Non-Naive Flooding Mesh Network using the Hope RF RFM69

James Adams (ja5g14, 26824744)

Advanced Computer Networks, University of Southampton

***Abstract*—Flooding is used in many networks to build routing tables, for example in Open Shorting Path First routing. This is often the prerequisite to sending routed packets which are not received by all nodes in the network. By creating a mesh network which uses only flooding, this project endeavors to achieve a low computational and spatial overhead, leading to low node power usage and efficient broadcasting, as well as a simple implementation. This efficient flooding may see usefulness in home automation settings where messages are often relevant to all nodes in the network.**

I. Introduction

Low power mesh networks dominate the wireless home automation industry, with protocols such as Z-Wave and Zigbee as well as more ubiquitous consumer protocols such as WiFi. Often the radio modules required to produce devices compatible with these networks are expensive due to licensing fees and high component cost.

By using low cost, off-the-shelf radio modules and micro-controllers, we endeavor to avoid being tied in to expensive ecosystems.

This project attempts to overcome the aforementioned issues

Additionally, the stack size of mesh network protocols such as Zigbee are often large and complicated, requiring higher performance, higher cost, micro controllers. This project aims to avoid the high complexity by not implementing a complicated standard such as IEEE 802.15.4, as well as minimizing code size through eliminating maintenance of routing tables and the associated logic. To avoid this, a wait-listen-repeat approach to routing packets is taken, with an intentional dependence on memory to increase network performance.

II. Background Research

A. Z-Wave

Z-Wave is a proprietary protocol, used primarily in home automation. Radio modules are produced exclusively by Sigma Designs.

The protocol describes a source-routed mesh network, meaning that the sender specifies the route the packet shall take; a route which is computed at install time[1]. This creates the disadvantage that when devices are moved, the reliability of the network is majorly compromised, and the paths must either be recomputed (requiring a manually initiated reconfiguration), or routed by attempting all other routes.

Z-Wave achieves very low resource requirements and hence requires lower performance devices with a far smaller flash size, when compared to Zigbee devices[3].

B. Zigbee

Zigbee, unlike Z-Wave is an open source network protocol capable of forming mesh and star networks. It uses 802.15.4 for Layer 1 (PHY) and Layer 2 (MAC), and implements its own network and application layer[5]. It operates on a range of different frequency bands depending on required application, and is far more resilient than Z-Wave.

The mesh network element of the protocol is self configuring, and self healing, hence nodes can be moved around, join and leave the network, without compromising reliability due to static routes[3].

Zigbee specifies different types of devices. Each network has a single coordinator which manages security and policy, and up to 65535 routers and end devices. Unfortunately a single coordinator adds a single point of failure to the network; a problem which is often dealt with through manufacturer specific protocol extensions.

C. Flooding

Flooding is a simplistic routing algorithm in which incoming packets are re-transmitted by each receiving router. If flooding is uncontrolled this causes packets to be re transmitted continuously, creating a broadcast storm, potentially rendering the network unusable. Hence a controlled flooding approach is necessary for a stable mesh network.

Research by That person what actually did one.

III. Development Board

IV. RFM69 Radio Module and Development Board

The RFM69 is a 'one-dollar' radio transceiver. It can send raw or packetised data, and contains a hardware CRC engine, whitening (as discussed in VII), AES-128 and an on-board temperature sensor[4]. Encryption (though symmetric) along with 868MHz transceiver make it appropriate for in home mesh networking due to greater wall penetration and little need for high data rate. When compared to the NRF24L01[1] we observed a greater indoor and outdoor range, along with a lower power consumption when sleeping.

Unlike Zigbee radios, the RFM69 does not build on a predefined networking standard, and is instead a far simpler module, helping to keep cost and complexity to a minimum.

The Open-source hardware Moteino[2] design was used as a development board. Printed circuit-boards were fabricated using OSHpark[3] and assembled by my co-author and I.

[1]NRF24l01 http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01

[2]Moteino: https://lowpowerlab.com/guide/moteino/

[3]OSHpark PCB: https://oshpark.com/

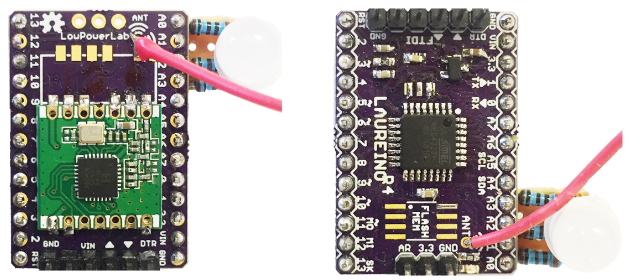

Figure 1. Moteino

This board contains the bare minimum for an ATmega 328p and RFM69. An option regulator is soldered to the board but was bypassed for low power testing. In addition for power testing, the internal oscillator was used, and the board was down-clocked to 8Mhz.

V. Required Features

The following features were determined based on the properties of the aforementioned protocols, and the desired properties.

- A single node type to avoid master- router-sleepy architecture.
- Acknowledgments for reliability.
- Controlled flooding of packets as opposed to routing
- Message repeating to reach nodes out of range
- Broadcast messages for creating a system with shared state, e.g. multiple lights in a room being set to the same color.
- Low overhead and compiled size to allow for deployment on low cost micro-controllers.
- Radio Independent protocol, such that the protocol can be used with a range of low cost modules e.g. NRF24l01, RFM12B, NRF905.

VI. Protocol Design

Node packet memory.

Each packet sent has contains a 16 bit frame counter. This means that the counter will loop around long after a node's packet memory has forgotten previous packets.

only repeat the first time the packet is not in memory – wait a while to repeat. A known time + a little offset for collision avoidance

send message with retry – if it fails wait a while and try again.

packet records its true origin

The only limit to hop count is the network topology, as there is no packet hop counter.

Unlike Z-Wave, protocol does not need to be aware of 'sleepy' nodes, as there is no route calculation, however flags in the library have been created so that space is not reserved for packet history.

no coordinator.

A. Packet Format

1) Unicast Packets (UCAST): Experiments in a household showed that most places were reachable by simple unicast communications. Because of this we determined that this should be prioritised before any repeating occured.

2) Broadcast Packets (BCAST): Broadcast packets are sent on address 254.

3) Acknowledgment Packets (ACKS): Acknowledgment packets are often smaller in size than the payload bearing data packets. Due to this being a demonstration project we opted to still use a payload as although it adds packet bloat, increasing the probability of message failure, it also enables the transmitting of debug information.

VII. Difficulties

Initially when testing the network, the payloads sent were empty, and a large packet size was used, resulting in a large sequence of zero bits in the radio packets. This was further exacerbated by the fact we initially weren't using encryption and hence most packets were 90% zeroes.

This caused a very poor reliability, with a 70% error rate. After much research it was discovered that the RFM69 has built in data whitening, utilizing linear feedback shift registers (LFSR) to avoid long sequences of the same bit value. When this feature was enabled, packet transmission through walls increased to above a 99% success rate when sent with two retries.

Data whitening does not completely solve the equal consecutive bits problem, for example, one could construct a packet which contains the same values as the LFSR, causing all zeroes to be transmitted[2]. Additionally there may be other factors at play reducing reliability, such as distance, and high background noise.

Some time was spent tweaking the default transmit properties of the Radio to get the best reliability inside our home. This involved changing retry time, increasing radio sensitivity, and reducing the bit rate.

VIII. Technologies Used

A. Persistence

For persistence and speed, Redis was used to store previous messages and discovered nodes. This was done due to the many easily accessible data structures available, for example circular queues, allowing the easy storing of the N most recent messages.

B. Pub Sub

Initially Redis was also to be used for publish subscribe methodology however we discovered that MQTT was far more useful due to the fact it supports Web-sockets when used in conjunction with the Mosquito MQTT Broker. Though MQTT does offer data storage through the persistence of topics it did not grant us access to the same data structures as Redis, such as set's.

IX. Visualization

Cytoscape.js[4] was used to display the mesh network. We are able to trace

References

[1] *Z-Wave Technical Basics*, 2011.
[2] Grant Christiansen. Data whitening and random tx mode. *TI Design Notes*, 2010.
[3] Lou Frenzel. What's the difference between zigbee and z-wave? web, 2012.
[4] HOPERF. *RFM69HW ISM TRANSCEIVER MODULE V1.3*, 1.3 edition.
[5] Ankur Tomar. Introduction to zibgbee technology, 07 2011.

[4]Cytoscape.js - A graph visualization library: http://js.cytoscape.org/