Comparison of XMAC and BMAC Protocols

Performance Evaluation of
Computer Systems and Networks (CS681)
Course Project

Vijay Gabale (07305004)

Ashutosh Dhekne (07305016)

Outline

- Motivation
- X-MAC, B-MAC protocols
- Implementation
- Architecture
- Simulation
- Metrics
- Experiments and results
- Conclusion

Motivation

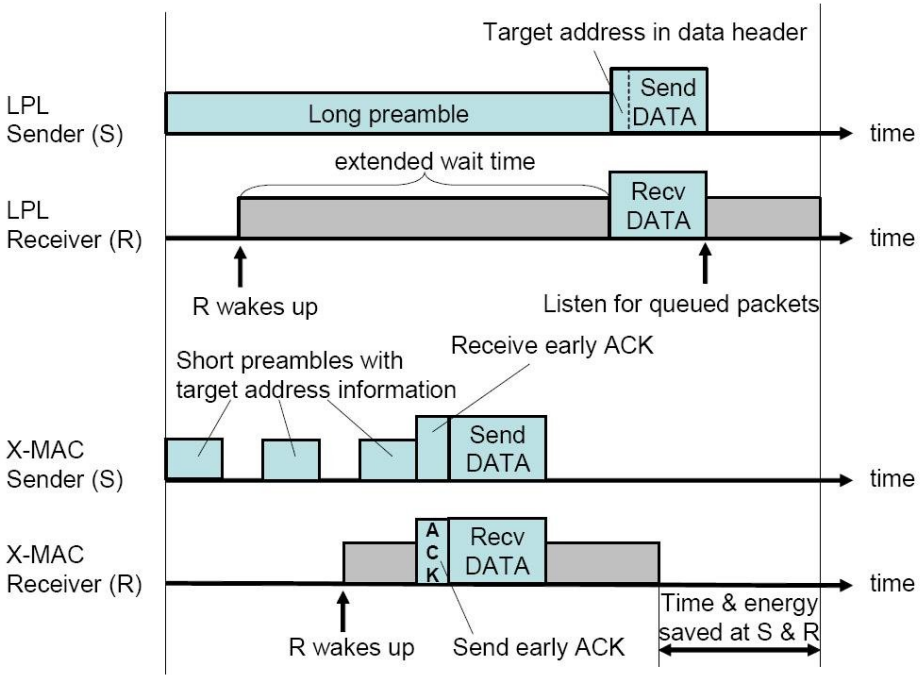
- Comparison of the two protocols
- Experimenting with multi hop routes
- Investigate energy savings and latency deficits
- Interesting Questions
  - What are the load conditions under which X-MAC outperforms B-MAC (or vice versa)
  - What is the upper bound on end-to-end latency?
  - Given a lifetime of the network, what should be the sleep period?

The Protocols

- Both are asynchronous protocols for wireless sensor networks
- Data transfer across duty cycled nodes is the primary objective
- Constraints on energy available at the node
- Sampling frequency is usually very low, possibly once in few minutes
- Use of preamble to inform neighbor about impending data transfer

Difference in preamble mechanism

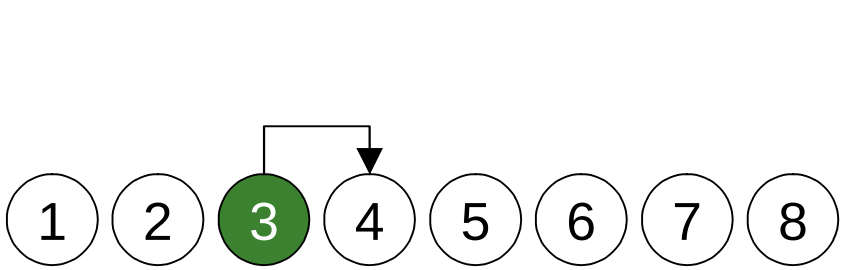
- BMAC sends a long continuous preamble
- XMAC sends short strobes
- Target ID is encoded in the XMAC strobe

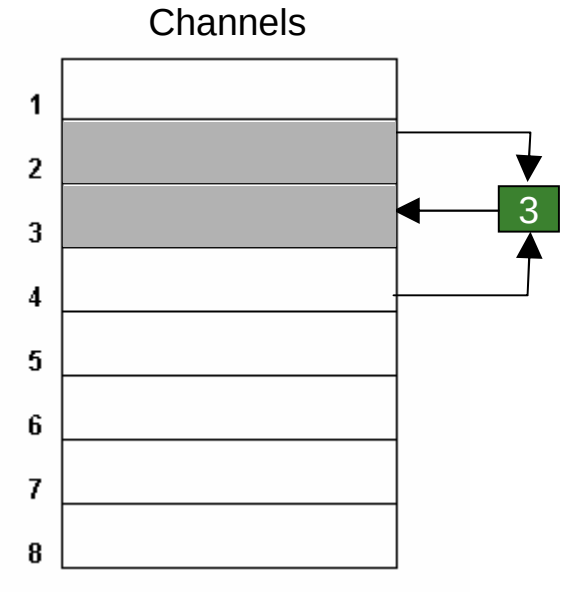


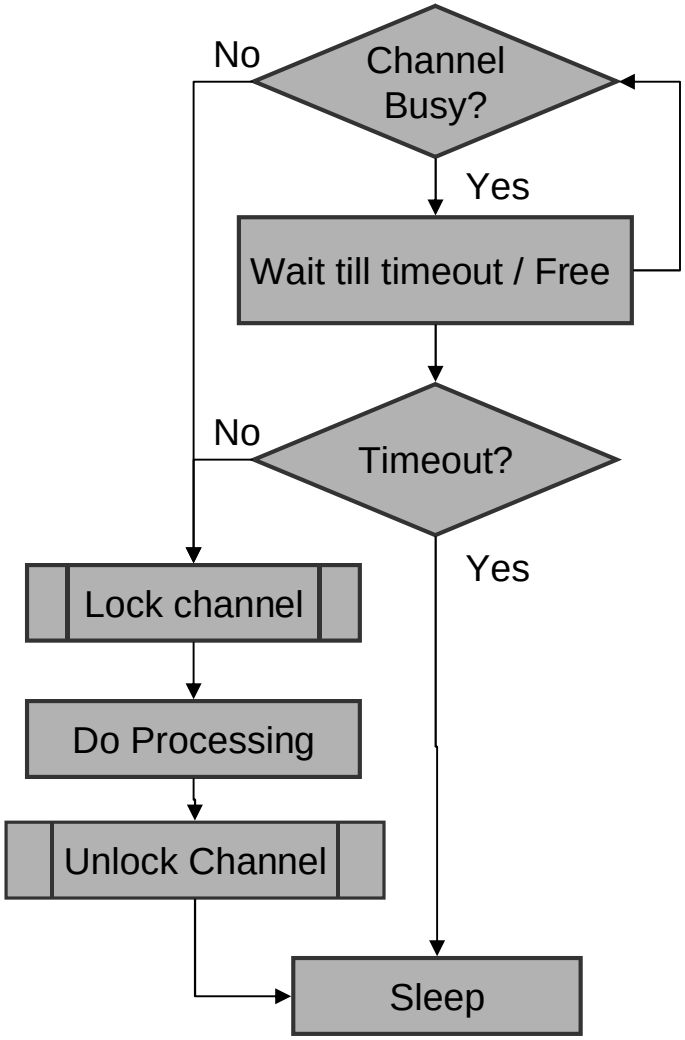
Implementation

- CSMA/CA protocol
- Lossless wireless medium – global buffers
- Grid network as well as linear topology
- Simulated individual node – use thread!
- Scheduling unpredictability – use locks, conditioned waits and signals!
- Per node measurement – thread-specific clock!
- Randomness inherent to scheduling of threads

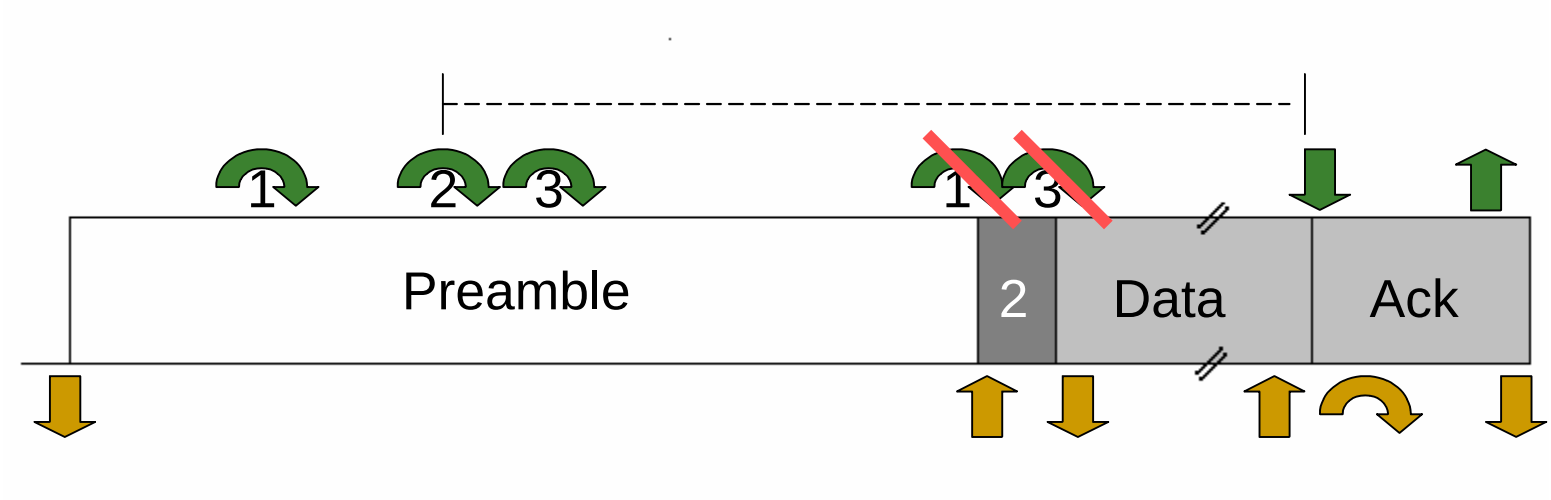
Architecture







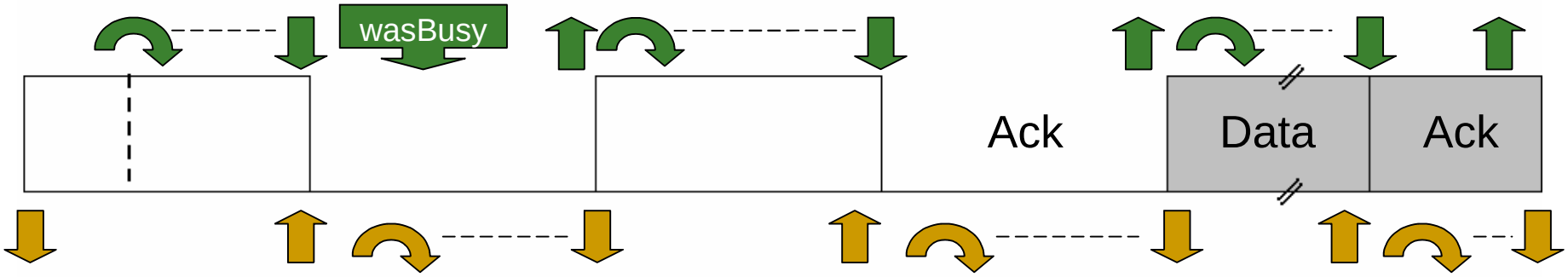
B MAC Data Send and Receive

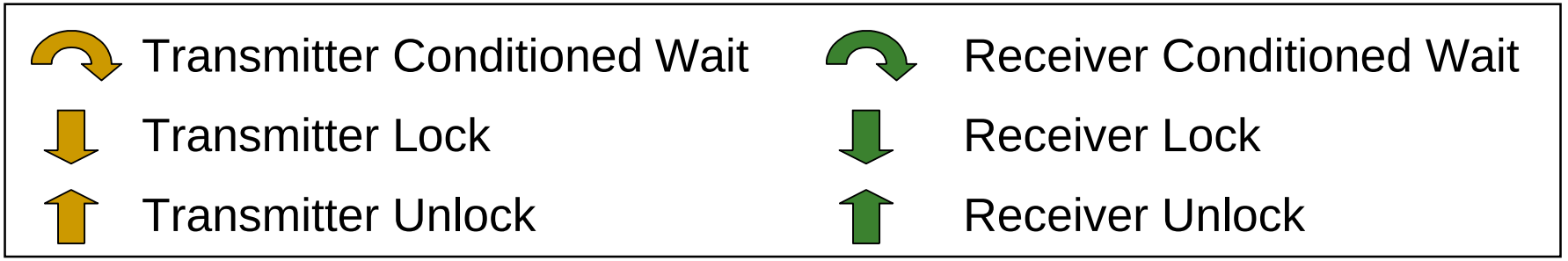


Transmitter Conditioned Wait	Receiver Condition Wait
Transmitter Lock	Receiver Lock
Transmitter Unlock	Receiver Unlock

X MAC Data Send and Receive

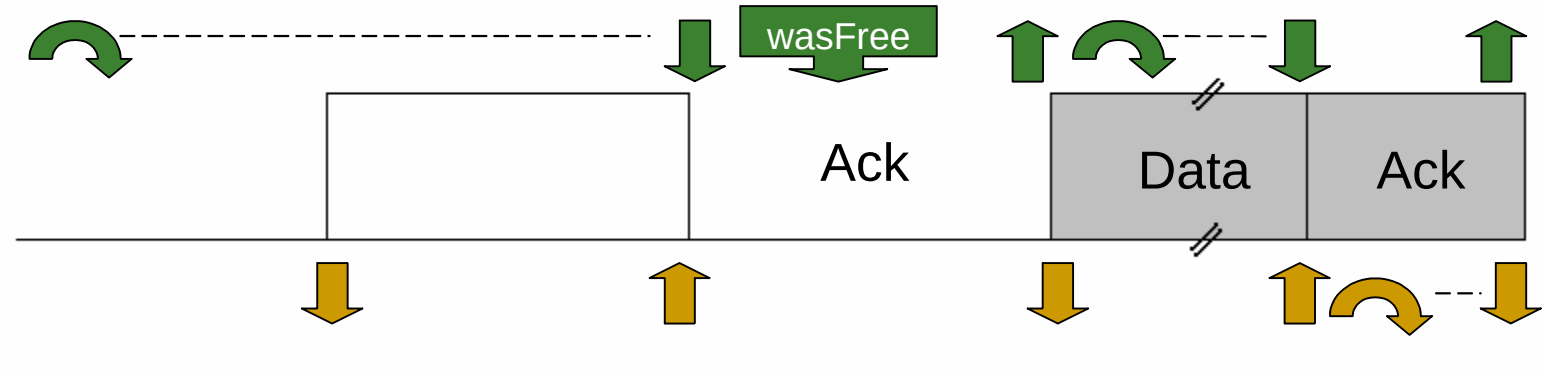
- Receiver woke up in between a strobe





X MAC Data Send and Receive

- Receiver woke up when channel was free



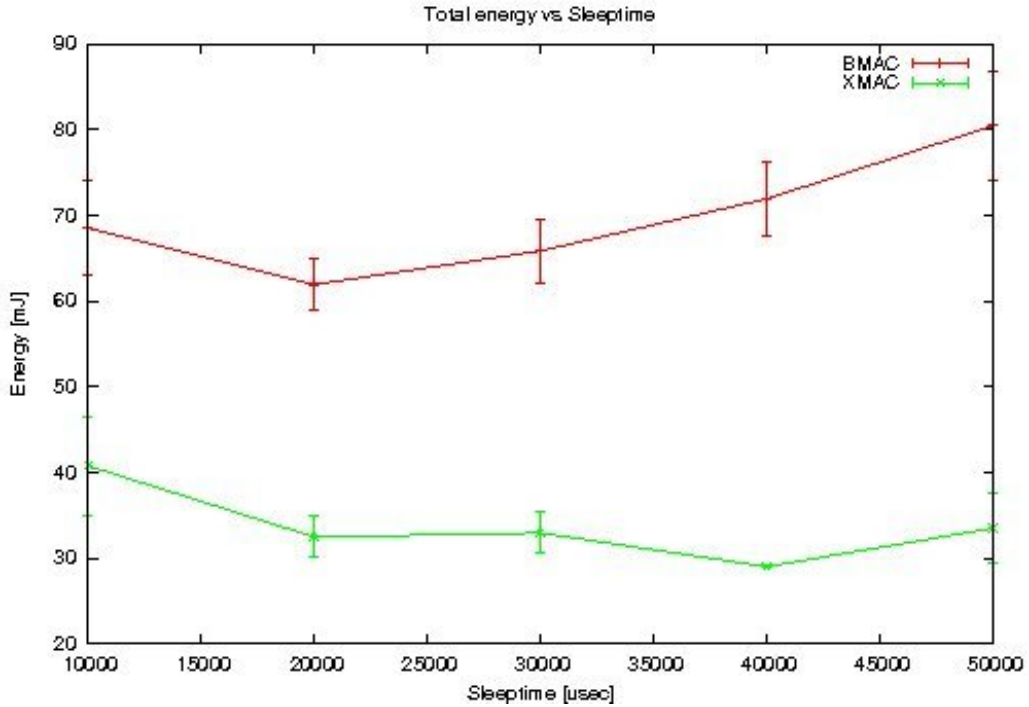
Transmitter Conditioned Wait	Receiver Conditioned Wait
Transmitter Lock	Receiver Lock
Transmitter Unlock	Receiver Unlock

System parameters and Metrics

- System parameters
  - Sleep time
  - Number of nodes
  - Strobe interval
- Metrics
  - Energy
  - Latency
  - Duty cycle

How protocols behave for different sleep times?

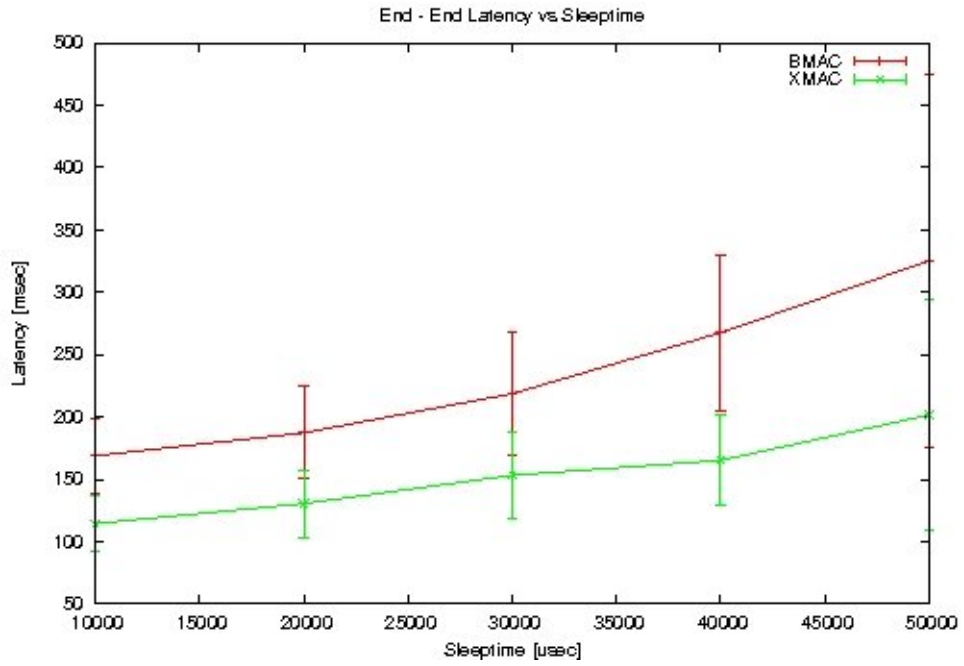
- Set up
  - Nodes: 9 (Grid)
  - Sleep time: 10ms to 50ms
- Measure
  - Energy



B-MAC requires more energy than X-MAC at any sleep time. Moreover, energy required increases with increased sleep time. It is so due to the overhead of listening to the long preamble in B-MAC

How protocols behave for different sleep times?

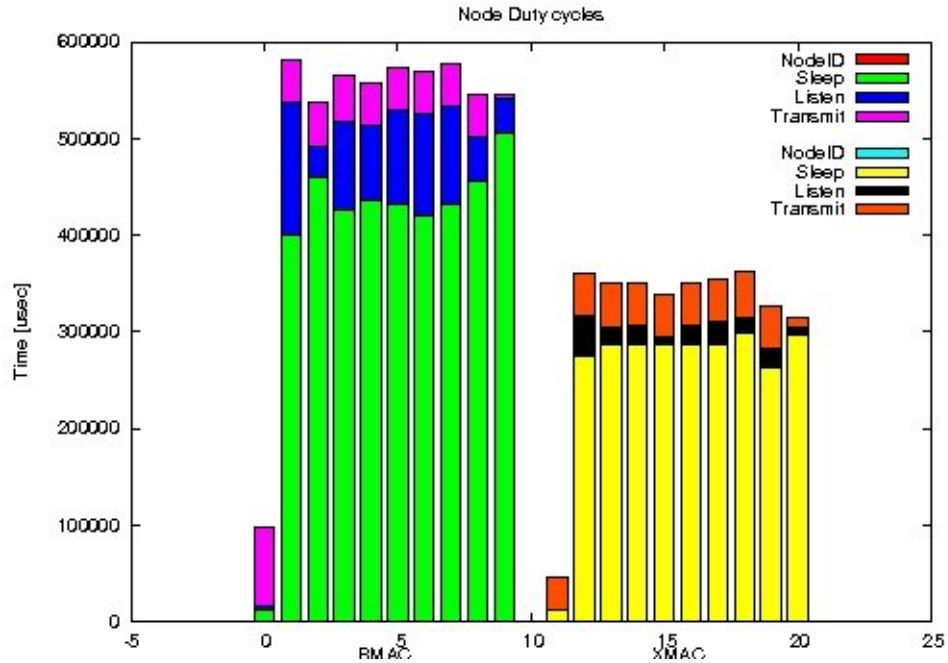
- Set up
  - Nodes: 9 (Grid)
  - Sleep time: 10ms to 50ms
- Measure
  - Latency



B-MAC requires more time to transfer packets from the source to the destination. This is because the entire preamble has to be always sent, even though the receiver was already awake. X-MAC saves this time.

How protocols behave with respect to duty cycling?

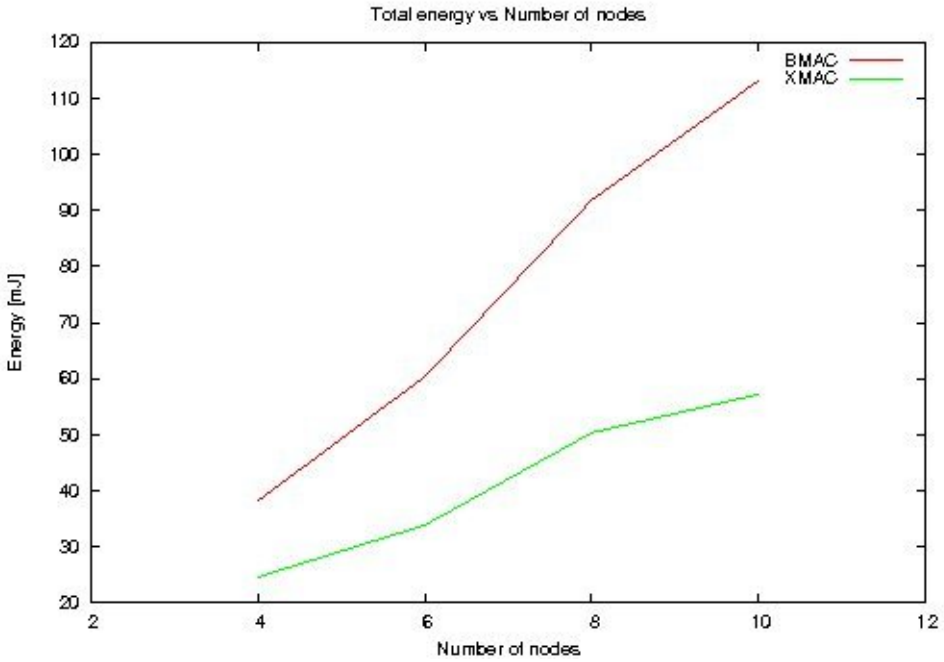
- Set up
  - Nodes: 10
  - Sleep time: 10ms
- Measure
  - Duty Cycle



X-MAC not only completes a transfer faster, it also allows lower duty cycling.

How protocols behave for different number of nodes?

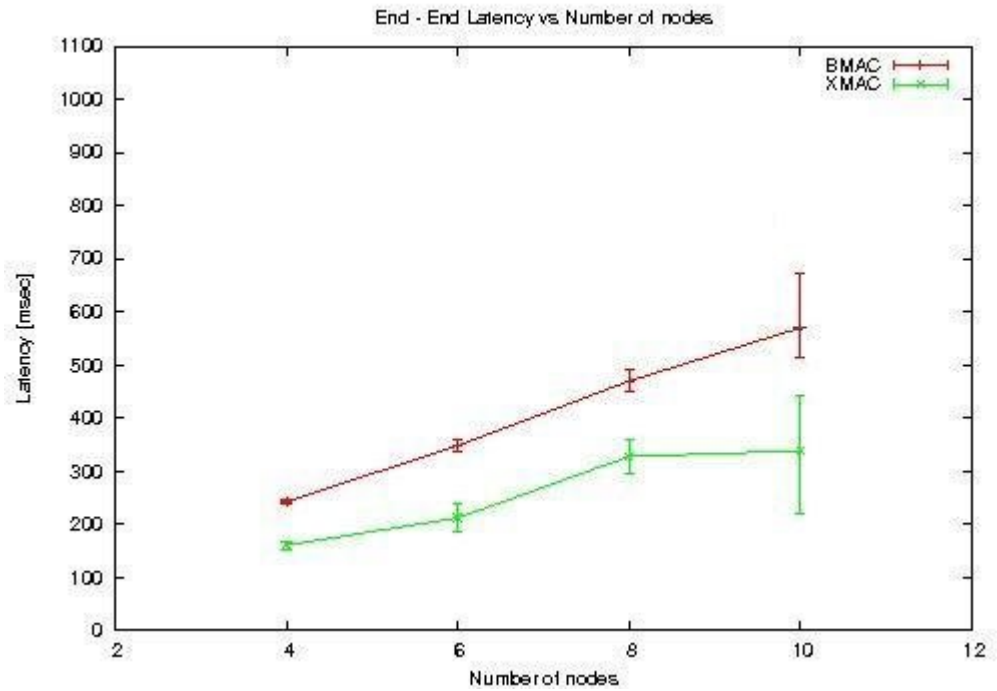
- Set up
  - Nodes: 4 to 10
  - Sleep time: 10ms
- Measure
  - Energy



Energy required to transfer data obviously increases with number of intermediate hops. However, for B-MAC, it increases faster than that for X-MAC. This is a benefit of using strobe preamble.

How protocols behave for different number of nodes?

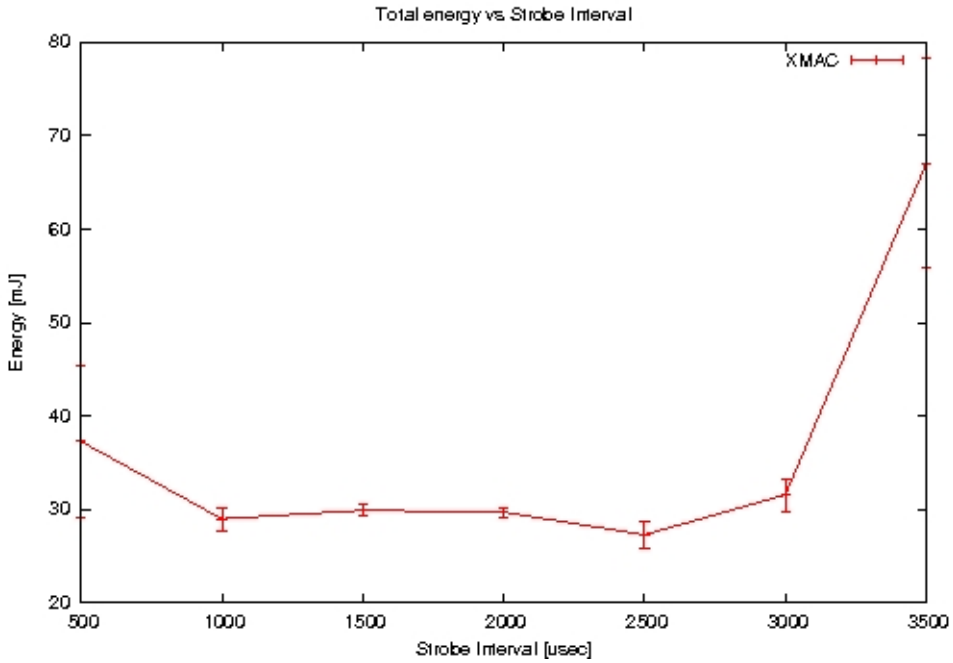
- Set up
  - Nodes: 4 to 10
  - Sleep time: 10ms
- Measure
  - Latency



B-MAC almost always requires more time to transfer packets end to end. With increasing nodes, this time does not increase as fast as we would have expected.

How X-MAC behaves for different strobe intervals?

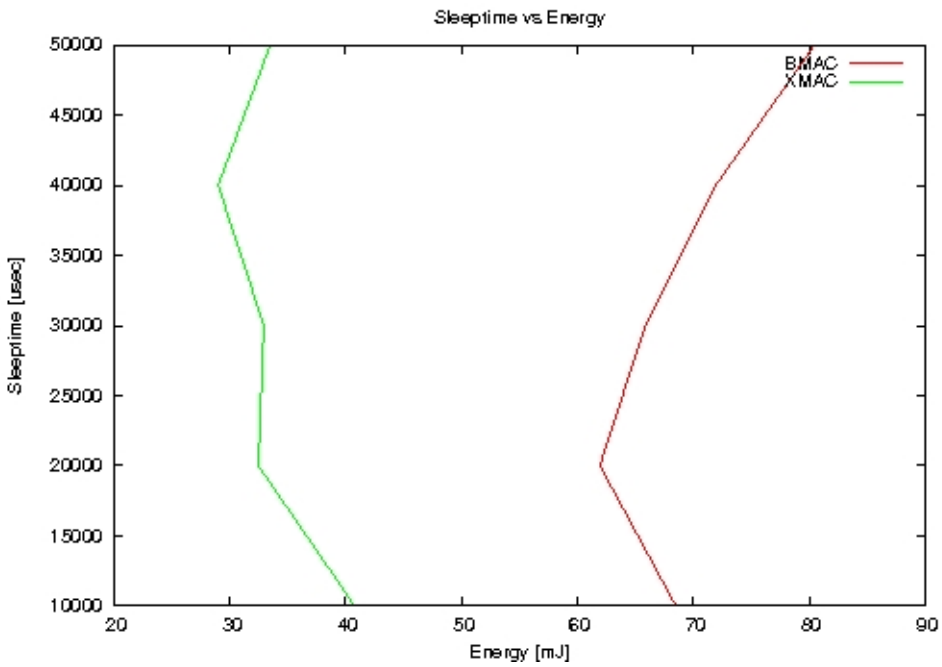
- Set up
  - Nodes: 9
  - Sleep time: 10ms
  - Strobe Interval 500 to 3500
- Measure
  - Energy



With increased strobe interval, the X-MAC comes closer to B-MAC. With increased strobe interval, the overall energy consumption also increases.

How much energy will be required?

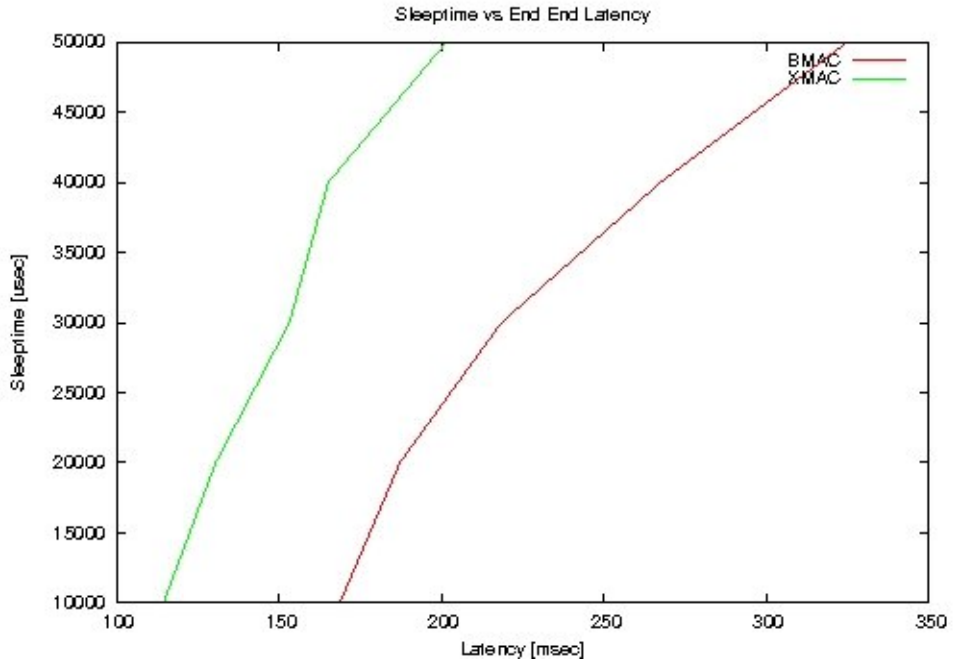
- Set up
  - Nodes: 9
  - Sleep time: 10ms to 50ms
- Measure
  - Energy



Given a certain energy constraint on the network, we can choose the appropriate sleep time. This directly translates to the expected lifetime of the network.
Energy (J) = Power (W) * Time (sec)

How much latency to expect?

- Set up
  - Nodes: 9
  - Sleep time: 10ms to 50ms
- Measure
  - Latency



Increasing the sleep time does not help B-MAC. Both energy as well as latency worsen. With X-MAC, the energy slightly dips at an optimal sleep period. Latency almost monotonically increases with sleep period.

Conclusion

- X-MAC outperforms B-MAC in both energy consumed and the latency of end to end packet transfer
- X-MAC offers better duty cycling opportunities—nodes sleep for more time
- Nevertheless, the “Sleep time” system parameter must be judiciously chosen for optimal performance

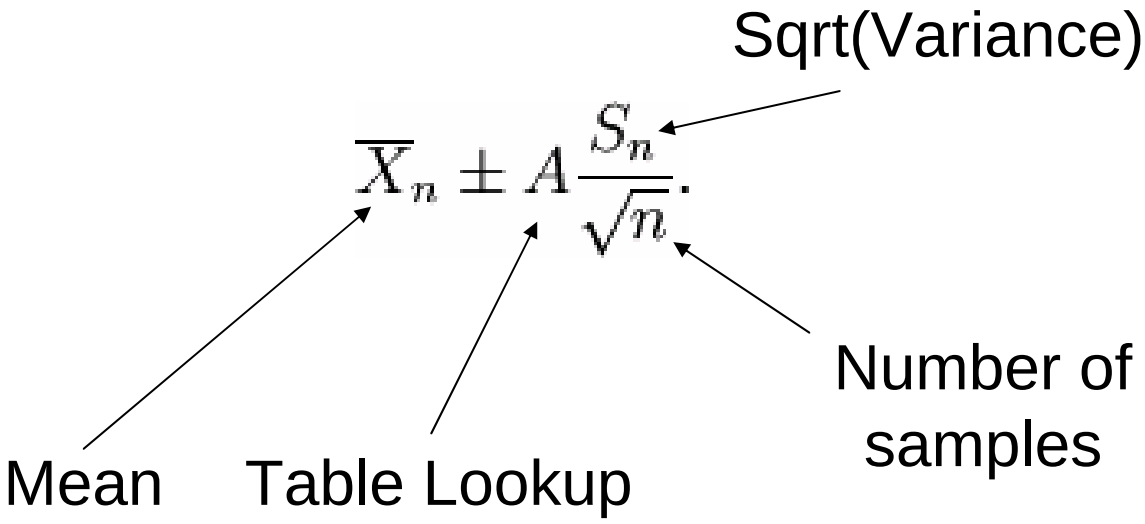
References

- Michael Buettner, Gary V. Yee, Eric Anderson, Richard Han. X-MAC: A Short Preamble MAC Protocol for Duty-Cycled Wireless Sensor Networks. *Sensys '06.*
- J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *The Second ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 95–107, November 2004.
- Confidence Interval calculation in http://www.cse.usf.edu/~christen/tools/toolpage.html
- Confidence interval general information from http://en.wikipedia.org/wiki/Confidence_interval and http://en.wikipedia.org/wiki/Student%27s_t-distribution

Thank you

Confidence Interval Calculations

- 90% confidence interval
- Calculated using the formula:



Division of work

- B-MAC, X-MAC frame formats
- Sensor network design
- Threading architecture
- Measurements
- Shell, Python scripting
- One is alone, two is company, three is crowd!

Software requirements

- The simulation program is created in C
- Uses shell script, python script and gnuplot for producing results
- We require the pthread library and the math library to be installed. (–lpthread –lrt –lm flags required)
- GNUPlot 4.2 for the histogram graphs.