RMV856
Stepping Motor Controller IC

User's Guide

Contents

1- Introduction

The RMV856 is a high-performance stepper motor controller IC that provides all the tools for performing sophisticated motion control, in a quick and easy way. An asynchronous serial port provides the control link between the microcontroller or host computer and the IC.

A refined algorithm provides stepping rates from 16 to 8500 steps/sec. with linear acceleration-deceleration and all the necessary functions for executing smooth and complex motion profiles. External control lines make the stepper controller to react according to external events such as limit switches, home sensors and shaft position encoders.

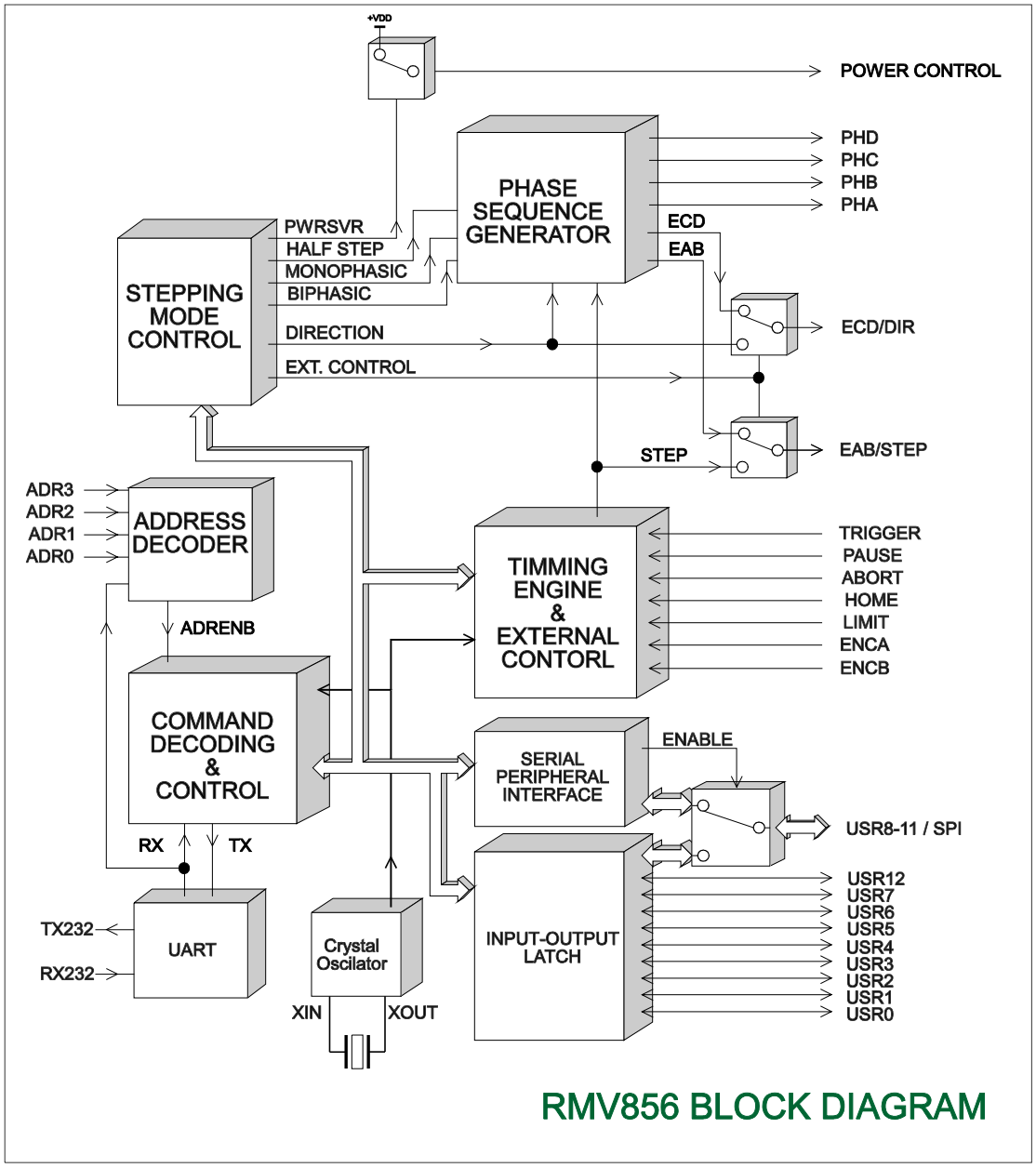
Three stepping modes are available in order to driving the phase outputs. The device can also operate in external driver mode, providing signals for interfacing with standard high-power drivers. A power save control feature allows stand by power to be reduced while the motor is idle for a given period of time.

A remarkable feature, the velocity profiling mode, gives the user the freedom to program any kind of ramping or speed shaping using the internal 128 bytes FIFO.

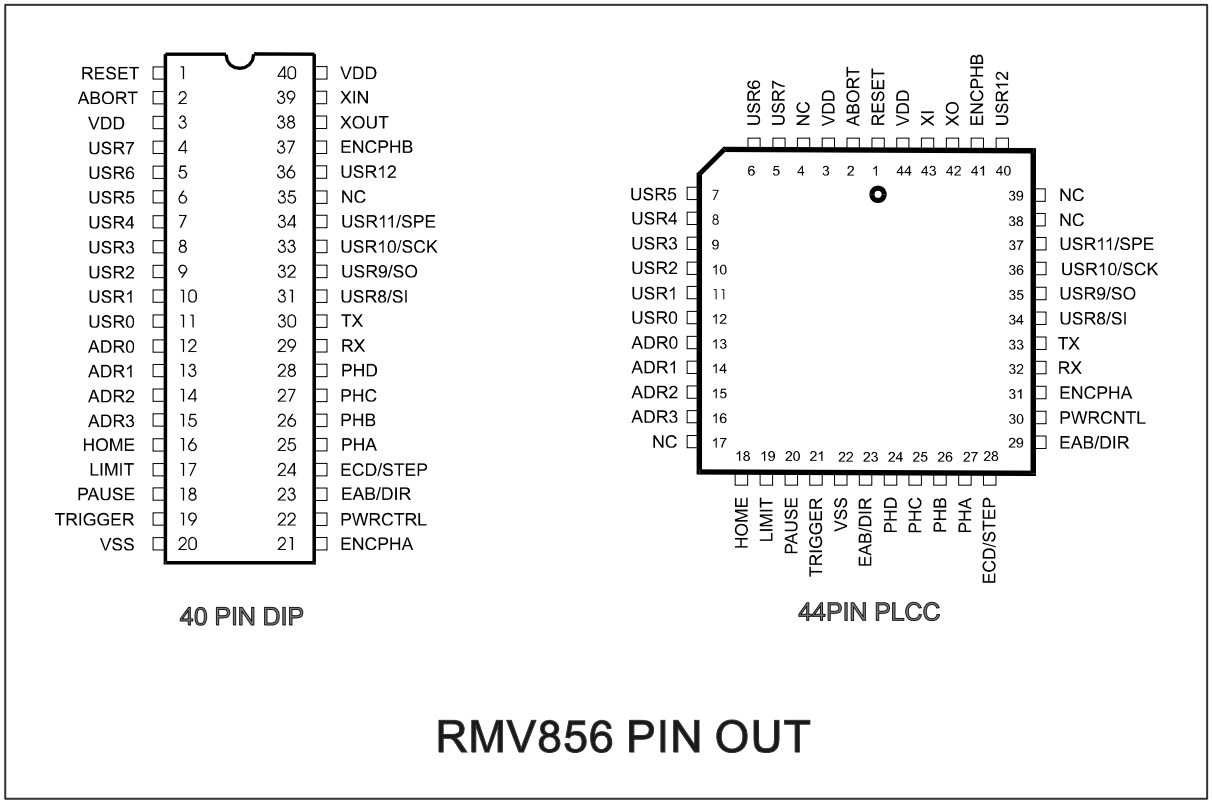
The RMV856 has been conceived as peripheral IC which can be connected as any one of the possible 16 controllers of a multi-motor system, which allows them to operate using a RS232 or RS485 link.

Thirteen general-purpose bi-directional digital lines provide data handshaking and control of external devices. An SPI (Serial Peripheral Interface) port is also available for controlling 3 or 2 wire devices such as Analog to Digital Converters (ADC's) or Digital to Analog Converters (DAC's). This interface can operate either as 8 or 16 bit width mode, and the clock speed is also selectable between 115.2 Kbits/sec and 1.8432 Mbits/sec.

2- Block Diagram



3- Pin Out Function Description



Pin Number	Pin Name	Description
1	RESET	A low level on this pin, will make the controller to erase all parameters and reinitialize.
2	ABORT	A negative edge signal on this input will make the RMV856 to stop the motor at once and kill the contents of the FIFO.
22	VSS	0 V power-supply connection.
3,44	VDD	+5 V power supply connection.
27	PHA	Phase #1 control output
26	PHB	Phase #2 control output
25	PHC	Phase #3 control output
24	PHD	Phase #4 control output
28	ECD/STEP	In normal mode, controls the H bridge enable signal for winding AB (for bipolar drivers). In External Driver mode, outputs a high pulse every time a step is taken.
29	EAB/DIR	In normal mode identical function as ECD/STEP but for winding CD. In External Driver mode, it goes high for CW direction or low for CCW.
30	PWRCNTL	If powersave is enabled it goes low after the powersave timer has expired.
21	TRIGGER	Controller will wait for a low level on this pin in order to start the motion.
20	PAUSE	While this pin is at low level, steps are not taken and the controller holds the position.
19	LIMIT	A low signal is received on this pin, the motor will stop at once and either the CW limit or CCW limit flag will be set, depending on the direction the motor was stepping.
18	HOME	If Seek Home feature is enabled, when this signal goes high, the motor will stop at once and a flag will be set.

Pin Out Description (Continued)

Pin Number	Pin Name	Description
16	ADDR3	Address selection input MSB
15	ADDR2	Address selection input #2
14	ADDR1	Address selection input #1
13	ADDR0	Address selection input LSB
40	USR12	User selectable I/O (MSB bit)
37	USR11	User selectable I/O - SPI Enable
36	USR10	User selectable I/O - SPI Clock
35	USR9	User selectable I/O - SPI Data Output
34	USR8	User selectable I/O - SPI Data Input
5	USR7	User selectable I/O
6	USR6	User selectable I/O
7	USR5	User selectable I/O
8	USR4	User selectable I/O
9	USR3	User selectable I/O
10	USR2	User selectable I/O
11	USR1	User selectable I/O
12	USR0	User selectable I/O (LSB bit)
33	TX	RS232 data output pin (TTL level: Mark =0 V Space =5 V)
32	RX	RS232 data input pin (TTL level)
31	ENCPHA	Quadrature Encoder channel A, direction
41	ENCPHB	Quadrature Encoder channel B, clock

4 - Interfacing and operating modes

The RMV856 interfaces with any device able to handshake TTL asynchronous signals from 9600 to 115200 bauds (some limitations apply over 38400 bauds). The handshaking is done using 8 bits, none parity, 1 stop bit. Four addressing line inputs can accommodate up to 16 controllers on the same serial port. In most applications, the easiest way of controlling motors is just adding a RS232 adapter IC such as charge pump type (e.g. : MAX232) or standard drivers and receivers (e.g. : 1488 & 1489). Another possibility is to use a RS485 adapter IC so that the RMV856 becomes part of a network and can take advantage of the opto-isolation and extended operating distance found on the RS485 networks.

5- Operating Modes

Operating mode refers to the way commands are executed by the controller. There are three operating modes:

1. Live
2. Queued Commands (FIFO operation)
3. Program Execution

Live Mode: In this mode commands are received and executed at once. If there is any constraint in order to execute the command (e.g. the motor is not ready and the user wants to start a new motion profile), error messages are generated indicating the situation.

Queued: in this mode commands are stored in circular FIFO buffer for sequential execution. Buffer size is monitored and an error message is generated when the buffer has over-flown. The host controller is able to read at any moment the remaining FIFO bytes as well as the current value of the program counter (which is incremented with every new command fetched from the program list).

Program Execution: during program execution, a stored program (any combination of motor related or user I/O commands plus some flow control commands, if necessary) is executed. Program termination may happen either by normal termination (all stored commands have executed) or upon receiving a kill-FIFO command. During

program execution the controller is able to execute some commands as long as they do not interfere with the normal flow of the program. Program list requires to be downloaded first while in Live Mode.

Mode 1 (Live) is the default mode of operation.

Mode 2 is entered at any time, with the exception of Mode 2 running. Commands to be queued must have the '&' character following the Address Mask and Address Target characters. (E.g. FA&W128). While commands are being executed, the user is free to send further commands for execution, though restrictions apply regarding the motion control feature.

Mode 3 is entered upon the storage if the program list in the program memory buffer and is instructed by the X command (eXecute program). A program can be aborted or terminated in two ways: by using the normal flow control sequence of the program list (normal termination) or by issuing a Q (Quit) command.

6-Command Description

There are two sets of commands:

1. commands involved in the motion control process
2. commands related to the user I/O and miscellaneous features of the RMV856

Commands are sent using ASCII characters. Every command can be seen as a two-part string. The 1st part is the Prefix and handles the addressing of the network. The 2nd part is made of the command character follow by the associated parameter (if any required for that command).

COMMAND STRING STRUCTURE

Address Prefix	Command Identifier	Parameter	ASCII 13 Carrier Return

6-1 Addressing Formats

The Prefix sets the addressing mode.

6-1-1 Single Addressing

The single address identifier (ASCII 95) marks the beginning of a single controller addressing operation, after which the address character (0 to F) is sent. For instance, if the host wants to target the controller which address is 7, the prefix is ASCII 95 followed by ASCII 55 (_7).

6-1-2 Broadcasting Operation

Another possibility is to send a broadcast message. In this case the prefix can be calculated by using table I.

	Controller Address															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Add to Prefix	32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

TABLE I- Prefix Value

Example: the host wants to address controllers #1, #2 and #9. The Prefix value is calculated as:

2+4+512 = 518 or 0106 hexadecimal, i.e. the host needs to send 0106 as the prefix.

In this way a set of controllers or the whole network can be addressed at once when sending a command. Acknowledgment will be sent by the controller which address is the lowest of the controllers being addressed. In the example above, that would be controller #1.

In the following description, prefix symbols are not shown but must included in order to satisfy the command string headlines.

6-2 Motion Related Functions

6-2-1 Motor Configuration Register

This is an 8-bit register, which controls several features on the RMV856. Table IV shows the meaning of each bit on the Motor Configuration Register

6-2-1-1 Stepping Mode

Bits 0,1,2 and 3 control the behavior of signals on the PHA-D outputs, EAB and ECD.
In Full Step or Bi-phasic, two windings are always on. In Mono-phasic mode, only one winding at a time is on. Half Step mode makes the motor to step using alternatively one winding on and two winding on.

6-2-1-2 External Driver Mode

External driver mode (bit #3 set), uses the EAB and ECD (Enable Windings signals) to generate the following two signals:

DIR: this output goes to high logic state for Clockwise direction, low state for Counterclockwise.
STEP: A high pulse 12 µs long is sent every time a step is taken.

MOTOR CONFIGURATION REGISTER		
BIT	NAME	FUNCTION
0	HALF STEP	set for half step driving
1	FULL STEP	set for full step driving
2	MONOPHASIC	set for monophasic driving
3	EXTERNAL DRIVER	set for external driver operation
4	STALL DECTECT	set for Encoder feedback operation
5	POWER SAVE	set for Power Saving operation
6	PROFILE	set when velocity profile mode on
	SEEK HOME	set for seeking home signal

TABLE II- Motor Configuration Register

	EXTERNAL DRIVER SIGNALS	
MODE	Signal Name	Function
Phase Outputs	EAB	Enable Winding AB
	ECD	Enable Winding CD
External Driver	DIR	Direction Control Output
	STEP	Step Output

TABLE III - External Driver signals

6-2-1-3 Stall Detection

When this feature is enabled (bit #4 set), the controller will watch up for a valid position change read from the shaft encoder inputs. If there is no confirmation, the motor will stop at once and a corresponding flag on the Motor Configuration Register will be set (motor Stalled)

6-2-1-4 Power Saving Mode

When enabled it waits during the Power Saving Time and then it sets the MOTORPWR output to low level, which if used for controlling the operating current, can be used to decrease the winding current while the motor is idle. The Power Saving time is set to 1 second by default, but can be changed to any value up to 10 seconds by the Motor Power command

6-2-1-5 Velocity Profile

When enabled, it instructs the controller to consider the SLOPE parameter as a 2's complement number. By this mean, the speed can be increased or decreased linearly during a given number of steps. This also means the speed profile is no longer going to be trapezoidal as it is when this feature is disabled but, dependent on the FIRST RATE, SLOPE and how many steps to go. If the SLOPE is set to a number between 0 and 127 the rate is going to change linearly according to the following equation:

$R_n=F+(n-1)*S'$ n = 1,2… Steps to GO S'=SLOPE

So if for example the controller is set to run 500 steps, the initial rate F=1000 and the Slope=1 the Rate at the 24th step will be:

R_{24} =100+(24-1)*2 = 1046 steps/sec.

and the final rate, before the motor stops will be

R_{500}=1000+(500-1)*2 = 1998 steps/sec

If the Slope is set to a value higher than 127 then effective slope is computed as S'=256-Slope

So if in the above example the Slope=255 the Rate at the 24th step will be:

R_{24} =1000+(24-1)*(-1) = 977 steps/sec.

and the final rate, before the motor stops will be

R_{500}=1000+(500-1)*(-1) = 501 steps/sec

Also is important to note that when this feature is enabled, the last rate calculated rate (R_{500} in the example), is copied into the FIRST RATE when the motion is completed. In this way the new starting point can easily be determined by equation #1

6-2-1-6 Home Seeking
When enabled, this feature will make the controller to watch out for a high level on the HOME input. When received, this makes the motor to stop at once, and to set the HOME flag on the MOTORSTATUS register.

6-2-2 Motion Profile Definition

Initial Rate: sets the initial rate in steps/sec. This is the rate at which the motor will begin stepping. Together with the Slope and Slew rate parameters, they define the trapezoidal motion profile. The syntax is Ff, where f represents an integer between 16 and 8500.

Slew Rate: sets the slew rate in steps/sec. This is the rate the motor will attempt to reach when depending on the acceleration, steps to go and initial rate parameters. Syntax: Rr, where r represents an integer between 16 and 8500.

Acceleration-Deceleration or Slope rate: controls the speed increase or decrease in order to accelerate or decelerate the load. Expressed as steps / sec.2. Syntax: Ss, where s represents an integer between 0 and 255. See Velocity Profile mode for further details on this parameter

Prescalling Factor: sets the prescaller value. This command makes the actual stepping rate to be divided by the prescalling value. Syntax: /p, where p is an integer between 1 and 255. Since this prescalling is in effect step by step, the Slope parameter also is affected.

Go to Target Position: Steps to match the position specified as 23 bit plus sign number. Syntax: T p, where p represents an integer between –8,388,607 and 8,388,607.

Go Steps Relative: Steps a specified number of steps from current position according to the entered parameter. The sign of the parameter determines the direction: (+) Steps CW, (-) Steps CCW. Syntax: N p, where p represents an integer between –8,388,607 and 8,388,607.

Halt Motor: Stops the motor at once. Syntax: H. This also clears the entire contents of the FIFO memory.

Jog motor: jogs the motor in the specified direction (by the “+” or “-“ commands) using the initial rate parameter. Syntax: J

Change Jog Speed: Changes the rate on the fly while the motor is jogging. Syntax: Yy where y is an integer between 16 and 8500.

6-3 External Hardware Control
Special signals are dedicated to sensing and controlling external devices in order to coordinate motion with another controllers or to get some sort of feedback from the motor that is being controlled.

6-3-1 Trigger Input This is a low-level active signal that is checked every time the motor is going to start the motion. If high, the motion will not start until this signal changes to low level.

6-3-2 Abort Input and halt command Abort is a negative edge and low level active signal that causes the controller to stop the motion at once. The entire content of the FIFO is cleared and the abort flag on the MOTORSTATUS register is set. Motion will not start if this signal is low; an error message will be generated. Another way of stopping the motion at any time is using the Halt command. The syntax is H. This command will stop motion at once, set corresponding flags and kills all contents on the FIFO or programming memory.

6-3-3 Pause Input This is a low-level active signal. While this signal is low, the motor will take no further steps. A corresponding flag on the MOTORSTATUS register is set.

6-3-4 Limit Input This is a low-level active signal. When this signal changes to low, the motion will be terminated and a corresponding flag on the MOTORSTATUS register will be set. Motion will not start if this signal is low; an error message will be generated.

6-3-5 Home Input If the seek home feature has been enabled on the MOTORCONFIG register, the controller will watch out for a high level on this pin. If present, will stop the motion at once and set a flag on the MOTORSTATUS register.

6-3-6 Shaft Encoder Inputs These inputs are available for connecting a 2-channel quadrature shaft encoder. Four-time decoding (4 edges per valid position change) is used for calculating the position. False decoding protection is also supported, which is very useful when the encoder is working attached to a shaft experiencing resonance problems.

6-4 User I/O Control

A total of 13 TTL bi-directional digital lines are available on each controller. USR11,10,9 and 8 are shared with the SPI interface, which takes control of them when it is enabled.

6-4-1 Configuring the direction
Any bit can be configured either as input or output using the Configure User I/O command. Syntax C*c*, where *c* is ant integer between 0 and 8191. Use table IX in order to calculate *c*.

	User bit												
	12	11	10	9	8	7	6	5	4	3	2	1	0
Add to *c* for bit direction = output	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

TABLE IX- User bit direction

Example: if user bits #4 and #8 are required to be set as output on controller address #0, then c=16+256 = 272, an the command will be _0C272[CR].

6-4-2 Writing data to the User I/O
This function works in the same way as above. In order to set a bit the value must be calculated using the table above mentioned. Syntax Ww, where w is an integer between 0
and 8191.
Another way of writing to the user I/O is possible by using the Conditional Writing command . This command writes a byte of data to the first 8-bit of the I/O port (USR7 to USR0),
using the following syntax:

Prefix+ASCII(58)+ASCII(36)+X3X2X1X0

X3X2X1X0 represents a 32-bit coded hexadecimal number, and Xx a 2-digit hexadecimal figure which will be written to the USR7-0 according to the following table:

Controller Address	Value to be written
15 to 12	X3
11 to 8	X2
7 to 4	X1
3 to 0	X0

6-4-3 Reading data from the User I/O
This function returns the state read from those user pins configured as input, and the state of the output latch of those configured as outputs. Syntax B. Returns and integer between 0 and 8191.

6-5 Serial Peripheral Interface
This interface handles synchronous serial communications with devices using SPI ports such as ADC's or DAC's.

6-5-1 Configuring the SPI
SPI can be configured using the SPI configure command. The syntax is:

Oo, where o is an integer between 0 an 255.
SPI clock speed can be selected as shown on table VI

6-5-2 Writing data to the SPI
Data can be written to the SPI using either 8 or 16 bits format, using the selected polarity, phase and clock speed as described in the configuration command. Data to be sent is copied into two 8 bit registers, SPI_HI (MSB) and SPI_LO. When 16 bit is selected the MSB byte is sent first. Syntax Vv, v represents an integer between 0 and 65535.

6-5-3 Reading data from the SPI
Reading from the SPI is accomplished by shifting out the value at SPI_HI and SPI_LO register (in that order). If 8 bit (WSIZE bit set to 0) mode has been chosen, only SPI_LO is shifted out. Eight or sixteen data bits are read using the settings mandated by the configuration word. Syntax U.

SPI Configuration Register		
bit	Name	Function
0	SPR1	SPI clock speed (LSB)
1	SPR0	SPI clock speed (MSB)
2	CPHA	SPI clock phase
3	CPOL	SPI clock polarity
4	Unused	
5	Unused	
6	WSIZE	SPI data word size 1:16 bits, 0:8 bits
7	ENABLED	SPI enable control

TABLE V-SPI Configuration Register

SPI Clock Speed (MHz)	SPR1	SPR0
1.8432	1	1
0.9216	1	0
0.4608	0	1
.1152	0	0

TABLE VI - SPI Clock Speed

RMV856 pin	ADC pin
USR11 (SPI Enable)	+5 VDC
USR9 (SPI data output)	Data Input
USR8 (SPI data input)	Data Output
USR10 (SPI serial clock)	Clock
USRx	Chip Select

TABLE VII - ADC connection

ADC resolution (bits)	Add to parameter
8	16
10	32
12	64

TABLE VIII - ADC reading parameters

6-5-4 External Digital to Analog Converter
Analog to Digital Converters (ADC's) can be hooked up to the SPI port using a 4-wire interface connection. The fourth wire, Chip Select line, has to be provided by any of the remaining User I/O lines and the host is in command of controlling the state of this line as required by the target ADC chip. Syntax for this command is Zz, with z representing an integer from 16 to 75. Table VIII shows how this parameter is calculated:

In order to specify a channel, add the channel number to the parameter z. For example for reading channel 11 (half scale channel) on a industry standard 10-bit ADC the parameter z will be: z=32+11=43

So the command to send to the RMV856 is Z43 which will return 511 (half scale). The channel number ranges from 0 to 13 for 12 and 10 bits ADC's and from 0 to 11 for 8-bit ADC's.

6-6 Programming and FIFO operation

FIFO operation is entered upon receiving the FIFO character (& or ASCII 38) in the prefix string. For example the sequence :

_0&N100

_0&W128

tells controller address 0, to take 100 steps in the CW direction, and when motion is completed to write 128 to user data port. FIFO operation is irrelevant and will have no effect in configuration command or commands that involve any data request or inquiry. This means that the controller being addressed by this command will executed the command at once without altering the FIFO stack.

Using this feature the host can send commands without having to wait for the controller to complete a motion. Besides this provide a way of executing customized acceleration profiles by executing a series of small segments that contour a given profile. This allows a way of interpolating speed and position for composing multi-dimensional motion.

6-6-1 Entering and Quitting Programming mode

Another feature is program load and execution. This works instructing the controller to enter into programming mode, after which a list of commands is loaded sequentially. When a command is entered to the program list, the controller will answer telling the remaining free memory into the program memory (a total of 128 bytes). This also applies to FIFO operation. In this way the host can foresee the free memory available and waits until further memory is freed in the FIFO operation mode. The syntax for this command is E.

When the complete program list has been entered the controller should be instruct to quit program-entering mode and return to normal mode. The syntax for this command is Q.

The program list remains in program memory, until the controller receives the "eXecute program" command. Then, program execution begins and will continue until normal termination or the controller is instructed to kill the programming memory. This later operation will reset the FIFO memory pointers and erase all its contents. The syntax is K.

6-6-2 Executing Programs

Host starts the entered program execution using the "eXecute program" command. Syntax X.

6-6-3 Program Loops

Program list can be terminated using a loop to the begin command. This makes the program to repeat until the kill command is receiving. Syntax L.

6-6-4 Killing FIFO and stopping programs

This command will stop any program being executed or erase all commands pending for execution on the FIFO. Syntax K.

6-6-5 FIFO reports

FIFO available memory can be requested at any time using this command. Syntax [.

7 Reports

Report is a special feature, which allows the host to retrieve a previously entered value into some configuration register or motion parameters. For example "_0R?", will instruct the controller address 0 to send back the current Slew rate value. The following commands accept a report request

R, F,S, C, O, M.

7-1 Shaft Position Register Reading and Setting

Shaft position can be read at any time by using this command. Syntax P. Returns a long number between 0 and 16,777,215. In order to set the register the syntax is Aa, where a is a long number between 0 and 16,777,215 or between -8,388,607 to 8,388,607.

7-2 Shaft Encoder Register Reading and Setting

Shaft encoder register can be read or set any time. For reading, the syntax is {. It returns a long number between 0 and 16,777,215. To set the register to a specific value, the syntax is }p, where p is a long number between 0 and 16,777,215 or between -8,388,607 to 8,388,607.

7-3 Reading back the current Motion Parameter

Using the report feature, the complete set of motion parameters can be read back by the host :
First rate : e.g. "_0F?"
Slew rate: e.g. "_0R?"
Acceleration-Deceleration rate: e.g. "_0S?"
Motor Configuration Register: e.g. "_0M?"

7-4 Motor Status Register

The status of the controller can be read at any time by using the Get Motor Status Register command ("=",ASCII 61). It returns a number between 0 and 2047.

For example : _0= returns the status register value of controller #0.

For a complete reference, see table X.

TABLE X – Motor Status Register

Bit	Name	Status (set when…)
0	TRGD	Motor has been triggered (TRIGGER input has been LOW).
1	PSED	Motor is being paused (PAUSE input is LOW).
2	ABRTD	Motion has been aborted (ABORT input was set LOW or HALT command has been received from the host).
3	RUNING	Motor Is running (at least one step or half step has been taken).
4	CCWL	Limit switch has been activated while motor was moving CCW.
5	CWL	Limit switch has been activated while motor was moving CW.
6	HOME	Motor has reached home position (HOME input has been HIGH).
7	READY	Motion has been completed.
8	DIR	Direction flag: set to ONE when moving CW.
9	STALL	Missing Quadrature Encoder pulse (motor is stalled).
10	ENCERR	No use, for diagnostic only.

8- Error Messages

Under certain circumstances errors messages may be received by the host computer when sending commands to the controllers. Table XI show this error messages. The syntax used to send this messages is as follows:

Error Name	Number	Meaning
Parameter Out Of Range	1	Parameter is out of range.
Command Not Found	2	Command ID not found.
FIFO Buffer Full	3	FIFO is full and can not store any more commands.
SPI not Configured	4	An attempt to read or write to the SPI has been made without being configured first.
SPI not Master	5	USR11/SPE input is low while trying to read or write to SPI.
Motor is Running	6	Can not execute the received command while stepping.
Motor is Stopped	7	Can not start moving: Abort or Limit inputs in active state
Set Direction Forbidden	8	An attempt to change direction has been made while controller was stepping and not in jog mode.
Can Not Execute	9	Can not execute command sent while in the current status.
No Hexadecimal Value	10	The “$” character was expected, or the value sent was not an Hexadecimal number
Wrong Direction	11	The direction is the same before , a limit switch was activated

TABLE XI – Error Messages

ASCII(63)+Error Number+ASCII(62)

For example attempting to configure controller #0 with a first rate set to 10 step/sec (_0F10) will result in the reception of the following error message string: ?1>

9- Command Summary

Table XII shows a summary for all commands available on the RMV856, the ASCII commands are **case sensitive** .

TABLE XII - RMV856 COMMANDS WITH ONE PARAMETER

Command	Description	Parameter	Returned Value	FIFO Operation	Multiple Addressing
A	SET POSITION	-8,388,607 TO 8,388,607	--	N	Y
B	IO_READ	---	0 TO 8191	N	N
C	IO_CONFIG	0 TO 8191	--	N	Y
D	WAIT_PROG	0 TO 65535	--	Y	Y
E	ENTER_PROG	--	--	N	Y
F	FIRST_RATE	16 TO 8500	--	Y	Y
G	REPEAT_MOVE	---		Y	Y
H	HALT_MOTOR	---		N	Y
I	SOFT_RESET	---		N	Y
J	JOG_MOTOR	---		N	Y
L	LOOP_PROG	---		Y	N
K	KILL_FIFO	---		N	Y
M	STEP_MODE	0 TO 255		N	Y
N	GO_STEPS_REL	-8,388,607 TO 8,388,607		Y	Y
P	POSITION_REQ	---	0 TO 16,388,607	NA	N
Q	QUIT_PROG	---		NA	Y
O	SPI_CONFIG	0 TO 255		N	Y
R	SLEW_RATE	16 TO 8500		Y	Y
S	ACCEL_RATE	0 TO 255		Y	Y
T	GO_ABS_POS	-8,388,607 TO 8,388,607		Y	Y
U	SPI_WRITE	0 TO 65,535		Y	Y
V	SPI_READ	---	0 TO 65,535	N	N
W	IO_WRITE	0 TO 8191		Y	Y
X	EXEC_PROG	---		NA	Y
Y	CHNGE_JOG_SPD	16 TO 8500		N	Y
Z	GET_ADC	16 TO 79		N	N
+	DIR_CW	---		Y	Y
-	DIR_CCW	---		Y	Y
^	BAUD_RATE	0 TO 2 3 to 5 are not recommendable		NA	NA
:	COND_IO_WRITE	0 TO FFFFFFFF		NA	NA
=	MOTOR_STATUS		0 TO 2047	N	N
[	FIFO_INDEX_CTR	---	0 TO 65,535	N	N
]	FIFO_BYTE_FREE	---	0 TO 128	N	N
*	MOTOR_POWER	0 TO 9		N	Y
{	READ_ENCODER	---	O TO 16,388,607	N	N
}	SET_ENCODER	-8,388,607 TO 8,388,607		N	Y

TABLE XIII - RMV856 MULTI PARAMETERS COMMANDS IN HEXADECIMAL FORMAT ONLY

Command	*Description*	*Parameters*	*Returned Value*	*FIFO Operation*	*Multiple Addressing*
i	IO/SPI-CONFIG	1) Byte $ HEXA CHAR. 2)Byte Constant = 07 3)Byte USER[7.0] 0 to FF 4)Value = 0 to FF 5) SPI Config 0 to FF	N	N	Y
c	SET-W-CURRENT	1) Byte "$" HEXA CHAR 2) Byte Channels: 80 ALL Selected 00 Chan. 0 02 Chan 1 04 Chan 2 06 Chan 3 3) Byte Value = 0 to FF	N	N	Y
g	MOTOR-GO	1)Byte "$" HEXA CHAR 2-3) Bytes Initial Speed 4-5) Bytes Final Speed 6) Byte Acceleration CASE A: Positive Movement 7-9) Number of Steps 3 bytes Positive Dir 0 to 7FFFFF CASE B: Negative Movement 7) Byte ASCII "-" 8-10) Number of Steps 3 bytes Dir 0 to 7FFFFF	N	N	Y
~	LIMIT-SWITCH-OFF	1) Byte + or – Character 2) Byte "$" HEXA CHARAC 3) Number Steps 0 to FF	N	N	N

10- Serial Communication Commands.

The following descriptions will show how ASCII commands can be sent to the RMV856 controller, via any terminal software of communication. It is necessary to mention, that all the motion examples are based on the ST400NT Series topology, especially setting the winding current for the stepper motor trough a Digital to Analog Converter with a SPI compatible control, and four channels.
The different motions examples will demonstrate the capabilities of the RMV856 controller for a single axis, and (16) multi axes control.

NOTE1: The following commands examples are based on the ST400NT Series boards.

10-1 Initialization of the master controllers

The master controller address are: “0” , “4”, “8” and “12”. These master controllers in the ST400NT boards are used for controlling the PAUSE and TRIGGER signals trough USR[7..0] for each axis and itself.
In addition an analog to digital converter with 11 channels is connected to the SPI of the master controller.

Master Users IO Bits Description

USR0 -> TRIGGER(0)
USR1 -> TRIGGER(1)
USR2 -> TRIGGER(2)
USR3 -> TRIGGER(3)
USR4 -> PAUSE(0)
USR5 -> PAUSE(1)
USR6 -> PAUSE(2)
USR7 -> PAUSE(3)

USR12 -> Control CS signal from the ADC

10-1-a Commands Initialization

Initial Conditions: USR[7..0] must be All Output
USR[7..0] = FF ; PAUSE and TRIGGER signals are set to one
SPI Init = 80 , configured as master, 2MHz clock Speed

Case A : Single Address

Command: ***_0i$07FFFF80***
Return : >

Case B : Broadcasting (16 axes)

Command: ***1111i$07FFFF80***
Return : >

10-2 Initialization of the Motor Winding Current

The winding current in the ST400NT Board is handle by the RMV856 controllers which addresses are: “1”, “5”,”9”, and “13” respectively.
The winding current is programmed via a Digital Analog Converter of 8 bits resolution and with four buffered channels controlled via a SPI compatible processor.
Before setting the winding current, it is mandatory to initialize all the controllers with the address specified above, in order to set up the Digital IOs, SPI, and Digital to Analog Control signal.

10-2-a Initialization of Controllers Address “1”, “5”, “9”, and “13”

In this case the USR[7..0] will be configured all as Input.
The SPI will be configured as follow:

SPI Value = SPI_ENB + SPI_16BIT + SPI_PHASE + SPI_SPEED3 = 198 = $C6 (HEX)

Case A : Single Address

Command: ***_1i$070000C6***
Return : >

Case B : Broadcasting (16 axes)

Command: ***2222i$070000C6***
Return : >

10-2-b Calculate the value for the DAC

In the ST400NT board a digital to analog converter sets up the winding current for each axis in the ST400NT board. The DAC has buffered channels and can be controlled via a SPI from the RMV856 controller.

A value between 0 and $FF programmed in the DAC is the result of multiplying the Thevenin constant Kt = 0.114 by the winding current desired in milli-Amperes.

Thus the equation for setting up the winding current is as follow:

DAC № = (0.114) * Current Desired in Milli-Amperes

As an example, for a motor in which the Rated Current is 1600 mA , the value to digital to analog converter can be calculated as follow:

DAC №= (0.114) *1600 = 182.40 ≈ 182

DAC № = 182 (DEC) = B6 (HEX);

10-2-c Winding Current setup for each axis

The following table shows the correspondence between RMV856 controller address in which the DAC is connected, and programming the winding current value for each axis. According to the above explanation, the winding current is setup to = $B6

ST400NT Daisy Chain	RMV856 DAC ADDR.	DAC Channel Set	Axis ADDR.	Serial Commands
One ST400NT	"1"	First Chan. = 00	"0"	_1c$00B6
		Second Chan.= 02	"1"	_1c$02B6
		Third Chan. = 04	"2"	_1c$04B6
		Fourth Chan = 06	"3"	_1c$06B6
		ALL Channels = 80	*ALL*	*_1c$80B6*
Two ST400NT	"1" , "5"	First Chan. = 00	"0"	_1c$00B6
		Second Chan.= 02	"1"	_1c$02B6
		Third Chan. = 04	"2"	_1c$04B6
		Fourth Chan = 06	"3"	_1c$06B6
		First Chan. = 00	"4"	_5c$00B6
		Second Chan.= 02	"5"	_5c$02B6
		Third Chan. = 04	"6"	_5c$04B6
		Fourth Chan = 06	"7"	_5c$06B6
		ALL Channels	*ALL*	*0022c$80B6*
Three ST400NT	"1","5","9"	…	…	…
Four ST400NT	"1","5","9","13"	First Chan. = 00	"0"	_1c$00B6
		….	…	….
		First Chan. = 00	"4"	_5c$00B6
		….	…	….
		First Chan. = 00	"8"	_9c$00B6
		….	…	….
		First Chan. = 00	"12"	_13c$00B6
		….	…	….
		ALL Channels	*ALL*	*2222c$80B6*

10-3 TRIGGER and PAUSE in single or multi ST400NT scheme

According to the paragraph described in 10-1, the Trigger and Pause are controlled via the master controllers, in which the address are: “0,4,8,12” respectively.
The purpose is to control both signals with one command, and maintain all 16 axes status in one variable of 32 bits wise.
With the command “:” , “Conditional Write” Trigger and Pause signals for 16 axes can be controlled with this function as follow:

ADDRESS MASTER	Byte# 3 ADDR: C – F	Byte #2 ADDR: 8 - B	Byte #1 ADDR: 4 – 7	Byte #0 ADDR: 0-3
ADDR M[0..3]				Master[0] = Byte#0
ADDR M[4..7]			Master[4] = Byte#1	
ADDR M[8..B]		Master[8] = Byte#2		
ADDR M[C..F]	Master[C] = Byte#3			

The following example shows how the Conditional Write can be used for triggering on or more axis.:

Case A : Axis Address “0” need to be triggered

Command: ***0001:$FFFFFFFE***
Return **:** >

Case B : Broadcasting (16 axes)

Command: ***1111:$F0F0F0F0***
Return : >

10-4 Reading the Stepper Counter Position

When a command "P" has been sent to the RMV856 controller, the value returned will be between 0 to 16,388,607. This value always is positive (Module (Position)), in order to discriminate the host If positive or negative the following calculations need to be done:

Request the stepper counter to the RMV856 controller address "0"

Command : _0P
Return Value : rc = { 0 to 16,388,607}

if (rc > 8388607) then
rc = 8388608 – rc

With the above equation, rc if bigger than 8,388,607 then rc will be negative.
If rc is less than 8,388,607, the counter result will be positive.

10-5 Changing baud rate to the RMV856 Controller

The safest baud rate for the RMV856 controller are : 9600, 19200, and 38400. The following table shows the correspondence between the baud rate and the integer associated to each speed:

0 → 9600
1 → 19200
2 → 38400

In order to change the speed for the serial communication, it is necessary to send the command in broadcasting mode, due to the speed of all RMV856 controllers must be the same, another way one or more controllers will stop the communication with the host due to framing errors.
The following procedures shows how the baud rate can be changed in safe mode:

I) Send the command broadcasting : **FFFF^2** ; Change the baud rate to 38400
II) Make sure that the Prompt has been received "**>**"
III) Change baud rate from Host to **38400**

NOTE: After *"Power ON"* or *"Hardware Reset"*, the default bad rate is = 9600

11 Constants Definition

The following constants are useful to program the RMV856 controller from any serial terminal software.

User Digital IO's

```
#define USR0                    1
#define USR1                    2
#define USR2                    4
#define USR3                    8
#define USR4                    16
#define USR5                    32
#define USR6                    64
#define USR7                    128
#define USR8                    256
#define USR9                    512
#define USR10                   1024
#define USR11                   2048
#define USR12                   4096
```

Analog to Digital converter resolution.

```
#define ADC_12_BIT              0x40
#define ADC_10_BIT              0x20
```

SPI constants.

```
#define SPI_ENB                 128
#define SPI_16BIT               64
#define SPI_POLARITY            8
#define SPI_PHASE               4
#define SPI_SPEED4              3
#define SPI_SPEED3              2
#define SPI_SPEED2              1
#define SPI_SPEED1              0
```

Stepping modes.

```
#define HALF_STEP               1
#define BIPHASIC                2
#define MONOPHASIC              4
#define EXT_DRIVER              8
#define ENCODER_FDBK            16
#define POWER_SAVE              32
#define VEL_PROFILE             64
#define SEEK_HOME               128
```

// Motor direction.

```
#define CKWISE                  1
#define CCKWISE                 0
```

// Status flags.

```
#define TIGGER_ON               1       Set when motor has been triggered
#define PAUSED                  2       Set when motor is being hold (paused)
#define FULL_STOP               4       Set when motor was stopped/aborted by hardware or software
#define RUNNING                 8       Set when motor is moving
```

#define SWITC_CCW	16	Set when CCW limit has been reached
#define SWITCH_CW	32	Set when CW limit has been reached
#define AT_HOME	64	Set when home switch is on
#define MOTION_CMPLT	128	Set when motion has been completed
#define CURRENT_DIR	256	"1" set for direction for CW, "0" set for direction for CCW
#define MOTOR_STALLED	512	"1" set when position error is bigger than two steps or the value programmed.

Some other definitions.

```
#define STALL_DETECT        16
#define SEEK_HOME           128
```