UNIVERSITY OF
Massachusetts Amherst

Maximalist Cryptography and Computation on the WISP UHF RFID Tag

Hee-Jin Chae*, Daniel J. Yeager**, Joshua R. Smith**, and Kevin Fu*

*University of Massachusetts Amherst
Department of Computer Science
**Intel Research Seattle

Conference on RFIDSec-2007, 12 July 2007

Motivation

- Strong cryptography available for HF and LF tags
  - MIFARE DESFire: triple-DES security
  - Many ISO14443-compliant tags (HF)

- No support for such cryptography on UHF tag

UHF Security, Why now?

- EPC Gen2 replacing barcode systems
- 5 cents? Probably not.
- 5 dollars? Until recently NO!
  - Improvements in the efficiency of microelectronics
  - Conventional cryptography no longer beyond the reach of a **general purpose** UHF tag

Challenges for UHF RFID Security

- Extremely resource-limited
  - At the very low-end of RFID tags
- Longer reading range
  - More vulnerable to attacks
- No development platform available for UHF
  - RFIDGuardian [Rieback06]: HF
  - DemoTag [Aigner06] : HF
  - Proxmark3 [Westhues] : LF and HF

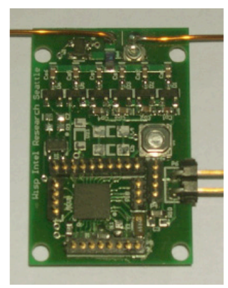
Minimalist vs. Maximalist Approaches

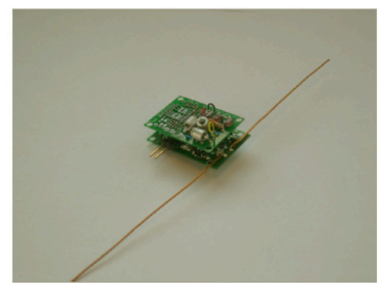
- Minimalist Approach
  - Minimize cryptographic operations to ensure feasibility on an RFID tag
  - Lightweight crypto [Juels04]
  - Often with serious vulnerabilities [Defend07, Kwon06, Li07]
  - Hard to quantify: No UHF development platform

- Maximalist Approach
  - Maximize the security on an RFID tag within given set of resources
  - Fully utilize available computational resources

WISP: Intel's Wireless Identification and Sensing Platform [Smith06]

- Powered wirelessly by RFID reader
- Implemented with microcontroller (TI MSP430)
  - 8MHz 16-bit microcontroller
  - 8KB ROM, 256 bytes RAM, 256 bytes Flash





- Follows EPC GEN1 protocol
  - ~2 msec for one interrogation
  - Transmits one 64 bit "packet" in one interrogation

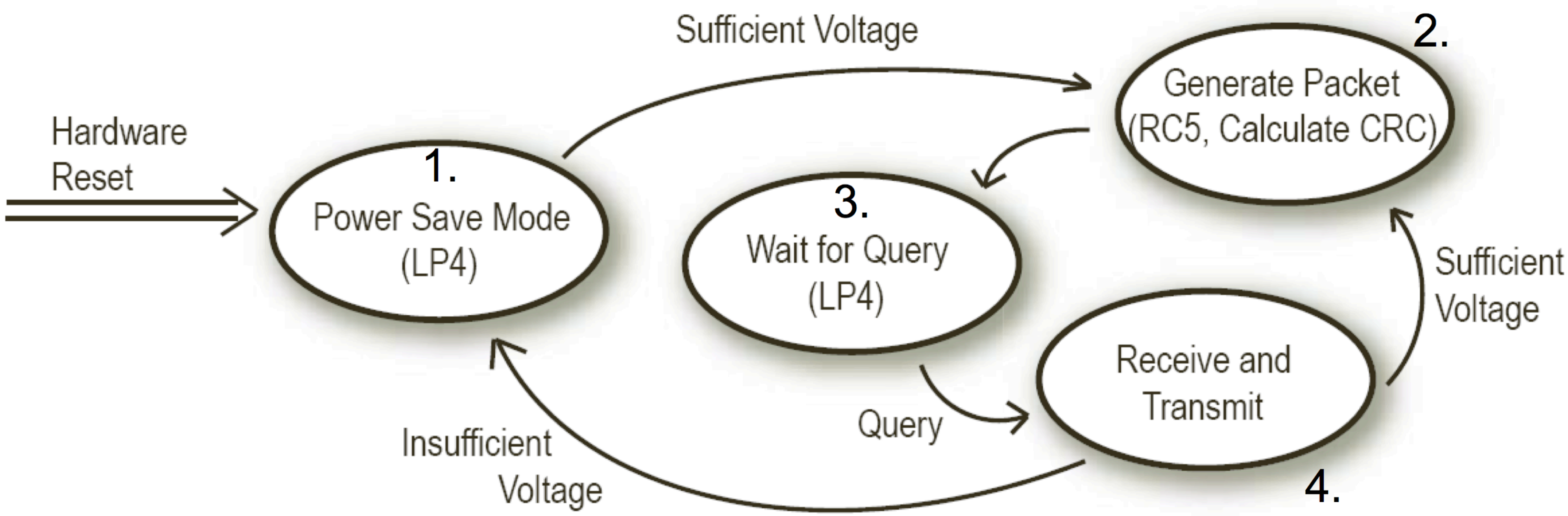
WISP Constraints

- No power source on-board
- Microcontroller runs at 4MHz with max speed 8MHz
- Small RAM – 256 bytes

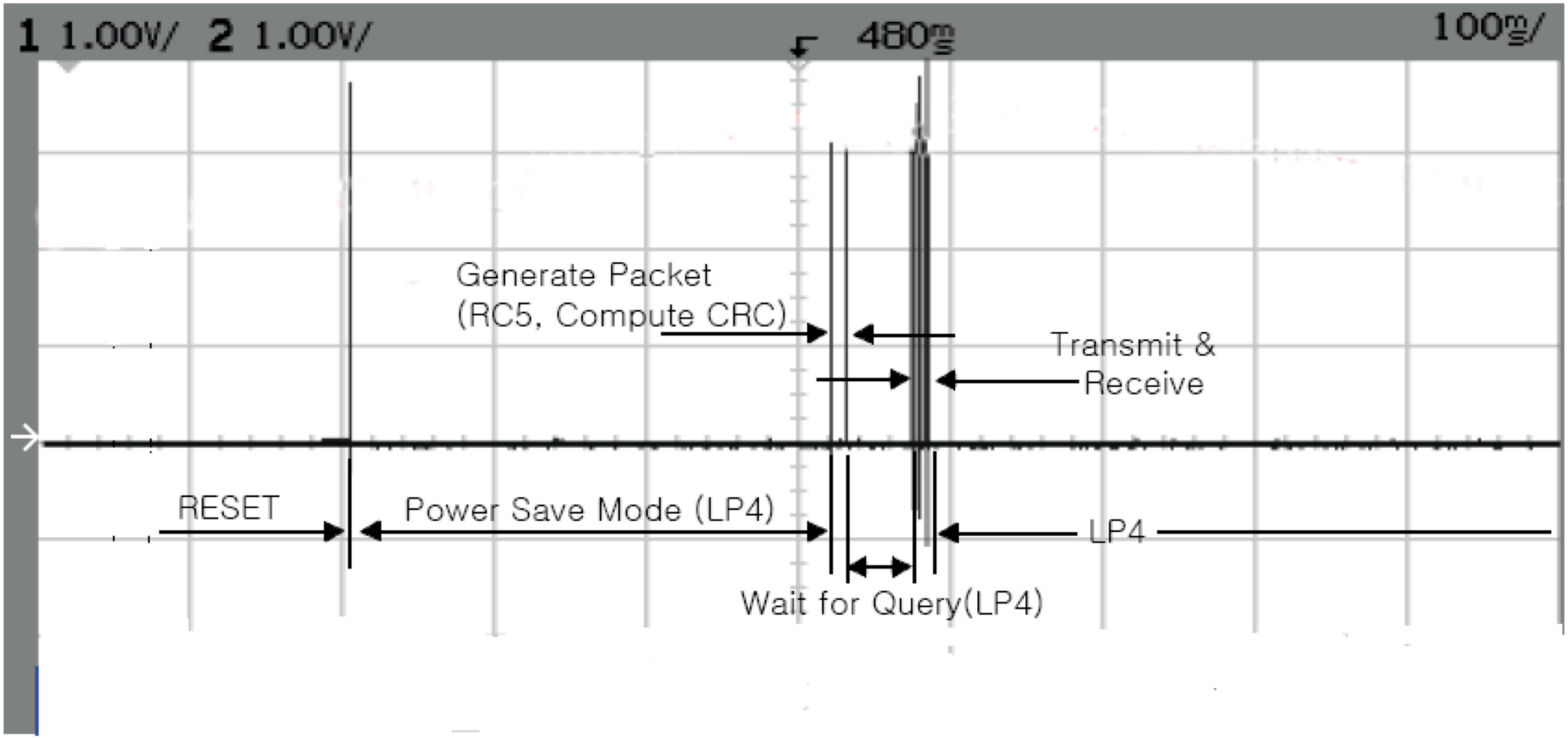
Platform	Power	Computing	Storage	Distance
WISP	UHF RF	16-bit 4MHz (max. 8MHz)	8KB ROM 256 bytes RAM	< 4.5m
EPC Gen2	UHF RF	State machine	96/128 bits	< 7.5m
Mica2	Battery	8-bit 8MHz	128KB ROM 4KB RAM	< 50m

How much computation is available in one transaction?

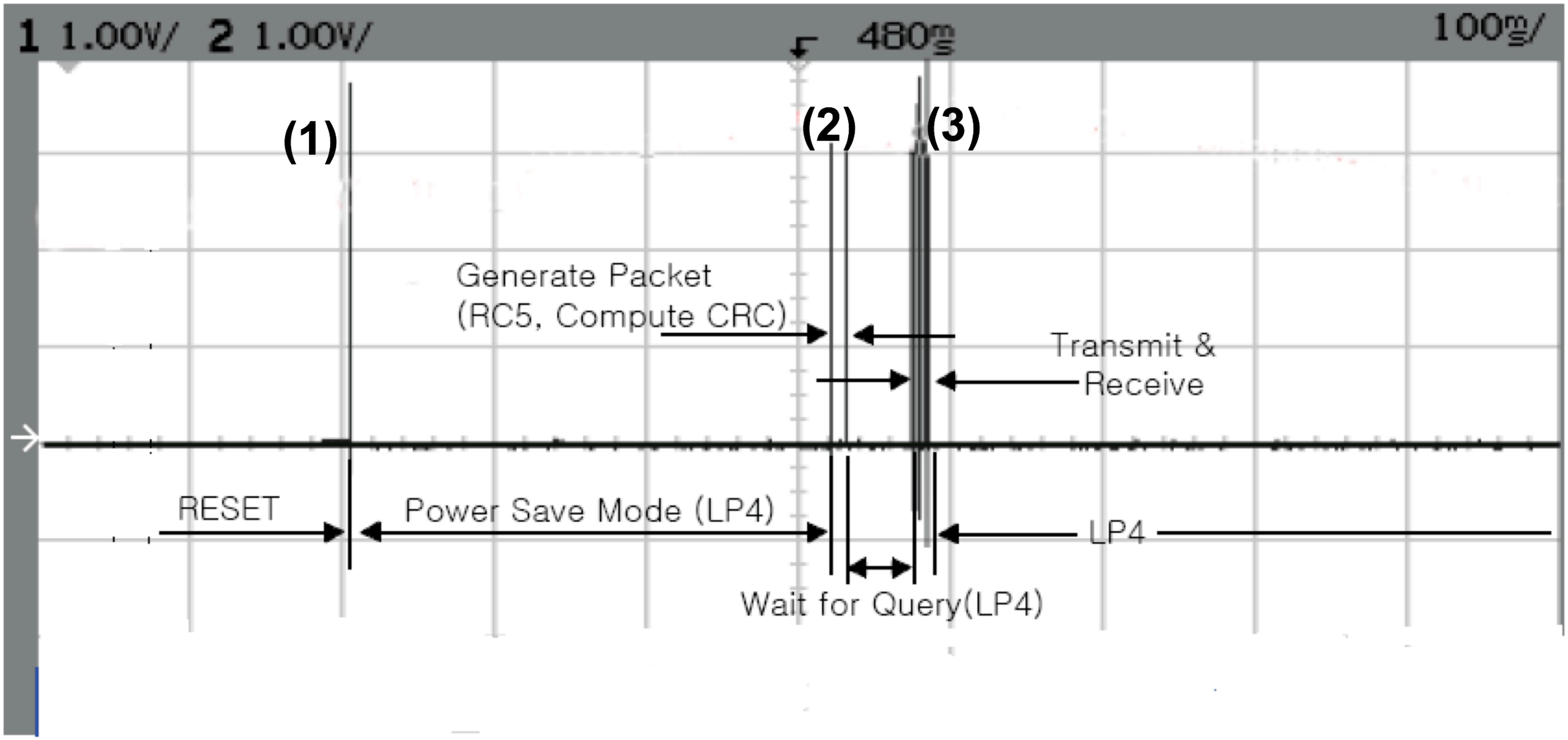
WISP Lifecycle



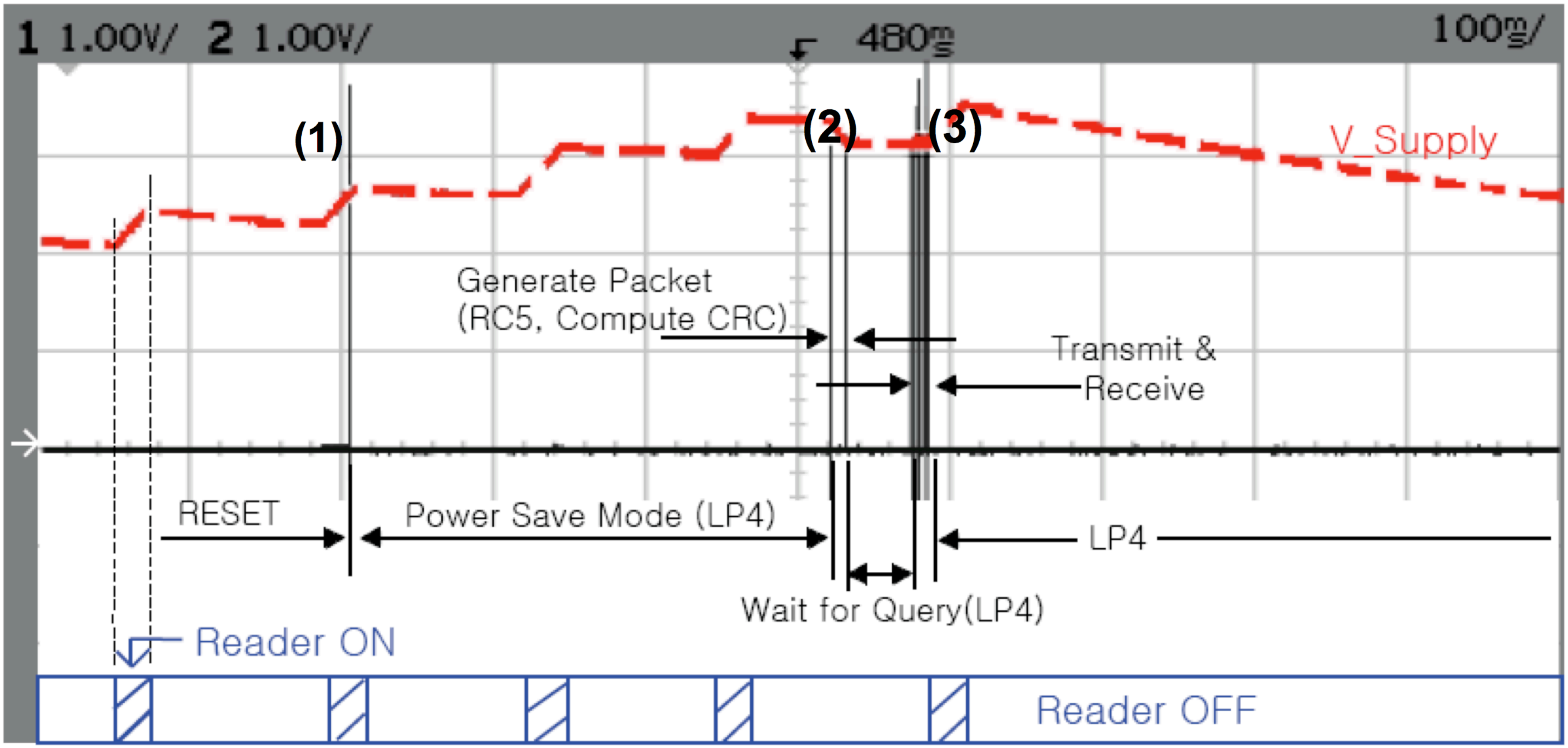
WISP Lifecycle: Scope Trace



WISP Lifecycle: Scope Trace



WISP Lifecycle: Scope Trace



Symmetric Cryptography on WISP

- Is classical cryptography feasible on a **general purpose** UHF RFID tag?

- Why RC5?
  - Simple
  - Relatively small code size (1.6 Kbytes)
  - Small memory requirement

RC5 on WISP

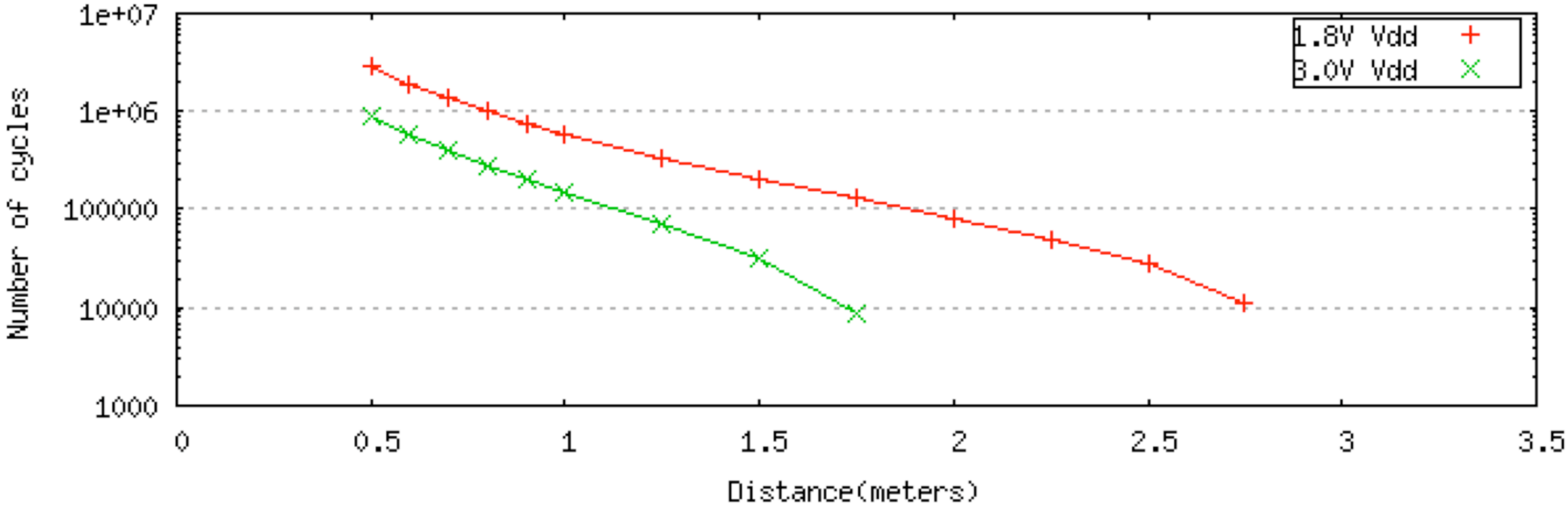
- Implementation of RC5-32/12/16
  - 32-bit words, 12 rounds, 16-byte secret key
  - 16-byte secret key hard-coded
  - Expanded key table of size 2(r+1)
  - Encrypt/decrypt 64-bit Tag ID

RC5 Functions	Execution Time (ms)	Throughput (bits/sec)
setupKey()	7.93	-
encrypt()	1.43	44755
decrypt()	1.39	46043

Can we do more?

- How much computation is available?
  - Theoretical maximum
    - Friis transmission equation
    - Experimental data on WISP performance
    - Published microcontroller power consumption specification
  - Actual WISP measurement
    - Measured number of cycles available during one lifecycle
    - With varying workloads – writing to flash

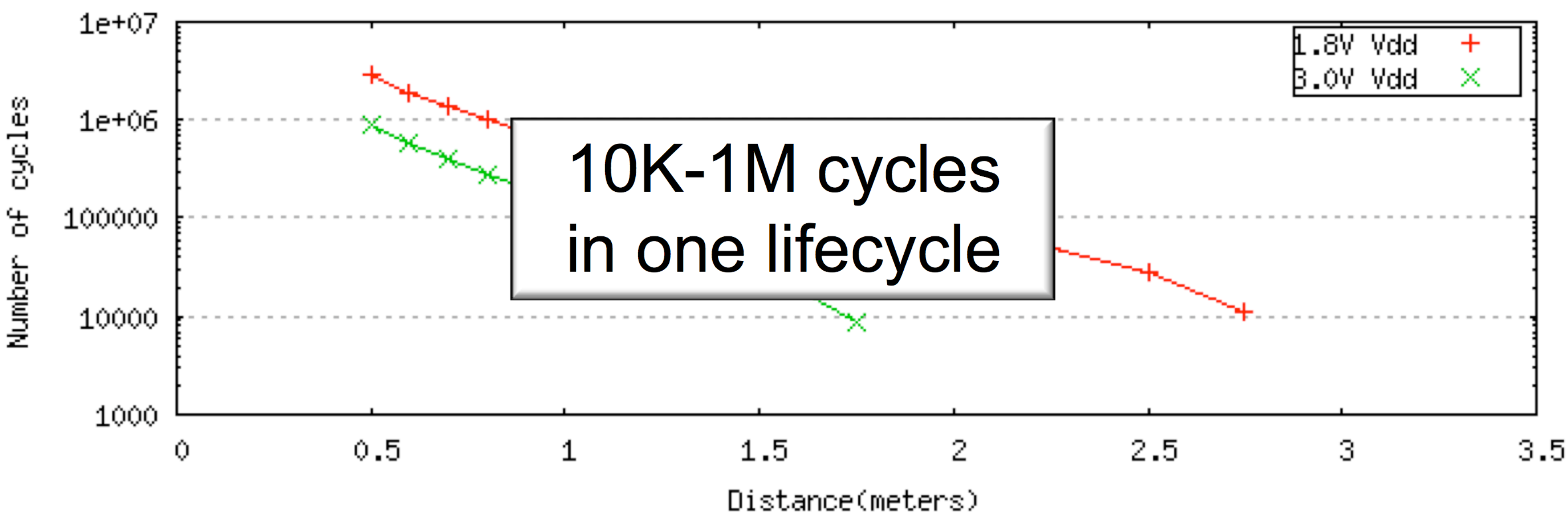
Theoretical Maximum Computation



- Model based on the Friis transmission equation

$$P_R = P_T - 20\log(4\pi d / \lambda) + G_T + G_R$$

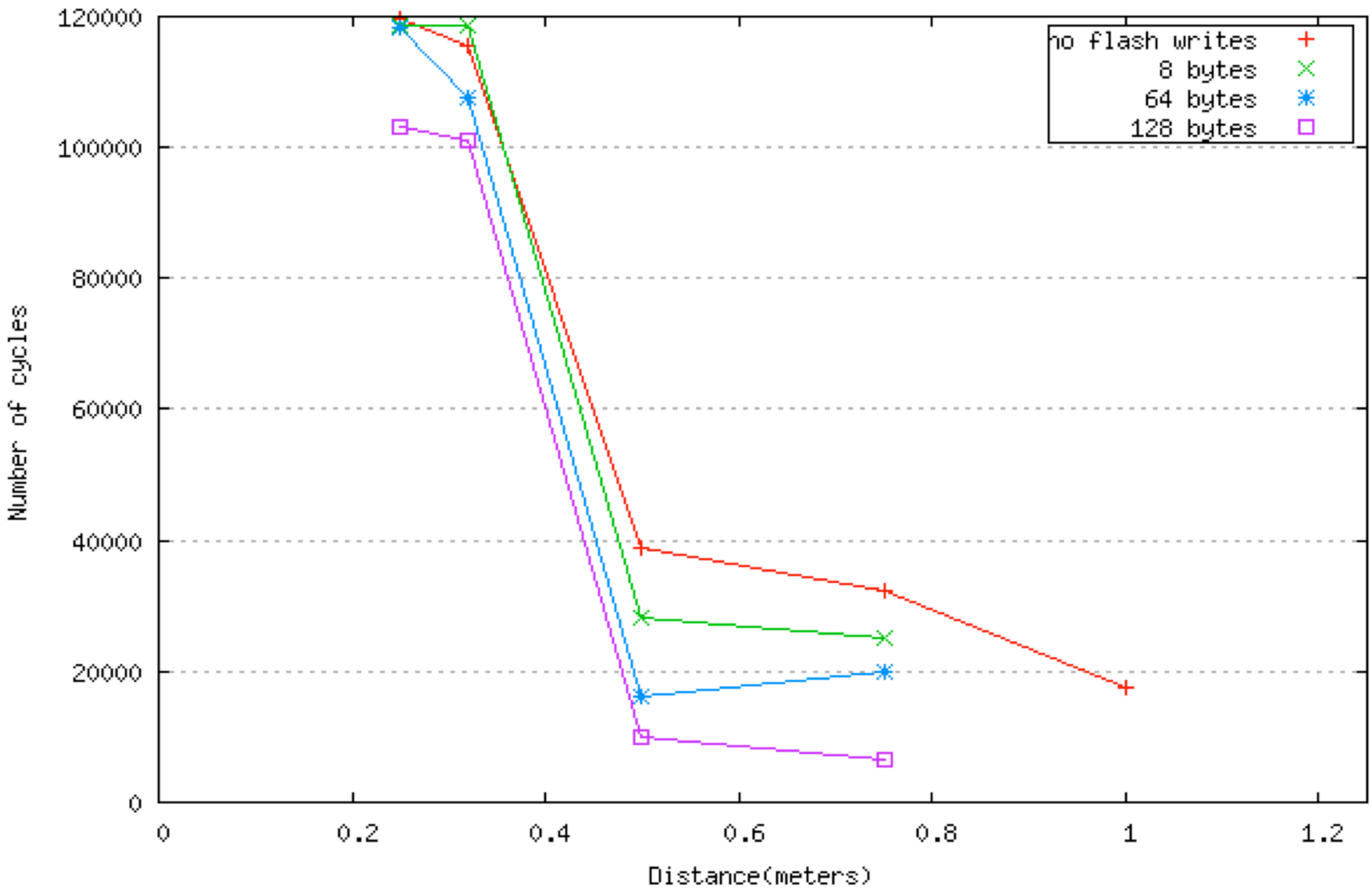
Theoretical Maximum Computation



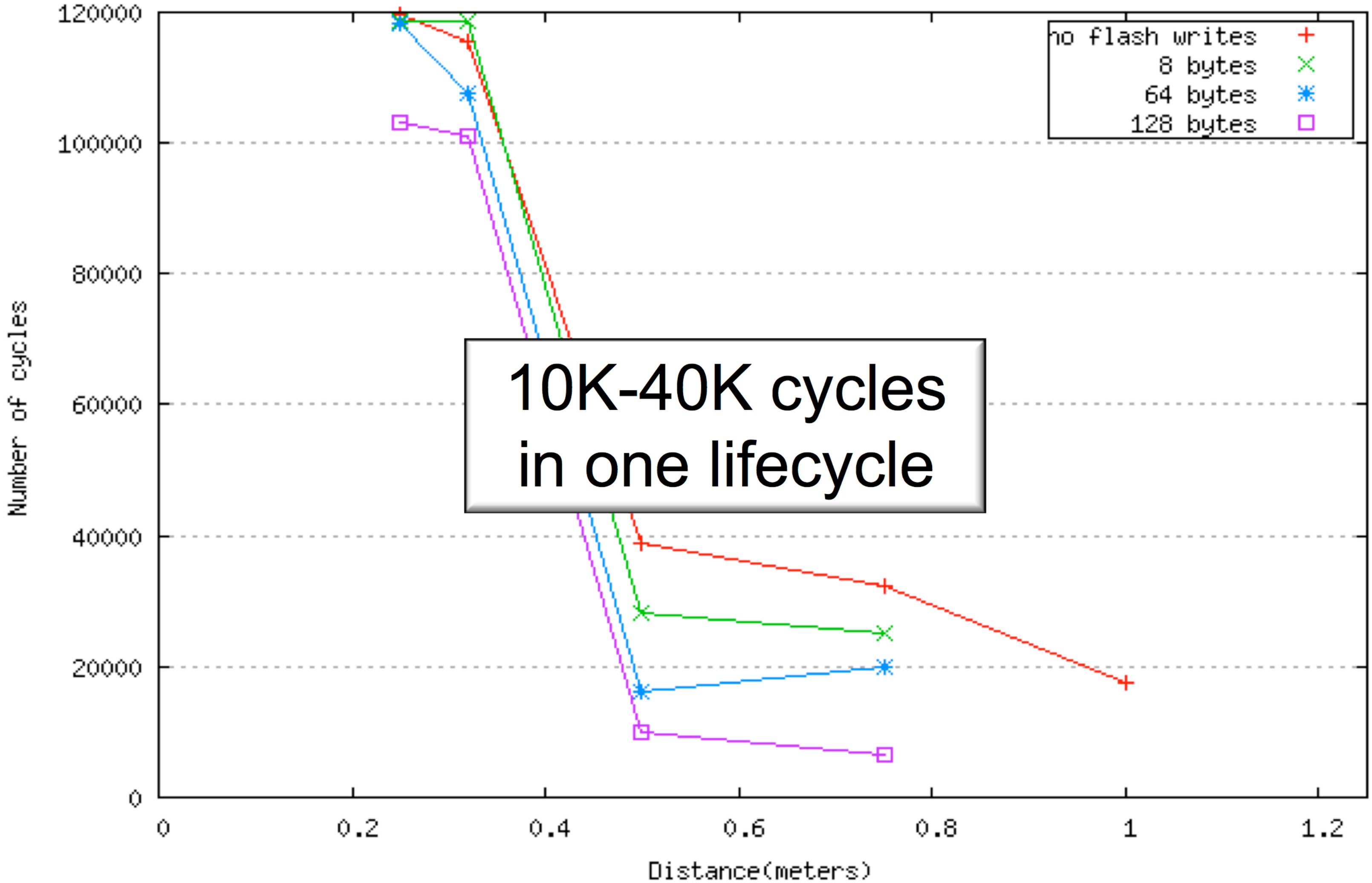
- Model based on the Friis transmission equation

$$P_R = P_T - 20\log(4\pi d / \lambda) + G_T + G_R$$

Available Computation on WISP: Varying Workloads (at 3.0V)



Available Computation on WISP: Varying Workloads (at 3.0V)



10K-40K cycles in one lifecycle

Maximalist Cryptography: difficult, but feasible

- Minimizing the stack
- Minimizing flash writes
- Optimizing with precomputation

Conclusion

- **General purpose** UHF RFID tags with cryptographic capabilities are no longer infeasible
  - RC5 can be implemented in UHF RFID tags with
    - 4MHz computing speed
    - 256 bytes RAM
- Maximizing cryptography requires model that
  - Quantifies memory (read vs. write)
  - Quantifies power
  - Quantifies computation