Industrial IoT with Crystal-Free Mote-on-Chip

Tengfei Chang*, Timothy Claeys*, Mališa Vučinić*, Xavi Vilajosana†, Titan Yuan‡, Brad Wheeler‡,
Filip Maksimovic‡, David Burnett‡, Brian Kilberg‡, Kris Pister‡, Thomas Watteyne*
* Inria, Paris, France.
† Universitat Oberta de Catalunya, Catalonia, Spain.
‡ UC Berkeley, California, USA.

Abstract

SCμM is a $2\times3\times0.3$ mm^3 system-on-chip that contains an ARM Cortex-M0 and a 2.4 GHz IEEE802.15.4 radio. This paper describes the two-step calibration routine needed to run a full 6TiSCH stack on SCμM. It is, to the best of our knowledge, the first time a fully standards-compliant protocol stack runs on a crystal-free radio, such that it can participate in a network with off-the-shelf radios.

Keywords: 6TiSCH, crystal-free, IoT, IEEE802.15.4, TSCH.

Introduction

The Single-Chip Micro Mote (SCμM, Fig. 1) is a $2\times3\times0.3$ mm^3 system-on-chip that contains an ARM Cortex-M0 and a 2.4 GHz radio compatible with IEEE802.15.4 and BLE [1]. Unlike existing radios, SCμM only requires an antenna and a power source, no external crystal oscillator. SCμM realizes the "Smart Dust" vision by offering standards-compliance: it communicates with off-the-shelf radios.

Applications such as smart wearables or microrobotics require dependable networking between SCμM chips. A reference standardized Industrial Internet of Things protocol stack is 6TiSCH [2], standardized by the Internet Engineering Task Force (IETF). We were able to run the 6TiSCH stack on a SCμM using an external crystal oscillator [3]. The work we present here is the first to show the 6TiSCH stack running on SCμM with no crystal oscillator.

Calibrating SCμM's Oscillators

Without a crystal oscillator, SCμM relies on a free-running 2.4 GHz RF LC oscillator for tuning the communication frequency. Three separate RC oscillators are used to generate the transmitter chipping rate (2 MHz), clock the receiver (64 MHz), run the micro-controller (5 MHz), and drive the scheduling timer. These oscillators exhibit a clock drift of up to 16,000 ppm over an 80 min 25-70 C temperature ramp, compared to 10-40 ppm for typical crystal oscillators. Calibrating these oscillators is a significant challenge because of the non-linearity of the tuning, and the 40 ppm target. This paper uses two successive calibration steps.

We use optical calibration to provide an initial coarse calibration to all four oscillators. SCμM contains a photodiode and the circuitry for an optical programming board to load firmware onto SCμM using an LED [4]. We augment this by having the LED send a specific 32-bit sequence that triggers an interrupt on the micro-controller 100 times after loading the binary, with a 100 ms period. At each LED sequence, SCμM stores the value of the clock counters for calibration (Fig. 2). This is sufficient to initially calibrate the RC oscillators with enough accuracy to establish communications.

To calibrate the RF oscillator, we calibrate against a "QuickCal box": 16 off-the-shelf IEEE802.15.4-compliant devices (OpenMote) programmed to transmit a beacon frame every 600 μs, each on one of the 16 IEEE802.15.4 2.4 GHz frequencies. Calibrating the RF oscillator consists of adjusting three 5-bit capacitive tuning DACs. As shown in Fig. 3, SCμM first sweeps through 1024 radio RX settings, listens on each for 800 μs, and records the radio setting for each of the beacons received. Frequency calibration for RX and TX are independent. SCμM therefore sweeps a second time to calibrate radio TX settings by transmitting a frame every 1.32 ms and listening for an acknowledgement frame. RF calibration takes a total of 174 s, including the time for receiving the first frame from the QuickCal box. This version of SCμM has not been fully optimized for energy, the active power of the entire system is 1 mW.

Running Industrial IoT protocol

After calibrating against the QuickCal box, SCμM takes an average of 3.2 s to synchronize to an in-range OpenMote running TSCH. SCμM's timers are designed for running the 6TiSCH stack: during a 20 ms timeslot, SCμM's advanced timers orchestrate the execution of a timeslot state machine without microcontroller intervention (Fig. 4). This removes the need to wait for the microcontroller to be active, resulting in a potentially shorter timeslot and reducing energy consumption.

After optical and RF calibration, SCμM runs the full 6TiSCH protocol stack and communicates with an OpenMote, forming a 6TiSCH network (Fig. 5). After joining the network, SCμM relies only on receiving periodic beacon frames from the OpenMote for re-calibration. Fig. 6 indicates SCμM stays synchronized to the 6TiSCH network as long as an OpenMote in range sends a beacon at least once every 440 ms.

Future work involves the extension of the temperature compensation used in [1] to apply frequency corrections derived from network traffic to all the free-running oscillators.

References

[1] Maksimovic, "A Crystal-Free Single-Chip Micro Mote with Integrated 802.15.4 Compatible Transceiver, sub-mW BLE Compatible Beacon Transmitter, and Cortex M0," in *VLSI*. IEEE, 2019.
[2] Vilajosana, "6TiSCH: Industrial Performance for IPv6 Internet-of-Things Networks," *Proceedings of the IEEE*, pp. 1–13, 11 April 2019.
[3] Chang, "Demo: 6TiSCH on SCuM, Running a Synchronized Protocol Stack without Crystals," in *EWSN*, 2020.
[4] Wheeler, "A Low-Power Optical Receiver for Contact-free Programming and 3D Localization of Autonomous Microsystems," in *UEMCON*, 2019.

978-1-7281-9942-9/20/$31.00 ©2020 IEEE

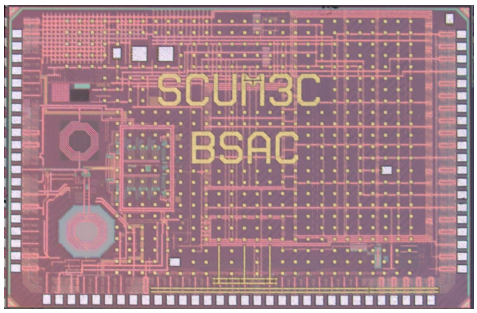

Fig. 1. The Single-Chip Micro-Mote (SCμM) is a 2×3×0.3 mm^3 crystal-free mote-on-chip which features an ARM Cortex-M0 microcontroller and a 2.4 GHz radio compatible with IEEE802.15.4 and Bluetooth Low Energy [1]. It operates without requiring any external crystal oscillators.

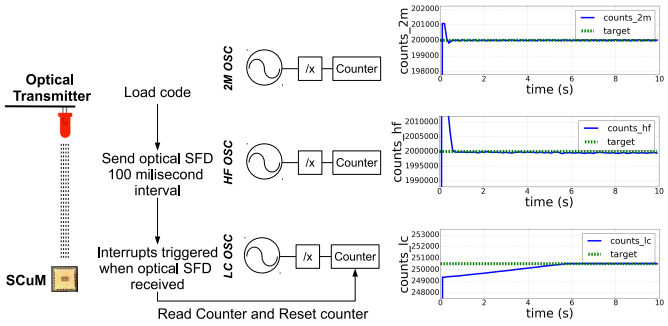

Fig. 2. Optical calibration, using the optical bootload board. After loading the firmware into RAM, the optical bootload board sends 100 pulses, one every 100 ms. The firmware captures and resets the counters of timers attached to the 3 oscillators at each pulse. It then uses that information for calibrating the 2M and HF oscillators.

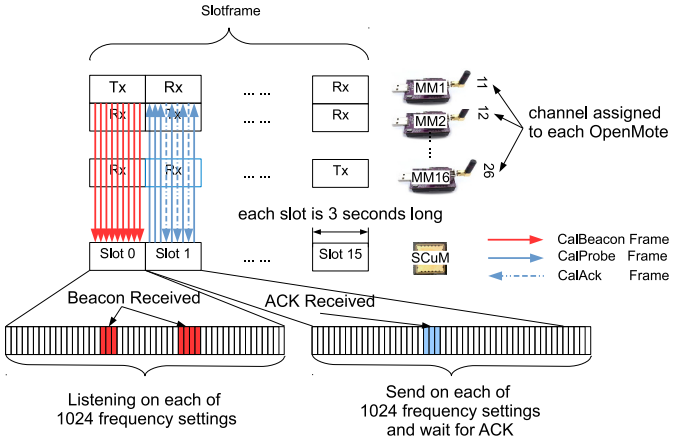

Fig. 3. RF calibration, needed to calibrate the RF oscillator. This calibration uses a "QuickCal box" of 16 OpenMotes. SCμM sweeps all frequency settings twice: first to calibrate its RX frequency settings by receiving frames from the QuickCal box and then to calibrate its TX frequency settings by transmitting frames to the QuickCal box and listening for acknowledgement frames.

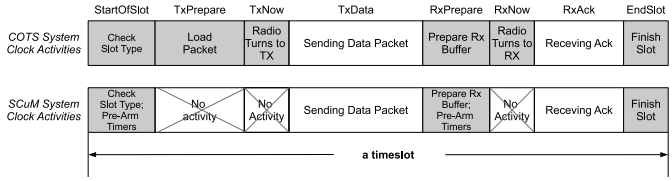

Fig. 4. SCμM provides advanced timers to trigger specific radio activities, including loading packets or issuing transmitting/receiving commands. This allows SCμM's Cortex-M0 to stay off during most of the slot. All other TSCH implementations on off-the-shelf IEEE802.15.4 chips require the microcontroller to execute code at each state transition.

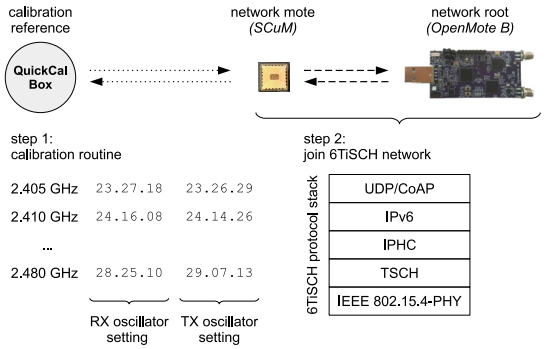

Fig. 5. After quick calibration has finished, SCμM records the frequency settings for both transmitting (TX) and receiving (RX) on each of the 16 frequencies. These settings are represented as `X.Y.Z` where `X`, `Y` and `Z` each represent the setting of a 5-bit tuning DAC. The firmware then applies these settings while executing the channel-hopping 6TiSCH protocol.

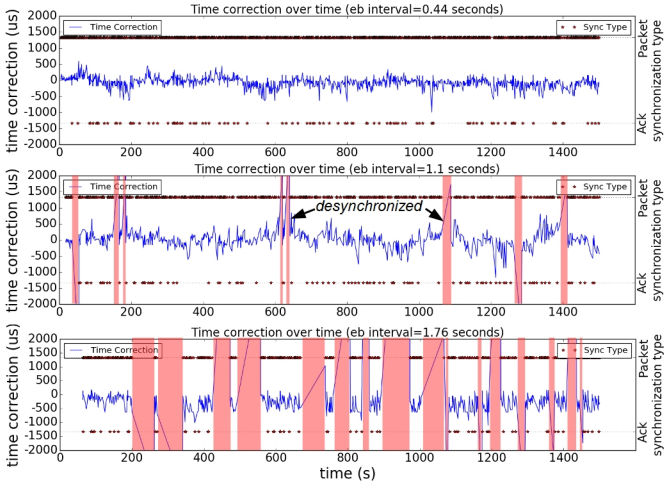

Fig. 6. Re-synchronization activity of a SCμM chip against an OpenMote while executing the 6TiSCH protocol. The OpenMote sends a beacon frame that allows SCμM to synchronize every 400 ms (top), 1.1 s (middle), or 1.76 s (bottom). By synchronizing to these beacons or to specific keep-alive messages it sends to the OpenMote, SCμM applies a time correction (blue line). This time correction cannot exceed 1,300 ms, the "Guard Time" of the 6TiSCH implementation. If it does, SCμM cannot communicate with the OpenMote anymore, desynchronizes, and needs to resynchronize, an operation that takes time and energy and during which it cannot generate data. Periods of desynchronization are highlighted in red. The less frequently the OpenMote sends beacons, the more frequent are the desynchronization periods.