Viability Of Energy Harvesting For Cellular IoT

4th July 2019

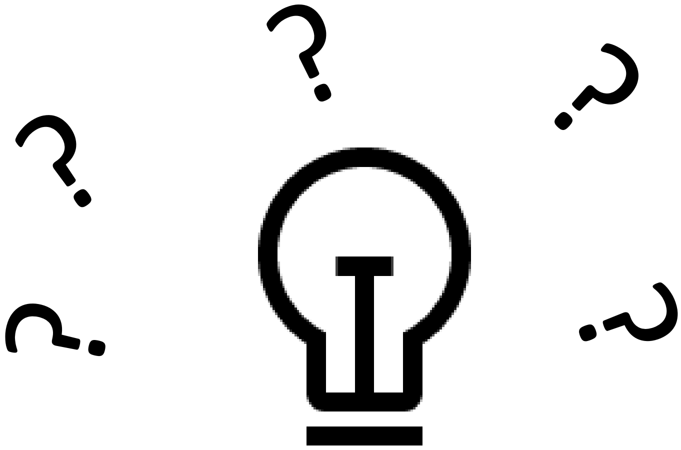
Rob Meades, u-blox Cellular Technology

Hi, my name is **Rob Meades** and I work in the **Cellular Technology** group in **u-blox**. u-blox makes **modules** for GNSS, cellular, Wifi and Bluetooth.

My group does the **wacky** stuff, things that **no-one else** has the **justification** to do, I work mainly on **embedded software**.

What If Your IoT Device Needed No Battery?

“Asymptotic To”



In one of those “out there” **brainstorming** moments the idea “**what if your IoT device needed no battery?**” came up.

Of course, **no-one is getting rid** of the battery any time soon, **storage** is always required and **batteries** are a good way to **store** energy. It is probably better to think, as your **calculus teacher** will have said, **“asymptotic to”**, **“tending towards”** batteries **not being required**.

Connected How?

Power Requirements

4 dBm

12 dBm

23 dBm

500 mA

3 u-blox AG

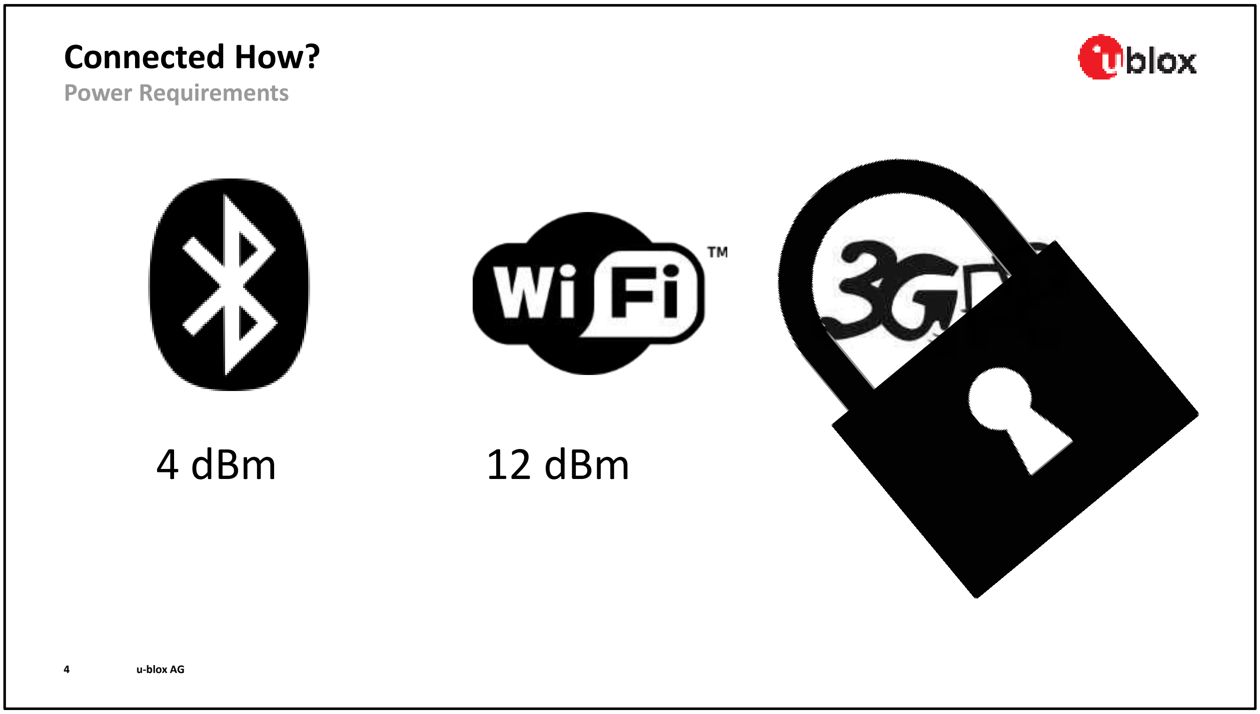
When you read articles about energy harvesting powering **communications** devices the **means** of communication is not always front and centre, which is odd since **RF** is usually the more **power-hungry** thing in the system.

Generally the means seems to be **Bluetooth**, in which the **normal** operating mode has a **max Tx power** of **4 dBm**.

At a stretch, maybe a device will use **Wifi**, which requires **eight times** more power at **12 dBm**.

But I'm in the **cellular** technology group, so the **lowest** power I have to play with is **NBIoT**/**Cat-M1**, where the max Tx power is **23 dBm**, **sixteen times** more again.

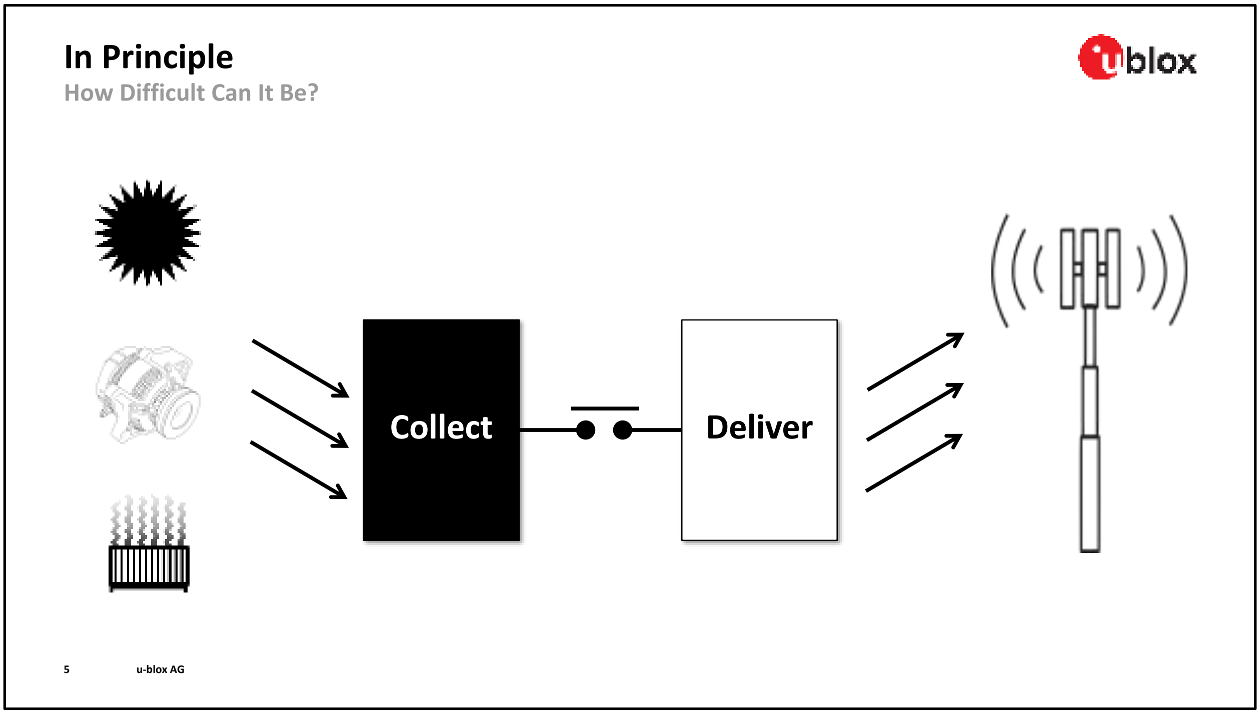
Worse than that, the **peak power** requirement, for a **very short** period of time of course, can be up to 500 **mA** into the **power amplifier**, **stressing** the power source of an **IoT** device.



The other **subtle** difference between these communication schemes, which we trip-up on with customers that are moving their devices' **communications** path from the **left to the right**, is that under **3GPP** the **network** is in **control**. The device **must do**, is **conformance tested** to do, what the **network dictates**: number of transmit repetitions, etc.

We are frequently asked: can I not just **stop transmitting**, change my mind, take **control** of **exception cases**, 'cos I really **care** about the **power consumed**. Unfortunately that's **not usually possible** with 3GPP.

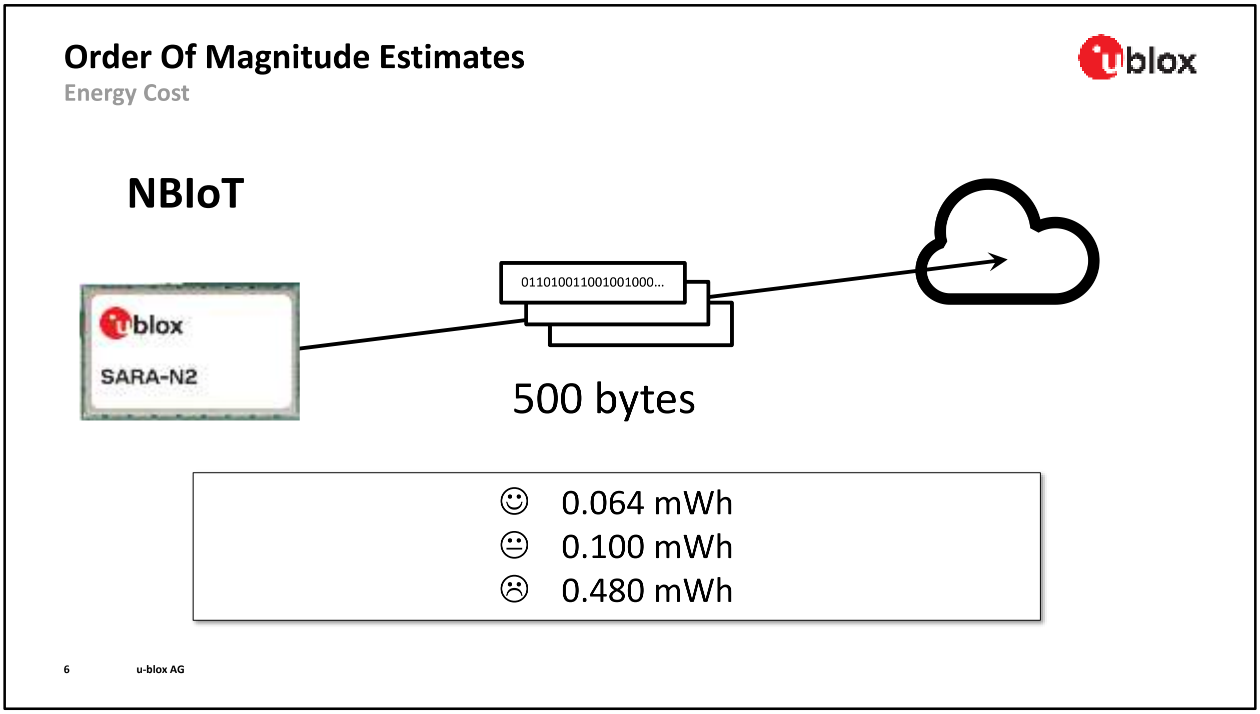
In fact that is **often why** the customer is making the move from the left to the right: **licensed spectrum** is well **behaved** and of high **quality**; that brings with it necessary **constraints**.



Anyway, I thought: **how difficult** can it be? It **doesn’t matter** that energy is coming in **slowly** if you have an **efficient**, **low leakage**, **low voltage**, **collection** system.

It **doesn’t matter** that **cellular** has high a **peak current** requirement if, when there is **sufficient** collected energy, you can **transfer** it into a **delivery** system that can feed the **peaks**.

But this is a **software engineer** talking…

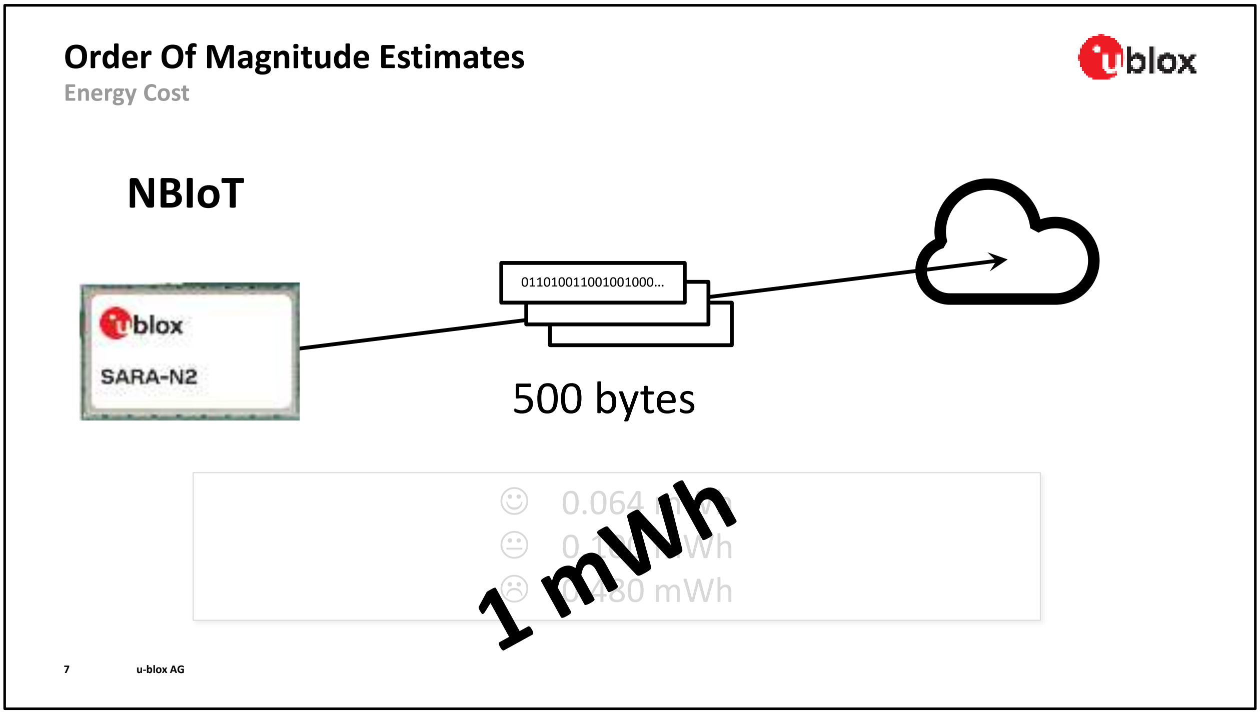


I made some **order of magnitude** estimates to see if it was even **worth starting**.

In terms of **energy cost**, taking one of our **SARA-N2 NBIoT** modules, we know the amount of **energy** required to **register** with the network, **transmit** 500 bytes to a server on the public internet and **disconnect** again.

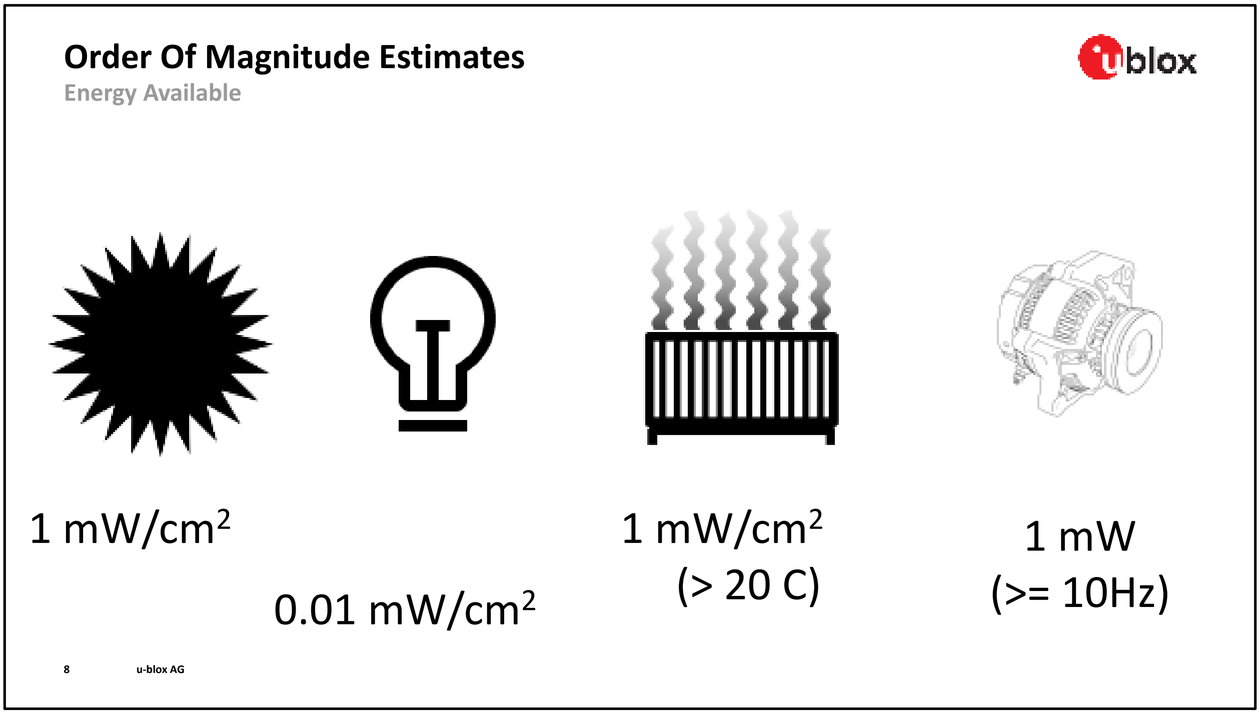
These are the **numbers** for good, OK and bad conditions: a **factor of ten** difference between one end and t'other.

Note that in the **bad** case the number can be **significantly worse**; because it is very bad it is just very **variable**.



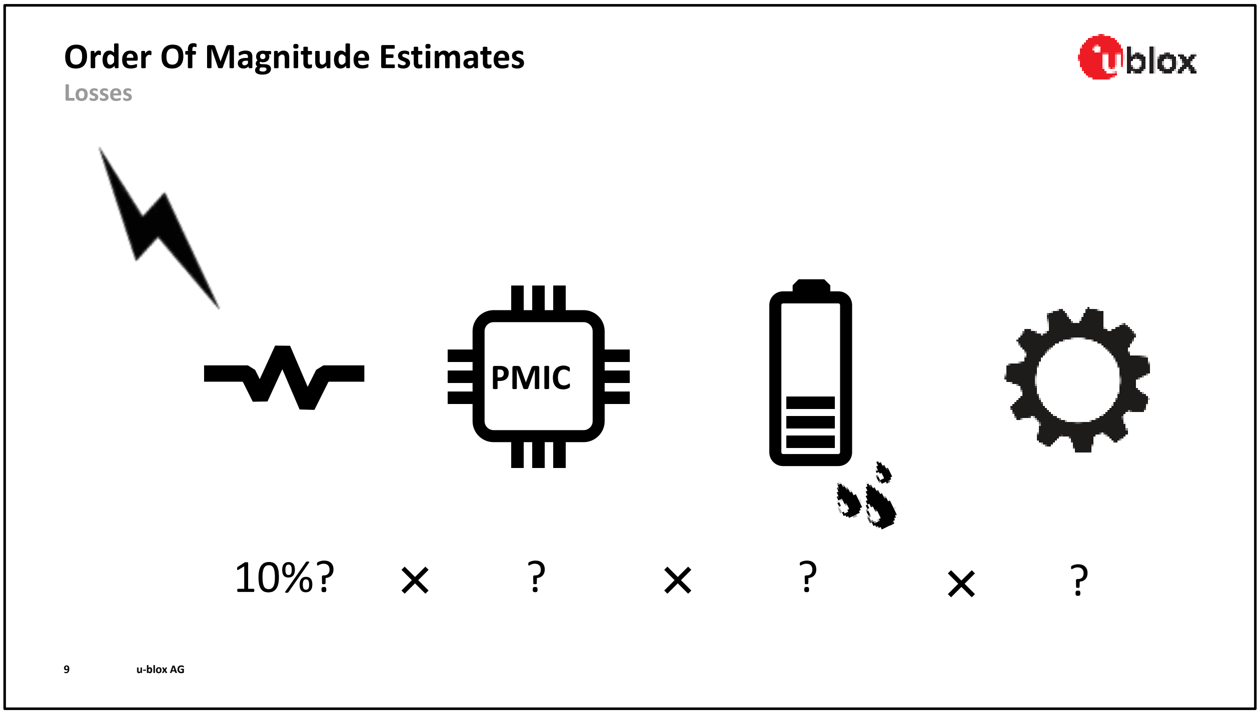
Sticking to **round numbers**, allowing for **truly bad** conditions, we set a **budget** of **1 mWh** to **register**, **transmit stuff** and **disconnect** again.

This assumes that the **interval** between wake-ups is so **long** that it is worth switching the cellular device **off**, **rather than** going to **3GPP sleep** (see later) and that the **protocol/server** is fairly **nippy**, not leaving us hanging around with the cellular modem’s **receiver switched on** unnecessarily, sympathetic to **low power operation** (acking efficiently etc., UDP v TCP).



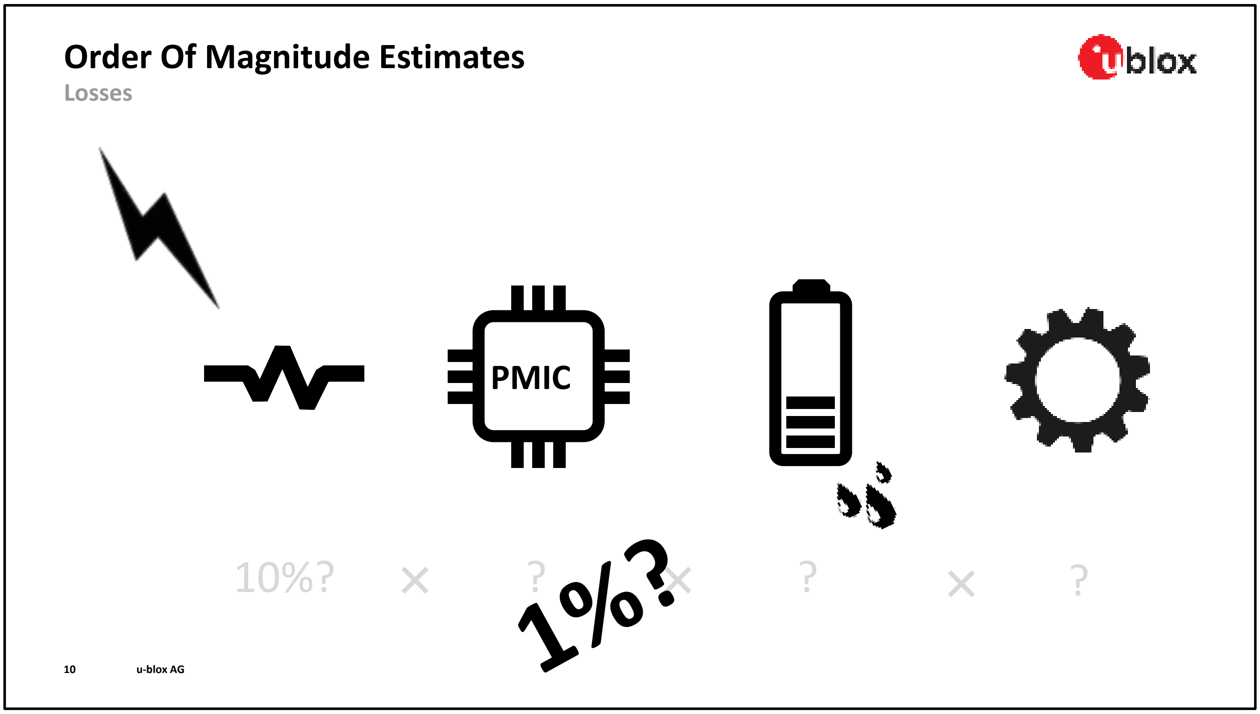
In terms of **energy availability,** a search for papers on the **internet** gave me the numbers above. Note, in particular, the **difference** between **sunlight** and **indoor** light, **100 times** less.

These numbers may **not be accurate**, and **technology** is always **moving on**, but I was happy to **believe** the **orders of magnitude** (maybe because they are all 1).

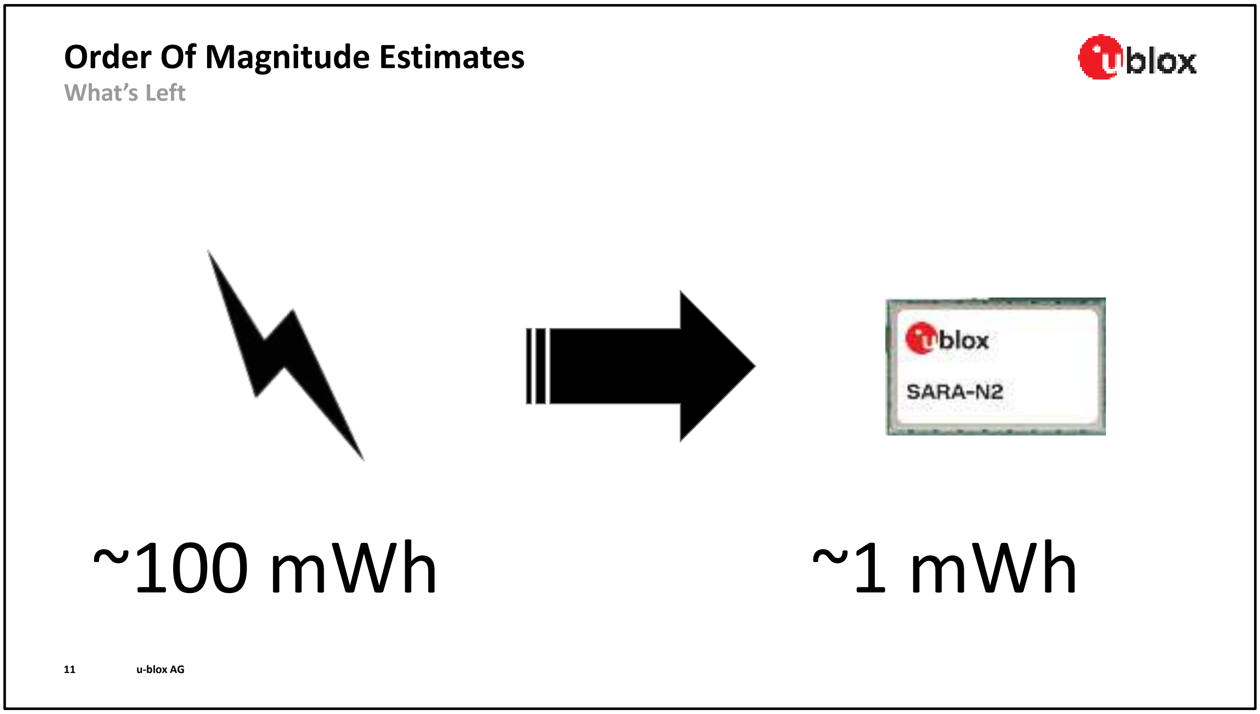


We also have to consider the **efficiency** of the components that convert incident **environmental energy** into **usable electrical** energy.

10% efficiency for a **transducer** is good, then there's a **PMIC**, a **leaky battery** and, of course there's some **application** or other with the audacity to consume power ***as well*** as the cellular module.



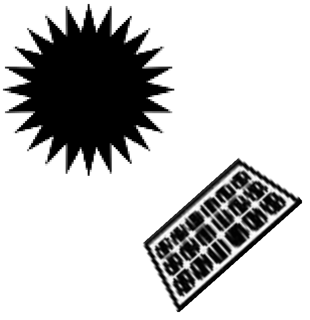
Maybe all of this stuff means we only get to see **1%** of the incident **environmental energy**.

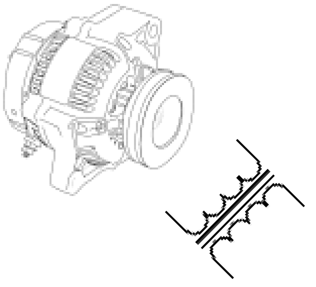


So to get **1 mWh** for the **cellular** modem we maybe need **100 mWh** from the **environment**.

Order Of Magnitude Estimates

Adding Up





10 cm^2?

20 Hz?

Duty Cycle ~1 - 2 days

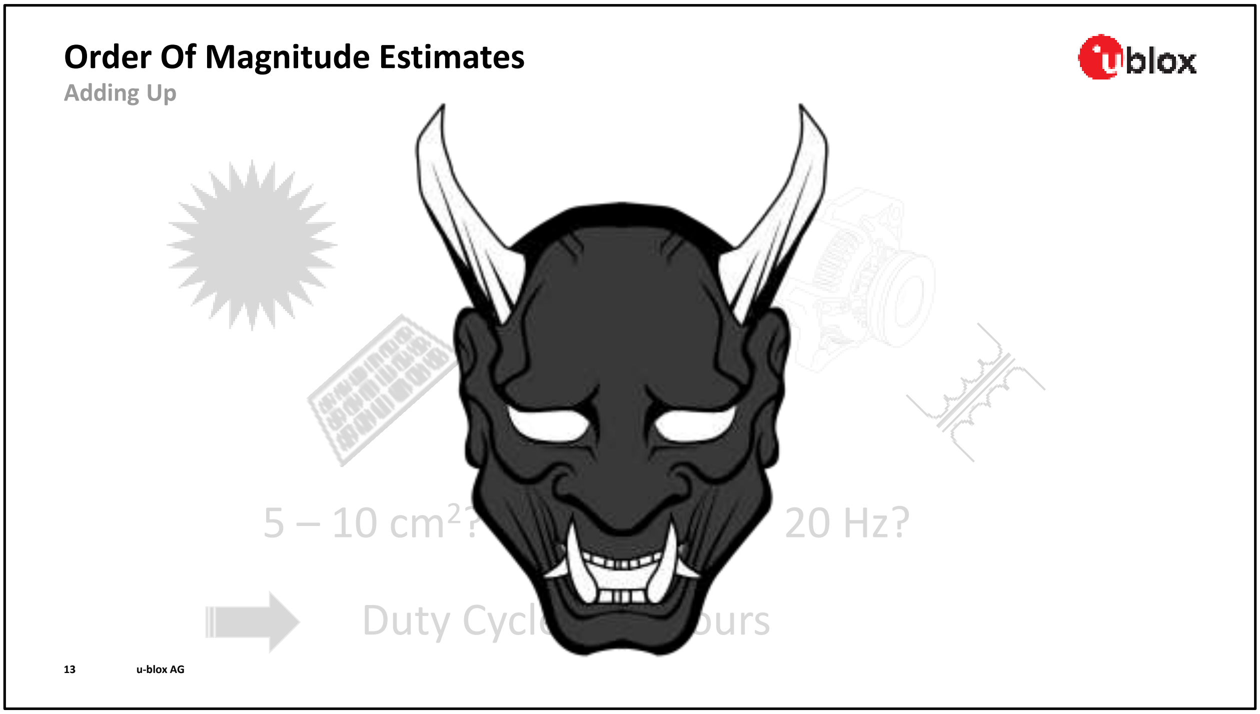
12 u-blox AG

Based on the **previous order of magnitude** estimates then maybe a **10 cm^2 solar** cell or the **vibrations** from a **1200 RPM** electric motor might be enough for us to **report** every **one or two days** in **worst case** conditions.

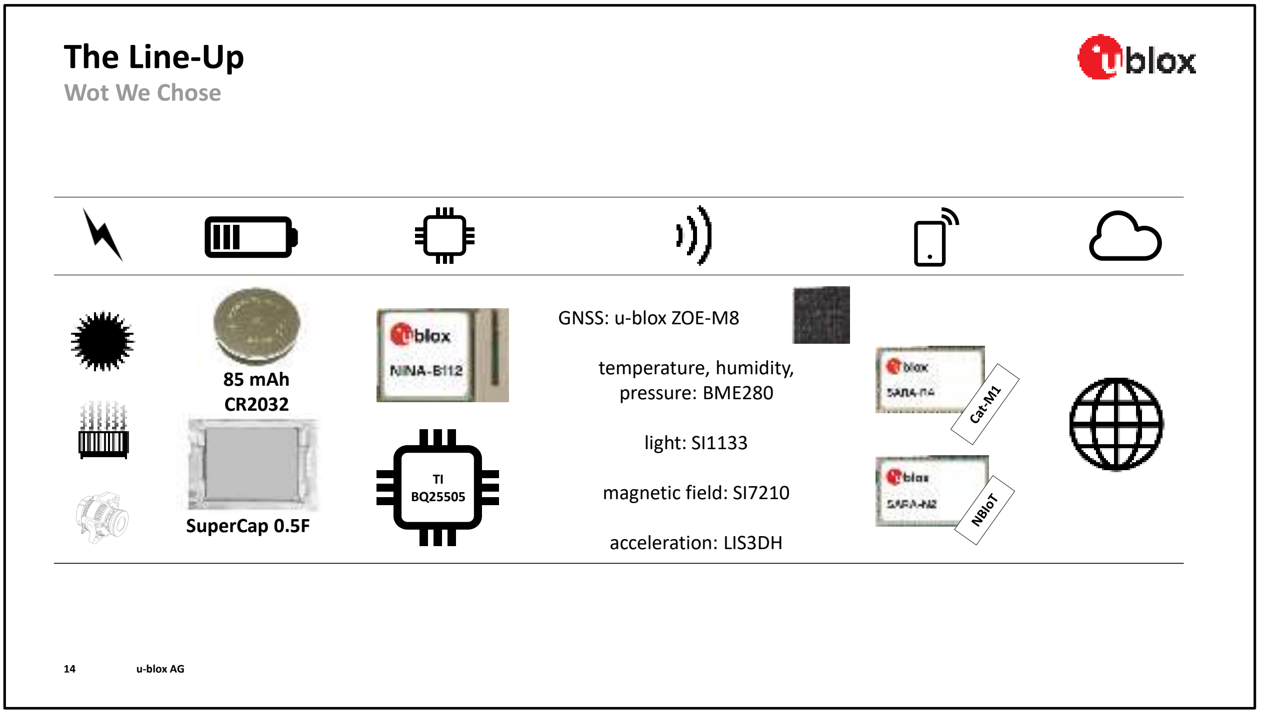
So it is **possible**, it is worth **having a go**.

1 mW/cm^2 x 10 cm x 10 hours = 100 mWh

1 mW x ((1200 RPM / 60 seconds) / 10 Hz) x 24 hours = 48 mWh



Enough **spreadsheets/PowerPoint**: as always the **devil** lies in the **detail**, let's do it **for real** and find out.

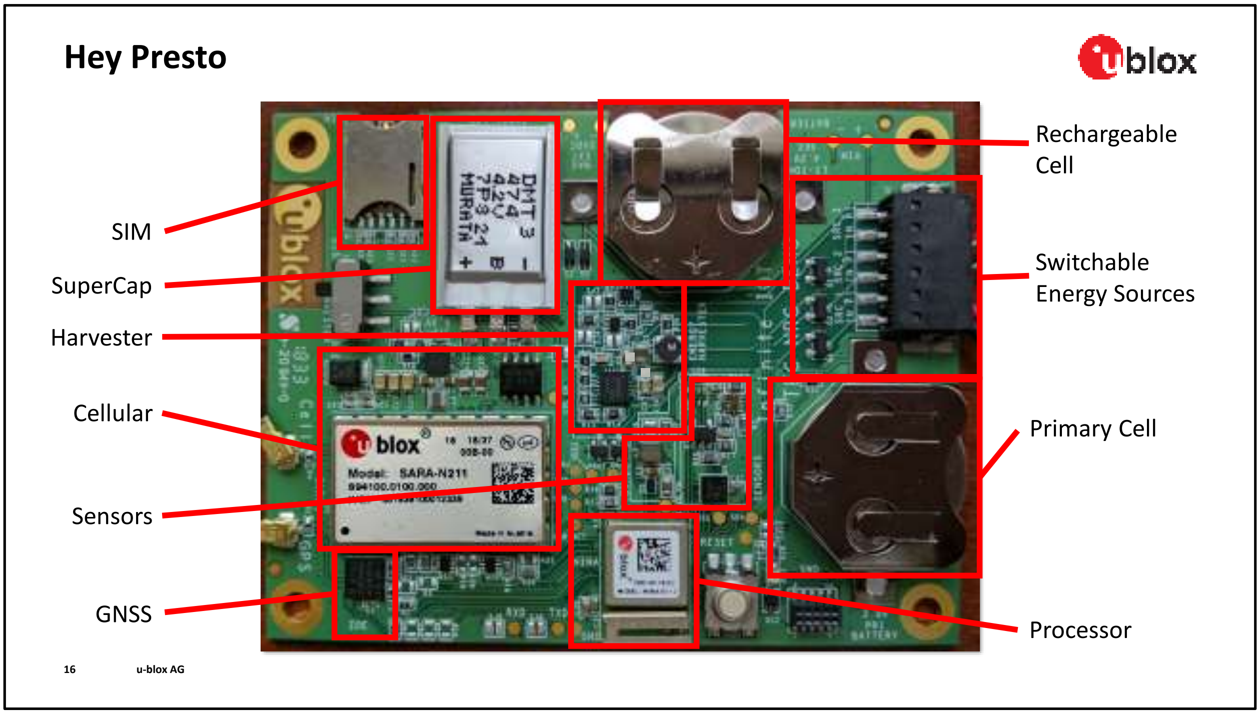


This is what we **chose**:

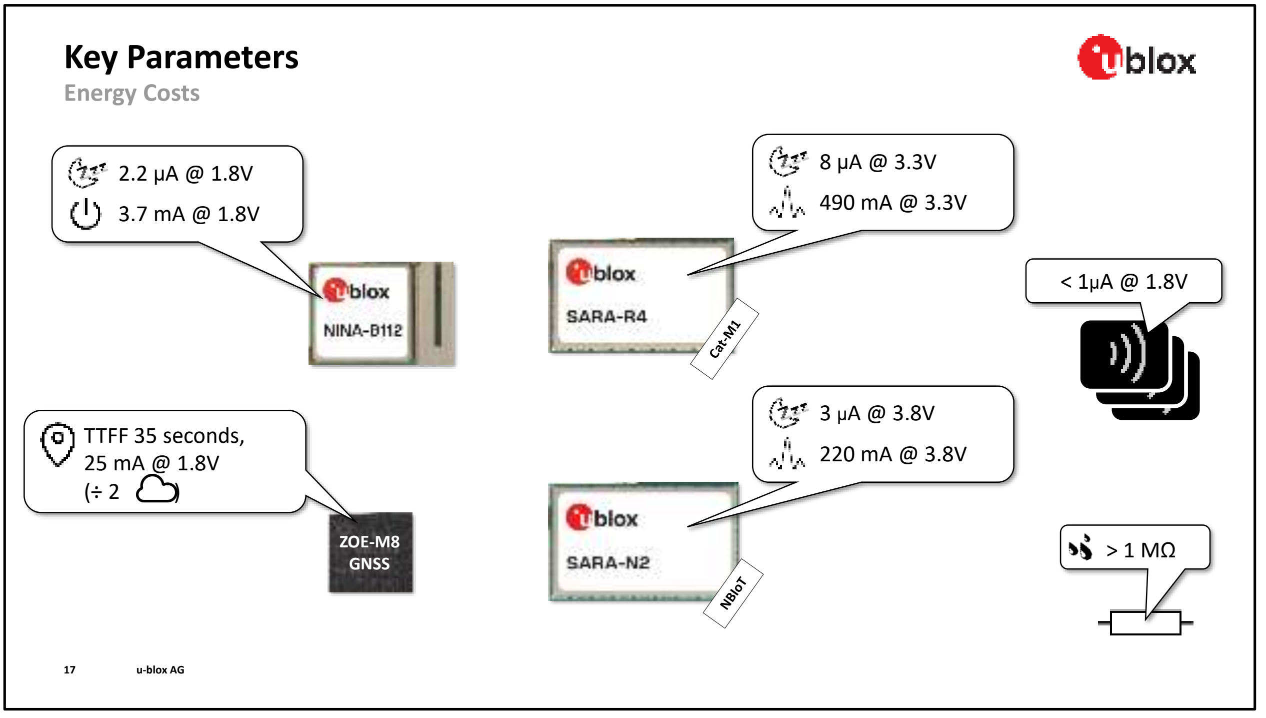
- A design **switchable** between up to **three** energy **sources**.
- A **tiny** 85 mAh **CR2032** Lithium Ion **coin cell** for **storage**.
- A 0.5F **super capacitor** for rapid current **delivery**, taking the stress off the coin cell.
- A **u-blox NINA-B1** module as the **control** processor. Note: though we didn’t use it this device also has **Bluetooth**.
- A **TI BQ25505** energy harvesting ASIC.
- We wanted this board to do **something real** so we piled on **sensors, chosen** for their **low power**:
    - **u-blox ZOE-M8** GNSS chip,
    - **Bosch BME280** environment sensor,
    - **SiLabs SI1133** light sensor,
    - **SiLabs SI7210** hall effect sensor,
    - **Linear Technology LIS3DH** accelerometer.
- We **designed** the board to accommodate the u-blox **SARA-N2 NBIoT** module or the **u-blox SARA-R4** module, which supports both Cat-M1 and NBIoT, used in Cat-M1 mode.
- A **web server** on the public internet for the application to talk to.

Hey Presto

And here it is [waves board around], slightly taller than a **credit card** in size.



Note: the **primary cell** is **required** by the **BQ25505** energy harvesting ASIC for **start-up** and can also be switched in as a **back-up** energy source.



Here are the key parameters for **energy cost**.

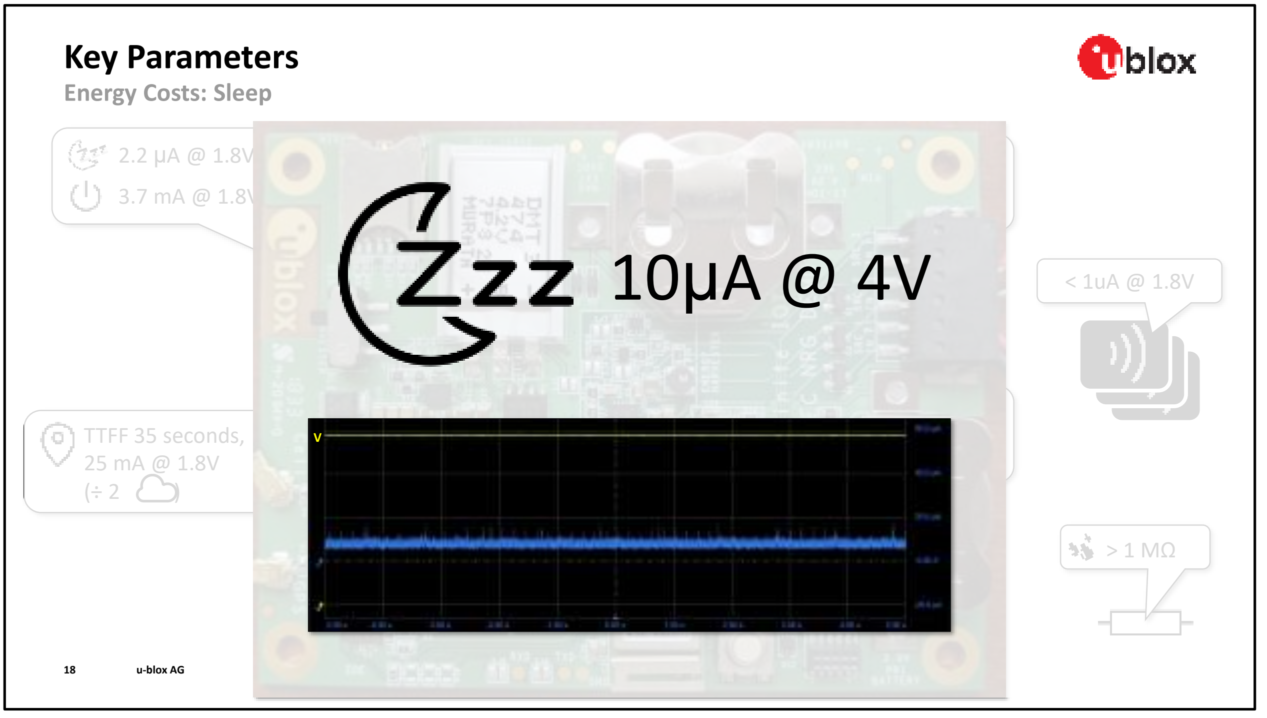
Note, in particular, the **very low sleep current** of the **NINA-B1** control processor; this is with all 64 kbytes of **RAM retained** and is **essential** for the back-bone of a sleepy system.

All of the **non-RF** based sensors hanging off the **I2C** bus, **put together**, consumed **less than 1 μA**.

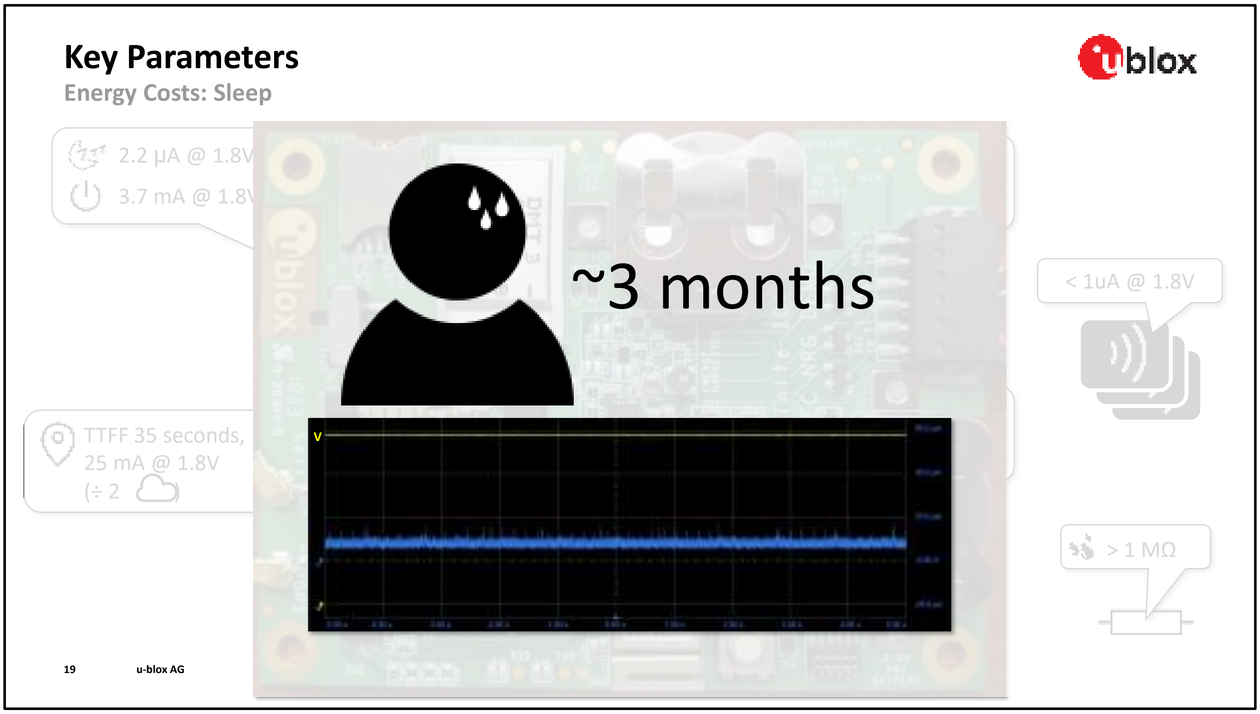
Our **pull-up resistors** etc. were kept at the **highest** possible values to **reduce leakage**.

Notes:

- The Time To First Fix for the ZOE-M8 GNSS module could have been halved through retaining and using satellite ephemeris information across wake-ups but this was not an optimisation we had time to try,
- We chose to optimise the regulator powering the cellular modem for peak current rather than sleep current since we were expecting to be powering the modem once every one or two days, and hence we never took advantage of the cellular module 3GPP sleep numbers quoted (since the regulator leakage would have dwarfed those numbers), instead turning the module entirely off between wake-ups and suffering the registration cost.



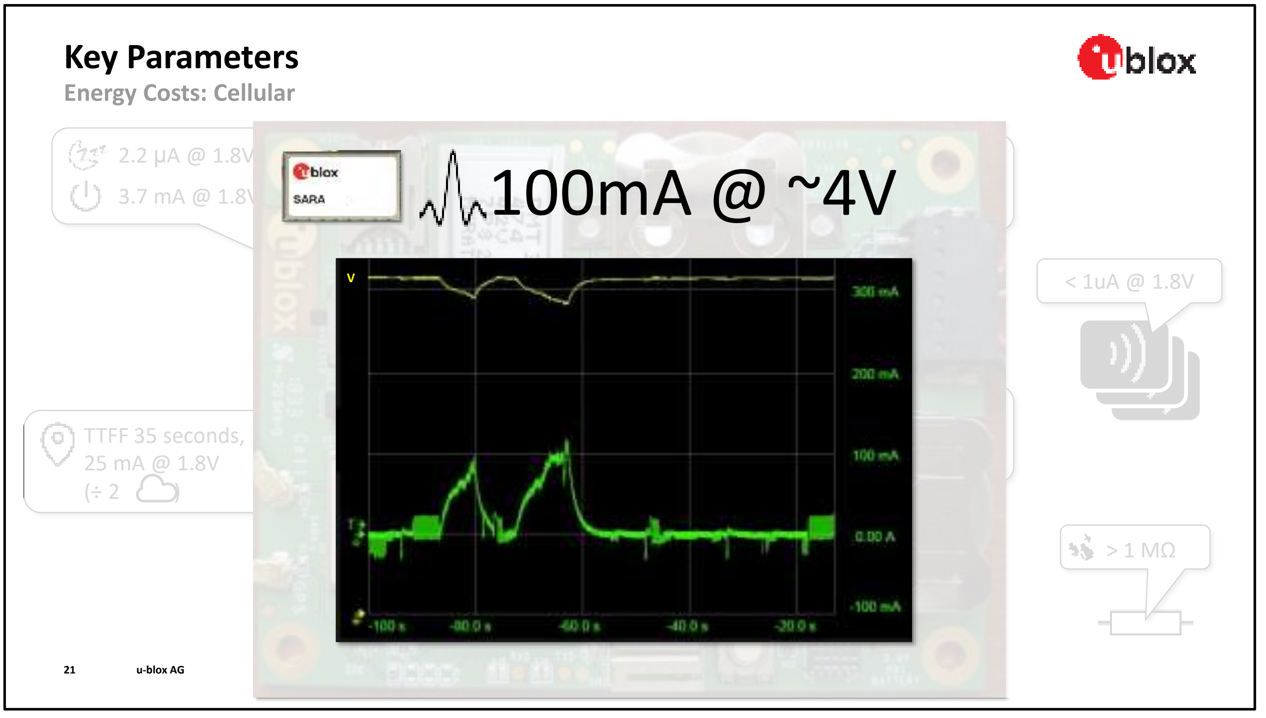
Measuring the **entire board** we managed to achieve **10 µA** sleep current with **wake-up** from **RTC**, **accelerometer** interrupt or **hall effect** sensor interrupt (so you **joggle** the board or **wave a magnet** over the board to **wake** it up).



This took by far the **bulk** of the **development time**: searching for every **lost µA**, every **parasitic** current path, finding the best **operating mode** for every sensor, setting the best **drive mode** for every GPIO pin, optimising the **parallelisation** of actions, **washing flux residue** from the surface of the board, etc.

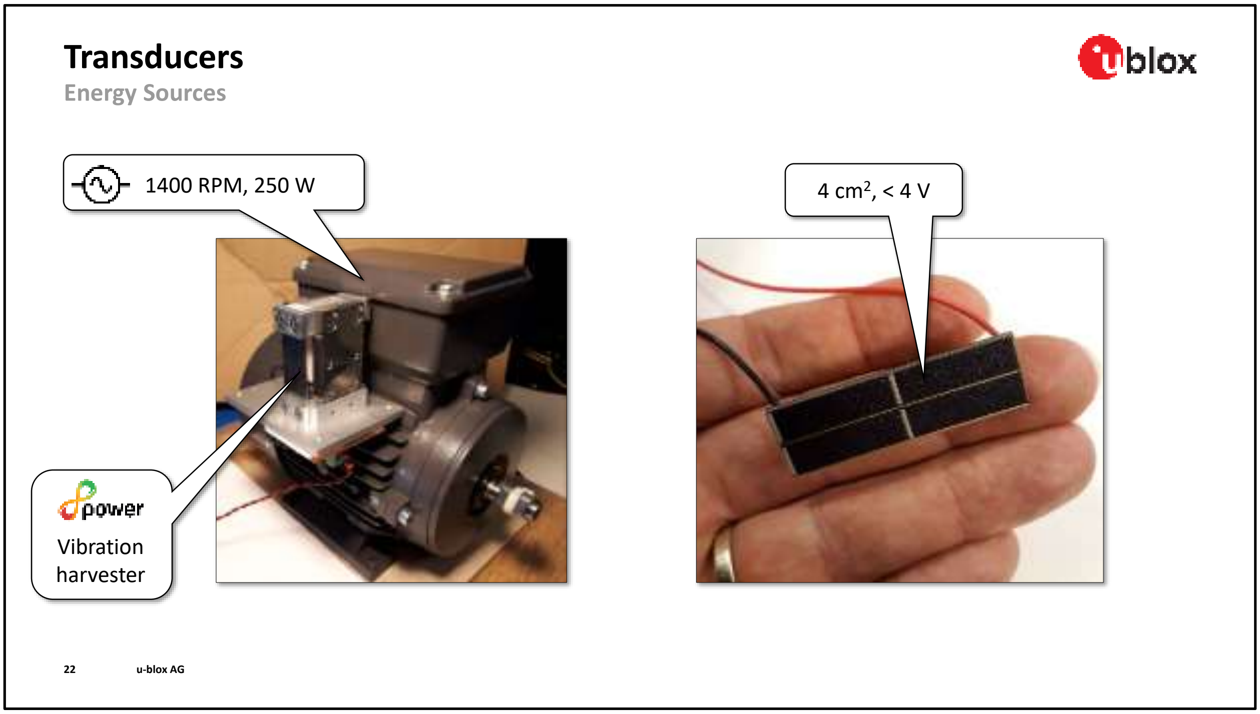
Here is the **current consumed** by everything **except** the **cellular** module (i.e. this is the 1.8V rail measured on the battery-side of the regulator) over the 80 second **"awake"** period. This is **dominated** by the current drain of the **GNSS** device obtaining a **first fix from cold**: the GNSS **antenna** was deliberately left **unplugged** to obtain a worst-case number, i.e. **unable** to obtain a **fix**.

Note the **droop** in the supply **voltage** as the super **capacitor** is being **drained**.



And here is the **current consumed** by the **SARA-R4** cellular module running in **Cat-M1 mode** over the **same period**. The **two 100 mA peaks** showing network **registration** and **data transmission** respectively.

Note: the **resolution** of this trace is such that the very **high**, very **short peaks** are **not visible**.



Over to the harvesting part.

We used **two** different energy sources:

- A **vibration harvester**, kindly provided by **8Power**, attached to a perfectly **standard** 1400 RPM 250 Watt AC **electric motor** purchased off E-Bay for **£50**.

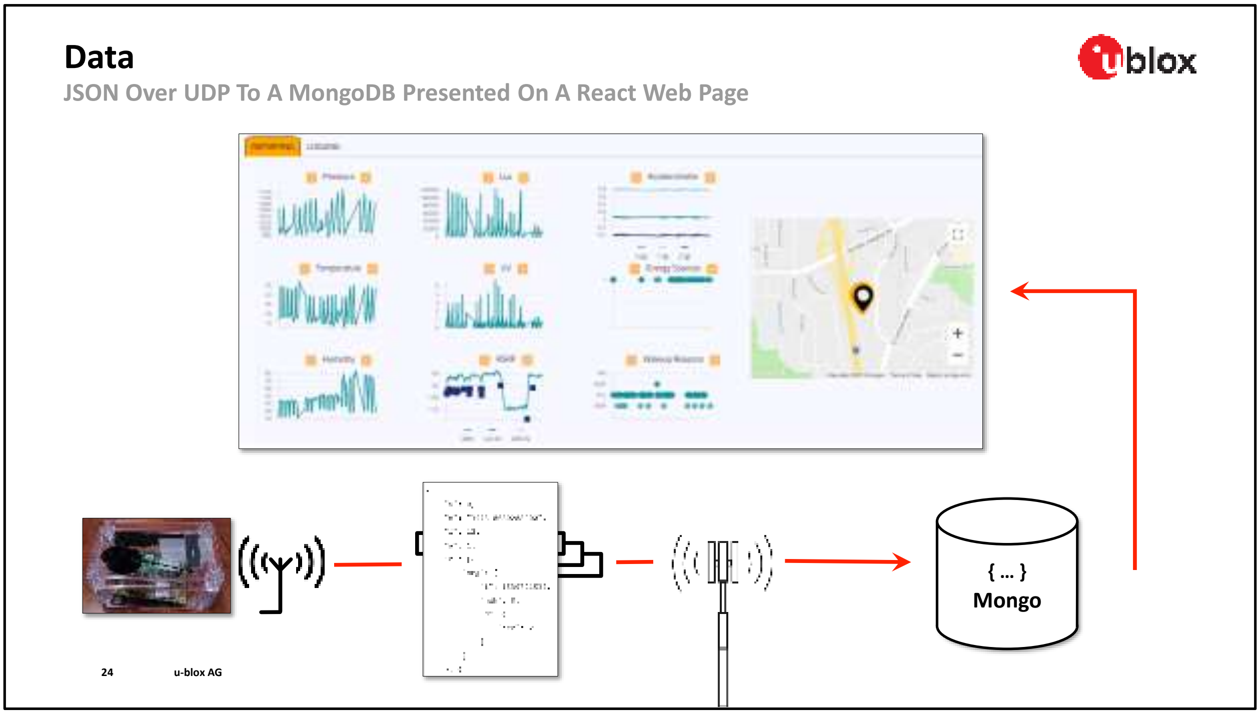
- A 4 cm^2 **solar cell** from **IXYS**, chosen as this represents a **realistic size** for an IoT device, its output voltage is no higher than **4 V**, which doesn't breach the **input range** of the TI **BQ25505** energy harvesting ASIC, and as a monocrystalline solar cell it is **very efficient** (claimed **22%** over the visible light range).

For Outdoor Use

Stuffed Into A Waterproof Box

23 u-blox AG

For **outdoor** use we **stuffed** the lot into a **clear plastic** water-tight box with a **breathable lug** to stop our precious board drowning in **condensation**.



The bit that **matters**, to **customers**, the **data**, is sent as **plain-text JSON** in **UDP** packets over **cellular** to a **server** of our own which pours the JSON directly into a **Mongo** database (which is itself natively JSON).

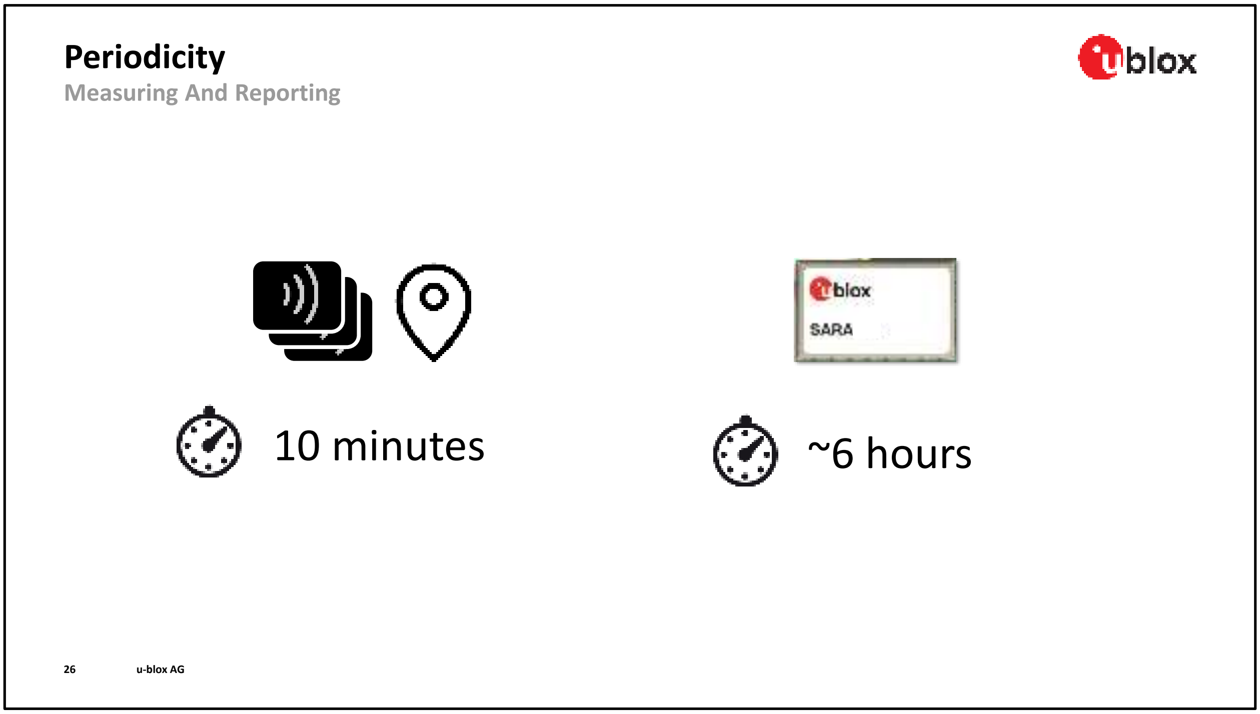
A **web page**, written in React, pulls the information from the database to display it.

Locations

Strapped To A Pole

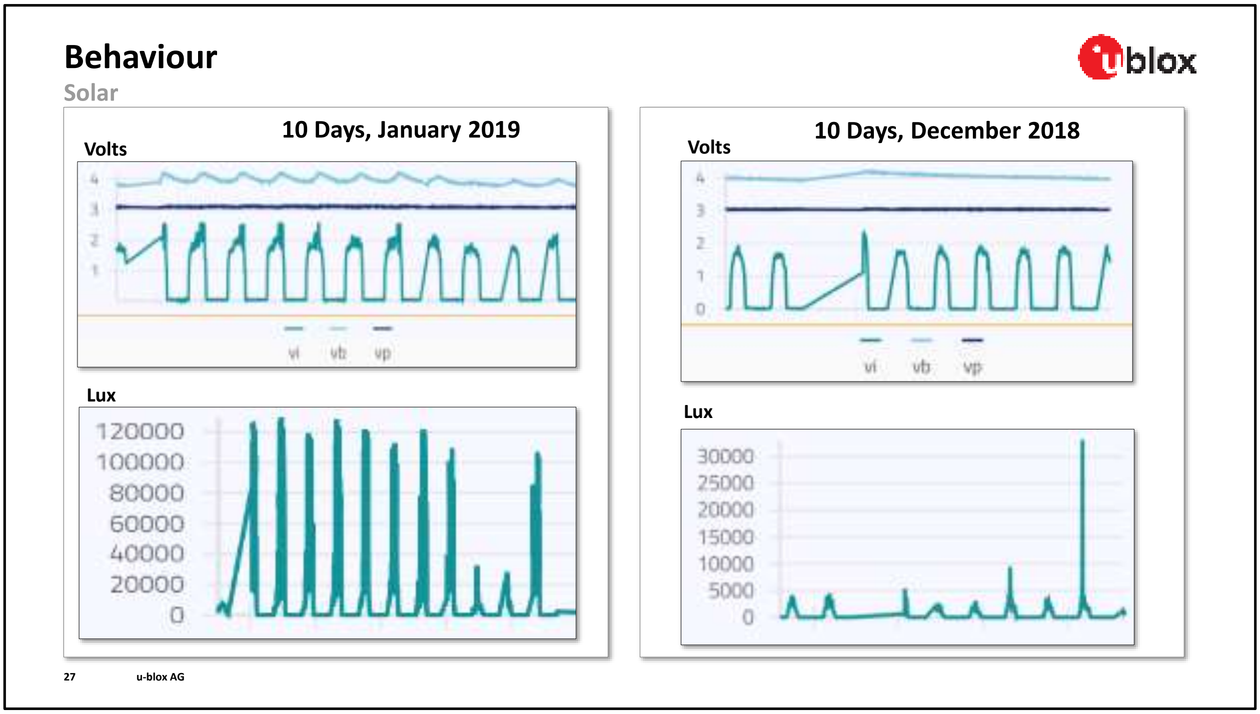
We ran these boxes **strapped to a pole** or similar:

- in the lab in **Cambridge**, powered by **vibration** energy,
- on the **NBIoT trial network** at **Vodafone** in **Newbury**,
- at our **Berlin office**,
- and in **San Diego** (where the HW engineer on the project resides).



The devices were set to wake-up and take **measurements** every **10 minutes**. Once **RAM** has filled up with measurements, which is about **every 6 hours**, the **cellular** module is powered-up to send a **report** containing all of these measurements to the **server**.

The server can then display **contiguous** measurements at 10 minute intervals, update every 6 hours.

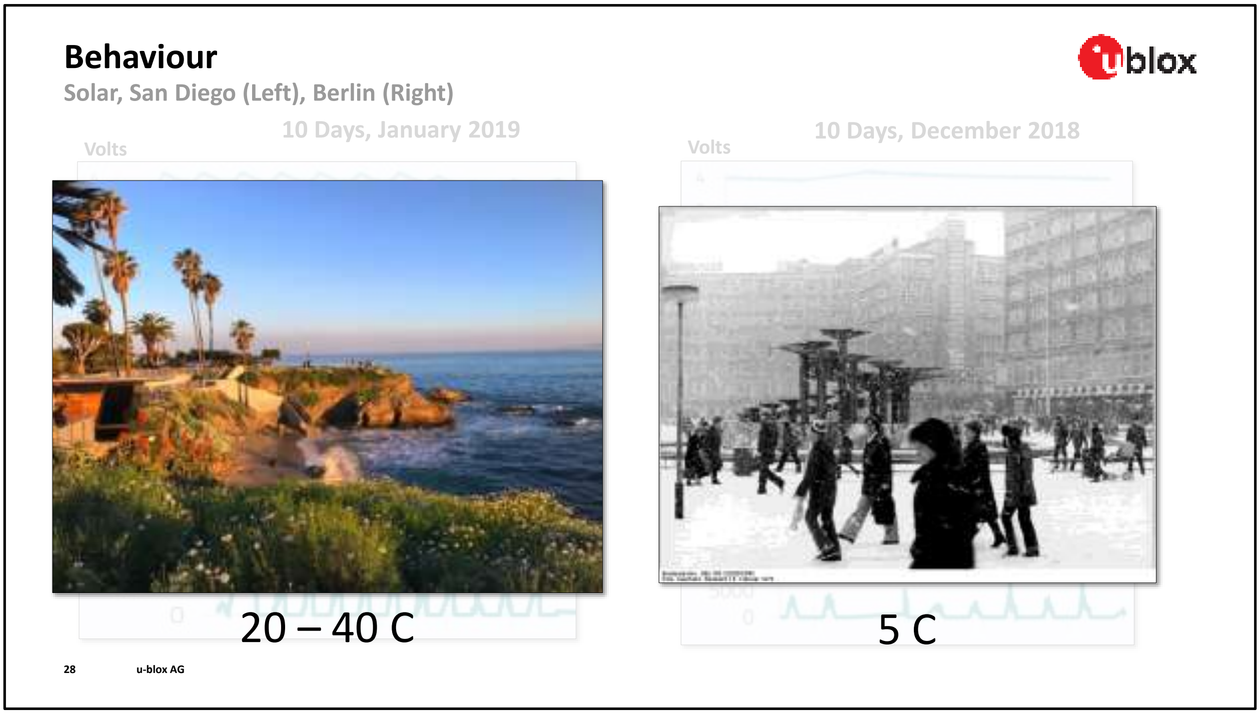


Here are snapshots of the behaviour of **two devices** powered by **solar** energy over a **10 day** period last **Christmas**; one device on the **left**, **another** device in a **different location** on the **right**.

In the **graph** on the **top left** you can see a regular **daily jump** in **VIn**, the voltage **across** the **solar** cell, (the **greenish line**), which **matches** the **light** measurements sent by the device in the graph below. Most **critically**, in the **same top left graph** you can see **VBatt** (the top line), the voltage across the CR2032 **button cell**, **rise** during the day as it is **charged** and the device **continues** to report **without a break** throughout the **night**.

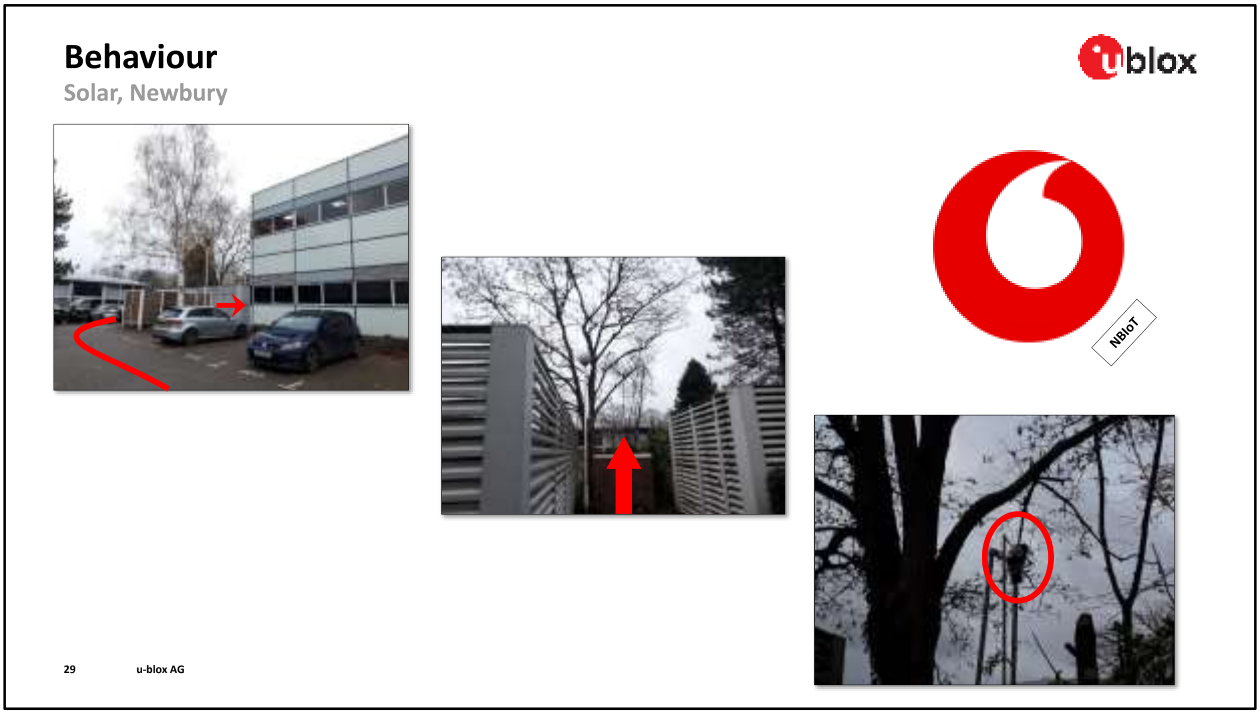
The graphs on the **right** are from a **different location**. The long gap you can see is where the CR2032 **button cell** was manually **recharged** from a bench power supply. And look at the **light levels**: aside from one peak, which I cannot explain, **daylight** was measured at **5000 lux**, compared with **120,000 lux** on the left. And **VIn** shows no sign of a daily rise, just a **slow decay** after being manually raised.

Can you **guess** which **locations** these are?



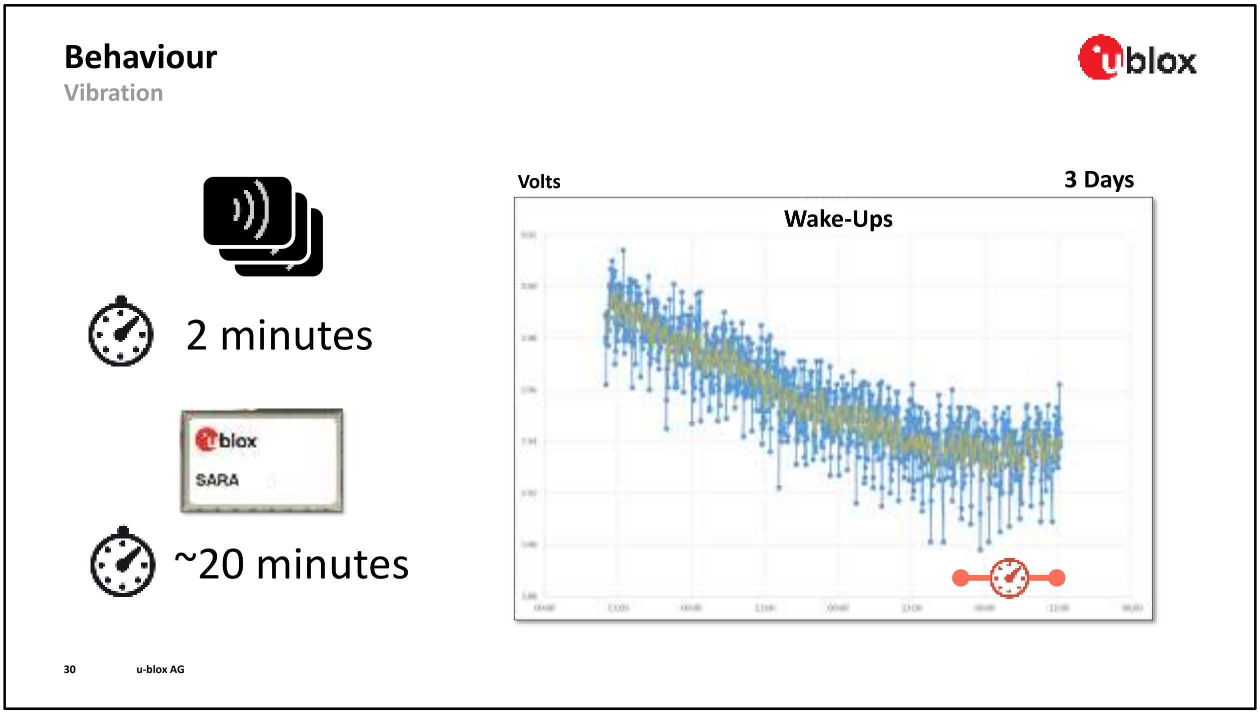
With the **pictures** I'm **joking** of course but the **temperature** numbers were the **real** numbers reported by the device **inside** the **transparent** boxes in **San Diego** on the left and **Berlin** on the right.

In fact, the **behaviour** of the **non-industrial**-temperature-range **CR2032 button cells** may well have been **enhanced** by the **San Diego** heat.



And **Newbury**? Well, the only location we could find that was in range of the **Vodafone NBIoT** test network (which is radiating at very low power **inside the building** on the right of the top-most picture) was on the **north side** of the building, between the **bins** on a pole underneath a **tree**.

It **barely** lasted a **week**.



The device in our **Cambridge lab** was set to **wake-up** and measure every **2 minutes** and to **report** those measurements roughly every **20 minutes**.

The **graph** shows the wake-up events over a **three day** period. The part on the **right** where the graph **flattens out** is where the **software** in the device has determined what it can **afford** to do, **missing** out some wake-up opportunities to **balance** the energy **expenditure** with the energy **available**.

Basically it amounts to roughly **2 mW** of **power** available **continuously**.

Cost

Versus A Primary Cell (Using Digikey Single Unit Prices)

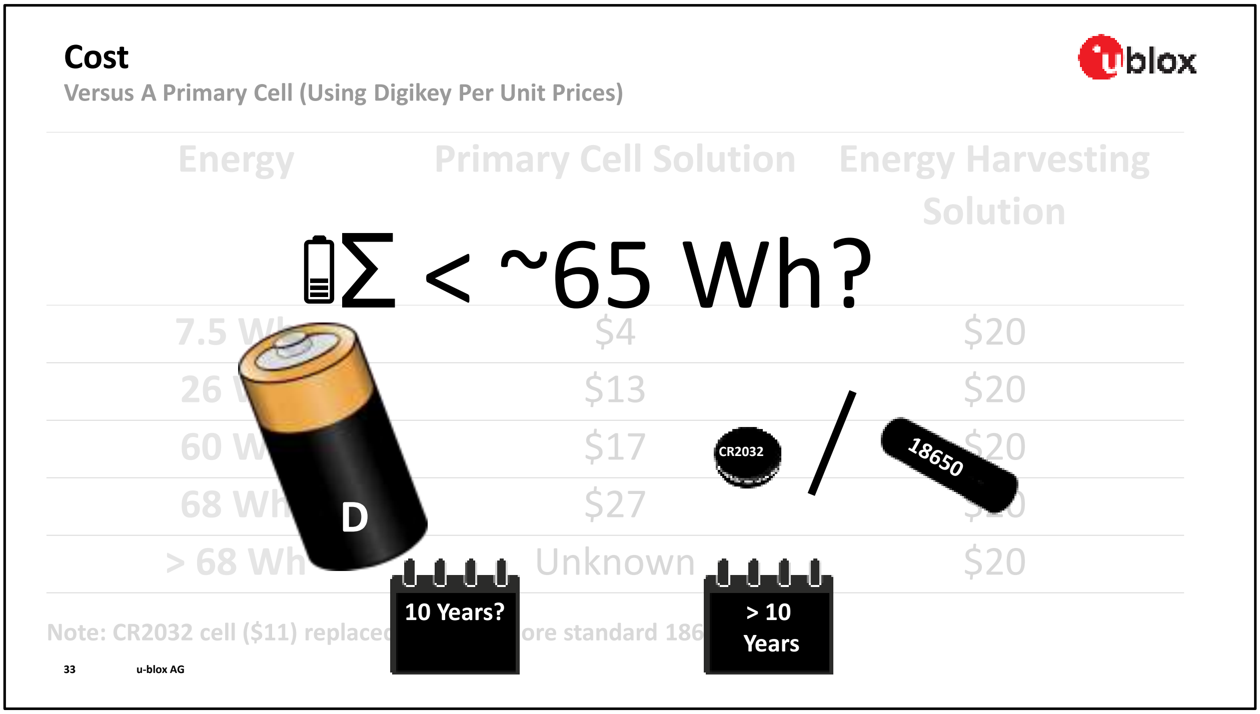
Energy	Primary Cell Solution	Energy Harvesting Solution
7.5 Wh	$4	$20
26 Wh	$13	$20
60 Wh	$17	$20
68 Wh	$27	$20
> 68 Wh	Unknown	$20

Note: CR2032 cell ($11) replaced with the more standard 18650 cell ($6).

There's the question of **cost** versus a **primary cell**: if you're **not** going to **lose** the **battery** then why not just put in a **nice big one**?

These **numbers** are based on **our design** costed at **DigiKey per-unit** prices, so ***way*** more expensive than any **real** product would be but useful for **comparative** purposes.

Losing the **harvesting components** and replacing them with a **primary cell**, on these numbers, suggests a **break-even** point at around **65 Wh**.



So, in **our example**, if your **lifetime** system **energy** cost is less than **65 Wh** just putting in a bigger battery would offer the **cheapest** solution, if **cost** is your **motivation**. 65 Wh would **probably** bring you **10 year battery life**.

I **emphasize** that this is true for ***our experiment only***, it is NOT a general rule, you need to **work out** the numbers for your product.

Note: the tiny CR2032 cells we chose were particularly expensive (three times the cost of any other component on the board) so these numbers assume use of the much more standard 18650 Lithium Ion rechargeable cell.

Summary

What We Did

Measure every few minutes, report collected data every few hours over cellular.	✓
Do this using NBIoT and Cat-M1.	✓
Do this using standard u-blox components.	✓
Do this using solar and vibration energy sources.	~
Do this at an acceptable product cost.	?

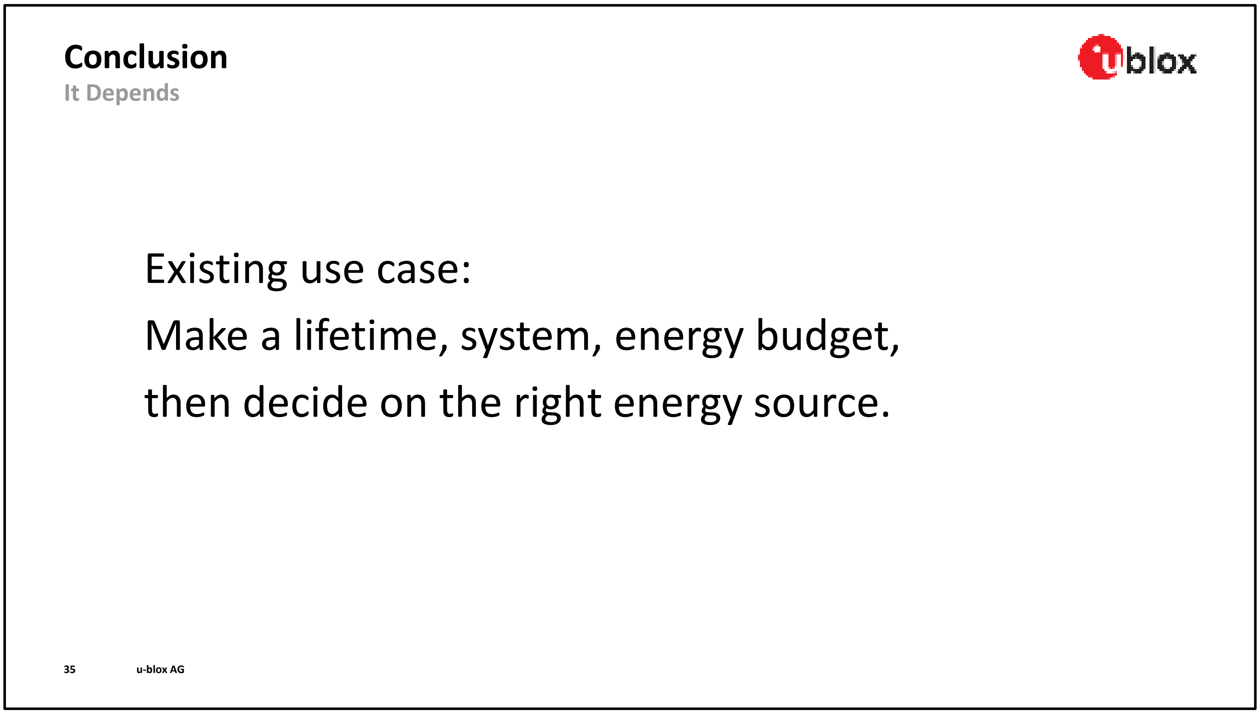
34 u-blox AG

In **summary**, it was **possible** to measure and report data at an "every **few minutes**" **resolution** over **cellular NBIoT/Cat-M1** using **harvested** energy with **standard u-blox** components.

Solar was good when it was **good**, bad when it was **bad**.

Vibration was reliable/**continuous** but in the **middle**.

We worked out the **product cost** relative to a **primary cell** solution but I make **no judgement** as to **acceptability**, that's down to the **use case**.



Conclusion:

For an **existing** use case, make a **lifetime**, **system-wide energy budget** and then you can decide on the **right** energy **source**.

Conclusion
Think Out Of The Box

Think laterally:

What new use-cases could I address with a truly "fit and forget" product?

But that's **boring**, it is much more **interesting** to think what ***new*** use cases you could meet if you could truly make a "**fit and forget**" product:

- built into **infrastructure** during construction,
- in **dangerous/inhospitable/unreachable** locations,
- inside the **human body**,
- etc.

Conclusion

It Doesn't Depend At All

90% of the work applies just as well to a primary cell design: in connected IoT every µWh counts.

37 u-blox AG

When it comes down to it, **90%** of this applies just as well to a **primary cell** design: **connected IoT** should always be design with **energy** on the **front** page of the **product spec** rather than in the **addendum**.

Thank you
for your attention

White Paper:

https://www.u-blox.com/white-papers?utm_source=Autopilot&utm_medium=email&utm_campaign=energy-harvesting

This talk is based on the contents of the white paper linked above.