G3 Modbus configuration guide (v2.2)

Revision log:
v2.2> add support for data type INT8
> add support for Modbus/RTU over TCP
v2.1> add block marker MBM_START, MBM_STOP
>add multiplier, adder optional arguments for DATA TYPE INTEGER only
v2.0> add support for FLOAT data type.
>add support for special Modbus protocol using 32bit registers
>revision of the REGS data type definition.

Filename : iotasset.txt
Location : \user

A. Introduction

This file 'iotasset.txt' contains the assets configuration that is read by the Modbus BOT program when it starts up.

B. IOT asset 'KEY,VALUE' general format

Each IOT asset is setup using a BLOCK of 'key, value' pairs (CSV format).
There are four (4) default key names that must be present for each IOT asset.
These default key names are reserved and cannot be used for custom key names.

MODBUS

Default key	Description
TYPE	To define the type of Modbus communication
ADDR	To define the address of the Modbus slave device
MBFC	To define the Modbus function code to query device
REGS	To define the Modbus registers and other data format setting

Customer can add-on their custom key but limited to a maximum of eight (8) unique custom key only. This means a maximum of 12 keys per asset (ie 4 default keys + 8 custom keys).
Each asset must be setup using the same set of unique custom keys.
Space char will be automatically removed, empty line will also be ignored.
Backslash(\) and double inverted commas(") chars cannot be used in 'key, value' setup.

Modbus asset blocks must start and end with MBM_START marker and MBM_STOP marker respectively. It is possible to have more than one set of block marker in the iotasset.txt file.

C. IOT asset 'KEY,VALUE' setup information

TYPE,m

Argument	Value	Description
m	R	Standard Modbus/RTU (over RS232/485)
	T	Standard Modbus/TCP (over Ethernet)
	R32	Modbus/RTU using 32bit registers (MBFC=3,4 only)
	T32	Modbus/TCP using 32bit registers (MBFC=3,4 only)

ADDR,n0 **[Modbus/RTU]**
ADDR,n1.n2.n3.n4:p **[Modbus/TCP]**
ADDR,n1.n2.n3.n4:p:u **[Modbus/RTU over TCP]**

Argument	Value	Description	Notes
n0	1-247	Slave address	For Modbus/RTU
n1.n2.n3.n4:p	n1,n2,n3,n4=0-255 p=502 (default)	Slave IP address Slave port	For Modbus/TCP
n1.n2.n3.n4:p:u	n1,n2,n3,n4=0-255 p=502 (default) u=1-247	Slave IP address Slave port Unit ID	For Modbus/RTU over TCP note: use TYPE,T

MBFC,s

Argument	Value	Description
s	1	Read Coil Status (FC=01)
	2	Read Input Status (FC=02)
	3	Read Holding Registers (FC=03)
	4	Read Input Registers (FC=04)

REGS,t,u,v,x,y

Argument	Value	Description
t	0,1,2,3..max depend on slave device	Data address of first register requested (decimal)
u	[Number of register]	Number of register requested (decimal)
v	[Data Type]	Data type as converted from Modbus raw data
x *	INTEGER value multiplier	Value = Value*Multiplier + Adder (decimal)
y *	INTEGER value adder	Value = Value*Multiplier + Adder (decimal)

*optional for DATA TYPE INTEGER only, requires both arguments x, y when applied.

<DATA TYPE BOOLEAN FOR MODBUS 1-BIT REGISTER>

v[Data Type]	u[Number of register]	Description
BOOL	1	Boolean value, ie 0 or 1

<DATA TYPE INTEGER FOR MODBUS 16-BIT REGISTERS>

v[Data Type]	u[Number of register]	Description
UINT8H/UINT8L	1	16-bit register to 8-bit unsigned integer, HI byte / LO byte
SINT8H/SINT8L		16-bit register to 8-bit signed integer, HI byte / LO byte
UINT16HL	1	16-bit register to 16-bit unsigned integer, big endian
UINT16LH		16-bit register to 16-bit unsigned integer, little endian
SINT16HL		16-bit register to 16-bit signed integer, big endian
SINT16LH		16-bit register to 16-bit signed integer, little endian
UINT32HLhl	2	16-bit register pair to 32-bit unsigned integer, big endian
UINT32hlHL		16-bit register pair to 32-bit unsigned integer, Word – little endian, Byte – big endian
UINT32LHlh		16-bit register pair to 32-bit unsigned integer, Word – big endian, Byte – little endian
UINT32lhLH		16-bit register pair to 32-bit unsigned integer, little endian
SINT32HLhl		16-bit register pair to 32-bit signed integer, big endian
SINT32hlHL		16-bit register pair to 32-bit signed integer, Word – little endian, Byte – big endian
SINT32LHlh		16-bit register pair to 32-bit signed integer, Word – big endian, Byte – little endian
SINT32lhLH		16-bit register pair to 32-bit signed integer, little endian

<DATA TYPE STRING FOR MODBUS 16-BIT REGISTERS>

v[Data Type]	u[Number of register]	Description
STRING16	8	Set of eight 16-bit registers to 16 ASCII characters (abcdefghijklmnop)
STRING16R		Set of eight 16-bit registers to 16 ASCII characters, byte swapped (badcfehgjilknmpo)
STRING8	4	Set of four 16-bit registers to 8 ASCII characters (abcdefgh)
STRING8R		Set of four 16-bit registers to 8 ASCII characters, byte swapped (badcfehg)

<DATA TYPE FLOAT FOR MODBUS 16-BIT REGISTERS>

v[Data Type]	u[Number of register]	Description
FLOAT32ABCD	2	16-bit register pair to single precision (32-bit) floating point number. Byte orientation=ABCD,DCBA,BADC,CDAB AB=word data from first 16-bit register CD=word data from second 16-bit register
FLOAT32DCBA		
FLOAT32BADC		
FLOAT32CDAB		

<DATA TYPE INTEGER FOR MODBUS 32-BIT REGISTERS>

v[Data Type]	u[Number of register]	Description
UINT32HLhl	1	32-bit register to 32-bit unsigned integer, big endian
UINT32hlHL		32-bit register to 32-bit unsigned integer, Word – little endian, Byte – big endian
UINT32LHlh		32-bit register to 32-bit unsigned integer, Word – big endian, Byte – little endian
UINT32lhLH		32-bit register to 32-bit unsigned integer, little endian
SINT32HLhl		32-bit register to 32-bit signed integer, big endian
SINT32hlHL		32-bit register to 32-bit signed integer, Word – little endian, Byte – big endian
SINT32LHlh		32-bit register to 32-bit signed integer, Word – big endian, Byte – little endian
SINT32lhLH		32-bit register to 32-bit signed integer, little endian

<DATA TYPE STRING FOR MODBUS 32-BIT REGISTERS>

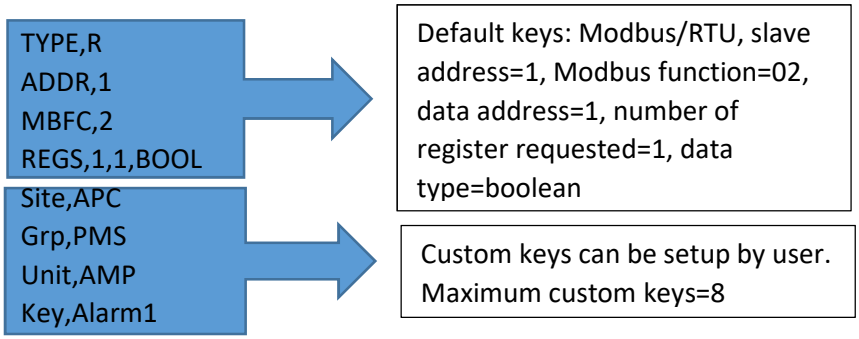
v[Data Type]	u[Number of register]	Description
STRING16	4	Set of four 32-bit registers to 16 ASCII characters (abcdefghijklmnop)
STRING16R		Set of four 32-bit registers to 16 ASCII characters, word swapped (dcbahgfelkjiponm)
STRING8	2	Set of two 32-bit registers to 8 ASCII characters (abcdefgh)
STRING8R		Set of two 32-bit registers to 8 ASCII characters, word swapped (dcbahgfe)

<DATA TYPE FLOAT FOR MODBUS 32-BIT REGISTERS>

v[Data Type]	u[Number of register]	Description
FLOAT32ABCD	1	32-bit register to single precision (32-bit) floating point number. Byte orientation=ABCD,DCBA,BADC,CDAB ABCD=double word data from a 32-bit register
FLOAT32DCBA		
FLOAT32BADC		
FLOAT32CDAB		

D. Example setup IOT asset

MBM_START



TYPE,R
ADDR,2
MBFC,3
REGS,25,1,UINT16HL,0.65,100
Site,APC
Grp,PMS
Unit,AMP
Key,Voltage

Optional multiplier=0.65
Optional adder=100

TYPE,T
ADDR,192.168.1.100:502
MBFC,3
REGS,100,2,UINT32HLhl
Site,APC
Grp,PMS
Unit,AMP
Key,Temperature

TYPE,T
ADDR,192.168.1.120:502
MBFC,4
REGS,200,8,STRING16
Site,APC
Grp,PMS
Unit,AMP
Key,RAMmessage

MBM_STOP

E. Method to download ‘iotasset.txt’ file to G3

Save the file inside \user folder of a USB drive labelled ‘FATBOX’.
Plug in the USB drive and click the ‘Download to FATBOX’ button in the ‘Management’ tab of web configuration.

<EOF>