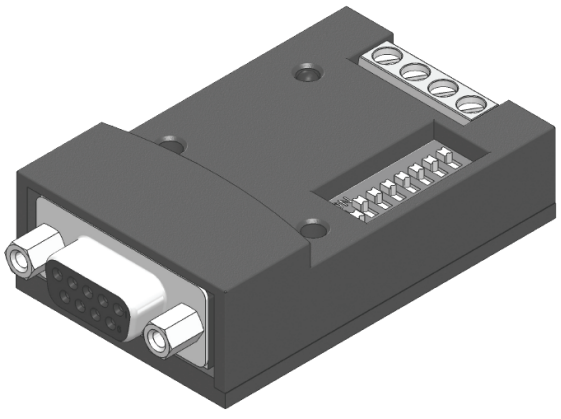
UIROBOT®
United Intelligence Robot Technology

User Manual

UIM25001 CAN-RS232 Converting Controller for UIM242xx Controller



UIM25001
CAN-RS232 Converting Controller for UIM242xx Controller

Features

Embedded DSP Microprocessor

- Embedded 16-bit high-performance digital signal micro processor
- Simple structured, intuitive, rich instructions
- Intelligent, fault-tolerating, user-friend interface

CAN2.0B Active Communication

- 2-wire interface
- 1 Mega bit/sec operation, long distance
- High noise immunity due to differential bus implementation

RS232 Communication

- RS232 three-wire serial communication
- Max baud rate 112500 bps
- Automatic Baud Rate Detection

Wide supply voltage

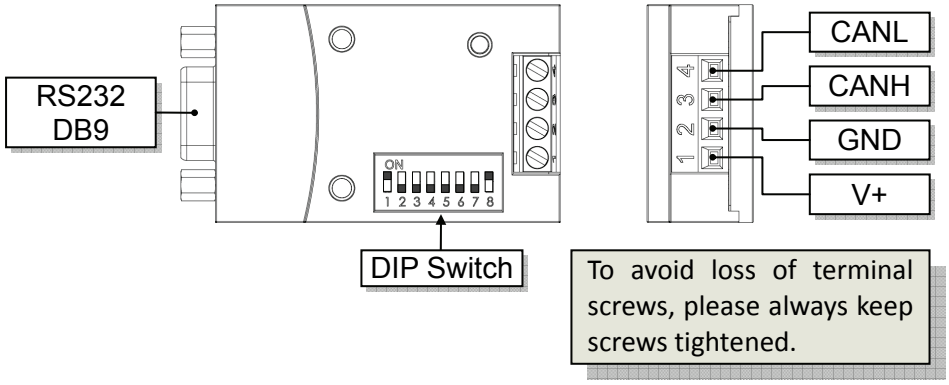
- Wide supply voltage range 6 ~ 40VDC

Description

The UIM25001 CAN-RS232 converting controller is used in conjunction with UIM242xx stepper motor controller to provide a RS232 interface on the user side and a CAN bus interface on the motor side (factory side). By using the UIM25001 controller, users can be benefited from the advantages of CAN base control network without dealing with the complicated CAN protocol. User only needs to communicate with the UIM25001 controller using the simple, intuitive and ASCII coded RS232 instructions. All CAN bus issues are taken care of by the UIM25001 converting controller.

Both the UIM25001 controller and the UIM242xx controller support network. One UIM25001 controller can command up to 100 UIM242xx controllers. Interfacing the UIM25001 is simple, intuitive and fault tolerating. Users are not required to have stepper motor driving or CAN protocol knowledge.

Terminal Description

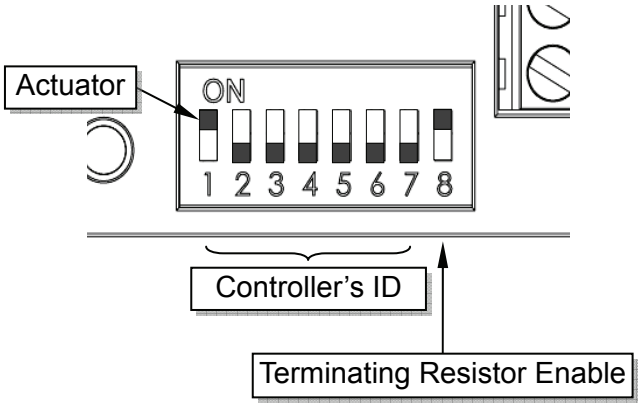


Screw Terminals

Terminal No. / Color	Name	Description	MIN	NOM	MAX	UNIT
1 / Red	V+	Supply voltage	12		40	VDC
2 / Black	GND	Supply voltage ground		0		VDC
3 / White	CANH	CAN High-Level Voltage I/O				
4 / Green	CANL	CAN Low-Level Voltage I/O				

DIP switch functions

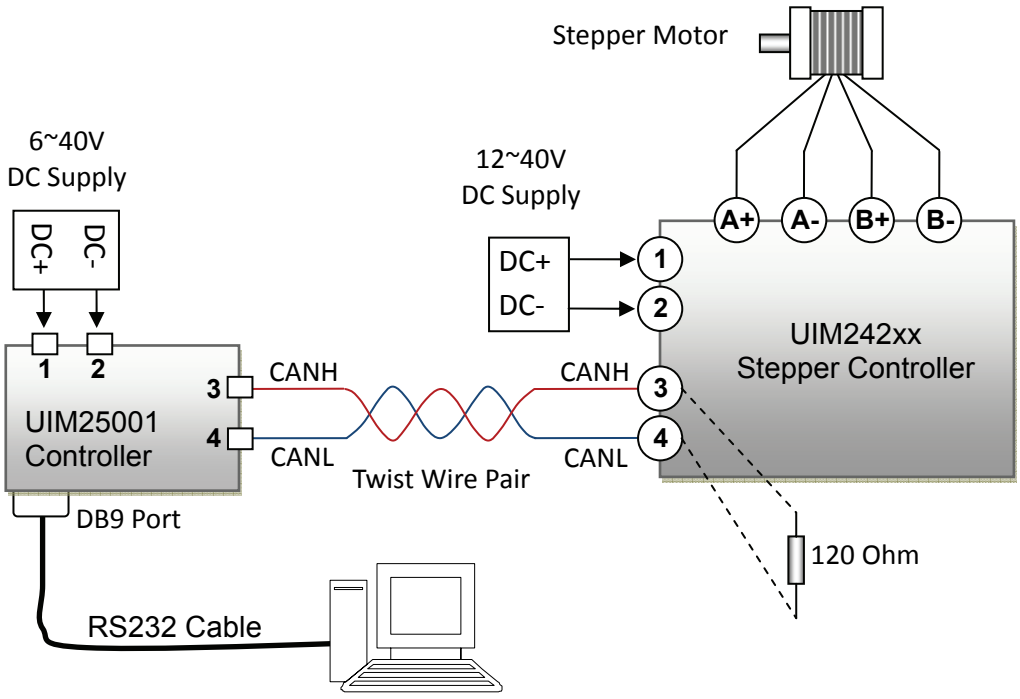
UIM25001 controller has an 8-bit DIP switch, which serves multiple functions. During power-up, D1~D7 are read as the controller's ID/address. After that is done, D1 and D2 can be used achieve some RS232 special features (refer to the "using the RS232 communication" section). Put D8 on the "ON" position will enable the built-in terminating resistor. Unless necessary, please maintain the DIP positions as shown in following Figure.



Typical Applications

Standalone Operation

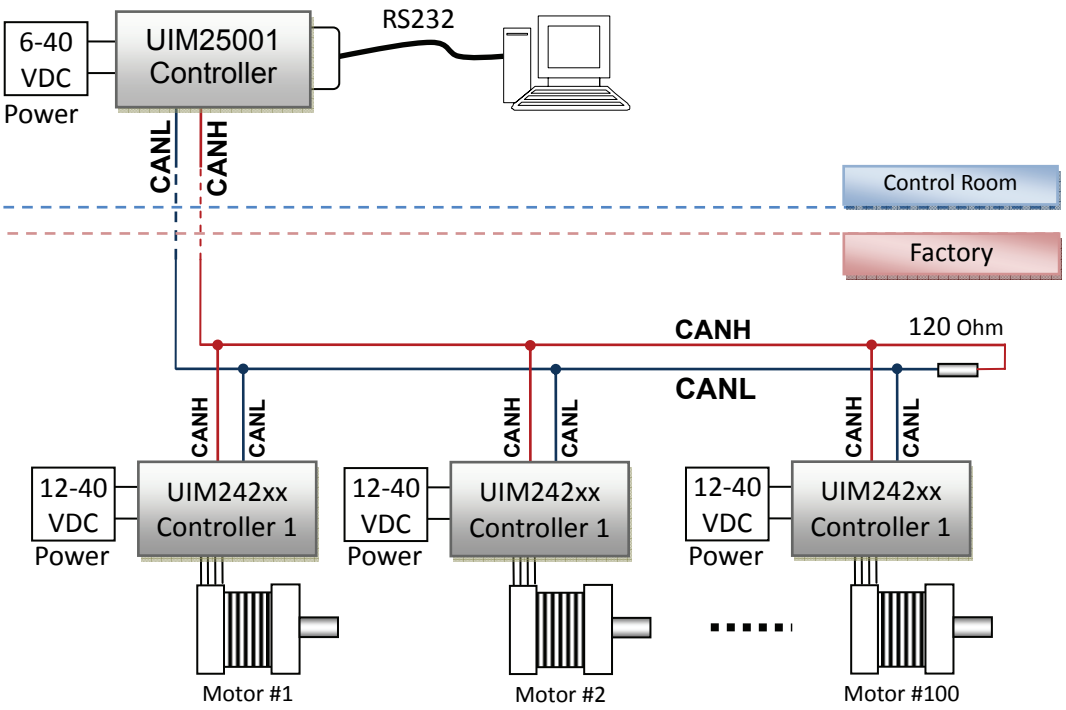
UIM242xx controllers can work in a network as well as standalone. When working standalone, there is only one RS232/CAN converter linked with one UIM242xx controller. In this case, user can use wiring scheme as shown below.



Note: For long distance transfer, the two ends of the bus should be terminated with120Ω terminating resistors. UIM25001 converter has a build-in terminating resistor. User only needs to attach a resistor at the other end of the bus. To enable the UIM25001 converter's terminating resistor, see the UIM25001 manual. To achieve the best communication performance, CANH and CANL should use a twisted pair.

Network Operation

Multiple UIM242xx controllers can be wired together to form a reliable local network. Following figure provides a typical network wiring solution. Detailed terminal wiring on each controller can be found in previous "standalone Operation" solution.



Note: In multi-node CAN applications, it is important to maintain a direct point-to-point wiring scheme. A single pair of wires should connect each element of the CAN bus, and the two ends of the bus should be terminated with 120Ω resistors. A star configuration should never be used. UIM25001 converter has a build-in terminal resistor. User needs to attach a resistor at the other end of the bus. To enable the UIM25001 converter's terminating resistor, see the UIM25001 manual.

Any deviation from the point-to-point wiring scheme creates a stub. The high-speed edge of the CAN data on a stub can create reflections back down the bus. These reflections can cause data errors by eroding the noise margin of the system. Although stubs are unavoidable in a multi-node system, care should be taken to keep these stubs as small as possible.

Characteristics

Absolute Maximum Ratings

Supply Voltage	40VDC
RS232 interface Voltage for RX to GND Voltage for TX to GND	 ±25V ±13.2V
Store Temperature	-20℃ ～ +125℃
Working Temperature	-20℃ ～ +100℃

Working under environment exceeding maximum value could damage the controller

Electrical Characteristics（Ambient Temperature 25℃）

Supply Voltage	6V ～ 40VDC
Current consumption	Max 100 mA

Communication (Ambient Temperature 25℃）

To User Device	RS232
Wiring Method	DB9 Female Connector
RS232 Baud Rate	MAX 115200 bps
To UIM24xx Controller	Active CAN 2.0B
CAN wiring Method	2-Wire,CANH,CANL
CAN bus	• Supports 1 Mb/s operation • ISO-11898 standard physical layer requirements • Suitable for 12V and 24V systems • Up to 100 nodes can be connected

Environment Requirements

Cooling		Free Air
Working Environment	Environment	Avoid dust, oil mist and corrosive gases
	Temperature	-20 ℃ ～+ 85 ℃
	Humidity	<80%RH， n condensation, no frosting
	Vibration	3G Max
Storage Temperature		-40 ℃ ～+ 125 ℃

Size and Weight

Size	66mm x 38mm x 19mm
Wight	0.1 kg

Overview

The UIM25001 CAN-RS232 converting controller is to be used in conjunction with UIM242xx stepper motor controller to provide a RS232 interface on the user side and a CAN bus interface on the motor side (factory side). UIM25001 Controller's main functions can be summarized as below.

1) Receive instructions in RS232 form from user device, and convert the instructions into more concise and efficient CAN instructions to send to downstream UIM242xx controllers

2) Convert CAN messages from UIM242xx controllers and send back to the user devices in RS232 form

3) Coordinates UIM242xx controllers in the network

4) Store user batch instructions in the EEPROM and control the subordinating UIM242xx controller to work together in the absence of user device intervention. (This feature is available in the enhanced version)

By using the UIM25001 controller, users can be benefited from the advantages of CAN base control network without dealing with the complicated CAN protocol. All CAN bus issues are taken care of by the UIM25001 converting controller. User only needs to communicate with the UIM25001 controller using the simple, intuitive and ASCII coded RS232 instructions. For example, in order to implement (speed = 1000 steps / sec) The following instructions are all valid:

"SPD = 1000;" or "SPD: 1000;" or "SPD 1000;" or "SPD1000;" or "SPD %?&%* 1000;"

If in any case, a wrong instruction is received, the controller will return and error messages. Incorrect instructions will not be discarded without execution to prevent accidents.

UI Robot Co. provides free Microsoft Windows XP bases VB / VC demo software and their source code, to facilitate the quick start of user device side programming.

UIM25001 converting controller can work on a wide range voltage of 6~40VDC.

Connect to UIM25001 through RS232

Handshaking

UIM25001 CAN-RS232 Converting Controllers communicate with user devices using the RS232 serial protocol. The factory default baud rate is 9600. User can use the 9600 baud rate to connect to a newly bought UIM25001 controller.

There are two situations under which the UIM25001 will send a Greeting Message to the user device:

1) When the controller is powered up, it will send the following 13 bytes greeting message to the user device:

0xAA, 0xAB, 0xAC, 0x19, 0, 0, 0, 0, 0, 0, ver1, ver0, 0xFF

where, 0xAA, 0xAB, 0xAC is the greeting message header. 0x19 (i.e. 25) indicate the UIM25001. If ver1 = 0x1 and ver0 = 0x0, then the firmware version is 1.0.

2) When the controller receives following ASCII message:

ABC; (case sensitive and ended with a semicolon ;)

If the user device receives a message starting with 0xAA, 0xAB, 0xAC, the connection between user device and the controller could be considered established.

Change Baud Rate through Instruction

For users want to use other baud rates, please first use 9600 baud rate to establish the connection between UIM25001 controller and user device, then use the following instruction to change the baud rate. The new baud rate will be stored in the onboard non-volatile memory.

Function	Set the RS232 communication baud rate		
Syntax	**BDR = x;**	Variable	Integer x = 1200 … 115200
ACK Message	**0xAA, don't care, 0xBD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	New baud rate will be stored in the controller's non-volatile memory. New baud rate will take effect after the controller is rebooted. Please be aware that not all user devices support the 115200 baud rate.		

Note: All instructions are NOT case sensitive. All instructions need a semicolon at the end. The UIM25001 controller will not process any instruction until a semicolon is received. Instruction length, including end semicolon, cannot exceed 20 characters.

Reset the Baud Rate to Factory Default 9600

In case that the user forgets the baud rate and cannot establish the connection, following process can reset the baud rate to the factory default of 9600:

1. Reboot the controller.
2. In 10 seconds, toggle the DIP1 (DIP switch 1) for two rounds. During toggling, the LED on the controller will flash. If exceed 10 seconds, please restart from the first step.
3. If step 2 is successful, the LED will turn off for one second and re-lit. That indicates the baud rate has been changed to 9600 and ready to use.

Automatic Baud Rate Detection

In cases the user device cannot use above two solutions, the controller can also automatically detect user device's baud rate. To enable the auto baud rate detection function, please follow the process described below:

1. Reboot the controller
2. In 10 seconds, toggle the DIP1 (DIP switch 1) for two rounds. During toggling, the LED on the controller will flash. If exceed 10 seconds, please restart from the first step.
3. If step 2 is successful, the LED will turn off for one second and then re-lit. That indicates the auto baud rate detection function is provoked, and the controller is waiting for the signals from the user device.
4. Send an ASCII code "**U**" (i.e. 0x55, capital letter, no punctuation at the end) from the user device to the UIM25001 controller.
5. In around 0.05 seconds, the controller will be able to calculate the baud rate used to send that letter **U,** store the detected baud rate in the onboard non-volatile memory, and sent back the greeting message. The controller is ready to use.

Assign Identification Code to a UIM242xx Controller

Before operation, every UIM242xx controller needs to be assigned a unique identification code (ID, or address). In a network, one UIM25001 links to multiple UM242xx controllers. Without the ID, the UIM25001 controller will not know to which UIM242xx controller the instruction should be sent.

Before leave the factory, all UIM242xx controllers have been assigned a default ID of 5. User can change the ID using SETADR instruction. Before assign an ID to a UIM242xx controller, please connect the UIM25001 controller and the UIM242xx controller using the standalone operation solution. That is, only one UIM242xx should be connected to the UIM25001 controller (Motor is not needed at this time). After rebooting both controllers, user can use the following instruction to assign an ID to the UIM242xx controller.

Function	Assign identification code/address to a UIM242xx controller		
Syntax	**SETADR = ID;**	Variable	Integer ID = 5、6、7 … 125
ACK Message	**0xAA, ID, 0xDD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	ID assignment has to be performed one controller followed by another. In other word, every time there can only be one UIM242xx controller linked to the UIM25001 controller to get the ID assigned. Once an ID is assigned, the ID will be stored in the UIM242xx controller's non-volatile memory.		

Object Specific Operations

After specified the ID of a UIM242xx controller using ADR instruction, user can use following instructions to operate the specified controller.

Once the specified UIM242xx controller receives an instruction, it will return an ACK message. If the instruction is incorrect, the message will be an error message. The incorrect instruction will be discarded without execution.

Specify the Operation Object by an ID

To operate a certain UIM242xx controller, user needs to first tell the UIM25001 controller which UIM242xx is going to receive instructions. Therefore, before sending a motor control related instructions, user needs to use the following instruction to specify the UIM242xx controller's ID.

Function	Specify the operation object by its ID		
Syntax	**ADR = ID;**	Variable	Integer ID = 5、6、7 … 125
ACK Message	**0xAA，ID，0xD0，0，0，0，0，0，0，0，0，0xFF**		
Comment	ADR is the abbreviation of address. Once an operation object is specified, all object specific instructions will be sent to this specified object (the UIM242xx Controller), UNTILL another operation object (new ID) is specified.		

Set Output Current

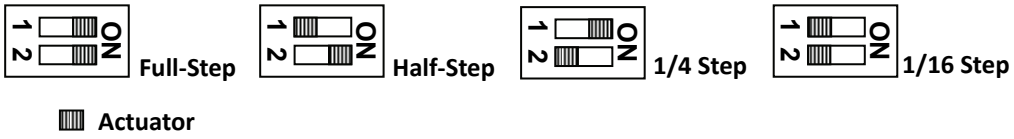
Function	Set the value of output current		
Syntax	**CUR = x;**	Variable	Integer x = 0、1 … 80
ACK Message	Refer to the "General ACK Message" section in this document.		
Comment	CUR instruction is used to set actual output current. It is executed in real-time. Integer 0 ... 80 represent 0 ... 8.0 amps. Once received, the current value is stored in the controller's EEPROM. If the received current value is not one of the above integers, an Error ACK will be sent to the user device through RS232. Incorrect instructions will be discarded without execution.		

Automatic Current Reduction

Function	Enable/disable ACR (automatic current reduction) function		
Syntax	**ACR = x;**	Variable	Integer x = 0、1(or nonzero number)
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	To reduce the power consumption and temperature raise, UIM242xx controller has the build-in function to reduce output current by half, 0.5 seconds after the motor stopped. Caution has to be taken before enable this function, since current reduction also means holding torque reduction. If ACR = 1; the function is enabled, vice versa. ACR will be automatically set to 0, after the CUR instruction is received, to avoid potential confliction.		

Set Micro Step Resolution

UIM240xx controller can provide complete micro-stepping control at full-step, half-step, quarter-step, and sixteenth-step resolutions. Only during power up, the DIP setting is read as the initial value of micro step resolution. After that, the DIP value has no effect. The micro step resolution need be changed through following instruction.



Function	Set/change micro step resolution		
Syntax	**MCS = x;**	Variable	Integer x = 1,2,4,16
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	x = 1, 2, 4, 16 represents the full-step, half-step, quarter-step, and sixteenth-step resolution, respectively.		

Set Desired Motor Direction

Function	Set the desired motor direction		
Syntax	DIR = x;	Variable	Integer x = 0、1(or nonzero number)
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	DIR =0; and DIR =1; only denotes two opposite turning direction. Actual motor turning direction also depends on the physical wiring situation. For example, swapping the wiring between A+ and A- , or B + and B-, can lead to an opposite turning direction.		

Set Desired Motor Speed

Function	Set the desired speed		
Syntax	**SPD = x;**	Variable	Integer x = 0,1,2… 65535
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	Speed is defined as how many steps per second, PPS (Pulses per Second) or Hz. x = 0,1 ... 65535 represents 0,1 ... 65535 pulses / sec.		
Example	For a 1.8° stepper motor, if the SPD =100; then if MCS = 1; motor speed = 1.8*100 = 180°/sec = 30 rpm if MCS =16; motor speed = 1.8*100/16 = 11.25°/ s = 1.875rpm		

Set Desired Relative Displacement

Function	Set the desired steps or micro steps (if MCS≠1) beyond current position		
Syntax	**STP = x;**	Variable	Integer x = 0,1 ... 2147483647
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	2147483647 is the maximum value of a signed 32-bit number. If an instruction of “STP=0;” is received before the previous STP instruction is fulfilled, the motor will stop rotating (i.e. SPD is reset to 0), and previous STP instruction is considered fulfilled. If the STP instruction is received when the motor is running, the steps the motor will run before stopping is counted when the STP instruction is executed.		
Example	For a 1.8° stepper motor, if STP =200; then, if MCS = 1, motor rotation angle = 1.8 * 200 = 360° if MCS = 16, motor rotation angle = 1.8 * 200 / 16 = 22.5°		

Enable the Motor Driver

Function	Enable the stepper motor driver (i.e. H-bridge driving circuit)		
Syntax	ENABLE;	Variable	N/A
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	ENABLE instruction turns on the dual H-bridge motor driving circuit.		

Disable the Motor Driver

Function	Disable the stepper motor driver (i.e. H-bridge driving circuit)		
Syntax	OFFLINE;	Variable	N/A
ACK Message	Refer to the “General ACK Message” section in this document.		
Comment	OFFLINE instruction turns off the dual H-bridge motor driving circuit. Once an OFFLINE instruction is executed, the motor has no power supply, the power consumption is cut to minimum (the logic circuit is still working). User needs to use the ENABLE instruction to turn the motor driver back to working.		

General ACK (acknowledgement) Message

After receiving an instruction, the controller will immediately return an ACK message of all desired settings including the most current setting. Specifically, the following information is included in the ACK message: STP, SPD, DIR, MCS, CUR, ENABLE/OFFLINE, and ACR.

Error ACK Message

0xEE, Error Code, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0xFF

0xEE denotes an error message.

When the received instruction is incorrect, the UIM25001 controller will issue the above message. There are two kinds of errors: Syntax error and value error (the variable range is incorrect). The corresponding error code is list below.

Error Code	0x3D	0x3E
Meaning	Syntax Error	Value Error

Normal ACK Messages

When a valid instruction is received, the UIM25001 controller will send back a normal ACK message. The normal ACK message is 13 bytes long and has a structure shown below.

0xAA, ID, ASM byte, CUR, STP4, STP3, STP2, STP1, STP0, SPD2, SPD1, SPD0, 0xFF

0xAA denotes a normal ACK message.

ID is the identification code of the controller which sends back this ACK message.

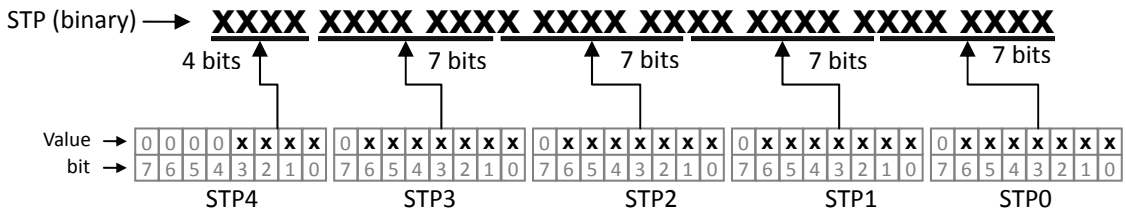
ASM (assembled) byte structure:

Value	N/A Read 0	ACR	Enable=1 Offline=0	DIR	MCS-1 (0 = full step, 15 = 1/16 step)			
Bit	7	6	5	4	3	2	1	0

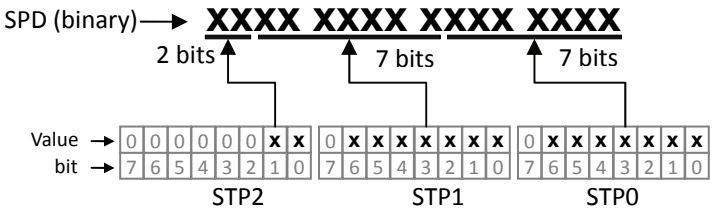
CUR (desired output current) structure: (xxx xxxx is the binary form of the current value)

Value	**0**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
Bit	7	6	5	4	3	2	1	0

Desired displacement structure:



Desired speed structure:



It is easy to use bit shifting operations to obtain the STP and SPD value. There are detailed examples of above bit operation in the source code of the demo control software. Those software and related source code are VC++/VB based and are free. User can go to UI Robot Co. website to download the free copy.

Motor Status Feedback Message

Motor's current working status can be obtained by following instruction:

FBK;

Once receives the FBK instruction, the controller will send back the feedback message comprising following up-to-date information: steps, speed, direction, micro step resolution, current, enabled/offline status and ACR status. The feedback Message is 13 bytes long, and has the following format:

0xCC, ID, ASM bytes, current, STP4, STP3, STP2, STP1, STP0, SPD2, SPD1, SPD0, 0xFF

0xCC means the feedback message, and values in the message relate to the motor status.

ID is the identification code of the controller which sends back this message.

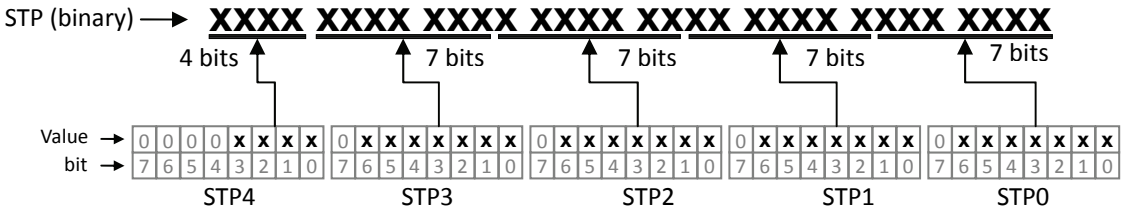
ASM (assembled) Byte structure:

Value	N/A Read 0	ACR	Enable=1 Offline=0	DIR	MCS-1 (0 = full step, 15 = 1/16 step)			
Bit	7	6	5	4	3	2	1	0

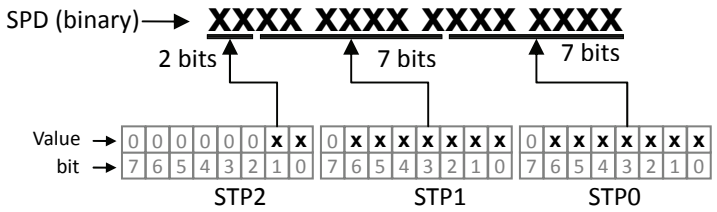
Current (output current) structure: (xxx xxxx is the binary form of the output current)

Value	**0**	**x**	**x**	**x**	**x**	**x**	**x**	**x**
Bit	7	6	5	4	3	2	1	0

Displacement structure:



Speed structure:



It is very easy to use bit shifting operations to obtain the STP and SPD value. There are detailed examples of above bit operation in the source code of the demo control software. Those software and related source code are VC++/VB based and are free. User can go to UI Robot Co. website to download the free copy.

Global Operation

Besides the Object specific operation, UIM25001 controller can also send commands to all subsidiary UIM242xx controllers.
In some case, user may need to send instructions to all UIM242xx controllers such as emergency stop. UIM25001 controller accepts a set of global instructions as described below.

Global Registration

Function	Record the total number of all subsidiary UIM242xx controllers and their ID		
Syntax	**gREG;**	Variable	N/A
ACK Message	**0xAA, QTY, 0xD1,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of all UIM242xx controllers that are operable. gREGis the abbreviation of Global Register. Once this instruction is executed by the UIM25001 Controller, the total number of all subsidiary UIM242xx controllers and their ID are recorded in the UIM25001 controller. The returned QTY server one indicator of the health of the network. In addition, if user specifies an invalid ID, the UIM25001 controller can notify the error by comparing to the valid ID list.		

Global Auto Current Reduce (ACR)

Function	Enable/disable the ACR function of all subsidiary UIM242xx Controllers		
Syntax	**gACR = x;**	Variable	Integer x = 0，1(or nonzero number)
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction. gACR = 1；enables the ACR function, vice versa。If user issues a CUR instruction on a certain controller, the ACR function on that controller will be automatically disabled.		

Global Speed

Function	Set the desired speed to all UIM242xx controllers.		
Syntax	**gSPD = x;**	Variable	Integer x = 0、1 … 65535
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction.		

Global Direction

Function	Set the desired direction to all UIM242xx controllers.		
Syntax	**gDIR = x;**	Variable	Integer x = 0， 1(or nonzero number)
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction.		

Global Relative Displacement

Function	Set the desired Relative Displacement to all UIM242xx controllers.		
Syntax	**gSTP = x;**	Variable	Integer x = 0、1 … 2147483647
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction.		

Global Enable

Function	Enable the stepper motor drivers of all UIM242xx Controller		
Syntax	**gENABLE；**	Variable	N/A
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction. User can first set the desired parameters to every UIM242xx Controller one by one, and then issue this instruction to enable all the motors' motion.		

Global Disable

Function	Disable the stepper motor drivers of all UIM242xx Controller		
Syntax	**gOFFLINE；**	Variable	N/A
ACK Message	**0xAA, QTY, 0xAD,0,0,0,0,0,0,0,0,0,0xFF**		
Comment	**QTY** is the quantity of UIM242xx controllers that receives this instruction. User can shutdown all motors using this instruction, especially in emergency.		

Dimension

Unit：mm