications can become costly as the power required to transmit the signal increases exponentially with the distance required. A typical antenna will transmit in all directions so when transmitting over these large distances to a receiver located 5km away, the signal is also being transmitted up to 5km in every other direction as well. This could lead to security issues if the signal being transmitted has sensitive data. A simple solution would be to use a dish to focus the signal in only the desired direction, though

that leads to issues with future expansion. Another obstacle is overcoming line of sight issues.

–*Transmitter/receiver pairs are used in many applications*

–*Cellphones, remote control vehicles....*

–*Power use increases related to the range at an exponential rate*

–*signal may be blocked by obstacles*

2.2 Problem Motivation (andrew)

Wireless communications allow us to overcome many obstacles that would be difficult or unfeasible for wired communications. Whether it is harsh terrain, long distances, or temporary installations, wireless networks allow quicker and more inexpensive setup. Wireless communications do have some shortcomings though.

Unlike wired communications, where someone would have to have physical access to the transmission cable to "listen in" on the messages being sent, wireless is broadcast in the open air, allowing anyone potential access to your signal. This security issue becomes more apparent when the signal strength is increased to reach longer distances.

Power usage is another major drawback of wireless communications as the power requirements scale exponentially with distance. Most of this power is then wasted as it results in a coverage area that is most likely much larger than what was desired which ties in with the security issue mentioned above.

Line of sight is another issue which can lower the potential range of wireless communications. Physical barriers such as walls, mountains, or trees can lower the signal quality by quite a bit.

–Many issues related to single transmitter/receiver pairs:

–Much wasted power if devices communicating with base transmitter are not arranged uniformly within the broadcast region

–Less secure communications – large area where communications can be intercepted

–Problems associated with RF communication in tunnels (waveguides)

–Works poorly in areas lacking line-of-sight to transmitter/receiver

2.3 Problem Statement (peter)

Our project seeks to address and improve upon the attributes of range, expandability, power consumption, reliability, and security of a standard long range transmitter/receiver design.

2.4 Proposed Solution (andrew)

To overcome many of the drawbacks of wireless networks, we chose to implement a 802.15.4 based wireless mesh network using the ZigBee specification.

A ZigBee mesh network consists of three types of modules: a coordinator, several routers, and one or more end devices. The coordinator is the central part of the network, and has the responsibility of forming the network by assigning addresses to the other devices. Routers act as relay nodes that can pass the message along to the desired device. End devices are paired with the devices that you need to interact with.

The coordinator can send messages to any node on the network. If the node is not in range, the message will be sent to a neighbouring node which will then forward it onward to the destination. The mesh network can cover a larger range while using less power than a traditional point-to-point network as each node transmits at only a fraction of the power.

Because each node has a smaller transmission range, and relies on the relaying of

messages between nodes to reach further destinations, the coverage area is more precise, which means that the chances of having somebody listening in on the transmissions is less likely.

Line of sight can also be overcome by adding router nodes over or around the obstacles that would normally cause signal degradation in typical point-to-point communications.

The ZigBee mesh network is capable of growing or shrinking depending on one's needs just by adding or removing nodes. Should a node malfunction, the network will try to route packets around the failed node and give the impression of seamless operation.

–Use ZigBee over XBee modules to create a mesh network over which packets can be transmitted. Using this network, demonstrate increased range and better power consumption. Encryption adds security.

–zigbee mesh networks are made up of many transmitter/receivers

–zigbee mesh networks automatically add nodes as they are detected within range of any of the routers

–packets are routed along nodes using a least cost algorithm

–if a node is removed from the mesh, the packets will be routed differently to avoid the missing node

2.5 Accomplishments

–detail accomplishments by the time we've finished this report.

3.0 Technical Report

3.1 Implementation Requirements

–Packets be delivered to the target device in a timely manner

–Vehicle will not drive off on its own if it loses connection

–(I think I have more of these written down somewhere but feel free to add onto it)

3.2 Hardware Overview

–Handy Board specs and reason for choosing over alternatives

–Xbee specs and reason for choosing over alternatives

–Interactive C and reason for choosing over alternatives

–RC car and reason for choosing over alternatives

3.3 ZigBee Mesh Network

How packets are sent to the correct node (ie. sending a packet via node name instead of address)

–Nodes are set up in transparent mode to transmit packets as soon as data is received, rather than buffering it until X bytes have been received

–

3.4 Communication Protocol

3.4.1 Overview (peter)

The communication protocol was designed as a reliable transfer protocol, using one byte for all the data and control bits. The protocol is implemented as a negative-acknowledgement (NAK)-less alternating bit protocol.

An alternating bit protocol uses one bit for sequence number, which alternates between 0 and 1 for each packet. This protocol has the disadvantage of only supporting one data packet in transit at a time, but reduces packet size due the single sequence bit.

A single byte was chosen since it is large enough to support the data that needs to be sent, reduces the processing time for the control board, and simplifies the interrupt routines and

decoding process.

One bit for even parity was considered sufficient for our purposes, as the XBee modules include error checking themselves. The parity bit was added to detect an odd number of bit errors between the receiving XBee module and the serial port on the Handy Board.

The choice was made to use a reliable alternating bit protocol as it was reasoned that the vehicle should not receive command packets out of order, nor should it lose a packet entirely. If the system were being used to control a full sized vehicle, a lost or out of order packet could have severe consequences. The protocol meets the above requirements by using ACK packets and a sequence number to notify the controller of successful, corrupted, and out of order packets so the correct packet can be retransmitted.

3.4.2 Packet Structure (peter)

One bit is used as a parity bit, one bit for the sequence number, one bit to denote an acknowledgement, and five bits for data, for a total of 8 bits, or one byte. See FIGURE_PROTOCOL_BITFIELD.

3.4.3 Protocol Operation (peter)

The protocol functions as follows (See FIGURE_PROTOCOL_STATEDIAGRAM). The controller sends a packet containing the desired commands. When the receiver receives the packet, it first checks the parity bit and sequence number bit. If the parity bit indicates an error, or the sequence number is not the next expected sequence number, the receiver responds with an ACK addressed to the controller, with the sequence bit set to the last successfully received packet, or 0 if the vehicle has not yet received a packet. If the sequence bit is expected and the parity bit does not indicate an error, then the receiver decodes the data bits and performs the operations specified by the commands.

When the controller receives a packet, it first checks the parity and ACK bit. If the parity bit indicates an error, or the ACK bit is not set (ie. a data packet), then the controller ignores the

packet. If the parity bit does not indicate an error and the ACK bit is set, the controller compares the packet's sequence bit to the last sent packet. If the two sequence bits differ, the controller re-sends its last sent packet. If the two sequence bits are equal, then the controller sends the next packet.

–Bitmask for the commands so multiple commands can be sent in the same packet

3.5 Vehicle Drive System (peter)

The vehicle drive system takes the packets obtained through the network from the controller, decodes them, and drives the car forward, reverse, left, or right, in any logical combination.

The vehicle drive system uses an interrupt service routine[REFERENCE TO THE ROUTINE FOUND IN THAT MAILING LIST] to handle incoming packets from the serial port. When a full byte is received in the serial port data buffer, the interrupt code executes and transfers the byte into a 4 byte buffer in memory.

The vehicle control software is written in Interactive C. The control program consists of two motor processes which drive the forward/reverse and the left/right motors, and a main process which polls the buffer for packets, does error checking on the packets, and sends the appropriate commands to each motor process.

The main process polls the buffer in an infinite loop. When a packet arrives in the buffer, the program checks the parity bit and sequence number to ensure the packet not corrupt and has the correct sequence number. If the packet is correct, it will extract the data portion and determine which motors should be stopped, or driven, and in which direction. Conflicting commands, such as forwards and reverse at the same time, are ignored and neither direction is activated. INSERT STUFF IN HERE ABOUT HOW LONG THE MOTORS RUN FOR.

An interrupt-driven approach was taken in the design of the vehicle drive software to ensure that no packets would be lost. The Handy Board does not provide any timing guarantees

for each process, so polling the serial buffer directly was deemed too unreliable.

3.6 Vehicle Control System (johnny)

The vehicle control system takes the input commands given by the user through the joysticks and encodes the instructions into a packet. The possible instructions for the car are forward, reverse, left, right or any logical combination of commands.

The joysticks are connected to microswitches, each dedicated to its own digital port on the Handyboard. The Handyboard polls the digital ports to gather which microswitches have been hit. The control system gathers the input, determines if the input is logical and encodes the instructions into a packet. Once the packet is verified for non-conflicting instructions, the packet is sent into the network for the vehicle drive system to receive and decode. If there is no input, the controller will not send any packets until a new input occurs.

The vehicle control system was designed with intuitive controls in mine. The controller is designed after standard R/C car controllers.

Using an interrupt service routine for the car was deemed unnecessary. Since the vehicle control code was done in Interactive C, polling for inputs was a simple and effective solution within the available resources of the Handyboard. The Handyboard would have not been able to run an interrupt based control scheme while running Interactive C code.

Due to the overall low quality of R/C car used for the project, there was no feasible method of adding precision and accuracy to the control of the car. Having analog inputs would have needless complicated things without giving the car any additional precision or accuracy. Furthermore, digital switches can be still be used to control analog devices. Digital switches were suitable for the project.

4.0 Conclusions

4.1 Discussion (johnny)

While there were many obstacles and difficulties, the overall project was a success, as we met our goals in terms of our proposal. We were able to communicate and control the car over a wireless mesh network. However, there were some technical difficulties within our project.

The components behind our project worked flawlessly. The vehicle drive system, the vehicle control system and the ZigBee mesh network were all functioning as planned before integration. Once we integrated the components together, we realized there was a slight problem communicating between the controller and the car. While the car was able to receive the encoded instructions from the controller, there would be a considerable delay every couple of instructions. We could not measure the exact timing of the delay since we were unable to reproduce the delays consistently. We suspect that XBee nodes and the Handyboards are to blame for this delay.

While testing mesh networking simply with the nodes, we found a similar delay. We attributed this delay to latency between nodes. However, after further testing after integration of components, we suspect that the XBee nodes are clearing a cache or buffer before receiving new instructions. The car would still receive instructions input before, during and after the delay, suggesting the behaviour mentioned.

During tests, we connected the controller and car Handyboards through a wire to test the protocol. We noticed that the physically touching the wire would sometimes produce garbage packets for the car to read. The car would not decode the instruction since the drive system saw that it was an invalid instruction packet. Nonetheless, the Handyboards pose a possible source of delays.

In conclusion, through our project, we discovered that while a wireless mesh network is a great method of low-power wireless communication over long distances, it may not be suitable for a constant stream of data over the network. The delay between the controller and car over the network was restricting control of the car, making seamless car control impossible.

4.2 Future Goals (johnny)

Since we ran into the issue of delays between instructions, an obvious future goal would be to shorten these delays, if not remove them entirely.

Another goal would be to use a wireless mesh network on an interface besides the Handyboard. The Handyboard allowed us to prove that wireless communication between any interfaces was possible, as long as the interface was capable of serial communications. Moving to another interface such as an Arduino, which has an XBee shield, allowing simple integration and full control of an XBee node, would be a good idea.

Another goal would be adding peripherals to the car, such as a camera. With a camera, the car would not have be constantly monitored and would allow full control as long as there's a XBee node nearby.

6.0 References

–All the web references (especially the random assembly stuff for uart communication)

–maybe my 4502 notes for the protocol?

7.0 Appendices

–System block diagram

–Mesh network diagram

–Other reports we did (car teardown, etc)

–All of our code

–State diagram of the protocol I think