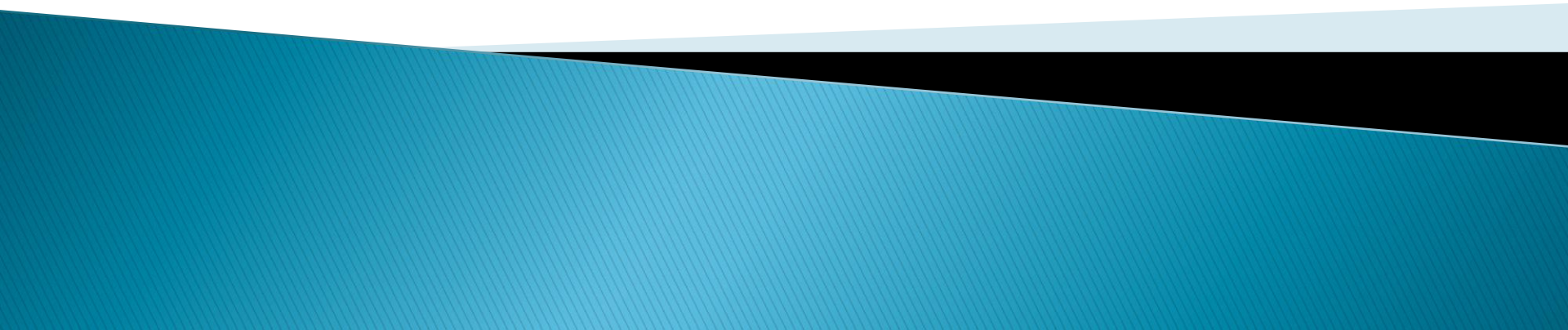


Ultra-Low Duty Cycle MAC with Scheduled Channel Polling

Wei Ye, Fabio Silva, and John Heidemann
USC Information Sciences Institute

Presented by:
Murad Kaplan



Outline

- ▶ **Introduction.**
- ▶ Design of SCP-MAC.
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ Protocol Implementation.
- ▶ Experimental Evaluation.
- ▶ Related Work.

Introduction

- ▶ Energy is a critical resource in battery-powered sensor networks.
- ▶ Major sources of energy waste are idle listening, collision, overhearing, and control overhead.
Among them, idle listening is a dominant factor in most sensor network applications.
- ▶ Central approach to reducing energy lost to idle listening is to lower the radio duty cycle.
- ▶ Three approaches are generally used: TDMA, scheduled contention, or low-power listening.

Introduction (Cont'd)

- ▶ TDMA is not considered, difficulties that arise in networks of peers (lack centralized or cluster-based control, and at very low duty cycles).

Introduction (Cont'd)

- ▶ *Schedule coordinated transmission and listen periods:*
 - Seen in S-MAC, TMAC, and TRAMA.
 - Determines when a node should listen and when it should sleep.
 - Receiver, only listens to brief contention periods.
 - Senders, contend during these periods.
- Only nodes participating in data transfer remain awake after contention periods, while others can then sleep.*
- Reduces energy.

Introduction (Cont'd)

- ▶ *Low-power listening (LPL):*
 - Presented in WiseMAC and B-MAC.
 - *Channel polling*, nodes wake up very briefly to check channel activity without actually receiving data.
 - *Long preamble*, so senders rendezvous with receivers.
 - Consume much less energy than existing scheduled protocols.

Introduction (Cont'd)

- ▶ *low-power listening (LPL) – Problems:*
 - Receiver and polling efficiency is gained at the much greater cost of senders.
 - Very sensitive to tuning for an expected neighborhood size and traffic rate because of the balance.
 - Challenging to adapt LPL directly to newer radios like 802.15.4.

Introduction (Cont'd)

- ▶ Scheduled channel polling (SCP-MAC) !!!
 - Synchronize the channel polling times of all neighbors.
 - Eliminates long preambles in LPL for all transmissions.
 - Able to operate at ultra-low duty cycles when traffic is light.

Introduction (Cont'd)

▶ SCP-MAC

◦ Challenges:

- Understanding the optimal behavior of both scheduling and channel polling separately and together.
- Placing a lower bound on energy costs.
- Developing a protocol that adapts to dynamically changing traffic patterns efficiently.
- Understanding how these techniques apply both to existing (CC1000) and new (802.15.4, CC2420).

Outline

- ▶ Introduction.
- ▶ **Design of SCP-MAC.**
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ Protocol Implementation.
- ▶ Experimental Evaluation.
- ▶ Related Work.

Design of SCP-MAC

- ▶ Designed with two main goals:
 - Pushing the duty cycle an order of magnitude lower than is practical with current MAC protocols.
 - Adapt to variable traffic loads common in many sensor network applications.
- ▶ To meet these goals:
 - Combining scheduling with channel polling.
 - Deriving and employing optimal intervals for schedule synchronization based on worst-case clock drift.
 - Developing a new algorithm to dynamically adjust duty cycles in the face of busy networks and streaming traffic, reducing the latency in multi-hop networks.

Design of SCP-MAC (Cont'd)

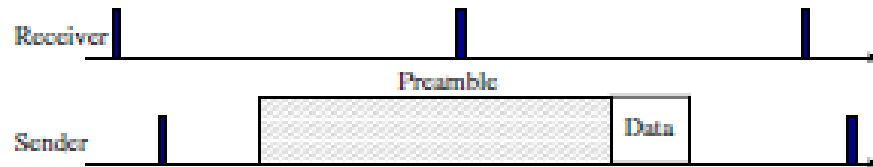
▶ Synchronized Channel Polling

- Channel polling reduces the cost of discovering traffic.

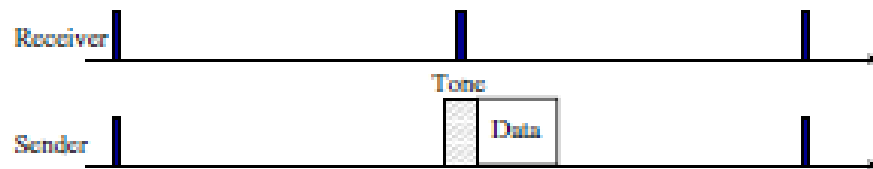
Checking for the presence or absence of network activity is much cheaper than knowing what the activity is

- SCP-MAC adopts channel polling from LPL approaches.
- Unlike LPL, SCP-MAC *synchronizes the polling* times of all neighboring nodes.

Design of SCP-MAC (Cont'd)



(a) Low-power listening (LPL)



(b) Synchronized channel polling (SCP)

- Synchronization reduces the cost of overhearing.
- Synchronization SCP works efficiently for both unicast and broadcast traffic.
- Short wakeup tones make SCP-MAC more robust to varying traffic load.

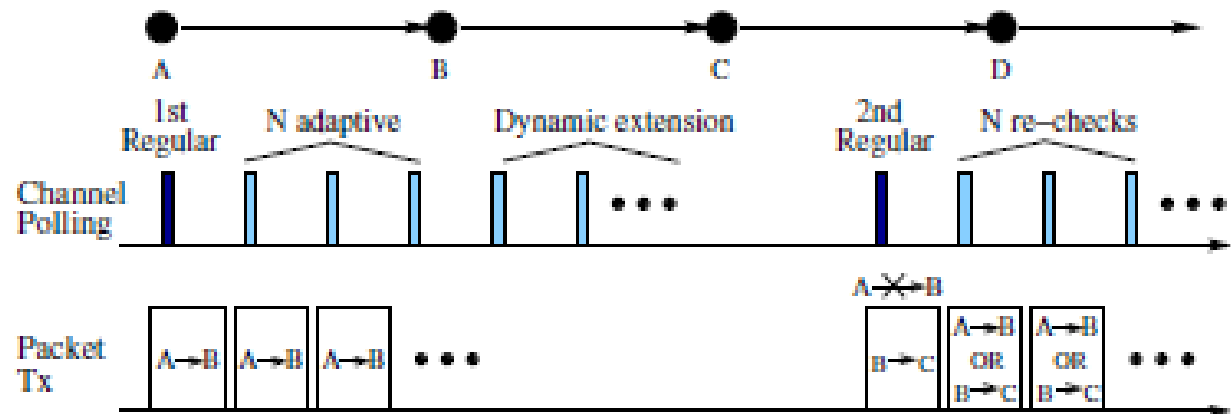
Design of SCP-MAC (Cont'd)

▶ Adaptive Channel Polling and Multi-hop Streaming

- A large set of applications mix periodic and bursty traffic or consist of unpredictable traffic mixes.
- At times of heavy traffic, each hop in a scheduled MAC potentially adds additional latency and reduces throughput.

Design of SCP-MAC (Cont'd)

- ▶ Adaptive Channel Polling and Multi-hop Streaming
- ▶ To reduce multi-hop latency:
 - Detect bursty traffic and dynamically add additional, high-frequency polling slots to nodes on the path.



Design of SCP-MAC (Cont'd)

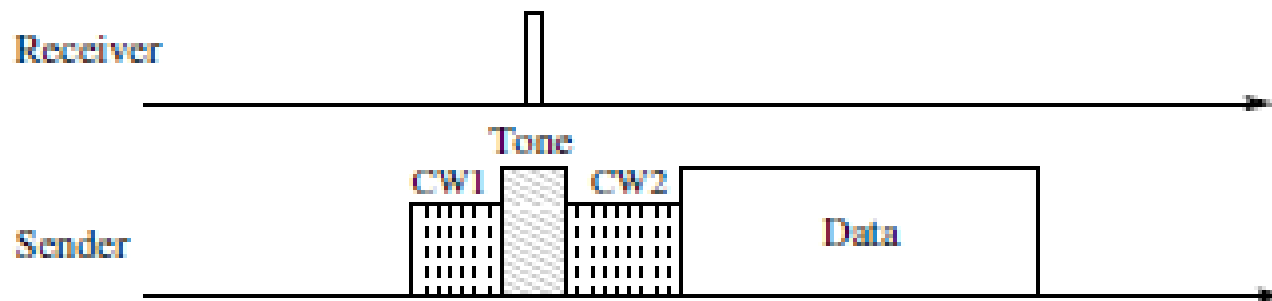
▶ Adaptive Channel Polling and Multi-hop Streaming

- Unlike prior approaches, this approach works over every hop on the path.
- Unlike B-MAC optimizations and fast-path, this approach requires no explicit signaling.
- Adaptive polling slots are dynamically added and extended as driven by traffic.
- Regular polling slots are always reserved for new nodes to enter the high-rate polling and transmission quickly.

Design of SCP-MAC (Cont'd)

▶ Other Optimizations

- *Two-Phase Contention:*
 - Sender transmits a short wakeup tone timed to intersect with the receiver's channel polling.
 - After waking up the receiver, the sender transmits the actual data packet.



Design of SCP-MAC (Cont'd)

- **Other Optimizations**
- *Two-Phase Contention:*
 - The two-phase contention lowers the collision probability compared to a single contention period of equal duration.
 - Splitting the contention with fewer slots so that SCP tolerates collisions on tone transmissions.
The wakeup tone only indicates network activity, not actual data.

Design of SCP-MAC (Cont'd)

▶ Other Optimizations

- *Overhearing Avoidance Based on Headers:*
 - Unnecessary overhearing can be a large energy cost in high-density networks.
 - SCP-MAC performs overhearing avoidance from MAC headers alone.

Outline

- ▶ Introduction.
- ▶ Design of SCP-MAC.
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ Protocol Implementation.
- ▶ Experimental Evaluation.
- ▶ Related Work.

Lower Bound of Energy Performance with Periodic Traffic

▶ Models and Metrics:

- Analysis focuses on the energy consumption by the radio, and does not model other components, such as the CPU or sensors.
- Four stable radio states: transmitting, receiving, listening, and sleeping. P_{tx} , P_{rx} , P_{listen} and P_{sleep}
- Expected energy consumption, per node:

$$\begin{aligned} E &= E_{cs} + E_{tx} + E_{rx} + E_{poll} + E_{sleep} \\ &= P_{listen}t_{cs} + P_{tx}t_{tx} + P_{rx}t_{rx} + P_{poll}t_{poll} + P_{sleep}t_{sleep} \end{aligned}$$

Models and Metrics (Cont'd)

Symbol	Meaning	CC1000	CC2420
P_{tx}	Power in transmitting	31.2mW	52.2mW
P_{rx}	Power in receiving	22.2mW	56.4mW
P_{listen}	Power in listening	22.2mW	56.4mW
P_{sleep}	Power in sleeping	3 μ W	3 μ W
P_{poll}	Power in channel polling	7.4mW	12.3mW
t_{p1}	Avg. time to poll channel	3ms	2.5ms
t_{cs1}	Avg. carrier sense time	7ms	2ms
t_B	Time to Tx/Rx a byte	416 μ s	32 μ s
T_p	Channel polling period	Varying	Varying
T_{data}	Data packet period	Varying	Varying
r_{data}	Data packet rate ($1/T_{data}$)	Varying	Varying
L_{data}	Data packet length	50B	50B
n	Number of neighbors	10	10

Symbols used in radio energy analysis, and typical values for the Mica2 radio (CC1000) and an 802.15.4 radio (CC2420)

Models and Metrics (Cont'd)

▶ Asynchronous Channel Polling: LPL

- Energy consumption with asynchronous channel polling:

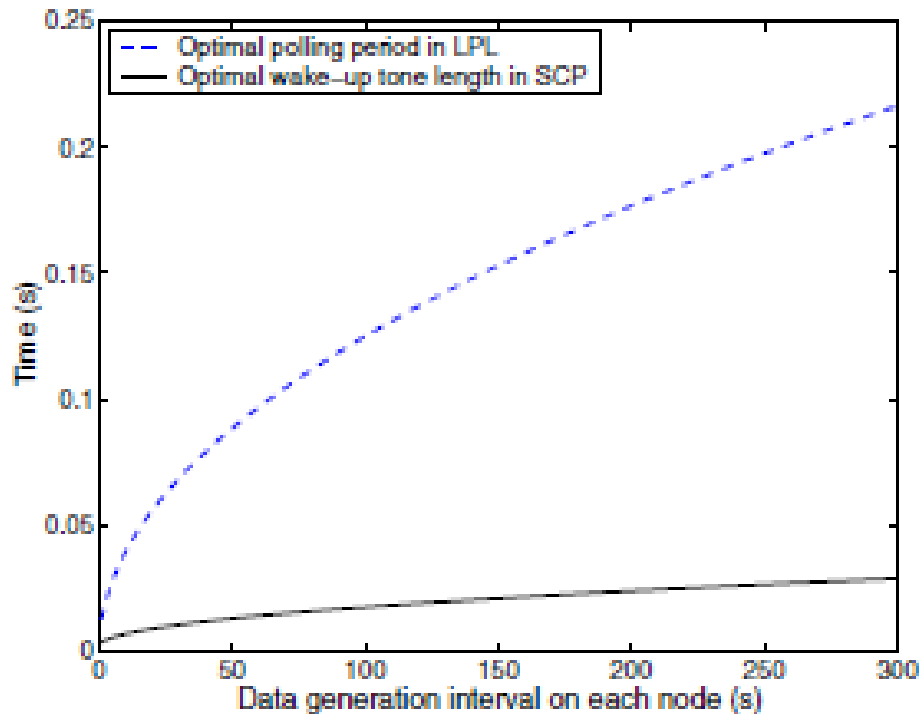
$$E_r = (P_{listen}t_{cs1} + P_{tx}(T_p + t_{pkt}) + nP_{rx}(T_p/2 + t_{pkt}))r_{data} + P_{poll}t_{p1}/T_p + P_{sleep}(1 - (t_{cs1} + (n/2 + 1)T_p + (n + 1)t_{pkt})r_{data} - t_{p1}/T_p)$$

- The optimal preamble length T_p for LPL:

$$T_{p,r}^* = \sqrt{\frac{(P_{poll} - P_{sleep})t_{p1}}{r_{data}(P_{tx} + nP_{rx}/2 - (n/2 + 1)P_{sleep})}}$$

Models and Metrics (Cont'd)

▶ Asynchronous Channel Polling: LPL



Optimal channel polling period in LPL (dotted), and wakeup-tone length in SCP (solid), given neighborhood size of 10

Models and Metrics (Cont'd)

- ▶ Scheduled Channel Polling: SCP

Symbol	Meaning	Value
T_{sync}	SYNC packet period	Varying
r_{sync}	SYNC packet rate ($1/T_{sync}$)	Varying
L_{sync}	SYNC packet length	18B
L_{sB}	SYNC bytes piggybacked to data	2B
t_{mtone}	Minimum duration of wake-up tone	2ms

Additional parameters in SCP-MAC

Models and Metrics (Cont'd)

- ▶ **Scheduled Channel Polling: SCP**
 - *Best Case: Perfect Piggybacking:*
 - Energy consumption of the scheduled channel polling with piggybacked synchronization:

$$\begin{aligned} E_{sp} = & P_{listen} t_{cs} I r_{data} \\ & + (P_{tx} + nP_{rx})(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\ & + P_{poll} t_{p1} / T_p \\ & + P_{sleep} [1 - t_{cs} I r_{data} \\ & \quad - (n + 1)(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\ & \quad - t_{p1} / T_p] \end{aligned} \quad (C)$$

Models and Metrics (Cont'd)

- ▶ **Scheduled Channel Polling: SCP**
 - *Best Case: Perfect Piggybacking:*
 - optimal polling period T_p for scheduled polling:

$$T_{p,sp}^* = \frac{1}{n(r_{data})}$$

Models and Metrics (Cont'd)

▶ Scheduled Channel Polling: SCP

- *Worst Case: All Explicit Synchronization*
- Energy consumption in scheduled channel polling with independent SYNC packets:

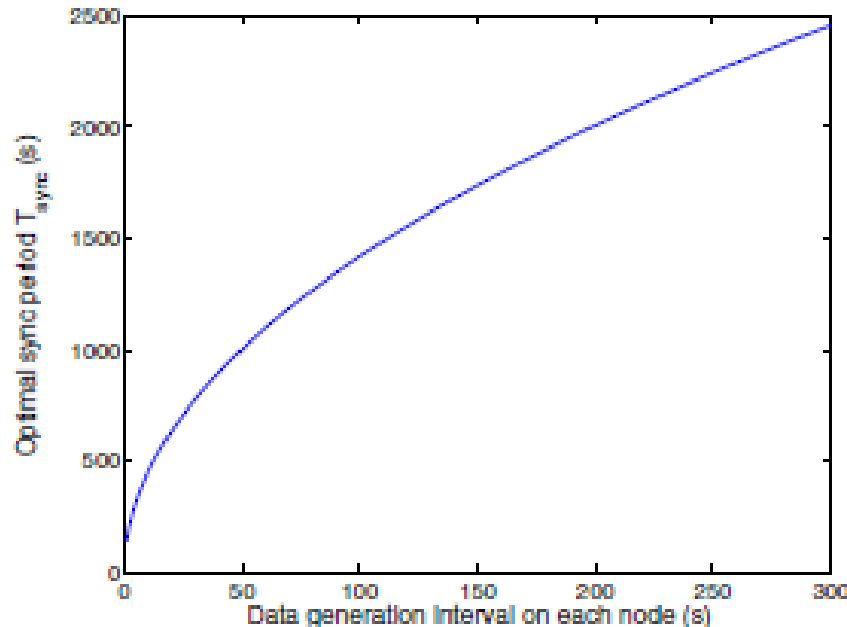
$$\begin{aligned} E_{snp} = & P_{listen} t_{cs1} (r_{data} + r_{sync}) \\ & + (P_{tx} + nP_{rx}) (t_{tone} + L_{data} t_B) r_{data} \\ & + (P_{tx} + nP_{rx}) (t_{tone} + L_{sync} t_B) r_{sync} \end{aligned}$$

- Optimal polling period for scheduled polling with independent SYNC packets:

$$T_{p,snp}^* = \frac{1}{n(r_{data} + r_{sync})}$$

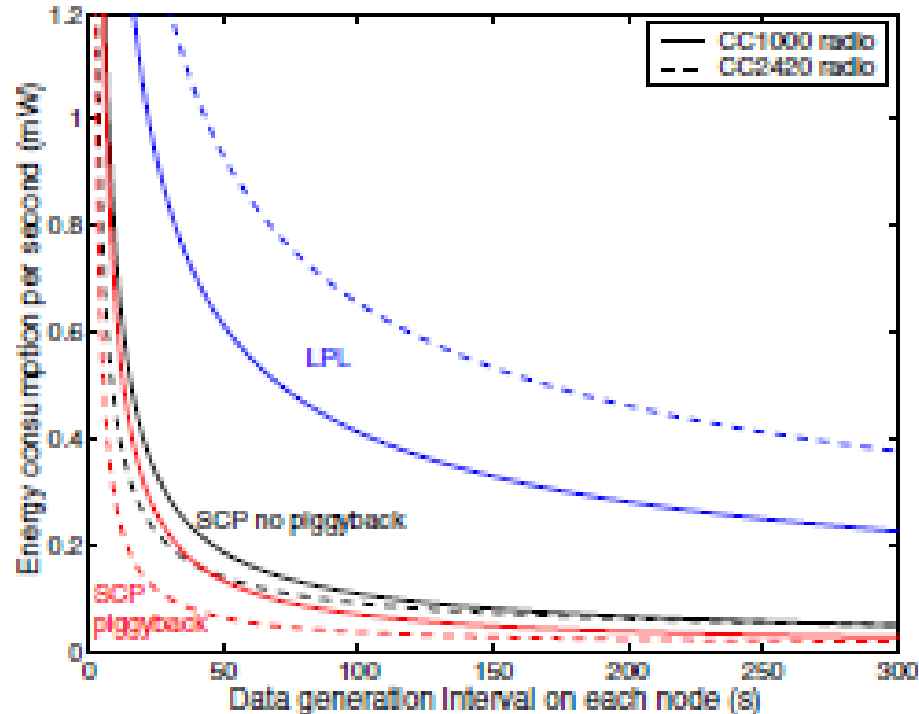
Models and Metrics (Cont'd)

- ▶ Scheduled Channel Polling: SCP
 - *Worst Case: All Explicit Synchronization*



Optimal SYNC period for SCP-MAC

Models and Metrics (Cont'd)



Analysis of optimal energy consumption for LPL and SCP with and without piggyback for CC1000 (solid lines) and CC2420 (dashed)

Outline

- ▶ Introduction.
- ▶ Design of SCP–MAC.
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ **Protocol Implementation.**
- ▶ Experimental Evaluation.
- ▶ Related Work.

Protocol Implementation

- ▶ Implementing SCP-MAC in TinyOS over the Mica2 motes with the CC1000 radio.
- ▶ Describe the preliminary port to MicaZ motes with the CC2420 radio supporting IEEE802.15.4.

Protocol Implementation (Cont'd)

Software Architecture

Implementation breaks MAC functionality into

- ▶ Four layers (separate TinyOS components):
 - Physical layer (PHY)
 - Basic CSMA layer,
 - LPL layer
 - SCP layer.
- ▶ several parameters and options:
 - RTS/CTS handling
 - Overhearing avoidance
 - Adaptive channel polling.

Protocol Implementation (Cont'd)

Physical layer (PHY)

- ▶ The bottom of the stack.
- ▶ Handles the radio states (sending, listening, receiving, sleeping, and warming up).
- ▶ Sends byte-by-byte with the Mica2
- ▶ Sends packet-by-packet with the MicaZ
- ▶ Implements and exports:
 - Interfaces for physical carrier sense
 - Transmission of the wakeup tone
 - CRC check
 - Time-stamping on transmitting and receiving of packets (to support time synchronization).

Protocol Implementation (Cont'd)

Basic CSMA Layer

- ▶ Above the PHY.
- ▶ Provides a common service to both LPL and SCP.
- ▶ Includes preamble length as a parameter to packet transmission.
- ▶ Responsible for performing carrier sense and random backoff.
- ▶ Supports full RTS/CTS/DATA/ACK or simply DATA/ACK.

Protocol Implementation (Cont'd)

LPL Layer

- ▶ Implemented on top of the CSMA.
- ▶ Periodically poll the channel and send the radio to sleep when there is no activity.
- ▶ Adjusts preamble lengths on transmitted packets to ensure they intersect with polling frequency.

Protocol Implementation (Cont'd)

SCP Layer

- ▶ Implemented above the LPL.
- ▶ Uses basic LPL to bootstrap schedules with SYNC packets.
- ▶ Coordinates packet transmission timing.
- ▶ Implements the randomized contention window before wake-up tone transmission.

Protocol Implementation (Cont'd)

Interaction with TinyOS

- ▶ Implementing a new timer in TinyOS to add support for dynamically adjusting timer values and asynchronous, low-jitter triggers.
- ▶ Timer implementation is based on the 8-bit hardware counter on Mica2.
- ▶ Runs independently from the CPU, allowing the CPU to sleep when no other activity is present.

Each timer event is about 0.4% the cost of a channel poll

Protocol Implementation (Cont'd)

Port to IEEE 802.15.4 Radio

- ▶ SCP-MAC to run on the 802.15.4 radios found on the MicaZ hardware.
- ▶ Challenges:
 - CC2420 is a packet-level radio, and the microcontroller cannot get bytelevel access.
 - Potentially affects the accuracy of time synchronization.
 - CC2420 limits the preamble length to 16 bytes with a default length of 4 bytes.

Protocol Implementation (Cont'd)

Port to IEEE 802.15.4 Radio

To implement long preambles:

- ▶ Sequentially send multiple wakeup packets back to back.
- ▶ Ensure that a receiver does not miss the “preamble” even if its channel polling time falls in a gap between the wakeup packets.

To reduce these gaps: pre-load the wakeup packet into the radio buffer before carrier sense, then resend the same packet from the buffer multiple times to make up a long preamble

Outline

- ▶ Introduction.
- ▶ Design of SCP–MAC.
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ Protocol Implementation.
- ▶ **Experimental Evaluation.**
- ▶ Related Work.

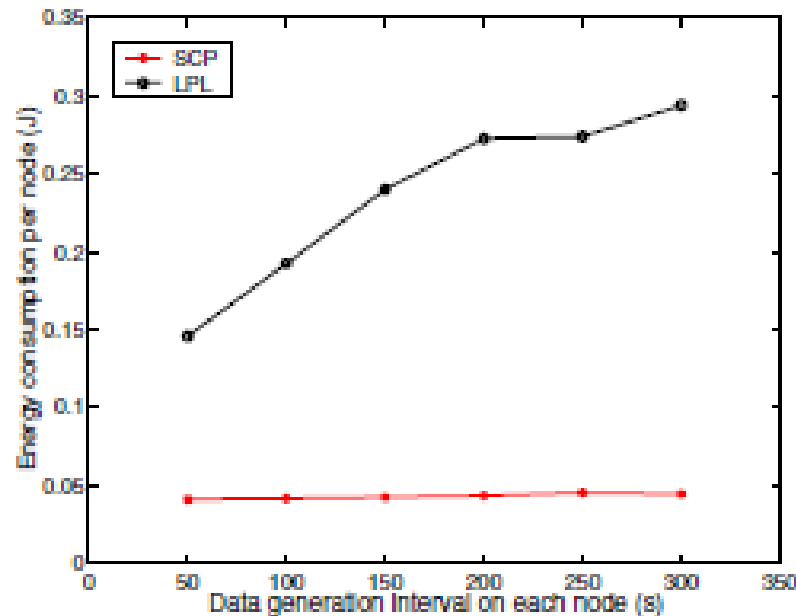
Experimental Evaluation

Optimal Setup with Periodic Traffic

- ▶ Comparing the energy performance of SCP and LPL under optimal configuration with completely periodic.
- ▶ MAC parameters vary based on network size and data rate.
- ▶ Placing 10 nodes in a single hop network.
- ▶ Each node periodically generates a 40B message (not including preamble).
- ▶ Each node's message generation interval from 50–300s.
- ▶ Run each experiment for 5 message periods, generating 50 total messages over each experiment.

Experimental Evaluation (Cont'd)

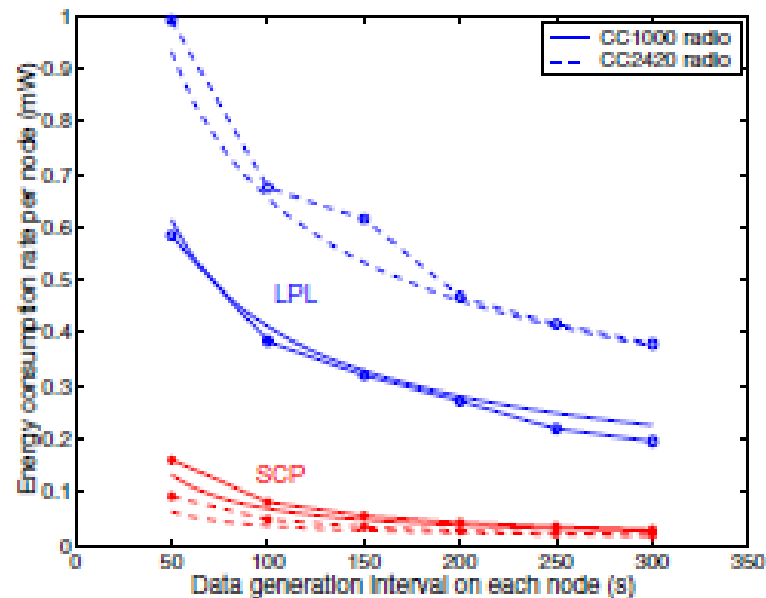
Optimal Setup with Periodic Traffic



Mean energy consumption (J) for each node as traffic rate varies (assuming optimal configuration and periodic traffic)

Experimental Evaluation (Cont'd)

Optimal Setup with Periodic Traffic



Mean energy consumption rate (J/s or W) for each node as traffic rate varies. The radios are the CC1000 (solid lines) and CC2420 (dashed)

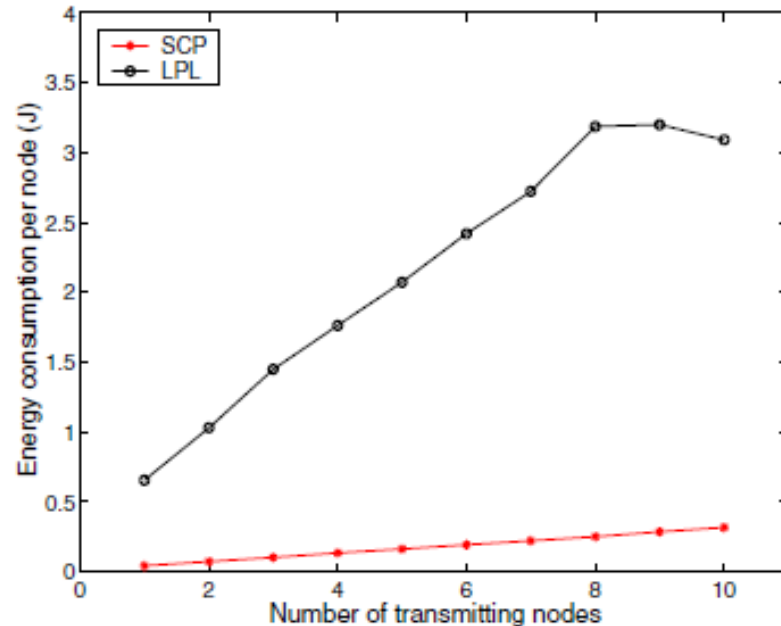
Experimental Evaluation (Cont'd)

Performance with Unanticipated Traffic

- ▶ In many applications the traffic load is less predictable (fire detection in forests).
- ▶ Tuning LPL and SCP for a 0.3% duty cycle, polling every second.
- ▶ All other parameters match the prior experiment.
- ▶ Each node generates 20 100B long messages.

Experimental Evaluation (Cont'd)

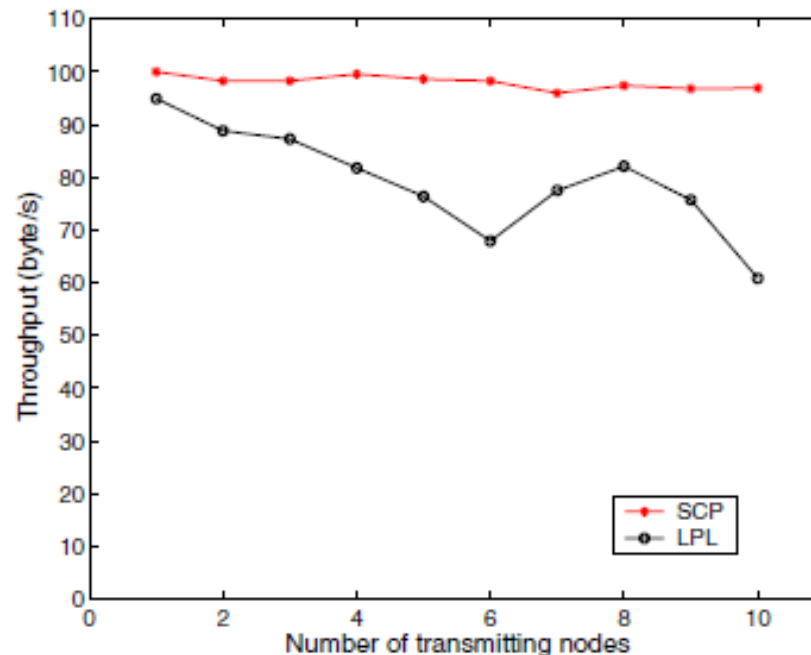
Performance with Unanticipated Traffic



Energy consumptions on heavy traffic load with very low duty cycle configurations

Experimental Evaluation (Cont'd)

Performance with Unanticipated Traffic (two-phase contention)



Throughput on heavy traffic load with very low duty cycle configurations

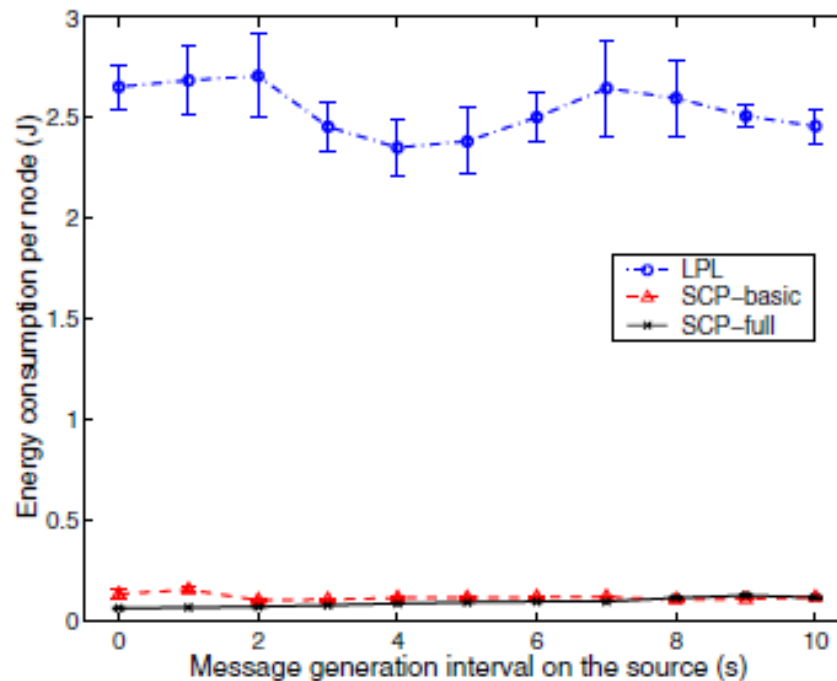
Experimental Evaluation (Cont'd)

Performance in a Multi-hop Network

- ▶ 9-hop linear network with 10 nodes.
- ▶ Adaptive channel polling is designed to reduce latency.
- ▶ All packets are sent as unicast without RTS/CTS.
- ▶ Acknowledgments with up to three retries.

Experimental Evaluation (Cont'd)

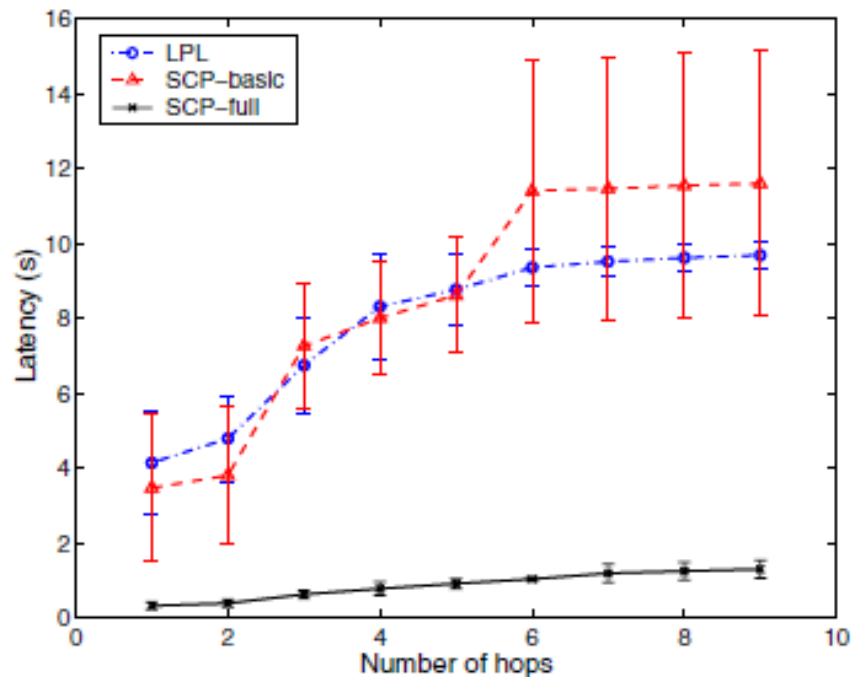
Performance in a Multi-hop Network



Mean energy consumption per node for multihop experiments (20 packets over 9 hops)

Experimental Evaluation (Cont'd)

Performance in a Multi-hop Network



Mean packet latency over 9 hops at the heaviest load

Outline

- ▶ Introduction.
- ▶ Design of SCP-MAC.
- ▶ Lower Bound of Energy Performance with Periodic Traffic.
- ▶ Protocol Implementation.
- ▶ Experimental Evaluation.
- ▶ **Related Work.**

Related Work

- ▶ Power-save mode in IEEE 802.11 synchronizes wakeup times of nodes in a single-hop network.
- ▶ S-MAC developed a fully distributed algorithm to synchronize the wakeup schedules of nodes in a multi-hop network.
- ▶ T-MAC improves S-MAC by reducing the wakeup duration controlled by an adaptive timer.
- ▶ WiseMAC can reduce the preamble length after an initial unicast packet with a long preamble.

Related Work (Cont'd)

TDMA, second class of MAC protocols.

- ▶ LEACH and BMA.
- ▶ LMAC and TRAMA.
- ▶ ZMAC, proposed a hybrid protocol to combine TDMA with CSMA.

Thanks !