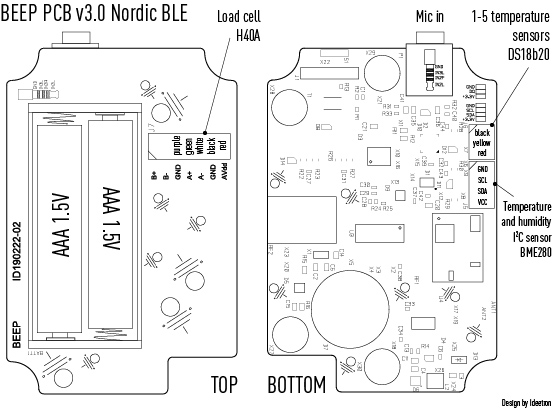
**Beep Base - PCB ID190222-02**

**Firmware handleiding**



Datum: 18-12-2019

Versie: 1.3

Titel: Beep Base - ID190222-02 - Firmware - Handleiding

# Document revisie en verdeling

Document revisie:

|  |  |  |  |
| --- | --- | --- | --- |
| **Versie** | **Datum** | **Door** | **Wijzigingen** |
| 1.0 | 31-10-2019 | Adri | 1e uitgave: Inleiding, Hardware, Software toegevoegd. |
| 1.1 | 4-11-10 | Adri | HX711 opdracht aangepast, HX711 meet configuratie toegevoegd, buzzer opdrachten toegevoegd, Programmeren hoofdstuk toegevoegd. |
| 1.2 | 29-11-2019 | Adri | Pincode lees/schrijf opdrachten gewijzigd, Flash opdrachten toegevoegd; read, erase, size, TX log karakteristiek toegevoegd, flash log inhoud. Pin code reset toegevoegd. |
| 1.3 | 12-12-2019 | Pim | Puur cosmetisch en functioneel: Google Drive doc ervan gemaakt om samenwerking te vergemakkelijken. Afbeelding op voorpagina, titel heeft geen versienummer meer. |
| 1.3 | 18-12-2019 | Adri | Versie nummer niet verhoogd, zodat dit gelijk is aan de firmware versie. Toegevoegd:   * BME280 BEEP protocol berichten * Alarm instellingen protocol berichten * FFT beschrijving toegevoegd |
| 1.3 | 20-12-2019 | Adri | Beschrijving flash erase aangepast met extra erase optie |

# Inhoudsopgave

[**Documentrevisie en verdeling**](#_heading=h.gjdgxs) **2**

[**Inhoudsopgave**](#_heading=h.1fob9te) **3**

[**Inleiding**](#_heading=h.1d9fx1m4n95c) **5**

[**Hardware**](#_heading=h.2et92p0) **5**

[**nRF52840**](#_heading=h.tyjcwt) **5**

[**AA batterijen**](#_heading=h.3dy6vkm) **5**

[**DS18B20 temperatuur sensor**](#_heading=h.1t3h5sf) **5**

[**HX711 rekstrook sensor**](#_heading=h.4d34og8) **6**

[**Reedswitch**](#_heading=h.2s8eyo1) **6**

[**TPS boostconverter en supply switch**](#_heading=h.17dp8vu) **6**

[**Buzzer**](#_heading=h.3rdcrjn) **6**

[**Flash storage**](#_heading=h.26in1rg) **6**

[**RFM95**](#_heading=h.lnxbz9) **6**

[**ATECC608A**](#_heading=h.35nkun2) **6**

[**BME280**](#_heading=h.1ksv4uv) **6**

[**TLV320ADC3100**](#_heading=h.44sinio) **6**

[**Logging**](#_heading=h.2jxsxqh) **6**

[**Software**](#_heading=h.z337ya) **7**

[**Beep Protocol**](#_heading=h.3j2qqm3) **7**

[0d/0x00 – RESPONSE](#_heading=h.1y810tw) 8

[1d/0x01 - READ\_FIRMWARE\_VERSION](#_heading=h.4i7ojhp) 9

[2d/0x02 - READ\_HARDWARE\_VERSION](#_heading=h.2xcytpi) 9

[3d/0x03 - READ\_DS18B20\_STATE](#_heading=h.1ci93xb) 10

[131d/0x83 - WRITE\_DS18B20\_STATE](#_heading=h.3whwml4) 10

[4d/0x04 - READ\_DS18B20\_CONVERSION](#_heading=h.2bn6wsx) 11

[132d/0x84 - WRITE\_DS18B20\_CONVERSION](#_heading=h.qsh70q) 11

[9d/0x09 - READ\_HX711\_STATE](#_heading=h.3as4poj) 12

[137d/0x89 - WRITE\_HX711\_STATE](#_heading=h.1pxezwc) 13

[10d/0x0A - READ\_HX711\_CONVERSION](#_heading=h.49x2ik5) 14

[138d/0x8A - WRITE\_HX711\_CONVERSION](#_heading=h.2p2csry) 14

[20d/0x14 - READ\_LORAWAN\_STATE](#_heading=h.147n2zr) 16

[145d/0x91 - WRITE\_BUZZER\_DEFAULT\_TUNE](#_heading=h.3o7alnk) 16

[146d/0x92 - WRITE\_BUZZER\_CUSTOM\_TUNE](#_heading=h.23ckvvd) 17

[148d/0x94 - WRITE\_LORAWAN\_STATE](#_heading=h.ihv636) 17

[21d/0x15 - READ\_LORAWAN\_DEVEUI](#_heading=h.32hioqz) 18

[149d/0x95 - WRITE\_LORAWAN\_DEVEUI](#_heading=h.1hmsyys) 18

[22d/0x16 - READ\_LORAWAN\_APPEUI](#_heading=h.41mghml) 18

[150d/0x96 - WRITE\_LORAWAN\_APPEUI](#_heading=h.2grqrue) 19

[23d/0x17 - READ\_LORAWAN\_APPKEY](#_heading=h.vx1227) 19

[151d/0x97 - WRITE\_LORAWAN\_APPKEY](#_heading=h.3fwokq0) 19

[136d/0x98 - WRITE\_LORAWAN\_TRANSMIT](#_heading=h.1v1yuxt) 19

[27d/0x1B - READ\_nRF\_ADC\_CONVERSION](#_heading=h.4f1mdlm) 20

[155d/0x9B - WRITE\_nRF\_ADC\_CONVERSION](#_heading=h.2u6wntf) 20

[29d/0x1D - READ\_APPLICATION\_CONFIG](#_heading=h.19c6y18) 21

[157d/0x9D - WRITE\_APPLICATION\_CONFIG](#_heading=h.3tbugp1) 21

[30d/0x1E - READ\_PINCODE](#_heading=h.28h4qwu) 22

[158d/0x9E - WRITE\_PINCODE](#_heading=h.nmf14n) 22

[31d/0x1F - READ\_BOOT\_COUNT](#_heading=h.37m2jsg) 23

[32d/0x20 - READ\_MX\_FLASH](#_heading=h.1mrcu09) 23

[33d/0x21 - ERASE\_MX\_FLASH](#_heading=h.46r0co2) 24

[34d/0x22 - SIZE\_MX\_FLASH](#_heading=h.2lwamvv) 25

[**Bluetooth Low Energy**](#_heading=h.111kx3o) **25**

[Pin code](#_heading=h.3l18frh) 25

[Device information service](#_heading=h.206ipza) 25

[Battery service](#_heading=h.4k668n3) 26

[DFU](#_heading=h.2zbgiuw) 26

[Beep service](#_heading=h.1egqt2p) 26

[DS18B20 temperatuur resultaat karakteristiek](#_heading=h.ehs7sqsrezff) 27

[TX log data](#_heading=h.uk6solfpwlrm) 27

[Beep Control point](#_heading=h.3ygebqi) 28

[**LoRaWAN**](#_heading=h.2dlolyb) **28**

[Standaard bericht types](#_heading=h.sqyw64) 28

[**Applicatie**](#_heading=h.3cqmetx) **29**

[**Buzzer geluiden**](#_heading=h.1rvwp1q) **29**

[**Flash log**](#_heading=h.4bvk7pj) **29**

[Bericht opbouw](#_heading=h.2r0uhxc) 29

[Opstart-bericht](#_heading=h.1664s55) 30

[Meet gegevens bericht](#_heading=h.3q5sasy) 30

[**Programmeren**](#_heading=h.25b2l0r) **31**

[**BEEPBASE**](#_heading=h.kgcv8k) **31**

[**Programeer script.**](#_heading=h.34g0dwd) **31**

[**nRFutil**](#_heading=h.1jlao46) **32**

[**Segger Embedded Studio**](#_heading=h.43ky6rz) **32**

[**Applicatie debuggen**](#_heading=h.2iq8gzs) **32**

[**Compilatie scripts**](#_heading=h.xvir7l) **33**

[**Elektrisch**](#_heading=h.3hv69ve) **33**

# 

# Inleiding

Het Beep meetsysteem (Beep Base) is een systeem voor het monitoren van een bijenkast door middel van het gewicht, temperatuur en geluid. Al deze gemeten data wordt gelogd door de Beep Base.

LoRaWAN wordt gebruikt om regelmatig dat te versturen richting het beep back-end, alarmering en om op afstand instellingen te wijzigingen.

Met bluetooth low energy kan de gelogde data worden uitgelezen door de Beep App. De Beep App wordt ook gebruikt voor de initiële configuratie van de Beep Base. Met de App kunnen de instellingen van de sensoren en meet intervallen worden aangepast, maar bijvoorbeeld ook de encryptie sleutels voor LoRaWAN worden aangepast.

# Hardware

Voor Beep is de hardware voor dit project al ontworpen: ID190222. De print heeft de volgende onderdelen:

* nRF52840 BLE low power microcontroller (BMD-340 module).
* 2x AA batterij.
* DS18B20 temperatuur probe sensor met one-wire interface.
* HX711 dubbele weegbrug sensor voor het meten van het gewicht van de bijenkast.
* SQ-SEN-645 Tilt switch: horizontaal en verticaal detecteren.
* Reed switch voor gebruiker activatie
* TPS61292 boost converter.
* TPS22917 supply switch.
* Buzzer voor audio feedback aan de gebruiker.
* Flash voor loggen van meetgegevens.
* RFM95 voor LoRaWAN communicatie.
* ATECC608A encryptie en unieke key voor DEVEUI.
* BME280 temperatuur, luchtvochtigheid en luchtdruk meten.
* TLV320ADC3100 Electret signaal conditioner en recorder voor Fourier analyse.

## nRF52840

De nRF52840 microcontroller van Nordic wordt gebruikt om de functionaliteit van de Beep Base te implementeren. De nRF52840 heeft een radio module die door middel van het SDK van Nordic bluetooth low energy ondersteund.

## AA batterijen

Om de elektronica van energie te voorzien worden er twee lithium AAA batterijen van Energizer in serie gebruikt.

## DS18B20 temperatuur sensor

Om de temperatuur op verschillende plekken in de bijenkast te meten worden er meerdere DS18B20 temperatuur probes gebruikt. Deze sensoren gebruiken een one-Wire protocol om de sensor in te stellen, een temperatuur conversie te starten en het resultaat uit te lezen.

## HX711 rekstrook sensor

De HX711 rekstrook sensor wordt gebruikt om de rekstrook te meten waarop het gewicht van de bijenkast rust. Met het meetresultaat en de gevoeligheid van de rekstrook kan het gewicht van de bijenkast worden berekend.

## Reedswitch

De gebruiker kan door middel van de reed switch de BLE communicatie activeren. Optioneel wordt de reed switch ook gebruikt om de pincode te resetten.

## TPS boost converter en supply switch

De TPS61291 boost converter wordt gebruikt om de batterij spanning te verhogen naar 3V als de batterijspanning lager is. De boost converter kan worden uitgeschakeld waarna de batterij spanning direct aan de uitgang wordt doorgegeven.

Omdat niet alle onderdelen werken of zijn gespecificeerd voor onder de 3V is er nog een voedingsschakelaar gebruikt om die onderdelen van de voedingsspanning los te koppelen.

## Buzzer

Als er met BLE parameters worden geschreven of de sensor in een nieuwe oriëntatie wordt geplaatst wordt de buzzer gebruikt voor feedback aan de gebruiker. De buzzer zal slechts een aantal tonen/melodieën ondersteunen.

## Flash storage

De MX25R6435 flash storage IC wordt gebruikt om de gemeten gegeven op te slaan. Het flash IC heeft een opslag grote van 64Mb. Met BLE kan dit vervolgens worden uitgelezen met de Beep App.

## RFM95

De RFM95 voor de 868MHz EU band wordt gebruikt voor LoRaWAN communicatie. Een antenne kan worden aangesloten door middel van een micro UFL connector.

## ATECC608A

De ATEC608A is een crypto authenticatie IC die verschillende vormen van encryptie, decryptie, hash berekeningen, een 72 bits unieke serienummer en opslag van sleutels of certificaten ondersteund. Voor de Beep Base wordt echter enkel de unieke serienummer gebruikt om de DEVEUI van af te leiden en de Beep Base hardwarematig te identificeren in het back-end.

## BME280

Temperatuur, luchtvochtigheid en luchtdruk sensor van Bosch die op een 1 meter lange kabel wordt gemonteerd, zodat deze in het bijenhok kan worden geplaatst. Wordt via I2C aangestuurd door middel van de nRF52840.

## TLV320ADC3100

De TLV320ADC3100 is een audio ADC die twee electret microfoons kan voeden en kan uitlezen. De Audio meetdata wordt door middel van een I2S interface naar de nRF52840 getransporteerd. Met een I2C interface wordt de audio ADC ingesteld op de juiste ingang en filter responses.

In de nRF52840 wordt de gemeten audio data met een FFT omgezet naar amplitudes in een aantal frequentiebanden, wat vervolgens wordt gelogd.

## Logging

Het logging protocol is nog niet gespecificeerd. Naar alle waarschijnlijkheid zal dit een ASCII protocol worden. Optioneel een binaire log in het Flash geheugen en bij het uitlezen vertalen naar een ASCII formaat.

# Software

## Beep Protocol

Het beep protocol is opgebouwd uit complementaire lees en schrijfopdrachten die worden geïdentificeerd door een enkele byte waarvan altijd de zevende bit 1 is voor schrijfopdrachten. Bijvoorbeeld het READ\_DS18B20\_CONVERSION commando met de waarde 4d/0x04h heeft een complementaire schrijf commando 132d/0x84h.

Hieronder een kort overzicht van de gedefinieerde commando’s:

|  |  |  |  |
| --- | --- | --- | --- |
| **Dec/hex** | | **Naam** | **Omschrijving** |
| 0 | 0x00 | RESPONSE | Antwoord op een schrijfopdracht |
| 1 | 0x01 | READ\_FIRMWARE\_VERSION | Lees de firmware versie uit |
| 2 | 0x02 | READ\_HARDWARE\_VERSION | Lees de hardware versie uit |
| 3 | 0x03 | READ\_DS18B20\_STATE | Lees de temperatuur resolutie en status uit. |
| 131 | 0x83 | WRITE\_DS18B20\_STATE | Beschrijf de temperatuur resolutie en status. |
| 4 | 0x04 | READ\_DS18B20\_CONVERSION | Lees de laatste temperatuur conversie waardes |
| 132 | 0x84 | WRITE\_DS18B20\_CONVERSION | Start een temperatuur conversie |
| 5 | 0x05 | READ\_DS18B20\_CONFIG | N.A. |
| 6 | 0x06 | BME280\_CONFIG\_READ | N.A. |
| 7 | 0x07 | BME280\_CONVERSION\_READ | Lees de laatste temperatuur, luchtvochtigheid en Barometrische druk conversie uit. |
| 135 | 0x87 | BME280\_CONVERSION\_START | Start een temperatuur, luchtvochtigheid en Barometrische druk conversie. |
| 8 | 0x08 | READ\_BME280\_I2C | N.A. |
| 9 | 0x09 | READ\_HX711\_STATE | Lees de HX711 conversie kanalen en aantal metingen per kanaal waarvan een gemiddelde wordt berekend |
| 137 | 0x89 | WRITE\_HX711\_STATE | Stel de conversie kanalen en aantal metingen per kanaal waarvan een gemiddelde wordt berekend in. |
| 10 | 0x0A | READ\_HX711\_CONVERSION | Lees het laatste meet resultaat |
| 138 | 0x8A | WRITE\_HX711\_CONVERSION | Start een nieuwe meting op een kanaal |
| 11 | 0x0B | READ\_AUDIO\_ADC\_CONFIG | Lees het audiokanaal, de versterking, het volume en de FFT instellingen uit. |
| 139 | 0x8B | WRITE\_AUDIO\_ADC\_CONFIG | Stel het audio kanaal, de versterking, het volume en de FFT instellingen in. |
| 12 | 0x0C | READ\_AUDIO\_ADC\_CONVERSION | Lees de laatste Audio ADC conversie uit. |
| 13 | 0x0D | START\_AUDIO\_ADC\_CONVERSION | Start een Audio ADC conversie met FFT berekening. |
| 14 | 0x0E | READ\_ATECC\_READ\_ID | Leest de ATECC ID uit het flash. |
| 15 | 0x0F | READ\_ATECC\_I2C | N.A. |
| 16 | 0x10 | READ\_BUZZER\_STATE | N.A. |
| 17 | 0x91 | WRITE\_BUZZER\_DEFAULT\_TUNE | Speelt een standaard buzzer tune aan de hand van de opgegeven index waarde |
| 18 | 0x92 | WRITE\_BUZZER\_CUSTOM\_TUNE | Speelt een buzzer tune af met de opgegeven aan en uit tijd, duty cycle en het aantal herhalingen. |
| 19 | 0x13 | READ\_SQ\_MIN\_STATE | N.A. |
| 20 | 0x14 | READ\_LORAWAN\_STATE | Lees de LoRaWAN status: aan/uit, joined, duty-cycle, Adaptive Data Rate, correcte sleutels |
| 148 | 0x94 | WRITE\_LORAWAN\_STATE | Schrijf de LoRaWAN status: aan/uit, duty-cycle, Adaptive Data Rate |
| 21 | 0x15 | READ\_LORAWAN\_DEVEUI | Lees de DEVEUI, 8 bytes |
| 149 | 0x95 | WRITE\_LORAWAN\_DEVEUI | Schrijf de DEVEUI, 8 bytes |
| 22 | 0x16 | READ\_LORAWAN\_APPEUI | Lees de APPEUI, 8 bytes |
| 150 | 0x96 | WRITE\_LORAWAN\_APPEUI | Schrijf de APPEUI, 8 bytes |
| 23 | 0x17 | READ\_LORAWAN\_APPKEY | Lees de APPKEY, 16 bytes |
| 151 | 0x97 | WRITE\_LORAWAN\_APPKEY | Schrijf de APPKEY, 16 bytes |
| 136 | 0x88 | WRITE\_LORAWAN\_TRANSMIT | Zend een LoRaWAN bericht |
| 25 | 0x19 | READ\_CID\_nRF\_FLASH | N.A. |
| 26 | 0x1A | READ\_nRF\_ADC\_CONFIG | N.A. |
| 27 | 0x1B | READ\_nRF\_ADC\_CONVERSION | Lees de laatste conversie waardes van de batterij, nRF voedingsspanning en batterij percentage. |
| 155 | 0x9B | WRITE\_nRF\_ADC\_CONVERSION | Start een ADC conversie |
| 28 | 0x1C | READ\_APPLICATION\_STATE | N.A. |
| 29 | 0x1D | READ\_APPLICATION\_CONFIG | Lees het meet interval en de verhouding tussen meten en versturen uit. |
| 157 | 0x9D | WRITE\_APPLICATION\_CONFIG | Stel het meet interval en de verhouding tussen meten en versturen in. |
| 30 | 0x1E | READ\_PINCODE | Lees de BLE pin code, 6 getallen: ‘0’ – ‘9’ |
| 158 | 0x9E | WRITE\_PINCODE | Schrijf de BLE pin code, 6 getallen: ‘0’ – ‘9’ |
| 31 | 0x1F | READ\_BOOT\_COUNT | Lees de bootcount van de Beepbase |
| 32 | 0x20 | READ\_MX\_FLASH | Commando om de log uit te lezen van het flash geheugen. Er kan een offset worden meegegeven om niet de complete flash geheugen uit te hoeven lezen. |
| 33 | 0x21 | ERASE\_MX\_FLASH | Commando om de log te wissen |
| 34 | 0x22 | SIZE\_MX\_FLASH | Commando om de grote van de log te lezen |
| 35 | 0x23 | ALARM\_CONFIG\_READ | Alarm configuratie voor een specifieke sensor uitlezen. |
| 163 | 0xA3 | ALARM\_CONFIG\_WRITE | Alarm configuratie voor een specifieke sensor instellen. |
| 36 | 0x24 | ALARM\_STATUS\_READ | De huidige actieve Alarmen uitlezen. |
| **Tabel:1** | | | |

Alle commando’s en waardes zijn in big endian.

Voor LoRaWAN is het mogelijk om meerdere opdrachten in een enkel bericht te stoppen, bijvoorbeeld:

|  |  |
| --- | --- |
| **Hex** | **Opdrachten** |
| 0102 | READ\_FIRMWARE\_VERSION, READ\_HARDWARE\_VERSION |

Bij BLE wordt er maar 1 opdracht per bericht uitgevoerd en zal per opdracht een bericht moeten worden verzonden en het eventuele antwoord worden afgevangen.

LoRaWAN maximale buffer grote is 52 bytes en het BLE control point heeft een grote van 30 bytes.

### 0d/0x00 – RESPONSE

Antwoord op een opdracht met een status indicatie van de fout of succes. Wordt enkel door de BEEPBASE verstuurdt

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| RESPONSE | Uint8\_t | 0x00 | RESPONSE opdracht ID |
| command | Uint8\_t | - | Het opdracht ID waarop een antwoord wordt gestuurd |
| Error code | Uint32\_t | - | Zie de tabel hieronder voor de error code en de omschrijving |

nRF SDK Foutcodes:

|  |  |  |
| --- | --- | --- |
| **Error code** | **#** | **Omschrijving** |
| NRF\_SUCCESS | 0 | Successful command |
| NRF\_ERROR\_SVC\_HANDLER\_MISSING | 1 | SVC handler is missing |
| NRF\_ERROR\_SOFTDEVICE\_NOT\_ENABLED | 2 | SoftDevice has not been enabled |
| NRF\_ERROR\_INTERNAL | 3 | Internal Error |
| NRF\_ERROR\_NO\_MEM | 4 | No Memory for operation |
| NRF\_ERROR\_NOT\_FOUND | 5 | Not found |
| NRF\_ERROR\_NOT\_SUPPORTED | 6 | Not supported |
| NRF\_ERROR\_INVALID\_PARAM | 7 | Invalid Parameter |
| NRF\_ERROR\_INVALID\_STATE | 8 | Invalid state, operation disallowed in this state |
| NRF\_ERROR\_INVALID\_LENGTH | 9 | Invalid Length |
| NRF\_ERROR\_INVALID\_FLAGS | 10 | Invalid Flags |
| NRF\_ERROR\_INVALID\_DATA | 11 | Invalid Data |
| NRF\_ERROR\_DATA\_SIZE | 12 | Invalid Data size |
| NRF\_ERROR\_TIMEOUT | 13 | Operation timed out |
| NRF\_ERROR\_NULL | 14 | Null Pointer |
| NRF\_ERROR\_FORBIDDEN | 15 | Forbidden Operation |
| NRF\_ERROR\_INVALID\_ADDR | 16 | Bad Memory Address |
| NRF\_ERROR\_BUSY | 17 | Busy |
| NRF\_ERROR\_CONN\_COUNT | 18 | Maximum connection count exceeded. |
| NRF\_ERROR\_RESOURCES | 19 | Not enough resources for operation |
| **Tabel: 2** | | |

Als een opdracht wordt gestuurd dat nog een aantal extra bytes verwacht, maar er worden te weinig bytes meegestuurd met de opdracht. Dan wordt de error code “Invalid Length” terug gestuurd.

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x96 | WRITE\_LORAWAN\_APPEUI |
| **Antwoord** | 0x009600000009 | RESPONSE voor WRITE\_LORAWAN\_APPEUI, error code: Invalid Length |

Als een onbekend commando wordt gestuurd wordt

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0xFE | Geen gespecificeerde opdracht |
| **Antwoord** | 0x00FE00000005 | RESPONSE voor 0xFE, error code: Not found |

### 1d/0x01 - READ\_FIRMWARE\_VERSION

Met dit commando wordt de firmware versie uitgelezen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_FIRMWARE\_VERSION | Uint8\_t | 0x01 | READ\_FIRMWARE\_VERSION opdracht ID |
| Major | Uint16\_t | - | Firmware major number |
| Minor | Uint16\_t | - | Firmware minor number |
| Sub | Uint16\_t | - | Firmware sub number |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x01 | READ\_FIRMWARE\_VERSION opdracht |
| **Antwoord** | 0x01000000000001 | READ\_FIRMWARE\_VERSION antwoord: firmware versie 0.0.1 |

### 2d/0x02 - READ\_HARDWARE\_VERSION

Met dit commando wordt de Hardware versie en ID nummer uitgelezen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HARDWARE\_VERSION | Uint8\_t | 0x02 | READ\_HARDWARE\_VERSION opdracht ID |
| Major | Uint16\_t | - | Hardware major number |
| Minor | Uint16\_t | - | Hardware minor number |
| ID | Uint32\_t | - | Hardware ID number |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x02 | READ\_HARDWARE\_VERSION opdracht |
| **Antwoord** | 0x02000100000002E70E | READ\_HARDWARE\_VERSION antwoord: Hardware versie 1.0, ID= 190222 |

Firmware en hardware uitlezen

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x0102 | READ\_FIRMWARE\_VERSION , READ\_HARDWARE\_VERSION opdracht |
| **Antwoord** | 0100000000000102000100000002E70E | READ\_FIRMWARE\_VERSION= 0.0.01  READ\_HARDWARE\_VERSION=1.0, ID= 190222 |

### 3d/0x03 - READ\_DS18B20\_STATE

Lees de temperatuur resolutie en status uit.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_DS18B20\_STATE | Uint8\_t | 0x03 | READ\_DS18B20\_STATE opdracht ID |

Antwoord2:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_DS18B20\_STATE | Uint8\_t | 0x03 | READ\_DS18B20\_STATE opdracht ID |
| status | Uint8\_t |  | Bit[0] = Aan/Uit: 0=Uit, 1=Aan  Bit [1:3] = Temperatuur Resolutie  1 = 9 Bit resolutie  2 = 10 Bit resolutie  3 = 11 Bit resolutie  4 = 12 Bit resolutie  Bit[4:7]= ongebruikt |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x03 | READ\_DS18B20\_STATE |
| **Antwoord** | 0x0309 | 0b0000 1001 =Aan/uit=1 en resolutie=4:12 bit resolutie |

### 131d/0x83 - WRITE\_DS18B20\_STATE

Zet de temperatuur resolutie en status.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_DS18B20\_STATE | Uint8\_t | 0x83 | WRITE\_DS18B20\_STATE opdracht ID |
| status | Uint8\_t |  | Bit[0] = Aan/Uit: 0=Uit, 1=Aan  Bit [1:3] = Temperatuur Resolutie  1 = 9 Bit resolutie  2 = 10 Bit resolutie  3 = 11 Bit resolutie  4 = 12 Bit resolutie  Bit[4:7]= ongebruikt |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8309 | WRITE\_DS18B20\_STATE, aan/uit=1, 12-bit resolutie |
| **Antwoord** | 0x008300000000 |  |

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8305 | WRITE\_DS18B20\_STATE, aan/uit=1, 10-bit resolutie |
| **Antwoord** | 0x008300000000 |  |

### 4d/0x04 - READ\_DS18B20\_CONVERSION

Lees de laatste temperatuur conversie waardes van de aangesloten DS18B20’s.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_DS18B20\_CONVERSION | Uint8\_t | 0x04 | READ\_DS18B20\_CONVERSION opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_DS18B20\_CONVERSION | Uint8\_t | 0x04 | READ\_DS18B20\_CONVERSION |
| N | Uint8\_t | < 10 | Aantal DS18B20 sensoren |
| Temperatuur sensor | N \* int16\_t |  | MSB van int16\_t temperatuur in honderdste graden nauwkeurigheid |

Voorbeeld enkele temperatuur sensor:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x04 | READ\_DS18B20\_CONVERSION |
| **Antwoord** | 0x04010898 | READ\_DS18B20\_CONVERSION, 1 DS18B20, temperatuur[0] = 0x0898/2200d = 22.00°C |

Voorbeeld meerdere temperatuursensoren:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x04 | READ\_DS18B20\_CONVERSION |
| **Antwoord** | 0x0403089809C4FF9C | READ\_DS18B20\_CONVERSION,  3 DS18B20 sensoren  temperatuur[0] = 0x0898/2200d = 22.00°C  temperatuur[1] = 0x09C4/2500d = 25.00°C  temperatuur[2] = 0xFF9C/-10000d = -100.00°C |

Als een gemeten temperatuur op -100.0C staat betekent dit dat de sensor een communicatie fout had tijdens het starten van de conversie of tijdens het uitlezen. Dit kan voorkomen tot nu toe als het soft device bezig is met een actie met hoge prioriteit, bijvoorbeeld het schrijven of lezen van het flash.

### 132d/0x84 - WRITE\_DS18B20\_CONVERSION

Start een temperatuur conversie voor een enkele DS18B20 met een opgegeven index of voor alle temperatuur sensoren.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_DS18B20\_CONVERSION | Uint8\_t | 0x84 | WRITE\_DS18B20\_CONVERSION opdracht ID |
| Index | Uint8\_t | < 10 | DS18B20 index. Voor waardes onder de 10 wordt er enkel een specifieke sensor gemeten. Voor waardes boven de 10 worden alle sensoren gemeten en uitgelezen. |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_DS18B20\_CONVERSION/ WRITE\_DS18B20\_CONVERSION | Uint8\_t | 0x04 | READ\_DS18B20\_CONVERSION voor alle temperatuursensoren of WRITE\_DS18B20\_CONVERSION voor een enkele specifieke sensor |
| Index | Uint8\_t | < 10 of 0xFF | Voor WRITE\_DS18B20\_CONVERSION geeft dit de specifieke sensor waarde aan.  Voor READ\_DS18B20\_CONVERSION is dit het aantal temperatuursensoren |
| Temperatuur sensor | N \* int16\_t |  | MSB van int16\_t temperatuur in honderdste graden nauwkeurigheid |

Voorbeeld enkele temperatuur sensor:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8400 | WRITE\_DS18B20\_CONVERSION, start een temperatuur conversie met de temperatuursensor op index 0. |
| **Antwoord1** | 0x008400000000 | NRF\_SUCCESS, conversie wordt gestart. Dit bericht wordt bij de LoRaWAN interface niet terug gestuurd als de error code NRF\_SUCCES is. |
| **Antwoord2** | 0x04000898 | WRITE\_DS18B20\_CONVERSION, DS18B20 temperatuur sensor op index 0, temperatuur[0] = 0x0898/2200d = 22.00°C |

Voorbeeld temperatuur index = 8, met maar 2 sensoren aangesloten

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8408 | WRITE\_DS18B20\_CONVERSION, start een temperatuur conversie met de temperatuursensor op index 8 met slechts 2 sensoren aangesloten. |
| **Antwoord** | 0x008400000007 | NRF\_ERROR\_INVALID\_PARAM voor opdracht WRITE\_DS18B20\_CONVERSION |

Voorbeeld alle temperatuur sensoren:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x84FF | WRITE\_DS18B20\_CONVERSION, start een temperatuur conversie met alle aangesloten temperatuursensor. |
| **Antwoord1** | 0x008400000000 | NRF\_SUCCESS, conversie wordt gestart. Dit bericht wordt bij de LoRaWAN interface niet terug gestuurd als de error code NRF\_SUCCES is. |
| **Antwoord2** | 0x0403089809C4FF9C | READ\_DS18B20\_CONVERSION,  3 DS18B20 sensoren  temperatuur[0] = 0x0898/2200d = 22.00°C  temperatuur[1] = 0x09C4/2500d = 25.00°C  temperatuur[2] = 0xFF9C/-10000d = -100.00°C |

Als een gemeten temperatuur op -100.0C staat betekent dit dat de sensor een communicatie fout had tijdens het starten van de conversie of tijdens het uitlezen. Dit kan voorkomen tot nu toe als het soft device bezig is met een actie met hoge prioriteit, bijvoorbeeld het schrijven of lezen van het flash.

### 7d/0x07 - BME280\_CONVERSION\_READ

Lees de temperatuur, luchtvochtigheid en barometrische druk uit van de laatste conversie.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| BME280\_CONVERSION\_READ | Uint8\_t | 0x07 | BME280\_CONVERSION\_READ opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| BME280\_CONVERSION\_READ | Uint8\_t | 0x07 | BME280\_CONVERSION\_READ |
| Temperatuur | int16\_t | - | Temperatuur in twee decimalen nauwkeurig signed integer getal |
| Relatieve luchtvochtigheid | uint16\_t | - | Relatieve luchtvochtigheid in twee decimalen nauwkeurig unsigned integer getal |
| Barometrische druk | uint16\_t | - | Barometrische druk in hPa |

**Voorbeeld:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x07 | 0x07 BME280\_CONVERSION\_READ |
| **Antwoord** | 0x07086C11D602A8 | BME280\_CONVERSION\_READ: Temp=21.56 C, RH=45.66%, Pressure=680 hPa |

### 7d/0x87 - BME280\_CONVERSION\_START

Start een temperatuur, luchtvochtigheid en barometrische druk conversie door de BME280. Als de conversie is afgelopen wordt er een BME280\_CONVERSION\_READ bericht teruggestuurd.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| BME280\_CONVERSION\_START | Uint8\_t | 0x87 | BME280\_CONVERSION\_START opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| BME280\_CONVERSION\_READ | Uint8\_t | 0x07 | BME280\_CONVERSION\_READ |
| Temperatuur | int16\_t | - | Temperatuur in twee decimalen nauwkeurig signed integer getal |
| Relatieve luchtvochtigheid | uint16\_t | - | Relatieve luchtvochtigheid in twee decimalen nauwkeurig unsigned integer getal |
| Barometrische druk | uint16\_t | - | Barometrische druk in hPa |

**Voorbeeld:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x87 | 0x87 BME280\_CONVERSION\_START |
| **Antwoord** | 0x07086C11D602A8 | BME280\_CONVERSION\_READ: Temp=21.56 C, RH=45.66%, Pressure=680 hPa |

### 9d/0x09 - READ\_HX711\_STATE

Met de READ\_HX711\_STATE opdracht worden de HX711 kanalen en het aantal samples waarover een gemiddelde wordt berekend uitgelezen vanuit het flash geheugen van de nRF52840. Bij elke meeting op het meet interval worden deze instellingen gebruikt voor de HX711.

Er kunnen meerdere kanalen worden ingesteld waarover een gemiddelde wordt berekend. Voor elk kanaal worden wordt het ingestelde aantal samples gemeten en daarover het gemiddelde berekend.

|  |  |
| --- | --- |
| **Meet kanaal** | **Waarde** |
| CH\_A\_GAIN128 | 0x01 |
| CH\_B\_GAIN32 | 0x02 |
| CH\_A\_GAIN64 | 0x04 |

Bericht opbouw:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_STATE | Uint8\_t | 0x09 | READ\_HX711\_STATE opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_STATE | Uint8\_t | 0x89 | WRITE\_HX711\_CONVERSION opdracht ID |
| Meetkanalen | Uint8\_t | 1 - 7 | HX711 meet kanalen, zie bovenstaande tabel |
| Aantal samples | Uint8\_t | >0 | Aantal samples waarover het gemiddelde wordt berekend. |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x09 | READ\_HX711\_STATE |
| **Antwoord** | 0x090102 | 0x09=READ\_HX711\_STATE  0x01= CH\_A\_GAIN128  0x02= 2 samples per kanaal |

**Voorbeeld 2:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x09 | READ\_HX711\_STATE |
| **Antwoord** | 0x09070A | 0x09=READ\_HX711\_STATE  0x07= CH\_A\_GAIN128, CH\_B\_GAIN32, CH\_A\_GAIN64  0x0A= 10 metingen per kanaal |

### 137d/0x89 - WRITE\_HX711\_STATE

Met de WRITE\_HX711\_STATE opdracht worden de HX711 kanalen en het aantal samples waarover een gemiddelde wordt berekend ingesteld. Bij elke meeting op het meet interval worden deze instellingen gebruikt voor de HX711. Deze gegevens worden opgeslagen in het flash van de nRF52840.

Bericht opbouw:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE \_HX711\_STATE | Uint8\_t | 0x89 | WRITE\_HX711\_CONVERSION opdracht ID |
| Meet kanalen | Uint8\_t | 1 - 7 | HX711 meet kanalen, zie bovenstaande tabel |
| Aantal samples | Uint8\_t | >0 | Aantal samples waarover het gemiddelde wordt berekend. |

**Voorbeeld:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x89070A | 0x89= WRITE\_HX711\_STATE  0x07= CH\_A\_GAIN128, CH\_B\_GAIN32, CH\_A\_GAIN64  0x0A= 10 metingen per kanaal |
| **Antwoord** | 0x008900000000 | 0x00=RESPONSE\_COMMAND  0x89=WRITE\_HX711\_STATE  0x00000000=NRF\_SUCCESS |

### 10d/0x0A - READ\_HX711\_CONVERSION

Lees het laatste meetresultaat met de HX711. Vanaf 1.1 ondersteund de HX711 statemachine het meten van de verschillende kanalen achter een volgend. In plaats van het aantal klokpulsen wordt nu de kanalen door gegeven waarom gemeten is of gemeten moet worden en volgen er meerdere meetresultaten in een enkel bericht. Als een kanaal niet wordt gemeten wordt er geen meetresultaat of 0 waarde meegestuurd in het resultaat bericht. In de onderstaande tabel zijn de meet kanalen te vinden met de bit waarde voor elk kanaal.

|  |  |
| --- | --- |
| **Meet kanaal** | **Waarde** |
| CH\_A\_GAIN128 | 0x01 |
| CH\_B\_GAIN32 | 0x02 |
| CH\_A\_GAIN64 | 0x04 |

Bericht opbouw:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_CONVERSION | Uint8\_t | 0x0A | READ\_HX711\_CONVERSION opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_CONVERSION | Uint8\_t | 0x0A | READ\_HX711\_CONVERSION opdracht ID |
| Meet Kanalen | Uint8\_t | 1 - 7 | HX711 meet kanalen, zie bovenstaande tabel |
| Meet resultaat | Int24\_t | - | Signed meet resultaat CH\_A\_GAIN128, CH\_B\_GAIN32 of CH\_A\_GAIN64 |
| Meet resultaat (optional) | Int24\_t | - | Signed meet resultaat CH\_B\_GAIN32 of CH\_A\_GAIN64 |
| Meet resultaat (optional) | Int24\_t | - | Signed meet resultaat CH\_A\_GAIN64 |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x0A | READ\_HX711\_CONVERSION |
| **Antwoord** | 0x0A010182E6 | READ\_HX711\_CONVERSION, kanaal 0x01:CH\_A\_GAIN128:, resultaat: 99046/0x0182e6 |

### 138d/0x8A - WRITE\_HX711\_CONVERSION

Start een nieuwe gemidelde meting op de opgegeven kanalen. Als de conversie wordt gestart volgt er eerst een bevestiging van de opdracht of een fout code indien een parameter fout is. Als het meten is voltooid worden de resultaten terug gestuurd.

|  |  |
| --- | --- |
| **Meet kanaal** | **Waarde** |
| CH\_A\_GAIN128 | 0x01 |
| CH\_B\_GAIN32 | 0x02 |
| CH\_A\_GAIN64 | 0x04 |

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_CONVERSION | Uint8\_t | 0x8A | WRITE\_HX711\_CONVERSION opdracht ID |
| Meet kanalen | Uint8\_t | 1 - 7 | HX711 meet kanalen, zie bovenstaande tabel |
| Aantal samples | Uint8\_t | >0 | Aantal samples waarover het gemiddelde wordt berekend. |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_HX711\_CONVERSION | Uint8\_t | 0x8A | WRITE\_HX711\_CONVERSION opdracht ID |
| Meet Kanalen | Uint8\_t | 1 - 7 | HX711 meet kanalen, zie bovenstaande tabel |
| Meet resultaat | Int24\_t | - | Signed meet resultaat CH\_A\_GAIN128, CH\_B\_GAIN32 of CH\_A\_GAIN64 |
| Meet resultaat (optional) | Int24\_t | - | Signed meet resultaat CH\_B\_GAIN32 of CH\_A\_GAIN64 |
| Meet resultaat (optional) | Int24\_t | - | Signed meet resultaat CH\_A\_GAIN64 |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8A010A | WRITE\_HX711\_CONVERSION, CH\_A\_GAIN128, 10 samples |
| **Antwoord1** | 0x008A00000000 | WRITE\_HX711\_CONVERSION, NRF\_SUCCESS. Wordt niet teruggestuurd bij een opdracht vanuit de LoRaWAN interface als de error code gelijk is aan NRF\_SUCCESS. |
| **Antwoord2** | 0x8A010183A4 | 0x8A = WRITE\_HX711\_CONVERSION  0x01 = CH\_A\_GAIN128  0x0183A4 = 99.236decimaal |

**Voorbeeld 2:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8A070A | WRITE\_HX711\_CONVERSION, CH\_A\_GAIN128, CH\_B\_GAIN32, CH\_A\_GAIN64 , 10 samples |
| **Antwoord1** | 0x008A00000000 | WRITE\_HX711\_CONVERSION, NRF\_SUCCESS. Wordt niet teruggestuurd bij een opdracht vanuit de LoRaWAN interface als de error code gelijk is aan NRF\_SUCCESS. |
| **Antwoord2** | 8A030183B90058D1 | 0x8A = WRITE\_HX711\_CONVERSION  0x03 = CH\_A\_GAIN128, CH\_B\_GAIN32  CH\_A\_GAIN128: 99257/0x0183b9  CH\_B\_GAIN32: 22737/0x0058d1 |

**Voorbeeld 3:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8A010A | WRITE\_HX711\_CONVERSION, CH\_A\_GAIN128, CH\_B\_GAIN32, CH\_A\_GAIN64 , 10 samples |
| **Antwoord1** | 0x008A00000000 | WRITE\_HX711\_CONVERSION, NRF\_SUCCESS. Wordt niet teruggestuurd bij een opdracht vanuit de LoRaWAN interface als de error code gelijk is aan NRF\_SUCCESS. |
| **Antwoord2** | 0x8A070183A700564100C1FE | 0x8A = WRITE\_HX711\_CONVERSION  0x03 = CH\_A\_GAIN128, CH\_B\_GAIN32  CH\_A\_GAIN128: 99257/0x0183b9  CH\_B\_GAIN32: 22737/0x0058d1  CH\_A\_GAIN64 : 49662/0x00c1fe |

### 11d/0x0B - READ\_AUDIO\_ADC\_CONFIG

Lees de Audio ADC en FFT configuratie uit.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x0B | READ\_AUDIO\_ADC\_CONFIG opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x0B | READ\_AUDIO\_ADC\_CONFIG |
| Audio kanaal | uint8\_t |  | AIN\_IN3LM = 0,  AIN\_IN2LP = 1,  AIN\_IN2RP = 2, |
| Versterking + verzwakker | uint8\_t |  | bit[7]: 1= -6dB, 0=0dB  bits[6:0]= versterking in 0.5 dB per bit:  80d/0x50 = +40.0 dB  40d/0x28 = +20.0 dB  1d/0x01 = 0.5 dB  0d/0x00 = 0 dB |
| Volume | int8\_t | -24 - 40 | Volume in 0.5 dB per bits  40d/0x28 = +20.0 dB  -24d/0xE8 = -12.0 dB |
| FFT bins | uint8\_t | 0 - 12 | Aantal bins dat het FFT resultaat naar gereduceerd wordt. |
| FFT start | uint8\_t | 0-255 | Start bin maal 2:  255 = 510 |
| FFT stop | uint8\_t | 0-255 | Stop bin maal 2  255 = 510 |

**Voorbeeld:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x0B | 0x07 READ\_AUDIO\_ADC\_CONFIG |
| **Antwoord** | 0x0B0228000A00FF | Zie onderstaande tabel |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x0B | READ\_AUDIO\_ADC\_CONFIG |
| Audio kanaal | uint8\_t | 02 | AIN\_IN2RP |
| verzwakker  Versterking | uint8\_t | 0x28 | bit[7]: 0 = 0dB  bits[6:0] = 40d/0x28 = +20.0 dB |
| Volume | int8\_t | 00 | Volume 0 dB |
| FFT bins | uint8\_t | 0x0A | 10 Bins resultaat |
| FFT start | uint8\_t | 0x00 | Start bin 0 |
| FFT stop | uint8\_t | 0xFF | Stop bin 510 |

### 139d/0x8B - WRITE\_AUDIO\_ADC\_CONFIG

Lees de Audio ADC en FFT configuratie uit.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x8B | WRITE\_AUDIO\_ADC\_CONFIGopdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x8B | WRITE\_AUDIO\_ADC\_CONFIG |
| Audio kanaal | uint8\_t |  | AIN\_IN3LM = 0,  AIN\_IN2LP = 1,  AIN\_IN2RP = 2, |
| Versterking + verzwakker | uint8\_t |  | bit[7]: 1= -6dB, 0=0dB  bits[6:0]= versterking in 0.5 dB per bit:  80d/0x50 = +40.0 dB  40d/0x28 = +20.0 dB  1d/0x01 = 0.5 dB  0d/0x00 = 0 dB |
| Volume | int8\_t | -24 - 40 | Volume in 0.5 dB per bits  40d/0x28 = +20.0 dB  -24d/0xE8 = -12.0 dB |
| FFT bins | uint8\_t | 0 - 12 | Aantal bins dat het FFT resultaat naar gereduceerd wordt. |
| FFT start | uint8\_t | 0-255 | Start bin maal 2:  255 = 510 |
| FFT stop | uint8\_t | 0-255 | Stop bin maal 2  255 = 510 |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8B0228000A00FF | Zie onderstaande tabel |
| **Antwoord** | 0x008B00000000 | NRF\_SUCCESS |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x8B | WRITE\_AUDIO\_ADC\_CONFIG |
| Audio kanaal | uint8\_t | 02 | AIN\_IN2RP |
| verzwakker  Versterking | uint8\_t | 0x28 | bit[7]: 0 = 0dB  bits[6:0] = 40d/0x28 = +20.0 dB |
| Volume | int8\_t | 00 | Volume 0 dB |
| FFT bins | uint8\_t | 0x0A | 10 Bins resultaat |
| FFT start | uint8\_t | 0x00 | Start bin 0 |
| FFT stop | uint8\_t | 0xFF | Stop bin 510 |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x8B0228000D00FF | Zie onderstaande tabel |
| **Antwoord** | 0x008B00000007 | NRF\_SUCCESS |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_AUDIO\_ADC\_CONFIG | Uint8\_t | 0x8B | WRITE\_AUDIO\_ADC\_CONFIG |
| Audio kanaal | uint8\_t | 02 | AIN\_IN2RP |
| verzwakker  Versterking | uint8\_t | 0x28 | bit[7]: 0 = 0dB  bits[6:0] = 40d/0x28 = +20.0 dB |
| Volume | int8\_t | 00 | Volume 0 dB |
| FFT bins | uint8\_t | 0x0D | 13 Bins resultaat |
| FFT start | uint8\_t | 0x00 | Start bin 0 |
| FFT stop | uint8\_t | 0xFF | Stop bin 510 |

### 12d/0x0C - READ\_AUDIO\_ADC\_CONVERSION

Lees de laatste Audio ADC FFT resultaat. De Audio ADC neemt een aantal samples die vervolgens met een FFT worden geconverteerd naar een frequentiespectrum van 512 bins. Elke bin heeft een resolutie van 3,937752016 Hz per FFT bin.

De start en stop indexes samen met het aantal bins waar de FTT naar gereduceerd wordt bepaald hoeveel FFT bins er worden opgeteldt. De berekening hiervoor is als volgt:

In dit voorbeeld is het aantal gesommeerde bins een geheel getal. Als dat niet zo worden er in het laatste resultaat bin het overgebleven aantal bins opgeteld tot aan de stop index.

De frequentie voor een resultaat bin kan als volgt worden berekend:

Voorbeeld:

start = 0, stop = 255,

|  |  |
| --- | --- |
| **Resultaat bin** | **Frequentie** |
| 0 | 0.0 Hz |
| 1 | 200.8 Hz |
| 2 | 401.6 Hz |
| 3 | 602.5 Hz |
| 4 | 803.3 Hz |
| 5 | 1004.1 Hz |
| 6 | 1204.9 Hz |
| 7 | 1405.8 Hz |
| 8 | 1606.6 Hz |
| 9 | 1807.4 Hz |

**Opdracht:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0C | READ\_AUDIO\_ADC\_CONVERSION opdracht ID |

**Antwoord:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0C | READ\_AUDIO\_ADC\_CONVERSION |
| Aantal bins N | uint8\_t | 0-12 | Aantal meetresultaten |
| Start bin | uint8\_t | 0-255 | Start bin maal 2 |
| Stop bin | int8\_t | 0-255 | Stop Bin maal 2 |
| Meetresultaten[N] | uint16\_t[N] | 0 - 65535 | Array met N meetresultaten |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x0C |  |
| **Antwoord** | 0x0C0A00FF014E00560039001B00190017001000130014000F | TLV FFT[10:0:255]  0.0 Hz = 334,  203.2 Hz = 86,  406.4 Hz = 57,  609.6 Hz = 27,  812.8 Hz = 25,  1016.0 Hz = 23,  1219.2 Hz = 16,  1422.4 Hz = 19,  1625.7 Hz = 20,  1828.9 Hz = 15 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0C | READ\_AUDIO\_ADC\_CONVERSION |
| Aantal bins N | uint8\_t | 0x0A | 10 meetresultaten |
| Start bin | uint8\_t | 0x00 | Start bin 0 |
| Stop bin | int8\_t | 0xFF | Stop Bin 510 |
| Meetresultaten[N] | uint16\_t[N] | 0 - 65535 | Array met N meetresultaten |

### 13d/0x0D - START\_AUDIO\_ADC\_CONVERSION

Start een Audio ADC conversie en een FFT berekening

**Opdracht:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| START\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0D | START\_AUDIO\_ADC\_CONVERSION opdracht ID |

**Antwoord:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0C | READ\_AUDIO\_ADC\_CONVERSION |
| Aantal bins N | uint8\_t | 0-12 | Aantal meetresultaten |
| Start bin | uint8\_t | 0-255 | Start bin maal 2 |
| Stop bin | int8\_t | 0-255 | Stop Bin maal 2 |
| Meetresultaten[N] | uint16\_t[N] | 0 - 65535 | Array met N meetresultaten |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x0C |  |
| **Antwoord** | 0x0C0A00FF012B008A004B0020001C0013001100130013000D | TLV FFT[10:0:255]  0.0 Hz = 299,  203.2 Hz = 138,  406.4 Hz = 75,  609.6 Hz = 32,  812.8 Hz = 28,  1016.0 Hz = 19,  1219.2 Hz = 17,  1422.4 Hz = 19,  1625.7 Hz = 19,  1828.9 Hz = 13 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_AUDIO\_ADC\_CONVERSION | Uint8\_t | 0x0C | READ\_AUDIO\_ADC\_CONVERSION |
| Aantal bins N | uint8\_t | 0x0A | 10 meetresultaten |
| Start bin | uint8\_t | 0x00 | Start bin 0 |
| Stop bin | int8\_t | 0xFF | Stop Bin 510 |
| Meetresultaten[N] | uint16\_t[N] | 0 - 65535 | Array met N meetresultaten |

### 20d/0x14 - READ\_LORAWAN\_STATE

Lees de LoRaWAN status: aan/uit, joined, duty-cycle, Adaptive Data Rate, correcte sleutels.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_LORAWAN\_STATE | Uint8\_t | 0x14 | READ\_LORAWAN\_STATE opdracht ID |
| Status | Uint8\_t | - | Zie onderstaande status table voor bit waardes. |

|  |  |  |
| --- | --- | --- |
| **Bit** | **Functie** | **Bit waarde** |
| 0 | Aan/uit | 0=LoRaWAN uit, 1=LoRaWAN aan |
| 1 | Joined | 0=Nog niet gejoined, 1=netwerk gejoined |
| 2 | Duty-cycle restrictie | 0=Duty cycle limitatie uit, 1= DutyCycle limitatie aan |
| 3 | Adaptive Datarate | 0=ADR uit, 1= ADR aan. |
| 4 | Sleutels correct | 0=Incorrecte sleutels, 1=Correcte sleutels, |
| 5:7 | Ongebruikt | Altijd 0 |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x14 | READ\_LORAWAN\_STATE |
| **Antwoord** | 0x141F | 0x1F = 0001 1111b: LoRaWAN aan, netwerk gejoined, DutyCycle limitatie aan, ADR aan, Correcte sleutels |

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x14 | READ\_LORAWAN\_STATE |
| **Antwoord** | 0x141B | 0x1F = 0001 1011b: LoRaWAN aan, netwerk gejoined, DutyCycle limitatie aan, ADR uit, Correcte sleutels |

### 145d/0x91 - WRITE\_BUZZER\_DEFAULT\_TUNE

Met de WRITE\_BUZZER\_DEFAULT\_TUNE opdracht wordt er een standaard ingestelde pwm patroon afgespeeld op de BEEPBASE. Tot op heden zijn er maar 3 patronen, maar dat kan door de klant worden uitgebreid.

Bericht opbouw:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_BUZZER\_DEFAULT\_TUNE | Uint8\_t | 0x91 | WRITE\_BUZZER\_DEFAULT\_TUNE opdracht ID |
| Geluids patroon index | Uint8\_t | 0-2 |  |

Geluids patronen

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Patroon** | **Duty cycle** | **Frequentie** | **Aan-tijd** | **Uit-tijd** | **Herhaling** |
| 0 | 50% | 2.8 kHz | 100ms | 1000ms | 4 |
| 1 | 50% | 2.8 kHz | 1000ms | 1ms | 1 |
| 2 | 50% | 2.8 kHz | 50ms | 100ms | 2 |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9101 | 0x91=WRITE\_BUZZER\_DEFAULT\_TUNE  0x01=PWM patroon 1 |
| **Antwoord** | 0x009100000000 | 0x00=RESPONSE\_COMMAND  0x91= WRITE\_BUZZER\_DEFAULT\_TUNE  0x00000000=NRF\_SUCCESS |

### 146d/0x92 - WRITE\_BUZZER\_CUSTOM\_TUNE

Met de WRITE\_BUZZER\_CUSTOM\_TUNE opdracht wordt er een pwm patroon afgespeeld volgens de gegeven parameters.

Bericht opbouw:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_BUZZER\_CUSTOM\_TUNE | Uint8\_t | 0x92 | WRITE\_BUZZER\_CUSTOM\_TUNE opdracht ID |
| Duty Cycle | Uint8\_t | 0-100 | Duty Cycle in procenten  1d=1%  100d=100% |
| Frequentie in /100 Hz | Uint8\_t | 1-255 | 1kHz = 10d  2kHz = 20d  2.8kHz = 28d |
| Uit-tijd | Uint16\_t | >0 | Tijd dat de PWM uit is in milliseconden |
| Aan-tijd | Uint16\_t | >0 | Tijd dat de PWM aan is in milliseconden |
| Herhalingen | iint16\_t | >0 | Aantal keer dat de aan-uit cyclus wordt herhaald. |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x92321C03E801F403 | 0x92= WRITE\_BUZZER\_CUSTOM\_TUNE  0x32=Duty Cycle: 50%  0x1C=Frequentie:2.8kHz  0x03E8=Uit-tijd:1000ms  0x01F4=Aan-tijd:500ms  0x03=Herhalingen:3 |
| **Antwoord** | 0x009200000000 | 0x00=RESPONSE\_COMMAND  0x92= WRITE\_BUZZER\_CUSTOM\_TUNE  0x00000000=NRF\_SUCCESS |

### 148d/0x94 - WRITE\_LORAWAN\_STATE

Schrijf de LoRaWAN status: aan/uit, duty-cycle, Adaptive Data Rate.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_LORAWAN\_STATE | Uint8\_t | 0x94 | WRITE\_LORAWAN\_STATE opdracht ID |
| Status | Uint8\_t | - | Zie onderstaande status table voor bit waardes. |

|  |  |  |
| --- | --- | --- |
| **Bit** | **Functie** | **Bit waarde** |
| 0 | Aan/uit | 0=LoRaWAN uit, 1=LoRaWAN aan/reset  Als deze bit 1 is wordt de LoRaWAN stack gereset en worden de sleutels en instellingen opnieuw geladen vanuit het flash. Voor LORaWAN communicatie betekent dit dat de sensor eerst opnieuw aanmeldt op het LoRaWAN netwerk en er geen antwoord komt via de LoRaWAN interface. |
| 1 | Ongebruikt | Wordt genegeerd |
| 2 | Duty-cycle restrictie | 0=Duty cycle limitatie uit, 1= Duty Cycle limitatie aan |
| 3 | Adaptive Datarate | 0=ADR uit, 1= ADR aan. |
| 4:7 | Ongebruikt | Altijd 0. Wordt genegeerd |

Voorbeeld om de LoRaWAN stack te resetten:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 940D | 0x0F = 0000 1101b: LoRaWAN aan, DutyCycle limitatie aan, ADR aan, |
| **Antwoord** | 009400000000 | NRF\_SUCCESS voor WRITE\_LORAWAN\_STATE |

Voorbeeld om de LoRaWAN testen zonder Duty-cycle limitatie:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 9409 | 0x0F = 0000 1001b: LoRaWAN aan, DutyCycle limitatie uit, ADR aan, |
| **Antwoord** | 009400000000 | NRF\_SUCCESS voor WRITE\_LORAWAN\_STATE |

### 21d/0x15 - READ\_LORAWAN\_DEVEUI

Lees de DEVEUI, 8 bytes

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_LORAWAN\_DEVEUI | Uint8\_t | 0x15 | READ\_LORAWAN\_DEVEUI opdracht ID |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x15 | READ\_LORAWAN\_DEVEUI opdracht |
| **Antwoord** | 0x150001020304050607 | READ\_LORAWAN\_DEVEUI antwoord, DEVEUI: 01020304050607 |

### 149d/0x95 - WRITE\_LORAWAN\_DEVEUI

Schrijf de DEVEUI, 8 bytes

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_LORAWAN\_DEVEUI | Uint8\_t | 0x95 | WRITE\_LORAWAN\_DEVEUI opdracht ID |
| DEVEUI | 8 x uint8\_t | - |  |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x950001020304050607 | WRITE\_LORAWAN\_DEVEUI opdracht |
| **Antwoord** | 0x009500000000 | WRITE\_LORAWAN\_DEVEUI succesvol. |

### 22d/0x16 - READ\_LORAWAN\_APPEUI

Lees de APPEUI, 8 bytes

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_LORAWAN\_APPEUI | Uint8\_t | 0x16 | READ\_LORAWAN\_APPKEY opdracht ID |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x16 | READ\_LORAWAN\_APPEUI opdracht |
| **Antwoord** | 0x160001020304050607 | READ\_LORAWAN\_APPEUI succesvol. |

### 150d/0x96 - WRITE\_LORAWAN\_APPEUI

Met deze opdracht wordt de APPEUI ingesteld. De APPEUImoet 8 bytes lang zijn

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_LORAWAN\_APPEUI | Uint8\_t | 0x96 | WRITE\_LORAWAN\_APPEUI opdracht ID |
| AppEUI | 8 x Uint8\_t |  |  |

Als het schrijven succesvol is, wordt dit door middel van een response met NRF\_SUCCESS weergegeven.

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x960001020304050607 | WRITE\_LORAWAN\_APPEUI opdracht |
| **Antwoord** | 0x009600000000 | WRITE\_LORAWAN\_APPEUI succesvol. |

### 23d/0x17 - READ\_LORAWAN\_APPKEY

Lees de APPKEY, 16 bytes

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_LORAWAN\_APPKEY | Uint8\_t | 0x17 | READ\_LORAWAN\_APPKEY opdracht ID |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x17 | READ\_LORAWAN\_APPKEY opdracht |
| **Antwoord** | 0x17000102030405060708090A0B0C0D0E0F | READ\_LORAWAN\_APPKEY succesvol. |

### 151d/0x97 - WRITE\_LORAWAN\_APPKEY

Schrijf de APPKEY, 16 bytes

Met deze opdracht wordt de APPKEY ingesteld. De APPKEY moet 16 bytes lang zijn

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_LORAWAN\_APPKEY | Uint8\_t | 0x97 | WRITE\_LORAWAN\_APPKEY opdracht ID |
| Appkey | 16 x Uint8\_t | 0 - 9 |  |

Als het schrijven succesvol is, wordt dit door middel van een response met NRF\_SUCCESS weergegeven.

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x97000102030405060708090A0B0C0D0E0F | WRITE\_LORAWAN\_APPKEY opdracht |
| **Antwoord** | 0x009700000000 | WRITE\_LORAWAN\_APPKEY succesvol. |

### 136d/0x98 - WRITE\_LORAWAN\_TRANSMIT

Zend een LoRaWAN bericht met de gegeven payload op fport 5.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_LORAWAN\_APPKEY | Uint8\_t | 0x97 | WRITE\_LORAWAN\_APPKEY opdracht ID |
| Lengte | Uint8\_t | < 28 | Maximale grote van 28 bytes |
| Payload | N x uint8\_t |  | Maximum payload van 28 Bytes, aangezien de control point een maximum grote heeft van 30 bytes. |

Als het schrijven succesvol is, wordt dit door middel van een response met NRF\_SUCCESS weergegeven.

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9810000102030405060708090A0B0C0D0E0F | WRITE\_LORAWAN\_APPKEY opdracht met een lengte van 16 bytes. |
| **Antwoord** | 0x009800000000 | WRITE\_LORAWAN\_APPKEY succesvol. Wordt enkel gestuurd wanneer de communicatie interface BLE is. |

### 27d/0x1B - READ\_nRF\_ADC\_CONVERSION

Lees de laatste conversie waardes van de batterij, nRF voedingsspanning en batterij percentage.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_nRF\_ADC\_CONVERSION | Uint8\_t | 0x1B | READ\_nRF\_ADC\_CONVERSION opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_nRF\_ADC\_CONVERSION | Uint8\_t | 0x1B | READ\_nRF\_ADC\_CONVERSION opdracht ID |
| Batterij spanning | Uint16\_t |  | Batterij spanning in millivolts |
| Voedingspanning | Uint16\_t |  | nRF52840 voeding spanning in millivolts |
| Batterij percentage | Uint8\_t | 0-100% | Batterij percentage, zie batterij spanning hoofdstuk voor de berekening. |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 1B |  |
| **Antwoord** | 1B0B530BB85C | Batterij spanning 0x0B53=2899mV, nRF spanning=3000mV, Batterij percentage 92% |

### 155d/0x9B - WRITE\_nRF\_ADC\_CONVERSION

Start een ADC conversie.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_nRF\_ADC\_CONVERSION | Uint8\_t | 0x9B | WRITE\_nRF\_ADC\_CONVERSION opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_nRF\_ADC\_CONVERSION | Uint8\_t | 0x9B | WRITE\_nRF\_ADC\_CONVERSION opdracht ID |
| Batterij spanning | Uint16\_t |  | Batterij spanning in millivolts |
| Voedingsspanning | Uint16\_t |  | nRF52840 voeding spanning in millivolts |
| Batterij percentage | Uint8\_t | 0-100% | Batterij percentage, zie batterijspanning hoofdstuk voor de berekening. |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9B | WRITE\_nRF\_ADC\_CONVERSION, start conversie |
| **Antwoord** | 0x9B0B530BB85C | Batterijspanning 0x0B53=2899 mV, nRF spanning=3000 mV, Batterij percentage 92% |

### 29d/0x1D - READ\_APPLICATION\_CONFIG

Lees het meet interval in minuten en de verhouding tussen het aantal sensor metingen en LoRaWAN berichten uit.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_APPLICATION\_CONFIG | Uint8\_t | 0x1D | READ\_APPLICATION\_CONFIG opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_APPLICATION\_CONFIG | Uint8\_t | 0x1B | READ\_APPLICATION\_CONFIG opdracht ID |
| Ratio meetingen-zenden | Uint8\_t | - | De verhouding tussen het aantal metingen en de LoRaWAN berichten die worden verzonden. Als dit getal drie is, wordt een op de drie metingen met LoRaWAN verstuurt, mits de duty-cycle dit niet beperkt. |
| Interval | Uint16\_t | >0 | Het meetinterval in minuten. |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x1D | READ\_APPLICATION\_CONFIG |
| **Antwoord** | 0x1D03000A | Ratio van 1:3 voor meten en verzenden. 1 op de drie metingen wordt met LoRaWAN verstuurt. Meet interval is 0x000A/10d minuten. |

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x1D | READ\_APPLICATION\_CONFIG |
| **Antwoord** | 0x1D000001 | Alle metingen worden verzonden.. Meet interval is 0x0001/1d minuut. |

### 157d/0x9D - WRITE\_APPLICATION\_CONFIG

Stel het meet interval en de verhouding tussen meten en versturen in.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_APPLICATION\_CONFIG | Uint8\_t | 0x9D | WRITE\_APPLICATION\_CONFIG opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_APPLICATION\_CONFIG | Uint8\_t | 0x9D | WRITE\_APPLICATION\_CONFIG opdracht ID |
| Ratio meetingen-zenden | Uint8\_t | - | De verhouding tussen het aantal metingen en de LoRaWAN berichten die worden verzonden. Als dit getal drie is, wordt een op de drie metingen met LoRaWAN verstuurt, mits de duty-cycle dit niet beperkt. |
| Interval | Uint16\_t | >0 && < 1440 | Het meetinterval in minuten. Moet minimaal 1 zijn en maximaal 1440. Anders error code NRF\_ERROR\_INVALID\_PARAM |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9D03000A | Ratio van 1:3 voor meten en verzenden. 1 op de drie metingen wordt met LoRaWAN verstuurd. Meet interval is 0x000A/10d minuten. |
| **Antwoord** | 0x009D00000000 | NRF\_SUCCES voor WRITE\_APPLICATION\_CONFIG |

Voorbeeld met incorrecte sample interval:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9D03FFFF | Ratio van 1:3 voor meten en verzenden. 1 op de drie metingen wordt met LoRaWAN verstuurd. Meet interval is 0xFFFF/10d minuten. |
| **Antwoord** | 0x009D00000007 | NRF\_ERROR\_INVALID\_PARAM |

Testen:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9D000001 | Ratio van 1:0 voor meten en verzenden. Alle metingen worden met LoRaWAN verstuurd. Meet interval is 0x0001/65535d minuten, wat hoger is dan het maximum interval van 1440 minuten. |
| **Antwoord** | 0x009D00000000 | NRF\_SUCCES voor WRITE\_APPLICATION\_CONFIG |

### 30d/0x1E - READ\_PINCODE

Lees de BLE pin code, 6 getallen: ‘0’ – ‘9’

Met deze opdracht wordt de BLE pin code uitgelezen. Het antwoord is opgebouwd volgens het WRITE\_PINCODE opdracht, maar dan met de READ\_PINCODE command. De BLE standaard specificeert een pin code van 6 getallen. De lengte moet dus altijd 6 zijn.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_PINCODE | Uint8\_t | 0x1E | WRITE\_PINCODE opdracht ID |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x1E | READ\_PINCODE opdracht |
| **Antwoord** | 0x1E06303132333435 | READ\_PINCODE antwoord, pincode 8 bytes: “012345” |

### 158d/0x9E - WRITE\_PINCODE

Met deze opdracht wordt de BLE pin code ingesteld. De pin code moet tussen 6 ASCII getallen bevatten tussen ‘0’(0x30h) en ‘9’(0x39h).

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| WRITE\_PINCODE | Uint8\_t | 0x9E | WRITE\_PINCODE opdracht ID |
| lengte | Uint8\_t | 6 | Aantal bytes van de pincode moet 6 zijn, anders errorcode: NRF\_ERROR\_INVALID\_LENGHT |
| pincode | 6 x Uint8\_t | ‘0’ - ‘9’ | Byte waardes moeten tussen de 0x30 en 0x39 zijn, anders errorcode: NRF\_ERROR\_INVALID\_DATA |

Als het schrijven succesvol is, wordt dit door middel van een response met NRF\_SUCCESS weergegeven.

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9E06303930393035 | WRITE\_PINCODE opdracht: “090905” |
| **Antwoord** | 0x009E00000000 | WRITE\_PINCODE succesvol. |

Voorbeeld van foutbericht:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x9E06003930393035 | WRITE\_PINCODE opdracht: “/090905” |
| **Antwoord** | 0x009E0000000B | WRITE\_PINCODE error: NRF\_ERROR\_INVALID\_DATA. |

### 31d/0x1F - READ\_BOOT\_COUNT

Met deze opdracht kan het aantal resets worden opgevraagd vanuit het flash geheugen.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_BOOT\_COUNT | Uint8\_t | 0x1F | READ\_BOOT\_COUNT opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_BOOT\_COUNT | Uint8\_t | 0x1F | READ\_BOOT\_COUNT opdracht ID |
| lengte | Uint32\_t | - | Boot count |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x1F | READ\_BOOT\_COUNT opdracht |
| **Antwoord** | 0x1F00000005 | Reset teller staat op 5 |

### 32d/0x20 - READ\_MX\_FLASH

Met deze opdracht kan het uitlezen van de log in het FLASH geheugen van de MX25R6435F. Met deze opdracht wordt ook een offset meegegeven vanaf het begin van de log. Hiermee is het mogelijk om een deel van het flash geheugen uit te lezen.

Als de offset waarde groter is dan de grote van het log dan wordt de complete log uitgelezen vanaf offset 0.

Als de opdracht wordt geaccepteerd, dan wordt er in de beep service een RESPONSE gestuurd met NRF\_SUCCESS. De flash data wordt verstuurd via de TX karakteristiek onder de BEEP service. Als de TX notificaties niet zijn ingeschakeld wordt het uitlezen genegeerd en een foutcode terug gegeven.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_MX\_FLASH | Uint8\_t | 0x20 | READ\_MX\_FLASH opdracht ID |
| Offset | Uint32\_t | - | Offset in bytes |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| RESPONSE | Uint8\_t | 0x00 | RESPONSE opdracht ID |
| command | Uint8\_t | 0x20 | READ\_MX\_FLASH opdracht ID |
| Error code | Uint32\_t | - | Zie tabel 2 |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x2000000000 | READ\_MX\_FLASH opdracht, offset 0 |
| **Antwoord** | 0x0020000000 | RESPONSE: READ\_MX\_FLASH, error code = NRF\_SUCCESS |

### 33d/0x21 - ERASE\_MX\_FLASH

Met deze opdracht wordt de beeplog in het flash geheugen van de MX25R6435F gewist. Er zijn echter twee erase opties: fatfs erase en full erase. Bij optie 0 wordt enkel de log verwijderd uit het bestandssysteem. Bij de erase MX optie wordt op chip niveau de complete flash opslag terug naar 1 geschreven.

Als de fatfs erase opdracht wordt geaccepteerd, dan wordt er in de beep service een RESPONSE gestuurd met een fatfs error code.

Voor de MX erase wordt er enkel NRF\_SUCCESS teruggestuurd zodra de erase klaar is. Data opslag of uitlezen tijdens het wissen is niet mogelijk. De nRF controleert op een timeout van 250 seconden. Volgens de datasheet van de MX duurt een erase maximaal 240 seconden. In het geval van een time-out tijdens een MX erase wordt er NRF\_ERROR\_TIMEOUT teruggestuurd.

Na het wissen wordt er altijd weer een nieuwe opstart bericht in de nieuwe log geschreven.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| READ\_MX\_FLASH | Uint8\_t | 0x21 | ERASE\_MX\_FLASH opdracht ID |
| Erase option | uint8\_t |  | 0 = fatfs verwijderd beeplog  1-0xFF = MX flash IC doet een full erase die max 250s kan duren. |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| RESPONSE | Uint8\_t | 0x00 | RESPONSE opdracht ID |
| command | Uint8\_t | 0x21 | ERASE\_MX\_FLASH opdracht ID |
| fatfs error Error code | Uint32\_t | - | Zie onderstaande tabel 2 |

|  |  |
| --- | --- |
| **Waarde** | **Omschrijving fatfs error code** |
| 0 | Succeeded |
| 1 | A hard error occurred in the low level disk I/O layer |
| 2 | Assertion failed |
| 3 | The physical drive cannot work |
| 4 | Could not find the file |
| 5 | Could not find the path |
| 6 | The path name format is invalid |
| 7 | Access denied due to prohibited access or directory full |
| 8 | Access denied due to prohibited access |
| 9 | The file/directory object is invalid |
| 10 | The physical drive is write protected |
| 11 | The logical drive number is invalid |
| 12 | The volume has no work area |
| 13 | There is no valid FAT volume |
| 14 | The f\_mkfs() aborted due to any problem |
| 15 | Could not get a grant to access the volume within defined period |
| 16 | The operation is rejected according to the file sharing policy |
| 17 | LFN working buffer could not be allocated |
| 18 | Number of open files > \_FS\_LOCK |
| 19 | Given parameter is invalid |
| **Tabel 2 – fatfs error codes** | |

Voorbeeld 1:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x2100 | ERASE\_MX\_FLASH opdracht, erase fatfs |
| **Antwoord** | 0x0020000000 |  |

Voorbeeld 2:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x2101 | ERASE\_MX\_FLASH opdracht, erase MX |
| **Antwoord** | 0x0020000000 | Wordt verstuurd na max 240 seconden. |

### 34d/0x22 - SIZE\_MX\_FLASH

Met deze opdracht wordt de grote van de log in het flash geheugen van de MX25R6435F opgehaald. Als de opdracht wordt geaccepteerd wordt er een antwoord gestuurd met daarin de grote van de log in bytes.

Opdracht:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| SIZE\_MX\_FLASH | Uint8\_t | 0x22 | SIZE\_MX\_FLASH opdracht ID |

Antwoord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| Command | Uint8\_t | 0x22 | SIZE\_MX\_FLASH antwoord |
| Log grote | Uint32\_t | - | Grote van de log in bytes |

Voorbeeld:

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x22 | SIZE\_MX\_FLASH opdracht |
| **Antwoord** | 0x220000FB40 | Grote van de log: 0x0000FB40/64320d bytes |

### 35d/0x23 - ALARM\_CONFIG\_READ

Lees de alarm configuratie voor een specifieke sensor.

**Opdracht:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ opdracht ID |
| Sensor type | uint8\_t | Sensor | 0 = DS18B20  1 = BME280  2 = HX711  4 = nRF ADC |

**Antwoord:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | uint8\_t | Sensor | 0 = DS18B20  1 = BME280  2 = HX711  4 = nRF ADC |
| Sensor specifieke data |  |  |  |

**DS18B20**

Als de gemeten temperatuur boven de maximum temperatuur, onder de minimum temperatuur of het absolute verschil ten opzichte van de vorige meetwaarde boven de ingestelde grenswaardes komt wordt er een alarm aangegeven. In de alarm status wordt de DS18B20 bit vervolgens op 1 gezet.

|  |  |  |
| --- | --- | --- |
| **Veld** | **Grote** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 = ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0 = DS18B20 |
| Max | Int16\_t | INT16\_MAX (32767)zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |
| Min | Int16\_t | INT16\_MIN (-32768) zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |
| Diff | uint16\_t | UINT16\_MAX (65535)zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0x2300 |
| **Antwoord** | 0x23001F40F63C03E8 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x00 | DS18B20 |
| Maximum | Int16\_t | 0x1F40 | 8000d = 80.00 C° |
| Minimum | Int16\_t | 0xF63C | -2500d = -25.00 C° |
| Verschil | uint16\_t | 0x03E8 | 1000d = 10.00 C° |

**BME280**

De BME280 meet temperatuur, luchtvochtigheid en de barometrische druk, voor elk meetresultaat is er een maximum, minimum en verschil waarde in te stellen.

|  |  |  |
| --- | --- | --- |
| **Veld** | **Grote** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 = ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x01 = BME280 |
| Temperatuur maximum | Int16\_t | INT16\_MAX (32767)zet deze controle uit |
| Temperatuur minimum | Int16\_t | INT16\_MIN (-32768) zet deze controle uit |
| Temperatuur verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Luchtvochtigheid maximum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Luchtvochtigheid minimum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Luchtvochtigheid verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Barometrische druk maximum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Barometrische druk minimum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Barometrische druk verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0x2301 |
| **Antwoord** | 0x23011F40F63C03E8232803E801F4271000C800C8 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | 0x23 = ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x01 | 0x01 = BME280 |
| Temperatuur maximum | Int16\_t | 0x1F40 | 8000 = 80.00 |
| Temperatuur minimum | Int16\_t | 0xF63C | -2500 = 25.00 |
| Temperatuur verschil | Uint16\_t | 0x03E8 | 1000 = 10.00 |
| Luchtvochtigheid maximum | Uint16\_t | 0x2328 | 9000 = 90.00% RH |
| Luchtvochtigheid minimum | Uint16\_t | 0x03E8 | 1000 = 10.00% RH |
| Luchtvochtigheid verschil | Uint16\_t | 0x01F4 | 500 = 5.00% RH |
| Barometrische druk maximum | Uint16\_t | 0x2710 | 10000 hPa |
| Barometrische druk minimum | Uint16\_t | 0x00C8 | 200 hPa |
| Barometrische druk verschil | Uint16\_t | 0x00C8 | 200 hPa |

**HX711**

Per kanaal worden dezelfde grenzen gebruikt. Het is dan ook niet de bedoeling om op een specifieke waarde een alarm te laten generen. Dat kan ik het back-end het beste worden opgelost. Maar meer op een indicatie te krijgen als bijvoorbeeld de load-cell niet meer aangesloten is.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 2 | HX711 |
| Maximum | Int32\_t | - | INT32\_MAX (0xFFFFFFFF)zet deze controle uit |
| Minium | Int32\_t | - | INT32\_MIN (0x00000000)zet deze controle uit |
| Verschil | Uint32\_t | - | Een waarde boven de 0xFFFFFFFF zet deze controle uit |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0x2302 |
| **Antwoord** | 0x230200003E80FFFFC180000003E8 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x02 | HX711 |
| Maximum | Int32\_t | 0x00003E80 | 16000 |
| Minium | Int32\_t | 0xFFFFC180 | -16000 |
| Verschil | Uint32\_t | 0x000003E8 | 1000 |

**nRF ADC**

Met de grenswaardes voor de ADC in de nRF52840 worden de batterij spanning en de nRF52 voedingsspanning gecontroleerd.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x04 | nRF ADC |
| Maximum | Uint16\_t |  | UINT16\_MAX (65535)zet deze controle uit |
| Minimum | Uint16\_t |  | UINT16\_MAX (65535)zet deze controle uit |
| Verschil | Uint16\_t |  | UINT16\_MAX (65535)zet deze controle uit |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0x2304 |
| **Antwoord** | 0x23040CE4070801F4 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_READ | Uint8\_t | 0x23 | ALARM\_CONFIG\_READ |
| Sensor type | Uint8\_t | 0x04 | nRF ADC |
| Maximum | Uint16\_t | 0x0CE4 | 3300 mV |
| Minimum | Uint16\_t | 0x0708 | 1800 mV |
| Verschil | Uint16\_t | 0x01F4 | 500 mV |

### 163d/0xA3 - ALARM\_CONFIG\_WRITE

Stel de alarm configuratie voor een specifieke sensor in. De DS18B20, BME280, HX711 en nRF ADC kunnen ingesteld worden met het volgende bericht:

**Opdracht:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITEopdracht ID |
| Sensor type | uint8\_t | Sensor | 0 = DS18B20  1 = BME280  2 = HX711  4 = nRF ADC |
| Sensor afhankelijke data payload | X | X | X |

**Antwoord:**

Als het bericht wordt geaccepteerd wordt er een status bericht terug gestuurd met NRF\_SUCCESS. Als de bericht opbouw of een parameter een invalide waarde hebben, dan wordt er een gepaste status error code terug gestuurd.

**DS18B20**

Als de gemeten temperatuur boven de maximum temperatuur, onder de minimum temperatuur of het absolute verschil ten opzichte van de vorige meetwaarde boven de ingestelde grenswaardes komt wordt er een alarm aangegeven. In de alarm status wordt de DS18B20 bit vervolgens op 1 gezet.

|  |  |  |
| --- | --- | --- |
| **Veld** | **Grote** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0x23 = ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0 = DS18B20 |
| Max | Int16\_t | INT16\_MAX (32767)zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |
| Min | Int16\_t | INT16\_MIN (-32768) zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |
| Diff | uint16\_t | UINT16\_MAX (65535)zet deze controle uit, temperatuur is in 2 decimalen nauwkeurig int16\_t |

**Voorbeeld:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x00 | DS18B20 |
| Maximum | Int16\_t | 0x1F40 | 8000d = 80.00 C° |
| Minimum | Int16\_t | 0xF63C | -2500d = -25.00 C° |
| Verschil | uint16\_t | 0x03E8 | 1000d = 10.00 C° |

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0xA3001F40F63C03E8 |
| **Antwoord** | 0x00A300000000 |

**BME280**

De BME280 meet temperatuur, luchtvochtigheid en de barometrische druk, voor elk meetresultaat is er een maximum, minimum en verschil waarde in te stellen.

|  |  |  |
| --- | --- | --- |
| **Veld** | **Grote** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 = ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x01 = BME280 |
| Temperatuur maximum | Int16\_t | INT16\_MAX (32767)zet deze controle uit |
| Temperatuur minimum | Int16\_t | INT16\_MIN (-32768) zet deze controle uit |
| Temperatuur verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Luchtvochtigheid maximum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Luchtvochtigheid minimum | Uint16\_t | 0 zet deze controle uit |
| Luchtvochtigheid verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Barometrische druk maximum | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |
| Barometrische druk minimum | Uint16\_t | 0 zet deze controle uit |
| Barometrische druk verschil | Uint16\_t | UINT16\_MAX (65535)zet deze controle uit |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | 0xA3 = ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x01 | 0x01 = BME280 |
| Temperatuur maximum | Int16\_t | 0x1F40 | 8000 = 80.00 |
| Temperatuur minimum | Int16\_t | 0xF63C | -2500 = 25.00 |
| Temperatuur verschil | Uint16\_t | 0x03E8 | 1000 = 10.00 |
| Luchtvochtigheid maximum | Uint16\_t | 0x2328 | 9000 = 90.00% RH |
| Luchtvochtigheid minimum | Uint16\_t | 0x03E8 | 1000 = 10.00% RH |
| Luchtvochtigheid verschil | Uint16\_t | 0x01F4 | 500 = 5.00% RH |
| Barometrische druk maximum | Uint16\_t | 0x2710 | 10000 hPa |
| Barometrische druk minimum | Uint16\_t | 0x00C8 | 200 hPa |
| Barometrische druk verschil | Uint16\_t | 0x00C8 | 200 hPa |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0xA3011F40F63C03E8232803E801F4271000C800C8 |
| **Antwoord** | 0x00A300000000 |

**HX711**

Per kanaal worden dezelfde grenzen gebruikt. Het is dan ook niet de bedoeling om op een specifieke waarde een alarm te laten generen. Dat kan ik het back-end het beste worden opgelost. Maar meer op een indicatie te krijgen als bijvoorbeeld de load-cell niet meer aangesloten is.

De grenswaarden zijn

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x02 | HX711 |
| Maximum | Int24\_t | - | INT32\_MAX (0x7FFFFFFF)zet deze controle uit |
| Minium | Int24\_t | - | INT32\_MIN (0x80000000)zet deze controle uit |
| Verschil | Int24\_t | - | Een waarde boven de 0xFFFFFF zet deze controle uit |

**Voorbeeld 1:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x02 | HX711 |
| Maximum | Int32\_t | 0x00003E80 | 16000 |
| Minium | Int32\_t | 0xFFFFC180 | -16000 |
| Verschil | Uint32\_t | 0x000003E8 | 1000 |

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0xA30200003E80FFFFC180000003E8 |
| **Antwoord** | 0x00A300000000 |

**Voorbeeld 2:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x02 | HX711 |
| Maximum | Int32\_t | 0x7FFFFFFF | 2147483647 |
| Minium | Int32\_t | 0x80000000 | -2147483648 |
| Verschil | Uint32\_t | 0xFFFFFFFF | 1000 |

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0xA3027FFFFFFF80000000FFFFFFFF |
| **Antwoord** | 0x00A300000000 |

**nRF ADC**

Met de grenswaardes voor de ADC in de nRF52840 worden de batterij spanning en de nRF52 voedingsspanning gecontroleerd.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0xA3 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x04 | nRF ADC |
| Maximum | Uint16\_t |  | UINT16\_MAX (65535)zet deze controle uit |
| Minimum | Uint16\_t |  | 0 zet deze controle uit |
| Verschil | Uint16\_t |  | UINT16\_MAX (65535)zet deze controle uit |

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_CONFIG\_WRITE | Uint8\_t | 0x23 | ALARM\_CONFIG\_WRITE |
| Sensor type | Uint8\_t | 0x04 | nRF ADC |
| Maximum | Uint16\_t | 0x0CE4 | 3300 mV |
| Minimum | Uint16\_t | 0x0708 | 1800 mV |
| Verschil | Uint16\_t | 0x01F4 | 500 mV |

**Voorbeeld:**

|  |  |
| --- | --- |
|  | **Hex bericht** |
| **Opdracht** | 0xA3040CE4070801F4 |
| **Antwoord** | 0x00A300000000 |

### 36d/0x24 - ALARM\_STATUS\_READ

Lees de huidige alarm status uit. Een alarm wordt gegenereerd of uitgezet bij een nieuwe meting van de betreffende sensor. De DS18B20 temperatuur, HX711, BME280 en nRF ADC hebben maximum, minimum en verschil grenswaardes waar de meetwaarde tussen moet vallen om geen alarm te genereren. De verschil grens is ten opzichte van de vorige meetwaarde, mits de vorige meetwaarde valide is en dezelfde aantal meetwaardes of resolutie heeft.

**Opdracht:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_STATUS\_READ | Uint8\_t | 0x24 | ALARM\_STATUS\_READ opdracht ID |

**Antwoord:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| ALARM\_STATUS\_READ | Uint8\_t | 0x24 | ALARM\_STATUS\_READ |
| Alarms status bits | uint8\_t |  | bit 0 = DS18B20  bit 1 = BME280  bit 2 = HX711  bit 3 = Audio ADC  bit 4 = nRF ADC |

Alarmen worden enkel gegenereerd op de DS18B20, BME280, HX711 en de nRF ADC. De index waarde wordt gebruikt om een bit hoog of laag te zetten in de alarm status byte die wordt teruggegeven.

|  |  |
| --- | --- |
| **Sensor** | **(Bit) Index** |
| DS18B20 | 0 |
| BME280 | 1 |
| HX711 | 2 |
| Audio ADC | 3 |
| nRF ADC | 4 |
| SQ\_MIN | 5 |
| ATECC | 6 |
| BUZZER | 7 |
| LORAWAN | 8 |
| MX\_FLASH | 9 |
| nRF\_FLASH | 10 |
| Application | 11 |

**Voorbeeld 1:**

|  |  |  |
| --- | --- | --- |
|  | **Hex bericht** | **Inhoud** |
| **Opdracht** | 0x24 | ALARM\_STATUS\_READ |
| **Antwoord** | 0x2404 | ALARM\_STATUS\_READ, HX711 |

## Bluetooth Low Energy

De nRF52840 zal de volgende services ondersteunen:

1. DIS: Device information service.
2. BAS: Battery service.
3. BEEP unieke service
4. Log uitlezen.
5. DFU: Nordic’s firmware update service

Het apparaat zal zich adverteren als BEEPXXXXXXXX (Voorbeeld). Waarbij de laatste 8 karakters de 4 minst belangrijke karakters zijn van het DEVEUI in hexadecimaal. Als het DEVEUI bijvoorbeeld 0x01 23 45 67 89 AB CD EF is, dan is de BLE advertentie naam BEEP89ABCDEF.

Het DEVEUI wordt afgeleid van het unieke ID van de ATECC608A.

### Pin code

De pincode is standaard “123456”. Door middel van het BEEP protocol kan de pin code worden gewijzigd. De Pin code moet van de BLE specificatie altijd bestaan uit 6 ASCII getallen (‘0’ – ‘9’).

De Pincode kan handmatig worden gereset door met een magneet de reed switch te bekrachtigen voor 30 seconden. De BEEPBASE zal direct bij het bekrachtigen van de reed switch twee korte tonen laten horen ter indicatie dat de BEEPBASE de BLE advertising heeft gestart. Als de pincode wordt gereset geeft de BEEPBASE dit aan met een lange piep met de standaard buzzer melodie 1. Melodie 1 is een lange pieptoon van 4 seconden.

### Device information service

De device information service ondersteund de volgende karakteristieken met de volgende waardes:

* Manufacturer Name String: “BEEP”
* Model Number String: “BEEPBASE”
* Serial number String: “TODO:ATTEC”
* Hardware Revision String: “1.0”
* Firmware Revision String: “0.0.1”

### Battery service

De Batterij service (BAS) geeft een grove geschatte batterij percentage aan. Het percentage is slechts een indicatie, aangezien batterijen een zeer sterke temperatuur en stroomverbruik afhankelijkheid hebben.

De batterij percentage wordt berekend over de som van 10 ADC metingen waarmee een gemiddelde batterijspanning in mV wordt berekend. Aan de hand van de nominale batterijspanning en de cutt-off batterij spanning wordt lineair een batterij percentage berekend.

Voorbeeld:

De percentages worden afgerond op hele procenten, dus het batterij percentage wordt dan 92%

### DFU

Nog niet geïmplementeerd, deze vereist altijd een bootloader met SDK15.3.

### Beep service

De beep service heeft de volgende UUID:

|  |  |
| --- | --- |
| UUID: | 1bc3f8c5-ebc6-4050-ad4b-9f71d4a647be |
| Hex UUID | {0x1b, 0xc3, 0xf8, 0xc5, 0xeb, 0xc6, 0x40, 0x50, 0xad, 0x4b, 0x9f, 0x71, 0xd4, 0xa6, 0x47, 0xbe} |

De volgende short UUID worden gebruikt voor de karakteristieken in de service:

|  |  |  |
| --- | --- | --- |
| **Karakteristiek** | **Short UUID** | **Long UUID** |
| Beep Service | 0x68A1 | BE4768A1-719F-4BAD-5040-C6EBC5F8C31B |
| DS18B20 meetresultaat | 0x68A2 | BE4768A2-719F-4BAD-5040-C6EBC5F8C31B |
| TX log | 0x68A3 | BE4768A3-719F-4BAD-5040-C6EBC5F8C31B |
| Beep Control Point | 0x68B0 | 000068B0-0000-1000-8000-00805F9B34FB |

Let op! NRF Connect geeft de Long UUIDs weer in omgekeerde volgorde!

#### DS18B20 temperatuur resultaat karakteristiek

De DS18B20 temperatuur resultaat karakteristiek geeft de laatste temperatuurmeting weer. Net als in het Beep protocol is de eerste byte het aantal sensoren en volgt er daarna een int16\_t voor elke sensor met de temperatuur in honderdste graden nauwkeurigheid. Dus om de temperatuur om te rekenen naar graden Celsius moet het temperatuur getal worden gedeeld door honderd.

|  |  |
| --- | --- |
| Byte | Inhoud |
| 0 | Aantal DS18B20 sensoren |
| (N \* 2) + 0 | MSB van int16\_t temperatuur in honderdste graden nauwkeurigheid |
| (N \* 2) + 1 | LSB van int16\_t temperatuur in honderdste graden nauwkeurigheid |

Bijvoorbeeld:

|  |
| --- |
| 01-08-98 |

Byte 0 is 1, dus maar 1 temperatuur sensoren.

De temperatuurwaarde is 0x0898/2200d, maar moet geïnterpreteerd worden als int16\_t voor signedness. Door de waarde van 2200 te delen door honderd worden de temperatuur in graden berekend: 2200 / 100 = 22.00°C.

#### TX log data

De TX log karakteristiek wordt gebruikt om de data van de flash log te versturen naar de client, zodra hiervoor de opdracht wordt gegeven met de Beep Control Point.

Als het READ\_FLASH command wordt gestuurd naar het BEEP control point met een valide offset, dan begint vervolgens de datastroom uit de log. Van alle berichten die worden ontvangen met de TX karakteristiek zijn de eerste twee bytes een frame counter in big endian, die altijd bij nul begint. Hiermee kan de cliënt controleren of dat er berichten ontbreken. De rest van alle data is log data. Alle log data bytes moeten samengevoegd worden voordat de data geïnterpreteerd kan worden volgens de omschrijving in het hoofdstuk “Flash log”.

Voorbeeld vanuit nRF Connect:

|  |
| --- |
| I 09:34:06.734 Notification received from be4768a3-719f-4bad-5040-c6ebc5f8c31b, value: (0x) 00-00-02-25-30-31-30-30-30-31-30-30-30-32-30-30-30-30-30-32-30-30-30-31-30-30-30-30-30-30-30-32-45-37-30-45-30-45-30-31-32-33-33-44-32-33-30-38-45-43-38-45-39-31-45-45-31-46-30-30-30-30-30-30-30-31-30-33-30-39-31-44-30-30-30-30-30-31-0A-03-11-31-42-30-41-41-45-30-41-41-34-36-32-30-41-30-31-30-31-38-35-38-30-30-34-30-32-30-38-35-46-30-38-35-39-0A-03-11-31-42-30-41-41-37-30-41-41-35-36-32-30-41-30-31-30-31-38-35-43-34-30-34-30-32-30-38-35-33-30-38-35-46-0A-03-11-31-42-30-41-41-44-30-41-41-38-36-32-30-41-30-31-30-31-38-34-45-46-30-34-30-32-30-38-35-33-30-38-35-33-0A-03-11-31-42-30-41-41-44-30-41-41-34-36-32-30-41-30-31-30-31-38-35-30-35-30-34-30-32-30-38-34-44-30-38-35-39-0A-03-11-31-42-30-41-41-46-30-41-41-35-36-32-30-41-30  I 09:34:06.783 Notification received from be4768a3-719f-4bad-5040-c6ebc5f8c31b, value: (0x) 00-01-31-30-31-38-34-31-42-30-34-30-32-30-38-35-33-30-38-35-39-0A-03-11-31-42-30-41-41-39-30-41-41-38-36-32-30-41-30-31-30-31-38-34-35-37-30-34-30-32-30-38-35-46-30-38-35-46-0A  received |

De oranje gekleurde bytes geven de frame counters aan met de waardes 0 en 1.

#### Beep Control point

Het control point ondersteund het beep protocol, maar kan in tegenstelling tot LoRaWAN slechts een commando per keer aan. Als er geen notificaties aan staan worden gestuurde commando’s wel uitgevoerd, maar wordt het antwoord nooit ontvangen door de zender.

## LoRaWAN

De LoRaWAN stack zal zodra de BEEPBASE in een horizontale positie is beginnen met het initialiseren van de hardware en de LoRaWAN sleutels ophalen. Als LoRaWAN is uitgeschakeld door middel van het BEEP protocol of als een van de sleutels invalide is blijft de LoRaWAN communicatie uitgeschakeld.

De LoRaWAN sleutels zijn invalide als de gehele DEVEUI, APPKEY of APPEUI 0x00 of 0xFF is. Ook al is LoRaWAN aangezet via het BEEP protocol, als een van de sleutels incorrect is dan gaat de LoRaWAN stack naar een uit stand.

Als er nieuwe LoRaWAN sleutels via het BEEP protocol worden ingesteld moet de LoRaWAN stack gereset worden, zodat de nieuwe sleutels worden geladen. Totdat dit wordt uitgevoerd gebruikt de LoRaWAN stack de oude sleutels.

Als LoRaWAN opstart omdat het via het BEEP protocol is gereset of als de BEEPBASE in een horizontale positie wordt gelegd, dan gaat deze zich aan proberen te melden bij het back-end met de aangeleverde sleutels. De LoRaWAN stack gaat dan Join Request berichten versturen. Als er een gateway binnen het bereik van de BEEPBASE is en de mote is aangemeld bij het back-end, dan krijgt de BEEPBASE een Join Accept bericht terug.

Zodra de BEEPBASE is aangemeld bij het back-end, gaat deze eerst een bericht versturen met de firmware en hardware versie en het unieke ID van de ATECC. Als het back-end dit bericht mist, kan er met een downlink altijd achterhaald worden wat de firmware en hardware versies zijn.

Als er een downlink bericht wordt ontvangen wordt de payload gecontroleerd met het BEEP protocol. Indien er valide commando’s zijn worden deze uitgevoerd. Eventuele antwoorden worden gebufferd door de LoRaWAN stack en verzonden met het eerste volgende uplink bericht.

### Standaard bericht types

De volgende bericht types zijn gedefinieerd en worden aangegeven door de Fport waarde.

|  |  |  |
| --- | --- | --- |
| **Berichttype** | **Fport** | **Bevat Beep Protocol veld** |
| Sensor on | 1 | READ\_FIRMWARE\_VERSION, READ\_FIRMWARE\_VERSION |
| Keep alive | 3 | READ\_nRF\_ADC\_CONVERSION, READ\_HX711\_CONVERSION, READ\_DS18B20\_CONVERSION |
| Alarm | 4 | De eerste byte in de payload geeft de actieve alarmen weer. |
| Uplink custom | 5 | Uplink payload wordt gespecificeerd door het beep protocol commando. |
| Downlink response | 6 | Bevat het antwoord op een downlink beep commando. |
| **Tabel 3 – Berichttypes** | | |

### Alarm bericht

Een LoRaWAN alarm bericht toont de alarmen die sinds het laatste LoRaWAN bericht actief zijn geweest. Dit betekend dat als er een alarm was gegenereerd voor een bepaalde meetwaarde, maar de volgende meetwaarde heeft de de actieve alarm status al weer gereset, dat het alarm toch wordt verstuurd.

De inhoud van de alarm byte is als volgt:

|  |  |  |
| --- | --- | --- |
| **Sensor** | **Sensor bit** | **Opmerking** |
| **DS18B20** | 0 |  |
| **BME280** | 1 |  |
| **HX711** | 2 |  |
| **Audio ADC** | 3 | Kan geen alarm genereren |
| **nRF ADC** | 4 |  |

De rest van de payload is volgens het beep protocol zoals voor een Keep-alive bericht.

**Voorbeeld:**

|  |  |
| --- | --- |
| **LoRaWAN Bericht:** | 0x041B0B370B2D640A01018BBA040208BD08D00C0A00FF0117006D0045002A001E0019001700150015000C07086C11D602A8 |
| **Inhoud:** | Alarm: 0x04, bits: HX711  Saadc: Vcc: 2871 mV, Vbat: 2861 mV, Battery: 100%  HX711: A128: 101306/0x018bba  DS18B20 2 results:[0]: 0x08BD - 22.370 C, [1]: 0x08D0 - 22.560 C  TLV FFT[10:0:255]0.0 Hz=279,401.7 Hz=109,803.3 Hz=69,1205.0 Hz=42,1606.6 Hz=30,2008.3 Hz=25,2409.9 Hz=23,2811.6 Hz=21,3213.2 Hz=21,3614.9 Hz=12  BME read: Temp=21.56 C, RH=45.66%, Pressure=680 hPa |

# Applicatie

## Buzzer geluiden

De buzzer wordt gebruikt om de gebruiker te informeren van de staat van de BEEPBASE. Via BLE of LoRaWAN berichten kan de buzzer ook worden aangestuurd. De BEEPBASE geeft de volgende status indicaties:

|  |  |
| --- | --- |
| **Staat** | **Melodie/indicatie** |
| Als de Beepbase verticaal wordt geplaatst en de Beep base uitschakelt. | Een lange piep |
| Als de BEEPBASE horizontaal wordt geplaatst of opstart nadat de batterijen zijn geplaatst. | Vier piepjes |
| Als de reed schakelaar wordt bekrachtigd met een magneet | Twee korte piepjes |

## Flash log

Om meetgegevens en andere informatie op te slaan wordt de MX25R6435 flash IC gebruikt met het fatfs bestand systeem om de gegeven op te slaan. Bij het opstarten wordt er altijd een opstart bericht geschreven naar de log met relevante gegevens zoals de bootcount en firmware en hardware versie nummers. Na elke meting op basis van het sample interval worden de meetgegevens volgens het BEEP protocol opgeslagen.

|  |
| --- |
| Data: ADC; Batterij en voedingsspanning, HX711 (1 kanaal), DS18B20 (2 temperatuur sensoren), FFT (10 resultaten), BME280  Hex payload :48\*2 + 3 = 99 bytes per keer  Binair payload :48 + 3 = 51 bytes per keer  Hex payload: 5 minuten interval = 0,8 Jaar  Hex payload 15 minuten interval = 2.4 Jaar  Binaire payload 5 minuten = 1.56 Jaar  Binaire payload 15 minuten = 4.69 Jaar |

### Bericht opbouw

Om een opstart-bericht en een meetgegevens-bericht te onderscheiden begint elk bericht met een byte die het bericht type specificeert, vervolgens een byte die het aantal data bytes op geeft. Hierna volgt de data in binair formaat. Eerdere firmware versies <= 1.2.2 hadden de payload in ASCII karakters, maar dit haalde niet de gewenste data opslag. Aan het eind van elk bericht volgt er nog een new line feed karakter ‘\n’. Dit de weergave van de data in een tekst editor zoals notepad++ makkelijker aangezien elk bericht op een aparte regel wordt weergegeven.

|  |  |  |  |
| --- | --- | --- | --- |
| **Veld** | **Grote** | **Waarde** | **Omschrijving** |
| Bericht identificatie | Uint8\_t | 1 of 3 | Bericht types zijn volgens tabel 3. Enkel het Sensor on (1) en Keep alive bericht (3) types worden gebruikt |
| Payload grote | Uint8\_t | > 0 |  |
| Payload | Uint8\_t array [Payload grote] |  |  |
| Bericht einde | Uint8\_t | ‘\n’ = 0x0A/10d |  |

Elk bericht type heeft een hexadecimale data payload met de volgende parameters die volgens het BEEP protocol zijn opgebouwd:

|  |  |  |
| --- | --- | --- |
| **Berichttype** | **Fport** | **Bevat Beep Protocol veld** |
| Sensor on | 1 | READ\_FIRMWARE\_VERSION = 1d,  READ\_FIRMWARE\_VERSION = 2d,  READ\_ATECC\_READ\_ID = 14d,  READ\_BOOT\_COUNT = 31,  READ\_DS18B20\_STATE = 3,  READ\_APPLICATION\_CONFIG = 29, |
| Keep alive | 3 | READ\_nRF\_ADC\_CONVERSION = 27,  READ\_HX711\_CONVERSION = 10,  READ\_DS18B20\_CONVERSION = 4 |
| Alarm | 4 | Niet gebruikt |
| Uplink custom | 5 | Niet gebruikt |
| Downlink response | 6 | Niet gebruikt |
| **Tabel 4 – Bericht inhoud** | | |

### Opstart-bericht

Voorbeeld van een opstart bericht:

|  |
| --- |
| 0x02250100010003000002000100000002e70e0e01233d2308ec8e91ee1f0000000b03091d0000010a |

**Bericht opbouw:**

|  |  |
| --- | --- |
| **Veld** | **Inhoud** |
| Berichttype | 0x02/2d |
| Payload lengte | 0x25/37d data bytes, |
| Payload[37] | 0100010003000002000100000002e70e0e01233d2308ec8e91ee1f0000000b03091d000001 |
| Bericht einde | 0x0A/10d ‘\n’ |

**Payload parameters**

|  |  |  |
| --- | --- | --- |
| **Hex** | **ID** | **Parameters** |
| 01000100030000 | READ\_FIRMWARE\_VERSION = 1d | Firmware version: 1.3.0 |
| 02000100000002e7 | READ\_FIRMWARE\_VERSION = 2d | Hardware version: 1.0 ID:190222 |
| 0e0e01233d2308ec8e91ee | READ\_ATECC\_READ\_ID = 14d | ATECC ID: 01233D2308EC8E91EE |
| 1f0000000b | READ\_BOOT\_COUNT = 31 | Boot count: 11 |
| 0309 | READ\_DS18B20\_STATE = 3 | DS18B20 state: 9 |
| 1D000001 | READ\_APPLICATION\_CONFIG = 29 | App Config: ratio:0, interval 1 min |

### Meetgegevens bericht

**Voorbeeld van een meetbericht:**

|  |
| --- |
| 0x03301b0a410a33590a0101893c040208e308e30c0a00ff00710014000f000d000d000a000a00090007000707086c11d602a80a |

**Bericht Opbouw:**

|  |  |
| --- | --- |
| **Veld** | **Inhoud** |
| Berichttype | 0x03/3d |
| Payload lengte | 0x30/48d data bytes |
| Payload[48] | 1b0a410a33590a0101893c040208e308e30c0a00ff00710014000f000d000d000a000a00090007000707086c11d602a8 |
| Bericht einde | 0x0A/10d ‘\n’ |

**Payload parameters**

|  |  |  |
| --- | --- | --- |
| **Hex** | **ID** | **Parameters** |
| 0x1B0AAE0AA462 | READ\_nRF\_ADC\_CONVERSION 0x1B/27d | Vcc: 2734 mV, Vbat: 2724 mV, Battery: 98% |
| 0x0A01018580 | READ\_HX711\_CONVERSION 0x0A/10d | HX711: A128: 99712/0x018580 |
| 0x0402085f0859 | READ\_DS18B20\_CONVERSION 0x04/4d | DS18B20 2 results:  [0]: 0x085F - 21.430 C,  [1]: 0x0859 - 21.370 C |
| 0x0c0a00ff00710014000f000d000d000a000a000900070007 | READ\_AUDIO\_ADC\_CONVERSION 0x0C/12d | TLV FFT[10:0:255]  0.0 Hz = 113,  203.2 Hz = 20,  406.4 Hz = 15,  609.6 Hz = 13,  812.8 Hz = 13,  1016.0 Hz = 10,  1219.2 Hz = 10,  1422.4 Hz = 9,  1625.7 Hz = 7,  1828.9 Hz = 7, |
| 0x07086c11d602a8 | BME280\_CONVERSION\_READ 0x07/7d | Temp=21.56 C,  RH=45.66%,  Pressure=680 hPa |

# Programmeren

## BEEPBASE

De Beepbase moet na assemblage worden geprogrammeerd. Vanuit Ideetron worden zip en hex files aangeleverd per release. De Hex file is voor het programmeren met een programmer, bijvoorbeeld een ARM flasher of een nRF528xx development board. De zip bestanden zijn voor firmware update via BLE, bijvoorbeeld de nRF Connect desktop of telefoon App.

Om de zip en hex files te generen wordt een batch file gebruikt die de bootloader, applicatie en bootloader settings onder “Release” configuratie compileert en samenvoegt.

Voor de zip files wordt er ook nog gebruik gemaakt van encryptie door middel van nrfutil.exe, zodat enkel firmware die met dezelfde sleutels is gecompileerd als de bootloader geaccepteerd wordt door de bootloader.

Er worden altijd twee zip files aangeleverd: enkel de applicatie en bootloader, softdevice en applicatie in een. Die laatste is erg handig tijdens ontwikkeling, zodat altijd alle firmware compatibel is met elkaar en niet dat een oude bootloader een nieuwere applicatie niet kan laden. Hiervoor is wel de bootloader en applicatie versie nummer controle uitgeschakeld, omdat die anders niet accepteert dat bootloader of applicatie firmware met hetzelfde versienummer wordt overschreven. Als dit niet is uitgeschakeld weigert de bootloader de complete firmware update.

Voor de uiteindelijk product release is het aan de klant of dat de versie nummer controle weer ingeschakeld moet worden. Het voegt namelijk wel wat extra eisen en controle toe aan firmware releases en de firmware kan niet meer gedowngrade worden naar een vorige versie.

## Programeer script.

Om de beepbase met de aangeleverde hex file te programmeren wordt de volgende batch file gebruikt:

|  |
| --- |
| @ECHO OFF  SET hw\_major=1  SET hw\_minor=0  SET hw\_ID=190222  SET /A hw\_reg\_val=%hw\_major%\*65536 + %hw\_minor%  SET jlink\_id=682613435  ECHO Start programming HW %hw\_major%.%hw\_minor%; reg:%hw\_reg\_val%  start /B /wait nrfjprog --snr %jlink\_id% --eraseall  start /B /wait nrfjprog --snr %jlink\_id% --memwr 0x10001080 --val %hw\_reg\_val%  start /B /wait nrfjprog --snr %jlink\_id% --memwr 0x10001084 --val %hw\_ID%  start /B /wait nrfjprog --snr %jlink\_id% --program Release/Beepbase.hex  start /B /wait nrfjprog --snr %jlink\_id% --reset  ECHO Programming Done  GOTO End  :End  pause |

**Dit batch script zet niet de readback protectie aan!**

Met de SET hw\_major, hw\_minor en hw\_ID worden de hardware versie en id nummers in de UUICR geprogrammeerd. Deze waardes worden in de firmware gebruikt voor het tonen van de hardware versie in de DIS service en kunnen worden uitgelezen via het BEEP protocol.

Bovenstaande batch file heeft wel een vast Jlink ingesteld met ID 682613435. Dit zal aangepast moeten worden voor de programmer die wordt gebruikt. Het ID kan ook worden weggelaten, dan toont de driver een pop-up met de beschikbare interface bij elk batch commando als er meerdere programmers zijn.

## nRFutil

Om de zip bestanden te creëren wordt er gebruik gemaakt van nRFutil.exe, een programma van Nordic dat hex bestanden kan encrypten voor DFU. Zorg er voor dat die in de map Util staat, aangezien deze executable niet in de repository systeem zit vanwege de grote van de applicatie. > 10Mb.

Voor ontwikkeling is nRFUtil 5.2.0 gebruikt. Oudere versies kunnen problemen opleveren die geen duidelijke fout weergeven.

## Segger Embedded Studio

Voor de ontwikkeling van de firmware is er gebruik gemaakt van Segger Embedded Studio, oftewel SES afgekort in de Nordic SDK. Om dit programma te gebruiken is er wel een gratis licentie nodig die wordt gekoppeld aan een hardware ID van de PC.

## Applicatie debuggen

Om de applicatie te debuggen met de DFU service is er een bootloader nodig die de CRC check niet uitvoert en mee wordt geladen tijden het debuggen. Om een bootloader te compileren is er een batch file gemaakt genaamd “Compile\_Bootloader\_SkipCRC” die de bootloader compileert onder een release die de CRC check negeert.

Als de bootloader firmware is aangepast of de source code is gedownload dient dit script eerst een keer te worden uitgevoerd om de hex file te creëren. SES zal bij het laden van alle files voor het debuggen wel een waarschuwing geven als de hex file ontbreekt.

Mocht er al een bootloader aanwezig is in de nRF52840 dan moet die eerst gewist worden om de nieuwe bootloader zonder crc check te programmeren.

## Compilatie scripts

De volgende batch script zijn er in de source code in de map util/Program:

|  |  |
| --- | --- |
| **Batch file naam** | **Functie** |
| Compile\_Beep\_release | Compileert release versies van de applicatie en bootloader. Vervolgens worden die samengevoegd met het soft device om de hex file te creëren. Vanuit de verschillende gecreëerde batch hex files worden de zip file gecreëerd voor het updaten van de firmware over BLE |
| Compile\_Bootloader\_SkipCRC | Compileert een bootloader die geen CRC check uitvoert. Nodig voor het debuggen van een applicatie in SES. |
| EnableRBP | Zet de readback protectie aan. Hierna moet de BEEPBASE worden gerecoverd waarbij de volledig FLASH geheugen van de nRF52840 wordt gewist. |
| erase | Hiermee wordt een microcontroller die aan een programmer gewist. |
| erase\_682613435 | Hiermee wordt een microcontroller die aan een programmer met id 682613435 gewist. |
| FICR\_read | Batch file waarmee enkele hardware parameters van een nRF52840 chip kan worden uitgelezen, zoals silicon version. |
| program\_BeepFirmware | Batch file die een aangesloten BeepBase programeerd met de gecompileerde hex file in de release map. |

# Elektrisch