Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	**Date**	2014/10/01	**Ver.**	1.0

Global GPS NMEA over I^{2}C Software Guide V 1.0

Stanly Lin

Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	**Date**	2014/10/01	**Ver.**	1.0

Version History

History			
Date	**Rev.**	**Author**	**Description**
2014/10/01	1.0	Stanly Lin	First Release
2015/6/10	1.0	Archie Lin	Preface added: announcement of I²C

Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	**Date**	2014/10/01	**Ver.**	1.0

0. Preface

Announcement: I²C is now available for the GNSS modules* produced by GlobalTop Technology (programming on software; no hardware modification is needed).

**Modules available for I²C :*

Ivory 3 (GMM-U2P)

Ivory4 (FGPMMOSL3C)

Firefly1(GMM-G3)

Firefly1b(GMM-G3(B))

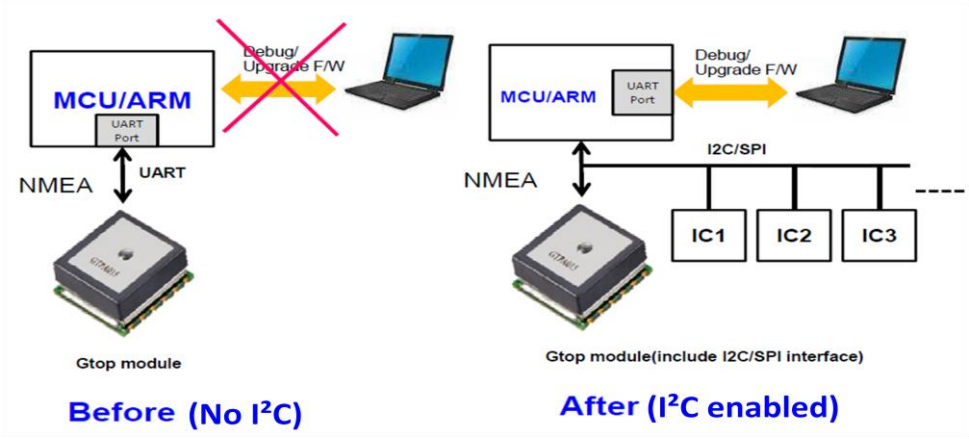
Titan2(GMS-G6)

Titan3(GMS-G9)

When I²C is not enabled, communication between MCU/ARM and chip is in single channel, meaning that the data(NMEA) are transmitted via the UART port, and that makes the UART port no room to perform other tasks such as “Debug” or “Firmware Upgrade” since the port is occupied.

In contrast, when I²C is enabled, data(NMEA) are transmitted via 2 wires; such method allows UART port to perform tasks such as “Debug” or “Firmware Upgrade” since UART port is left open to connect to other devices. In addition, the wires can string up modules as many as possible for communication between modules (please refer to Figure 1).

Figure1(without & with I²C):



Flexibility of I²C is the key that makes I²C powerful to various applications.

Here are some significant features and characteristics in I²C:

- It requires only two bus wires
- No strict baud rate requirements
- Straightforward master/slave relationship between all components (GlobalTop modules are configured as slave)
- Each device connected to the bus is assigned by a unique address (software addressable)
- I²C is an absolute multi-master bus providing arbitration and collision detection

However, please note that RTCM will be disabled once I²C is enabled. Please contact us at sales@gtop-tech.com for any question or request regarding to I²C.

Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	Date	2014/10/01	Ver.	1.0

1. Description

GlobalTop MT3339 & MT3333 GPS modules support both UART and I²C functions. This software guide illustrates how to setup GlobalTop GPS module with I²C two-wire peripheral devices.

2. Global Top GPS module I²C specification

MT3339 & MT3333 support fast mode (bit rate up to 400kbit/s)

MT3339 & MT3333 support 7-bit address

MT3339 & MT3333 support works in slave mode.

The default slave address for MT3339 & MT3333 is 0x10.

MT3339/MT3333 I²C Pin definition:

GPIO Pin	Description
GIO1	I²C SDA pin
GIO0	I²C SCL pin
GIO8	I²C interrupt pin

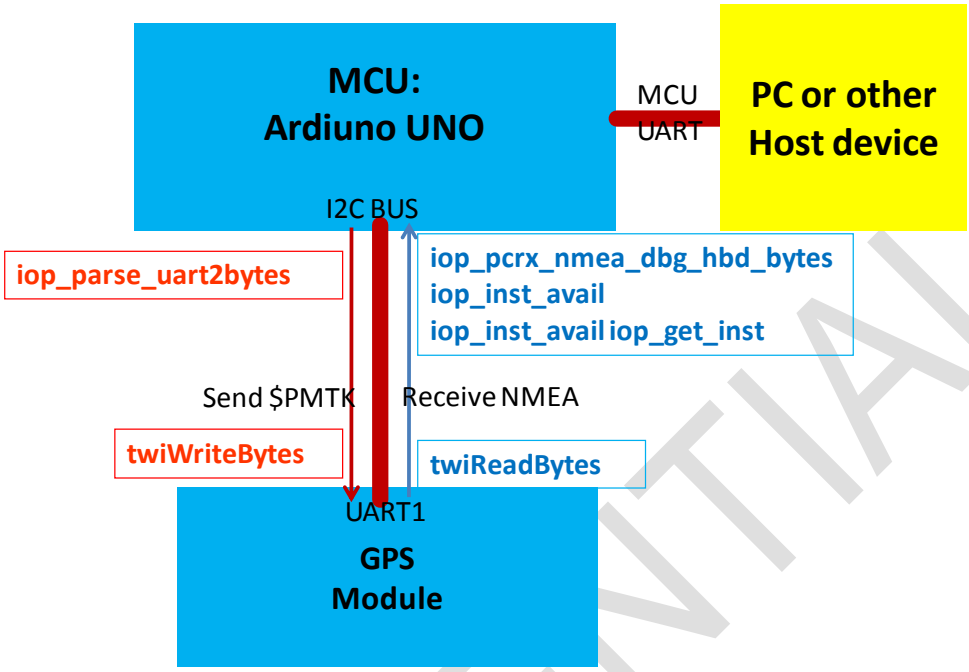
3. The Flow of Host Processor(I²C-master) Receiving NMEA

1. The capacity of buffer for MT3339 & MT3333 I²C TX is 255 bytes, and the master reads one I²C data packet for maximum 255 bytes at a time. To get complete NMEA packet of one second, the master will need to read several I²C data packets and then extract valid NMEA data from these packets.

2. After reading one I²C data packet, the master will then sleep for 2ms before it starts to receive next I²C data packet. The 2ms-sleep is a rule for MT3339/MT3333 to upload new NMEA data into I²C TX buffer. After a complete NMEA packet of one second is read, the master is set to sleep longer to wait next NMEA packet of one second being ready.

In this SW guide, the Ardiuno UNO EV-board is used as the platform(MCU). The hardware connection and software architecture are simply shown in the following chart:

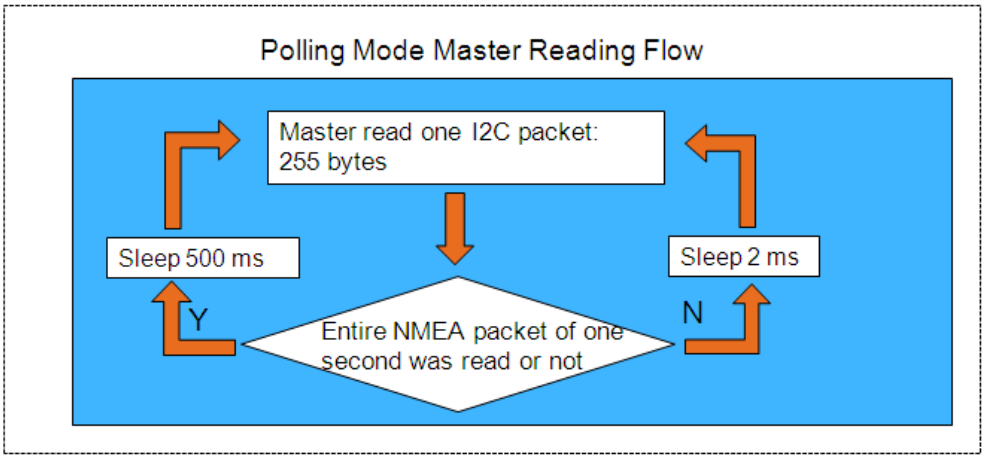
Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	Date	2014/10/01	Ver.	1.0



MediaTek supports two modes for reading NMEA via I²C: **Polling mode** and **Interrupt mode**.

1. **Polling mode**

   In polling mode, master will read entire NMEA packet of one second repeatedly in each polling time interval. The time interval can be configured according to GPS fix interval. Normally time interval must be less than GPS fix interval.

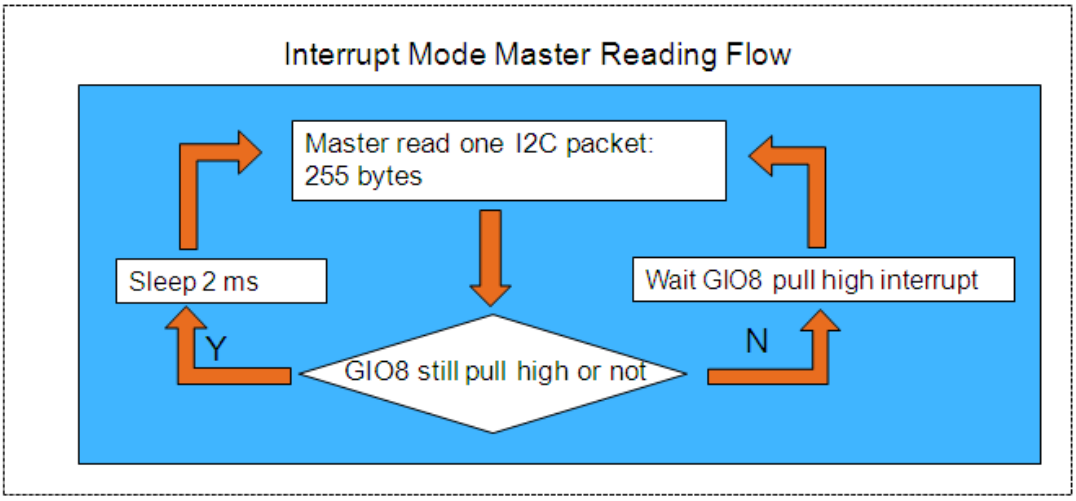


Note: the figure above assumes that GPS fix interval is 1 second. Therefore the polling time interval is set to 500ms.

2. **Interrupt Mode**

This mode is used when the module provides Interrupt Sync. Pin (GIO8). The GIO8 will pull high once NMEA data are ready and are uploading to I²C buffer. After a complete NMEA packet of one second is transmitted, the GIO8 will pull low.

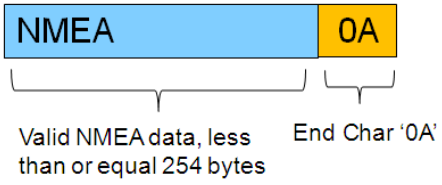
Note: GIO8 can also be used in Polling mode to determine whether the slave stores NMEA or not.



3.1 I²C data packet format in the slave buffer

An I²C data packet has 254 valid NMEA bytes at most and one end character <LF> in the slave buffer. The master can read 255-byte I²C data packet at one time. When the slave I²C TX buffer is empty, the slave will keep providing 255-byte data packet for the master to read; however the content in the packet could be garbage bytes because information in the data is nothing new. The garbage bytes will be explained in the following section.

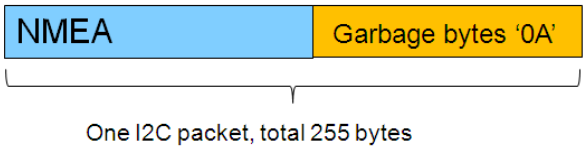
The format of Packet in the slave I²C buffer:



3.2 Three types of I²C packet that master reads from slave

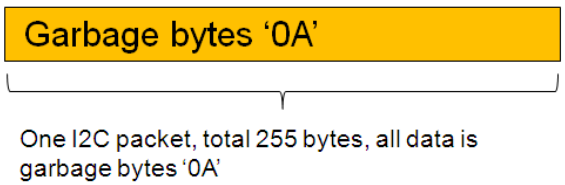
1. NM EA + Garbage bytes--- When the slave (MT3339/MT3333) buffer has already had some data stored, the master will read one I²C packet (255 bytes) from the slave, including some valid data in the header and garbage bytes in the end of packet.

I²C packet format:



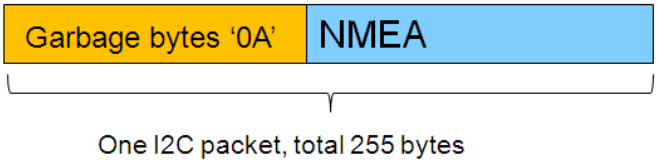
2. Garbage bytes--- When the slave buffer is empty, the slave will keep providing I²C packet (255 bytes)data for the master to read from slave. Since there is no new or useful data contained in the packet, all data will be garbage bytes.

I²C packet format:



3. **Garbage bytes + NMEA**--- When the MT3339/MT3333 I²C buffer is empty, the master will keep reading I²C packet (will read garbage bytes in the beginning). If this reading process is not completed before MT3339/MT3333 uploading next new data to I²C buffer, the master will accept new/valid NMEA data bytes and overlaps the previous unfinished data.

I²C packet format:



3.3 How to extract valid NMEA data from numerous I²C packets

As described in section above, valid NMEA data are to be extracted from many I²C packets, and sample code will be provided for valid NMEA data extraction. Please refer to next section for detail.

Note: According to the factory default setting, all '0A' Characters will be discarded when NMEA data is extracted from I²C packets. An SPI packet comes in 3 formats: (1) "0A" is allocated in the end char in I²C packet; (2) Garbage bytes ('0A' under normal circumstances); (3) "0A" is allocated in the beginning in an I²C packet of NMEA sentence. Discarding "0A" doesn't affect parsing NMEA sentence.

4. Inserting PMTK Command via I²C bus (MT3339/MT3333)

User can insert PMTK command via I²C bus since the capacity of MT3339/MT3333 I²C RX buffer is 255 bytes. For one I²C packet that the slave & master transmits, the size must be less than 255 bytes, and the time interval of two I²C packet inputs cannot be less than 10 milliseconds because the slave needs 10 milliseconds to process input data.

5. Sample Code to Receive NMEA and Send PMTK

MediaTek(MTK) provides API and sample code for the devices to receive NMEA data and send PMTK.

5.1 Provided API for Receiving Queue

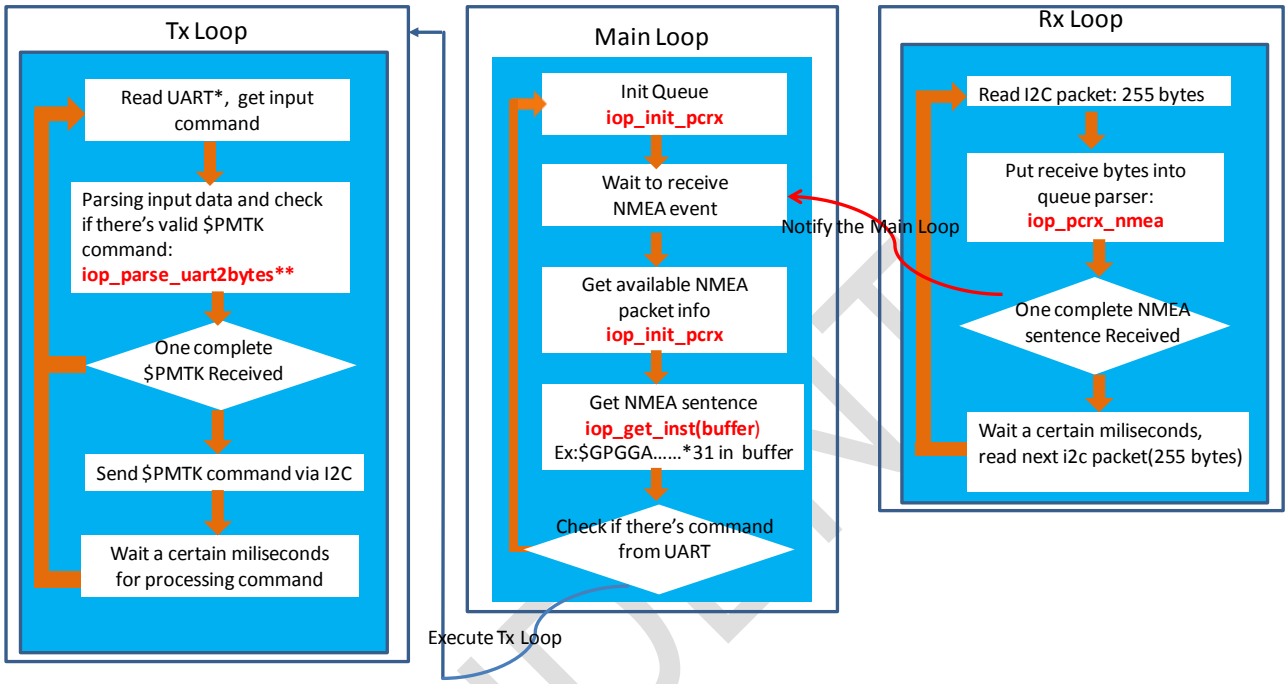
After queue is received, it will extract NMEA and MTK Debug data from the incoming I²C packets. It will also discard garbage bytes and will valid data automatically.

Document	Global Top GPS NMEA over I2C Software Guide				
Author	Stanly Lin	**Date**	2014/10/01	**Ver.**	1.0

Function Name	Description	Prototype and Parameter
iop_init_pcrx	Initialize receive queue	
iop_inst_avail	Get available NMEA sentence information.	BOOL iop_inst_avail(short *inst_id, short *dat_idx, short *dat_siz) inst_id - NMEA sentence type dat_idx - start data index in queue dat_siz - NMEA sentence size
iop_get_inst	Get NMEA sentence data from queue buffer	void iop_get_inst(short idx, short size, void *data) idx - start data index in queue size - NMEA sentence size data - data buffer used to save NMEA sentence
iop_pcrx_nmea	Process I²C packets, get valid NMEA data and discard garbage bytes.	void iop_pcrx_nmea(unsigned char data)
iop_pcrx_nmea_dbg_hbd_bytes	Process I²C packets, get valid NMEA data and debug log code.	void iop_pcrx_nmea_dbg_hbd_bytes(unsigned char aData[], int i4NumByte) aData[] - the array keeps all data read form I²C packets. i4NumByte - size of the array keeps all data read form I²C packets

5.2 Firmware Flow Chart After Receiving Queue

The flow chart is shown below:



Note:

*The item "UART" here means the MCU Serial Port connecting to PC or other Host. User can get NMEA and input $PMTK commands through this interface.

**This function is platform dependent. It is used for MCU to receive UART input data. For different platform, users must implement this function by themselves. The $PMTK command is ranging from 8 to 64bytes in length. It is required to have enough buffer size in firmware to keep these data bytes.