AVA BUS OVERVIEW

What is Modbus?

Modbus is a serial communication protocol developed by Modicon published by Modicon® in 1979 for use with its programmable logic controllers (PLCs). In simple terms, it is a method used for transmitting information over serial lines between electronic devices. The device requesting the information is called the Modbus Master and the devices supplying information are Modbus Slaves. In a standard Modbus network, there is one Master and up to 247 Slaves, each with a unique Slave Address from 1 to 247. The Master can also write information to the Slaves. Find more info here: http://www.simplymodbus.ca/FAQ.htm

What is CAN Bus?

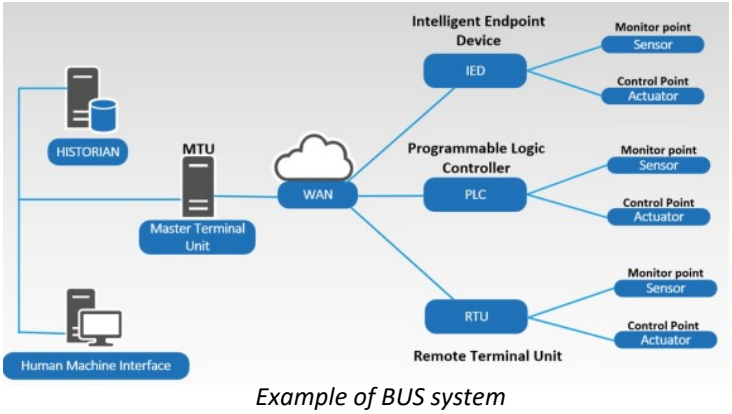
In short, the Controller Area Network (CAN) is a standard used to allow Electronic Control Units (ECUs) to communicate in an efficient manner without a central computer. Messages are broadcast in a system that requires very little physical wiring making **CAN bus** low cost, robust and efficient.

Can you offer Profi-Bus?

Our engineers are currently working on Profi-Bus capabilities for the AVA actuators and we hope to have further update on this later in year. Keep checking www.avacuators.co.uk for more information about our progress.

Which Actuators can work with BUS systems and how?

Our AVA Smart actuators are capable of being supplied compatible with the systems as detailed above, Modbus and Canbus are currently available with our AVA 20, 110, 200 and 400Nm actuators. Other models are being added to the programme. The actuators are available as BUS Smart On Off or BUS Smart Modulating. The following pages will give you an overview of how the Smart actuators Firmware can be used to calibrate and set-up our actuators to your BUS system. The Smart intuitive menu system allows the user to customise and setup actuators specific to your requirement. When we say our actuators are Smart, we really mean it and our BUS actuator series further



Example of BUS system

RTU mode：Host demand data message formats

ADDR FUNC LEN D0…Dn-1 CRC
[ADDR]:Slaver address
[FUNC]:Function code： [XX]
[LEN]: Length of [$D_{0\cdots Dn-1}$] [YY]
[$D_0 \ldots D_{n-1}$]: Information content LEN x 8bit [$DAT_0 \ldots DAT_{n-1}$]
[CRC]: CRC16 code,host calculation leads to CRC code [CRCL CRCH]
Notice：The function formats data listed in this document are all hexadecimal.

【Distribution details of register function】

Register Addr (hexadecimal)	Name	Read-write state	Data length	Data content	Detailed account
00 01	Actuator Status	R	1	specific data as follows：	
				0xAA	actuator acting command_ON
				0xBB	actuator acting command_OFF
				0xAC	actuator acted ON
				0xBC	actuator acted OFF
				0xBA	actuator acting command_BRAKE
				0xEA	actuator error status
00 02	Reserved	R	1		
00 03	Valve work mode	R	1	Specific data as follows：	
				0x22	two way mode
00 04	Controlling command	R/W	1	specific data as follows：	
				0x01	command-on
				0x07	command-off
				0xBA	command-brake
00 05	Delay feedback time	R/W	1	0x01-0x32	1-50ms（default：2ms）
00 06	Actuator Bus_ID	R	1	0x01-0xFC	bus ID（1-252）
00 0D-00 0E	Production Date	R	1	YY YY MM DD, 0X140D 0X041C ==>2013-04-28	
00 11-00 13	Use times	R	4	0x00LL 0xHHMM, Usetimes=LL+256*MM+65535*HH	
00 14	Accurate position	R	2	0xZHZL	Open Degree A(0-1000)
					ZL=A%256，ZH=A/256;
					The openning of actuator=(A/10)%
00 21	Fault status	R	1	0x00	No fault
				0xEA	Fault Status Clear the changed command

【Communicational function code between host and RS485 bus valve】

03/04 Read data out of valve register, even several continuous data is permitted.

06 Write single byte data in valve register

【 Command formats of reading valve operation sent by host 】

Valve_Addr	03/04	00	XX	00	YY	CRCL	CRCH

Within：Valve_Addr means valve bus address of reading operation；

03/04 means reading valve register；

00 XX means initial address to be read out(XX is only a 19-byte address from 0x01 to 0x13)

00 YY means (YY-XX+1)data read from 00 XX to(00 XX+00 YY) in register；

CRCL CRCH means two-byte CRC check code sent by host

【 Data formats of reading valve to operate valve returning accurately 】

Valve_Addr	03/04	YY*2	DAT_1	DAT_2	…	DAT_{YY*2}	CRCL	CRCH

Within：Valve_Addr means returning data of valve bus address after receive reading operation；

03/04 means reading operation of valve register ；

YY means having read out YY*2 byte data；

DAT_1 DAT_2…DAT_{YY} are read-out data；

CRCL CRCH means two-byte CRC check code calculated by valve MCU.

【 Command formats of single-byte writting valve operation sent by host 】

Valve_Addr	06	00	XX	00	DAT	CRCL	CRCH

Within：Valve_Addr means valve bus address to be written；

06 means writting valve register；

00 XX means register address in need of writting in register(only writtable register is permitted) ；

00 DAT is write-in data in demand；

CRCL CRCH means two-byte CRC check code calculated by host.

【 Data formats of single-byte writting valve to operate valve returning accurately 】

Valve_Addr	06	00	XX	00	DAT	CRCL	CRC

Within：Valve_Addr means returnig data of valve bus address after receive writtng operation ；

06 means writting valve register operation；

CRCL CRCH means two-byte CRC check code calculated by valve MCU.

【 Data formats of valve returning on inaccurate operation 】

Valve_Addr	Function code+0x80	Error Code	CRCL	CRCH	

Within：Valve_Addr means valve bus address on error operation；

Function code+0x80 means inaccurate return；

Position means having returned valve position ；

Error code：means inaccurate return，details as follow：

```
#define Err_FunCode              0x65    //101
#define Err_Register_Addr        0x66    //102
#define Err_Data_OutRange        0x67    //103
#define Err_Device               0x68    //104
#define Err_NOCMD                0xC9    //201
#define Err_Modbus_Param         0xCB    //203
```

II Read out valve data which are written in register

It's nessesary to realize bus address of valve to be operated before writting register operation in valve (Valve_Addr).

※ Read out valve bus address code without preconditions(Valve_Addr)：（Notice：This command is only suitable for single valve suspended in bus.）

Send command：FD 5F 00 01 00 01 11 FB

Correct return：Valve_Addr 5F 00 01 00 01 CRCL CRCH

Error return：Valve_Addr DF 01 Error code CRCL CRCH

1、Command formats of reading out single byte data

Send format：Valve_Addr 03/04 00 XX 00 01 CRCL CRCH

Within：Valve_Addr means valve bus address on present operation；

03/04 means reading valve register operation；

00 XX means register address to be read out (XX could only be 0x01-0x13)；

00 01 means having read out a byte data；

CRCL CRCH means two-byte CRC check code.

When Valve_Addr =0，null data return；

When Valve_Addr!=0，valve corresponded in address code could return；

When the returning code is correct,it will get：Valve_Addr 04 02 DATH DATL CRCL CRCH

Within：Valve_Addr means returning data of valve bus address after receive reading operation；

DATH,DATL are read-out data；

CRCL CRCH means two-byte CRC check code calculated by valve MCU.

【Example】Query valve bus addr. 01 state ：

Host sends：01 04 00 02 00 01 90 0A

Valve returns：01 04 02 00 64 B8 DB (Bus addr.is 01 valve is in open position)

2、Command formats of reading out continuous multibyte data：

Send format：Valve_Addr 04 00 XX 00 YY CRCL CRCH

Within：Valve_Addr means valve bus addr. at present operation；

00 XX means reading out initial addr. from several continuous registers.(XX can only be 0x01-0x18)；

00 YY means the numbers of read-out registers；

CRCL CRCH means two-byte CRC check code.

When Valve_Addr =0，null data return；

When Valve_Addr!=0，valve corresponded in address code could return；

Whenthe returning code is correct,it will get：Valve_Addr 04 YY*2 DAT_1 DAT_2…DAT_{YY*2} CRCL CRCH

Within：Valve_Addr means the returning data of valve bus address after receiving read operation；；

YY*2 means the numbers of bytes read out from valve code；

DAT_1 DAT_2…DAT_{YY*2} are read-out data；

CRCL CRCH means two-byte CRC check code calculated by valve MCU.

【Example 1】Query bus addr. 00 12 to 00 15;

Host sends： 01 04 00 12 00 04 51 CC

Valve returns next：01 04 08 00 14 00 0C 00 04 00 1C 20 05

【Example 2】Query used time of bus addr.01 valve:

Host sends: 01 04 00 16 00 03 51 CF

Repeater returns first: 01 04 01 E5 80 02

Valve returns next: 01 04 03 05 00 00 E0 CF (means bus addr. 01 valve has used 5 times)

III Modify the data written in valve register

Only the writtable register data can be modified

1、 Command formats of modifying Valve_Addr1 to Valve_Addr2 :

Send formats: Valve_Addr1 06 00 06 00 Valve_Addr2 CRCL CRCH

Within: Valve_Addr1 means valve bus addr.at present operation;

Valve_Addr2 means modified valve bus addr.;

CRCL CRCH means two-byte CRC check code.

When Valve_Addr =0, null data return;

When Valve_Addr!=0, valve corresponded in address code could return;

Whenthe returning code is correct,it will get: Valve_Addr1 06 01 Valve_Addr2 CRCL CRCH

【Example】Modify bus addr. 01 to bus addr. 02:

Host send : 01 06 00 06 00 02 E8 0A

Valve returns: 01 06 00 06 00 02 E8 0A (means having succeed to modify bus addr. 01 to bus addr. 02)

2、Control valve action

Send format: Valve_Addr 06 00 04 00 XX CRCL CRCH

Within: Valve_Addr means valve bus addr.at present operation;

00 04 means register which memoried valve controlling command.

00 XX means controlling command code.Please refer to the following chart about the details;

CRCL CRCH means two-byte CRC check code.

When Valve_Addr =0, null data return;

When Valve_Addr!=0, valve corresponded in address code could return;

Whenthe returning code is correct,it will get: Valve_Addr 06 00 04 00 XX CRCL CRCH

【Example】Bus addr.01 valve controlling command as follows:

ControlCommand	Send Command	Valve return	Remark
close valve	01 06 00 04 00 00 C8 0B	01 06 00 04 00 00 C8 0B	
open valve	01 06 00 04 00 64 C9 E0	01 06 00 04 00 64 C9 E0	
open30°	01 06 00 04 00 1E 48 03	01 06 00 04 00 1E 48 03	peculiar to 4-position 2-way
open60°	01 06 00 04 00 3C C8 1A	01 06 00 04 00 3C C8 1A	peculiar to 4-position 2-way
open toB33	01 06 00 04 00 B3 89 BE	01 06 00 04 00 B3 89 BE	peculiar to 3-position 3-way
brake	01 06 00 04 00 BA 49 B8	01 06 00 04 00 BA 49 B8	