AN12768

Using a stability algorithm in a cold-chain use case

Rev. 1.1 — 29 April 2021 **Application note**

Document information

Information	Content
Keywords	NHS31xx, coldchain, RTC, validation, stability
Abstract	How to build up a stability algorithm

Revision history

Rev	Date	Description
v1.1	20210429	Update for SDK 12.4.1
v1.0	20200312	Update for SDK 12.4
Modifications:	• Major format update and refresh of contents	
v0.2	20170113	Update for SDK 9
v0.1	20161215	Initial version

1 Introduction

All ICs of the NHS31xx family are ideally suited for cold-chain logging and monitoring use cases. The HW provides the building blocks for complex and standalone operations: an internal RTC driven by a free-running oscillator, an accurate temperature sensor, and plenty of volatile and non-volatile memory. The NHS31xx ICs also include an open Arm Cortex M0+ processor where the firmware can be fully tailored for each customer and use case.

To decide after each measurement if the attached drug or perishable good is still good for consumption, the firmware must typically implement logic. This validation is captured in a so-called "stability algorithm".

When creating a use-case specific stability algorithm, the notes below speed up the development and help in deciding on some algorithm parameters.

1.1 Typical cold-chain use case

This document assumes the following generic situation:

- A tag reader is used to communicate with a label containing an NHS31xx IC. The tag reader can be a mobile phone, a standalone NFC reader, or a USB NFC reader connected to a PC.
- The tag reader taps the label and configures the label by sending application-specific commands over the NFC interface. Among others, it instructs the label to start monitoring and logging by making a measurement every `x` minutes.
- From that moment on, the NHS31xx IC wakes up every `x` minutes to check the current situation.
  - The current situation is assessed by making a temperature measurement and coupling this value with the current IC time.
  - At least, excursions and anomalies are recorded. These reports can be used for corrective actions to the cold chain later.
  - Between two measurements and corresponding calculations, the IC puts itself in a low-power mode (NHS31xx's deep power-down mode).
- To assess the situation after each measurement, a stability algorithm is used.
  - When fed with a new measurement, it executes all required checks. Its output indicates the validity of the attached product.
  - A check consists of a series of calculations, using the current time, the current temperature, and the past temperature values.

1.2 Example

As an example, an algorithm could be designed to validate a product that must be kept in a fridge. Outside a fridge, the product slowly deteriorates. We can then imagine two thresholds have been determined:

- The temperature cannot exceed 20 °C for more than 60 minutes
- The temperature cannot exceed 8 °C for more than 10 hours

For simplification, we ignore freezing temperatures in this example.

Combinations of the two thresholds mentioned above must also be checked. For example, the following situations also render the product invalid:

- After having exceeded 20 °C for half an hour and having been between 8 ºC and 20 °C for 5 hours
- After having exceeded 20 °C for 15 minutes and having been between 8 °C and 20 °C for 7.5 hours
- …

When the temperature exceeds a threshold, a so-called "excursion" occurs. The length of an excursion can vary widely. One long excursion can render a product invalid or the same result is only reached after multiple smaller disjoint excursions. The stability algorithm must add up all these excursions correctly.

2 Pitfalls

Each time the firmware requests a new temperature measurement, the temperature sensor block is only powered for a short time. The exact length is only a little more than the required conversion time for a temperature measurement. The conversion time depends on the configured resolution of the temperature measurement, with 100 ms required for the highest resolution. When a new measurement is obtained, the temperature sensor HW block is powered off again.

Between two measurements, the temperature sensor block remains powered off. So, temperature fluctuations can pass unnoticed. The temperature sensor block is different from a chemical sensor which is continuously observing the temperature.

The configured measurement interval, which is use-case specific, defines the granularity of the measurements. When set too high, fast fluctuations are not detected. When set too low, unnecessary current is consumed.

2.1 Excursion length uncertainty

An excursion is a period of time where the temperature value continuously exceeds a use-case specific upper threshold or is continuously below a specific lower threshold.

When measuring every `x` minutes, an excursion translates to a number of temperature measurements where the excursion is detected. The stability algorithm can count the number of temperature values that exceeds the threshold value and use that for its decision making.

One long excursion is easily detected. Take, for example, Figure 1 below, with a measurement interval of 15 minutes:

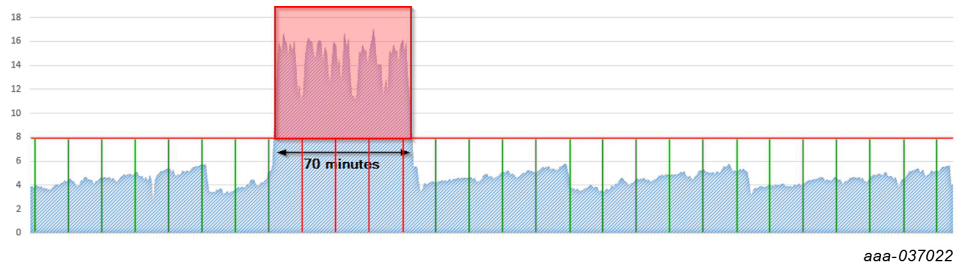

Figure 1. One long excursion with marked measurement times

In this case, four measurements give temperature readings that are too high. The stability algorithm can estimate that the temperature was too high for `4·15 = 60` minutes. In reality, the length of the excursion can be anywhere within `]60+15, 60-15[`.

Note: *For a measurement interval of x minutes, the uncertainty of any excursion length is 2·x minutes.*

2.2 Combining multiple excursions

Excursions can come in different forms:

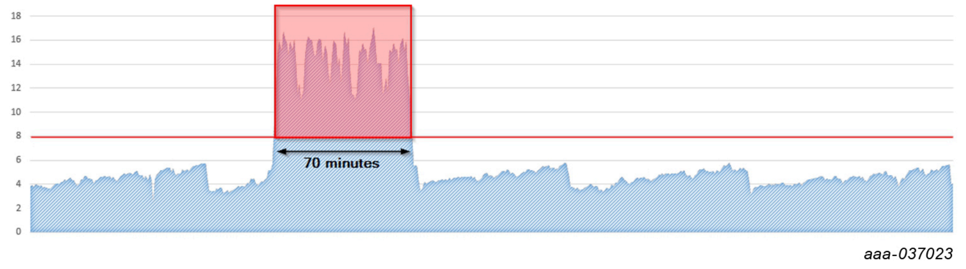

Figure 2. Temperature curve with one long excursion

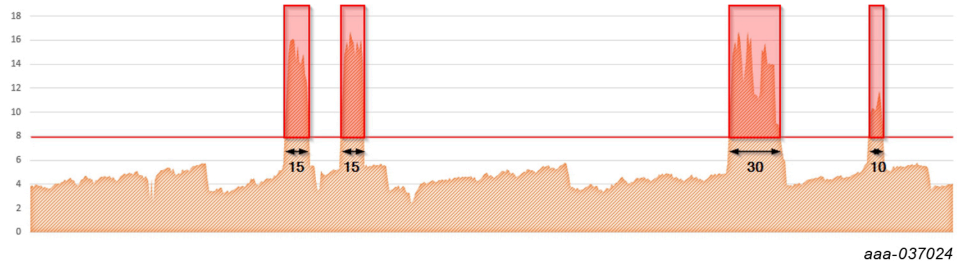

Figure 3. Temperature curve with multiple shorter disjoint excursions

In the latter graph, the uncertainty on the exact duration of the total excursion time is 4 times higher.

2.3 Missed and misjudged excursions

A more capricious temperature curve can cause multiple misjudgments:

Figure 4. Less predictable temperature curve

- Several smaller excursions are not captured, yet they may or may not add up to the deterioration of the attached good.
- The stability algorithm considers two shorter excursions (near the end of the graph) as one longer excursion.

These errors are inherent to the digital nature of the usage of the NHS31xx temperature measurement block. To ensure that all fluctuations are captured, a shorter measurement interval is required. An interval that is too short yields unnecessary data and may tax the battery too much. To attain an estimate of the required battery capacity, given the different parameters that can be tweaked, check the CurrentConsumption.xlsx tool in the SDK. The thermal inertia of the product determines the optimal measurement interval.

2.4 RTC deviation

The RTC relies on an 32 kHz oscllator. The RTC deviation can, uncorrected, be as high as 2.5%. This impacts the exact measurement interval. Suppose a deviation of precisely 2.5%:

- For an interval of 10 minutes, the error is 15 seconds in either direction. Instead of the expected 600 seconds, the real value is somewhere between 585 seconds and 615 seconds.
  For an interval of 15 minutes, the error is 22.5 seconds in either direction. Instead of the expected 900 seconds, the real value is somewhere between 877 seconds and 923 seconds.

Aging has no impact on the RTC deviation. The past lifetime and the past excursions of the RTC have no influence on the current RTC deviation.

For decision making in a stability algorithm, the RTC deviation is low compared to the measurement interval. The uncertainty of the excursion length due to the measurement interval is factors higher than the uncertainty introduced by the RTC deviation.

3 Mitigations

Knowledge of the specific use case and the end-product, makes it possible to avoid the pitfalls.

3.1 Reducing false positives

The stability algorithm can make two types of errors.

- A **false positive** occurs when missing a product which has expired.
  The stability algorithm decides a product is still good for consumption, but in reality it has expired or has suffered reduced effectiveness.
  If the RTC runs too slow, this number increases. When the RTC runs too slow, the RTC clock reports `x` minutes have passed when in reality more than `x` minutes have passed. The stability algorithm then passes a product which has already expired more easily.
- A **false negative** when flagging a product which is still OK.
  The stability algorithm decides a product is not good for consumption anymore, while in reality it was still fully effective (or effective enough).
  If the RTC runs too fast, this number increases. When the RTC runs too fast, the RTC clock reports `x` minutes have passed when in reality less than `x` minutes have passed. The stability algorithm then concludes that a product has expired more easily.

If the error due to the RTC deviation is considered too high for some strict criteria checked in the stability algorithm, the configured measurement interval can be adjusted to eliminate false positives.

Picking up our example from the previous section, for an interval of `x` minutes, the wake-up interval can be configured to expire after `x to 2.5 %` minutes. The stability algorithm continues to use an interval of `x` minutes for its calculations. This configuration is enough to eliminate false negatives due to the RTC error fully. As a result, the stability algorithm is a bit more restrictive. It flags and rejects faster, which means the number of false negatives is increased.

3.2 Reducing RTC deviation

When time stamps can be repositioned, the RTC deviation can be corrected during a readout. For more information, see the application note AN12769 (Ref. 1).

3.3 Shortening the measurement interval

A straightforward and effective way to reduce the effects of the sampling uncertainty is to reduce the measurement time. However, reducing the measurement time has two downsides:

- Higher power consumption:
  If the interval is chosen to be a few minutes or more, the dominant factor is the current consumption between two measurements. To check the impact of the estimated required battery capacity when playing with the measurement interval, use our tool, *CurrentConsumption.xlsx*, which is part of the SDK.
- More logging space required:
  Depending on the active lifetime of the IC, the logging space may or may not be a concern. To reduce the storage space, lossless and lossy compression can be employed. For example, an algorithm may measure every 5 minutes, but only store one

out of three measurements. For more information, see *DataStorage.pdf*, which is part of the SDK.

3.3.1 Adaptive periods

Another possibility that reduces the uncertainty while avoiding the downsides of the above suggestion is to use adaptive periods.

The next wake-up period can be adjusted based on the last temperature measurement:

- When the current temperature is close to a threshold value or when a decision point is almost reached, use a shorter period.
- When the current temperature is in the middle of the optimal temperature range, use a longer period.

The use of adaptive periods increases the complexity of the logic of the stability algorithm. Also, a significant increase of the required storage space must be avoided. "Replaying" the logic employed to define the next interval between measurements can avoid the necessity to store a costly timestamp for each measurement value.

The amount of adjustment in the interval is use case-specific. It must be tweaked in alignment with the expected thermal behavior. At the cost of complexity, it provides a compromise between accuracy, power consumption, and data logging storage.

4 References

[1] **AN12769 application note** — NHS31xx RTC - On the RTC deviation; 2021, NXP Semiconductors

[2] **DataStorage.pdf slides** — NHS31xx data storage; 2020, NXP Semiconductors

5 Legal information

5.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

5.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

5.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

Contents

Date of release: 29 April 2021
Document identifier: AN12768