gps_movement_detector.c

This program works with the TD1205P module from TDNEXT. The aim is to detect if this device is in motion. If there is movement, it will send some information every T1 period. If there is no movement, it will send information every T2 period (longer than T1). You will find more explanation about T1 and T2 periods in the commands part.

Movement is linked to acceleration. The frequency for reading accelerometer data is fixed. Its value is 10 Hz. The scale which is used for detecting accelerometer events is 2G. This value is fixed too. The device can detect movements on all axis (x, y and z). It considers that there is movement if one of the acceleration values for any of the 3 axes exceeds a threshold. This threshold is fixed in the program but can be changed directly in the code (+- 160 mg here). In the program, we consider that there is movement if the device detects more than one acceleration value which is upper than the threshold.

A keepalive message is sent at every boot.

Useful information

- For configuring the environment, use the tutorial: http://www.instructables.com/id/Sigfox-GPS-Tracker/
- This example is only built and tested for TD1205P module.
- In order to see the received messages, use Sigfox backend: https://backend.sigfox.com

General message format

For transceiving information on the Sigfox network, this program uses a 12 bytes format. Message is formed by using some code from the tutorial:

- 4 bytes: GPS longitude or free
- 4 bytes: GPS latitude or free
- 1 byte: voltage
- 1 byte: temperature
- 2 bytes: free

Commands

Users can change some parameters of the program using serial communication. Indeed, it is possible to communicate with the module thanks to a serial console. The baud rate which is used here for serial communication is 9600 bits/s. Commands must be typed during the 3 minutes after the launching of the program. Indeed, serial communication is stopped after 3 minutes in order to save power. In that way, the module doesn't wait for serial connection, so it doesn't consume energy for that. If you don't want to stop serial communication, in order to see some messages after the 3 minutes, comment this line in the delayed_start function:

```
// Stop the use of serial communication
TD_UART_DisableExtended(out_stream);
```

AT$MODE=

- Used for choosing GPS Power mode.

- 0 for *TD_GEOLOC_OFF* mode: with this mode, there is no RAM retention, everything is off, there is no consumption. The module doesn't store any information about the fixing and the satellites.

- 1 for *TD_GEOLOC_HW_BCKP* mode: with this mode, there is RAM retention. The module can store the last information about the fixing and the satellites in the RAM. It means that the fixing time for finding GPS data will be significantly shorter the next time the GPS will wake up in order to get data.

- Default value: *TD_GEOLOC_HW_BCKP.*

AT$INTERVAL=

- This interval T1 is the time between two consecutive messages when the module is in motion.

- If empty, it prints the current value.

- When there is a positive value after this command, this value become the new interval T1 in seconds. Don't forget that Sigfox supports 140 messages a day!

- Default value: 3600 seconds (1 hour).

AT$T2PERIOD=

- This interval T2 is the time between two consecutive messages when the module is not in motion.

- If empty, it prints the current value.

- When there is a positive value after this command, the new interval T2 will be the interval T1 multiplied by this number.

- Default value: 3.

AT$TIMEOUT=

- This data is the duration during which the GPS tries to find satellites in order to have GPS coordinates.

- If empty, it prints the current value.

- When there is a positive value after this command, this number will be the new timeout.

- Default value: 120 seconds.

AT$TYPE=

- Possibility to change the type of message sent by the module when there is no movement.

- If empty, it prints the current value and the type of message.

- 0: the device will only send temperature and voltage. The first eight bytes will be 0x00.

- 1: the device will send the last GPS coordinates, temperature and voltage. In this case, the device will not do a new measure for GPS coordinates: it will send the last GPS latitude and longitude which were measured when there was movement. If the device doesn't make measurement for GPS since the last reset, the first eight bytes will be 0x00.

- Default value: 0.

Consumption

- Sleep mode = approximately 10 uA.
- Timeout = approximately 30 mA.
- During the sending of the message = approximately 50 mA.

Possible ameliorations

I think that it could be good idea to add a command in order to modify the threshold for detecting movement. The objective with this command will be the detection of different types of movements. For that, it is possible to change the scale value of this function (*TD_ACCELERO_2G*), and a value linked to the threshold (10 here).

```
// Monitor accelerometer events (movements)
TD_ACCELERO_MonitorEvent(true,  // Monitoring enabled
    TD_ACCELERO_10HZ,           // Sampling rate 10 Hz
    TD_ACCELERO_ALL_AXIS,       // Axis mask: all axis
    TD_ACCELERO_2G,             // Scale 2 g
    TD_ACCELERO_ALL_HIGH_IRQ,   // Only monitor high IRQs, as low IRQs are always set in
                                // accelerometer registers
    10,                         // Threshold in mg = 10 * 2 g / 127 = +- 160 mg (with scale 2 g)
    1,                          // Duration in ms = 1 * (1 / 10 Hz) = 100 ms (with rate 10 Hz)
    1,                          // High-pass filter enabled
    EventCallback);
```

Another important change could about the interval. If the user decides to change the value of the interval, the device could try to find satellites directly after this command, and not wait for the end of the 3 minutes at the beginning.

I believe that it could be great to add other commands for configuring the device. For example, it could be good to have the possibility to choose if we want to detect movement on only one axis, or for two axes, or for the three axes. Another possible command could be about the possibility to change the sampling rate for reading accelerometer data.

A future work to do is to think about the calibration. For the voltage, the values are not really precise. For the temperature, the precision is equal to one degree, but I think that it is possible to do a better conversion in order to gain accuracy.

Moreover, I think that it could be great to find interesting other information to send. Indeed, only ten bytes are sent in this application, but Sigfox messages can contain twelve bytes of data.