Mobile Energy Requirements of of the Upcoming NIST Post-Quantum Cryptography Standards

Markku-Juhani O. Saarinen
mjos@pqshield.com

PQShield Ltd. – Oxford, UK

International Cryptographic Module Conference (ICMC20)
September 24, 2020 – Virtual Conference

Dear Vendor of Things,

The NIST PQC project will finish in one or two years – we will then transition from current cryptographic standards to new quantum-secure standards !

“Sounds great! What engineering changes do I need to do?”

We will have to replace RSA and Elliptic Curve Cryptography in communication protocols and secure applications with appropriate new PQC algorithms.

“I do wireless IoT. What’s the impact on energy consumption?”

Well, some of these new algorithms are quite fast ...

“Well, my things run on batteries. What about battery life?”

¯_(ツ)_/¯ *We’ll get back to you later..*

On July 22, 2020 NIST announced...

PQ SHIELD

Seven 3rd Round Finalists:

KEM Classic McEliece *(Code-based)*
KEM KYBER *(MLWE Lattice)*
KEM NTRU *(NTRU Lattice)*
KEM SABER *(MLWR Lattice)*

SIGN DILITHIUM *(MLWE/MSIS Lattice)*
SIGN FALCON *(SIS/NTRU Lattice)*
SIGN Rainbow *(Multivariate)*

Eight "Alternate" Candidates:

KEM BIKE *(Code-based)*
KEM FrodoKEM *(LWE Lattice)*
KEM HQC *(QCSD Code)*
KEM NTRU Prime *(NTRU/RLWE)*
KEM SIKE *(Elliptic Curve Isogeny)*
SIGN GeMSS *(Multivariate)*
SIGN Picnic *(Symmetric-based)*
SIGN SPHINCS+ *(Hash-based)*

Structured Lattice Schemes Dominated

Structured lattice schemes for Key Establishment { **KYBER, NTRU, SABER** } and Signatures { **DILITHIUM, FALCON** } most likely for mobile applications.

Engineering Observations / Assumptions

September 2020: *NIST PQC candidate algorithms have been out for* $2\frac{1}{2}$ *years. These algorithms have been widely evaluated, and selection of standards is in its final phase.*
Specs & Info: `https://csrc.nist.gov/Projects/post-quantum-cryptography`

Observation 1: Little impact on symmetric cryptography (we hope)

- Most bulk data transfer still with AEADs (e.g. AES-GCM) and stream ciphers (e.g. ChaCha, ZUC). PQC Impacts mainly handshake and authentication.

Observation 2: No significant protocol re-engineering (we hope)

- Quantum-Secure Signature and Key Establishment (KEM) schemes can use similar external APIs to current standard ECDSA, ECDH, RSA cryptography.
- **Drop-in** replacement to most current public-key applications and protocols (TLS, IPSec, etc) is feasible. Just needs engineering, standardization.
- IETF and ETSI have been sitting on the fence, waiting for NIST to finish.

Talk Outline

Quick recap: Power and Energy

These physical measures are surprisingly often confused..

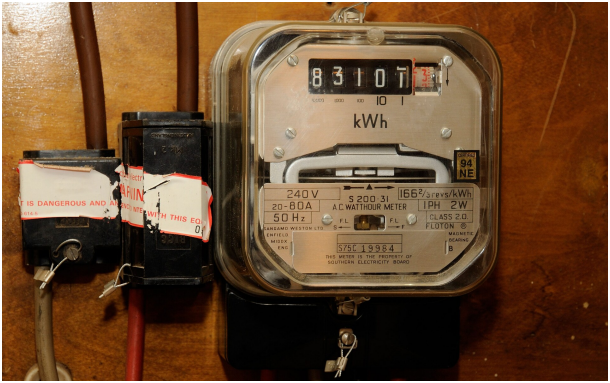
Electrical Power and Energy

$P = V \times I$ Power (W: Watt) = Voltage (V: Volt) $\times$ Current (A: Ampere)
$E = P \times t$ Energy (J: Joule) = Power (W: Watt) $\times$ Time (s: Second)

Lovely older units: Calorie (1 cal = 4.184 J), horsepower (1 hp = 764 W), etc.

- **Power** is momentary, **energy** is cumulative (*think velocity vs. distance*).
- To measure energy (J) we integrate (or "sum") power (W) over time.
- Voltage (V) is usually a known, regulated value (such as 3 V). We can use an *ammeter* to measure the current (Amps). Power is the product.

Common Derived Units



An older energy integrator (meter) using kWh = 3.6 MJ.

3.85 V × 3 Ah × 3600 s/h = 41.6 kJ.

- Derived units: e.g. **kWh** (kilowatt hour) = 1000 W × 3600 s/h = 3.6 MJ.
- Batteries are often specified in **mAh** (milliampere hour). One needs to know the voltage to compute the actual energy that the battery has.

PQC Computations: Dynamic power

From integrated circuit theory:

Dynamic Power Equation

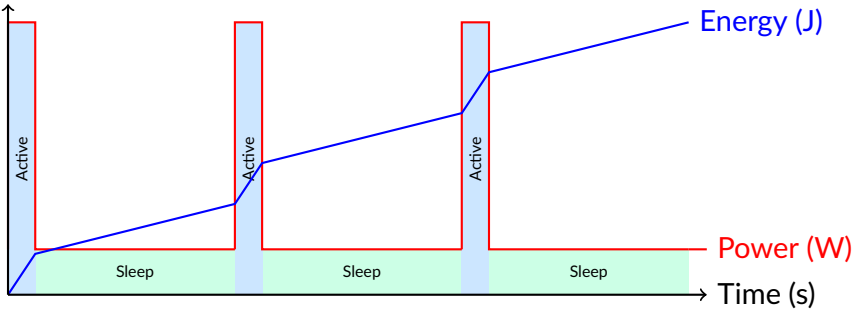
$$P_{\text{dyn}} = \alpha \cdot C \cdot V^2 \cdot f$$

P = Power, α = activity, C = Capacitance, V = Voltage, f = Frequency.

- **Dynamic power** dissipation P_{dyn} is caused by activity in the circuit.
- P_{dyn} is generally linear to frequency and area, quadratic to voltage.
- **Activity** α is sometimes called "**switching factor**" as the energy is consumed when the circuit transitions from one state to another.
- The α of a processor can vary a lot, *depending on what it is doing*.
- **Static power** dissipation P_{stat} when the circuit is idle. $P = P_{\text{stat}} + P_{\text{dyn}}$.

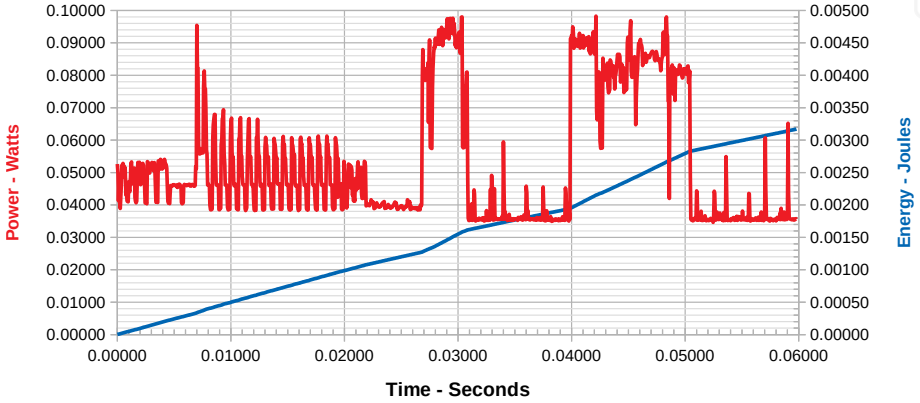
Sleep States and Power Management

- Most CPUs have one or more **sleep states**, also affecting peripherals.
- When asleep, instructions are not executed: P_{dyn} is very low.
- MCUs typically wake up only via an **interrupt** f (timer or external event).
- Modern CPUs can also control ("scale") their clock frequency *f* (and *V*).



A typical power consumption model for IoT microcontrollers ("sleepy edge node").

Reality: 60 ms of NTRU-HPS 4096-821



- This is what "**active**" can look like on a real MCU (ARM Cortex M4).
- Power is rapidly fluctuating between 35mW and 100mW (3× range).
- Cycle count is clearly **not** telling the full story about this algorithm.

Talk Outline

Cortex M4: Post-Quantum Power Sandwich

PQSHIELD

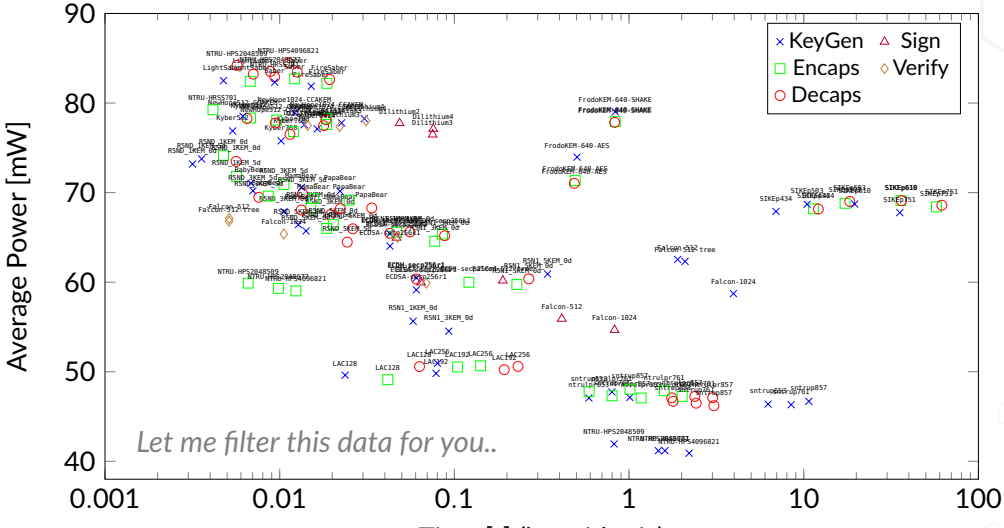
New PQC "IoT" Energy Measurements

- LPM01A "PowerShield" £50 power measurement board is also used for the EEMBC IoTConnect™ benchmarks.
- STM32F411RE target has a Cortex M4 core, the reference embedded platform of NIST PQC project.
- I measured PQC implementations from the PQM4 project, also Ken MacKay's "micro-ecc" ECDSA & ECDH code.
- **Goal:** *Precise, independently repeatable.*

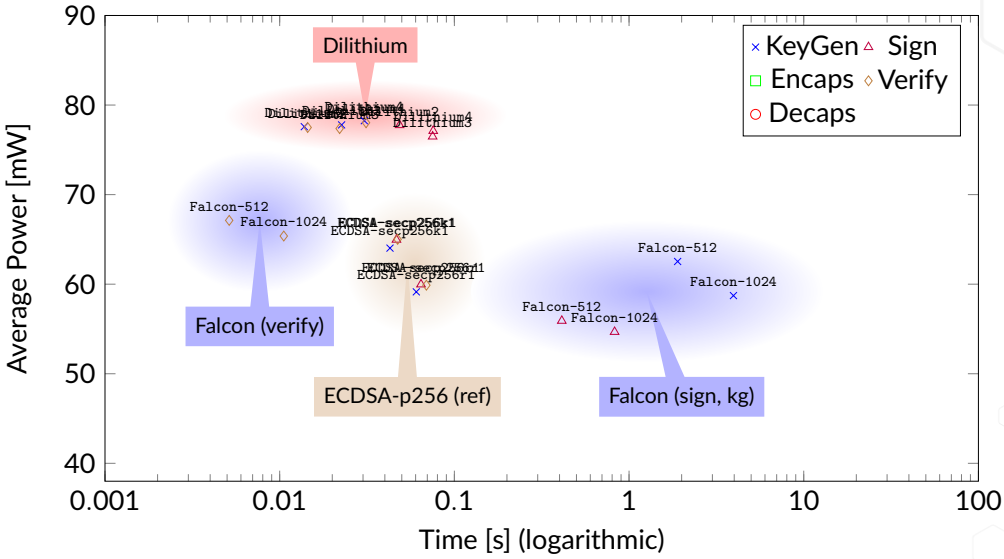
Source code and a description of the lab:
`https://github.com/mjosaarinen/pqps`

It's a dev board sandwich: LPM01A sits on top of the Nucleo64 target.

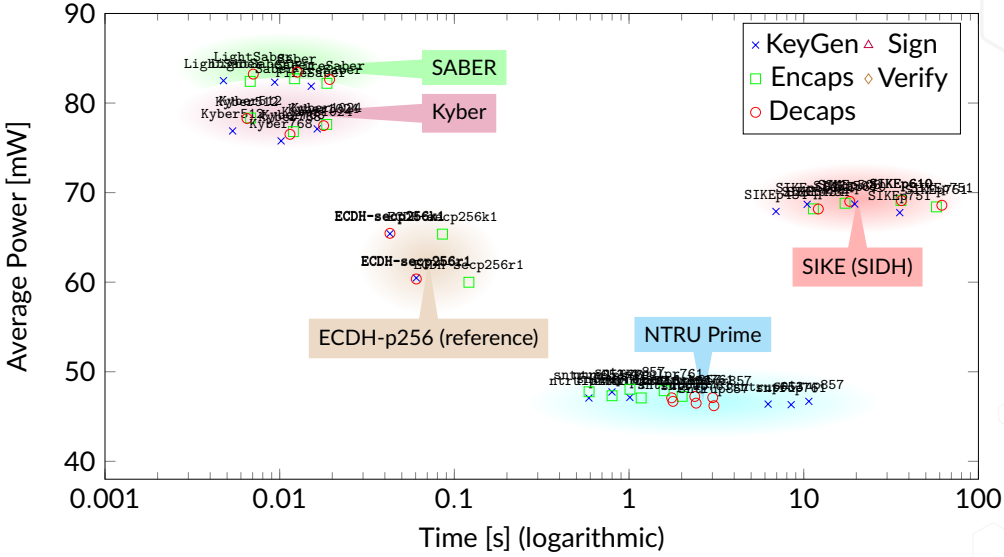
STM32F411RE: Round 2 Average Power @ 96 MHz



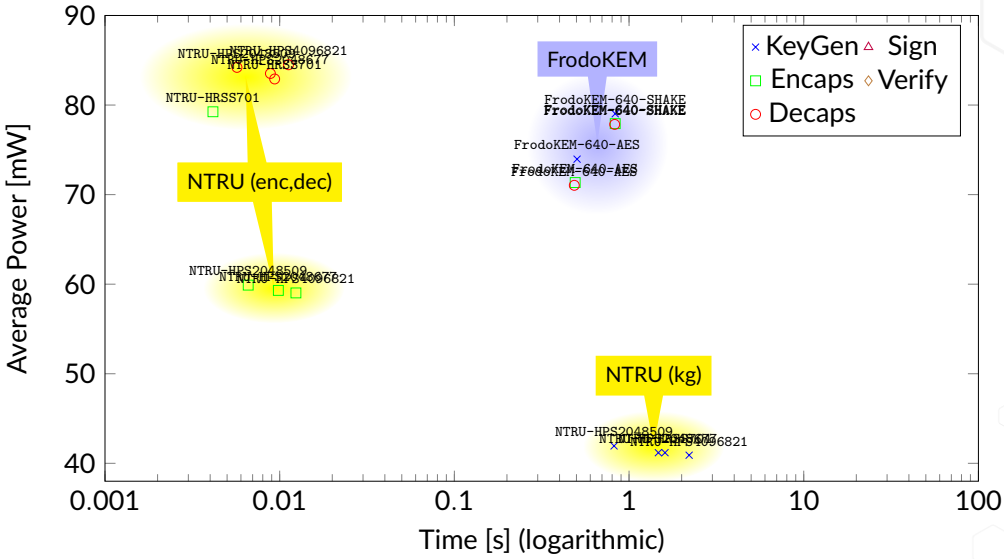
Cortex M4: Round 3 Signature Algorithms



Cortex M4: Distinctive R3 KEM Clusters (1/2)



Cortex M4: Distinctive R3 KEM Clusters (2/2)



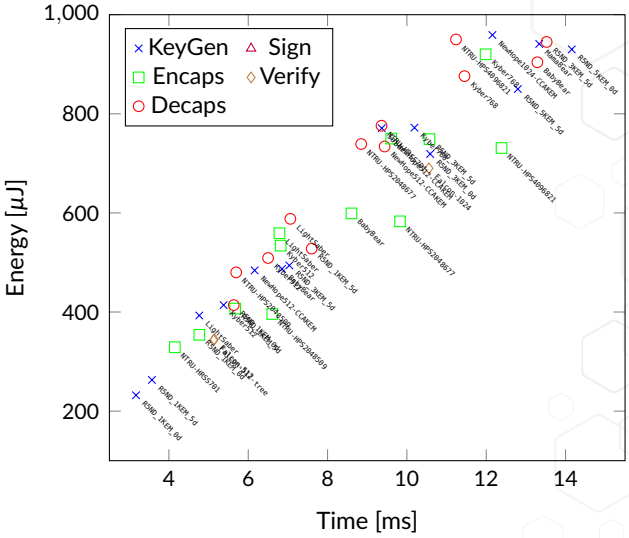
Cortex M4: Energy vs Time – Sub-mJ (μJ) Range

In the microjoule range there are many examples where algorithm's rank by timing is different from rank by energy.

Meaningful, but:

Are there general techniques to trade power for time?

Do the very distinctive power profiles translate to other microcontroller targets?

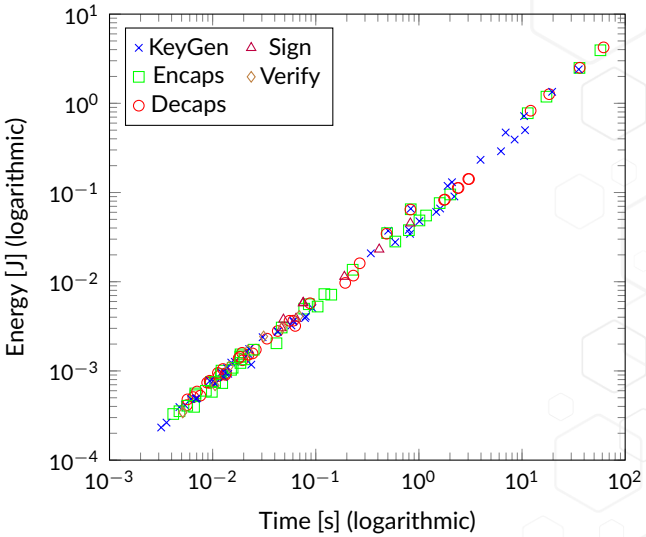


Cortex M4: Energy vs Time – Whole Range

A log-log plot of the NIST PQC 2nd round set looks quite linear.

There is a range of over **four orders of magnitude** in the complexity of PQC algorithms.

This completely dwarfs the observed ≈ 50% range in power. So “cycle counts” can be used to estimate energy, but results have **less than one significant digit of precision.**



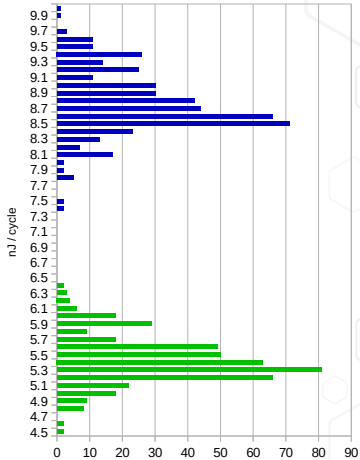
Intel PCs: RAPL (Running Average Power Limit)

Intel PC/Server Measurements

- Inspired by [1], I modified the "official" SUPERCOP benchmarking system to record energy usage via Intel's RAPL.
- Profiled 159 variants of about 20 NIST PQC algorithms in the benchmark.
- Power is **highly dependent on target**, but within that target **not as varied** as with IoT MCUs. Platform $[nJ/\text{cycle}]$ and cycle count leads to a good estimate.

[1] C. A. Roma, et al.: *"Energy Consumption of Round 2 Submissions for NIST PQC Standards"*, NIST PQC 2019.

[2] Source code in used in our work: `https://github.com/mjosaarinen/pqps/tree/master/suppercop`



Blue: *i7-8700 @ 4.6 GHz (Desktop)*
Green: *i5-8250U @ 3.4 GHz (Laptop)*

Talk Outline

A *very* Simplistic Model for Wireless

All PQC candidate algorithms have larger key- and message sizes than current RSA and Elliptic Curve cryptography. **How much is too much?**

Total Energy: Compute it + Transmit it

E_{KG}	+	$e_{tx}\|\text{pubkey}\|$	Key generation.
E_{Enc}	+	$e_{tx}\|\text{ciphertext}\|$	Encapsulation.
E_{Sign}	+	$e_{tx}\|\text{signature}\|$	Authentication.

.. or whatever is relevant in the protocol in question.

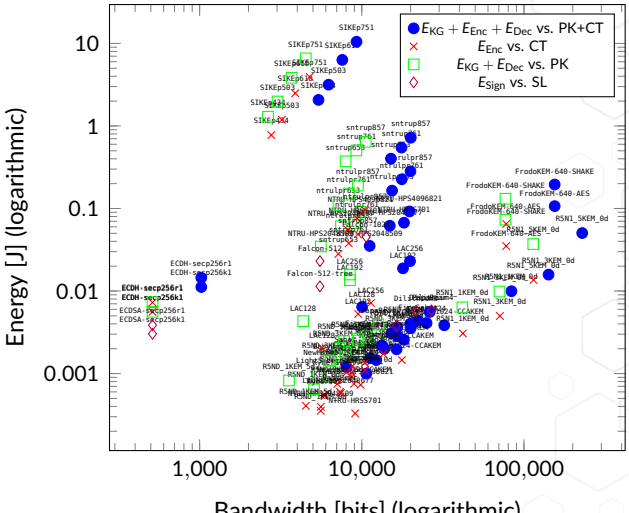
- Uplink (transmit) energy $e_{tx} > e_{rx}$ downlink (receive), both in $^{\text{Joule}}/_{\text{bit}}$.
- **Two** algorithm factors (complexity, message sizes) and **two** platform factors (computational efficiency and communication efficiency).
- We need **measurement data** to determine their relative importance.

Energy and Message Lengths: “It Depends”

KEMs: Need to relate KG, Enc, Dec computation energy to transmit energy of Public Key and Ciphertext (PK, CT).

For *ephemeral key exchange* we may consider total energy $E_{\mathrm{KG}} + E_{\mathrm{Enc}} + E_{\mathrm{Dec}} + e_{\mathrm{tx}}(\mathrm{PK} + \mathrm{CT})$.

For signer (auth): Consider $E_{\mathrm{Sign}} + e_{\mathrm{tx}} \cdot \mathrm{SL}$, where SL is signature length. **Verifier** E_{Ver} also downloads (e_{rx}) some certificates which have public keys and signatures.. depends.

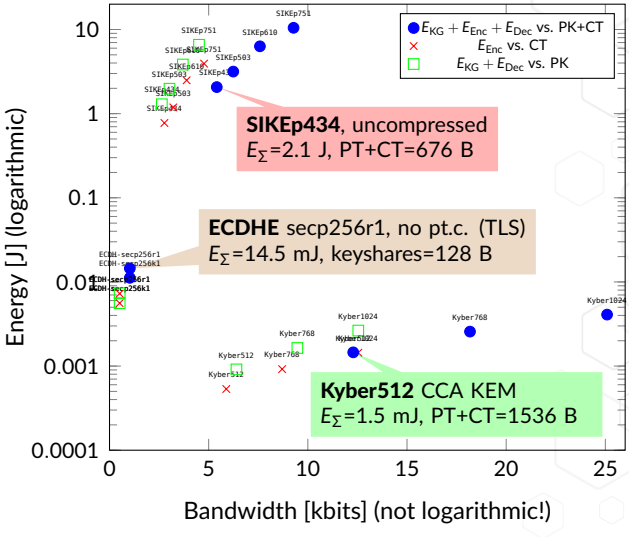


Example: Ephemeral Key Exchange

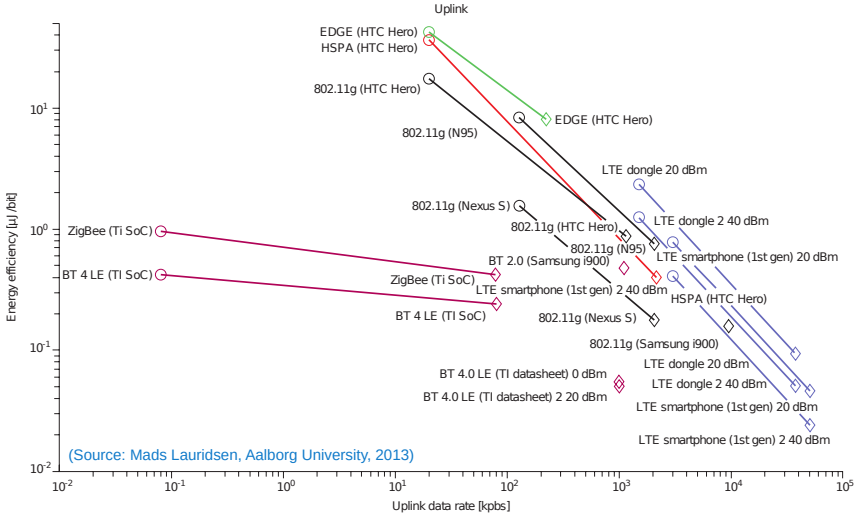
Use Case Observations

- PQC Lattice (RLWR) can be $\approx 10\times$ more efficient than current ECDHE but needs $\approx 10\times$ bytes.
- PQC Isogeny (SIDH) needs 30-60% of RLWR Lattice bytes, but need 100-2000 times more energy.

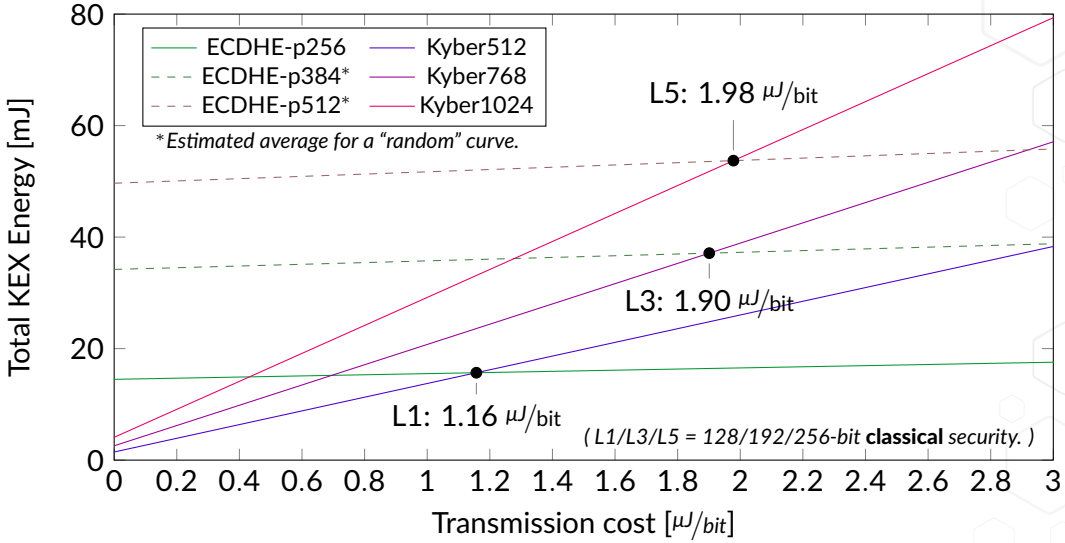
We can determine energy **crossover points** for e^{tx} $[J/\text{bit}]$ from lattice schemes to, say, elliptic curves and SIDH.



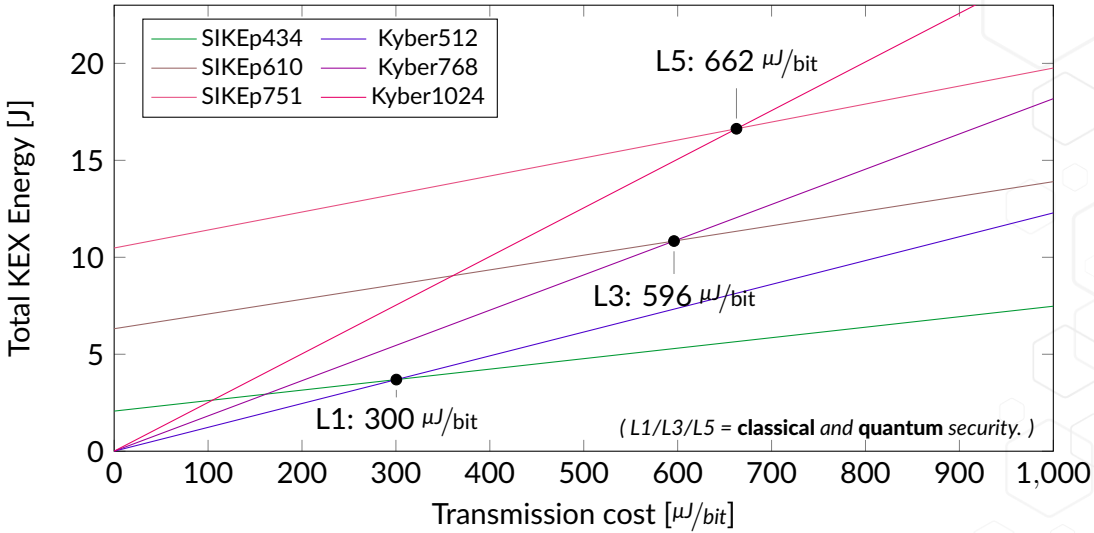
Communication efficiency $e_{tx} \approx 0.1^{\mu J}/_{bit}$ (pre 5G)



Energy Opt. Crossover from Kyber to ECDHE at $e_{tx} \gtrapprox 2^{\mu J}/_{bit}$



Energy Opt. Crossover from Kyber to SIKE at $e_{tx} \gtrapprox 500\mu J/bit$



Talk Outline

Conclusions

- PQC algorithms have *really* distinctive “IoT” MCU power profiles !
- Energy usage range is **4 orders of magnitude** in NIST PQC 2nd round.
  - Energy ranking differs from time ranking most with $< 1mJ$ algorithms.
  - Some PQC schemes actually need **significantly less energy** than ECC.
- Not so much power variation in our Intel PC / Server measurements.
- Consider **transmission cost** [J/bit] and **computation cost** [J/cycle].
  - ECC still has lower energy for (low bandwidth) channels $e_{tx} \gtrapprox 2\ ^{\mu J}/_{bit}$.
  - Isogeny/SIDH bandwidth savings are unlikely to lead to energy savings since computation cost dominates until $e_{tx} \gtrapprox 500\ ^{\mu J}/_{bit}$ (very high).
  - Plotting **total energy** cost as a function of e_{tx} is a good way to compare algorithms A and B, where other is faster but needs more bandwidth.

Thank You!