Demo I

- In class on 10/5 and 10/7.
- 12 min per team.
- Must show something real.
- Submit a video before class as backup.
The IoT Stack

- Miniaturized devices: embedded OS.
- **Low-power wireless: connect devices to the Internet.**
- Data analytics: make sense of sensor data.
- Cloud and edge computing: scalable real-time data processing.
IoT Wireless Technologies

- Wireless Personal Area Networks (WPAN)
  - Bluetooth Low Energy (BLE)
  - Based on IEEE 802.15.4 radio (PHY): Thread, Zigbee, WirelessHART

- Wireless Local Area Networks (WLAN)
  - Different versions of WiFi

- Wireless Wide Area Networks (WWAN)
  - Cellular: NB-IoT
  - Low Power Wide Area (LPWA): LoRa, SigFox, Sidewalk

- Choice of wireless technology depends on range, data rate, latency, power…
Thread: IP-based Mesh Networking

- Interoperability through IPv6
- IP protocol stack based on IEEE 802.15.4 radio
- OpenThread: open-source implementation
  - Released by Google based on Nest technology
  - https://openthread.io

Source: Thread Group
https://www.threadgroup.org
Channels in the 2.4 GHz ISM Band

- 802.11b/g
- Bluetooth (22 MHz)
- ZigBee
Power-Efficient Media Access Control

Chenyang Lu
CSMA/CA

Carrier Sense Multiple Access, Collision Avoidance
Hidden Terminal Problem
Virtual Carrier Sense: RTS/CTS

- RTS
- Payload
- CTS
- NAV
Power-Saving MAC

- Many applications’ expected lifetime: months or years
- Actual lifetime:
  - AA batteries: 2000 mAh (if you’re lucky)
  - CC2420 radio: 19.7 mA when idle but awake (RX mode)
  - $2000 \text{ mAh} / 19.7 \text{ mA} = 101.5 \text{ h} \approx 6 \text{ days}$

- This is a problem!
- Solution: keep the radio asleep most of the time
  - Duty cycles on the order of 0.1% – 1%
Types of Power-Saving MACs

- **Scheduled contention**: nodes periodically wake up in unison, contend for access to channel, then go back to sleep
  - S-MAC [Ye 2002], T-MAC [van Dam 2003]

- **Channel polling**: nodes independently wake up to sample channel
  - B-MAC [Polastre 2004], X-MAC [Buettner 2006], Box-MAC2

- **Time Division Multiple Access (TDMA)**: nodes maintain schedule of when to wake and when to transmit
  - DRAND [Rhee 2006], WirelessHART/TSCH

- Hybrid protocols: SCP [Ye 2006], Z-MAC [Rhee 2005], Funneling MAC [Ahn 2006], 802.15.4 MAC [IEEE 2003]
S-MAC: Synchronized Sleeping

- Nodes stay asleep most of the time
- Periodically wake up for short intervals to see if anyone’s sending
- Low energy consumption when traffic is low
- Sacrifice latency for longer lifetime
S-MAC: Sending a Packet

- Time awake divided into two parts: SYNC and RTS
- Node periodically send SYNC packet to keep clocks in sync
- CSMA/CA used to contend for access to wireless channel
If a node wants to send data, send RTS and wait for CTS from receiver.

Carrier Sense (CS) before RTS/CTS.
S-MAC: Sending a Packet

- CTS for someone else $\Rightarrow$ sleep
  - Overhearing avoidance

- Sender does one RTS/CTS then sends data for rest of frame
  - Performance $>$ node-level fairness

- All data packets are ACKed
  - Packet fragmentation $\Rightarrow$ reliability
A Look at S-MAC

- Power savings over standard CSMA/CA MAC

- Long listening interval is expensive
  - Everyone stays awake unless somebody transmits

- Time sync overhead even when network is idle

- RTS/CTS overhead when sending data
  - Expensive for small packets
  - Amortized for large packets through fragmentation
B-MAC
A Look at B-MAC

- Low overhead when network is idle
- Simple to implement
- Better power savings, latency, and throughput than S-MAC

- Lower duty cycle $\rightarrow$ longer preambles
  - Higher average latency
  - Higher cost to send
  - Higher cost for overhearing
B-MAC: Room for Improvement

Lower duty cycle $\rightarrow$ longer preambles
- Higher average latency
- Higher cost to send
- Higher cost for overhearing
X-MAC: Overhearing Avoidance
X-MAC: Preamble ACKing
A Look at X-MAC

- Better latency, throughput and power efficiency than B-MAC
- Little energy consumed by overhearing
- Still simple to implement

- On average, cuts preamble by half → sending packets is still expensive
Scheduled Contention + Channel Polling
SCP: Adaptive Channel Polling

- Add $N$ sub-intervals whenever packet received
  - Reduces latency of bursty data
- Continue adding sub-intervals as long as they’re needed
A Look at SCP

- Channel polling: low overhead when idle
- Scheduled contention: low cost to send
- Low latency for multi-hop traffic
- Complex to implement
- Overhead due to time synchronization
  - Reduced by piggybacking on data packets
TDMA

Frame

Time Sync
A Transmits
B Transmits
C Transmits
Sleep
Sleep

Slot
A Look at TDMA

- Predictable latency, throughput, and duty cycle
- Low packet loss due to less contention
- Schedules must be changed when nodes leave/enter neighborhood
- Time synchronization overhead
- Slots wasted when scheduled node has nothing to send
- Complex in multi-hop networks
Z-MAC: TDMA + Channel Polling

<table>
<thead>
<tr>
<th>Time Sync</th>
<th>A Transmits</th>
<th>B Transmits</th>
<th>C Transmits</th>
<th>Sleep</th>
<th>Sleep</th>
</tr>
</thead>
</table>

A

B

C
A Look at Z-MAC

- Reduces waste from unused slots → higher throughput and lower latency
- Throughput, latency and duty cycle no worse than pure TDMA
- Nodes still stay awake if no one transmits
- Still overhead from time sync
Time Slotted Channel Hopping (TSCH)

- Part of IEEE 802.15.4e
  - WirelessHART
  - 6TiSCH (IPv6 over the TSCH mode of IEEE 802.15.4e)

- Switch channel for every time slot to enhance reliability

- In each 10-ms time slot
  - Sender & receiver set channel
  - Sender waits for guard time to accommodate clock jitter
  - Sender transmits the packet
  - Receiver sends ACK if it receives the packet

- Superframe: a series of time slots defining the communication schedule of a set of devices
Challenges to Build/Maintain MAC

- MAC protocols have radio-dependent requirements beyond normal application code
  - Turn radio on/off, low-latency I/O, carrier sense, etc.

- Typically implemented by forking radio stacks
  - Hard to implement
  - Must be maintained when the original radio stack changes

- MAC layers supported by TinyOS
  - CC2420: BoX-MAC-2 (enhancements to X-MAC)
  - CC1100 and CC2500: Wake-on-Radio, X-MAC, B-MAC, BoX-MAC-1, BoX-MAC-2
A Unified Approach: MLA

- MAC Layer Architecture (MLA) [Klues'07]
  - Low-level abstractions of radio functionality
  - High-level components for common MAC features

- Used to create 5 platform-independent MAC
  - B-MAC, X-MAC, SCP, TDMA, variant of Z-MAC

References


