

SS-MST67 Manual

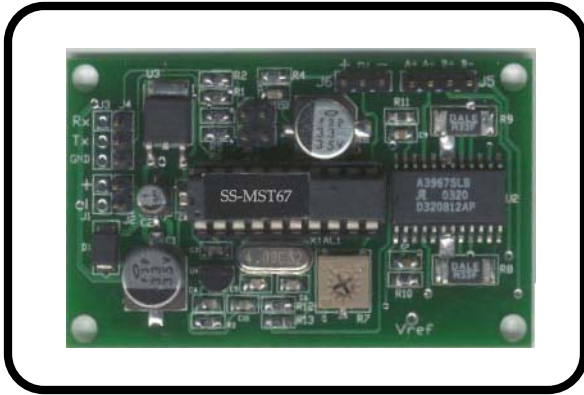
A Super Stepper Command Bus Architecture Product



For a Quick Start Tutorials, visit our web pages at www.superstepper.com



Stepper Controller – Micro Stepping Controller SS-MST67



Super Stepper Micro Stepping Controller (SS-MST67):
 Controls 1 stepper with up to 750 mA of current per phase.
 Controllable and Programmable Full, Half, Quad or Eight degrees of microstepping.
 Controllable and Programmable Fast, Slow and Mixed Decay Modes.
 Command both direction at hundreds of different speeds.
 Command both direction at hundreds of different speeds for a number of steps.
 Command both direction at hundreds of different speeds for a number of steps or until it reaches home.
 An optical sensor or switch input, works as the home sensor.

The SS-MST67 is a microstepping full controllable engine capable of running small to medium size bipolar stepper motors. Up to 8 degrees of microstepping will allow for practically vibrationless motion at even very slow speeds at the same time torque characteristics are improved.

Controller contains commands to start the motor at a slow speed and accelerate up to a commanded speed. This profile curve is fully programmable.

The board contains one sensor input and one motor output connector. The sensor input can be used to Home the stepper motor when moving on a special version of the move steps command. This way the motor can start moving from a predefined location.

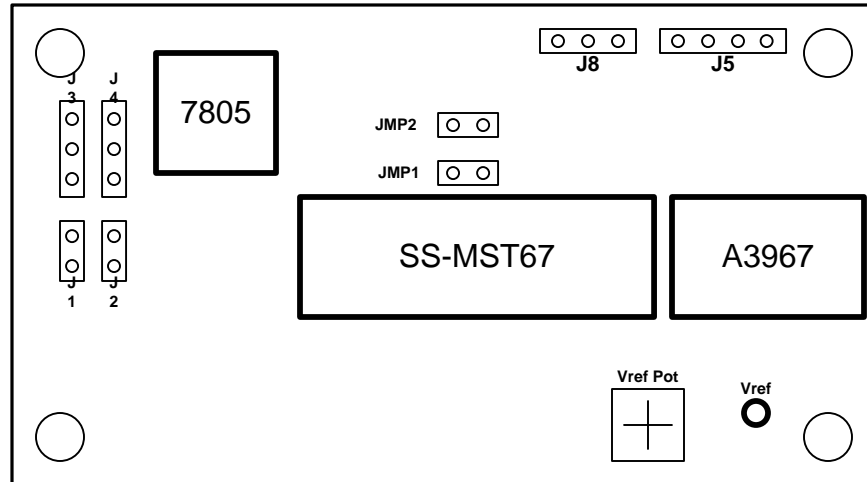
The module receives command through the SSB (Super Stepper Bus). Two jumpers, JMP1 and JMP2, configure the controller UART for either 8 or 9 bit communication and a different range of BAUD Rates and addresses.

Commands received through the bus are executed immediately. Available commands for the SS-MST67 are shown on Table 1.

Super Stepper Micro Stepping Controller SS-MST67							
Opcode	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Description	# of Bytes
0	Dir/Rate	Speed H	Speed L	-	-	Move at Speed	4
1	Home/Dir/Rate	Speed H	Speed L	Steps H	Steps L	Move # of Steps At Speed	6
2	Speed H	Speed L	-	-	-	Change Speed	3
3	MSDeg/Decay	-	-	-	-	Change MS Degrees and Decay Mode	2
4	Decel EN/DIS*	-	-	-	-	Stop/Decelerate	2
5	RS	RS	RS	RS	RS	Expanded	RS
6	Mem Sel	Address	Data	-	-	Write	4
7	Mem Sel	Address	-	-	-	Read	3

Table 1

Stepper Controller – Micro Stepping Controller SS-MST67



Block Diagram 1

Hardware:

J1 and J2 Power Connector. Connect a power source from 9V to 15V. J2 is a bypass to another Porta Board piggy backed on top.

- Pin1: Positive Power Source
- Pin2: Ground

J3 and J4 Serial Communications Connector: Connect the TTL level serial communication signal as obtained from a PC through an RS-232 driver or any microcontroller UART output.

- Pin1: Module Receive (Rx)
- Pin2: Module Transmit (Tx)
- Pin3: Ground

J5 Stepper Output Connector. Connect the four phases of a bipolar motor. Please observe polarity for proper rotation.

- Pin1: A+
- Pin2: A-
- Pin3: B+
- Pin4: B-

J6 and J7 Unused Connector in this application.

J8 Home Sensor Input. Connect a switch or optical sensor to determine when the home position has been reached. Since Input signal is asserted low, please connect the switch with respect to ground. The input signal has a pull up to Vcc.

- Pin1: Vcc
- Pin2: Input Signal
- Pin3: Ground

Stepper Controller – Micro Stepping Controller SS-MST67

Hardware (Cont.)

JMP1 MOD1 Jumper. Selects from UART hardwired parameters or EEPROM stored parameters.

Open: BAUD Rate is 2400 and SS-ADDRESS is 0.

Closed: BAUD RATE is stored on EEPROM address 0 and SS-ADDRESS is stored on EEPROM address 1.

JMP2 MOD0 Jumper. Selects from 8 bit or 9 bit communications.

Open: 8 bit communications selected.

Closed: 9 bit communications selected.

Vref: A pad utilized to set the reference voltage input on the stepper driver (DC voltmeter needed). Voltages read from 0V to 5VDC and set the current according to the following equation:

$$I = V_{ref}/8$$

R6 potentiometer: Adjusts Vref (see Vref Pad information). Rotate clockwise to decrease Vref voltage until reaching 0VDC. Rotate counterclockwise to increase Vref until reaching 5VDC.

Stepper Controller – Micro Stepping Controller SS-MST67

Commands:

Move Stepper at Speed X: (Opcode 0) Starts to move stepper at low speed and increases until reaching specified speed. The word operand (two bytes) is the velocity command. The Dir/Rate parameter specifies direction (1 for Clockwise and 0 for Counterclockwise) and speed rate (00 for Fast Rate, 01 for Medium Rate and 10 for Slow Rate). The rate is used to divide the internal timer which is responsible of generating the steps.

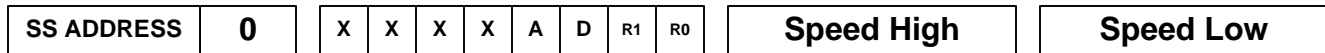


Figure 1

Rate1	Rate0	Description	Speed Range	Rate Divider
0	0	Motor moves at fast rate	100 to 65535	1
0	1	Motor moves at medium rate (Fast Rate / 8)	20 to 65535	8
1	0	Motor moves at slow rate (Fast Rate / 64)	2 to 65535	64
1	1	Do not use this rate as the motor will not move!	DO NOT USE	DO NOT USE

Table 2

Direction Bit (D): This bit is used to specify the direction at which the motor will move. Clockwise is specified by a 1 in this position, while clockwise with a 0.

Acce/Decel Bit (A): This bit is used to specify whether the acceleration profile internal to the microcontroller will be used. Writing a bit equal to 1 in this position will enable the Acceleration profile while putting a 0 disables it. (Refer to page 5 “Computing stepper Acceleration and Deceleration” section for more information)

How to compute Speed:

The internal timer generates clock ticks at a 4MHz rate (250 ns per clock tick). This timer drives the I/O pin which generates the steps at the driver chip. The driver chip requires a transition from high to low for a step to take place. The timer generates such step every two SPEED Parameter number of clock ticks (i.e. one SPEED # of ticks to transition from high to low and another SPEED # of ticks to transition back from low to high).

Hence a SPEED parameter equal to 4000, will yield a transition every 1 ms. This means that every 2 ms a step will take place. The higher the Speed Parameter, the slower the motor moves. This is because each step is now made out of more clock ticks. Hence it takes long for each phase to change which translates to less phase changes or steps per second.

$$\text{Step Period} = (250\text{E-}6) * (\text{Rate Divider}) * (\text{Speed Parameter}) * 2$$

$$\text{Steps Per Second} = 1 / \text{Step Period}$$

Motor speed is usually computed in RPM. This is a function of the stepper motor and how many steps make a revolution. Per example, steppers are graded in degrees per step. A 1.8 degree stepper, needs 200 steps to do a full revolution (360 degrees make up a revolution). The steps per second are then used to compute the Revolutions Per Minute (RPS). Once this is known multiplying by 60 will yield the Revolutions Per Minute (RPM).

Stepper Controller – Micro Stepping Controller SS-MST67

$$\text{RPS} = \text{Steps Per Second} / \text{Stepper Steps per Revolution}$$

$$\text{RPM} = \text{RPS} * 60$$

MicroStepping:

The previous section, clearly explains how to compute speed while working with Full Step Mode. However, this is a microstepping capable controller. Hence, the user will find that each step can be divided in two steps, four steps (quad) or eight steps. Dividing steps in such a fashion allows for an improved performance in motion control. Here is how to compute the speed while using one of these three improved modes.

$$\text{Step Period} = (25E-6) * (\text{Rate Divider}) * (\text{Speed Parameter}) * 2$$

$$\text{Steps Per Second} = 1 / \text{Step Period}$$

Equations shown above apply for Full, Half, Quad or Eight Step driving.

Motor speed is usually computed in RPM. This is a function of the stepper motor and how many steps make a revolution. Per example, steppers are graded in degrees per step. A 1.8 degree stepper, needs 200 steps to do a full revolution (360 degrees make up a revolution). Here is where the difference with microstepping arises

The controller has the ability of segmenting a step into multiple micosteps. A 1.8 degree stepper will behave as a .9 degree stepper while being run at half step mode. Hence 400 microsteps are needed to do a full revolution. Interpolating to eight step driving, the stepper now behaves as .225 degrees stepper, needing 1600 microsteps to perform a full revolution.

NOTE: While running at fractions of a step modes (i.e. half, quad, eight), each step is called a microstep.

The steps per second and step fraction are then used to compute the Revolutions Per Second (RPS). Once this is known multiplying by 60 will yield the Revolutions Per Minute (RPM).

$$\text{RPS} = \text{Steps Per Second} / (\text{Stepper Steps per Revolution} * \text{MS Degrees})$$

$$\text{RPM} = \text{RPS} * 60$$

Where:

MS Degrees is 2 for Half, 4 for Quad and 8 for eight step resolution

Stepper Steps per Revolution is the stepper motor physical quantity of how many full steps are on a shaft revolution

NOTE: Motors being driven with Eight Step Mode microstepping will move considerably lower. Very slow RPM speeds can be achieved this way, with the advantage of little vibration.

Stepper Controller – Micro Stepping Controller SS-MST67

Computing Stepper Acceleration and Deceleration:

Although deceleration can be disabled (at the user risk), acceleration can not. In this section, a technique to bypass it will be discussed, although it is not recommended for reasons concerning resonance.

The acceleration and deceleration profile is controlled by two variables: Accel/Decel Rate and Accel/Decel Time Base. The Rate specifies how much will the motor speed increase while the time base specifies how fast will these speed increases take place. (HINT: If the user wishes to jump to a speed right away, simply use large rates with a small time base).



Figure 2

Rate and Time Base can be programmed in EEPROM. After reset, both parameters are downloaded into RAM memory. The user can change these values at run time. A reason to do this would be to have a different deceleration profile than acceleration profile, since the parameters are shared for both profile segments.

Acceleration Rate is a word while Time Base is a byte. Hence, the slowest the acceleration or deceleration profile can take is 256 milliseconds per speed increase/decrease.

Move Stepper at Speed X for Number of Steps Y: (Opcode 1) Starts to move the stepper at specified speed until the number of steps have been processed or until the home sensor switches to a positive status. The Home/Dir/Rate parameter specifies whether home sensor lookup is desired (1 home sensing enabled, 0 home sensing disabled), direction (D = 1 for Clockwise and 0 for Counterclockwise) and speed rate (R1:R0 = 00 for Fast Rate, 01 for Medium Rate and 10 for Slow Rate). The rate is used to divide the internal timer which is responsible of generating the steps.



Figure 3

The first parameter byte carries information such as timer rate, motor direction and a homing command.

Refer to Table 2 for details on the Dir, R1 and R0 bits on the first parameter byte. Refer to the "How to Compute Speed" section on page 3 for more information on how to compute actual stepper speed with the Speed Hi and Speed Lo bytes.

The homing command is a bit that when logic high (1) will tell the controller to move the stepper until a falling edge transition has been observed in the Home Sensor input. Logic low (0) disables this feature.

Number of Steps: Specifies how many steps the stepper will move. The stepper stops as soon as the number of steps are processed. When the stepper stops, the motor windings will still be energized.

Stepper Controller – Micro Stepping Controller SS-MST67

Change Micro Stepping Degrees and Decay Mode: (Opcode 3) This command will change the number of microsteps per step (half, quad or eight) and the decay mode. This change takes place as soon as the command is executed and can happen while the motor is moving.

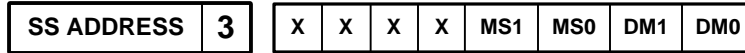


Figure 4

DM0:1 Decay Mode Bits – Select between Slow, Fast or Mixed Decay Mode as shown on table 3.
NOTE: Mixed decay mode eliminates some vibration while running on 8 degrees of microstepping.

DM1	DM0	Description
0	0	Fast Decay Mode Selected
0	1	Mixed Decay Mode Selected
1	0	Mixed Decay Mode Selected
1	1	Slow Decay Mode Selected

Table 3

MS1:0 Micro Stepping Degrees – Select between Full Step, Half Step, Quad Step or Eight Step microstepping resolution. See Table 4.

MS1	MS0	Description
0	0	Full Step
0	1	Half Step
1	0	Quad Step
1	1	Eight Step

Table 4

Stop: (Opcode 4) This command will always result in the motor stopping. There are different ways in which this can be achieved. The En/Dis parameter specifies whether the driver is disabled or is kept enabled (1 for driver enabled with current flow, 0 for driver disabled with no current flow). The Decelerate bit specifies whether the motor should stop immediately or go through the deceleration profile. (1 the motor goes through deceleration, 0 the motor stops immediately).

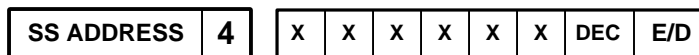


Figure 5

Stepper Controller – Micro Stepping Controller SS-MST67

Write RAM/EEPROM: (Opcode 6) Writes to internal RAM or EEPROM memory. The MEM SEL parameter specifies which type of memory will be accessed. Logic Low (0) accesses RAM while Logic High (1) accesses EEPROM. All writes are done to byte memory spaces. Writing to a word is not supported, so user must do two consecutive word writes to write to a word memory allocation, if needed.

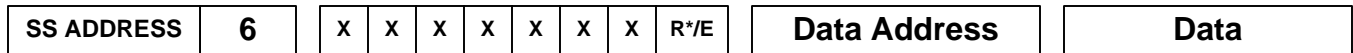


Figure 6

R*/E	Description
0	Read a RAM Byte from ADDRESS
1	Read an EEPROM Byte from ADDRESS

Table 3

Read RAM/EEPROM: (Opcode 7) Reads from internal RAM or EEPROM memory. The MEM SEL parameter specifies which type of memory will be read and if the read request will return a word or a byte. Only reads to RAM can return a word. EEPROM reads always returns a byte.

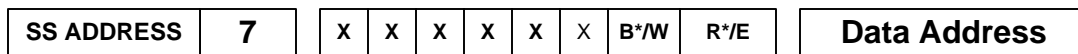


Figure 7

B*/W	R*/E	Description
0	0	Read a RAM Byte from ADDRESS
0	1	Read an EEPROM Byte from ADDRESS
1	0	Read a RAM Word (16bit) from ADDRESS
1	1	Read an EEPROM Byte from ADDRESS

Table 5

Data Return

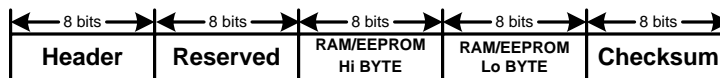


Figure 7

When reading a byte or word from EEPROM or RAM, the Positive Acknowledge data stream (as portrayed on the Super Stepper Architecture Data Sheet) includes the returned data segment read. In the Read Byte style instruction, the Hi Byte is returned on the third byte and the Lo Byte on the fourth byte, prior to the checksum byte. If reading a Byte only, the Hi Byte slot contains 0.

Stepper Controller – Micro Stepping Controller SS-MST67

BAUD Rate Selection:

The SS-ST68 board communicates through the Super Stepper Serial Communication Protocol. Since the board has been implemented with a 4MHz crystal, certain BAUD Rates can be obtained. Programming the EEPROM with one of the values found on Table 6 will configure the UART to operate with the selected BAUD Rate after the device comes out of power on reset next time. This feature is of course available if the JMP1 is closed.

Baud Rate	EEPROM	Baud Rate	EEPROM
2400	103	28800	8
4800	51	38400	6
9600	25	57600	3
14400	16	76800	2
19200	12	115200	1

Table 6

EEPROM Memory Map:

The user must refer to this section whenever the EEPROM is to be programmed. Programming the EEPROM memory on the device may result on non proper functionality.

EEPROM Address	Contents	EEPROM Address	Contents
0x00	SS BAUD Rate	0x04	Accel Decel Time Base
0x01	SS Address	0x05	Starting Speed (Hi)
0x02	Accel Decel Rate (Hi)	0x06	Starting Speed (Lo)
0x03	Accel Decel Rate (Lo)	0x07 to 0xFF	User defined

Table 7

Accel Decel Rate (0 to 65535) See Computing Computing Stepper Acceleration and Deceleration on page 4 for more information.

Accel Decel Time Base (0 to 255) See Computing Computing Stepper Acceleration and Deceleration on page 4 for more information.

Starting Speed (0 to 65535) Starting Speed at which the stepper controller will start the acceleration profile. (i.e. If Starting Speed is 32000, and the motor is commanded to move at 10000, the stepper motor will accelerate from 32000 until reaching Speed Command of 10000)

These bytes are transferred into RAM after powering the controller. Address 0x02 on the EEPROM gets transferred into address 0x00 on RAM.

Once the EEPROM is programmed with the desired values, a power-off/power-on sequence is needed for the values to take place.

User defined : Good place to store user defined parameters needed for future recall.

Stepper Controller – Micro Stepping Controller SS-MST67

RAM Memory Map:

The user must refer to this section whenever the RAM is to be read or written. Modifying RAM memory on the device could result on non proper functionality.

RAM Address	Contents	RAM Address	Contents
0x00	Accel Decel Rate (Hi)	0x06	Status Register
0x01	Accel Decel Rate (Lo)	0x07	Number Of Steps Run (Hi)
0x02	Accel Decel Time Base	0x08	Number Of Steps Run (Lo)
0x03	Starting Speed (Hi)	0x09 to 0x17	User defined
0x04	Starting Speed (Lo)	0x18 to 0x1F	Do not Use
0x05	Steps/DecayMode		

Table 8

Accel Decel Rate (0 to 65535) See Computing Computing Stepper Acceleration and Deceleration on page 4 for more information.

Accel Decel Time Base (0 to 255) See Computing Computing Stepper Acceleration and Deceleration on page 4 for more information.

Starting Speed (0 to 65535) Starting Speed at which the stepper controller will start the acceleration profile. (i.e. If Starting Speed is 32000, and the motor is commanded to move at 10000, the stepper motor will accelerate from 32000 until reaching Speed Command of 10000)

These bytes are transferred into the respective RAM space (from their EEPROM counterparts) after powering the controller. Address 0x02 on the EEPROM gets transferred into address 0x00 on RAM.

Values changed on EEPROM require a power-off/power-on sequence for changes to take effect on RAM space. The user can modify the RAM parameters if immediate action is necessary.

Stepper Controller – Micro Stepping Controller SS-MST67

Status Register:

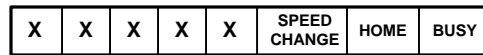


Figure 8

Status Register offers the following information to the user:

BUSY Bit: When the motor is rotating, the BUSY bit is asserted (1).

HOME Complete: When the motor stopped due to a transition on the home sensor, the HOME Complete bit is asserted (1)

SPEED CHANGE: When the motor is changing speed during acceleration or deceleration, the SPEED CHANGE bit is asserted (1)

Other RAM:

Number Of Steps Run (0 to 65535) Once the stepper is on rotation mode (Move at Speed Commands), it holds the amount of steps the controller has submitted to the driver. Good for Tachometry purposes.

User defined : Good place to hold user defined variables.

Stepper Controller – Micro Stepping Controller SS-MST67

Disclaimer:

Super Stepper is a trademark of Avayan Electronics. Protocol may not be used by third party providers either partially or totally without Avayan Electronics' consent.

The material in this data sheet reflects the technology as developed by Avayan Electronics Electrical Engineering and Research Laboratories. It may change at any time without prior notice.

Super Stepper is an architecture for motion control, automation and modular robotic design. The architecture does not comprise redundancy or error check and correction enough to make it robust against faults which may put in danger human life as in medical appliance applications, toxic waste handling control, space exploration, etc. Use of the protocol for such tasks may result in serious injury in the event of communication failure. Avayan Electronics can not be made responsible for such occurrences.