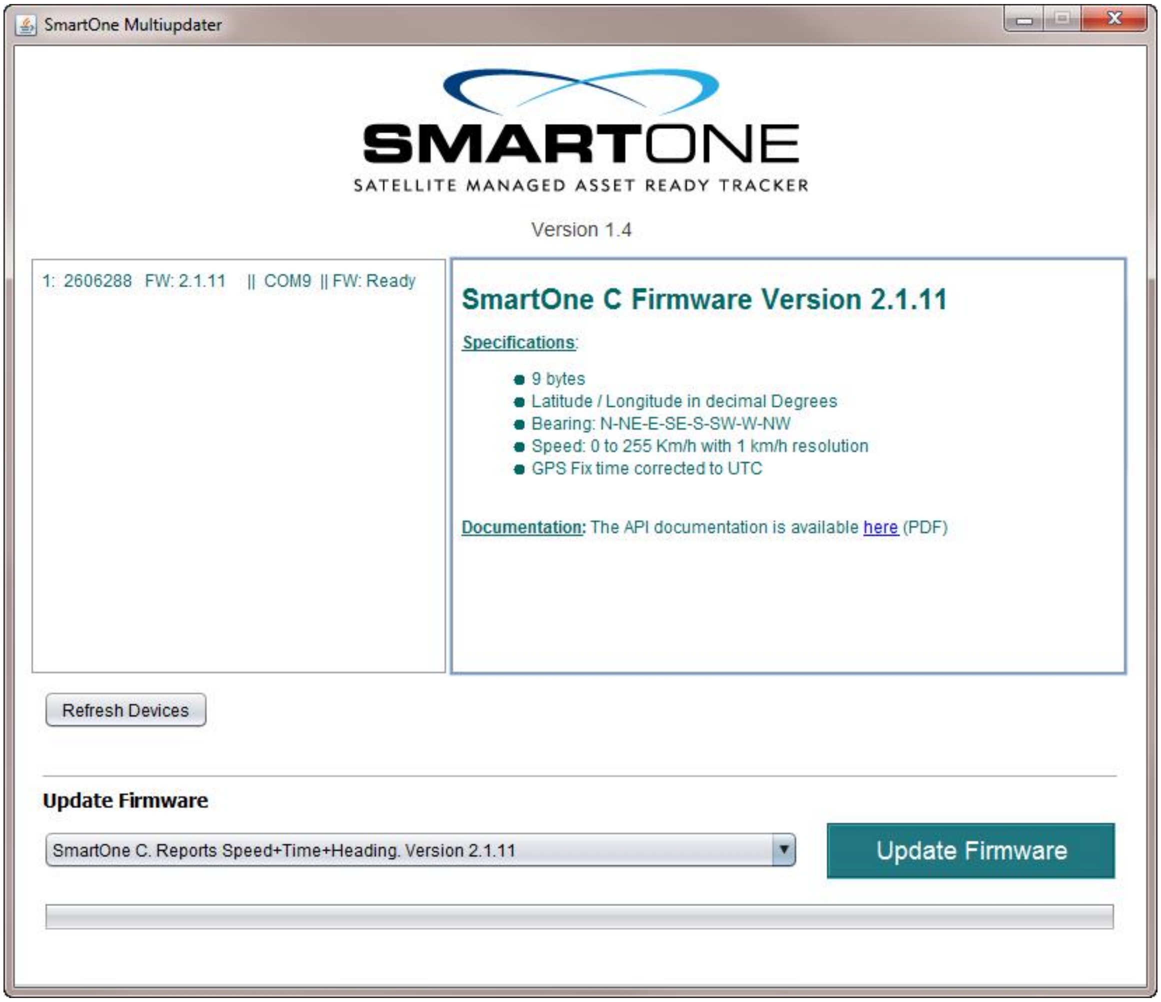
SmartOne C
FW - Version 2.1x

Short Technical Reference

AtlasTrax™ Corporation
555 Legget Drive, Suite 304, Tower A
Ottawa, ON K2K 2X3
Canada

Modified Multiple Updater

This version of the SmartOne C Firmware Updater gives access to the the most recent SmartOne C firmware versions available from the AtlasTrax servers. New versions will be automatically listed and available to users via the drop-down selector. A splash screen summarizes the main features of the selected version.



Multi Updater screen shot

SmartOne C Version 2.1.x

Payload structure

SmartOne C - FW 2.1.x				
9 bytes Payload W/ Battery / Ext.Power / Vibration / Heading / Speed / GPS Time / Digital Inputs				
Byte #	**Variable**	**Bit(s)**	**Description**	**Min / Max**
0	Message Type	1	Bit (0) 0 = This format 1 = Maintenance Message	0 / 1
0	Input 1	1	Bit (1) 0 = Closed 1=Opened	0 / 1
0	Input 2	1	Bit (2) 0 = Closed 1=Opened	0 / 1
0	External Power	1	Bit (3) 0 = Battery 1=Ext.Pwr	0 / 1
0	Vibration	1	Bit (4) 0 = Steady 1=In Vibration	0 / 1
0	Heading	3	Bit (7:5) 360/8 sectors (N / NE / E / SE / S / SW / W / NW)	0 / 7
1 - 3	Latitude	24	Byte 1=MSB Byte 2=SB Byte 3=LSB Latitude in decimal deg.	-90 / +90
4 - 6	Longitude	24	Byte 1=MSB Byte 2=SB Byte 3=LSB Longitude in decimal deg.	-180 / +180
7	Speed over ground	8	Speed in Km/h	0 / 255
8	Time	7	Bit (7:0) Modulo 720 of GPS Time Of Day / 6 (±3 seconds accuracy)	0 / 120
8	Battery	1	Bit (8) 0 = Good 1=Replace	0 / 1

FW Ver 2.1.9 - Payload

AtlasTrax™ Corporation
555 Legget Drive, Suite 304, Tower A
Ottawa, ON K2K 2X3
Canada

SmartOne C Version 2.1.x

Time Encoding / Decoding

Theory

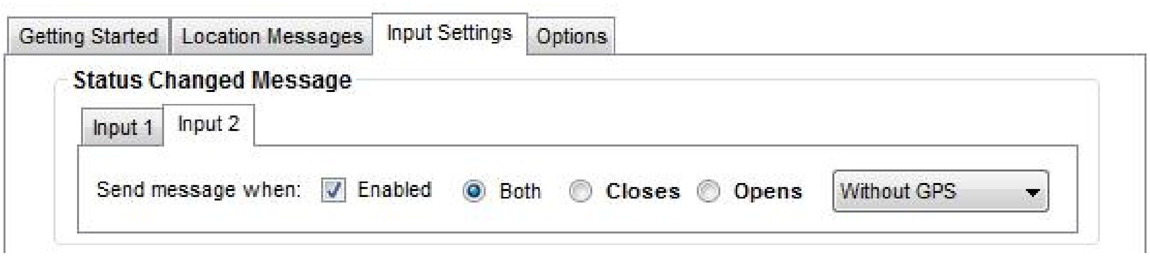
Simplex devices - as their name indicates - transmit messages one way only, from the ground to the satellite. Therefore, these devices has no means to ensure their messages were received. In order to increase the reliability of the system, Globalstar Simplex devices send up to 3 times the same message ~5 minutes apart so, out of three transmissions, one at least should reach a satellite. If the two first messages did not reach a satellite, the message can well arrive up to 10 minutes after the actual GPS acquisition. When a message hits a satellite, it is sent to a ground station gateway that adds its own time stamp and then finally pushes the message to the VAR systems.

The time encoding / decoding algorithm used in the FW described in this document is based on the assumption that the difference between the GPS time and ground station time stamp will not exceeds 720 seconds (12 minutes) so that the GPS time value can be expressed as the modulo 720 of the Time Of Day (TOD) value in seconds.

The day is divided into 120 equal chunks of 720 seconds. When the GPS acquire a fix, the algorithm calculates a what second of the current chink the event took place and sends this value only. On the decoding side, knowing the current chunk from the Unix time stamp, another code reconstitutes the time of acquisition. Actually, the value sent by the device is divided by 6 in order to fit on 7 bits.

Accurate Event Time

In some cases knowing the exact time an Input Status Change triggered a transmission is critical whereas the GPS location can be ignored.
In such cases it is possible to select the [Without GPS] option from the Configuration Software.

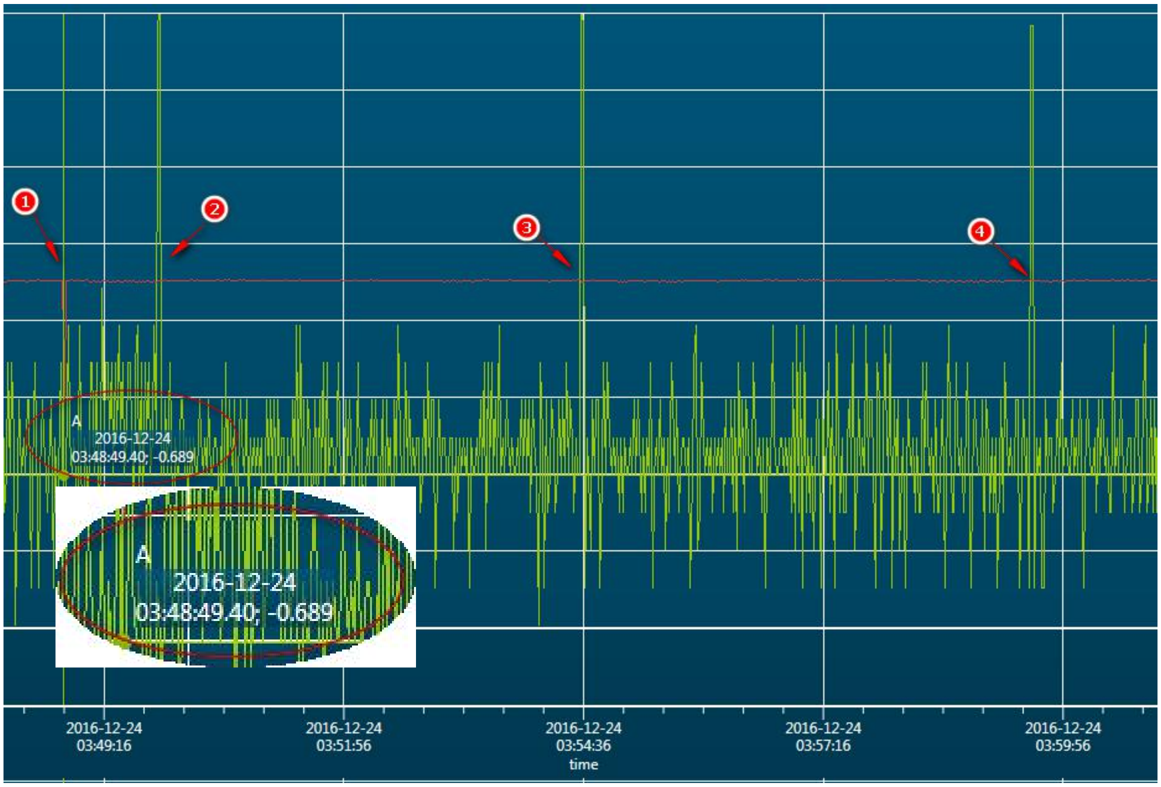


The firmware detects this setting for activating the GPS in order to read the exact time but not the location, the speed nor the the heading. The time to fix is subtracted from the GPS time thus yielding the exact event time.

Regardless the duration of the time to fix, the firmware limits the last retry time to a maximum of 715 seconds from the triggering event.

AtlasTrax™ Corporation
555 Legget Drive, Suite 304, Tower A
Ottawa, ON K2K 2X3
Canada

The picture below is a capture of a typical Input Change triggering a transmission.



Typical Input Change W/O GPS

1) Input 1 change
2) First transmission
3) Retry 1
4) Retry 2

event_time	modem_time	carrier_time
2016-12-24 03:48:49.0000000	2016-12-24 03:54:44.0000000	2016-12-24 03:54:27.0000000

Time decoded

Time encoding:

The GPS time encoding consists of using the modulo 720 of the GPS time in seconds divided by 6. Thus, the theoretical value can be comprised between 0 and 120 x 6 = 720 seconds for a resolution of ±3 seconds.

$$GPS_{MOD} = GPS_{time} \text{ (in seconds) mod } 720 / 6$$

Time decoding

Once received at the VAR systems, the value gets multiplied by 6 and added to the current 720 seconds chunk of the Gateway timestamp (**GW_{time}**).

Notice that GPS time differs form the UTC time by a value known as GPS leap second (**LS**). So, in order to obtain an accurate UTC time of acquisition, **LS** must be added to the GPS time. [**UTC = GPS + LS**]

$$\text{GPS time (UTC)} = GPS_{MOD} \cdot 6 + (GW_{time} \div 720) \cdot 720 + LS$$

Examples of time encoding / decoding:

GPS_{time} = 8:34:27	[GPS time, actually ahead of UTC time by 17 seconds]
GW_{time} = 8:34:12	[Gateway time of reception in UTC time (2 seconds after TX]
LS = - 17seconds	[Pseudo constant GPS leap second value]

Encoding:

Convert GPS_{time} (8:34:12) in seconds :	8 x 3600 + 34 · 60 + 27 = 30867	
GPS_{MOD} :	30867 mod 720 = 627/ 6 = **104**	

Decoding:

Convert GW_{time} (8:34:12) in seconds :	8 x 3600 + 34 · 60 + 12 = 30852	**[a]**
Truncate GW_{time} :	(30852 / 720) x 720 = 30240	
Add **GPS_{MOD} :**	30240+ **104** x 6 = 30864	
Add **LS :**	30864 + (-17) = 30847	**[b]**
Convert **[b]** to HH:MM:SS :	08:34:07 (Actual UTC TX time- 3 seconds)	

As long as both the GPS+LS and the GW times are in the same chunk, GPS time is always older than the GW time thus GPS value is smaller that GW value. What if the GW time is smaller than the GPS time ? It indicates that the GW time occurred in the next contiguous chunk so we need to add 720 the the calculated time as demonstrated seen in he examples below:

Case [b] > [a]

GPS_{TOD} = 5:55:18	[GPS time, actually ahead of UTC time by 17 seconds]
GW_{TOD} = 6:05:03	[Gateway time of reception in UTC time]
LS = - 17seconds	[Pseudo constant GPS leap second value]

Encoding:

Convert GPS_{TOD} (5:55:18) in seconds :	5 x 3600 + 55 · 60 + 18 = 21318	
$GPS_{TIME\ ENCODED}$:	21318 mod 720= 438 / 6 = **73**	

Decoding:

Convert GW_{time} (6:05:03) in seconds :	6 x 3600 + 5 · 60 + 3 = 21903	**[a]**
Truncate GW_{time} :	(21903 / 720) x 720 = 21600	
Add **GPS_{MOD} :**	21600 + **73** x 6 = 22038	
Add **LS :**	22038 + (-17) = 22021	**[b]**
[b] > [a] then subtract 720 from b	[b] = 22021 - 720 = 21301	
Converted to HH:MM:SS :	05:55:01 (Actual UTC time)	

AtlasTrax™ Corporation
555 Legget Drive, Suite 304, Tower A
Ottawa, ON K2K 2X3
Canada

C# code example:

```
DateTimedecode_time(
    int gps_value,     // The value sent by the device
    long unixtime)     // The gateway timestamp in UNIX format
{
    // Convert the UNIX timestamp into seconds of the day
    DateTime utc_time = new DateTime(1970, 1, 1).AddSeconds(unixtime);
    DateTime utc_date = new DateTime(utc_time.Year, utc_time.Month, utc_time.Day);

    long gw_time = utc_time.Hour * 3600 + utc_time.Minute * 60 + utc_time.Second;
    long gw_mod = gw_time % 720;

    // calculate the GPS time seconds of the day
    long gps_time = (gps_value * 6 + LEAP_SECONDS);

    if ( gps_time > gw_mod )
    {
        gps_time -= 720;
    }

    gps_time += Convert.ToInt64(gw_time / 720) * 720;

    // return the GSPS time Of Day corrected to UTC
    return (utc_date.AddSeconds(gps_time));
}
```

Important note on leap second:

The GPS leap second value is not a constant so decoding applications must eventually adjust it. Since 2015-07-01 this value is -17 seconds. The next adjustment is scheduled on December 2016. The GPS Leap Second will be positive so the cumulative value will be 18 seconds (GPS - UTC=18 seconds).

More readings on this topic

here: http://tycho.usno.navy.mil/leapsec.html
and
here http://maia.usno.navy.mil/

* * *