

Demonstration of High Data Rate Sensor Acquisition Utilizing a Mesh Network

Richard Ketcham
University of Vermont
Burlington, VT

Jeff Frolik
University of Vermont
Burlington, VT

ABSTRACT

For many successful commercial wireless systems, single hop clusters are utilized due to bandwidth constraints. Multihop systems have a number of advantages over the more traditional single hop topologies including the ability to extend network coverage and to self-heal. Unfortunately, a mesh network often incurs unacceptably long delays and thus becomes impractical for ‘real time’ applications. The proposed demonstration will illustrate a network in which high data rate accelerometer data is collected in a small multihop network.

1. INTRODUCTION

As is well recognized, sensor networks can be generically classified as either multihop, mesh networks or single-hop, (multi) cluster networks. For many successful commercial systems, single-hop clusters are utilized due to bandwidth constraints. Mesh networks however are seen to be beneficial in several ways: the coverage area of the network can be extended beyond the normal nodal transmit range, meshing allows for a network to be formed ad-hoc and to be self-healing. Hence, there are real benefits to employing mesh networks. The major problem with a mesh network is that it can incur unacceptably long delays and so becomes impractical for ‘real-time’ monitoring applications.

The proposed demonstration will illustrate a network in which high bandwidth accelerometer data is collected in a small multihop network. The network will include seven nodes (Freescale MC13192 SARD), one of which will act as the gateway and the other six will act as routers and end devices. The network will be able to collect ‘real-time’ data from all six sensor nodes simultaneously. It will do this by routing data via multiple hops. A PC connects to the gateway and runs a MATLAB-based script that displays and stores the data.

2. HARDWARE PLATFORM

For this work, a commercially available evaluation kit from Freescale, designed to use the 802.15.4 standard, was utilized. In particular, the hardware platform consisted of Freescale’s MC13192 SARD (Sensor Applications Reference Design) board. The MC13192 boards make use of the MC9S08GT60 MCU (a member of the low power, 8-bit HCS08 family) and the MC13192 RF transceiver which operates in the 2.4 GHz ISM band and is sensitive to a received power of -92 dBm. Each SARD board comes

equipped with a built in dipole antenna and also have two low-g accelerometers to sense accelerations in three dimensions (x,y, and z). Custom firmware was developed on top of the 802.15.4 stack as a means to control the hardware in forming and maintaining the proposed network.

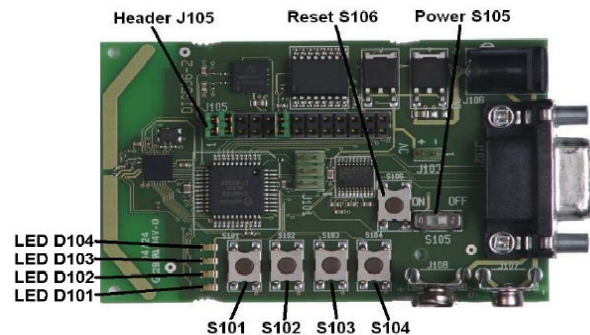


Figure 1. Freescale MC13192 SARD [2]

3. PROTOCOL

The major issues with mesh networking are network propagation time, channel, and system longevity. The scheme that was developed is an attempt at simplicity and employs a number of techniques to mitigate channel contention and system delay such as clear channel assessment and routing tables. The application rests on top of the 802.15.4 standard.

3.1.1 Data Transmitting Nodes

The idea behind the development of the stack for the transmitter was to keep the coding as simple as possible. In this way, there were six key actions that were deemed important for the transmitter to conduct which include: attaining accelerometer data, perform a clear channel assessment (CCA), send the packet, look for received packets, determine if the packets need to be routed, and update the node’s individual routing table. The first four actions are fairly self explanatory: the transmitters transmit accelerometer data and thus need to be able to attain accelerometer data, utilizing the CCA is a quick and easy way to help mitigate collisions, sending and receiving data are a given in a network.

Updating a node’s individual routing table is fundamental to this network. In order to mesh, the node needs to know what other nodes actually exist and, for an efficient network, the node needs to know which devices in

particular to route. As a means of mitigating delay in the transmitters, the gateway is setup to determine which nodes will route which. There are two different routing packets that each node may receive.

The first is a packet that contains the node number and its received power at the gateway. The packet is ordered from greatest power to least. Upon receiving this packet, the node (call it node 1) locates its packet number and compares the received powers it has on record for the other nodes that are listed in the received packet to the right of node 1. If this particular node (node 1) has a received power for another that's greater than that listed in the routing packet then node 1 is closer to the listed node than the gateway. As a result, node 1 is allowed to route packets from this other node.

The second packet contains a node specific routing table. Its purpose is trimming the original routing table such that it encourages packets to take the shortest route possible. This is important because in time critical and quasi-critical applications, delay is often unacceptable. Upon receiving the routing table, determining whether to route a packet becomes trivial: the node just needs to look up the node number in its routing table. It is also possible, though it is not implemented in this demo, that the network could route packets based on signal strength. This scheme would most likely promote reliability but would probably be afflicted by severe delays.

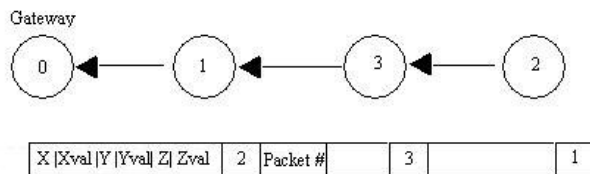


Figure 2. Packet Structure Containing Routed Packets

3.1.2 Data Receiving Node (Gateway)

The gateway, on the other hand, is a bit more complex than the transmitting nodes. Upon receiving a packet, the gateway first determines the length and link quality. The gateway then parses the received packet according to the origin of each section. It starts by checking if the data is valid (if it has the following structure X, Xval, Y, Yval, Z, Zval) and if it has been received before. If the data is valid and hasn't been received before, the data is then sent via serial port to a MATLAB script which displays it; otherwise it just discards the packet. Once the accelerometer data is sent, it checks to see if its routed path has changed. If it has, the gateway then sends the path from which the packet arrived to the MATLAB script. Referring to Figure 2, it can be easily seen that the data from node 2 was routed by node 3 and then by node 1, data from node 3

is routed through node 1, and finally, node 1 has a direct link to the gateway.

The gateway has two algorithms to compute the routing table. The first algorithm, employed at start-up, orders nodes based on link quality from best to worst. This packet is sent out after at least 25 packets (user defined) have been received from at least 1 node. After which, the gateway starts monitoring the paths taken by each packet. It catalogs the paths based on shortest distance. So, if at first the gateway receives a packet where node 2 was routed through 3 and 1 and then another packet where node 2 was just routed through node 1 the path 21 would replace the original path 231. If it happened visa versa (the path 21 was received before 231), then the original path (21) would remain. After receiving a defined number of packets (i.e., 50), the new routing paths are broadcasted to the network. If any of the packet counts from any node is 1 or less, the gateway will default back to using the original routing table method mentioned above. This is to account for mobile nodes or a change in the environment.

For this demonstration, MATLAB was used to create a standalone GUI (via the MATLAB script compiler) that would print out routing paths and simultaneously display the data from all 6 nodes.

4. RESULTS

With the data rate set at the right value, the network operates as planned. MATLAB, unfortunately, is proving to be the bottleneck in this network. Although incredibly powerful, the script has significant delays due to the plotting mechanism. As a result, the maximum data rate depends on the dynamics of the network. The network itself tries to make each packet take the quickest route possible. As a result, two hops happen more often than three or four. This helps in reducing delay within the network. Data rates between the receiver and the PC can range between 2 Kbps and 2.5 Kbps. Specific data rates will depend on the final network setup.

5. REFERENCES

- [1] Freescale Semiconductor, "802.15.4 MAC PHY Software, User's Guide," http://www.freescale.com/files/rf_if/doc/user_guide/802154MPSUG.pdf, 9/2005.
- [2] Freescale Semiconductor, "13193EVK Evaluation Kit," http://www.freescale.com/files/rf_if/doc/user_guide/802154EVKUG.pdf, 6/2005