# ED1608 Application note #003

## Application note for normally closed contact Alarm

### ED1608 NC-Alarm tracker

The NC Alarm tracker is an extended version of a standard ED1608 tracker.
For all functions of the tracker, please see "1M2M Technical product description general sensor" on our website https://www.1m2m.eu/manuals .

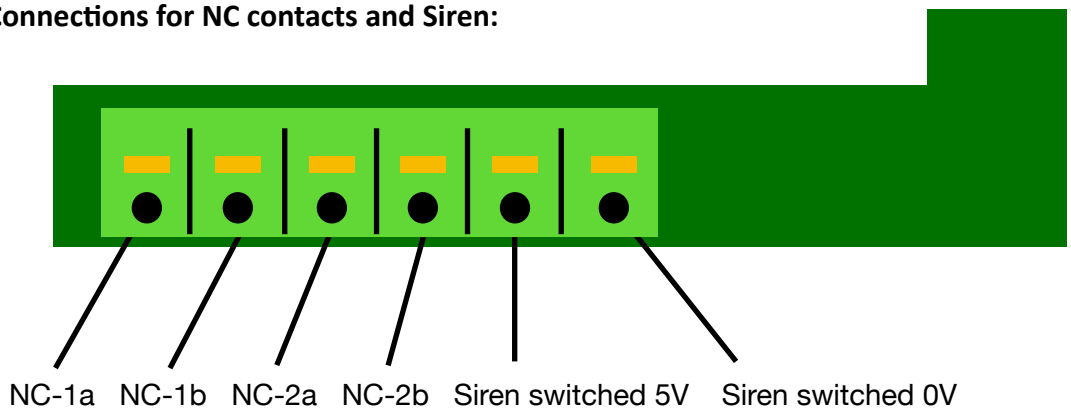
**Extension board**

The NC-Alarm tracker contains a standard ED1608 pcb with a small connector extension board placed in the 1M2M click-enclosure.

**ED1608 NC-Alarm hardware features**

- 1 NC contact (NC-1) circuit breaker door (reed)
- 1 NC contact (NC-2) circuit breaker cover
- 1 siren 5V/100mA (external 110 dB max.)
- Standard 3.7 6000mAh battery operated
- ESD/Surge protected for > 30 meters cables, according to IEC-610000-4-2 (level 4) and IEC-610000-4-5
- Easy fitting of external wires
- 1M2M Click enclosure with 3 cable glands for siren, cover and door contacts

**Connections for NC contacts and Siren:**




**Alarm functionality**

There are two NC (Normally Closed) inputs for alarm purposes. To save battery power the detection current uses a duty cycle of 0.1%, sampled every 1 second.
When an input state becomes High or Low and differs from the previous state 5 possible actions can be started.

1) Increment a 11-bit input counter. Bit 0 represents the high or low state of the input.
2) Uplink an alarm message with last known GPS location.
3) Sound the alarm for x seconds (0..250)
4) Stop the alarm
5) Get a GPS location (for max 240 seconds) and Uplink an alarm message with latest GPS location

If action 2 and 3 are started at the same time, action 3 waits for action 2 (siren is activated after first alarm message is sent)
When all config bits are set to 0 the input state is set to Disabled and the input is not sampled. In all other modes the input is sampled every second.

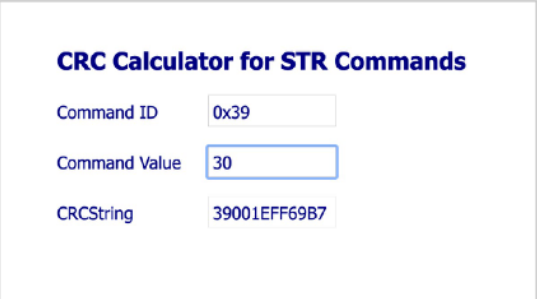
NCAlarmConfigX is one byte (8 bits) per input
Bit0 to Bit3 select the actions when the input state becomes high
Bit4 to Bit7 select the actions when the input state becomes low

| | | |
|---|---|---|
| B0 | When input state becomes High: | Uplink alarm and last known GPS |
| B1 | When input state becomes High: | Sound the alarm for <SirenDuration> seconds. |
| B2 | When input state becomes High: | Kill the alarm. |
| B3 | When input state becomes High: | Enable GPS and send new position when available. |
| B4 | When input state becomes Low: | Uplink alarm and last known GPS. |
| B5 | When input state becomes Low: | Sound the alarm for <SirenDuration> seconds. |
| B6 | When input state becomes Low: | Kill the alarm. |
| B7 | When input state becomes Low: | Enable GPS and send new position when available. |

**NC Alarm configuration parameters (downlink)**

| | |
|---|---|
| 0x37: NCAlarmConfig1 | 8 bits that define actions for input state changes in NC_Input1 |
| 0x38: NCAlarmConfig2 | 8 bits that define actions for input state changes in NC_Input2 |
| 0x39: SirenDuration | time in seconds the siren will be active after a trigger [0..250] |

**Example 1:**
Siren Duration command : 0x39
Time in seconds : 30 seconds
*->Downlink: 39001EFF69B7*




**Example 2:**
NCAlarmConfig1 command : 0x37
bits 0,1,4,6 (01001011) : 0x4B
- Input high: Uplink alarm, Siren ON, get fix and uplink GPS location
- input low: Siren off

*->Downlink: 37004BFF69EB*

See https://www.1m2m.eu/webtools.php for CRC calculator webtool.

# ED1608 Application note #003

## Application note for normally closed contact Alarm

**NC Alarm uplink messages**

Information to be uploaded in case of an alarm:

- NCAlarmCounter1 (11 bit)
- NCAlarmCounter2 (11 bit)
- GPS position, satellites in fix and fix age (8 bytes)
- Source of the alarm (2 bit)

Counters are packed in 2 times 11 bit = 1024 times open and 1024 times closed.
If a counter is incremented and no message has been uploaded for 512 counts, the alarm message is sent with NC_AlarmCause1 and NC_AlarmCause2 set to 0.
This allows for server-side extending of the counters if required.

```
#define MsgIDNC_Alarm (0x99) // used in NC Alarm messages

typedef struct {              // Size must be 12 byte for Sigfox
  byte MsgId;                 // Message Identification Value = MsgIDNC_Alarm
  byte NC_AlarmCntLo1;        // bit 0..7 of NCAlarmCounter 1
  byte NC_AlarmCntLo2;        // bit 0..7 of NCAlarmCounter 2
  bits NC_AlarmCntHi1  :3;    // bit 0..2  = bit 8 to 10 of NCAlarmCounter 1
  bits NC_AlarmCause1  :1;    // bit 3     = NC-input 1 triggered
  bits NC_AlarmCntHi2  :3;    // bit 4..6  = bit 8 to 10 of NCAlarmCounter 2
  bits NC_AlarmCause2  :1;    // bit 7     = NC-input 2 triggered
  byte GPSFixAge;             // bit 0..7  = Age of last GPS Fix in Minutes
  byte SatCnt_HiLL;           // bit 0..4  = SatInFix, bit5 Latitude 25
                              //             bit 6,7 = Long 25,26
  byte Lat[3];                // bit 0..23 = latitude  bit 0..23
  byte Lon[3];                // bit 0..23 = longitude bit 0..23
} TNC_AlarmMsg;
```

Example uplink:
Payload: 9901010000054F812A07A687

The GPS information is compressed in exactly the same way as in the alive and moving messages


The payloads can be expanded via the 1M2M Payload decoding JSON service
https://1m2m.eu/services/GETPAYLOAD?Human=0&PL=YourPayload

There are web tools for coding and decoding available on https://www.1m2m.eu/webtools.php .