

# Delphis v3.4

## User Guide

## 1. Introduction

Delphis' are a low-cost, low-power, miniature acoustic communication and ranging device for underwater vehicles, divers and subsea instruments. Data messages may be exchanged between units and an efficient "ping" protocol is implemented for range measurement by transponder operation. If multiple units are deployed in known locations, then long baseline positioning (LBL) operation is possible.

## 2. Specification

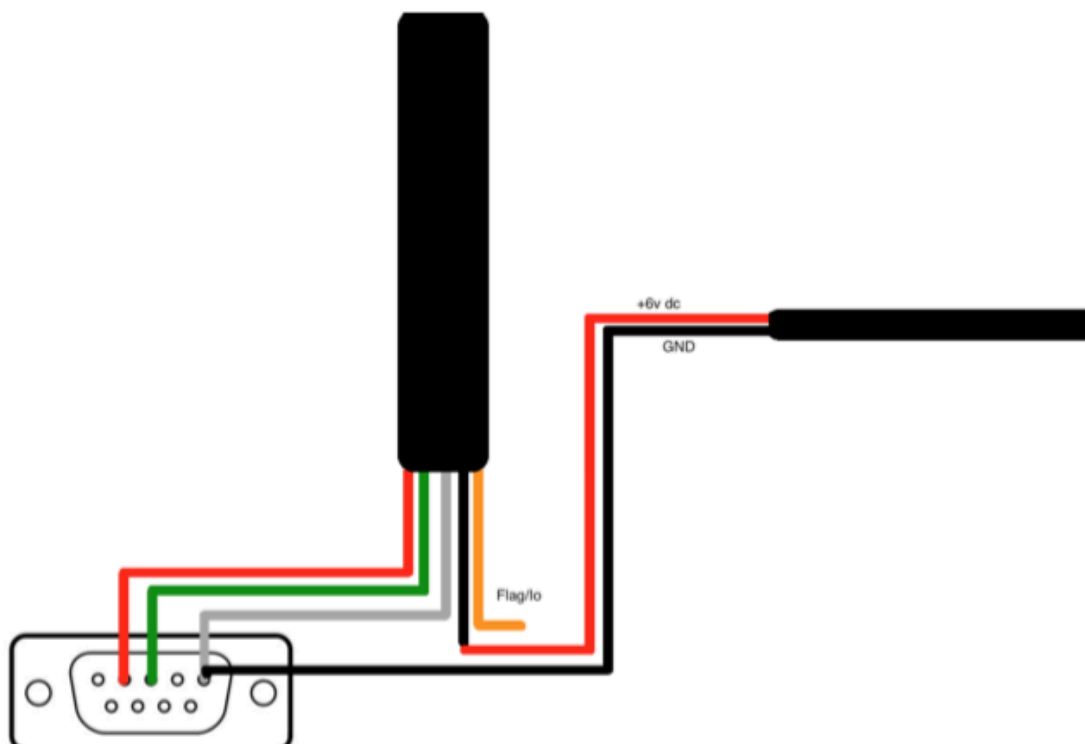
Supply voltage	3- 6.5v dc (5V or 6V supply recommended)
Supply current (5V supply)	Listening: 2.5mA Receiving: 5mA Transmitting: max 300mA
Acoustic frequency	24-32kHz
Acoustic source level	~168 dB re 1uPa @ 1m
Acoustic directivity	Near omnidirectional (reduction around cable entry of potted unit).
Physical Layer	Aperiodic orthogonal code keying with BPSK modulation and error correction code.
Acoustic data rate (raw)	640 bits/s, unicast and broadcast data messages up to 64 bytes in length.
Acoustic throughput (max)	463 bits/s
Addressing	Up to 256 units (addresses 0-255)
Ranging increment	4.7cm (c=1500m/s)
Ranging variance	~10 cm
Maximum range	> 2 km
RS232 interface	9600 Baud, 8-bit, no parity, 1 stop bit, no flow control

## Wiring Diagram – Potted Version

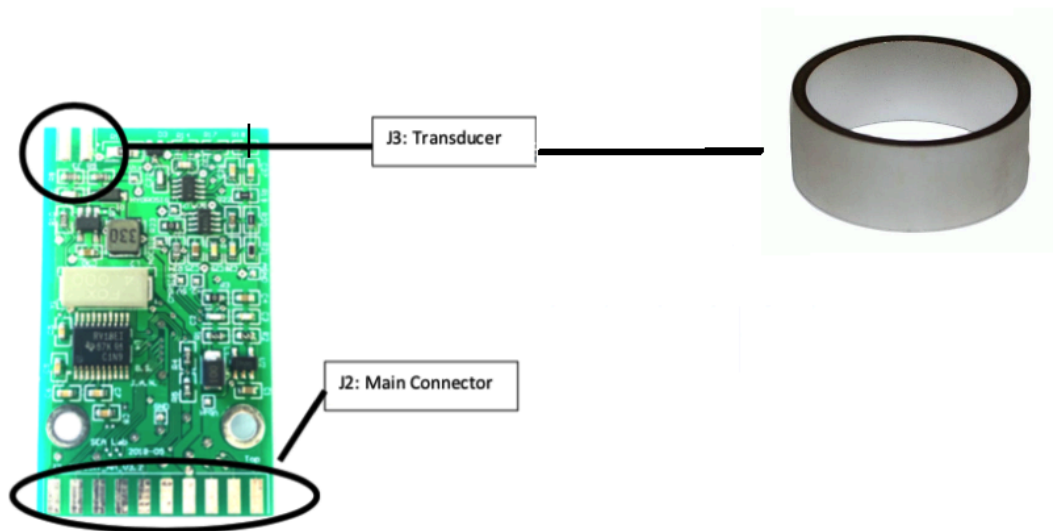


Colour	Signals
Black	Power + (positive)
White	Power – (negative)
Red	RS232 TDX
Green	RS232 RDX
Orange	Flag IO

## DB9 Connector



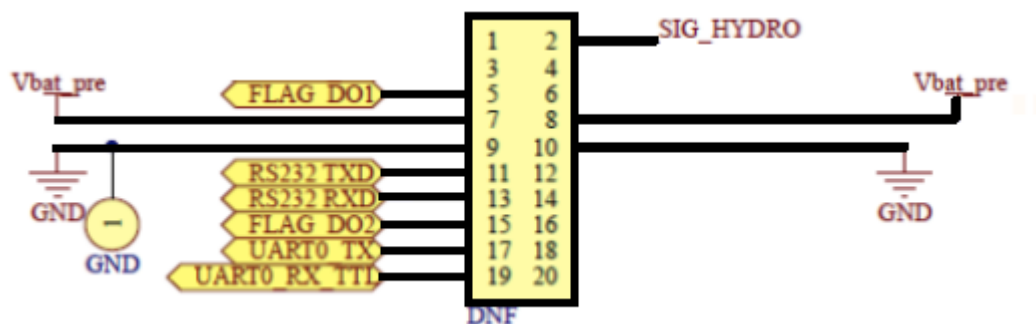
## Wiring Diagram – PCB Version



J3 Transducer

1	Inner electrode (signal)
2	Outer electrode (GND)

J2 Main connector



### 3. Basic connection

1. Connect 6V battery pack or power supply (5V recommended) positive (Black) and GND (White) on each Delphis. (Each Delphis will start up in receiving mode and draw no more than 2.5mA of current). **A short audible tone will be emitted by the modem to confirm startup.**
2. Connect unit(s) via RS232 serial cable (TXD, RXD, GND) to PC running a terminal programme (Termite is highly recommended) and configured with serial port settings (9600, 8, n, 1).

### 4. Serial communication protocol

Serial communication with the modem is possible using RS232 or TTL (2.5V) UART. Both use 9600 baud, 8 data bits, no parity, one stop bit, and no flow control.

All commands issued to the modem are prefixed with '\$' and require no terminating characters.

All responses from the modem are prefixed with '\$' (for a local acknowledgement) or '#' (for the result of an executed command) and terminated with <CR><LF>.

Unrecognised or invalid commands return 'E' to indicate an error.

Serial commands should be sent to the modem as a contiguous string with no more than 2 milliseconds between bytes. If the delay between bytes exceeds this the serial handler will time out and return 'E'.

### 5. Test Environment.

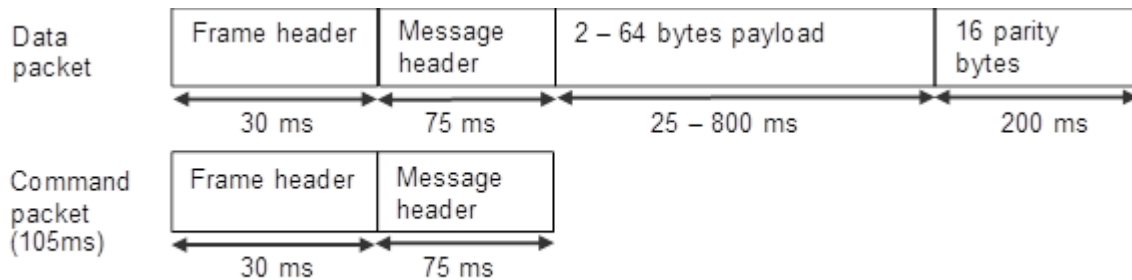
It is strongly recommended that air testing is the best option for short range lab and hardware/software integration testing rather than small volumes of water e.g. buckets, tubs and pools

#### 5.2 Acoustic packet durations

The following information can be used to calculate the expected transmission times for various acoustic message exchanges. Acoustic propagation delays should be added in calculating the expected duration of bidirectional data exchanges and this also excludes the time taken for serial data transmission at 9600 Baud. All values are in seconds

Ping request/response	0.105
Command message (\$Vxxx, \$Txxx)	0.105
Data message (n bytes)	$0.105 + (n+16) * 0.0125$

## 5.3 Acoustic packet format



The acoustic packet format for both data packets and ping/command packets is shown above. All packets have a robust frame header waveform which the modems detect, followed by a message header which contains message length, address and control information. The message header has error control coding to maximise reliability. The variable length data payload has a fixed amount of redundancy (16 bytes) for an error correction code which enables up to 8 bytes to be corrected in any packet. If all errors cannot be corrected then a packet is rejected. Communication performance may be monitored by using the \$Q command to report the number of errors corrected – packet reliability can be increased if necessary by reducing the payload size (and hence increasing the error rate that may be corrected).

## 5.4 Acoustic receive flags RxS and RxM

When the start of any acoustic packet is detected by a Delphis modem, the RxS flag is raised. The timing of this rising edge coincides precisely with the detection of the packet header waveform and so it may be used for time difference of arrival (TDOA) estimates where multiple Delphis modems are placed in an array. The RxS flag returns to zero at the end of the acoustic packet.

When a Delphis receives a unicast data message addressed to that unit, the RxM flag is raised for a short period corresponding to the transmissions of the received serial data. This signal may be used, for example, to wake up connected circuitry from a low power state.

## 5.5 In air acoustic testing

Delphis modems can communicate through air over at least 5m in a typical office/lab environment and **this is the recommended set up for short range hardware/software integration testing**. For best results in air, modems should be aligned end to end i.e. with the circular faces pointed at each other (this does not apply in water where the transducer beam pattern is more omnidirectional). Ranging information is accurate if you apply the speed of sound in air which is  $c = 340$  m/s.

## 6. Modem Commands

### \$? – Query Status

#### Command

**\$?** requests the modem status.

#### Response

**#A<xxx>V<yyyy>R<aaa>.<bbb>.<ccc>B<YYYY-MM-DDThh:mm:ss><CR><LF>** where <xxx> is node address and <yyyy> is an unsigned 16-bit supply voltage monitor value. To convert to a voltage: volts = <yyyy> \* 15/65536. R<aaa>.<bbb>.<ccc> is the release using semantic versioning e.g. R001.001.000. B is the build date and time in ISO8601 format.

#### Example

##### Local Device

```
> $?  
#A007V21996R001.001.000B2021-12-08T17:05:16<CR><LF>
```

### \$A – Set Address

#### Command

**\$A<xxx>** sets the modem address where <xxx> is ASCII decimal number between 000 and 255.

#### Response

**#A<xxx><CR><LF>** confirms node address has been set to <xxx> and stored in non-volatile memory or **E<CR><LF>** on error.

#### Example

##### Local Device

```
> $A007  
#A007<CR><LF>  
> $A500  
E<CR><LF>
```

### \$B - Broadcast Data

#### Command

**\$B<xx><data>[T<z\*14>]** transmits a message <data> of <xx> bytes to all units within range. Data can be any printable or non-printable byte value. <xx> is the number of transmitted bytes as two ASCII decimal digits (02-64). If the System Timer is enabled, then an optional transmit time can be appended.

#### Response

**\$B<xx><CR><LF>** is returned immediately to confirm <xx> bytes will be broadcast or **E<CR><LF>** on error.

#### Remote Device

**#B<aaa><yy><data>[Q<zz>D<+ddd>][T<t\*14>]<CR><LF>** is produced at the serial output on the receiving modem. Where <aaa> is the transmitting address, <yy> is

the number of bytes <data> received. If the Link Quality Indicator is enabled then additional statistics follow with quality, Q, as a number <zz> from 00-99. Doppler tracking is based on the average sample per symbol change over the whole packet. Fractional-sample per sample,  $s = \pm ddd / 1000 \%$ . To convert to a relative velocity  $v = c \times ddd / 100\,000$  where  $c$  is the sound velocity (assume 1500m/s if no data is available). Positive values indicate the units are moving apart, i.e. increasing range. If the system timer module is enabled, the optional timestamp will also follow, T, where  $\langle t * 14 \rangle$  is the absolute system time count at 1MHz. The timestamp is taken at the point the frame synch is detected.

## Example

### Local Device

```
> $B05Hello  
$B05<CR><LF>
```

### Remote Device Default

```
#B00705Hello<CR><LF>
```

### Remote Device with LQI Enabled

```
#B00705HelloQ56D+000<CR><LF>
```

## \$C – Channel Impulse Response

### Command

`$C<m><xxx>` transmits a ping request to modem <xxx> and process the returned ack to produce a channel impulse response. <m> can be M for magnitude, or C for complex values.

### Response

`$C<m><xxx><CR><LF>` is returned immediately or `E<CR><LF>` on error. On receipt of the ack `#C<m><xxx>T<tttt>L<zzzzz><vv*L><CR><LF>` is returned or `#TO<CR><LF>` on timeout. Where the propagation time <tttt> is a 16 kHz counter.  $\text{Range} = \langle tttt \rangle \times c \times 3.125 \times 10^{-5}$  where  $c$  is the sound velocity (assume 1500m/s if no data is available). The impulse response values of length <zzzzz> follow, where for Magnitude they are binary uint16 values over two consecutive bytes, and for Complex they are alternating binary int16 values over two consecutive bytes each for Real and Imaginary. Samples for correlation are taken at 16 kHz. For Magnitude mode the results cover the period -6.25 ms to 18.75 ms, whilst the Complex results cover -3.125 ms to 9.375 ms. The cross-correlation results are fully normalised using the received signal energy. To convert the integer units for Magnitude mode divide the uint16 values by 50 000.0 to produce a result between 0.0 and 1.0. For Complex mode divide the int16 value by 25000.0 to produce a result between -1.0 and 1.0.

## Example

### Local Device

```
> $CM100  
$CM100<CR><LF>  
#CM100T32000L00400<vvvv...vvvv><CR><LF>  
> $CM101  
$CM101<CR><LF>  
#TO<CR><LF>
```



## \$D – Device Unique ID

### Command

\$D request the device's unique id.

### Response

#DID:0x<a\*20>-A8:0x<b\*2>-A16:0x<c\*4>-A24:0x<d\*6>-A32:0x<e\*8><CR><NL> where <a\*20> is a hex encoded string of the 80-bit unique ID, and the <b...>,<c...>,<d...>,<e...> are hex encoded strings of the 8, 16, 24, and 32-bit CRC values generated from the 80-bit unique ID.

### Example

#### Local Device

```
> $D
#DID:0x00290035401151334E45-A8:0x99-A16:0x760E-A24:0x332545-A32:0xEF9C2B6A
```

## \$E – Echo Data

### Command

\$E<xxx><yy><data> transmits a message <data> of length <yy> to modem <xxx>. The remote device will rebroadcast the message <data> but will not pass the received data to the UART. This is a test and diagnostic function.

### Response

\$E<xxx><yy><CR><LF> is returned immediately or E<CR><LF> on error. If the transmission is successfully received by the remote device a broadcast packet will be transmitted and received by the local device as #B<xxx><yy><data>[Q<zz>D<+ddd>][T<t\*14>]<CR><LF>. See Broadcast section for parameter details of received broadcast data.

### Example

#### Local Device Default

```
> $E10005Hello
$E10005<CR><LF>
#B10005Hello<CR><LF>
```

#### Local Device with LQI Enabled

```
> $E10005Hello
$E10005<CR><LF>
#B10005HelloQ65D+000<CR><LF>
```

## \$F – Frame Beacon

### Command

\$F[T<z\*14>] transmits a frame beacon to all units within range. If the System Timer is enabled, then an optional transmit time can be appended.

### Response

\$F<CR><LF> is returned immediately to confirm the frame beacon will be broadcast or E<CR><LF> on error.

## Remote Device

#F<aaa>[T<t\*14>]<CR><LF> is produced at the serial output on the receiving modem.

Where <aaa> is the transmitting address. If the system timer module is enabled, the optional timestamp will also follow, T, where <t\*14> is the absolute system time count at 1MHz. The timestamp is taken at the point the frame synch is detected.

## Example

### Local Device

```
> $F
#F<CR><NL>
```

### Remote Device

```
#F007<CR><NL>
```

## \$H – Help Message

### Command

\$H tells the modem to print the help message.

### Response

Full printout of available commands.

## Example

### Local Device

```
> $H
-----
Modem Console Help
-----
$?                - Query Status - Address, Voltage, Build Version.
$A<xxx>           - Set Address to <xxx>.
$B<xx><data>[T<y*14>] - Broadcast Message <data> of length <xx>.
                   Optional:[Transmit at system time count <y*14>].
$C<m><xxx>         - Channel Impulse Response <M>ag/<C>omp from modem <xxx>.
$D                - Device Unique ID.
$E<xxx><yy><data>   - Echo Message <data> of length <yy> from modem address <xxx>.
$F[T<z*14>]        - Frame Beacon broadcast.
                   Optional:[Transmit at system time count <z*14>].
$H                - Print this Help message.
$M<xxx><yy><data>[T<z*14>] - Unicast & Ack Message <data> of length <yy> to modem <xxx>.
                   Optional:[Transmit at system time count <z*14>].
$N                - Noise Measurement over 1 second.
$P<xxx>           - Ping Modem address <xxx>.
$Q                - Corrected Errors in Last Message.
$R                - Reset the modem.
$S                - Spectrum Measurement over 1 second.
$T<xxx>           - Test Message from modem address <xxx>.
$U<xxx><yy><data>[T<z*14>] - Unicast Message <data> of length <yy> to modem <xxx>.
                   Optional:[Transmit at system time count <z*14>].
$V<xxx>           - Battery Voltage and Noise from modem <xxx>.
$X                - Extension Messages:
$XF<x>            - Flag RxS also high during transmit <E>nable/<D>isable.
$XQ<x>            - Link Quality Indicator <E>nable/<D>isable.
$XT<x>            - 1MHz System Time <E>nable/<D>isable/<C>lear/<G>et Count.
$Z                - Sleep and Wake On Serial
-----
```

## \$M – Unicast Data with Ack

### Command

`$M<xxx><yy><data>[T<z*14>]` transmits a message <data> of <yy> bytes to modem <xxx>. Data can be any printable or non-printable byte value. <yy> is the number of transmitted bytes as two ASCII decimal digits (02-64). If the System Timer is enabled, then an optional transmit time can be appended.

### Response

`$M<xxx><yy><CR><LF>` is returned immediately or `E<CR><LF>` on error. If the remote device successfully receives the data it will transmit an ack. The local device will then return `#R<xxx>T<ttttt><CR><LF>` or `#TO<CR><LF>` on timeout. Where the propagation time <ttttt> is a 16 kHz counter. Range = <ttttt> × c × 3.125 × 10<sup>-5</sup> where c is the sound velocity (assume 1500m/s if no data is available).

### Remote Device

`#U<yy><data>[Q<zz>D<+ddd>][T<t*14>]<CR><LF>` is produced at the serial output on the receiving modem. Where <yy> is the number of bytes <data> received. Additional statistics follow with quality, Q, as a number <zz> from 00-99. Doppler tracking is based on the average sample per symbol change over the whole packet. Fractional-sample per sample, s = ±<ddd>/1000 %. To convert to a relative velocity v = c × <ddd>/100 000 where c is the sound velocity (assume 1500m/s if no data is available). Positive values indicate the units are moving apart, i.e. increasing range. If the Link Quality Indicator is enabled, then the quality statistics will follow. If the system timer module is enabled, the optional timestamp will also follow, T, where <t\*14> is the absolute system time count at 1MHz. The timestamp is taken at the point the frame synch is detected.

### Example

#### Local Device

```
> $M10005Hello
$M10005<CR><LF>
#R100T23000<CR><LF>
```

#### Remote Device Default

```
> $
#U05Hello<CR><LF>
```

#### Remote Device with LQI Enabled

```
> $
#U05HelloQ56D+000<CR><LF>
```

## \$N – Noise Measurement

### Command

`$N` starts a local noise measurement process, sampling over 1 second.

### Response

`#NR<xxxxxx>P<yyyyyy>M<zzzzzz><CR><LF>` where <xxxxxx> is RMS noise, P is peak-to-peak noise, and M is mean magnitude noise. The noise is sampled at 16 kHz for 1 second. Returned values

are ADC units. To convert to a voltage =  $\text{adcunits} \times 2.5/65536.0$ . Receive sensitivity of NM3 is approximately  $-145 \text{ dB re } 1 \mu\text{Pa @ } 1\text{m}$ . Where noise =  $20 \log_{10}(\text{voltage}) + 145.0 \text{ dB}$

## Example

### Local Device

```
> $N  
#NR000121P001068M0000099<CR><LF>
```

## \$P – Ping Modem

### Command

**\$P<xxx>** transmits a ping request to modem <xxx>.

### Response

**\$P<xxx><CR><LF>** is returned immediately or **E<CR><LF>** on error. If the remote device successfully receives the request it will transmit an ack. The local device will then return **#R<xxx>T<ttttt><CR><LF>** or **#TO<CR><LF>** on timeout. Where the propagation time <ttttt> is a 16 kHz counter. Range =  $\text{<ttttt>} \times c \times 3.125 \times 10^{-5}$  where c is the sound velocity (assume 1500m/s if no data is available).

## Example

### Local Device

```
> $P100  
$P100<CR><LF>  
#R100T32000<CR><LF>
```

## \$Q – Get Quality Indicator

### Command

**\$Q** gets a quality indicator for the last received data message.

### Response

**\$C<n><CR><LF>** is returned immediately where <n> is the number of RS symbol errors corrected in the last received message, or '-' if the packet failed to decode.

## Example

### Local Device

```
> $Q  
$C2<CR><LF>
```

## \$R – Reset Modem

### Command

**\$R** resets the modem.

## Response

\$R<CR><LF> is returned immediately then the modem resets.

## Example

### Local Device

```
> $R  
$R<CR><LF>
```

## \$S – Spectrum Measurement

### Command

\$S starts a local spectrum measurement process, sampling short blocks and averaging over 2 second.

### Response

\$S<CR><LF> is returned immediately then after a second #S<xxxxx><vvvv...vvvv><CR><LF> is returned where <xxxxx> is the number of positive FFT magnitude bins returned from DC up to and including Nyquist. Each magnitude is binary uint16 in two consecutive bytes. Block lengths of 512 samples at a rate of 160 kHz are captured and FFT is applied with bin magnitudes stored. This process is repeated with a total of 64 blocks over the elapsed duration of 2 seconds.

## Example

### Local Device

```
> $S  
$S<CR><LF>  
#S00257<vvvv...vvvv><CR><LF>
```

## \$T – Test Message

### Command

\$T<xxx> sends a test message request to modem <xxx>.

### Response

\$T<xxx><CR><LF> is returned immediately or E<CR><LF> on error. When the remote device received the request, it will broadcast a test data packet.

## Example

### Local Device

```
> $T100  
$T100<CR><LF>  
#B10064Hello! This is a Nanomodem v3 DSSS test  
transmission at 640 bps.<CR><LF>
```

## \$U – Unicast Data

### Command

`$U<xxx><yy><data>[T<z*14>]` transmits a message <data> of <yy> bytes to modem <xxx>. Data can be any printable or non-printable byte value. <yy> is the number of transmitted bytes as two ASCII decimal digits (02-64). If the System Timer is enabled, then an optional transmit time can be appended.

### Response

`$U<xxx><yy><CR><LF>` is returned immediately or `E<CR><LF>` on error.

### Remote Device

`#U<yy><data>[Q<zz>D<+ddd>][T<t*14>]<CR><LF>` is produced at the serial output on the receiving modem. Where <yy> is the number of bytes <data> received. If the Link Quality Indicator is enabled then additional statistics follow with quality, Q, as a number <zz> from 00-99. Doppler tracking is based on the average sample per symbol change over the whole packet. Fractional-sample per sample,  $s = \pm\text{ddd}/1000\%$ . To convert to a relative velocity  $v = c \times \text{ddd}/100\,000$  where c is the sound velocity (assume 1500m/s if no data is available). Positive values indicate the units are moving apart, i.e. increasing range. If the system timer module is enabled, the optional timestamp will also follow, T, where <t\*14> is the absolute system time count at 1MHz. The timestamp is taken at the point the frame synch is detected.

### Example

#### Local Device

```
> $U10005Hello
$U10005
```

#### Remote Device Default

```
#U05Hello<CR><LF>
```

#### Remote Device with LQI Enabled

```
#U05HelloQ56D-001<CR><LF>
```

## \$V – Battery Voltage and Noise Measurement

### Command

`$V<xxx>` sends a voltage and noise measurement request to modem <xxx>.

### Response

`$V<xxx><CR><LF>` is returned immediately or `E<CR><LF>` on error. When the remote device receives the request it will take a battery voltage measurement, and a noise measurement over 1 second. The data will then be transmitted as a broadcast data packet. When the local device receives this as `#B<xxx><yy>V<vvvvv>R<rrrrrr>P<pppppp>M<mmmmmm>[Q<qq>D<+ddd>]<CR><LF>` where <yy> is the length of the data bytes, <vvvvv> is the battery voltage where volts = <vvvvv> \* 15/65536, <rrrrrr> is the RMS noise in adc units, <pppppp> is the peak-to-peak noise in adc units, <mmmmmm> is the mean magnitude in adc units. See Section 1.3.8 for information on converting adc units to noise dB units.

## Example

### Local Device Default

```
> $V100
$V100<CR><LF>
#B10027V21997R000124P001988M000099<CR><LF>
```

### Local Device with LQI Enabled

```
> $V100
$V100<CR><LF>
#B10027V21997R000124P001988M000099Q59D+001<CR><LF>
```

## \$W – Wake

### Command

**\$W** tells the modem to wake up and activate the acoustic receiver processes. This command may need to be sent repeatedly as the serial port will take a few milliseconds to start up before it can receive serial bytes correctly.

### Response

**\$W** is returned to confirm that the modem is awake.

## Example

### Local Device

```
> $W
> $W
$W<CR><NL>
```

## \$XF – Flag RxS During Transmit

### Command

**\$XF<x>** where <x> is the action. **\$XFE** = Enable RxS high during transmit. **\$XFD** = Disable RxS high during transmit. The RxS Flag by default is only high when the receiver detects a synch chirp. Some applications may need to inform associated electronics when the modem is also transmitting.

### Response

**#XF<e><CR><LF>** where <e> is E (enabled) or D (disabled).

## Example

### Local Device

```
> $XFE
#XFE<CR><LF>
> $XFD
#XFD<CR><LF>
```

## \$XQ – Link Quality Indicator

### Command

**\$XQ<x>** where <x> is the action. **\$XQE** = Enable LQI. **\$XQD** = Disable LQI. When the link quality indicator is enabled, incoming data packets have the statistics appended. The link quality indicator is disabled by default on modem power on. If the Link Quality Indicator is enabled then additional statistics follow as [Q<zz>D<+ddd>] with quality, Q, as a number <zz> from 00-99. Doppler tracking is based on the average sample per symbol change over the whole packet. Fractional-sample per sample,  $s = \pm\langle\text{ddd}\rangle/1000\%$ . To convert to a relative velocity  $v = c \times \langle\text{ddd}\rangle/100\,000$  where c is the sound velocity (assume 1500m/s if no data is available). Positive values indicate the units are moving apart, i.e. increasing range.

### Response

**#XQ<e><CR><LF>** where <e> is E (enabled) or D (disabled).

### Example

#### Local Device

```
> $XQE
#XQE<CR><LF>
> $XQD
#XQD<CR><LF>
```

## \$XT – System Time

### Command

**\$XT<x>** where <x> is the action. **\$XTE** = Enable system timer. **\$XTD** = Disable system timer. **\$XTC** = Clear system timer count. **\$XTG** = Get system timer count. When the system timer is enabled, incoming data packets have the timestamp appended using this system timer. The system timer is disabled by default on modem power on.

### Response

**#XT<e><z\*14><CR><LF>** where <e> is E (enabled) or D (disabled) and <z\*14> is the current ascii decimal counter value of the 1MHz system timer.

### Example

#### Local Device

```
> $XTG
#XTD0000000000000000<CR><LF>
> $XTE
#XTE0000000000000000<CR><LF>
> $XTG
#XTE000000004127977<CR><LF>
> $XTG
#XTE000000006527930<CR><LF>
> $XTC
#XTE0000000000000000<CR><LF>
> $XTG
#XTE000000002415802<CR><LF>
> $XTD
#XTD0000000000000000<CR><LF>
> $XTG
#XTD0000000000000000<CR><LF>
```



## \$Z – Sleep

### Command

**\$Z** disables the acoustic receiver and puts the modem into a low power sleep mode. The modem stays in this state until it receives a user command **\$W** to wake it up again.

If the system timer is enabled, the counter will pause while the unit is asleep.

### Response

**\$Z<CR><NL>** is returned immediately to confirm the modem is entering sleep state.

### Example

#### Local Device

```
> $Z
$Z<CR><NL>
```

## 7. BootLoader

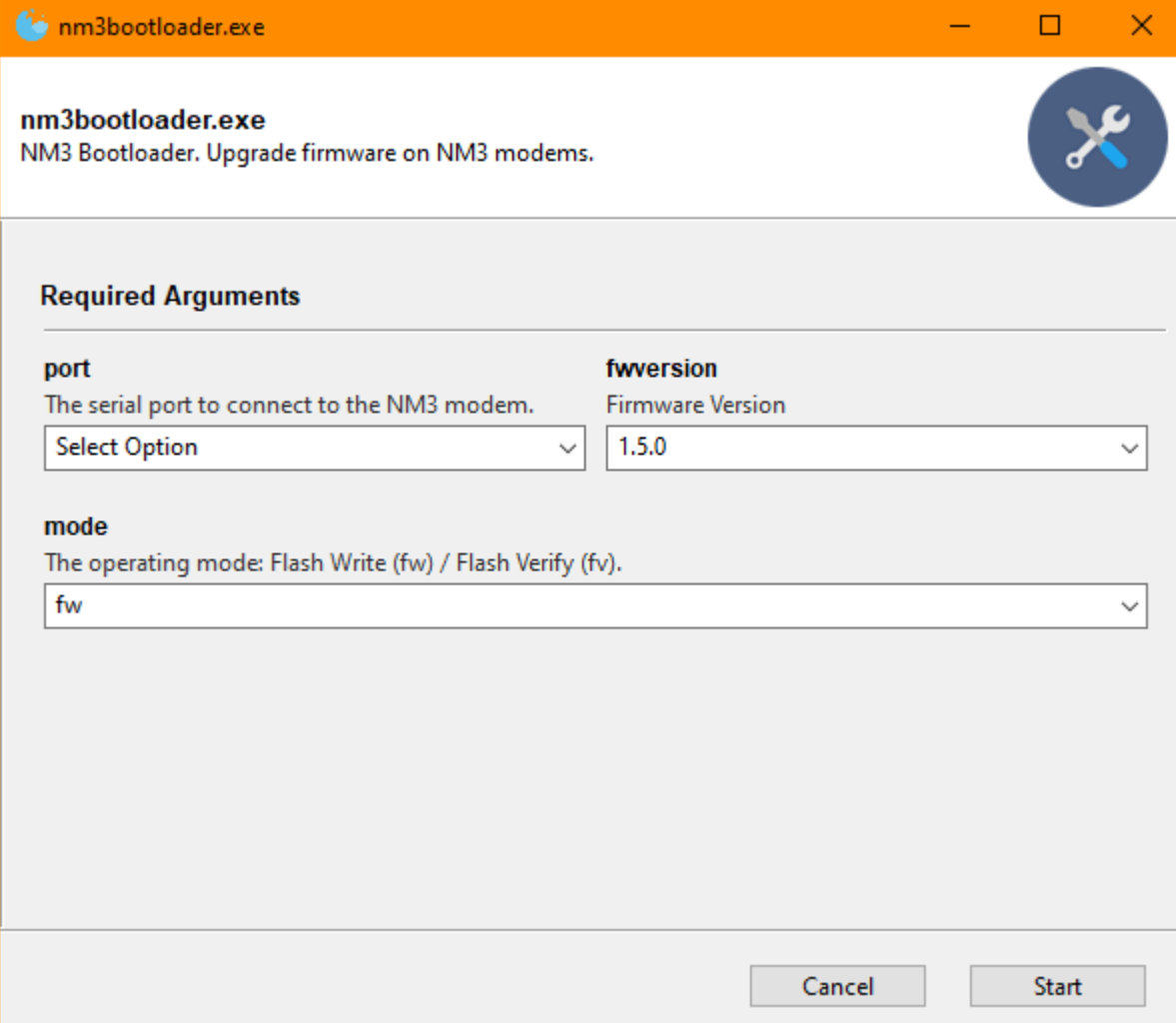
The Delphis hosts a bootloader allowing the user to update the application over serial port. For the first 6 seconds after power up the modem awaits any bootloader communication prior to launching the main application.

## 8. Flashing the Application Firmware

Following the screenshots below

RS232 Serial Connection is Required for this. With the device powered off to start with.

Software: nm3bootloader.exe



**nm3bootloader.exe**  
NM3 Bootloader. Upgrade firmware on NM3 modems.

**Required Arguments**

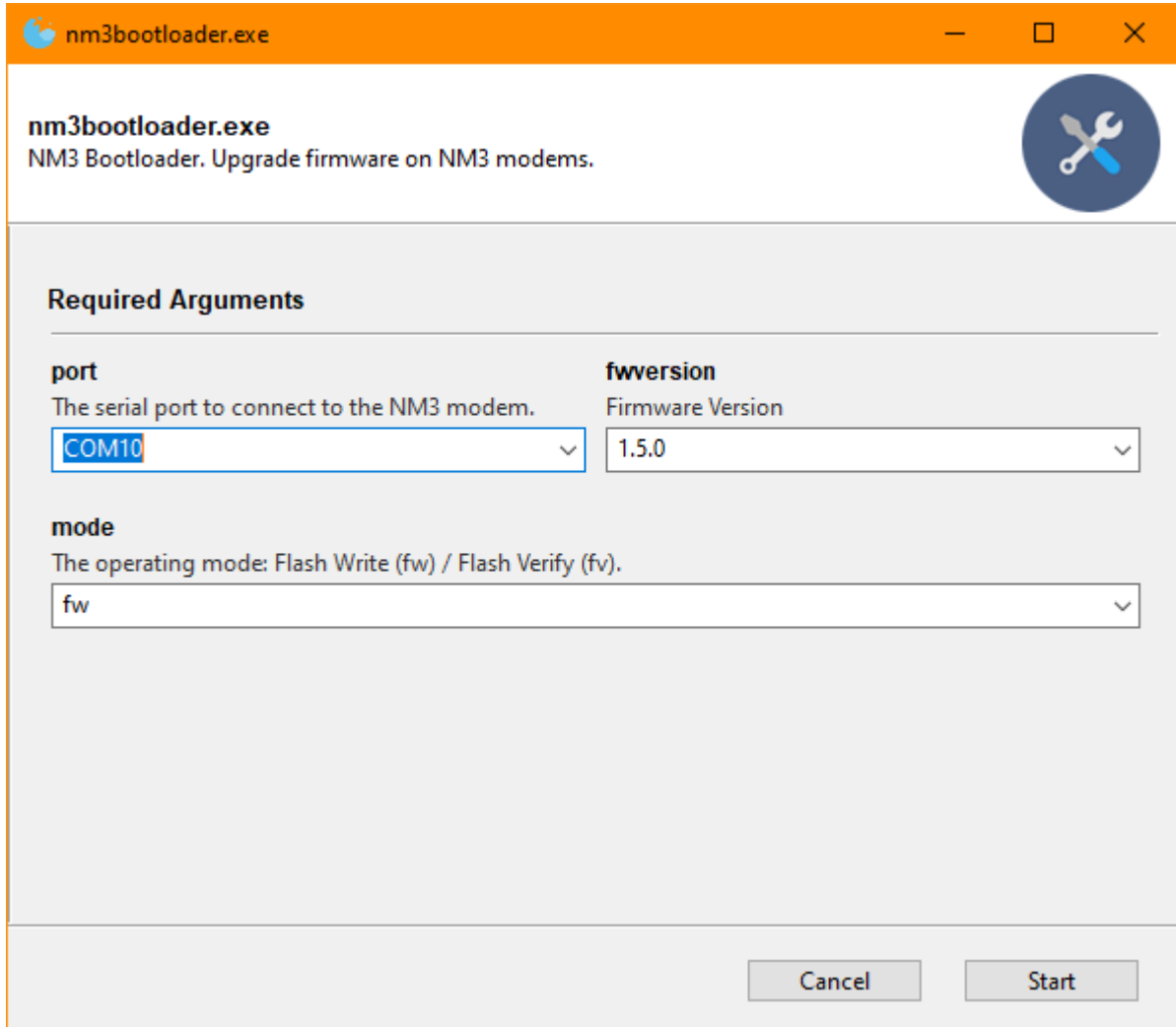
<b>port</b> The serial port to connect to the NM3 modem.	<b>fwversion</b> Firmware Version
Select Option	1.5.0

**mode**  
The operating mode: Flash Write (fw) / Flash Verify (fv).

fw

Cancel Start

Select the COM port that is connected to the Modem. Select the firmware version (the default is the latest). Select the mode: fw = Firmware Write. Power up the Modem then press Start.



**nm3bootloader.exe**  
NM3 Bootloader. Upgrade firmware on NM3 modems.

**Required Arguments**

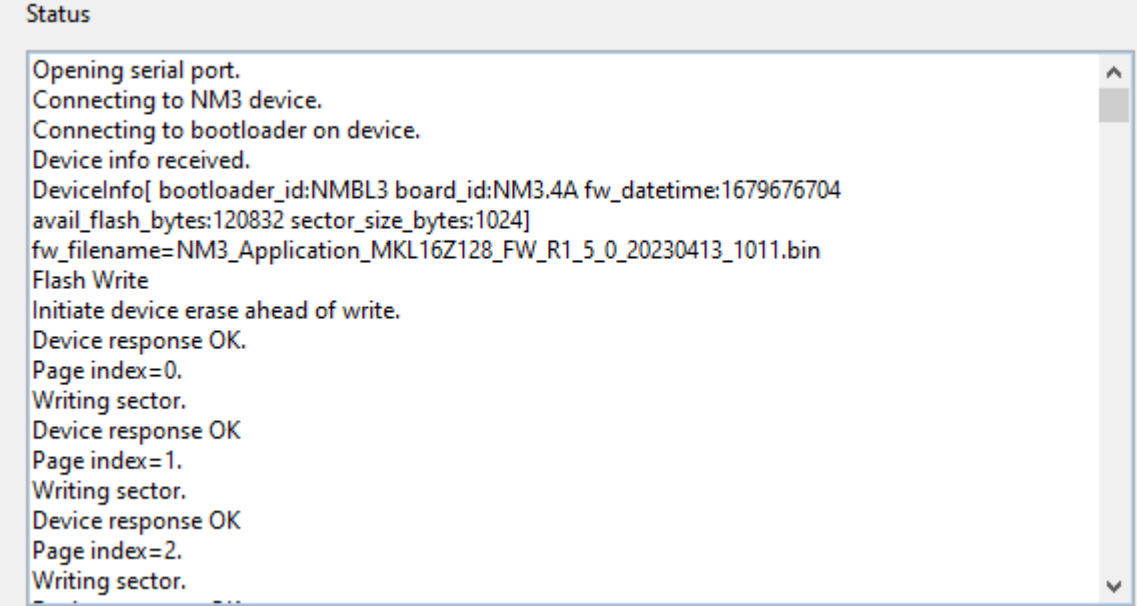
<b>port</b> The serial port to connect to the NM3 modem.	<b>fwversion</b> Firmware Version
COM10	1.5.0

**mode**  
The operating mode: Flash Write (fw) / Flash Verify (fv).

fw

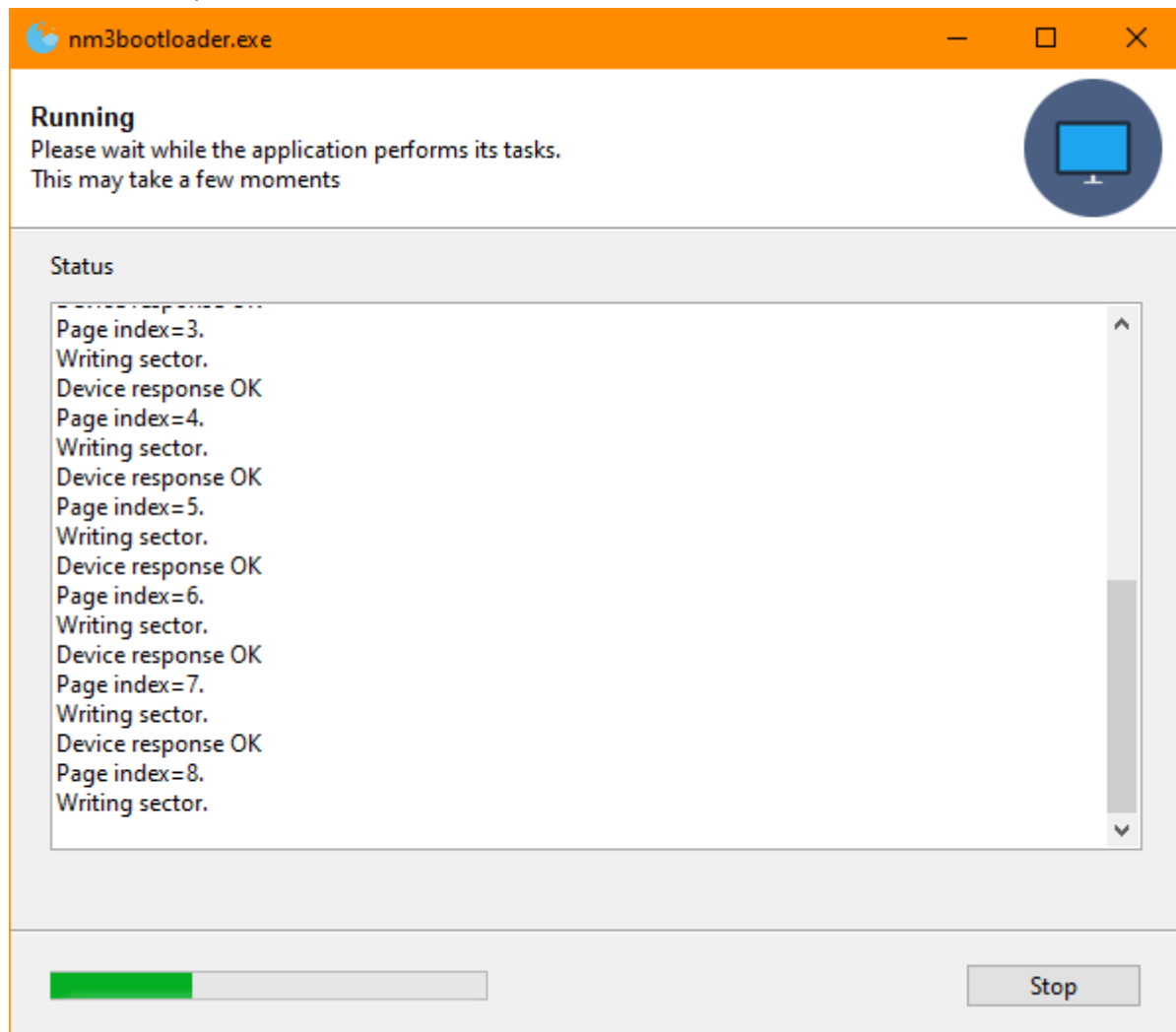
Cancel Start

The Status window will then update to show progress. It first tries to connect to the bootloader to determine which version of the Modem is connected. It then finds the correct binary for the Modem version to send to the device.

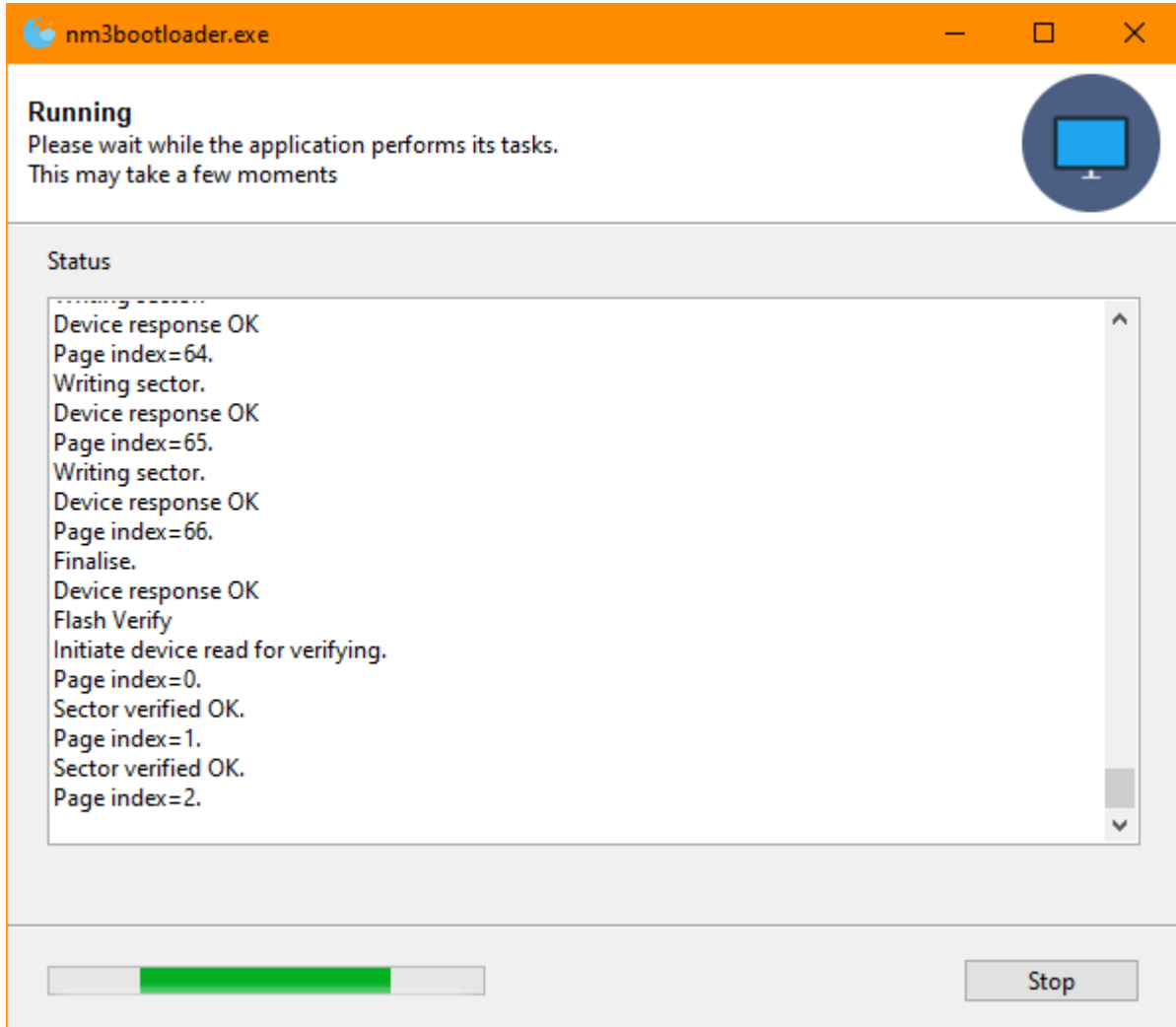


```
Status
Opening serial port.
Connecting to NM3 device.
Connecting to bootloader on device.
Device info received.
DeviceInfo[ bootloader_id:NMBL3 board_id:NM3.4A fw_datetime:1679676704
avail_flash_bytes:120832 sector_size_bytes:1024]
fw_filename=NM3_Application_MKL16Z128_FW_R1_5_0_20230413_1011.bin
Flash Write
Initiate device erase ahead of write.
Device response OK.
Page index=0.
Writing sector.
Device response OK
Page index=1.
Writing sector.
Device response OK
Page index=2.
Writing sector.
```

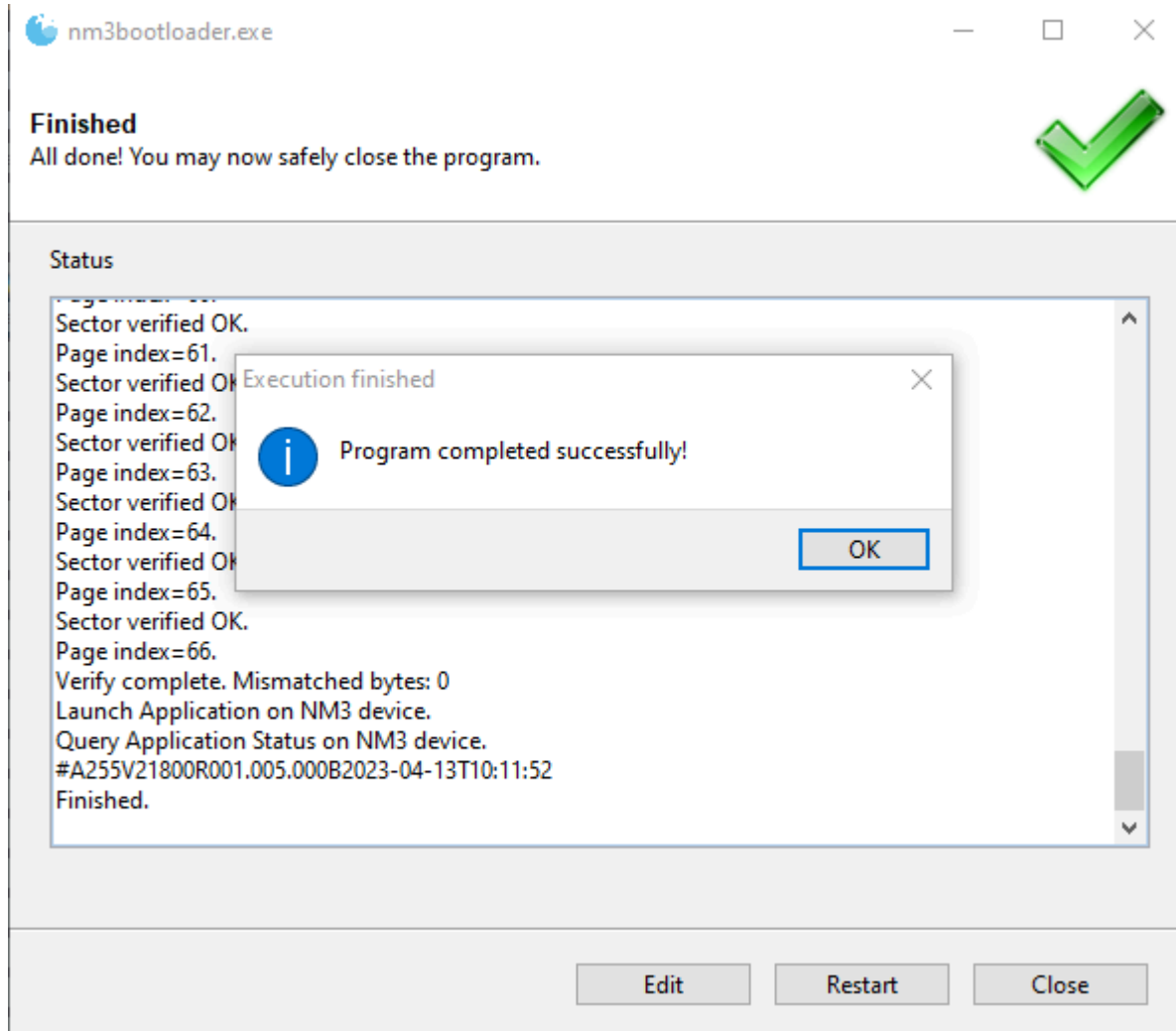
The Flash Write process will take about 1 minute.



After writing the binary it will then read back the flash memory to verify that it has written correctly. This will take about 2 minutes.



When it finishes verifying the flash it will launch the application on the PCB. If there is a transducer connected you will hear it beep. It will then query the modem status to get the build number. This should match the version you selected.



The modem now has the application flashed to it.

## 9. Resources

### Python Driver and tools

To accelerate development when using the Delphis acoustic modem, a number of python-based resources are available at :

<https://github.com/bensherlock/nm3-python-driver> The nm3driver.py wraps the UART modem commands with Python functions for commonly used functionality. Examples covering many of the modem's features can be found in nm3example.py.

A simulator framework is also available. This provides a real-time simulated acoustic environment and modems for connecting to either by human terminal interface, or a serial port with external hardware, or via Python programmed network nodes. Designed for all stages of prototyping prior to sea trials. Especially for checking embedded hardware implementations connected to modems by providing realistic propagation delays. <https://github.com/bensherlock/nm3-python-simulator>

## 10, FAQ/ Do's and don'ts