G3 CAN bus configuration guide (v1.3)

Revision log:
v1.3>add support for datatype BITS, CANRAW
v1.2>add support for #comment
>add support for different polling interval for each asset
>add [multiplier, adder] optional arguments for data type FLOAT
v1.1> add support for ISO15765-4 protocol (part of OBD-II)
>add support for CAN bus to run in query mode or read mode

Filename :	iotasset.txt
Location :	\user

1. Introduction

   The file 'iotasset.txt' contains the assets configuration that is required by the CAN bus program to acquire data from CAN bus devices and also pre-process for downstream IoT clients.

2. IOT asset 'KEY,VALUE' general format

   Each IOT asset is defined by using a BLOCK of 'key, value' pairs (CSV format).
   There are four CAN bus key names that must be present for each IOT asset.
   These CAN bus key names are reserved and cannot be used for custom key names.

CAN bus KEYS	**Description**
TYPE	Define the type of CAN bus communication
CANID	Define the CAN bus message ID
CANREQ	Define the CAN bus message data (for query message)
CANDATA	Define the CAN bus raw data parsing and data type conversion

   Custom keys can be freely defined but limited to eight custom keys.
   Each asset block must include the same set of custom keys.
   Backslash (\) and double quote mark (") char cannot be used.
   Comments can be inserted by using the hash (#) sign.

   To ease parsing of different types of assets, the asset blocks need to be located between the start and end of block markers.

CAN bus BLOCK MARKER	**Description**
CAN_START	Define the start of CAN bus assets
CAN_STOP	Define the end(stop) of CAN bus assets

3. IOT asset 'KEY,VALUE' setup information

TYPE, m [, i]

Argument	Value	Description
m	OBD	OBD mode: ISO15765-4 protocol (part of OBD-II)
	C2Q	Query mode: send CAN query and read CAN response
	C2R	Read mode: read CAN message only [#1]
i	1, 2, 3, 4, 5,….	Poll interval for each asset. [#2]

#1 For read mode, the poll interval [i] will be ignored if argument included.

#2 Optional: Argument [i] if excluded will result in default polling i=1 (polls on every interval).

Example of Poll Interval calculations with master Poll Period = 15 sec.

note: Poll Period is the time interval between polling, refer to web config 'IoT Hardware'.

Asset Poll Period	Calculation	Poll Interval (i)
1min	1*60/15	4
30min	30*60/15	120
1 hour	1*60*60/15	240
3 hour	3*60*60/15	720

CANID, n0 Read mode

CANID, n1, n2 OBD/Query mode

Argument	TYPE	Value	Description
n0	C2R		n0 = ID for read CAN message
		000-7FF	11-bit message ID for CAN 2.0A (std)
		00000000-1FFFFFFF	29-bit message ID for CAN 2.0B (ext)
n1, n2	OBD C2Q		n1 = ID for query/request CAN message n2 = ID for response CAN message
		000-7FF	11-bit message ID for CAN 2.0A (std)
		00000000-1FFFFFFF	29-bit message ID for CAN 2.0B (ext)

CANREQ, s

Argument	TYPE	Value	Description
s	OBD	0000-FFFF	"mode"+"PID" eg 010C where mode=01, PID=0C
	C2Q	0	No data byte (DLC=0) [#3]
		1, 2, 3, 4, 5 …. 8	eg E9 (DLC=1), 0C1122334F (DLC=5) [#4]
	C2R	0	No query message in read mode

#3 Data Length Code (DLC) in CAN message

#4 Each byte is represented by 2 hexadecimal chars

CANDATA, t, u, v [, x, y] datatype: INTEGER, STRING, FLOAT, CANRAW
CANDATA, B.b, c, v [, x, y] datatype: BITS

Argument	Value	Description
t, u	t=byte start u=byte length	Position of starting byte (dec: 1-8) Length of byte (dec: 1-8)
B.b, c	B.b=(Byte_start).(bit_start) c=bits length	Position of starting Byte.bit (dec: 1.1-8.8) Length of bits (dec: 1-8) #5
v	Data Type	Data type as conversion from CAN bus raw data
x	Multiplier	Value = Value*Multiplier + Adder #6
y	Adder	Value = Value*Multiplier + Adder #6

#5 Bits parsing can only be applied on single byte of CAN data and not across multiple bytes.
#6 Optional: for Data Type INTEGER & FLOAT, **both** x & y arguments required when applied.

4. Data Type definitions for OBD/CAN bus

DATA TYPE BITS

v [Data Type]	c [Data Length (bits)]	Description
BITS	1-8	1-8 bits to unsigned integer #7

#7 Binary value parsed will be converted to decimal value, eg 1110_2 will be reported as 14_{10}. Bits parsing can only be applied on single byte of CAN data and not across multiple bytes.

DATA TYPE INTEGER

v [Data Type]	u [Data Length (bytes)]	Description
UINT8	1	8-bit data to **8-bit unsigned integer**
SINT8		8-bit data to **8-bit signed integer**
UINT16HL	2	8-bit data pair to **16-bit unsigned integer**, big endian
UINT16LH		8-bit data pair to **16-bit unsigned integer**, little endian
SINT16HL		8-bit data pair to **16-bit signed integer**, big endian
SINT16LH		8-bit data pair to **16-bit signed integer**, little endian
UINT32HLhl	4	8-bit data quad to **32-bit unsigned integer**, big endian
UINT32hlHL		8-bit data quad to **32-bit unsigned integer**, Word – little endian, Byte – big endian
UINT32LHlh		8-bit data quad to **32-bit unsigned integer**, Word – big endian, Byte – little endian
UINT32lhLH		8-bit data quad to **32-bit unsigned integer**, little endian
SINT32HLhl		8-bit data quad to **32-bit signed integer**, big endian
SINT32hlHL		8-bit data quad to **32-bit signed integer**, Word – little endian, Byte – big endian
SINT32LHlh		8-bit data quad to **32-bit signed integer**, Word – big endian, Byte – little endian
SINT32lhLH		8-bit data quad to **32-bit signed integer**, little endian

DATA TYPE STRING

v [Data Type]	u [Data Length]	Description
STRING8	8	Set of eight 8-bit data to **8 ASCII characters** (abcdefgh)
STRING8R		Set of eight 8-bit data to **8 ASCII characters**, reversed (hgfedcba)
STRING4	4	Set of four 8-bit data to **4 ASCII characters** (abcd)
STRING4R		Set of four 8-bit data to **4 ASCII characters**, reversed (dcba)

DATA TYPE FLOAT

v [Data Type]	u [Data Length]	Description
FLOAT32ABCD	4	Set of four 8-bit data to IEEE-754 single precision floating point number. Byte orientation=ABCD,DCBA,BADC,CDAB A,B,C,D=canbyte1,canbyte2,canbyte3,canbyte4
FLOAT32DCBA		
FLOAT32BADC		
FLOAT32CDAB		

DATA TYPE CANRAW

v [Data Type]	u [Data Length (bytes)]	Description
CANRAW	8	String of 16 hexadecimal char

5. Example for IOT asset configuration

#iotasset example for ISO15765-4 protocol (part of OBD-II)

```
CAN_START                          #start of CAN bus block

TYPE, OBD, 10                      #CAN type=OBD, poll on every 10 polling interval
CANID, 7DF, 7E8                    #request ID=0x7DF, response ID=0x7E8
CANREQ, 0105                       #mode=0x01, PID=0x05
CANDATA, 4, 1, UINT8               #byte start=4, byte length=1, data type=unsigned 8-bit
                                   #integer
Unit, degC                         #custom key1
Key, EngineCoolantTemp             #custom key2, max supported=8

TYPE, OBD
CANID, 7DF, 7E8
CANREQ, 010C                       #mode=0x01, PID=0x0C
CANDATA, 4, 2, UINT16HL, 0.2, 0    #data type=unsigned 16-bit integer, value=value*0.2 + 0
Unit, rpm
Key, EngineRPM

TYPE, OBD
CANID, 7DF, 7E8
CANREQ, 010D                       #mode=0x01, PID=0x0D
CANDATA, 4, 1, UINT8
Unit, kmh
Key, VehicleSpeed

TYPE, OBD
CANID, 7DF, 7E8
CANREQ, 0101                       #mode=0x01, PID=0x01
CANDATA, 5.3, 1, BITS
Unit, -
Key, Misfire

CAN_STOP                           #end of CAN bus block
```

#iotasset example for industrial CAN “Query” mode

CAN_START

TYPE, C2Q, 20	#query mode, poll on every 20 polling interval
CANID, 200, 180	#request ID=0x200, respond ID=0x180
CANREQ, 0	#request data=0 (no data)
CANDATA, 1, 2, UINT16HL, 0.5, -2	#data type=unsigned 16-bit integer, value=value*0.5 - 2
Unit, lux	#custom key1
Key, LightSensor	#custom key2, max supported=8

TYPE, C2Q	
CANID, 300, 280	#request ID=0x300, respond ID=0x280
CANREQ, 021A05	#request data=0x021A05
CANDATA, 4, 2, UINT16HL	
Unit, kWh	
Key, Power	

TYPE, C2Q, 10	#poll on every 10 polling interval
CANID, 400, 380	#request ID=0x400, respond ID=0x380
CANREQ, 08FF	#request data=0x08FF
CANDATA, 0, 1, UINT8	
Unit, volt	
Key, Battery	

TYPE, C2Q	
CANID, 7DF, 7E8	#request ID=0x7DF, respond ID=0x7E8
CANREQ, 02010D5555555555	#request data=0x02010D5555555555
CANDATA, 3, 1, UINT8	
Unit, rpm	
Key, TurbineSpeed	

CAN_STOP

#iotasset example for industrial CAN “Read” mode

```
CAN_START

TYPE, C2R                       #read mode
CANID, 37F                      #message ID=0x37F
CANREQ, 0                       #request data=0 (no data)
CANDATA, 1, 2, UINT16HL         #data type=unsigned 16-bit integer
Unit, kg
Key, SensorA

TYPE, C2R                       #read mode
CANID, 38F                      #message ID=0x38F
CANREQ, 0                       #request data=0 (no data)
CANDATA, 1, 2, UINT16HL         #data type=unsigned 16-bit integer
Unit, mm
Key, SensorB

TYPE, C2R                       #read mode
CANID, 39F                      #message ID=0x39F
CANREQ, 0                       #request data=0 (no data)
CANDATA, 1, 2, UINT16HL         #data type=unsigned 16-bit integer
Unit, bar
Key, SensorC

CAN_STOP
```

6. Method to upload ‘iotasset.txt’ file to G3

-Upload the iotasset.txt file from your computer using the ‘Upload iotasset.txt’ button in the ‘IoT Hardware’ tab.

-Put the iotasset.txt file in \user folder of USB drive (with label ‘FATBOX’).
Plug the USB drive into G3 and click the ‘Upload to FATBOX’ button in the ‘Management’ tab.

-Use SCP/Putty console or WinSCP.

<EOF>