Evaluating a BASIC Approach to Sensor Network Node Programming

J. Scott Miller Peter A. Dinda Robert P. Dick*

Northwestern University
*University of Michigan

ABSYNTH Project
http://absynth-project.org/

Summary

- Evaluated the efficacy of BASIC for simple sensor network applications through user studies
- Half of users with no programming experience are able to complete sensor network tasks with BASIC
- Iterated on BASIC design using study data
- BASIC has minimal power overhead with a realistic workload, and can be compiled to virtually eliminate any overhead

Outline

- Summary
- Motivation
- BASIC implementation
- User study evaluation
- Application to structural monitoring
- Power consumption
- Conclusion

ABSYNTH Project

- **Goal:** make it easier for domain experts to design and implement wireless sensor network applications
- Combining use of language, compiler, and synthesis technologies
  - Extensive use of user studies

Motivation of This Work

- Collaboration with Civil Engineering structural monitoring group (www.iti.northwestern.edu/acm/)
  - ▫ Previously developed hardware [Jevtic IPSN ’07]
- Current WSN languages and toolchains present steep barrier for such **application domain experts**
  - ▫ Domain experts are not embedded systems developers
  - ▫ Published applications involve collaborations between domain and embedded systems experts
- Many applications are node-oriented
  - ▫ Our structural monitoring application is one example
  - ▫ Our IPSN ’09 work considers network-oriented applications

Existing Tools

- Node-level languages
  - C, NesC, TinyScript, SensorScheme, Micro.NET, Java, …
- Macro-programming languages
  - Regiment, TinyDB, Tables, WASP, Macrolab, …
- Single-purpose hardware
  - EkoMote

- Most leverage advanced programming concepts
  - Threading, SQL joins, event-driven programming, etc.
- Effectiveness of these languages/toolchains for application domain experts is largely unknown
  - IPSN '09 work begins to measure this

Our Approach

- Bottom-up approach to language design
  - Start with general purpose language and extend
  - Assume end-user has minimal programming experience
- Start with a simple language
  - BASIC (TinyBASIC dialect)
  - Proven effective for novice programmers (even children) in other areas
- Evaluation through user studies
  - Participants approximate domain experts
- Iterate on design with user feedback

Why BASIC?

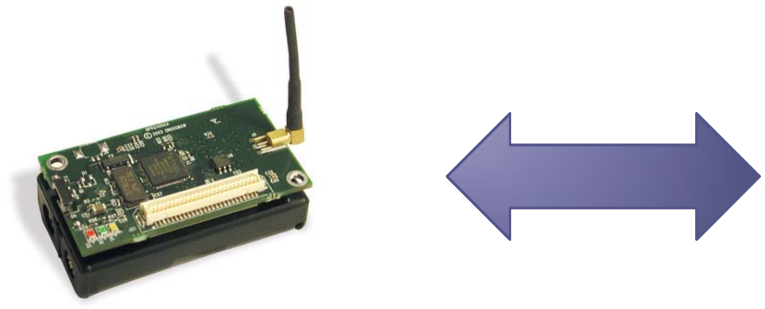
- Simple execution model
  - Few impediments to learning (e.g., threads, events)
  - Successful programmers understand execution model of their languages [V. Someren]

- Suitable for simple applications
  - Codebase unlikely to grow

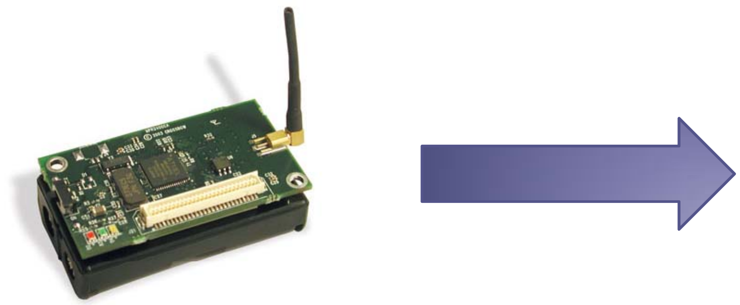
BASIC Implementation

- Started with Adam Dunkels’suBASIC codebase
  - Grammar similar to TinyBASIC
  - Ported to Mantis OS
- Extended with WSN primitives
  - SENSE, SLEEP, SEND, RECEIVE, LED, ADC, DAC statements
  - Follow BASIC conventions
- Developed BASIC IDE
  - Rapid development
- Created BASIC tutorial

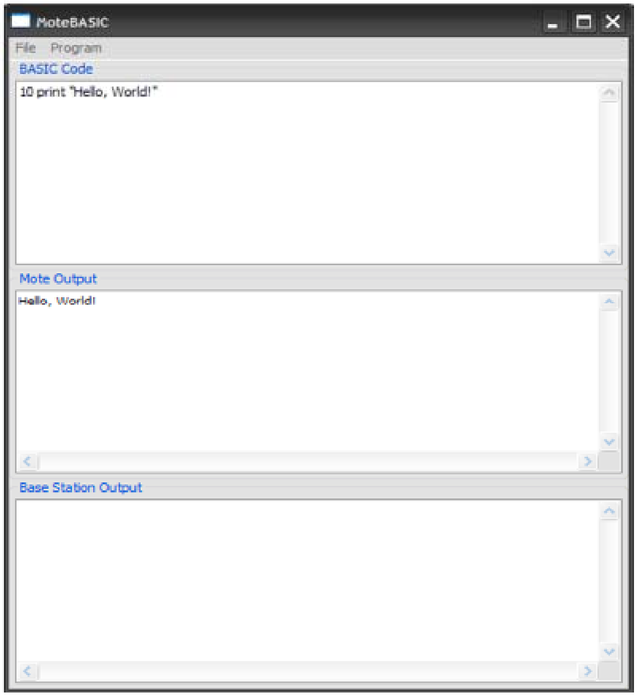
What the user sees



Target Mote



“Base Station” Mote



BASIC Code

```
10 print "Hello, World!"
```

Mote Output

```
Hello, World!
```

Base Station Output

Example Application 1

```
10 sleep period 15 min

20 dim a[1000]
30 sense adc 1 into a at 1000 hz for 1000 samples

40 send time
50 send average(a)

60 resume
```

- Implements collaborator’s crack sensor

Example Application 2

```
10 sleep channel 1 thresh 512

20 print "Start of Event:"
30 send time

40 dim a[5000]
50 sense adc 1 into a at 1000 hz for 5000 samples

60 print "Crack Data:"
70 send a

80 resume
```

- Implements collaborator’s crack sensor

Benchmark Languages

- Comparison with C/NesC impractical
- Network-oriented languages out-of-scope
- TinyScript closest (functionality/goals)
  - Event-driven model
  - Strongly-typed, shared variables
  - One-hop and base station-oriented communication

Experience of Domain Experts

Question	Response - mean (std. dev.)	
	Domain Experts	WSN Experts
Largest program written (LOC)	600 (935)	93,614 (182,558)
Largest program modified (LOC)	413 (440)	156,286 (154,286)
LOC changed or added	81 (146)	3,337 (5,419)
Languages known	4	8.9

- Surveyed collaborators at 4 Universities
- High variation in responses
- Domain experts report experience with Matlab, C++
- Consistent with IPSN '09 findings

User Study

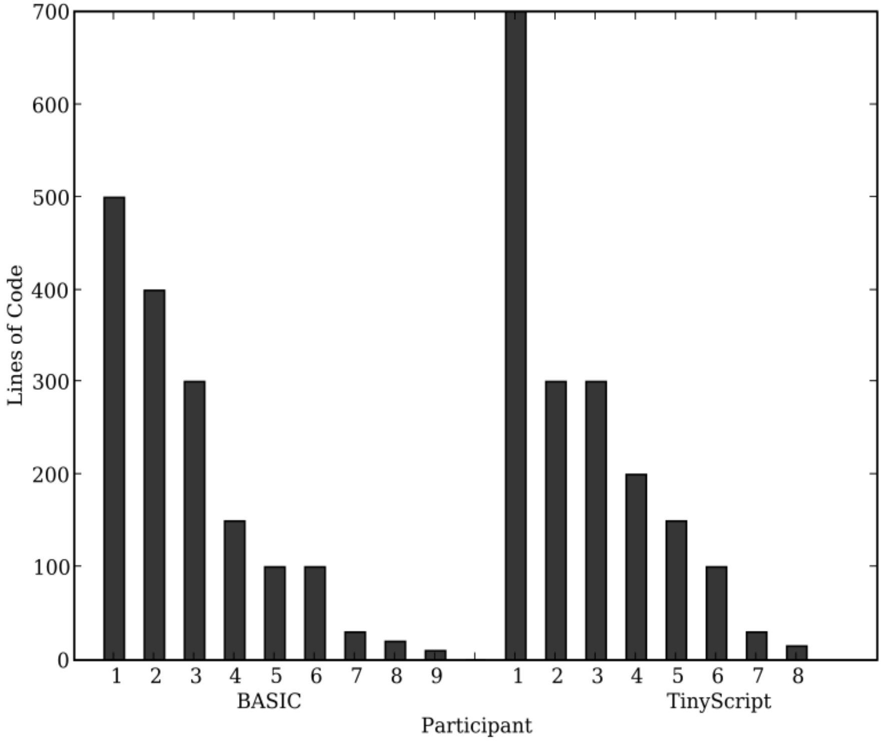
- **Goal**: Evaluate efficacy of BASIC for allowing such users to implement simple sensor node tasks
  - Also evaluated TinyScript
  - Tutorials for both carefully matched
- 3 tasks (must be implemented power-efficiently)
  - Blink
  - Sense-and-send
  - Actuation

Study Population

- Evaluated with 40 participants
  - 20 per language
- Recruited from Northwestern population
  - Mainly undergraduate and graduate students
  - Diverse backgrounds (IRB approval allowed for broad advertising)
  - Participants paid $15
  - Randomly assigned language
- Classified into two groups
  - **Novices**: No programming experience
  - **Intermediate**: Some programming experience

Previous Programming Experience

- Language experience
  - C/C++: 9 Participants
  - Java/C#: 6 Participants
  - Matlab: 6 Participants
- BASIC: 11 novices
  9 intermediates
- TinyScript:
  12 novices
  8 intermediates

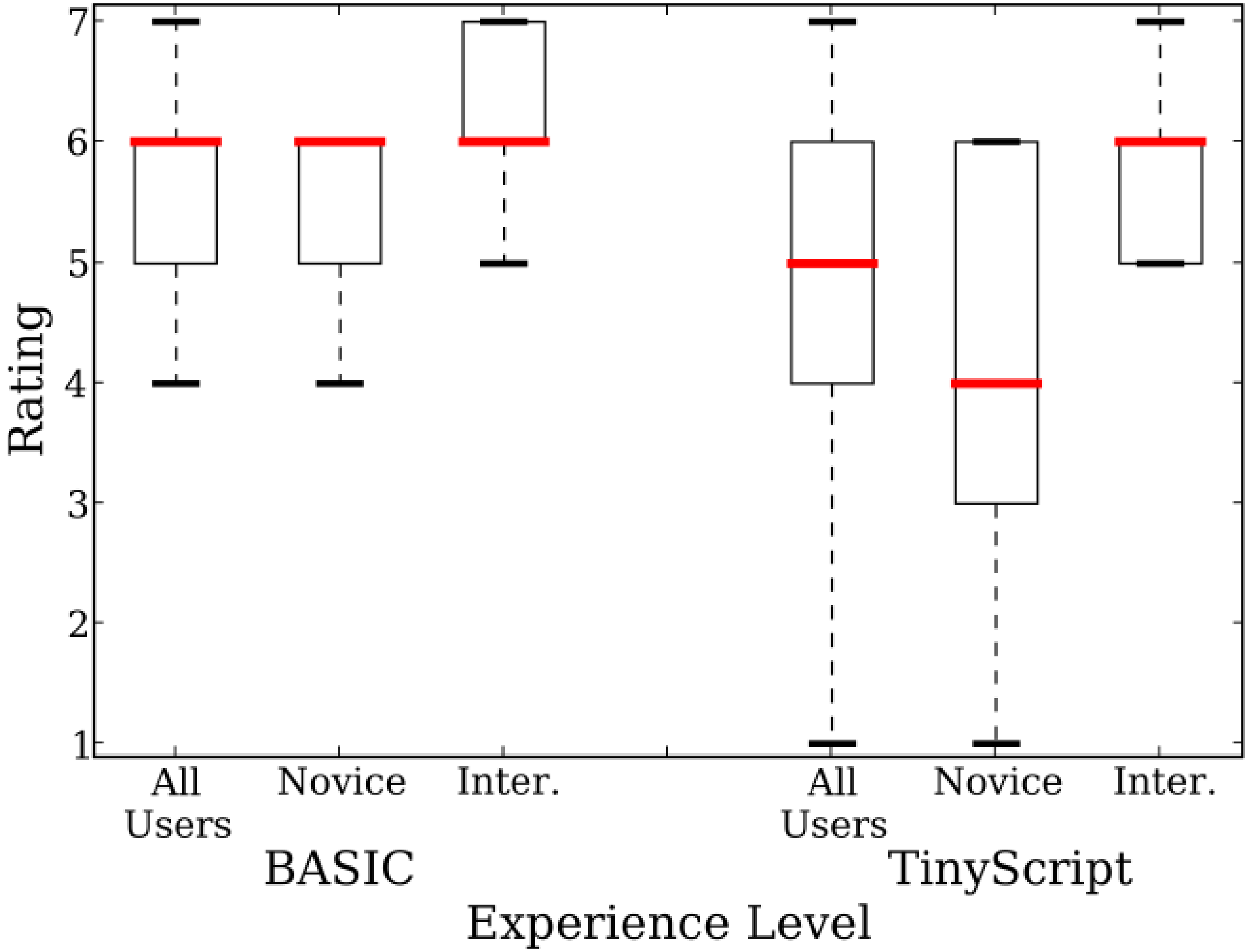


Largest program written for intermediate users in our study groups

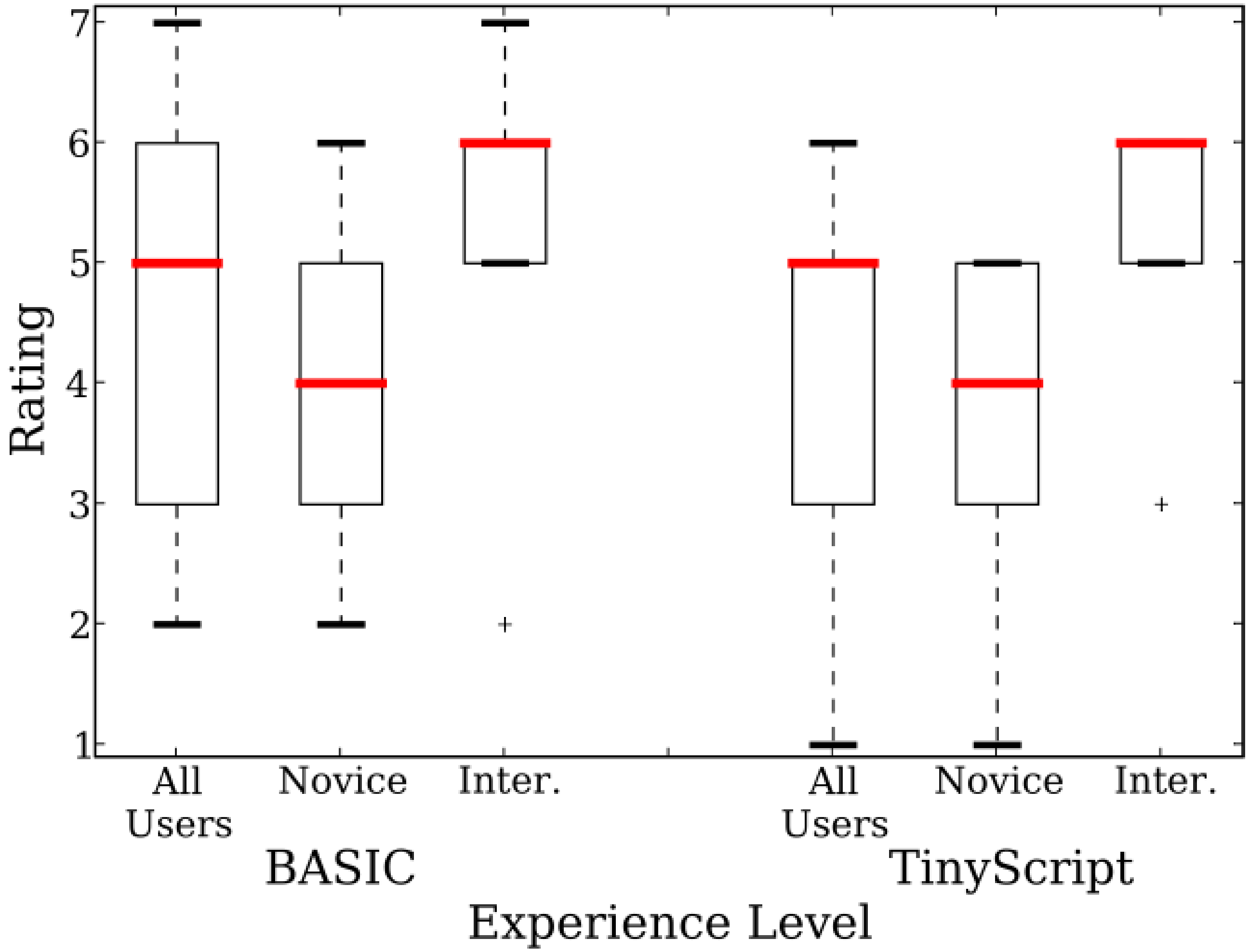
Study Design

- Experience classification questionnaire
- 30 minutes to read language tutorial
- 20 minutes for each task
- No proctor feedback
- Participants' work periodically saved to allow proctor assessment of progress/issues
- Participants provide feedback on the exercises and tutorial
    - Leikert scale

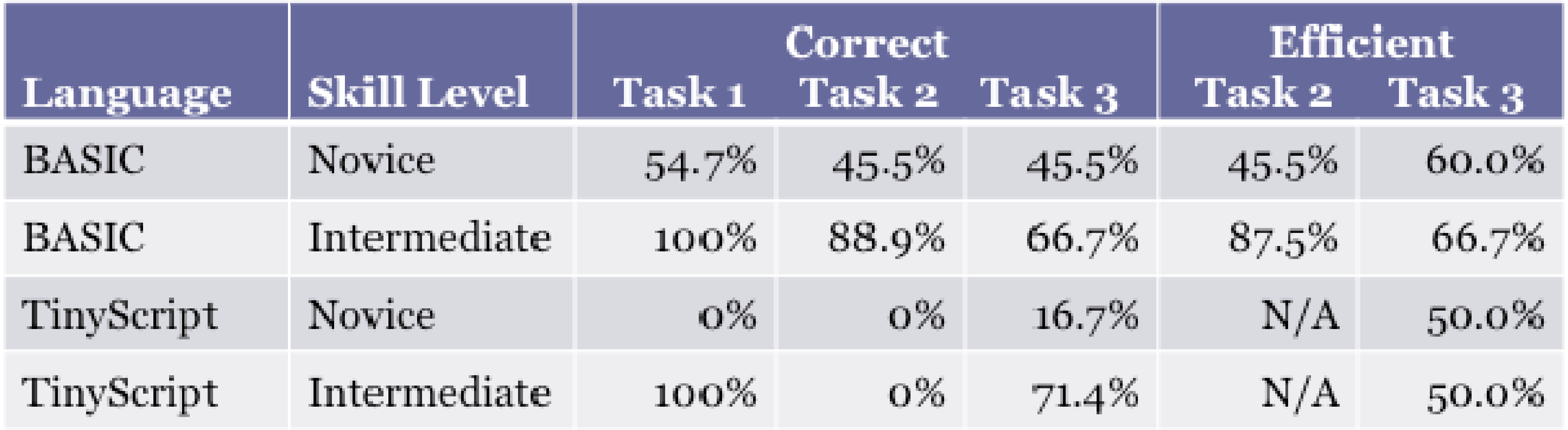
Tutorial understandability similar - User responses to the statement "I felt that the tutorial was easy to understand."



User confidence similar – User responses to the question “I feel that I understand [the language]”



Overall Results



		Correct			Efficient	
Language	Skill Level	Task 1	Task 2	Task 3	Task 2	Task 3
BASIC	Novice	54.7%	45.5%	45.5%	45.5%	60.0%
BASIC	Intermediate	100%	88.9%	66.7%	87.5%	66.7%
TinyScript	Novice	0%	0%	16.7%	N/A	50.0%
TinyScript	Intermediate	100%	0%	71.4%	N/A	50.0%

- Novice programmers (no experience) succeed half the time with BASIC
  - Few novices have success with TinyScript
- Intermediate programmers have similar rates of success with both languages
- Only 3 out of 15 correct TinyScript programs event-driven

Common Failures

- Confusion between serial and radio communication (both languages)
- Improper or missing duty cycling (both)
    - Missing sleep statement
    - Unnecessary or defensive sleep usage
- Array overflow (TinyScript)

User study-driven language enhancements

- RESUME statement added to simplify duty-cycling
- Arrays added
  - Pages transparently to flash
- SENSE statement extended to allow high-resolution sampling
- Modified SLEEP statement to allow wake from custom event detection hardware [IPSN ‘07]
- Minor syntactic changes to clarify keyword arguments

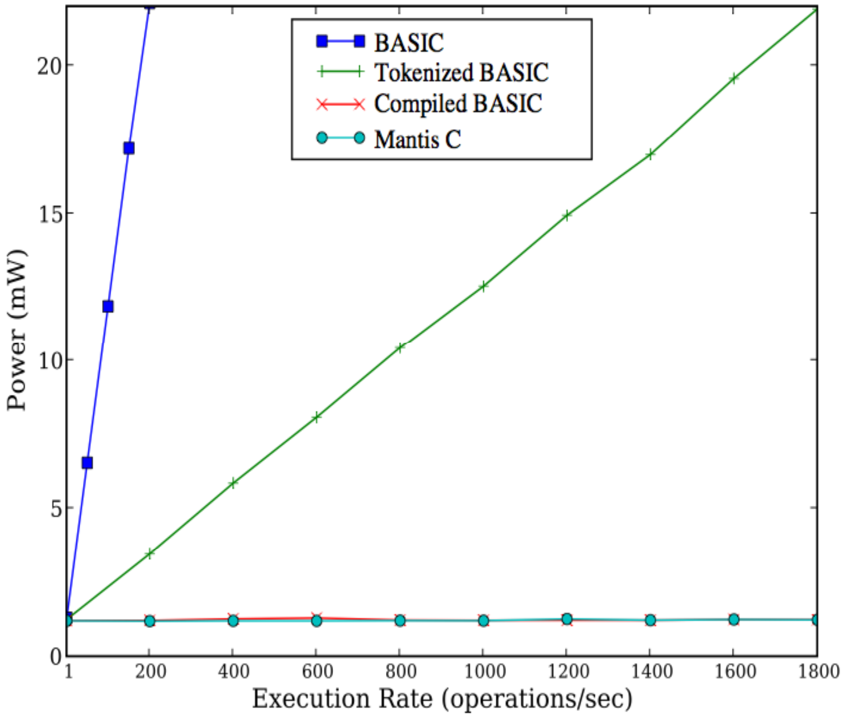
Domain Application

- Domain experts implemented an application in BASIC
  - Structural monitoring application
- Gathered two application specifications (in domain language) supplied by our collaborator
- Two of our collaborator’s students implemented both applications
  - Neither worked with sensor network hardware/software
- Study design similar to first
  - 30 minutes for each application
  - Solution checked by proctor

- **Result**: Both succeeded on first application after 1 iteration, immediate success on second

Power Consumption Manageable

- BASIC execution unsurprisingly slower than compiled C
  - Tokenization helps
- Compiled BASIC has identical power profile
- Sense-and-send application (1 Hz duty cycle) experiences only 1.5% increase in power consumption



Power consumption as a function of desired compute rate (loop iterations)

Conclusion

- BASIC enables domain experts with minimal or no programming experience to develop node-oriented sensor network applications
- User evaluation critical in understanding language efficacy and design

Questions?

For more information:

www.absynth-project.org