# Poster: Towards Robust Reprogrammability for Wireless Sensors

Nicole Tobias, Connor Bolton, Josiah Hester, Lanny Sitanayah, Jacob Sorber
School of Computing
Clemson University
{rtobias, mcbolto, jhester, lsitana, jsorber}@clemson.edu

## ABSTRACT

Embedded systems that are wirelessly reprogrammed can be rendered useless by certain programming errors, excessive power consumption, or misconfigurations in the hardware. These types of situations can leave a device in a state that compromises its programmability, often rendering the device useless. Existing attempts to address the problem of robust wireless reprogramming have all been software-based solutions, that are vulnerable to certain errors, such as memory corruption, can corrupt the recovery programs. We propose a hardware-based solution to wireless reprogramming, physically separating the programmer and target device. This separation limits the propagation of errors, and ensures the device will always be recoverable. In this poster we will present the design and an early prototype of our approach — an ultra-low-power, low-cost hardware solution to ensure recovery from fatal errors and reprogrammability in wireless systems. This poster discusses the current system design, initial results, and system analysis from our current prototype. We also present future and ongoing directions, as well as key research questions. This work was funded by National Science Foundation grants CNS-1314342 and CNS-1453607. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

## Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: Microprocessor/microcomputer applications

## General Terms

Hardware, Embedded Systems, Wireless Networks

## Keywords

Hardware Approach, Wireless Reprogramming, Robust Reprogramming, Low-Power, Embedded System


*SenSys '15,* November 01-04, 2015, Seoul, Republic of Korea
ACM 978-1-4503-3631-4/15/11.
http://dx.doi.org/10.1145/2809695.2817903

## 1. INTRODUCTION

Robust wireless reprogramming and seamless recovery from different classes of program errors are a must for a wireless system to have reliable data and a long lifetime. In the past decade, wireless sensor networks have become very prominent in many academic and industrial data acquisition applications. These networks must collect reliable and useful data for accurate analysis to take place. At times, these sensor nodes are deployed with logical errors in their application code. These errors can affect the reliability of the device by rendering data meaningless and hindering its ability to be reprogrammed wirelessly. Human intervention is sometimes required; however, manual recovery of a sensor node is a costly, time intensive, and often impossible process.

Enabling robust reprogramming of a wireless sensor network is a challenging task. Prior approaches to disseminating code updates to a network have been exclusively software-only solutions. A variety of software-based techniques, such as Deluge [2], Rateless Deluge [1], Stream [3], SYNAPSE++ [4], Trigger [5], attempt to address this issue. These techniques present some shortcomings in the presence of certain memory errors that can potentially overwrite portions of restorative code and are limiting in situations where the target device has intermittent power.

Software-only approaches can be rendered inoperable in the presence of certain classes of program errors. For example, if the current program running on the sensor misconfigures the radio, consumes too much energy, or overwrites the recovery program itself, the device may not be available for reprogramming. Programs that misuse memory can overwrite or corrupt portions of memory that are being used by the wireless network and possibly render sensor data meaningless.

We present a novel hardware-based approach to enable the wireless retasking of a sensor device in spite of software and configuration errors introduced by programmers. Our approach uses an ultra-low-power coprocessor that is physically separate from the primary application processor. The coprocessor will monitor the processing, communication, and energy consumption of the application processor to facilitate error analysis and enable robust reprogramming.

## 2. THE SYSTEM OVERVIEW

In contrast to previous systems, our approach is primarily a hardware-based solution that enables more robust reprogramming than prior techniques. This system uses a coprocessor to monitor and reprogram a devices main application processor. This allows for two physically separate computing

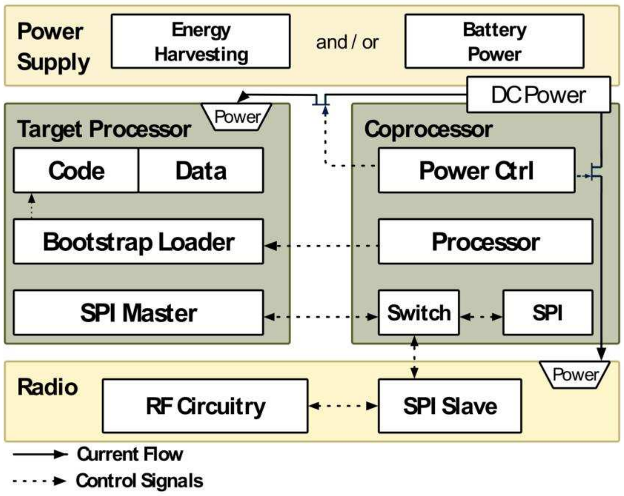


**Figure 1: This figure shows the high level view of the system.**

environments to be utilized for storing recovery programs such that one processor's potential memory errors cannot tamper with the coprocessor's stored restoration image. By using a design that physically separates the recovery program from the target program, the coprocessor cannot be affected by the target program. This is beneficial in scenarios where program errors that overwrite memory on the main application processor are now incapable of overwriting the recovery programs that reside in the coprocessor's memory. This means that the target device will always be able to be reprogrammed and restored in the presence of errors that limited the existing techniques as well as the ones that they still address.

This system is being designed with three goals in mind. We hope the hardware-based approach of this system will provide the following benefits to a wireless sensor:

- Reliable reprogramming
- Robust recovery/fault tolerant system
- Low cost, low power, and a small footprint

The coprocessor stores a recovery image that can be used in the event of target device failure. This recovery image will return the target device to an acceptable state. The coprocessor is positioned in such a way that it is able to observe all communication between the main processor and the device's radio, allowing it to monitor incoming and outgoing communications and changes made to the radio's configuration. Figure 1 shows how the coprocessor monitors the target device.

When the user wishes to reprogram the system, they will define a special packet, called a *restoration packet*, and send it over the radio to signal the need/want to reprogram the application processor. Once the coprocessor recognizes that a restoration packet has been sent, it will restore the main processor to a preset state so that it can accept and install new code updates reliably.

## 3. POSTER DESCRIPTION

This poster will present this on-going project and an overview of the current system. We will describe the designs and future implementation of both the hardware and software of our system. Goals and challenges, as described in the previous section, and initial results from our current work will also be presented.

## 4. CONCLUSIONS & FUTURE WORK

Wirelessly reprogrammed devices and systems will continue to have massive impact on future data collection and dissemination in both academic and industrial research for years to come. To ensure that these systems have a long lifetime and can continue to collect data reliably, they must be able to seamlessly recover from various program error cases and be kept in a state that can accept new programs/tasks throughout the system's continuance. Due to the physical separation of the coprocessor from the application processor, the coprocessor can reliably reprogram the target device regardless of error symptoms resulting from memory corruption, power consumption, and other system errors of the target device. Its ultra-low-power consumption can be optimized to work with devices that are power conservative in nature, such as energy-harvesting sensors.

Long-term monitoring of a low-power system would require the coprocessor device to come online only to assist reprogramming when necessary. Future directions of interest are to expand the current features of our system to also be used in scenarios of remote debugging as well as system recovery. The coprocessor should detect errors, if present, and return useful error information. Both current and future implementations of wireless device networks could benefit from such a low-power, low-cost, robust reprogramming system in many ways.

## 5. REFERENCES


[1] A. Hagedorn, D. Starobinski, and A. Trachtenberg. Rateless deluge: Over-the-air programming of wireless sensor networks using random linear codes. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 457–466. IEEE Computer Society, 2008.

[2] J. W. Hui and D. Culler. The dynamic behavior of a data dissemination protocol for network programming at scale. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 81–94. ACM, 2004.

[3] R. K. Panta, I. Khalil, and S. Bagchi. Stream: Low overhead wireless reprogramming for sensor networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 928–936. IEEE, 2007.

[4] M. Rossi, N. Bui, G. Zanca, L. Stabellini, R. Crepaldi, and M. Zorzi. Synapse++: code dissemination in wireless sensor networks using fountain codes. *Mobile Computing, IEEE Transactions on*, 9(12):1749–1765, 2010.

[5] R. Tompkins, T. B. Jones, R. E. Nertney, C. E. Smith, and P. Gilfeather-Crowley. Reconfiguration and management in wireless sensor networks. In *Sensors Applications Symposium (SAS), 2011 IEEE*, pages 39–44. IEEE, 2011.