Ultra-Low Duty Cycle MAC with Scheduled Channel Polling

Wei Ye and John Heidemann

CS577

Brett Levasseur

Outline

WPI

- Introduction
- Scheduled Channel Polling (SCP-MAC)
- Energy Performance Analysis
- Implementation
- Conclusions

Introduction

WPI

- Sensor networks need to save power
- Controlling the power and duty cycle is critical
- Synchronization techniques are power efficient but have complex management
- Contention based protocols used more often but must be kept at low duty cycles

Related Work

WPI

- Low Power Listening (LPL)
  - Low power probe to check channel activity
  - No long wake period
  - Requires transmission preamble
- Scheduled Protocols
  - Sleep/Wake schedules
  - Only transmit when receiver is listening
  - Requires coordination

Limitations

WPI

- Scheduling and LPL require 1 -2% duty cycle
- Scheduling has long wake time
- LPL has long transmit preamble
- Authors propose Schedule Channel Polling (SCP)
  - Ultra low duty cycles of 0.01 – 0.1%
  - Reduce energy consumption by factor of 10-100

Outline

WPI

- Introduction
- Scheduled Channel Polling (SCP-MAC)
- Energy Performance Analysis
- Implementation
- Conclusions

SCP-MAC

WPI

- Combines short channel polling from LPL with scheduling
- Periodic channel polling (LPL)
- Polling time is synchronized across nodes
  - Reduced transmit preamble size
  - Requires less energy

Synchronized Channel Polling

WPI

- LPL requires long preamble
  - Preamble at least as long as channel polling period
- SCP synchronizes polling time
  - Only short wake up tone is required
  - Requires synchronization

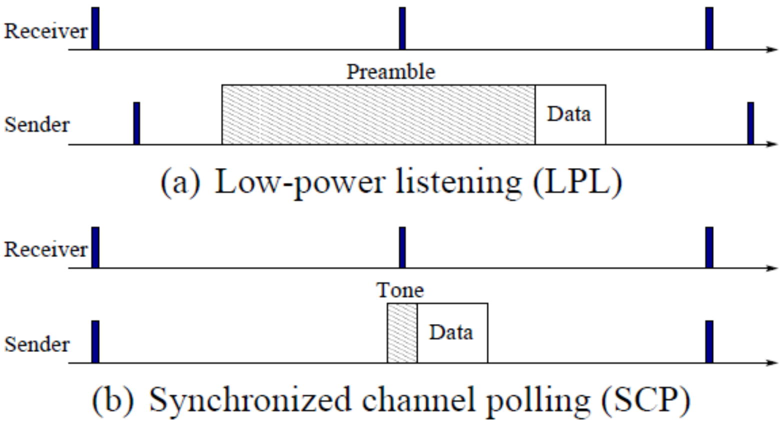


Figure 1. Sender and receiver synchronization schemes.

Bursty Traffic

WPI

- Running SCP-MAC at low duty cycle adds latency during heavy traffic periods
  - Low duty cycle means more time between transmission opportunities
- Detect bursty traffic and add polling slots
  - The new slots allows for more transmissions in less time

Adaptive Polling

WPI

- Node B adds n polling slots when it receives from A
- Node A can give up transmitting to B so B can transmit to C
  - This gets C to add its own n polling slots
- Should add one poll per node that needs to send

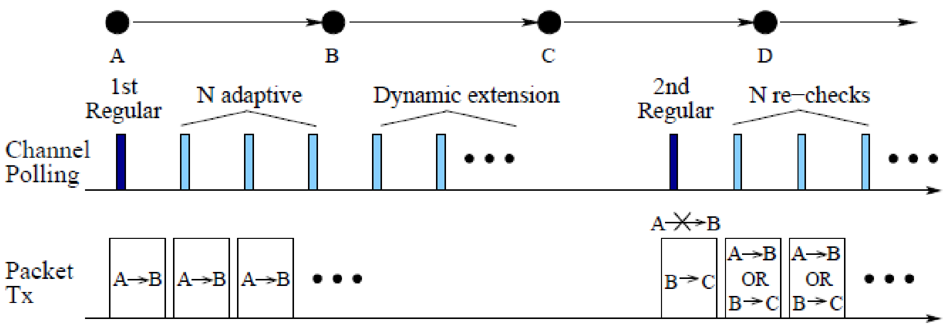


Figure 2. Adaptive channel polling and multi-hop streaming.

Two-Phase Contention

WPI

- Carrier sense in CW1 before sending tone
- If channel idle send wakeup tone
- If tone sent successfully node performs carrier sense in CW2
- If channel idle then send data

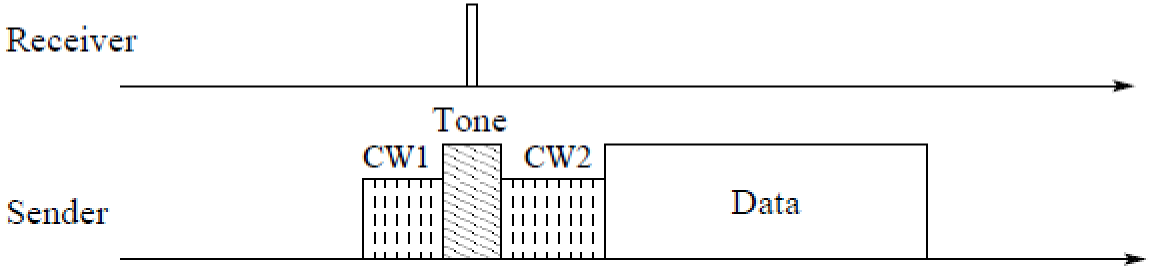


Figure 3. Two-phase contention in SCP-MAC.

Overhearing Avoidance

WPI

- Hearing a packet meant for another node
  - Causes overhearing node to waste power
- Stop listening to packets not meant for the node
  - With RTS/CTS nodes can see when the channel is busy
  - Without RTS/CTS nodes examine MAC headers and goes back to sleep if receiving address is not for them

Outline

WPI

- Introduction
- Scheduled Channel Polling (SCP-MAC)
- Energy Performance Analysis
- Implementation
- Conclusions

Energy Performance Analysis

WPI

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	$3\mu W$	$3\mu W$
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	$416\mu s$	$32\mu s$
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

Table 1. Symbols used in radio energy analysis, and typical values for the Mica2 radio (CC1000) and an 802.15.4 radio (CC2420)

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

Table 2. Additional parameters in SCP-MAC

Energy Consumption

WPI

- Sum of energy used for each state
  - Carrier Sense
  - Transmit
  - Receive
  - Poll
  - Sleep

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

Asynchronous Channel Polling

WPI

- What is the energy cost to poll using asynchronous channel polling in LPL?
  - Length of preamble
  - Time for carrier sense
  - Time for polling
  - Time spent sleeping
  - Transmission/Reception rate

LPL Preamble

WPI

- Preamble must be as long as the polling interval T_p

$$L_{preamble} = T_p / t_B$$

- t_b is the time to transmit a byte
- $L_{preamble}$ is the length of the preamble in bytes

LPL Carrier Sense

WPI

- Nodes perform carrier sense before preamble

$$t_{cs} = t_{cs1} / T_{data} = t_{cs1} r_{data}$$

- t_{cs1} is the average carrier sense time
- r_{data} is the rate of sending data

Time Transmitting/Receiving

WPI

- Transmit is sending preamble and data

$$
\begin{aligned}
t_{tx} &= (L_{preamble} + L_{data}) t_B r_{data} \\
&= (T_p + L_{data} t_B) r_{data}
\end{aligned}
$$

- Receive is sum across all nodes
  - n is the number of nodes

$$t_{poll} = t_{p1} / T_p$$

Polling and Sleeping

WPI

$$t_{poll} = t_{p1} / T_p$$

- Normalized time for polling and sleeping

$$t_{sleep} = 1 - t_{cs} - t_{tx} - t_{rx} - t_{poll}$$

- The node is asleep when not in carrier sense, transmission, reception or polling

Random Channel Polling LPL

WPI

- Power consumed determined by
  - Neighborhood size
  - Data rate
  - Channel polling
- Small T_p reduces cost of polling but increases transmit and reception cost

$$
\begin{aligned}
E_r \quad = \quad & P_{listen} t_{cs1} r_{data} + (P_{tx} + n P_{rx})(T_p + L_{data} t_B) r_{data} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep}(1 - t_{cs1} r_{data} - (n+1)(T_p + L_{data} t_B) r_{data} \\
& - t_{p1} / T_p)
\end{aligned}
$$

Optimize Polling Time LPL

WPI

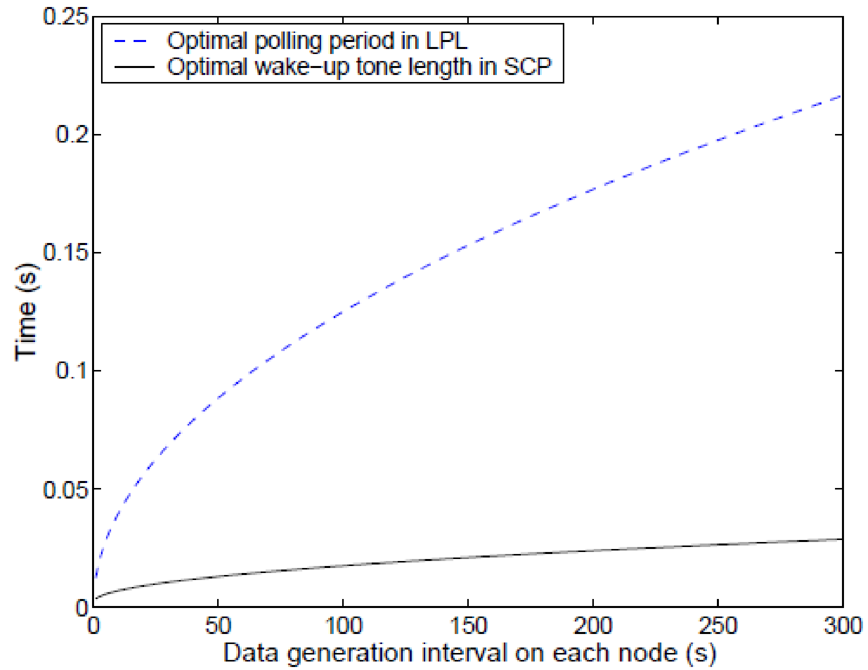


Figure 4. Optimal channel polling period in LPL (dotted), and wakeup-tone length in SCP (solid), given neighborhood size of 10.

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

Synchronization

WPI

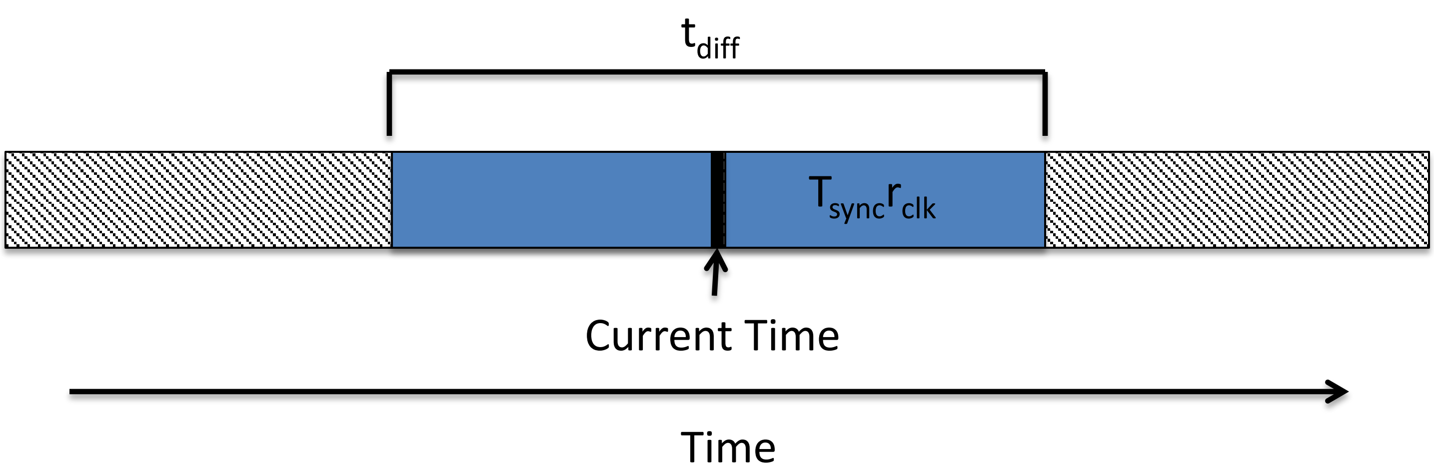
- Nodes broadcast scheduling information
  - Occurs every synchronization period
  - Required every 10 – 60 minutes
- Piggyback synchronization with data when possible
- Clock drift requires guard time

Clock Drift

WPI

- T_{sync} – Synchronization period
- r_{clk} – Clock drift rate

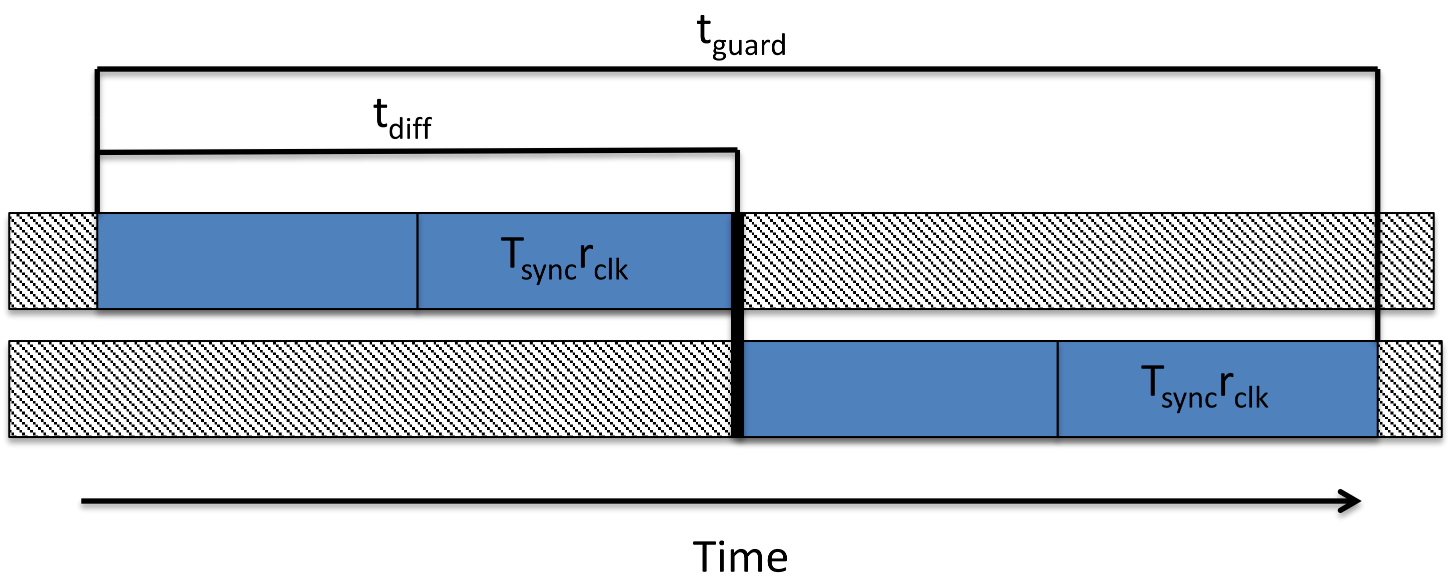
$$t_{diff} = 2T_{sync}r_{clk}$$



Guard Time 2 Nodes

WPI

- Difference between 2 nodes requires $2t_{diff}$

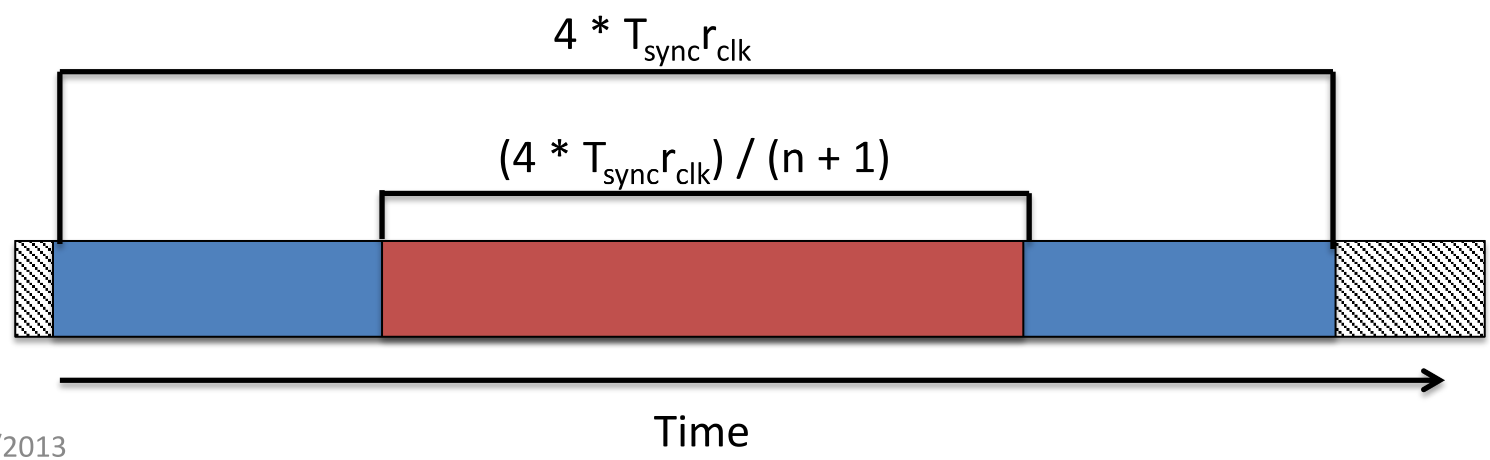


Guard Time n + 1 Nodes

WPI

- Every node sends a SYNC in each T_{sync} period
- For n neighbors this reduces clock drift (n+1) times

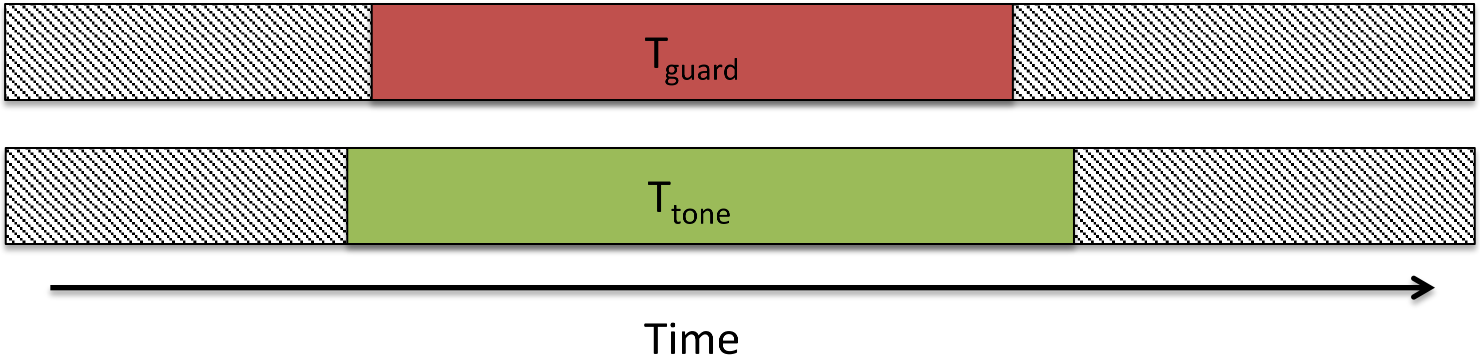
$$t_{guard} = 4T_{sync}r_{clk}/(n+1)$$



Wake-up Tone

- Guard time plus a short fixed time
- t_{mtone} is time needed to detect tone

$$t_{tone} = 4T_{sync}r_{clk}/(n+1) + t_{mtone}$$



SCP with Piggybacking

WPI

- Perfect piggyback means all synchronization goes out with application data

$$
\begin{aligned}
E_{sp} \;=\; & P_{listen} t_{cs1} r_{data} && \leftarrow \text{Carrier Sense} \\
\text{Tx / Rx} \rightarrow & + (P_{tx} + nP_{rx})(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
\text{Polling} \rightarrow & + P_{poll} t_{p1} / T_p \\
\text{Sleep} \rightarrow & + P_{sleep} [1 - t_{cs1} r_{data} \\
& \quad - (n+1)(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
& \quad - t_{p1} / T_p]
\end{aligned}
$$

SCP without Piggybacking

WPI

- More time needed to transmit and receive synchronization packets

$$
\begin{aligned}
E_{snp} \quad = \quad & P_{listen} t_{cs1} (r_{data} + r_{sync}) \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{data} t_B) r_{data} \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{sync} t_B) r_{sync} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep} [1 - t_{cs1} (r_{data} + r_{sync}) \\
& \qquad - \ (n+1)(t_{tone} + L_{data} t_B) r_{data} \\
& \qquad - \ (n+1)(t_{tone} + L_{sync} t_B) r_{sync} \\
& \qquad - \ t_{p1} / T_p]
\end{aligned}
$$

Poll Time (T_p) in LPL and SCP

WPI

$$
\begin{aligned}
E_r \;=\;& P_{listen} t_{cs1} r_{data} + (P_{tx} + nP_{rx})(T_p + L_{data} t_B) r_{data} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep}(1 - t_{cs1} r_{data} - (n+1)(T_p + L_{data} t_B) r_{data} \\
& - t_{p1} / T_p)
\end{aligned}
$$

LPL

Larger poll time adds to transmission and reception cost

$$
\begin{aligned}
E_{sp} \;=\;& P_{listen} t_{cs1} r_{data} \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep}[1 - t_{cs1} r_{data} \\
& \quad - (n+1)(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
& \quad - t_{p1} / T_p]
\end{aligned}
$$

SCP

Larger poll time does not add to transmission and reception cost

With/Without Piggybacking

WPI

$$
\begin{aligned}
E_{sp} = \; & P_{listen} t_{cs1} r_{data} \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep} [1 - t_{cs1} r_{data} \\
& \quad - (n+1)(t_{tone} + L_{sB} t_B + L_{data} t_B) r_{data} \\
& \quad - t_{p1} / T_p]
\end{aligned}
$$

$$
\begin{aligned}
E_{snp} = \; & P_{listen} t_{cs1} (r_{data} + r_{sync}) \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{data} t_B) r_{data} \\
& + (P_{tx} + nP_{rx})(t_{tone} + L_{sync} t_B) r_{sync} \\
& + P_{poll} t_{p1} / T_p \\
& + P_{sleep} [1 - t_{cs1} (r_{data} + r_{sync}) \\
& \quad - (n+1)(t_{tone} + L_{data} t_B) r_{data} \\
& \quad - (n+1)(t_{tone} + L_{sync} t_B) r_{sync} \\
& \quad - t_{p1} / T_p]
\end{aligned}
$$

- With piggybacking synchronization sent with data
- Without synchronization data sent on its own

Optimal T_{sync}

WPI

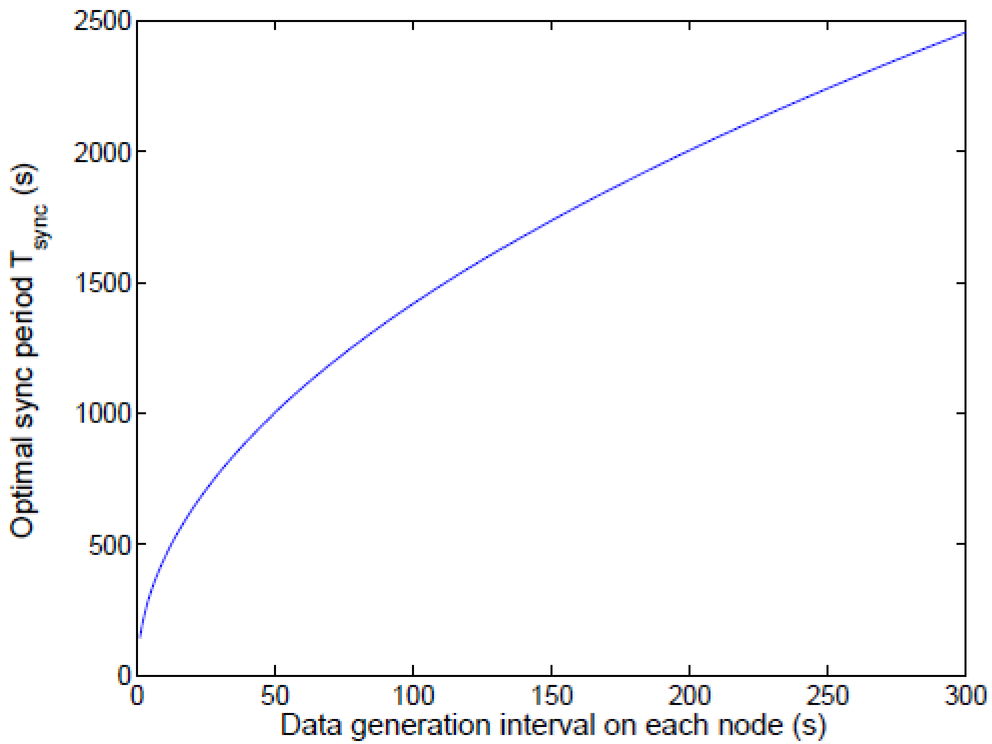


Figure 5. Optimal SYNC period for SCP-MAC.

$$T^*_{sync} = \sqrt{\frac{n(n+1)(E_l + P_t t_t + E_p)}{2 r_{data} r_{clk} P_t}}$$

Optimum Energy Consumption WPI

- LPL uses 3-6 times more energy than SCP
- Piggybacking reduces energy cost when data is rarely sent
- Benefits minimal when data sent frequently
- LPL worse on newer radio, SCP better

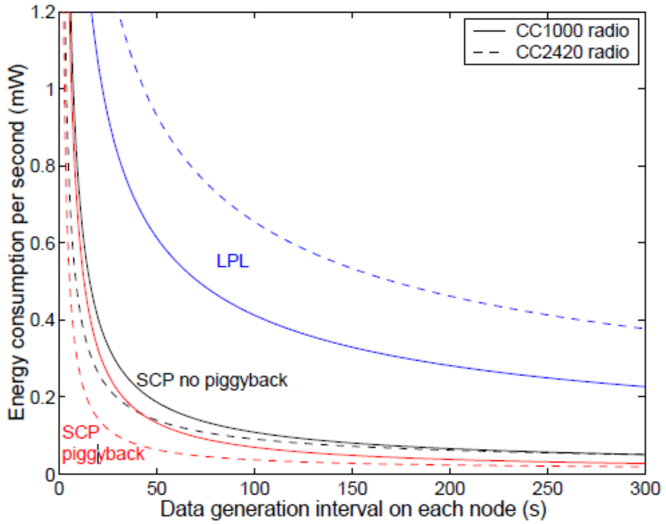


Figure 6. Analysis of optimal energy consumption for LPL and SCP with and without piggyback for CC1000 (solid lines) and CC2420 (dashed).

Outline

WPI

- Introduction
- Scheduled Channel Polling (SCP-MAC)
- Energy Performance Analysis
- Implementation
- Conclusions

Implementation

WPI

- TinyOS on Mica2 (CC1000) and MicaZ (CC2420) motes
- Layered approach
- LPL used for polling
- SCP used for scheduling
- TinyOS controls CPU power and timers

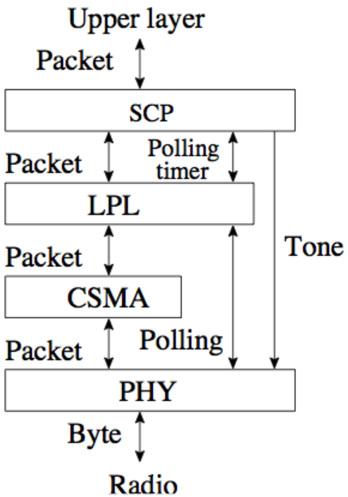


Fig. 7. Software architecture of the SCP-MAC implementation in TinyOS.

[Wei 05]

Power Consumption

WPI

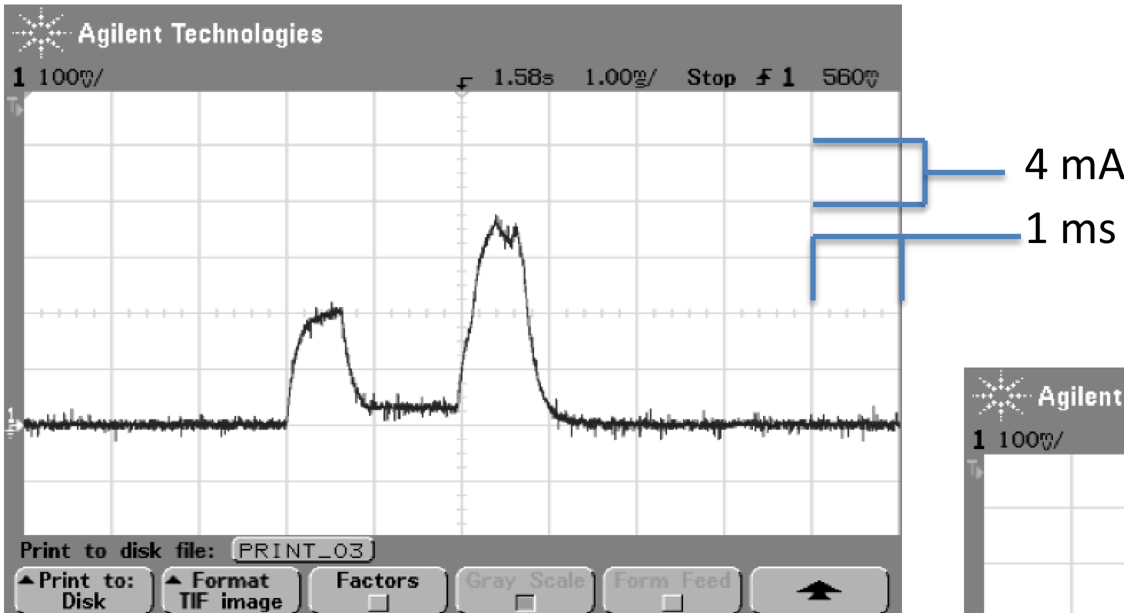


Fig. 8. Channel polling process implemented in SCP-MAC.

[Wei 05]

- Energy required to maintain timers is less than using the radio
- Adding timers for scheduling is a low energy impact

[Wei 05]

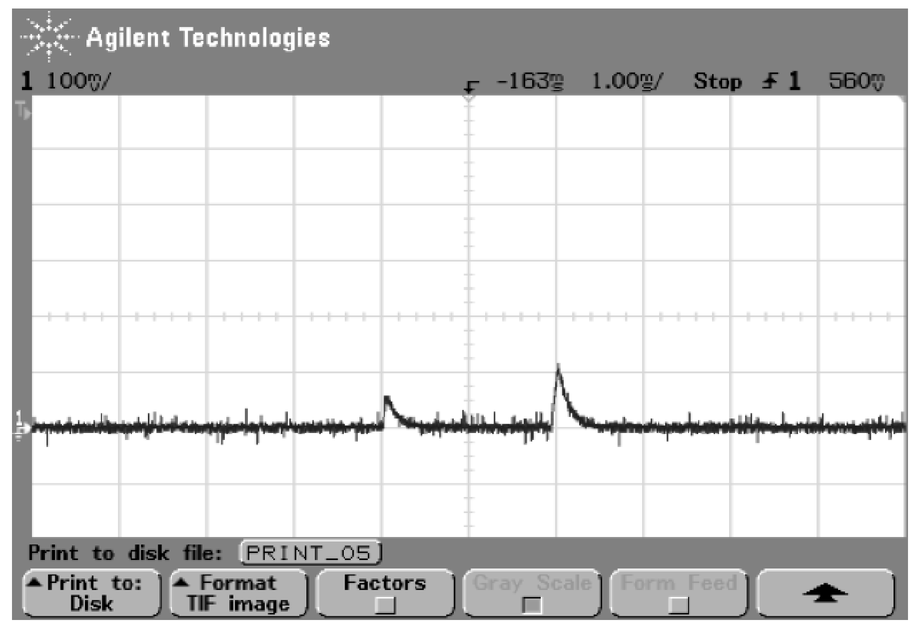


Fig. 9. CPU overhead on timer firing events.

Optimal Setup

WPI

- Periodic traffic
- 10 nodes all in range
- 40B data message
- 1 message every 5 – 30 seconds
- LPL requires 2 – 2.5 times more energy than SCP

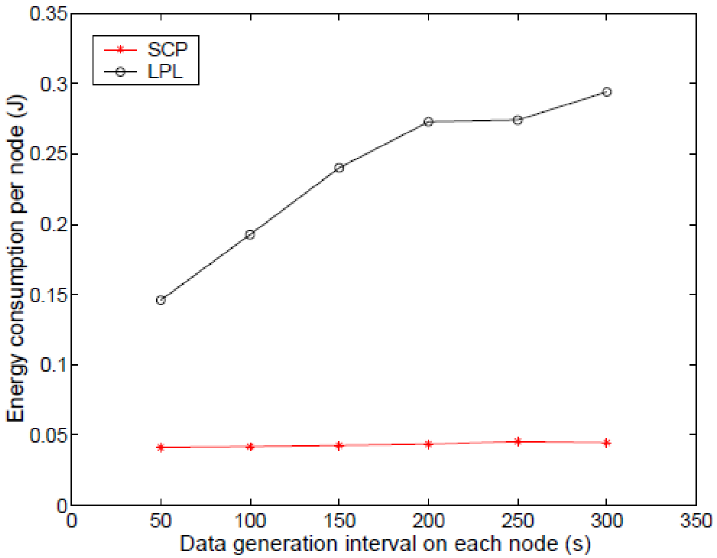


Figure 8. Mean energy consumption (J) for each node as traffic rate varies (assuming optimal configuration and periodic traffic).

Optimal Setup Cont

WPI

- LPL needs 3–6 times more power than SCF
  - Both optimized for periodic traffic
- Experimental results similar to analytical results

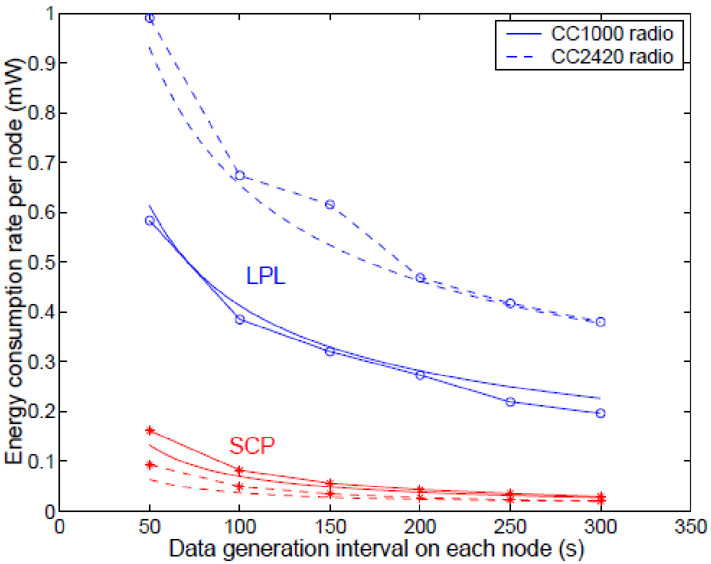


Figure 9. Mean energy consumption rate (J/s or W) for each node as traffic rate varies (assuming optimal configuration and periodic traffic). The radios are the CC1000 (solid lines) and CC2420 (dashed).

Experiment vs Analysis

WPI

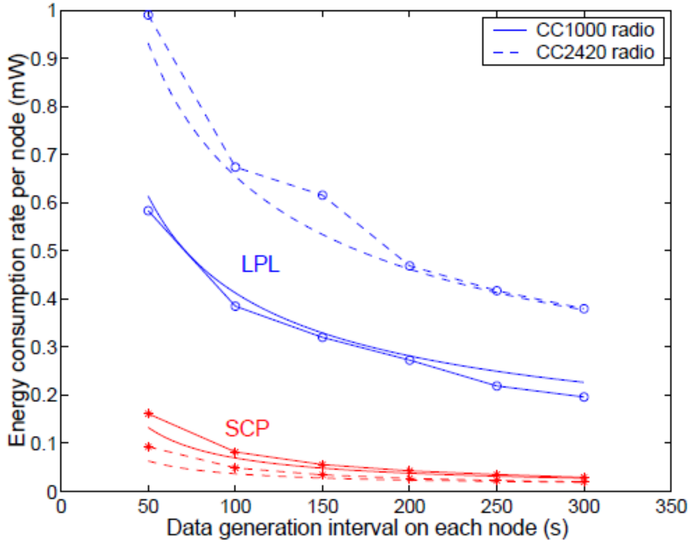


Figure 9. Mean energy consumption rate (J/s or W) for each node as traffic rate varies (assuming optimal configuration and periodic traffic). The radios are the CC1000 (solid lines) and CC2420 (dashed).

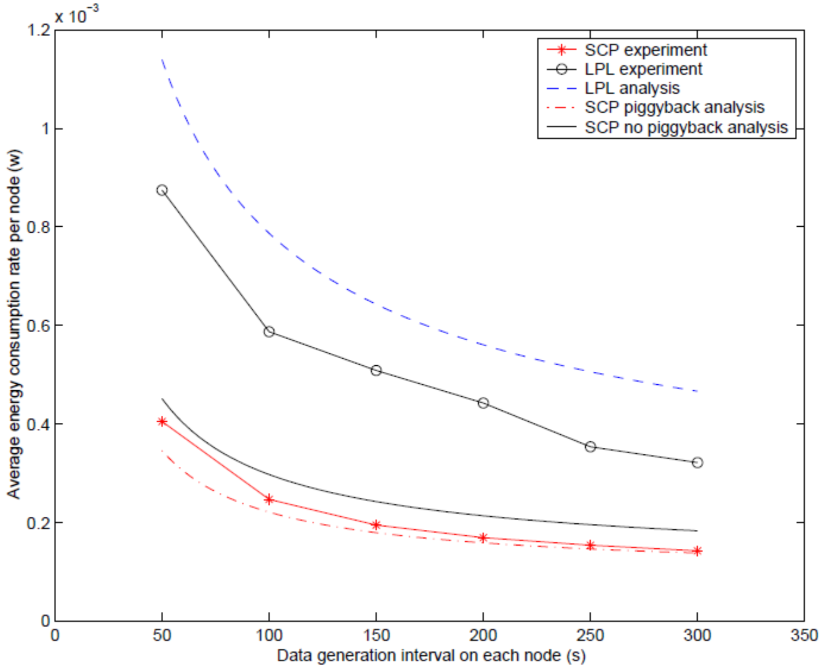


Fig. 11. Mean rate of energy consumption rate (W, or J/s) for each node as traffic send rate varies. (Assumes optimal LPL and SCP configurations, completely periodic traffic, and a 10-node network.)

[Wei 05]

Unanticipated Traffic

WPI

- Long down time and then large amount of traffic
- 0.3% duty cycle
- Poll every second
- Busy mode
  - 20, 100B long messages
- LPL uses 8 times more energy than SCP
  - Mostly preamble

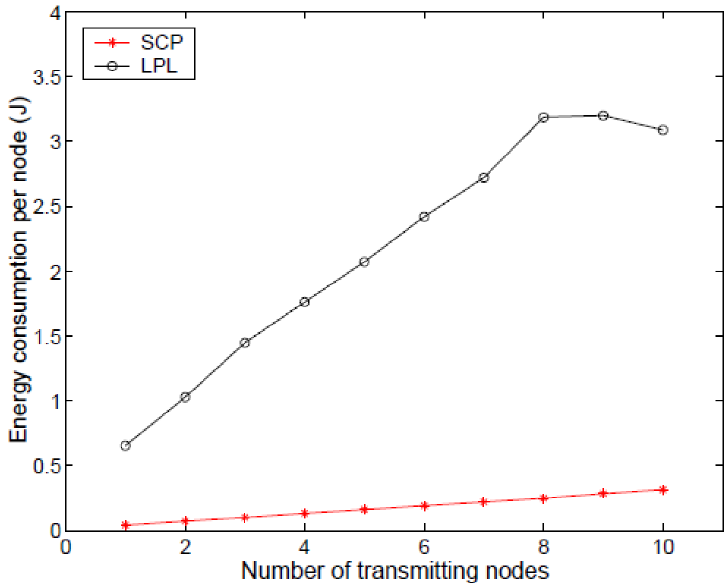


Figure 10. Energy consumptions on heavy traffic load with very low duty cycle configurations.

Unanticipated Traffic Cont

WPI

- Heavy traffic load leads to contention
- LPL has one congestion window
  - 32 slots, 10 nodes
  - About 1/3 chance of nodes conflicting
- SCP has two congestion windows
  - Collision rate about 4%

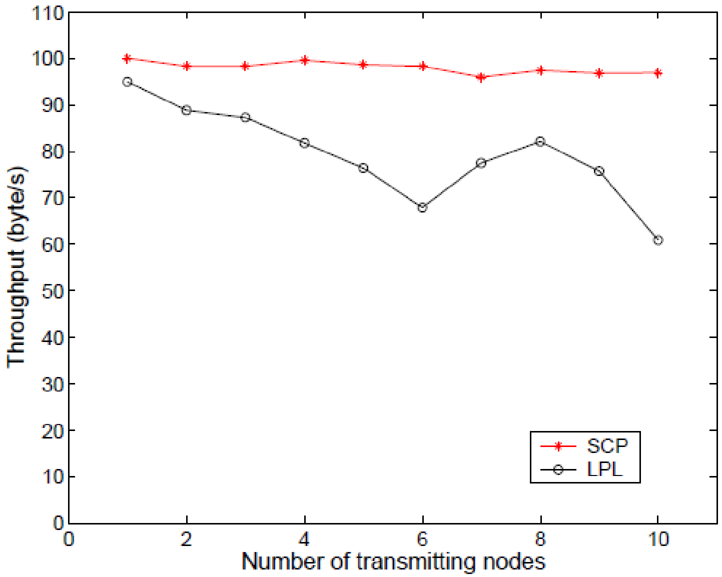


Figure 11. Throughput on heavy traffic load with very low duty cycle configurations.

Mean Energy

WPI

- LPL uses 20 – 40 times more energy than full (adaptive polling) SCP-MAC
  - Long preamble
  - Overhearing nodes
- False wakeups
  - LPL needs to receive full preamble

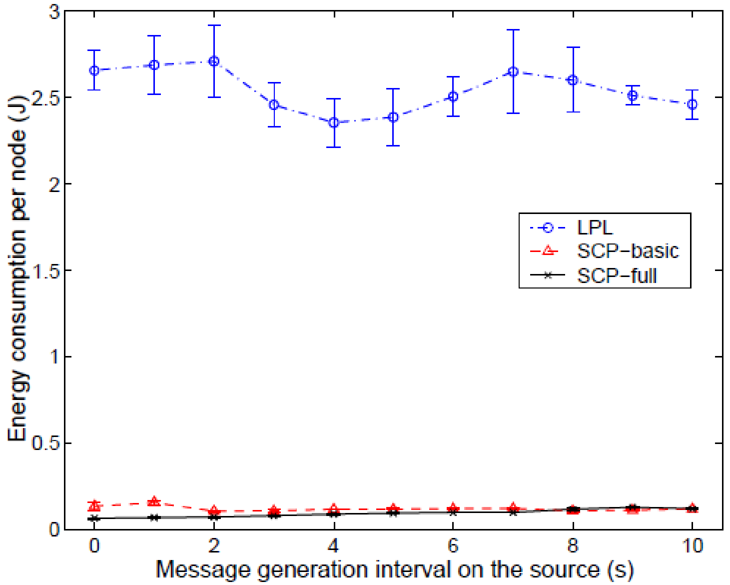


Figure 12. Mean energy consumption per node for multi-hop experiments (20 packets over 9 hops).

Mean Latency

WPI

- Basic SCP and LPL have similar latency
  - Polling interval latency
- Adaptive channel polling causes lower latency for SCP full
  - All nodes switch to higher duty cycle polling after first packet

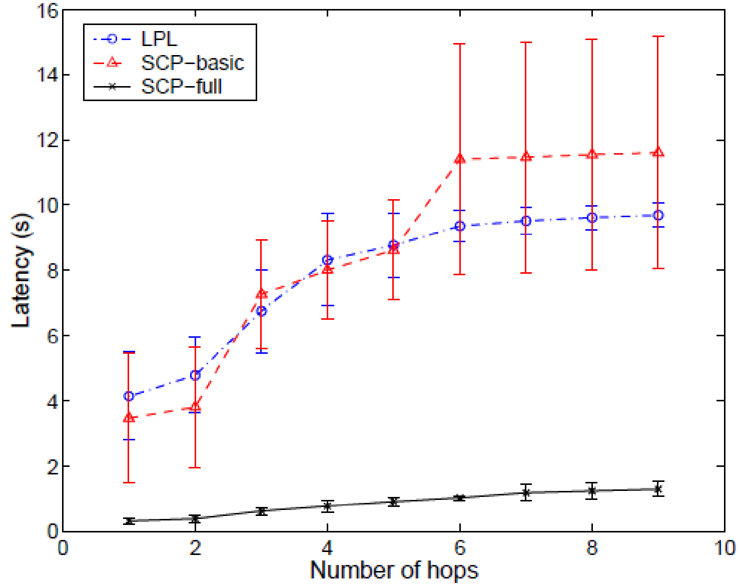


Figure 13. Mean packet latency over 9 hops at the heaviest load.

Conclusions

WPI

- Proposed SCP (LPL with scheduling)
- Found best operating points for LPL and SCP
- SCP showed less power usage than LPL
  - 3 – 6 times better under ideal scenario (periodic traffic)
- SCP has greater improvements when using newer radios

Questions

WPI

References

WPI

[Wei 06] - Wei Ye, Fabio Silva, and John Heidemann. 2006. Ultra-low duty cycle MAC with scheduled channel polling. In *Proceedings of the 4th international conference on Embedded networked sensor systems* (SenSys '06). ACM, New York, NY, USA, 321-334. DOI=10.1145/1182807.1182839 http://doi.acm.org/10.1145/1182807.1182839

[Wei 05] - Wei Ye, Fabio Silva, and John Heidemann. 2005. Ultra-low duty cycle MAC with scheduled channel polling. http://www.isi.edu/~johnh/PAPERS/Ye05a.pdf