The ContikiMAC Radio Duty Cycling Protocol

Adam Dunkels

Presenter - Lingling Sun
Outline

- Introduction
- ContikiMAC Mechanism
- Implementation of ContikiMAC
- Evaluates of Energy Efficiency
- Conclusion
Introduction

• **Contiki** is an open source operating system for networked, memory-constrained systems with a particular focus on low-power wireless Internet of things devices. It was created by Adam Dunkels in 2002.

• **Radio Duty Cycling (RDC)** mechanism specifies a predetermined method for communication between sleeping nodes. It allows nodes to sleep and periodically wake-up to check the medium activity.
Introduction

• ContikiMAC is a suitable and energy efficient RDC mechanism for sensor networks running Contiki.

• Contiki Structure

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>IETF CoAP / REST Engine</td>
</tr>
<tr>
<td>Transport</td>
<td>UDP</td>
</tr>
<tr>
<td>Network</td>
<td>IPv6 / RPL</td>
</tr>
<tr>
<td>Adaptation</td>
<td>6LoWPAN</td>
</tr>
<tr>
<td>MAC</td>
<td>CSMA / link-layer bursts</td>
</tr>
<tr>
<td>Radio Duty Cycling</td>
<td>ContikiMAC</td>
</tr>
<tr>
<td>Physical</td>
<td>IEEE 802.15.4</td>
</tr>
</tbody>
</table>
Introduction

- ContikiMAC has a power-efficient wake-up mechanism which is achieved by precise timing through a set of timing constraints.
- ContikiMAC uses a fast sleep optimization to allow receivers to quickly detect false-positive wake-ups.
- ContikiMAC uses a transmission phase-lock optimization to allow run-time optimization of the energy-efficiency of transmissions.
ContikiMAC Mechanism

Figure 1: ContikiMAC: nodes sleep most of the time and periodically wake up to check for radio activity. If a packet transmission is detected, the receiver stays awake to receive the next packet and sends a link layer acknowledgment. To send a packet, the sender repeatedly sends the same packet until a link layer acknowledgment is received.
ContikiMAC Mechanism

Figure 2: Broadcast transmissions are sent with repeated data packets for the full wake-up interval.
ContikiMAC Mechanism - Timing

**Figure 3.3:** ContikiMAC RDC mechanism.

- $\tau_{cycle}$: Wake up interval for sleeping nodes.
- $\tau_s$: Strobe transmitting time (dependent on frame length).
- $\tau_i$: Interval between strobes.
- $\tau_a$: Time needed before an acknowledgement to a sent frame can be detected by a sender.
- $\tau_r$: Time needed to perform a *Clear Channel Assessment* (CCA).
- $\tau_c$: Interval between CCA checks.

\[ \tau_a < \tau_i < \tau_c < \tau_c + 2\tau_r < \tau_s \]
ContikiMAC Mechanism – Fast Asleep

Figure 5: The ContikiMAC fast sleep optimization: if a silence period is not detected before $t_l$, the receiver goes back to sleep. If the silence period is longer than $t_i$, the receiver goes back to sleep. If no packet is received after the silence period, even if radio activity is detected, the receiver goes back to sleep.
ContikiMAC Mechanism – Phase Lock

Figure 6: Transmission phase-lock: after a successful transmission, the sender has learned the wake-up phase of the receiver and subsequently needs to send fewer transmissions.
Implementation

- The ContikiMAC implementation in Contiki 2.5 uses the Contiki real-time timers (rtimer) to schedule its periodic wake-ups.
- The ContikiMAC wake-up mechanism runs as a protothread which performs the periodic wake-ups and implements the fast sleep optimization.
- The phase-lock mechanism is implemented as a separate module from ContikiMAC which maintains a list of neighbors and their wake-up phases.
- The neighbor is evicted from the list after a fixed number of failed transmissions or having no link layer ack within a fixed time.
Evaluation

Figure 7: A ContikiMAC wake-up with no signal detected. The two CCAs are seen in the lower graph.

Figure 8: A ContikiMAC wake-up with radio activity detected and where the fast sleep optimization quickly turns the radio off.

Figure 7 shows the current draw of a ContikiMAC wake-up that did not result in any packet reception. In the lower graph, we see that the radio is turned on twice, to perform the two CCAs of the ContikiMAC wake-up.

Figure 8 shows a ContikiMAC wake-up where the second CCA detected spurious radio activity. The radio is then kept on for a while longer, until the fast sleep optimization turns off the radio.
Evaluation

Figure 9: Broadcast reception: wake-up, packet detected, broadcast packet received.

Figure 11: Broadcast transmission.

Figure 10: Unicast reception: wake-up, packet detected, unicast packet received

Figure 12: Non-synchronized unicast transmission (with subsequent wake-up at 110 ms)
Figure 14: The energy consumption of the individual ContikiMAC operations.
Evaluation – Power Consumption

- We use the radio duty cycle: the portion of time in which the radio is on as indicator of radio power consumption

RDC choices for MAC layer:

- ContikiMAC
- X-MAC
- LPP (Low-Power probing)
- CX-MAC (Compatibility X-MAC)
- NULLRDC

Figure 15: The radio duty cycle in a data collection network with path loss, with X-MAC and ContikiMAC, as a function of the wake-up frequency (in the graph called channel check rate).
Evaluation – Power Consumption

• Figure 16 shows that the fast sleep and phase-lock optimizations significantly reduce power consumption.
• Figure 17 shows that optimizations are more efficient in the face of loss. This is because of a phase-locked transmission being shorter than nonphase-locked transmissions, leading both to less energy being spent on transmissions and to less radio congestion.
Conclusion

• The ContikiMAC uses a simple but elaborate timing scheme to allow its wake-up mechanism to be highly power efficient, a phase-lock mechanism to make transmissions efficient, and a fast sleep optimization to allow receivers to quickly go to sleep when faced with spurious radio interference.

• The Measurements show that the energy cost of ContikiMAC mechanism is significantly lower than existing duty cycling mechanisms and that the phase-lock and fast sleep mechanisms reduce the network power consumption between 10% and 80%, depending on the wakeup frequency of the devices in the network.
Thank you.