8SMC1-USBh

1.5A Microstep Driver with USB Interface

User Manual

CE

Index

1 General information

1.1 Features

Electrical
- Average current per phase of stepping motor up to 1.5A
- Short circuit, overcurrent, overvoltage and temperature protections
- Screw mounted and easy to change current sense resistors
- Multi-purpose 40 pin connector for embedded applications
- A number of additional connectors (15 pin D-Sub, USB type A, 2x USB type B, DC input) for stand alone applications
- Different ways of power supply, including single power, dual power and USB power (last for 8SMC1-USBh logic only)

Motion
- Resolution: full step, 1/2, 1/4, 1/8
- Speed up to 5000 steps/s
- Programmable speed and trip points
- Programmable accel and decel ramps
- Soft start/stop mode
- Synchronization I/O

Control
- Two knobs and three LEDs for manual control and local indication
- Two programmable limit switches and emergency stop switch
- Quadrature encoder or revolution sensor support
- Remote control via USB 1.1 interface (up to 12 Mbps)
- Embedded 3-port USB Hub, two USB ports are free for other USB device connections

Graphical user interface for Windows 2000/XP/Vista
- Drivers and dynamic link library for Windows 2000/XP/Vista/Mobile host programming
- Set of virtual instruments for National Instruments LabVieW

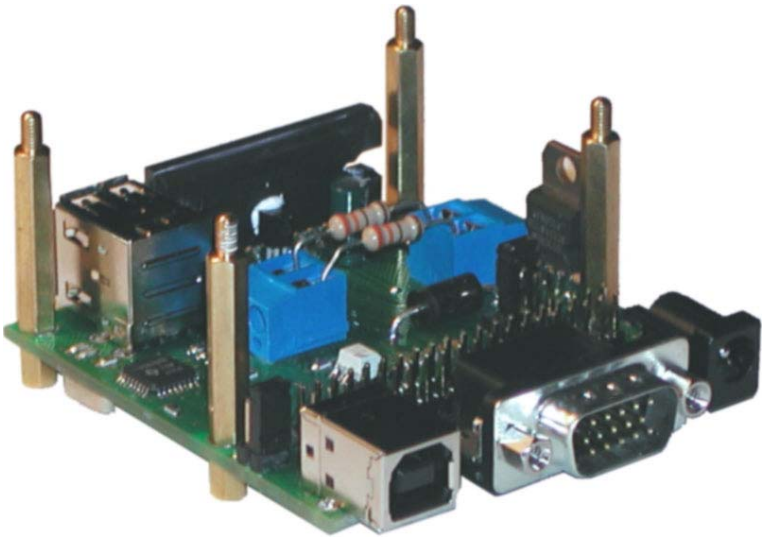


Figure 1. 8SMC1-USBh

1.2 Description

The 8SMC1-USBh controller is designed to drive one bipolar stepping motor by local and/or remote means. Local control and indication are implemented by two knobs and three LEDs. Remote control and monitoring are implemented via USB interface from PC. Inputs for two limit switches, emergency switch and revolution sensor are provided.

The 8SMC1-USBh incorporates PWM chopper type sinusoidal micro step bipolar stepping motor driver, fully integrated mixed-signal System-on-a-Chip MCU, USB controller and 3-port USB hub on one board. Rated current for each stepping motor is set by selection of two current sense resistors. The latter are screw mounted and easy to change. Sufficient

set of resistors is supplied with 8SMC1-USBh. Controller features include built in over temperature, over voltage, short circuit and reverse supply protections. There is a potential of 40% current reduction in hold mode. In order to improve the high speed performance and to reduce heating separate supply for logic and stepping motor can be used. Logic of 8SMC1-USBh can be powered by three different ways: directly from USB, from external +5V DC power supply, from external 7-12V power supply. Emergency stop switch for unconditional stepping motor power down is available. It may be utilized when safety is necessary. There are two ways of communication to 8SMC1-USBh board: by multi-purpose 40 pin connector (convenient for embedded applications) or by a number of special additional connectors (15 pin D-Sub for stepping motor, USB type A as USB input, 2x USB type B as USB outputs, DC input for stepper motor power supply). Embedded 3-port USB hub allows cascading of the 8SMC1-USBh boards: it possible to connect 8SMC1-USBh board to each of two free downstream USB ports on another 8SMC1-USBh board. Thus, it is possible to connect up to 30 devices per USB host controller in tree-like structure without any additional USB hubs.

The built-in powerful MCU on the 8SMC1-USBh allows the user, via USB interface, to control parameters such as position, acceleration/deceleration ramps, velocity, direction, resolution, drive current, etc., to form simple or complex motions. Most of all commands are executed on-the-fly. All parameters can be saved on PC or in MCU flash memory. The 8SMC1-USBh has a variety of built-in functions, including local control, programmable limit switch inputs, homing algorithm, quadrature encoder or revolution sensor support, additional input/output for synchronization.

Thus 8SMC1-USBh is compact, all-in-one, low cost microstep stepping motor driver with high functionality and modern USB interface.

1.3 Applications

Controller can be used to drive any motorizes devices if their stepper motor parameters match specifications for 8SMC1-USBh.

1.4 Compatibility

1.4.1 Connectivity

Controller is designed to work with IBM AT compatible computer systems (with Pentium or better processors) with Microsoft Windows 2000/XP/Vista operation systems or mobile devices with Microsoft Windows Mobile 5.0 and higher. Only presence a USB host port on your host device is necessary.

1.4.2 Stepper motors

Controller can operate with stepper motors according to the technical specification and wiring requirements. Maximum allowable average phase current is 1.5A, rated voltage is 40V.

1.5 Warranty

Producer warrants the controller card 8SMC1-USBh for the period of 1 year from the date of sale.

2 Functional description

2.1 Board overview

Appearance of the 8SMC1-USBh printed circuit board is shown on Figure 2.

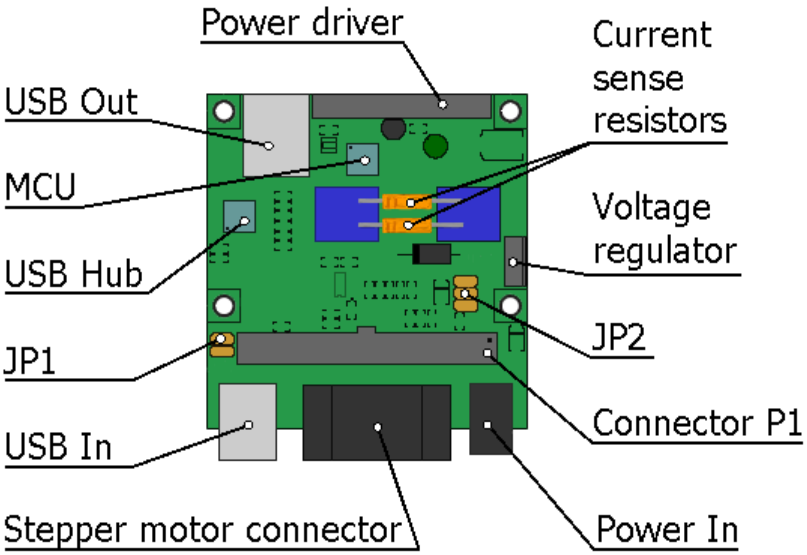


Figure 2. Appearance of 8SMC1-USBh printed circuit board

Power driver - Sinusoidal micro step bipolar stepping motor driver. It controls the stepper motor.

Warning: The Power driver IC fin (rear) is electrically connected to the rear of the chip. When current flows to the fin, the Power driver IC malfunctions. If there is any possibility of a voltage being generated between the ground of the 8SMC1-USBh and the fin, either ground the fin or insulate it.

Warning: There are appreciable power dissipation on Power driver (up to 6 W depending on input voltage and rated current of stepper motor). Appropriate heatsink must be used to maintain temperature range. Heating the Power driver over 85 ^{o}C is forbidden!

USB hub - This chip provides one upstream port and three downstream ports in compliance with the USB version 1.1 specification. Ports support both full-speed and low-speed devices by automatically setting the flow rate according to the speed of the device attached to the ports. Upstream port is wired to *USB In* and *P1* connectors and used for connections with computer or higher USB hub. One of downstream ports is wired to *MCU*.

MCU - Full Speed USB, 16K ISP FLASH Microcontroller. This chip provide interface between computer and stepper motor. It keeps all current data about stepper motor state and sends it to computer on demand.

USB Out - Two USB type A connectors are placed on the rear of 8SMC1-USBh printed circuit board. It allows the connection of other USB devices or lower USB hubs.

JP1 - Enable/Disable Emergency stop switch jumper, see Figure 3 and Figure 4.

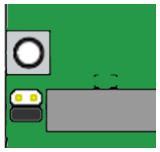


Figure 3. Emergency stop switch Enabled

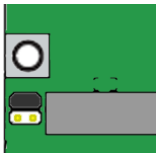


Figure 4. Emergency stop switch Disabled

USB In - USB type B connector is placed on the front of 8SMC1-USBh printed circuit board. It is used for connections with computer or higher USB hub.

Stepper motor connector - This 15 pin D-Sub (male) connector is used for connection of 8SMC1-USBh device to stepper motor. All additional wiring (limit switches, emergency limit switch, revolution sensor and corresponding power supply) is included in this connector too. Dimensions of stepper motor connector see on Figure 5. For more information see 2.2.2.

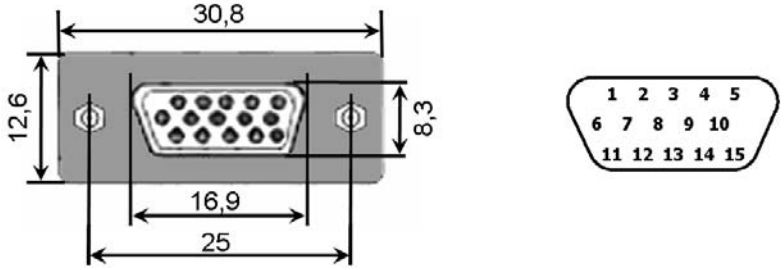


Figure 5. Stepper motor connector: dimensions and pinout

Current sense resistors - Pair of current sense resistors is necessary to keep correct rated current on windings of stepper motor. Resistors are screw mounted and easy to change. Sufficient set of resistors is supplied with 8SMC1-USBh.

Warning: Current sense resistors must be chosen for every stepper motor according to its rated current and connection diagram. For more details see 4.2.1. of User Manual. Wrong current sense resistors can cause the malfunction of 8SMC1-USBh or damage stepper motor.

Voltage regulator - It is used for supply board logic when external 7-12V power supply for logic is used (for more details see 2.2.1).

Warning: There is some power dissipation on Voltage regulator (up to 0.5 W depending on input voltage). Appropriate heatsink may be required. Heating the Voltage regulator over 85 ^{o}C is forbidden!

JP2 - Board logic power supply jumper. A way of logic power supply is determined by this jumper. There are three different ways: directly from USB (as shown on Figure 6), from external +5V stabilized DC power supply (as shown on Figure 7), from external 7-12V unstabilized DC power supply (as shown on Figure 8).

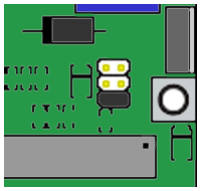


Figure 6. USB powered board logic

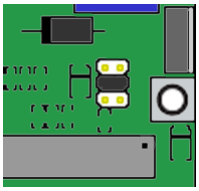


Figure 7. Board logic powered by 5V DC power supply

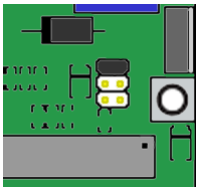


Figure 8. Board logic powered by 7-12V power supply

Note: If you want to change the power logic jumper position make the following steps: Turn the power of 8SMC1-USBh board OFF, unplug the USB In connector, change the JP2 position, turn the power of 8SMC1-USBh board ON and plug USB In connector back.

Don't change the JP2 position when USB cable is plugged because it can entail some mistakes in USB root hub operation.

Connector P1 - Multi-purpose 40 pin connector. All other connectors (USB In, USB Out, Stepper motor connector and Power In) are duplicated here. This connector is designed for embedded applications. Only one standard ribbon cable required for all connections by P1. Connector P1 Pin assignment is shown on Figure 9. Dimensions are shown on Figure 10.

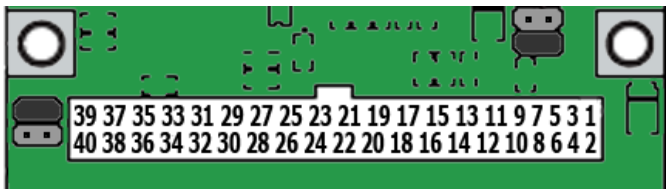


Figure 9. Connector P1. Pinout

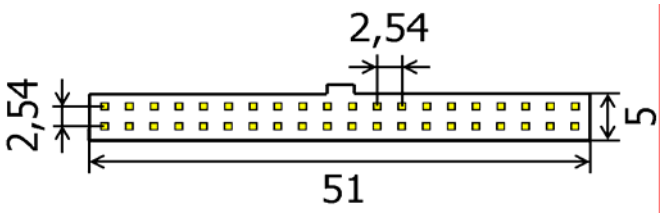
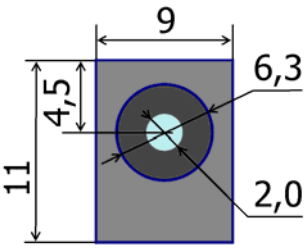


Figure 10. Connector P1. Dimensions.

Note: Some specific contacts (LEDs and knobs for manual control, for example) are only on connector P1.

Power In - Low-voltage power socket. 7-40V power input for stepper motor only or 7-12V power input for stepper motor and board logic. Operating mode is determined by JP2 and Connector P1.



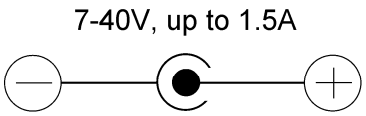


Figure 11. Power In connector. Dimensions. **Figure 12.** Power In connector. Pinout.

Dimensions are shown on Figure 11. Contacts appointment is shown on Figure 12. Outside contact has 6,3 mm diameter. It is Ground. Central contact has a 2,0 mm diameter. It is a positive contact for 7-40V stabilized DC power supply connection.

2.2 External connections

2.2.1 Power

8SMC1-USBh board allows several different ways of power supply connections. They are shown on Table 1.

		Stepper motor supply	
		Powered by 7-12V DC	Powered by 7-40V DC
Logic supply	USB powered	**USB Powered logic mode**, for details see 2.2.1.1	
	Powered by 5V DC	**5V DC powered logic mode**, for details see 2.2.1.2	
	Powered by 7-12V DC	Single 7-12V DC power supply mode, for details see 2.2.1.3	Double power supply mode, for details see 2.2.1.4

Table 1. 8SMC1-USBh power supply modes

Warning: 8SMC1-USBh power supply must never exceed 40V. Power driver can be damaged if such exceeding happened.

Note: Exact current consumption from external 7-40V DC power supply for stepper motor supplying purposes depends on stepper motor rated current and voltage. It is recommended that rated current and voltage loading of this DC power supply exceed the stepper motor rated current and voltage. It is true for all ways of power supply connection.

Let's examine this table more accurately.

2.2.1.1 USB Powered logic mode

In this case logic of 8SMC1-USBh board is powered by USB; stepper motor is powered by external 7-40V stabilized DC power supply. Connection diagram for Power In is shown on Figure 12, for multi-purpose 40 pin connector P1 on Figure 13. Correct position of board logic power supply jumper (JP2) is shown on Figure 6.

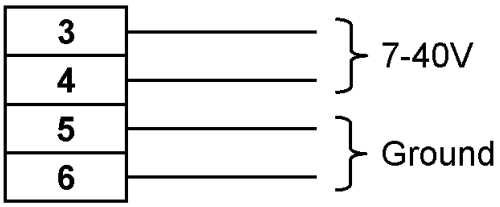


Figure 13. Connection of 7-40V stabilized DC power supply to P2 connector

Note: In this mode 8SMC1-USBh requires 80 mA from USB. Make sure that feeding USB hub can provide this current.

Note: Almost all active USB hubs (include USB root hubs) provide 500 mA power supply that is sufficient for five 8SMC1-USBh devices.

2.2.1.2 5V DC powered logic mode

In this case logic of 8SMC1-USBh board is powered by external stabilized +5V DC power supply; stepper motor is powered by external 7-40V stabilized DC power supply. Correct position of board logic power supply jumper (JP2) is shown on Figure 7. Connection diagram for multi-purpose 40 pin connector P1 is shown on Figure 14.

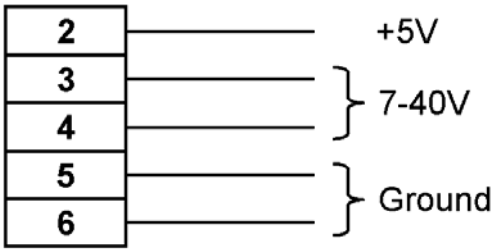


Figure 14. Power supply connection to P2 connector in +5V DC powered logic mode

Note: In this mode 8SMC1-USBh requires 80 mA from external stabilized +5V DC power supply. Make sure that external stabilized +5V DC power supply can provide this current.

2.2.1.3 Single 7-12V DC power supply mode

In this case logic of 8SMC1-USBh board and stepper motor is powered by single 7-12V DC power supply. Correct position of board logic power supply jumper (JP2) is shown on Figure 8. Connection diagram for powering through multi-purpose 40 pin connector P1 is shown on Figure 15. Connection diagram for powering through Power In connector is shown on Figure 12. In the last case pins 1 and 3 of multi-purpose 40 pin connector P1 must be connected together by jumper or other way as shown on Figure 16.

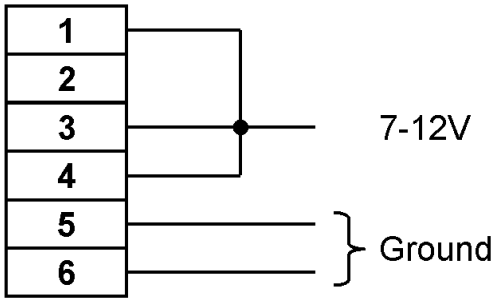


Figure 15. Power supply connection to P2 connector in single 7-12V DC power supply mode

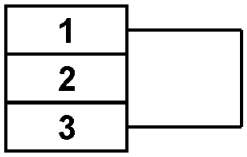


Figure 16. Pin 1 and 3 connection in single 7-12V DC power supply mode

Warning: If a single power supply mode is used, make sure that input voltage is not exceeding 12V. Otherwise the overheating of the Voltage regulator may happen. Heating the Voltage regulator over 85 ^{o}C is forbidden!

2.2.1.4 Double power supply mode

In this case logic of 8SMC1-USBh board is powered by external 7-12V DC power supply; stepper motor is powered by external 7-40V stabilized DC power supply. Correct position of board logic power supply jumper (JP2) is shown on Figure 8. Connection diagram for multi-purpose 40 pin connector P1 is shown on Figure 17.

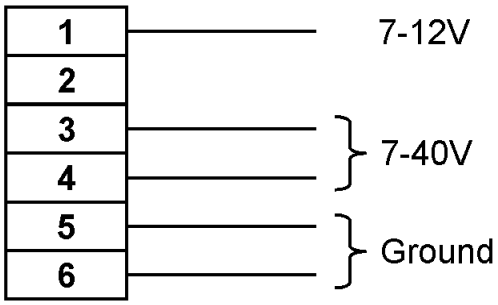


Figure 17. Power supply connection to P2 connector in double power supply mode

2.2.2 Stepper motor connection

There are two ways of stepper motor connections: by Stepper Motor Connector (see Figure 19) or by multi-purpose 40 pin connector P1 (see Figure 18).

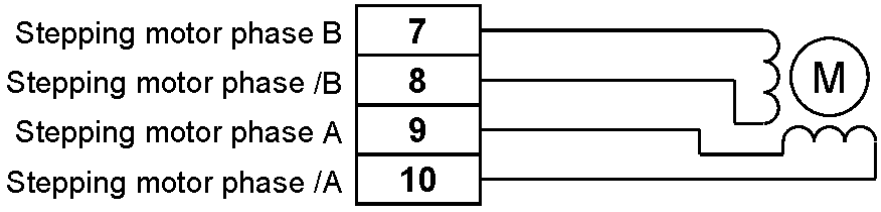


Figure 18. Stepper Motor Connection to multi-purpose 40 pin connector P1

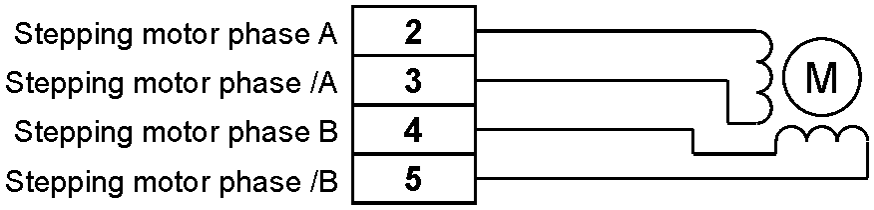


Figure 19. Stepper Motor Connection to Stepper Motor Connector

Warning: Make sure that there are no contact between stepping motor phase windings and 8SMC1-USBh ground. Power driver will be damaged obligatory if such grounding present.

Warning: You must never connect to, or disconnect from the 8SMC1-USBh any stepping motors while the controller keeps currents in the motor windings. Power driver can be damaged if such reconnection happened.

2.2.3 Connection of external sensors

All external sensors (limit switches, emergency limit switch, revolution sensor and knobs) are connected to 8SMC1-USBh by the same way. There are several different possibilities of connection for every sensor. Farther we examine these possibilities more accurately.

Electrically external sensors may be a dry contact type or phototransistor based. All inputs have an internal weak pull up to 3.3V. Open circuit correspond to high level on input, closed to ground circuit correspond to low level on input. Controller inputs are marked on Figure 20, Figure 21, Figure 22, Figure 23 and Figure 24 as **XX**.

Note: Be careful with optocouples. Makes sure that voltage drop on phototransistor exceed 2.5V in off state and do not exceed 0.5V in on state. It is important for correct work of 8SMC1-USBh input 3.3V logic.

The pattern of phototransistor based external sensor connection to multi-purpose 40 pin connector P1 is shown on Figure 20, to Stepper motor connector - on Figure 21. One optocouple can be connected to Stepper motor connector by simple way that illustrated on Figure 22. No external current-limiting resistors are required. Revolution sensor (see 2.2.6) is connected by this way usually. The pattern of dry contact type external sensor connection to multi-purpose 40 pin connector P1 is shown on Figure 23, to Stepper motor connector is shown on Figure 24.

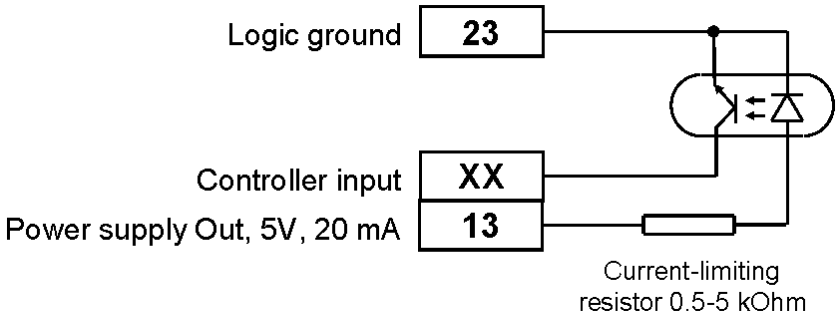


Figure 20. Pattern of optocouple connection to multi-purpose 40 pin connector P1

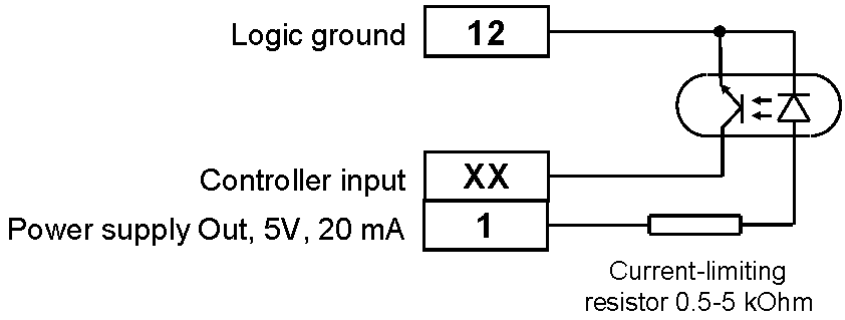


Figure 21. Pattern of optocouple connection to Stepper motor connector (Method 1)

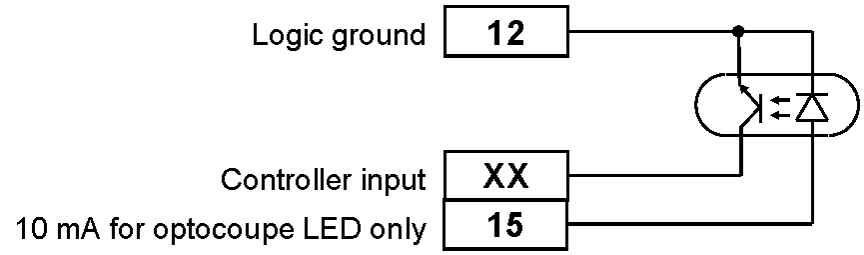


Figure 22. Pattern of optocouple connection to Stepper motor connector (Method 2)

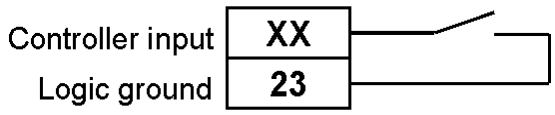


Figure 23. Pattern of dry contact connection to multi-purpose 40 pin connector P1

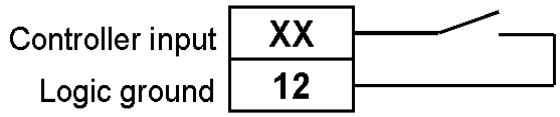


Figure 24. Pattern of dry contact connection to Stepper motor connector

2.2.4 Limit switches

Two limit switches can be used: Limit switch 1 and Limit switch 2. These contacts are used for determination of limits in translational stages, for determination a null position in rotational stages, etc. It can be programmed as normally opened or closed contact. It can be enabled or disabled by software (see 5.4.10). Also it is possible to swap limit switches by the software. Limit switches are connected to 8SMC1-USBh according to 2.2.3. Controller inputs for each limit switch are shown on Table 2.

	Pin number on	
External sensor	Stepper motor connector	Multi-purpose 40 pin connector P1
Limit switch 1	7	20
Limit switch 2	8	17

Table 2. Connection of Limit switches

2.2.5 Emergency Stop switch

Emergency stop switch is used for unconditional stepping motor power down. It can stop stepper motor forcedly if it is impossible to do by software in consequence of program error. This sensor is connected directly to power down circuit of Power driver. Emergency stop switch must be a dry break contact. There is no possibility to control this switch by software, only monitoring is possible. If Emergency stop switch is not used, set emergency stop jumper in Off position (see Figure 4), otherwise in On position (see Figure 3).

Note: If Emergency stop switch is not used but emergency stop jumper set in On position then stepper motor will be de-energized and **Ext. off** indicator will lighting up (see. 5.2.5). For correct working set the jumper in Off position (see Figure 4).

Connection diagram to multi-purpose 40 pin connector P1 is shown on Figure 25, to Stepper motor connector is shown on Figure 26.

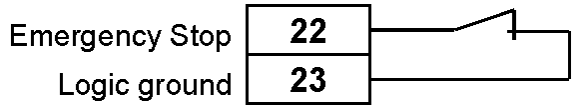


Figure 25. Connection Emergency Stop switch to multi-purpose 40 pin connector P1

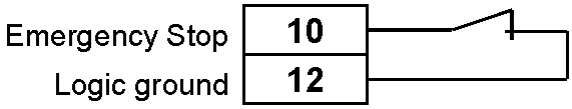


Figure 26. Connection Emergency Stop switch to Stepper motor connector

2.2.6 Revolution sensor

Revolution sensor is intended for stepper motor stall detection. 8SMC1-USBh can receive data about the real position of stepper motor shaft from Revolution sensor. Latter is an external sensor (see 2.2.3) that joins directly to stepper motor shaft and change electrical state several times per stepper motor shaft revolution.

Usually Revolution sensor is a small disk with narrow axial slit. Disk mounts directly on stepper motor shaft. Optocouple is used as an external sensor. Optocouple's emitter and detector are set near the slit of the disk, by different sides. When slit is positioned on emitter and detector optocouple is opened (optocouple output signal is low). If slit closes emitter from detector optocouple is closed (optocouple output signal is high).

Examine how the Revolution sensor works more accurately by the example of SMCVieW application (see chapter 5). When Revolution sensor is enabled and stepper motor is moving, controller counts tics (one full step is equal to 64 tics) which it gives to stepper motor and compare this number of pulses with the value **Tics per revolution** (see 5.4.11). If the

difference is greater than **ErrorVal**, Revolution sensor error flag is setting up, indicator **RT error** is lighting up (see 5.2.5) and stepper motor stops if **Stop driver if error is detected** checkbox is marked or continue moving in other way. If Revolution sensor error flag is setting up, it remains up until it will be reset by **Reset error status** button.

When revolution sensor is enabled error value is accumulated for all movements that happened (for right direction and left direction separately) and Revolution sensor error flag will be set if accumulated error value exceeds **ErrorVal**. It provides accuracy about **ErrorVal/64** steps for all movements when revolution sensor is enabled. Configuration of Revolution sensor is possible by software (for example, see 5.4.11).

Controller inputs for Revolution sensor is shown on Table 3.

	Pin number on	
External sensor	Stepper motor connector	Multi-purpose 40 pin connector P1
Revolution sensor	14	18

Table 3. Connection of Revolution sensor

2.2.7 Encoder

Encoder is a sensor of mechanical motion. Quadrature encoder is intended for direct measurement of stepper motor shaft position. It translates shaft angle into two shifted electrical signal on CH A and CH B outputs (see Figure 27). Mechanical action of optical quadrature encoder is illustrated on Figure 28. Two optocouples are used. Emitter and detector are set near the disk, by different sides. When slit is positioned on emitter and detector then optocouple is opened (optocouple output signal is low). If slit closes emitter from detector then optocouple is closed (optocouple output signal is high). Optocouple signals usually are shifted by half of encoder cycle (see Figure 27).

The main characteristic of quadrature encoder is a number of cycles per revolution (CPR). Most common encoders have 100-1000 CPR. Each cycle can be decoded into 1, 2 or 4 codes, referred to as X1, X2 or X4 resolution multiplication. This controller used X2 resolution multiplication.

Connection diagram of encoder to multi-purpose 40 pin connector P1 is shown on Figure 30, to Stepper motor connector is shown on Figure 29.

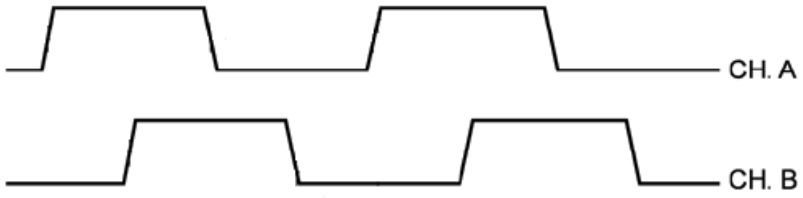


Figure 27. Electrical signal on CH A and CH B outputs of quadrature encoder

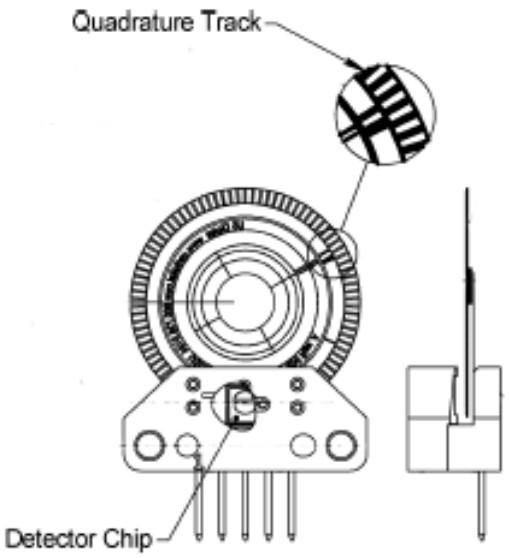


Figure 28. Optical quadrature encoder mechanical alignment

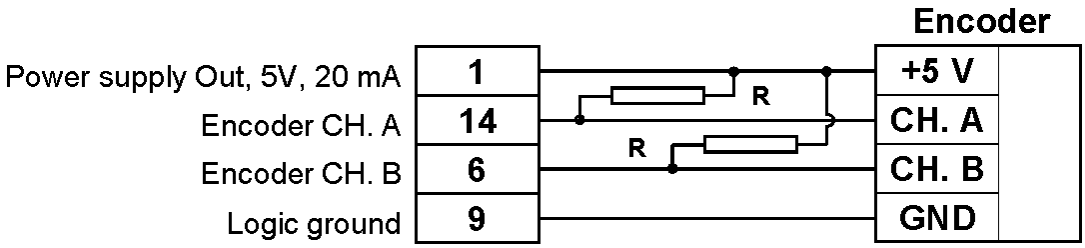


Figure 29. Encoder connection to Stepper motor connector

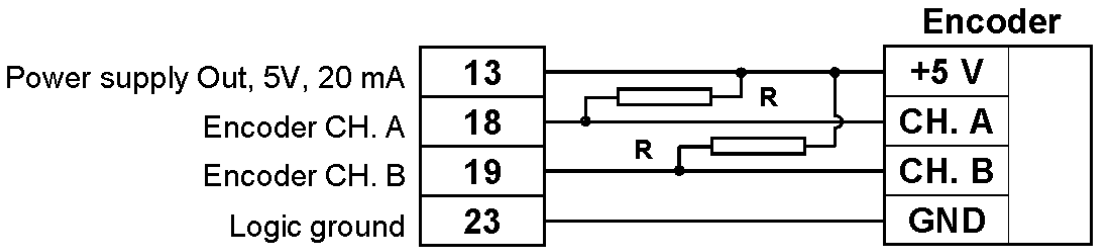


Figure 30. Encoder connection to multi-purpose 40 pin connector P1

Note: Encoder share one input on board with revolution sensor and one input with synchronization in. Therefore makes sure that revolution sensor and synchronization input are disconnected from corresponding devices before using the quadrature encoder.

Note: Usually encoder module outputs are open collector with internal pull-up resistors. When driving a cable, they will provide good high-to-low transition times. But low-to-high times will stretch out in proportion to the cable length and capacitance. If a cable driver is not used, you can add a pull-up resistor R=1-5K Ohm (see Figure 29, Figure 30) to +5 Volts on each output to improve these rise times.

Note: For correct work of encoder at high speed it is necessary to use special version of 8SMC1-USBh controller board. Ask this option your local dealer.

2.2.8 Synchronization Out

8SMC1-USBh can serve as an external synchronization for other devices. It can produce TTL output pulse every **N** steps, where **N** is controlled by software. It is usually used for starting some repeated action like a routine measurement.

Synchronization output sends a TTL signal. Maximum output high level current is 5 mA. Active state is high. Duration is programmable (see 5.4.13).

Connection diagram to multi-purpose 40 pin connector P1 is shown on Figure 31, to Stepper motor connector is shown on Figure 32.

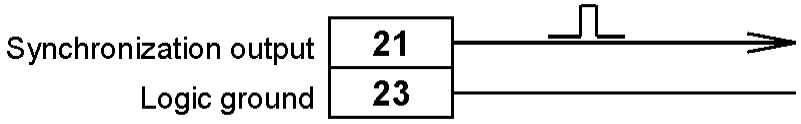


Figure 31. Synchronization output on multi-purpose 40 pin connector P1

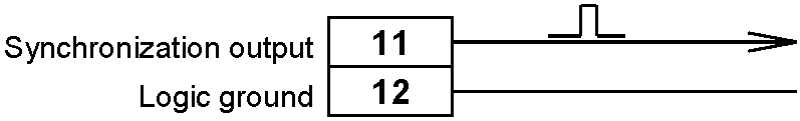


Figure 32. Synchronization output on Stepper motor connector

Note: It is possible to use synchronization output as general purpose digital output. Use USMC_SetMode() function (see Chapter 0) or Set Mode (uSMC) VI (see Chapter 6), or SMCVieW application (see Chapter 5) for control it.

2.2.9 Synchronization In

8SMC1-USBh can work in pulse triggering mode. In this case controller waits synchronization pulse then executes predefined movement. 8SMC1-USBh can shift stepper motor by predefined number of steps only after first synchronization pulse or on every synchronization pulse. It is usually used for starting predefined movement after each measurement by synchronization pulse from measuring instrument. Wide range of devices may use as a synchronization pulse generator. For example, synchronization outputs of other 8SMC1-USBh may be used as a synchronization pulse generator.

Synchronization input receives TTL signal. Required input current is 1 mA. Active state is high.

Connection diagram to multi-purpose 40 pin connector P1 is shown on Figure 33, to Stepper motor connector is shown on Figure 34.

Note: When synchronization input mode enabled, press **Start** button in SMCVieW (see 5.2.3) or execute Set Mode (uSMC).vi in VI's library (see 6.1.4), or run USMC_Start in USMCDLL.dll (see 7.5.11). Then controller starts waiting input synchronization pulse.

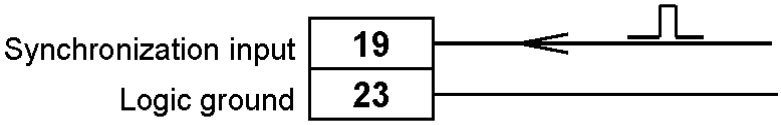


Figure 33. Synchronization input on multi-purpose 40 pin connector P1

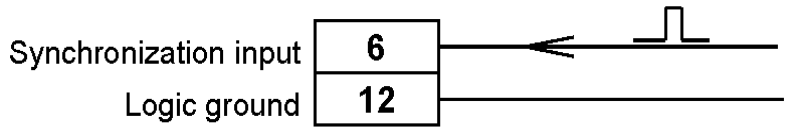


Figure 34. Synchronization input of Stepper motor connector

2.2.10 Manual control knobs

Manual control is possible on 8SMC1-USBh. There are inputs for two knobs: "Up" and "Down". Knob "Up" is intended for moving positioner "Up" (step counter in 8SMC1-USBh is incremented), Knob "Down" is intended for moving positioner "Down" (step counter in 8SMC1-USBh is decremented).

Knobs are usually used for manual precise positioning of stage, for example in calibrating. Another common application of knobs – control of stepper motors without computer.

When knob is pushed positioner makes one step (or microstep, depends on current resolution). If knob stays pushed during $\mathbf{T_{bto1}}$ (see 5.4.8 for more information) stepper motor starts rotation with $\mathbf{V_{bp1}}$ speed. If knob stays pushed during $\mathbf{T_{bto1}+T_{bto2}}$ stepper motor increases speed up to $\mathbf{V_{bp2}}$. If knob stays pushed during $\mathbf{T_{bto1}+T_{bto2}+T_{bto3}}$ stepper motor increases speed up to $\mathbf{V_{bp3}}$. If knob stays pushed during $\mathbf{T_{bto1}+T_{bto2}+T_{bto3}+T_{bto4}}$ stepper motor increases speed up to $\mathbf{V_{bp4}}$. If knob is released stepper motor stops rotation immediately.

If two knobs are pushed and held during **Reset timeout** (see 5.4.8) then 8SMC1-USBh resets and shifts to null position (in ticks).

All speeds and timeouts are programmable.

Knobs are connected to 8SMC1-USBh according to 2.2.3. Controller inputs for each knob are shown on Table 4.

	Pin number on	
External sensor	Stepper motor connector	Multi-purpose 40 pin connector P1
Knob Up	-	15
Knob Down	-	16

Table 4. Connection of knobs

2.2.11 Local indication LEDs

Local indication is possible on 8SMC1-USBh. There are outputs for three LEDs: "Up", "Down" and "Status". LED "Up" blinks when stepper motor is moving "Up" (step counter in 8SMC1-USBh is incremented) and light if Limit switch 1 is reached (or Limit switch 2, if Swap Limit switches option is enabled). LED "Down" blinks when stepper motor is moving "Down" (step counter in 8SMC1-USBh is decremented) and light if Limit switch 2 is reached (or Limit switch 1, if Swap Limit switches option is enabled). For more information about Limit switches see 2.2.4. LED "Status" blinks when 8SMC1-USBh is powered and works properly but USB connection to 8SMC1-USBh software driver on computer is unavailable. LED "Status" light when 8SMC1-USBh works properly and USB connection to 8SMC1-USBh software driver is established.

Output current for LED is limited by 10 mA. Any LEDs with rated current 5-10 mA may be used for local indication. LEDs wiring on Multi-purpose 40 pin connector P1 is shown on Figure 35.

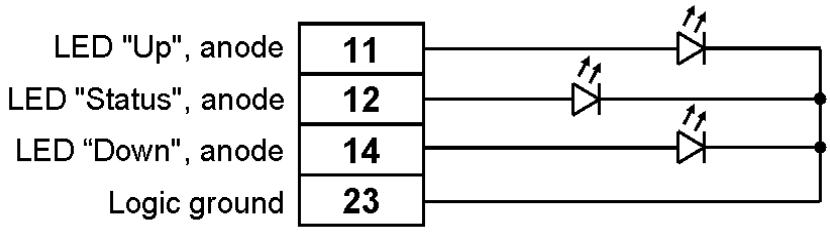


Figure 35. LED wiring on Multi-purpose 40 pin connector P1

2.2.12 USB In/Out

There are two ways of communication to 8SMC1-USBh board: by multi-purpose 40 pin connector (convenient for embedded applications) or by special USB In and USB Out connectors (see 2.1).

USB In (USB type B connector) is used for connection with computer or higher USB hub. Embedded 3-port USB hub allows cascading of the 8SMC1-USBh boards. USB Out (two USB type A connectors) allow the connection of other USB devices or lower USB hubs to this 8SMC1-USBh board.

Caution: Use only operable USB cables! Defective USB cable may be a cause of 8SMC1-USBh malfunction.

USB pins on multi-purpose 40 pin connector may be useful for embedded multi-axis applications. No bulky USB connectors are necessary. Only one ribbon cable is used.

Pinout of USB ports on multi-purpose 40 pin connector P1is shown on Table 5.

	Pin number on Multi-purpose 40 pin connector P1		
Pin description	USB In	USB Out 1	USB Out 2
D+	37	33	29
D-	38	34	30
GND	39	35	31
+5V	40	36	32

Table 5. USB ports pinout on Multi-purpose 40 pin connector P1

3 Specifications

3.1 Electrical

DC input voltage, double supply	
Supply for board logic	7-12V, up to 80 mA
Supply for stepping motor	7-40V, up to 1.5A
Power dissipation	up to 6 W
PWM chopper type current control	
Chopping frequency	45 kHz
Maximum average phase current	1.5A
Maximum peak phase current	2.5A
Maximum output voltage	38V
Protections	
Short circuit protection	Yes
Overvoltage protection	Yes
Reverse supply protection	Yes
Temperature protection	Yes, adjustable
Programmable inputs	
Limit switches	2
Emergency limit switch	1, nonprogrammable
Revolution sensor	1
Local control knobs	2
Synchronization	1
Programmable outputs	
Local indication LEDs	3
Synchronization	1

3.2 Motion

Resolution	full step, half step, 1/4, 1/8
Speed	programmable, 2-5000 step/s
Position counter	-2.147.483.647 – 2.147.483.647
Accel and decel ramps	programmable
Soft start/stop mode	programmable

3.3 Remote control

Communication Protocol	USB 1.1
Communications baud rate	up to 12 Mbps
Max devices per USB host	30

3.4 Mechanical

Dimensions of 8SMC1-USBh are shown below. Operating temperature range: 0-70 °C. Heatsink (see Figure 37) may be required to maintain temperature range.

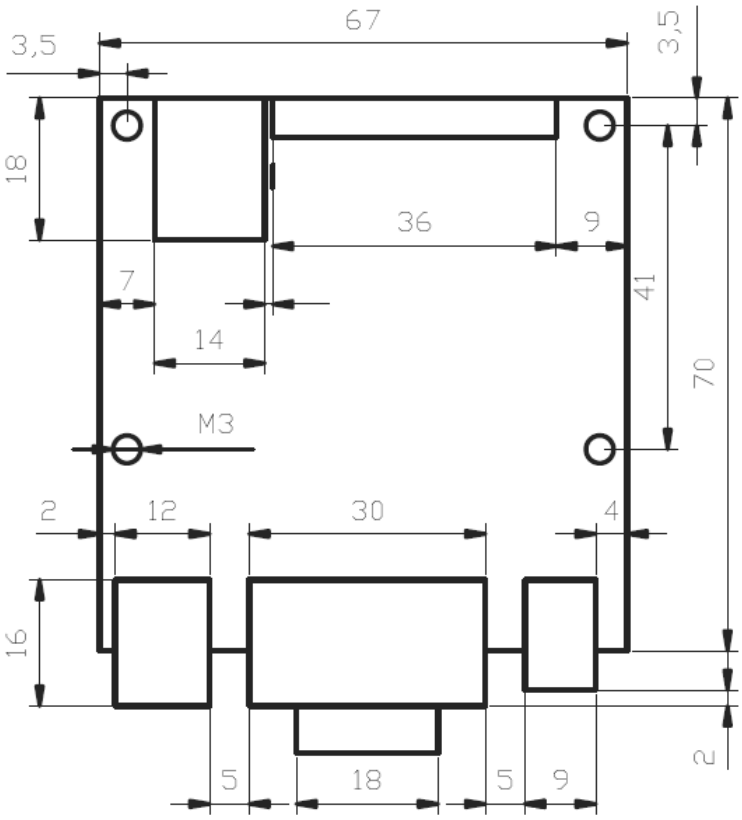


Figure 36. Dimensions of 8SMC1-USBh board

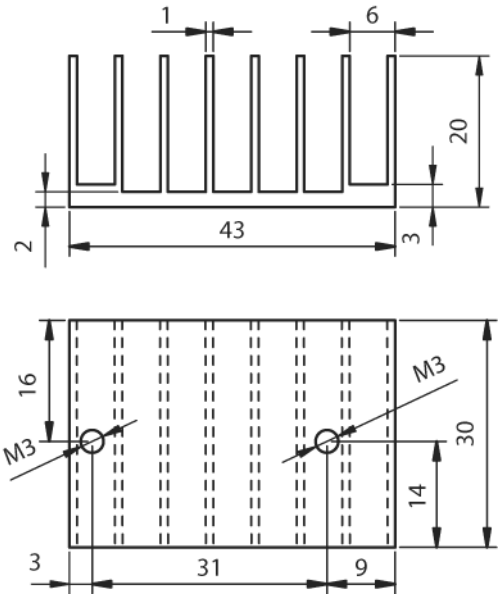


Figure 37. Heatsink for 8SMC1-USBh

3.5 Wiring diagram

3.5.1 Wiring diagram for multi-purpose 40 pin connector

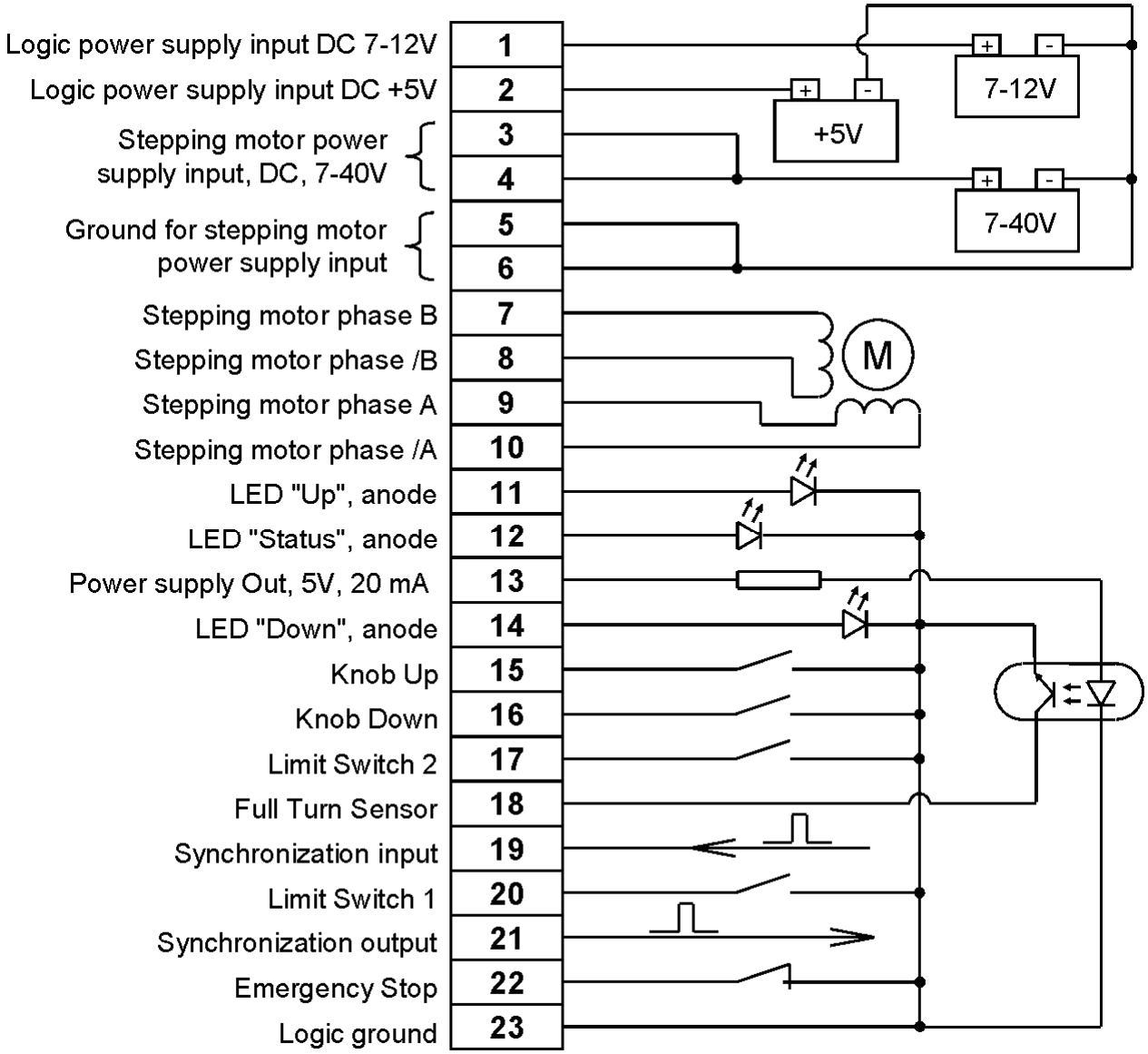


Figure 38. Wiring diagram for multi-purpose 40 pin connector P1

On Figure 38 three different power supplies are shown. Only two or one power supply is needed of course. Choose a convenient way of power supply connection according to 2.2.1. For more details see Functional description (Chapter 2).

3.5.2 Wiring diagram for Stepper motor connector

On Figure 39 wiring diagram for Stepper motor connector is shown. Choose a convenient way for all other connections according to Functional description (Chapter 2).

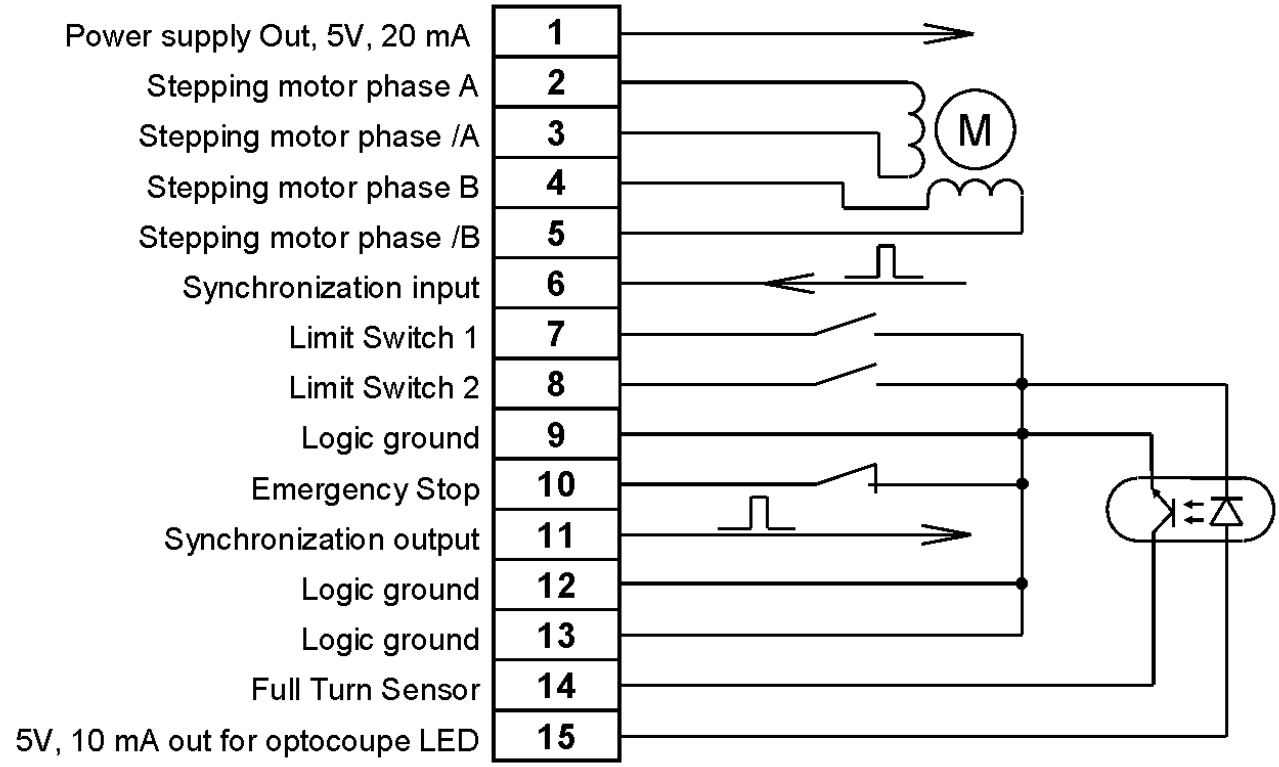


Figure 39. Wiring diagram for Stepper motor connector

3.6 EMC

8SMC1-USBh 1.5A Microstep Driver with USB Interface conforms to electromagnetic immunity requirements listed in LST EN 61000-6-2:2002 for industrial environments in accordance with standards LST EN 61000-4-2+A1+A2:2002, LST EN 61000-4-4:2002, LST EN 61000-4-4/A1:2002, LST EN 61000-4-4/A2:2002.

4 Installation

Installation procedure consists of several steps:

o SMCVieW application and NI-VISA USB driver installation (see 4.1.1).

Note: NI-VISA USB driver is required for correct work of SMCVieW and all LabVieW based software.

o Hardware installation of 8SMC1-USBh controller (see 4.2).
o First start (see 4.3).

If it is supposed to program 8SMC1-USBh controller on C, C++, Basic, Delphi, MatLab or other languages exclude LabVieW install MicroSMC driver (see paragraph 4.1.2).

If it is supposed to control 8SMC1-USBh using mobile devices working under Microsoft Windows Mobile 5.0 and higher install MicroSMC for WM driver (see paragraph 4.1.3).

4.1 Software installation

Next installation process will be shown for Windows Vista operating system. Software installation processes under Windows XP/2000 is just the same.

Make sure that all 8SMC-USBh devices are unplugged and switched off. Turn on computer. Insert the CD-ROM labeled "8SMC-USBh" in your CD-ROM drive or unzip "8SMC1-USB(h) Soft.zip" software package and open main folder. Open "SMCVieW and VISA driver" folder and run setup.exe.

4.1.1 SMCVieW installation

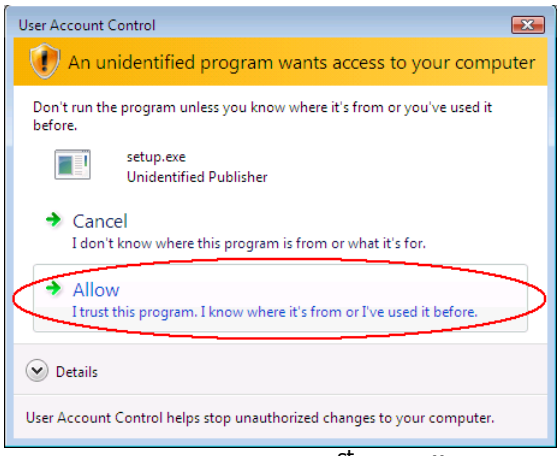


Figure 40. SMCVieW 1st installation screen

o Press the "Allow" button to continue the installation (see Figure 40).

Note: Screen on Figure 40 appear on Windows Vista with enabled UAC only. On Windows Vista with disabled UAC or Windows XP you will see Figure 41 after launch setup.exe.

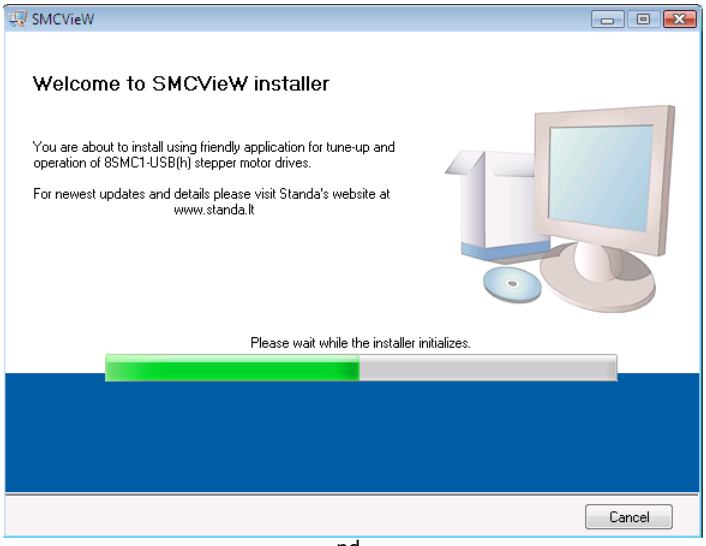


Figure 41. SMCVieW 2nd installation screen

o Please wait while the installer initializes (see Figure 41).

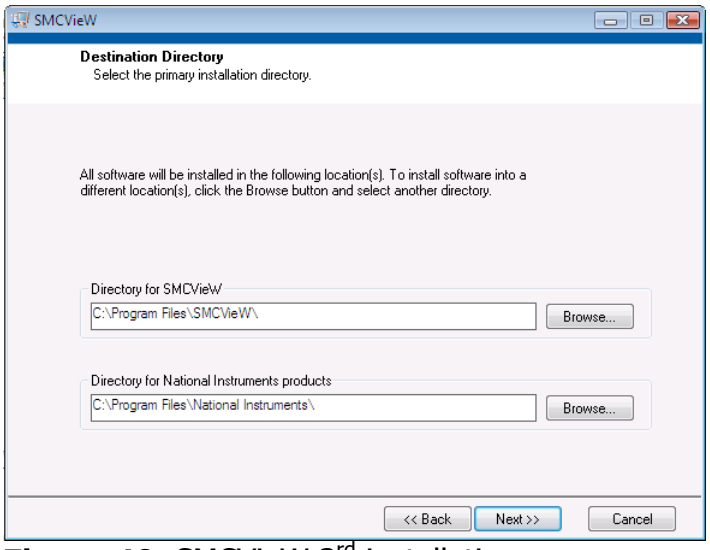


Figure 42. SMCVieW 3rd installation screen

o Select destination folders and press the "Next>" button (see Figure 42).

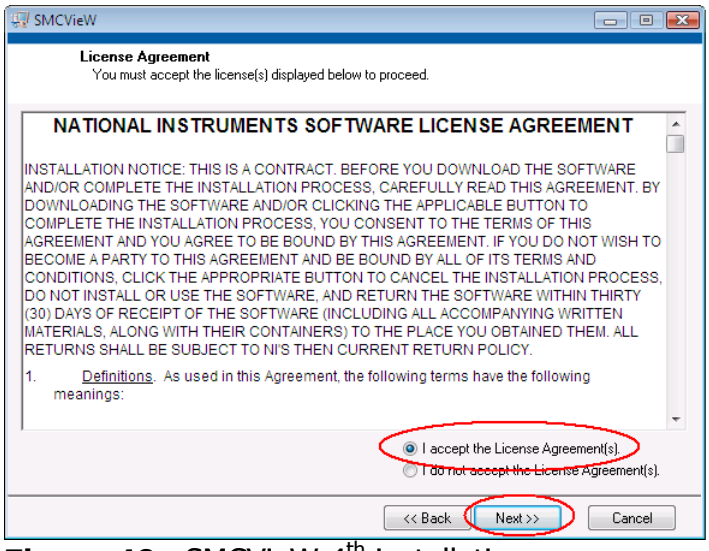


Figure 43. SMCVieW 4th installation screen

o Please choose "I accept..." and press the "Next>" button (see Figure 43).

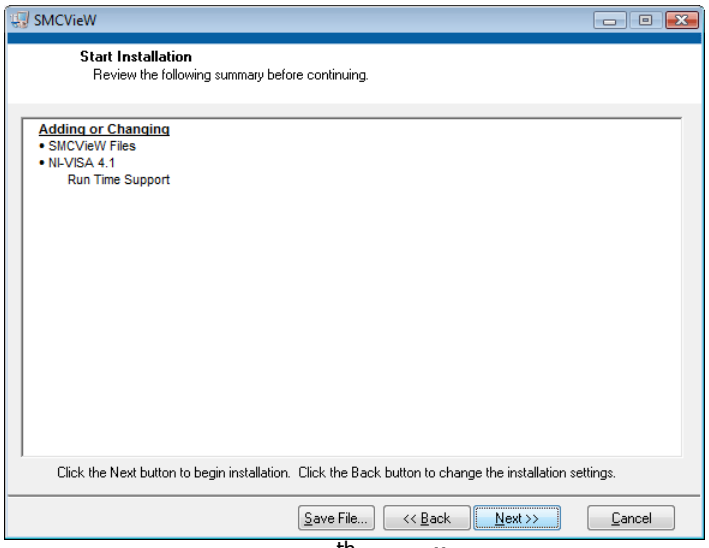


Figure 44. SMCVieW 5th installation screen

o Please press the "Next>" button and wait until installation process will be done (see Figure 44, Figure 45).

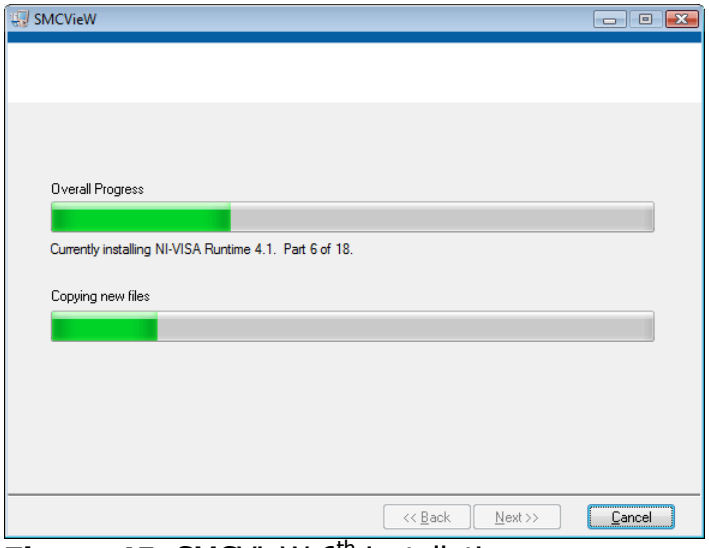


Figure 45. SMCVieW 6th installation screen

4.1.2 MicroSMC driver installation

Note: Install MicroSMC driver if suppose to program 8SMC1-USBh controller in C, C++, Basic, Delphi, MatLab or other languages excluding LabVieW. For LabVieW based applications, for example SMCVieW, only NI VISA driver is necessary.

Note: Next installation process will be shown for Windows XP operating system. Software installation processes under Windows Vista/2000 is just the same. The only difference on Windows Vista is presence screen like Figure 40 at the beginning of installation.

- Open MicroSMC folder on 8SMC1-USBh installation disk. And click on setup.exe file. Next screen will be shown (see Figure 46). Then press "Next>" button.
- On the next screen (see Figure 47) choose installation folder and press "Next>" button.
- Installer is ready to install MicroSMC on your computer (see Figure 48). Press "Next>" button.

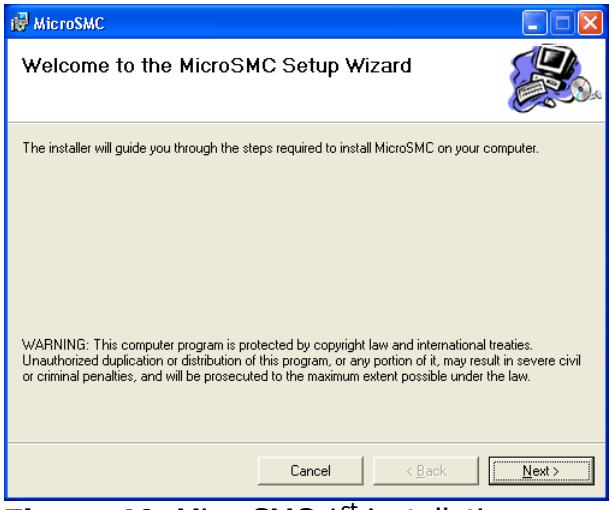


Figure 46. MicroSMC 1st installation screen

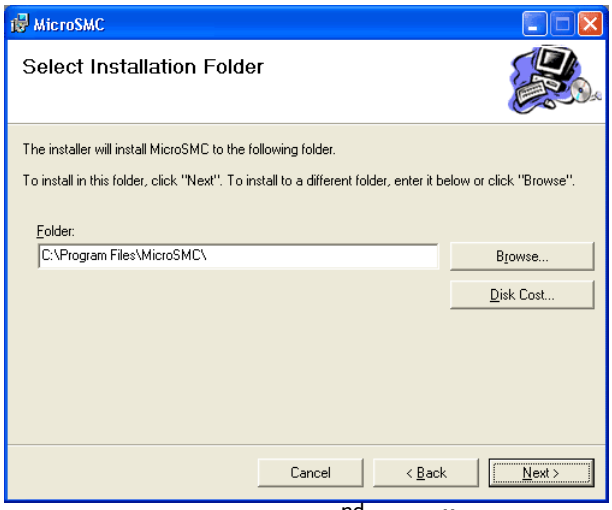


Figure 47. MicroSMC 2nd installation screen

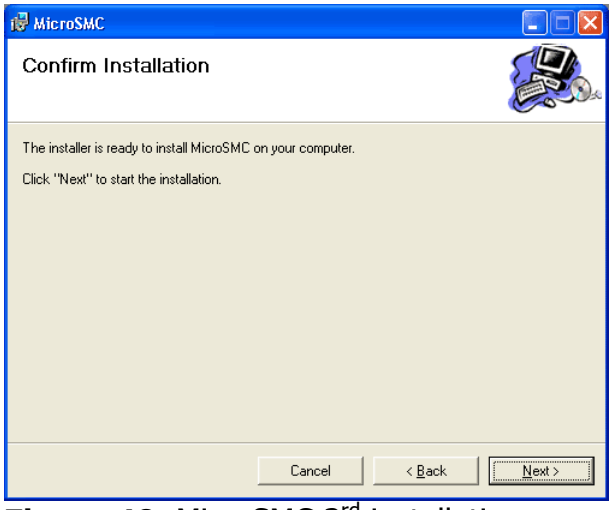


Figure 48. MicroSMC 3rd installation screen

- Wait until installation process will be done (see Figure 49).
- Read brief instructions on using the MicroSMC driver and dll (see Figure 50).

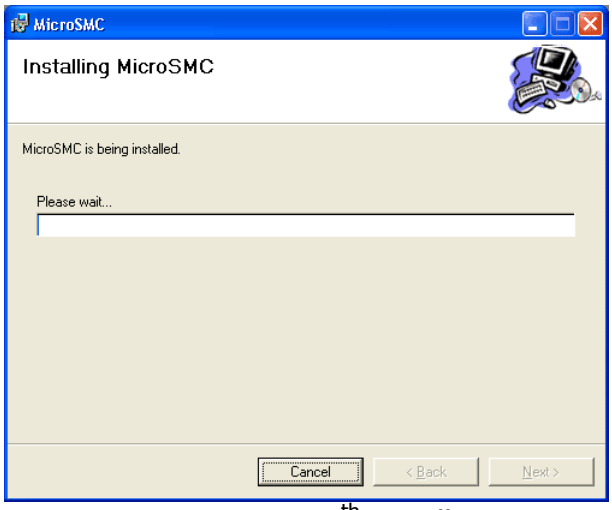


Figure 49. MicroSMC 4th installation screen

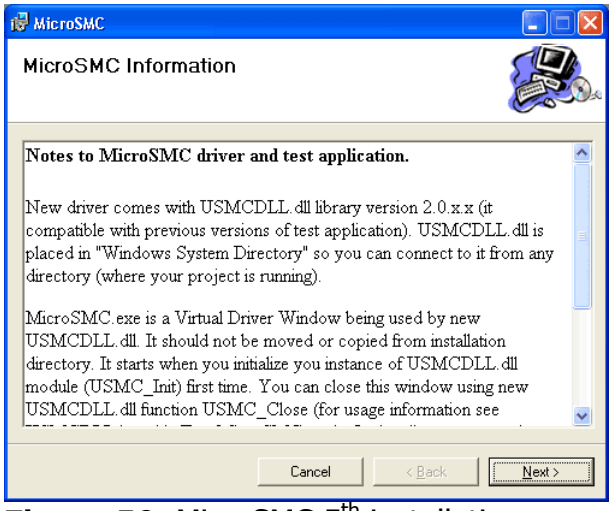


Figure 50. MicroSMC 5th installation screen

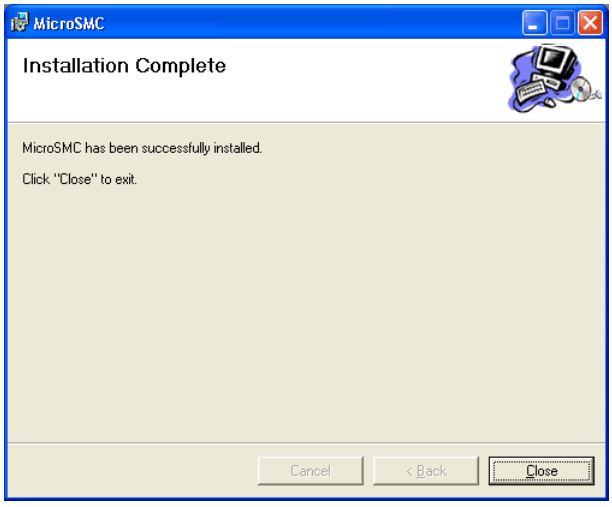


Figure 51. MicroSMC last installation screen

- Press “Close” to complete the installation (see Figure 51).

4.1.3 MicroSMC for WM installation

MicroSMC for WM is a USB host driver for 8SMC1-USBh controllers for mobile devices. It is designed for control and monitor 8SMC1-USBh controllers using mobile devices with Microsoft Windows Mobile 5.0 or higher operation system. Only presence a USB host port on your mobile device is necessary. Functionality of USMCDLL.dll for Microsoft Windows Mobile 5.0 is just the same as USMCDLL.dll for Microsoft Windows Vista/XP/2000 (for more details see chapter 7).

Note: Some mobile devices have only USB port for connection to host computer. Make sure that your mobile device has USB host port (It is used generally for connection to flash memory, keyboards etc.).

Microsoft ActiveSync is required for installation. Install ActiveSync that comes with your mobile device or download and install the latest version of Microsoft ActiveSync from www.mocrosoft.com. Then plug your pocket PC to host computer and wait when connection will be established.

MicroSMC for WM host PC installation

Note: Next installation process will be shown for Windows XP operating system. Software installation processes under Windows Vista/2000 is just the same. The only difference on Windows Vista is presence screen like Figure 40 at the beginning of installation.

o Open "MicroSMC for WM" folder and run setup.exe. The first installation screen of MicroSMC for WM installation procedure will appear (see Figure 52).

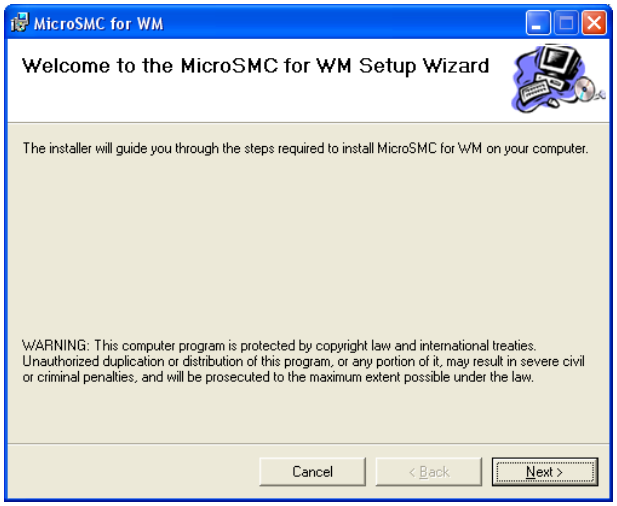


Figure 52. MicroSMC for WM host PC installation 1st screen

Note: Microsoft .NET Framework 2.0 is required for MicroSMC for WM host PC installation. This package (setup file dotnetfx.exe) is required to run all applications developed to target the .NET Framework 2.0 and usually installed by Windows update automatically. If it is absent on your PC download it for free from www.microsoft.com before MicroSMC for WM host PC installation.

o Press the "Next" button to continue installation.

Figure 53. MicroSMC for WM host PC installation 2nd screen

o Read carefully this screen (see Figure 53) and press the "Next>" button.

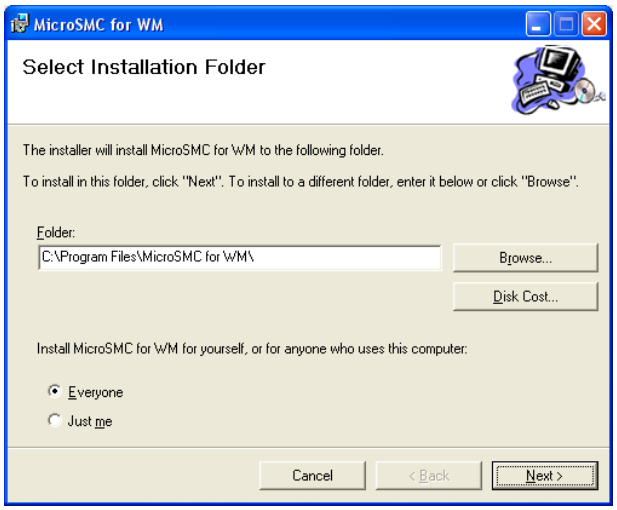


Figure 54. MicroSMC for WM host PC installation 3rd screen

o Select destination folder and press the "Next>" button (see Figure 54).

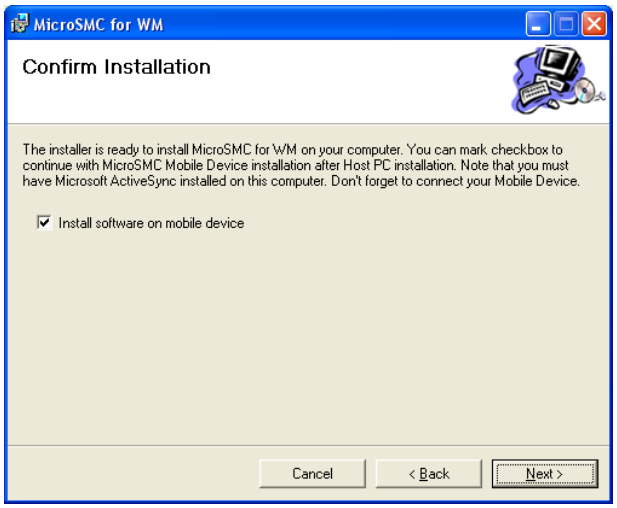


Figure 55. MicroSMC for WM host PC installation 4th screen

o Mark "Install software on mobile device" checkbox for installation software on mobile device and press the "Next>" button (see Figure 55).

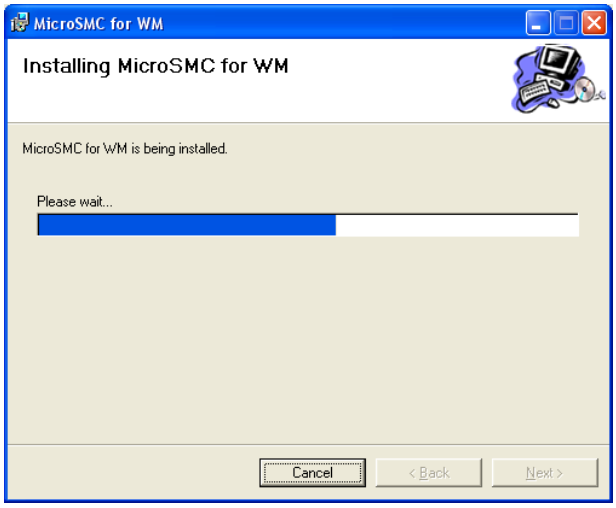


Figure 56. MicroSMC for WM host PC installation 5th screen

o Wait until MicroSMC for WM host PC installation process will be done (see Figure 56).

MicroSMC for WM Mobile Device installation

Note: Make sure that ActiveSync is running and connection with mobile device is established.

o After the previous step (see Figure 56) ActiveSync Application Manager window will appear (see Figure 57). If nothing happens navigate on host PC to MicroSMC for WM installation folder (usually `C:\Program Files\MicroSMC for WM\`) and run CEAppMgrSetup.exe application.

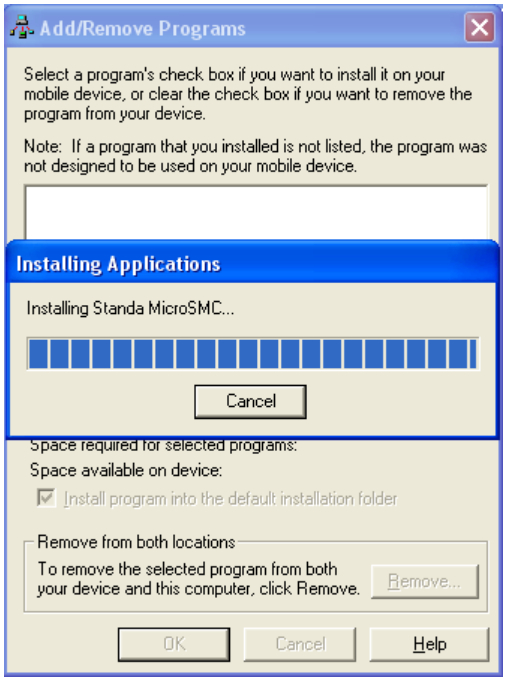


Figure 57. MicroSMC for WM Mobile Device installation 1st screen

o Wait until MicroSMC Mobile Device installation process will be done (see Figure 57).

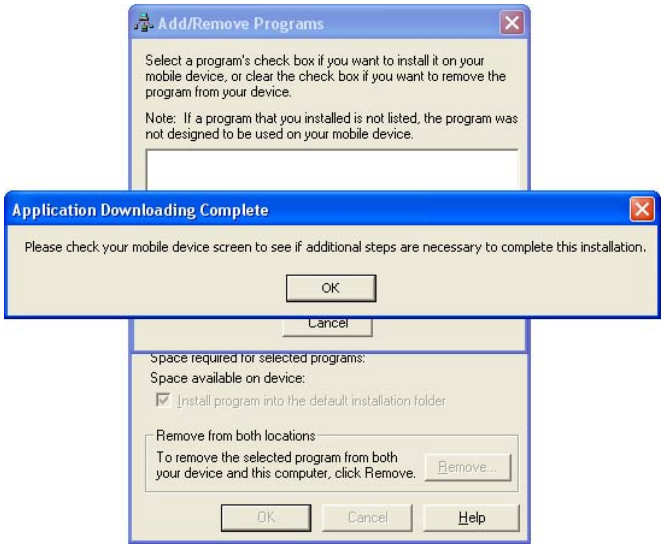


Figure 58. MicroSMC for WM Mobile Device installation 2nd screen

o Press "OK" button to finish installation process (see Figure 58)

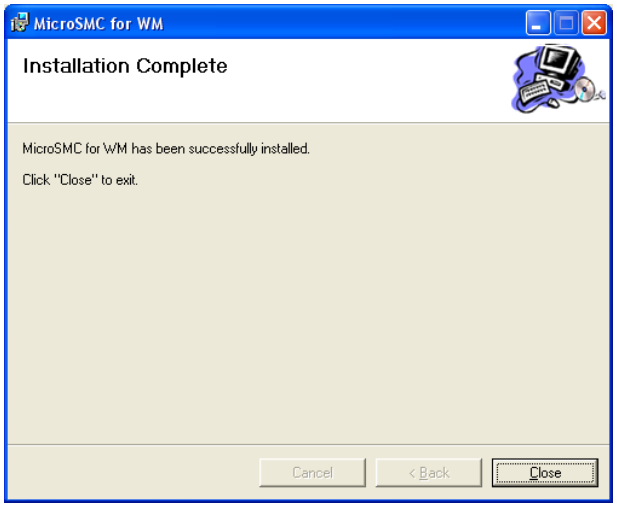


Figure 59. MicroSMC for WM host PC installation last screen

o Click "Close" to finish all installation process (see Figure 59)

Now take a look on your mobile device screen. There should appear notification that installation was completed. Click "OK" to finish process.

Note: For removing MicroSMC for WM from host PC navigate Start -> Settings-> Control Panel -> Add or Remove Programs, find MicroSMC for WM and click Remove. For removing MicroSMC for WM from mobile device navigate on mobile device Start -> Settings -> System -> Remove programs, pick MicroSMC for WM and click Remove.

One other way: on host PC right click on Microsoft ActiveSync icon in system tray, choose Open Microsoft ActiveSync. Navigate Tools -> Add/Remove Programs, pick MicroSMC and click Remove.

4.2 Hardware installation

Caution: Read chapter 2 before hardware installation!

- Make sure that power of 8SMC1-USBh is off.
- Verify that current sense resistors on 8SMC1-USBh board match the rated current that is supposed to be used with stepper motor. Change resistors if necessary (see 2.1 and 4.2.1). Stepper motor rated current is written on the motor nameplate.
- Set the emergency stop jumper (see 2.2.5 for more information).
- Connect the 8SMC1-USBh according to 3.5.
- Do not connect USB cable to PC.

4.2.1 Current sense resistors

Current sense resistors are connected in series with stepper motor windings. Power driver use current sense resistors for holding the right current on the stepper motor windings. If winding is under voltage power driver keep the current that lead to voltage drop of 0.8V on current sense resistor. Therefore rated resistance and power dissipation are calculated by the following way:

$R = \frac{0.8}{I}$ $P = I^2 \cdot R$, where

I - rated current of stepper motor [A],
R - calculated resistance of current sense resistors [Ohm],
P - calculated power dissipation of current sense resistors [W].

In following table certain values of rated currents for stepper motor and resistance for current sense resistors are shown:

I, A	*R*, Ohm	*P*, W
0.1	8.2	0.08
0.16	5.1	0.13
0.21	3.9	0.17
0.24	3.3	0.19
0.3	2.7	0.24
0.4	2	0.32
0.5	1.6	0.4
0.62	1.3	0.5
0.8	1	0.64
1	0.8	0.8
1.33	0.6	1.1
1.6	0.5	1.3

Table 6. Current sense resistors. Power dissipation and current.

Rated power dissipation of chosen current sense resistors must be no less than the corresponding value from Table 6.

Note: If resistance of current sense resistors is smaller than it is required then real current on windings will be larger than rated current. It may lead to stepper motor overheating and nonuniform motion in microstep modes.

Note: If resistance of current sense resistors is larger than it is required then real current on windings will be smaller than rated current. It will decrease the operating torque of stepper motor and maximal speed.

Note: If rated power dissipation of current sense resistors is smaller than it is required then overheating and breakdown of current sense resistors are possible.

4.3 First start on Windows XP

- Turn the power of stepping motor controller on.
- Check that the Status LED is blinking (see 2.2.11 for more information). It means that controller is powered, self tested and ready to work, but USB cable is not plugged or USB driver is not installed correctly.

Figure 60. Hardware wizard 1st screen

- Push and hold Up or Down knob and make sure that stepping motor moves. Omit this step if you do not intend to use local control and indication.
- Connect a USB cable to PC.
- Wait while Windows will find a new hardware and install driver for it.
- Depending on a Windows version it may pass automatically or demand to pass several steps:
- The main window of Hardware Wizard will be appeared (see Figure 60). Choose ”No, not this time” and press the “Next>” button. On some old version of WindowsXP this screen may be skipped.

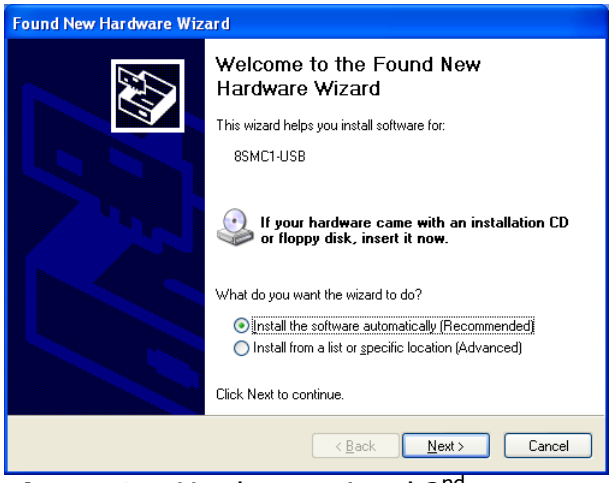


Figure 61. Hardware wizard 2nd screen

- Then choose (see Figure 61) Install the software automatically (Recommended) and press the "Next>" button.

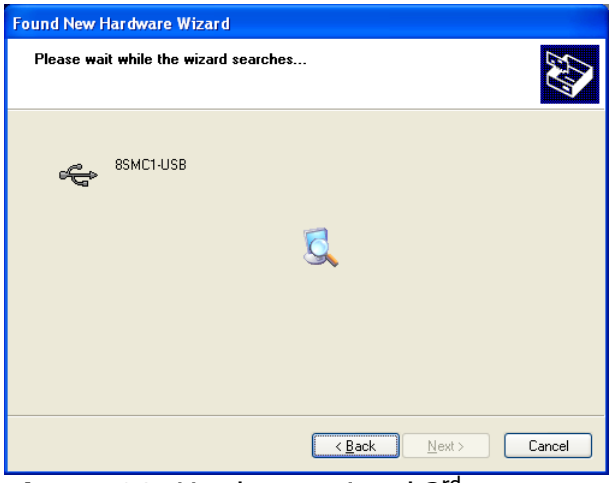


Figure 62. Hardware wizard 3rd screen

- Wait until installation will be done (see Figure 62) and press "Finish" button (see Figure 64).

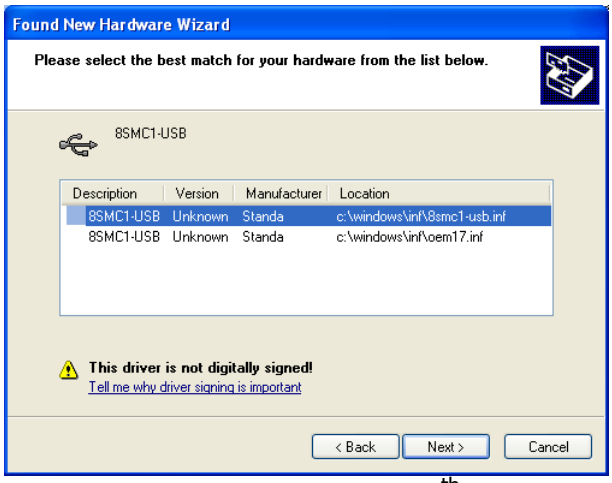


Figure 63. Hardware wizard 4th screen

- If other 8SMC1-USBh controllers were installed on computer earlier, screen Figure 63 may appear. Choose 8SMC1-USB.inf and press "Next" button.

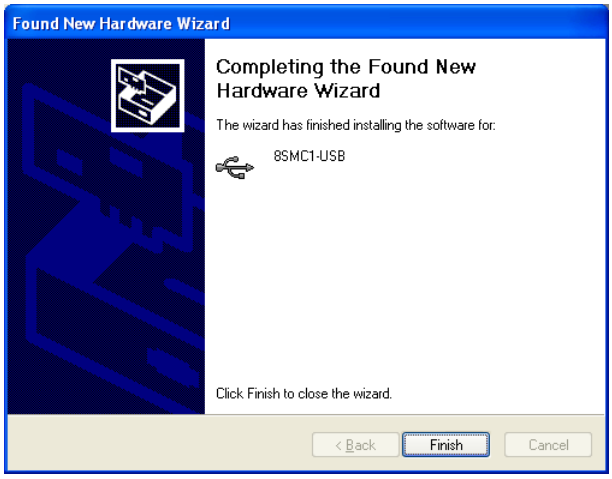


Figure 64. Hardware wizard last screen

- Check that the Status LED is alight. It means that VISA driver have found controller.
- Now the 8SMC1-USBh device is installed properly and ready to use.
- Make sure that the emergency stop jumper is set in OFF position (see 2.1 and 2.2.5).

Note: If you disconnect the USB cable of 8SMC1-USBh and connect it again to another USB port then installation of 8SMC1-USBh driver may be required once more. In this case choose "Install software automatically".

Note: If after some actions connection with 8SMC1-USBh is lost: disconnect the USB cable of 8SMC1-USBh and connect it again to the same USB port.

4.4 First start on Windows Vista

- Turn the power of stepping motor controller on.
- Check that the Status LED is blinking (see 2.2.11 for more information). It means that controller is powered, self tested and ready to work, but USB cable is not plugged or USB driver is not installed correctly.

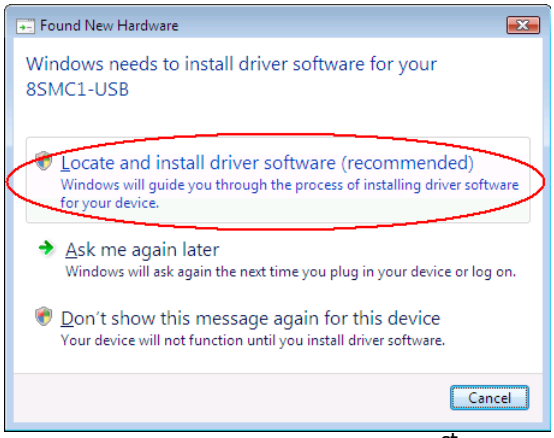


Figure 65. Hardware wizard 1st screen

- Push and hold Up or Down knob and make sure that stepping motor moves. Omit this step if you do not intend to use local control and indication.

- Connect a USB cable to PC.
- Wait while Windows will find a new hardware and install driver for it.
- Depending on a Windows version it may pass automatically or demand to pass several steps:
- The first window of Hardware Wizard will be appeared (see Figure 65). Choose ”Locate and install driver software”.

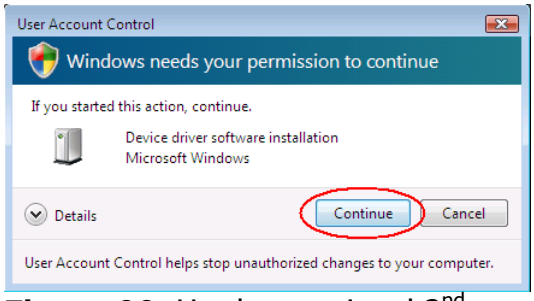


Figure 66. Hardware wizard 2nd screen

- Then choose “Continue” (see Figure 66).

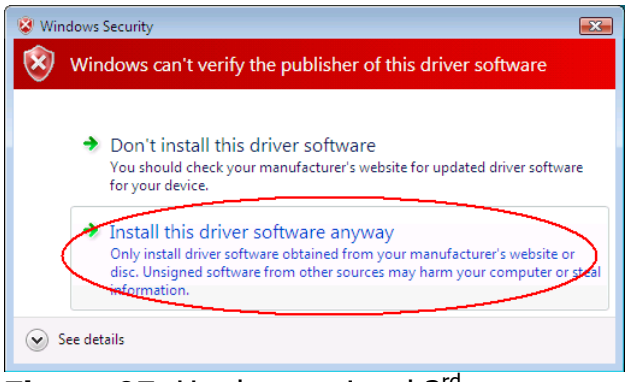


Figure 67. Hardware wizard 3rd screen

- Choose ”Install this driver software anyway” (see Figure 67).
- Wait until installation will be done.
- Check that the Status LED is alight. It means that VISA driver have found controller.
- Now the 8SMC1-USBh device is installed properly and ready to use.
- Make sure that the emergency stop jumper is set in OFF position (see 2.1 and 2.2.5).

Note: If you disconnect the USB cable of 8SMC1-USBh and connect it again to another USB port then installation of 8SMC1-USBh driver may be required once more. In this case choose ”Locate and install driver software”.

Note: If after some actions connection with 8SMC1-USBh is lost: disconnect the USB cable of 8SMC1-USBh and connect it again to the same USB port.

4.5 Switching between NI VISA and MicroSMC drivers on Windows XP

All 8SMC1-USBh boards that have connected to PC can work either with NI VISA driver or MicroSMC driver. It is possible to install on PC several drivers for one class of USB devices (NI VISA driver and MicroSMC driver, for example) but every particular 8SMC1-USBh board

can work only with one USB driver at a time. Therefore if 8SMC1-USBh board is connected to PC using, for example, MicroSMC driver then it will not be accessible for applications that use NI VISA driver and vice versa.

For switching from NI VISA to MicroSMC driver please do the following:

- Open Device Manager (choose My Computer then right click->Properties->Hardware->Device Manager), choose your 8SMC1-USBh device (it should be connected and switched on) and remove it (press Delete button, see Figure 68). 8SMC1-USBh device will be disappeared from Device Manager list.

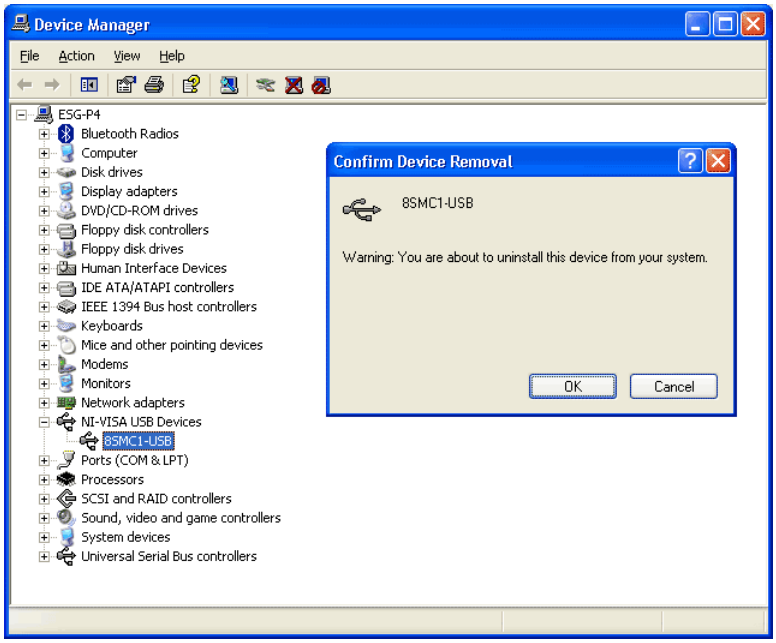


Figure 68. Uninstallation of NI VISA driver

- Then disconnect USB cable of your 8SMC1-USBh controller and connect it again. When Hardware Wizard found new hardware and will offer you to install a new driver choose "Install the software automatically" (see Figure 61) then from the list of found drivers choose 8SMC1-USB MicroSMC driver (see Figure 69).

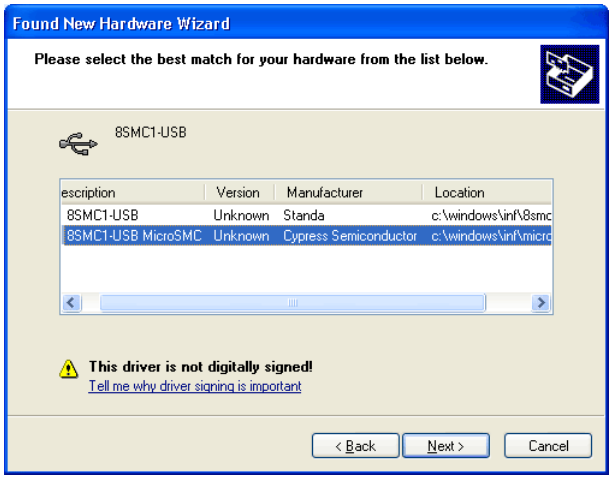


Figure 69. Choosing USB driver from list

Note: If 8SMC1-USB MicroSMC driver does not appear select ".inf" file from MicroSMC installation directory manually.

Note: SMCVieW application and all LabVieW based applications can work only with NI VISA driver.

For switching from MicroSMC to NI VISA driver please does exactly the same as it is shown above. Note that in Device Manager list MicroSMC driver is placed in "Universal Serial Bus controllers" folder (see Figure 70).

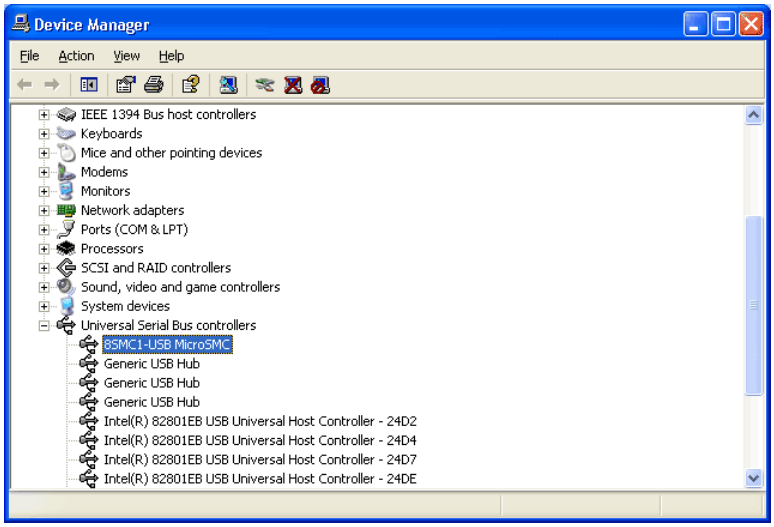


Figure 70. Position of MicroSMC driver in Device Manager list

4.6 Switching between NI VISA and MicroSMC drivers on Windows Vista

All 8SMC1-USBh boards that have connected to PC can work either with NI VISA driver or MicroSMC driver. It is possible to install on PC several drivers for one class of USB devices (NI VISA driver and MicroSMC driver, for example) but every particular 8SMC1-USBh board can work only with one USB driver at a time. Therefore if 8SMC1-USBh board is connected to PC using, for example, MicroSMC driver then it will not be accessible for applications that use NI VISA driver and vice versa.

For switching from NI VISA to MicroSMC driver please do exactly the following:

- Open Device Manager (choose Start->Control Panel->Device Manager), choose your 8SMC1-USBh device (it should be connected and switched on) and remove it (press Delete button, see Figure 71). In "Confirm Device Uninstall" window mark "Delete the driver software for this device" checkbox. Press "Ok". 8SMC1-USBh device will be disappeared from Device Manager list.

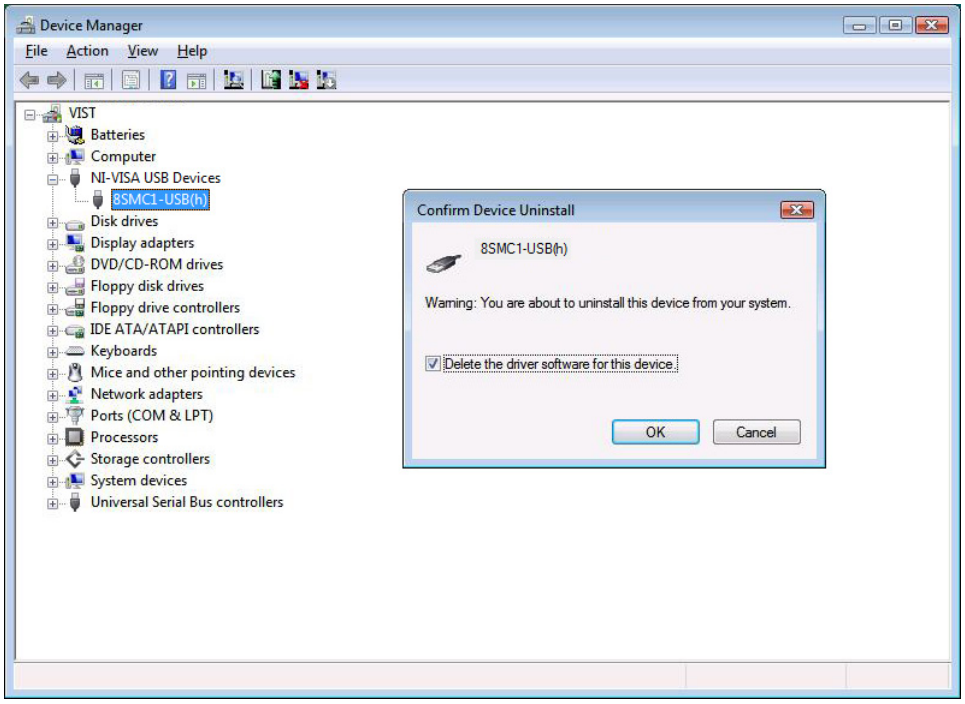


Figure 71. Uninstallation of NI VISA USB driver

o Then disconnect USB cable of 8SMC1-USBh controller and connect it again. When Hardware Wizard will found new hardware and offer you to install driver skip it choosing "Don't show this message again for this device" option (see Figure 72).

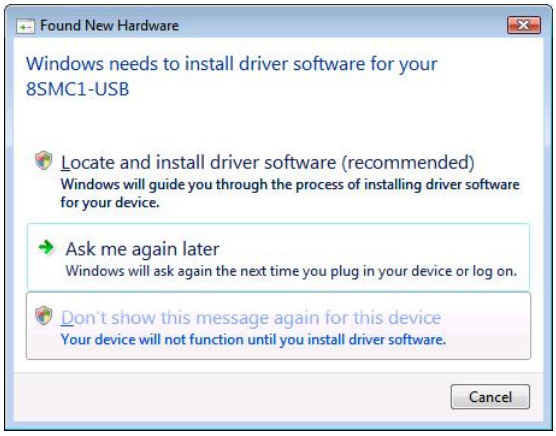


Figure 72. Skip automatic driver installation choosing "Don't show this message again for this device" option.

o Open Device Manager window, choose yellow marked 8SMC1-USB device in "Other devices" group or yellow marked "Unknown device" (it depends on Windows Vista settings), make a right click and choose "Update Driver Software…" (see Figure 73).

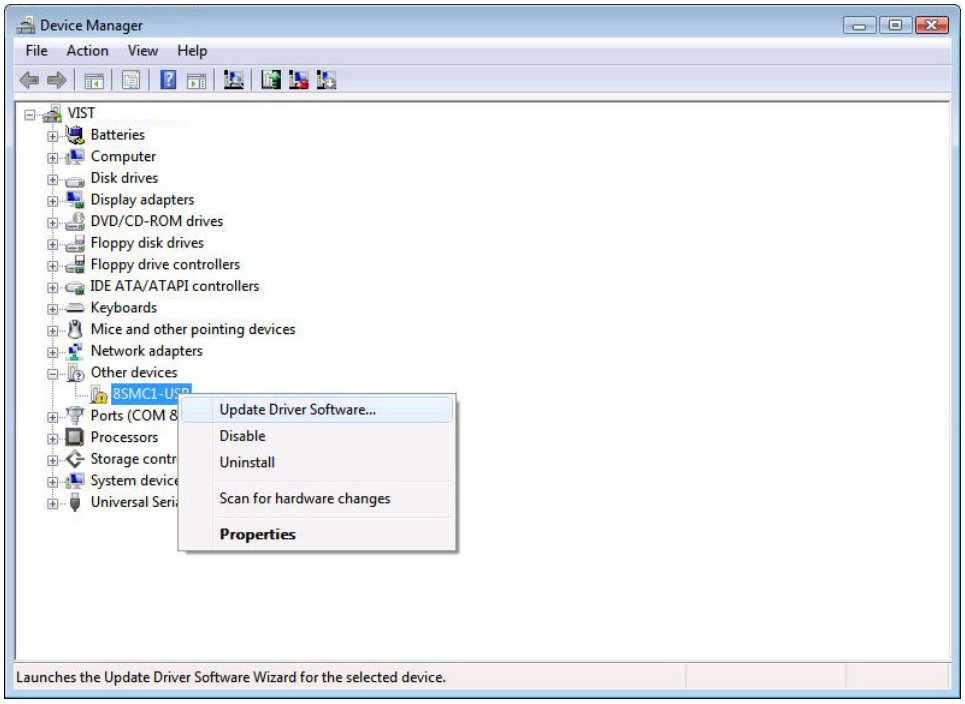


Figure 73. Update Driver Software.

o At the next screen choose ”Browse my computer for driver software” (see Figure 74).

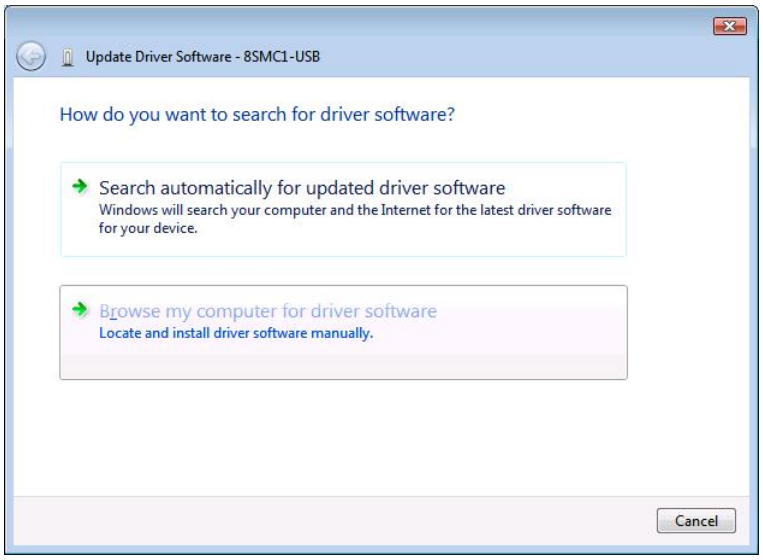


Figure 74. Browse my computer for driver software.

o Choose the folder with MicroSMC driver, ”C:\Program Files\MicroSMC\Driver” usually, and press “Next” (see Figure 75).

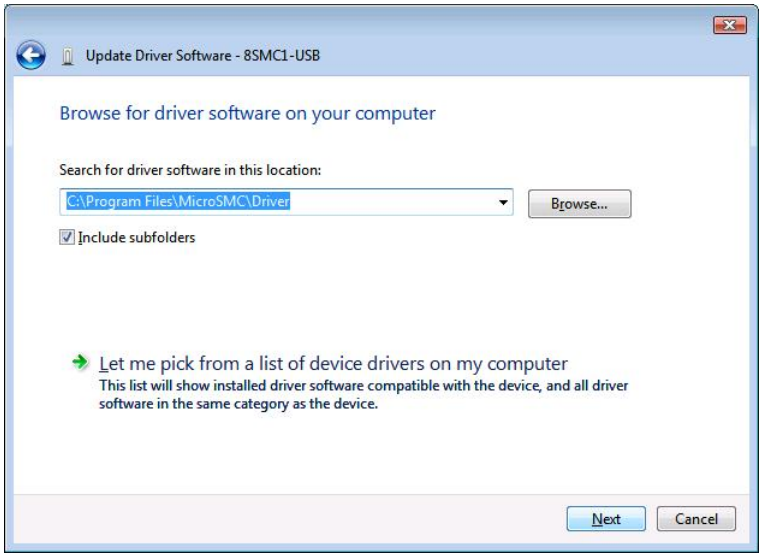


Figure 75. Choose the folder with MicroSMC driver.

- If window like Figure 67 will appear, press ”Install this driver software anyway”.
- Then new driver will be successfully installed (see Figure 76).

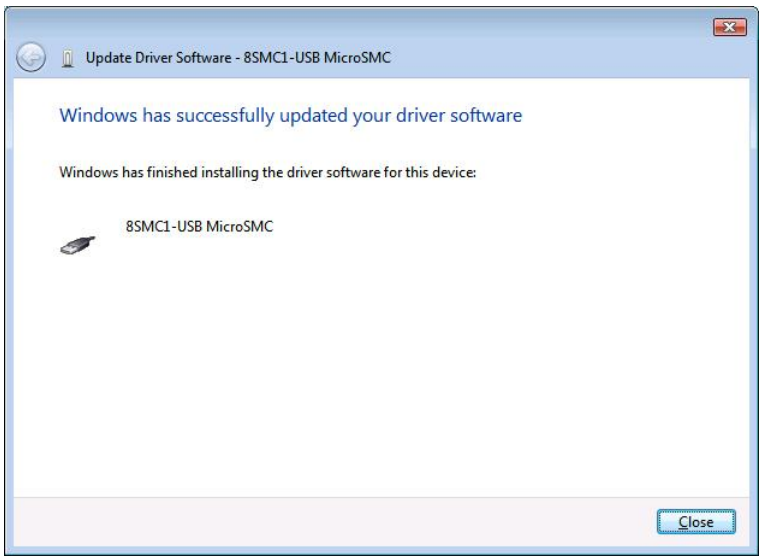


Figure 76. Windows has successfully updated driver.

Note: SMCVieW application and all LabVieW based applications can work only with NI VISA driver.

For switching from MicroSMC to NI VISA driver please does exactly the same as it is shown above. Moreover, mark “Delete the driver software for this device” checkbox when you delete MicroSMC driver (see Figure 71). Choose the folder with VISA driver, ”C:\Program Files\SMCVieW\Drivers” usually, on Figure 75.

Note that in Device Manager list MicroSMC driver is placed in “Universal Serial Bus controllers” group (see Figure 77).

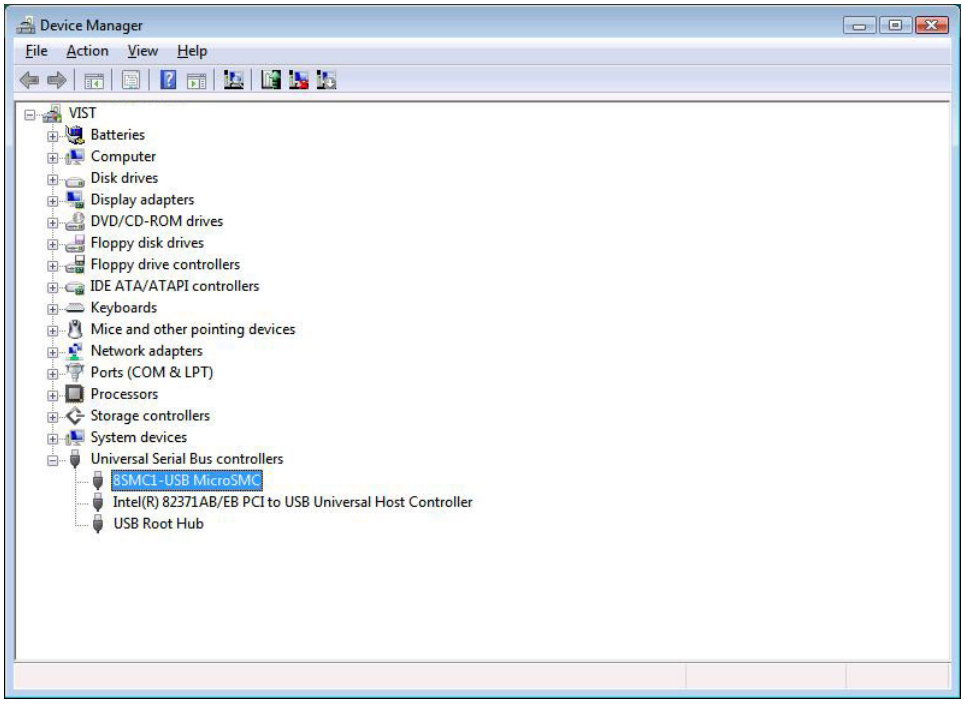


Figure 77. Position of MicroSMC driver in Device Manager list.

4.7 Known USB driver installation problems

Although MicroSMC driver supports up to 32 8SMC1-USBh devices and NI VISA driver supports up to 30 8SMC1-USBh devices simultaneously, WindowsXP Hardware wizard in "Install the software automatically (Recommended)" mode (see Figure 61) may works improper if more than 6 8SMC1-USBh devices are connected. In case of that improper work Hardware wizard has been looking for USB driver several minutes and find nothing, USB driver will not be installed and 8SMC1-USBh controller will mark with "?" in Device Manager device list. If it is happened make a right click on this 8SMC1-USBh controller and choose "Update driver…".

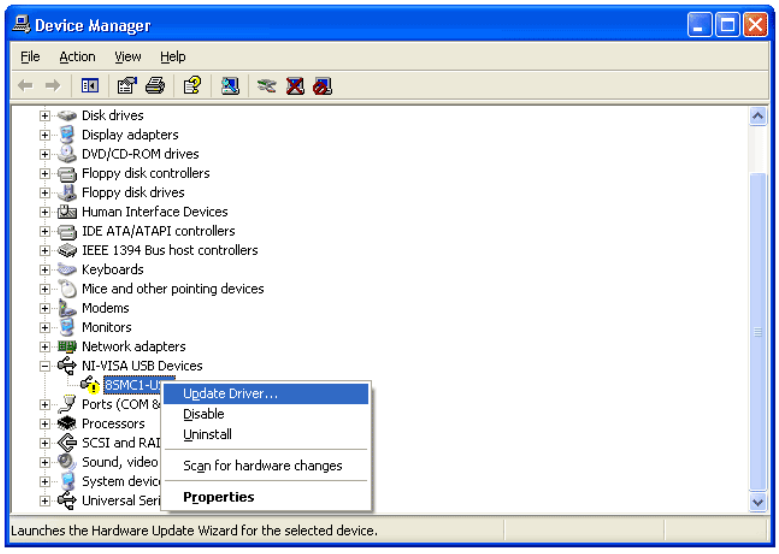


Figure 78. Driver updating

At screen like Figure 61 choose "Install from a list or specific location (Advanced)". At the next screen choose "Don't search. I will choose the driver to install" (see Figure 79). Then choose appropriate driver from list (see Figure 69).

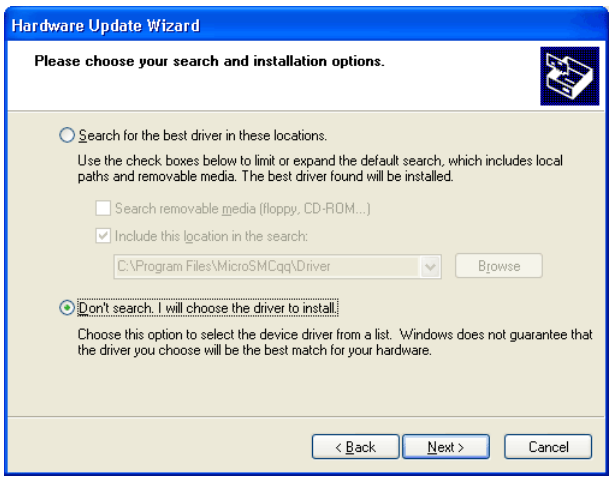


Figure 79. Manual USB driver installation

Use this way always when you are installing a lot of 8SMC1-USBh devices: choose at screen like Figure 61 "Install from a list or specific location (Advanced)" and "Don't search. I will choose the driver to install" then.

5 Application SMCVieW

5.1 General information

SMCVieW is a friendly graphical user interface for control, monitoring and tuning your stepping motors. It can also be used for easy setup and save/load of all parameters for each stepping motor. Interface supports up to 30 drivers simultaneously.

When windows is started, press "Start", choose "Programs" group, then "SMCVieW" subgroup and "SMCVieW" application.

5.2 Main screen

5.2.1 General view

When we start "SMCVieW" screen like Figure 80 will appear.

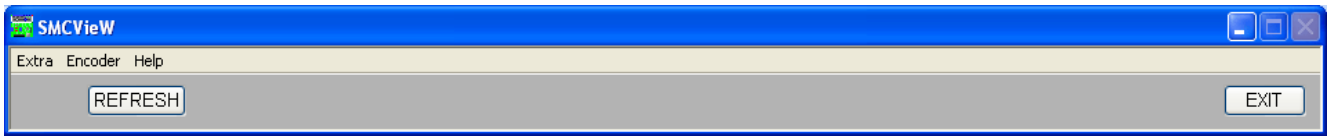


Figure 80. SMCVieW main screen. No 8SMC1-USBh drives found

Press "REFRESH" button. All available 8SMC1-USBh devices will be found. Screen with one device is shown on Figure 81. Screen with four devices is shown on Figure 82. Interface can represent up to three devices simultaneously. If more than three 8SMC1-USBh devices are used at the same time, slider is displayed at the left side of SMCVieW interface (see Figure 82).

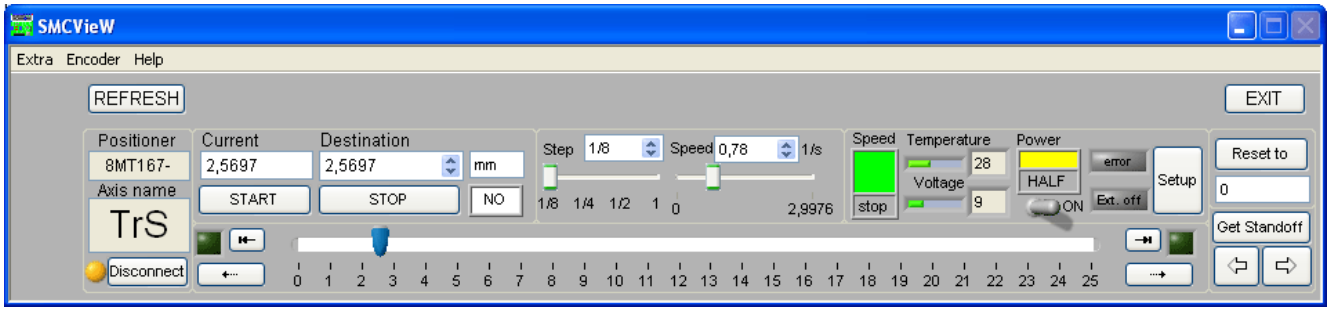


Figure 81. SMCVieW main screen. One 8SMC1-USBh drive found

Let's examine the main screen more accurately.

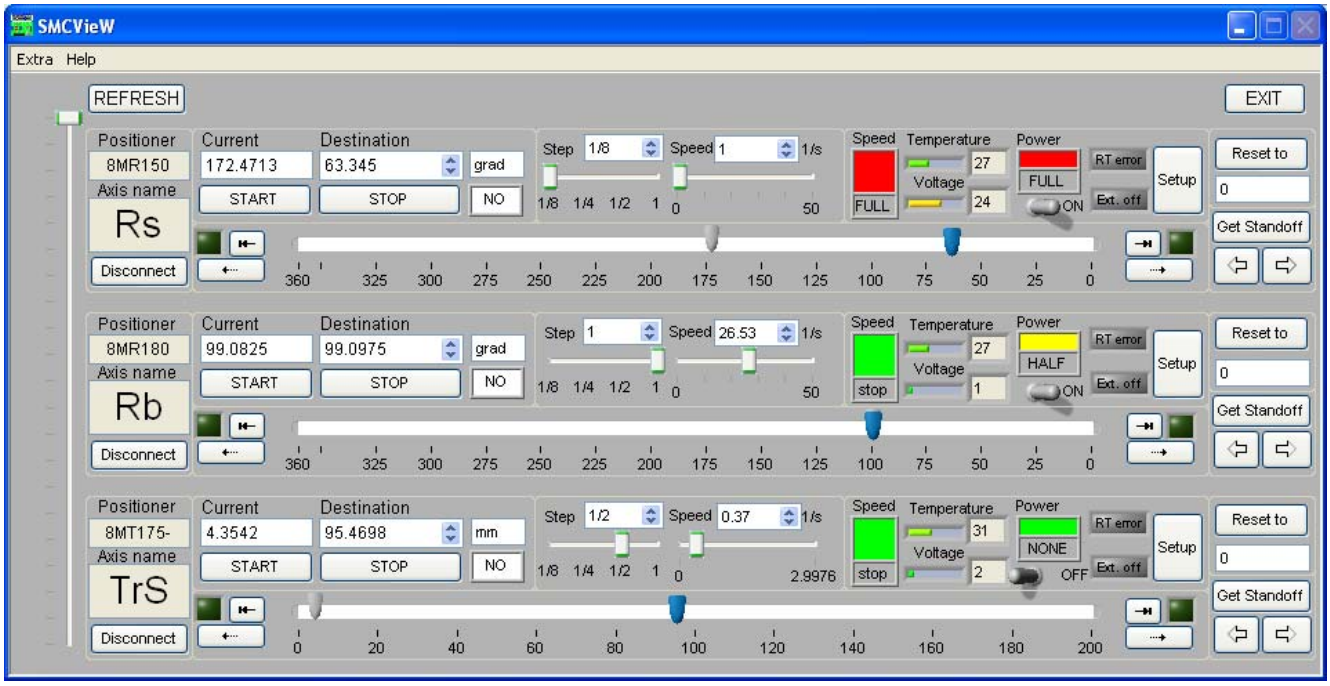


Figure 82. SMCVieW main screen. Four 8SMC1-USBh drives found

5.2.2 Positioners and axes names

Figure 83.

On left top of the main window positioner model is shown (see 83). It can be changed by Setup (see 5.4.2).

Under the positioner model box the lettering of axis is shown (see 83). It can be changed by Setup (see 5.4.5).

Button **Disconnect** is intended for disconnecting axis.

Round LED in the left bottom corner of 83 indicates the USB connection of axis to PC. It is orange if controller is attached and gray otherwise.

5.2.3 Current and Destination positions

Figure 84.

Indicator **Steps** represents current units of measurement (see 84.) It can be changed by Setup. Control **Destination** is used to set desired stop position (in mentioned units). Indicator **Current** – shows the current position of positioner in mentioned units.

Button **Start** – starts motion.

Button **Stop** – stops motion.

Right bottom indicator (see 84.) shows the backlash state.

Note: If synchronization input enabled, motion will start after 1st synchronization pulse input **only**.

5.2.4 Speed and precision

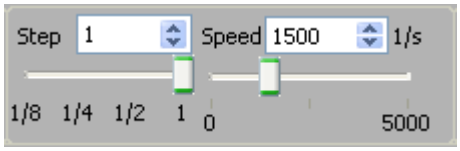


Figure 85.

Control **Step** is used for positioning accuracy setup. Note that maximal speed depends on accuracy. Soft start/stop is possible only in full step mode.

Control **Speed** is used for velocity setup. Speed is indicated in mentioned units per second.

5.2.5 Status and power

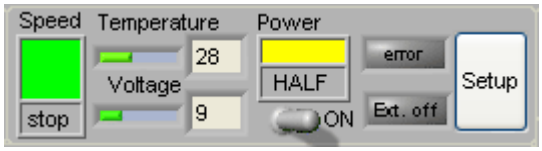


Figure 86.

Indicator **Speed** shows the stepping motor current speed status. Possible values:

stop - stepping motor is stopped

accel - stepping motor is accelerating or decelerating now

FULL - stepping motor is moving on specified speed

Indicator **Temperature** shows the temperature of Power driver of 8SMC1-USBh controller in Celsius.

Indicator **Voltage** shows the voltage on Power driver of 8SMC1-USBh controller.

Indicator **Power** shows the stepping motor power. Possible values:

None - stepping motor is not powered,

HALF - stepping motor in hold mode, 40% current reduction,

FULL –stepping motor is fed by full power.

Power toggle switch is used for switching stepping motor power On/Off. Note that it may be turned off internally by controller in over temperature or emergency stop cases.

Indicator **error** is lighting up when Revolution sensor or Quadrature encoder detects error (see 2.2.6). Error will be reset when **Start** button is pushed (see 5.2.3) or by **Reset error status** button (see 5.4.11).

Indicator **Ext. off** is lighting up when contact of Emergency stop switch is opened. Stepper motor de-energizes immediately and **Power** toggle switch turn to off position (for more details see 2.2.5).

Setup – opens Setup window

5.2.6 Position slider

Position slider graphically represents current and destination positions (see Figure 87), allows to choose the limits of displacement, moves to these limits (buttons ⇠ and ⇢ correspondingly). LEDs on the left and right corners represent the Limit switches states. Buttons ⇤ and ⇥ are used for fast setup of low and high limits of displacement. If button ⇤ is pressed stepping motor moves to smaller position (in tics – controller's internal absolute

units of measurement)). When it is reached, motor stops and current position is set as low limit. If button ⇥ is pressed stepping motor moves to larger position (in tics). When it is reached, motor stops and current position is set as high limit. One can use **Position slider** as well as digital display for setting destination position.

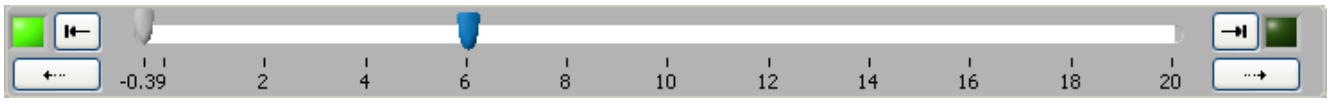


Figure 87. Position slider and Limit switches

5.2.7 Reset and standoff

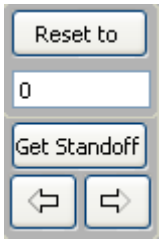


Figure 88.

Easy "Reset position" operation is available at SMCVieW (starting from version 1.14). It's very helpful if you have accidentally erased *.sav file or changed the position of the stepping motor without SMCVieW. Group of buttons, responsible for that action is shown on the 88.

The main idea of this feature is following: first you move step motor to the position that will be treated as reference point. (In the future, reset operation will move step motor to this very point). It is very convenient to use ⇦, ⇨ buttons to specify reference point precisely (Note: Limit switches are OFF while using buttons!). These buttons emulate hardware controller buttons. You should also enter the value corresponding to the reference point at the field below **Reset to** button. (You can change this value at Setup -> Standoff & Reset -> Reset to (see 5.4.9)) Then you use **Get Standoff** button. Step motor will rotate to the limit switch (The direction of rotation and hence limit switch is chosen at Setup -> Standoff & Reset -> Reset towards (see 5.4.9)), memorize the difference between reference point and Limit switch, return to the reference point position. The value memorized can be altered at Setup -> Standoff & Reset -> Standoff. Last thing you should do before performing reset operation is setting of your working range. You can do this at Setup -> Standoff & Reset -> Minimal Value, Maximal Value. Everything is ready now for the conducting "Reset position" operation. Push **Reset to** button. (You can use that button any time you wish) Step motor will move to the limit switch and then move away from it by the value obtained from **Get Standoff** action. In other words step motor will move to the reference point. Working range and current position, shown on the interface will be changed to the one specified earlier. (Note! It is recommended to perform reset operation in the 1/8 step mode, because this mode grants best accuracy) Remember, that if you are using *rotational* positioner limit switch(s) will be OFF after reset operation, and if you are using *linear* positioner - Limit switch(s) will be ON.

5.3 Main menu

SMCVieW main menu consists of three submenus: Extra, Encoder and Help. Menu Extra contains additional applications that can be used if you have special license. Contact you dealer for additional information. Menu Encoder contains special widows that show current shaft position (see Figure 89).

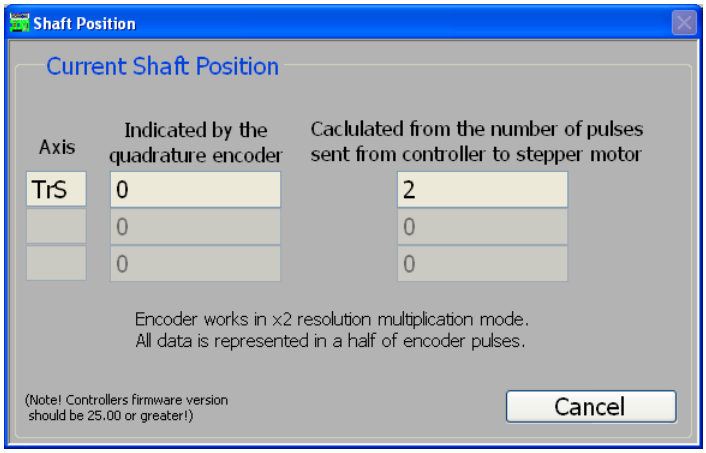


Figure 89. Shaft position window

Shaft position window can show position of three axes. It represents current position of the stepper motor shaft in a half of encoder pulses. Left windows on Figure 89 show the value calculated by the quadrature encoder. Right windows shows the value calculated from the number of pulses sent from controller to stepper motor. Shaft position window represent three axes that are shown on SMCVieW main screen.

5.4 Setup

5.4.1 Buttons

Buttons are shown in Figure 90.

Save params – saves current installation-specific settings on disk.

Load params – loads current installation-specific settings from disk.

Save to flash – saves current installation-specific settings on 8SMC1-USBh flash memory.

Default – restore default installation-specific settings.

Undo – restore previous installation-specific settings.

Apply – apply current changes.

OK – apply current changes and exit from **Setup**.

Exit – exit from **Setup** without applying current changes.

Figure 90. Buttons

5.4.2 Positioner

This window is used for load a set of installation-specific settings for the selected positioner. Choose the Manufacturer, Model name and appropriate wiring diagram. Press **Load Button** when selection is made. All fields of "Setup" will be changed to the values specific for this positioner.

R - Rated resistance for current sense resistors. Use this field for information purposes only. Make sure that real value of current sense resistors is equal to this (see 4.2.1 for more information).

I - Rated current of step motor. Use this field for information purposes only!

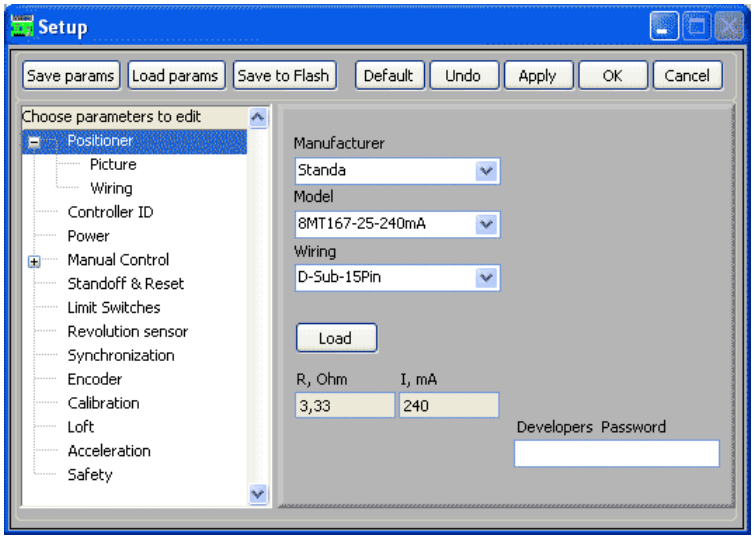


Figure 91. Positioner setup screen

5.4.3 Positioner -> Picture

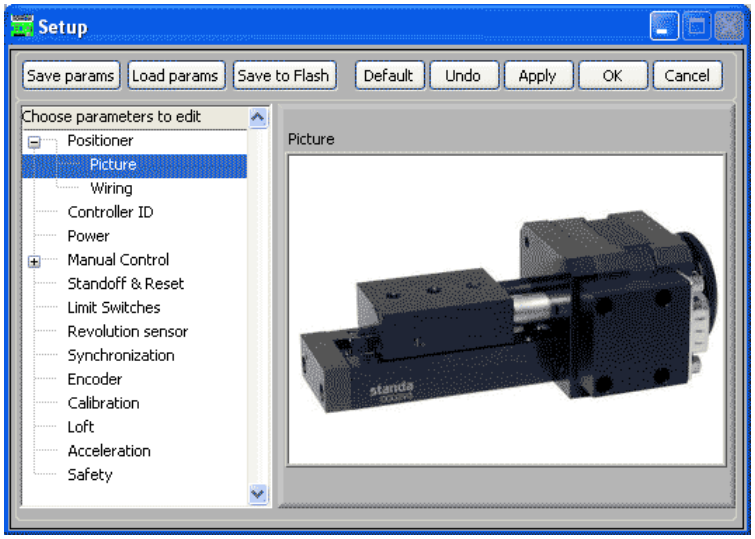


Figure 92. Setup screen "Positioner -> Picture"

This window shows the view of selected positioner.

5.4.4 Positioner -> Wiring

This window shows the selected wiring diagram. Use this field for information purposes only. Make sure that real wiring diagram is equal to this (see 3.5 for more information).

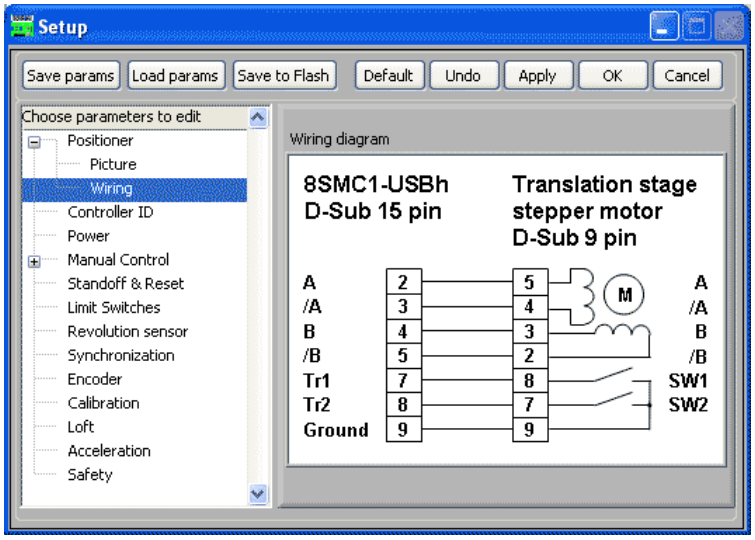


Figure 93. Setup screen "Positioner -> Wiring"

5.4.5 Controller identifier

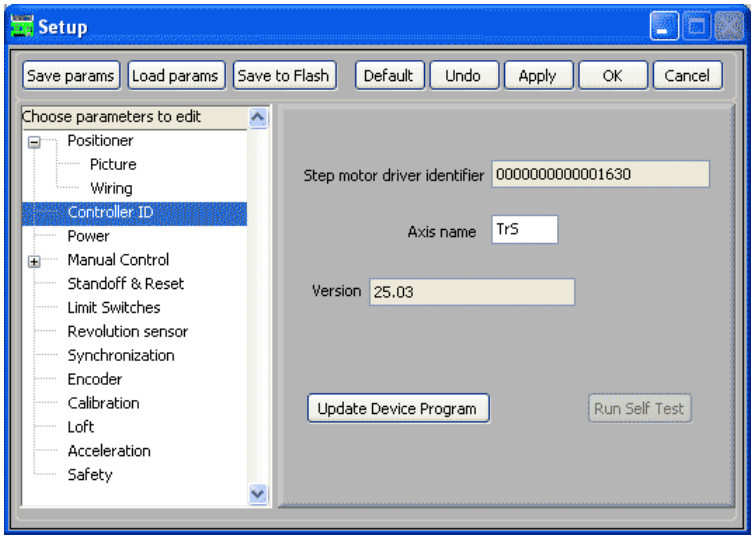


Figure 94. Setup screen "Controller identifier"

Step motor driver identifier - the unique numerical identifier of this 8SMC1-USBh driver.

Axis name - Lettering of this axis.

Version - Version of controller firmware.

Update Device Program - Updating the firmware of 8SMC-USBh controller.

Note: Download the latest version of SMCVieW application before firmware updating because the latest revision of firmware can contain some new features that is not supported by older version of SMCVieW application.

5.4.6 Power management

Current reduction in hold mode – Check box that allow to use 40% current reduction and 60% heating reduction in hold mode.

Current reduction delay, ms – delay from stepper motor stop to current reduction (1..9961 ms).

Power off when stop – This check box allows power off stepping motor when it stops.

Power off delay, s – delay from stepper motor stop to power off (1..1000 s).

Power off mode – Two choices are possible:

Power off and make a whole step – shifts stepping motor in full step position then power off. It makes possible not to miss the information about precise position that is important for microstep modes.

Just power off – power off in current position without any displacement.

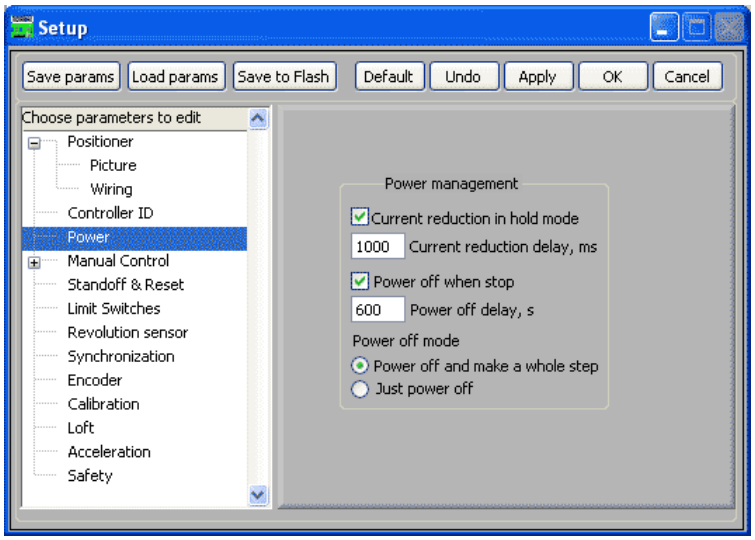


Figure 95. Power management setup screen

5.4.7 Manual control

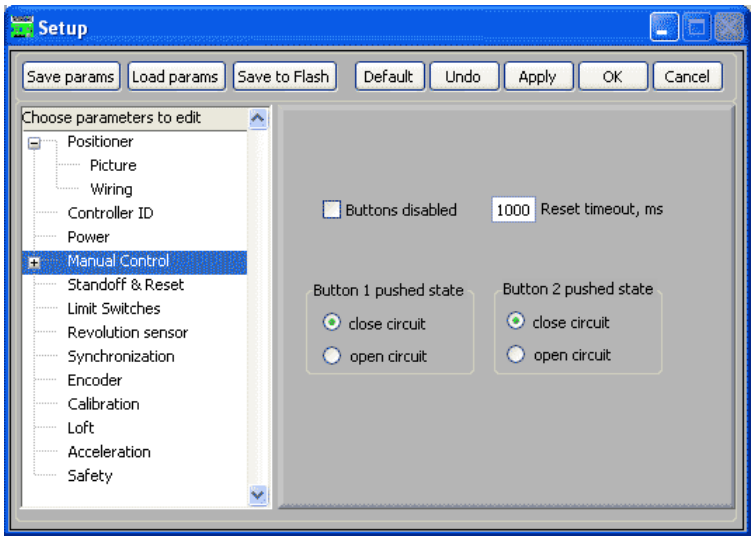


Figure 96. Manual control setup screen

Buttons disabled – This check box disables or enables buttons for local control.
Reset timeout, ms - Pressing of two buttons and holding during this timeout leads to reset driver and shifting to null position (in ticks). Possible value 1..9961 ms.

5.4.8 Manual control -> Acceleration Curve

This tab represents the acceleration in manual control mode (control with local knobs).

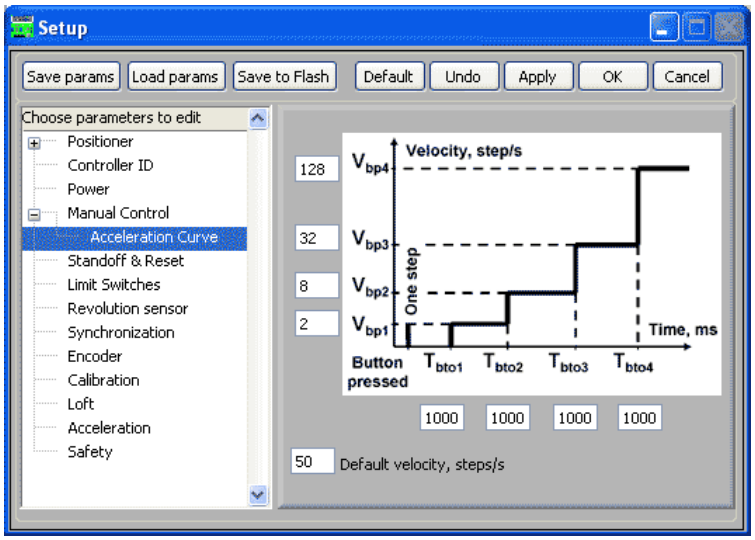


Figure 97. Manual control -> Acceleration Curve

V_{bp1} - Lowest speed of stepping motor in manual control mode (2..625 full steps/s).
V_{bp2} - Velocity of stepping motor in manual control mode after first speed up (2..625 full steps/s).
V_{bp3} - Velocity of stepping motor in manual control mode after second speed up (2..625 full steps/s).
V_{bp4} – Velocity of stepping motor in manual control mode after third speed up (2..625 full steps/s).
Default velocity, steps/s - Velocity of stepping motor in reset and homing mode (2..625 full steps/s).

T_{bto1} – Delay from pressure of button to start of continuous motion in manual control mode (1..9961 ms).
T_{bto2} – Delay from start of continuous motion to first speed up in manual control mode (1..9961 ms).
T_{bto3} – Delay from first to second speed up in manual control mode (1..9961 ms).
T_{bto4} – Delay from second to third speed up in manual control mode (1..9961 ms).

5.4.9 Reset and Standoff

This window is used for setting some specific parameters for "Reset" and "Standoff" procedure (for more information see 5.2.7).

Positioner's Type – type of positioner. Possible values: Rotational or Linear. Major difference: linear positioner has two limit switches and its working range locates between it, rotational stage has only one switch that correspond to zero position.

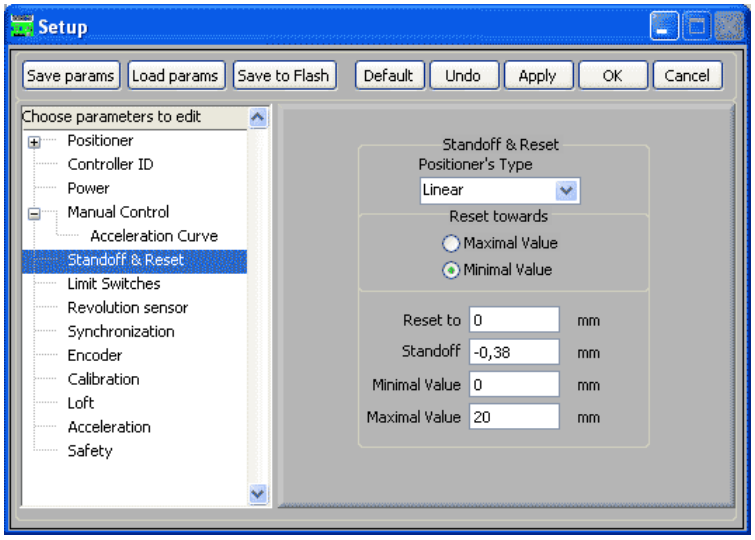


Figure 98. Reset and Standoff setup screen

Reset towards – direction of moving towards the switch.

Reset to - Reset position (in units). After "Reset" procedure positioner will be in this point.

Standoff –The difference between reference point and limit switch.

Minimal Value – Minimal value that shows Position slider (see 5.2.6) after "Reset" procedure.

Maximal Value - Maximal value that shows Position slider (see 5.2.6) after "Reset" procedure.

5.4.10 Limit switches

See Figure 99. It is possible to disable limit switches and choose pushed position. Also it is possible to swap limit switches.

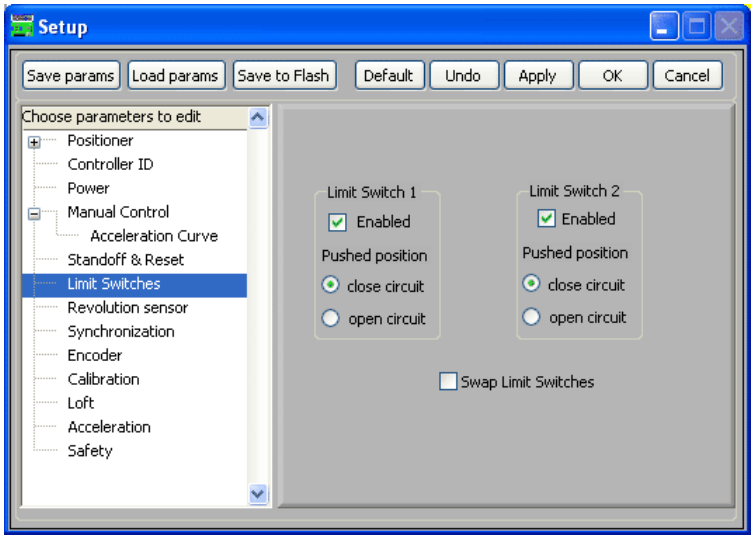


Figure 99. Limit switches setup screen

5.4.11 Revolution sensor

Revolution sensor is intended for sense stepper motor slipping and inform user if slipping is detected.

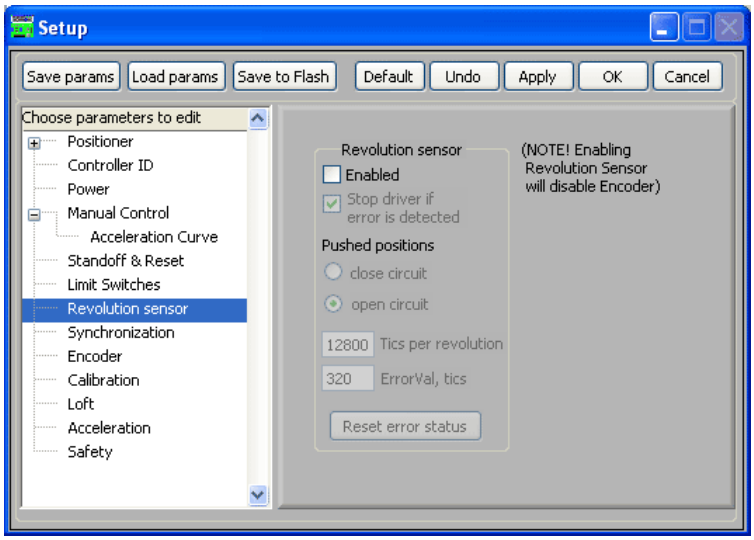


Figure 100. Revolution sensor setup screen

Enabled – This check box enables or disables revolution sensor.
Stop driver if error is detected - Stops stepping motor if given displacement distinguished from rotary transducer value more then **ErrorVal** tics.
Tics per revolution - The number of tics per revolution of this stepping motor (256..51200 tics). This value must be divisible by 256! One full step is equal to 64 tics.
ErrorVal, tics - Maximal value that can be accumulated by revolution sensor without appearance of revolution sensor error. 64..51200 tics, but no more then one full turn.
Reset error status - Resets the error value that is accumulated by revolution sensor. Commonly it is used for clearing revolution sensor error flag.
Pushed position – determine the logic level when the marker on revolution sensor disk locates between optocouple's emitter and detector.

Note: Revolution sensor detects slippering in forward and backward directions independently.

5.4.12 Encoder

SMCVieW uses quadrature encoder (for more details see 2.2.7) for two reasons: representation of the real position of stepper motor shaft (see paragraph 5.3) and slippage detection. Encoder setup screen is shown on Figure 101.

For slippage detection controller compare positions calculated by encoder and by the number of pulses sent from controller to stepper motor (shown on Figure 89). If this difference is more than **ErrorVal** (see paragraph 5.4.11) then indicator **Error** on main screen (see Figure 81) lights up. If check box "Stop on Encoder Error" is marked stepper motor will be stopped after slippage detection.

For discarding Error flag use **Reset Encoder Error** button (see Figure 101). Note that this action discards Error flag only. If the reason of error still present (difference between real and calculated current position more than **ErrorVal) Error** will be lighted up just after the next start.

Radio button on Figure 101 determines encoder state: Disabled, Normal Polarity and Reversed Polarity. Normal and reversed polarity define the direction of encoder incrementing

and decrementing. Change it if current position indicated by encoder decrements when current position calculated by the number of pulses sent from controller to stepper motor increments.

Control **Steps per Revolution** must be filled by number of steps per revolution for stepper motor that is used. Control **Encoder pulses per revolution** must be filled by CPR (see paragraph 2.2.7) of quadrature encoder that is used. These fields are used SMCVieW for **EncVSCP** calculation that is necessary for correct work of slippage detection.

Set to Encoder Position button changes Current position (calculated by the number of pulses sent from controller to stepper motor) to the nearest value to the position calculated by encoder. This button is usually used for elimination of Error flag reason before **Reset Encoder Error** button.

Reset to 0 button changes Current positions (calculated by the number of pulses sent from controller to stepper motor and) calculated by encoder) to 0. This button is used for setting suitable Current position before starting work with encoder.

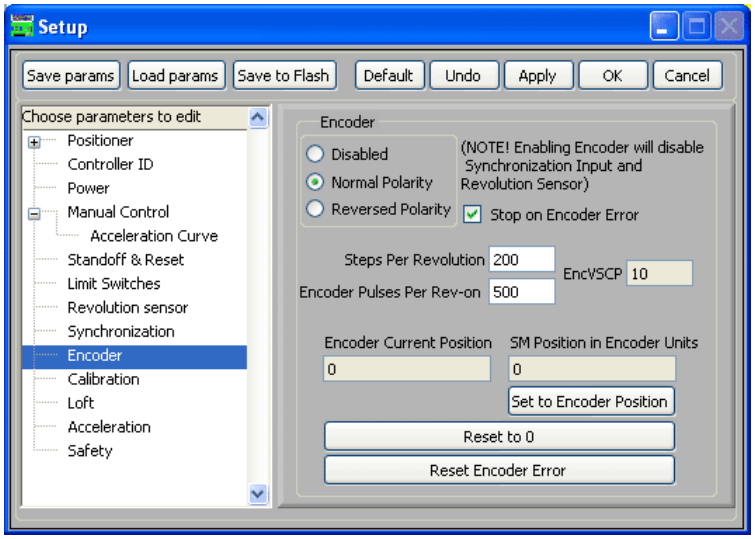


Figure 101. Encoder setup screen

5.4.13 Synchronization

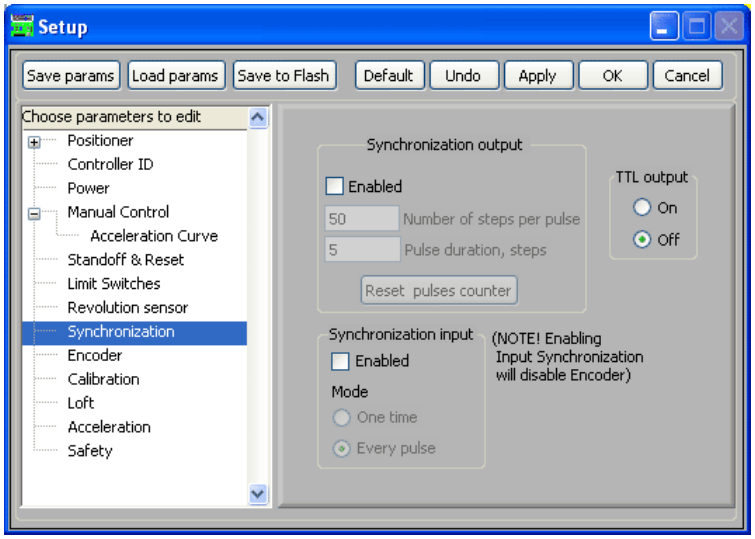


Figure 102. Synchronization setup screen

Synchronization output – this box unites controls connected with synchronization output mode.

Number of steps per pulse - number of steps (or microsteps, depending on resolution) between two output synchronization pulses. Possible value 1..2.147.483.647.

Pulse duration, steps - pulse duration in steps (or microsteps, depending on resolution) duration units. Pulse period equals ([value] – ½)*[step period]. Minimal available pulse duration is about 100 microseconds.

Reset pulses counter - Press this button to null the output synchronization pulses counter.

TTL output - synchronization output null state.

On – means that logic "1" is a state of synchronization output when output synchronization impulse is absent and "0" when output synchronization impulse is present.

Off – means that logic "0" is a state of synchronization output when output synchronization impulse is absent and "1" when output synchronization impulse is present.

Note: It is possible to use this switch for direct control of synchronization output.

Synchronization input – this box unites controls connected with synchronization input mode.

Enabled - This check box enables or disables input synchronization. If input synchronization is enabled stepping motor does not make a move until the synchronization pulse is received. Note: press **Start** button in main window after this mode has been activated to start waiting input synchronization pulse.

Mode - synchronization input mode.

One time - When synchronization pulse is received stepping motor shifts to destination position that is indicated in the main window of the interface.

Every pulse - With every synchronization pulse received stepping motor shifts by difference between given current and destination positions.

5.4.14 Calibration

Units name - selected units name.

1st gage point in units - position of 1st gage point in selected units. Note: Units must be greater than ticks.

2nd gage point in units - position of 2nd gage point in selected units. Note: Units must be greater than ticks.

1st gage point in tics - Position of 1st gage point in tics. Possible value: -2147483647..2147483647 tics. Note: this value must be divisible by 8. One full step is equal to 64 tics.

2nd gage point in tics - Position of 2nd gage point in tics. Possible value: -2147483647..2147483647 tics. Note: this value must be divisible by 8. One full step is equal to 64 tics.

Current position in tics - setting new current position in tics. Possible value: -2147483647..2147483647 tics. Note: this value must be divisible by 8. This control usually is used for normalize real current position in tics after some malfunctions. Do not use button **Set** for calibrating.

Current position in units - setting new current position in units. This control is usually used for calibrating if you want to assume current position defined value in units. Use button **Set** for calibrating and recalibrating.

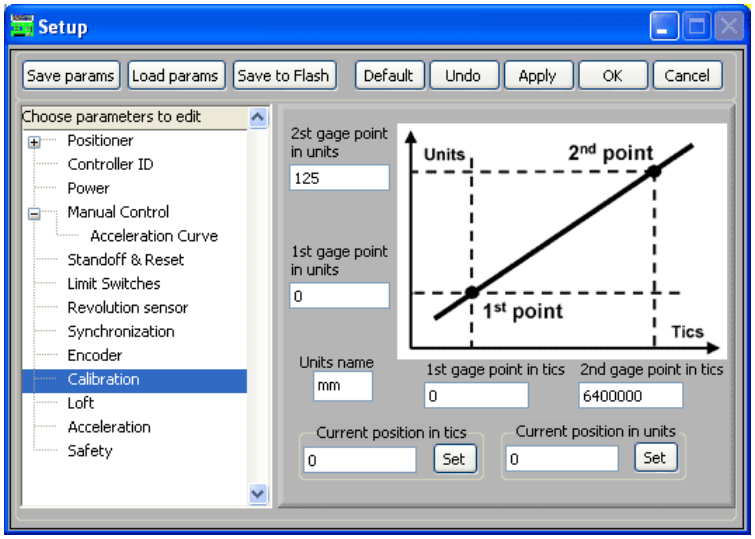


Figure 103. Calibration setup screen

5.4.15 Backlash compensation

Backlash compensation is compensation of the mechanical slippage of the positioner. That operation is carried out when stepper motor reaches destination position. It can be performed from "left" or "right" side.

Status

Off – disable this function

To Limit switch 1 - reach the destination position from the Limit switch 2 side

To Limit switch 2 - reach the destination position from the Limit switch 1 side

Number of tics for backlash compensation - number of tics for backlash compensation, possible value: 64..65472 tics. Note: this value must be divisible by 64. One full step is equal to 64 tics.

Loft last stage speed - speed of backlash compensation, possible value: 1..625 full steps/s.

Carry out backlash Compensation – press this button to carry out backlash Compensation.

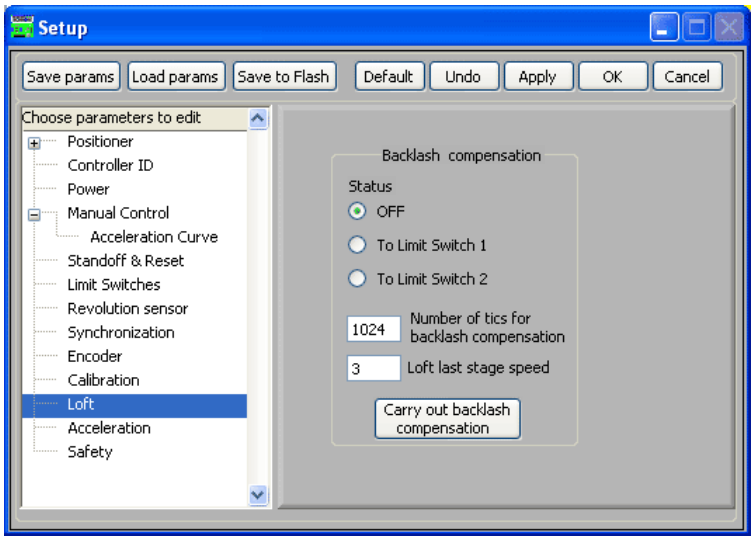


Figure 104. Backlash compensation setup screen

5.4.16 Acceleration

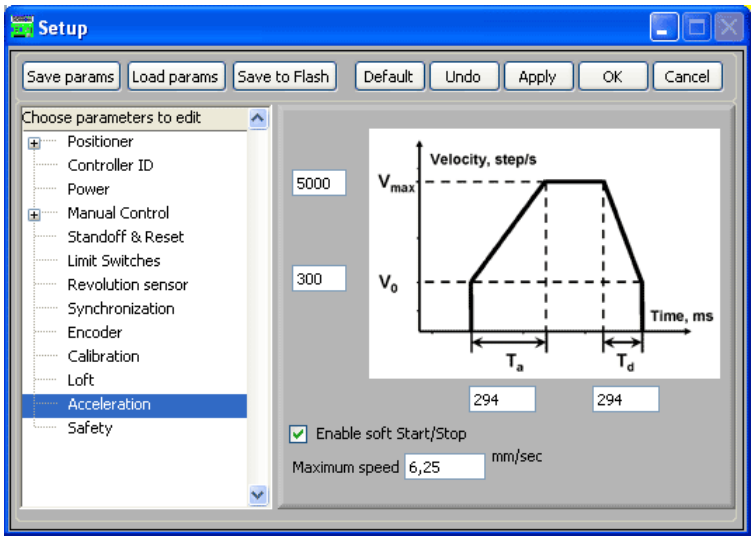


Figure 105. Acceleration setup screen

This tab sets Acceleration parameters for stepper motor.

5000 - maximum velocity of stepping motor. This value is fixed at 5000 full steps/s.

300 - initial velocity of stepping motor in soft start mode. This value is fixed at 300 full steps/s.

Ta - acceleration time (98..1470 ms).

Td - deceleration time (98..1470 ms).

Enable soft Start/Stop – enable soft Start/Stop mode. Note that soft Start/Stop is possible only in full step mode; in microstep modes this flag will be ignored.

Maximum speed, units/sec - high speed limit (in units/s). Please check this value after recalibration.

5.4.17 Safety

On this tab you can set the **Thermal power off** threshold. Upon reaching this temperature controller will turn off the power.

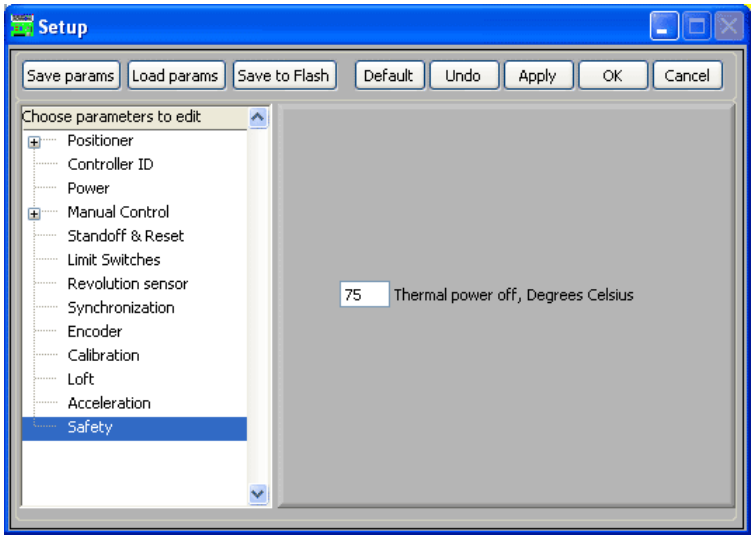


Figure 106. Thermal power off setup screen

6 VI's library

6.1 General information

Stepper motor controller comes with a suite of advanced software tools for control and tuning your stepping motors, third-party software development and upgrading firmware using National Instruments LabVieW 7.1.1 and higher. Open block diagram of the uSMC VI's tree (see file Development Kit\VIs\Tree (uSMC).vi on CD) to see all of them (Figure 107).

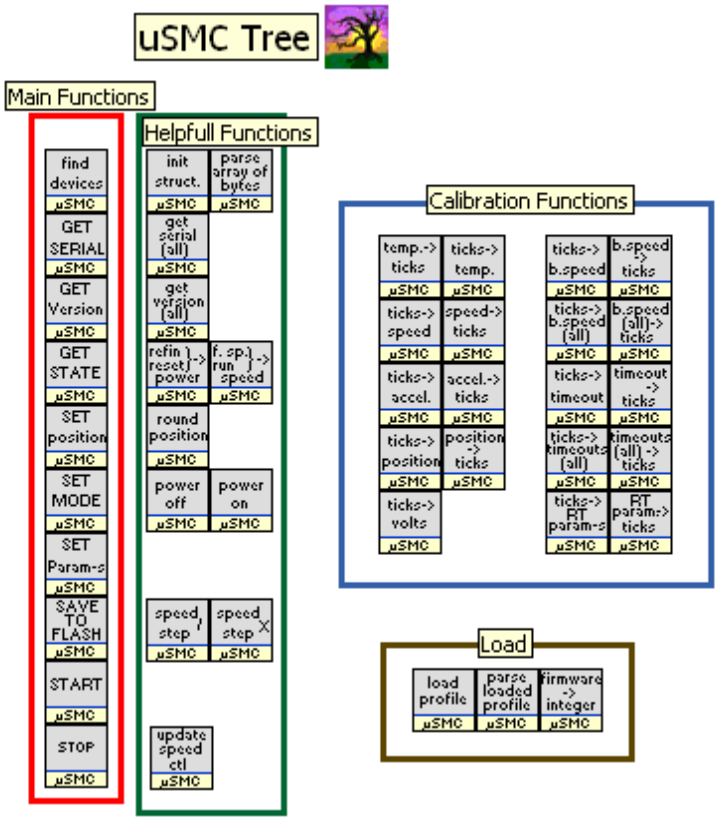


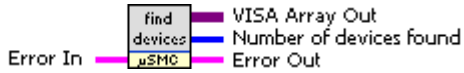
Figure 107. All VIs supplied with controller

Note: All LabVIEW based software require NI VISA driver (see paragraph **Error! Reference source not found.**). Make sure that you use NI VISA driver for 8SMC1-USBh. Change driver if it necessary (see paragraph 4.5).

Note: Makes sure that you use 8SMC1-USBh controllers with firmware usmc2503.usm or higher! In other case update it before programming (see paragraph 5.4.5).

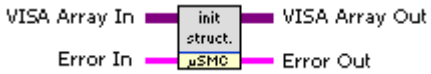
Note: Makes sure that you use NI LabVIEW 7.1.1 or higher.

6.1.1 Find Devices (uSMC)



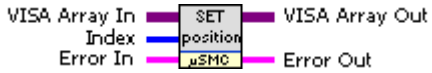
Finds all devices connected to PC and returns their serial numbers.
Error In, Error Out - standard LabVieW error handling clusters.
Visa Array Out - array containing handles to the resources in the National Instruments VISA format.
Number of devices found – number of the found controllers

6.1.2 Initialize All Structures



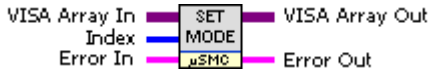
Creates all structures (see section 6.2 on page 64) with the default parameters for each controller found by "Find Devices (uSMC).vi" and stores it in the Global variables ("Globals (uSMC).vi")
Visa Array In, Visa Array Out - arrays containing handles to the resources (controllers) in the National Instruments VISA format.

6.1.3 Set Current Position (uSMC)



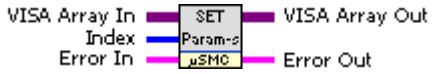
Sets position stored in the "Set Cur Pos Units" of the specified Position structure (see section 6.2.2) as the current position.
Index – Number of the controller.

6.1.4 Set Mode (uSMC)



Sets the mode of the specified controller. Mode is taken from the global variables.

6.1.5 Set Parameters (uSMC)



Sets parameters for the specified controller. Parameters are taken from the global variables.

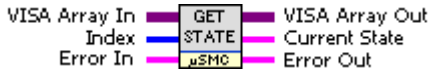
6.1.6 Save Parameters to Flash (uSMC)

Saves current Parameters and Mode in the specified controller flash memory

6.1.7 Get Serial (uSMC)

Gets serial number of the specified controller
Serial Number – unique identifier of the controller

6.1.8 Get State (uSMC)

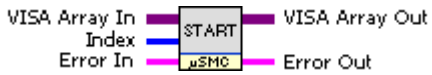


Gets information about the current state of the specified controller
Current State – cluster described at the section 6.2.3 on page 67.

6.1.9 Get Version (uSMC)

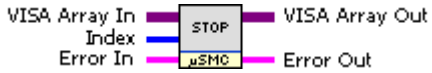
Gets version of the specified controllers software
Firmware Version – Version of the microcontroller's program

6.1.10 Start (uSMC)



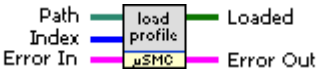
Starts the movement of the specified step motor. Start parameters are taken from global variables.

6.1.11 Stop (uSMC)



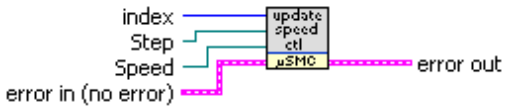
Stops the rotation of the specified step motor

6.1.12 Load Profile (uSMC)



Loads the profile – complete set of controllers parameters created by SMCVieW application and stores it in the global variables.
Path – path to the file
Loaded – TRUE if parameters have be successfully loaded and FALSE otherwise.

6.1.13 Update Speed Ctl (uSMC)



Useful VI that helps to update maximal and minimal values of the Speed control as well as to limit current speed according to current selection of step.
Step – reference to the Step control (0 – 1/8 step, 1- ¼ Step Mode, 2- ½ Step Mode, 3 – Full Step)
Speed – reference to the Speed control. (Note! Type of this controller should be "Slider".)

6.2 Structures

6.2.1 Mode

Title	Default	Description
Turn Buttons Off	FALSE	If TRUE - buttons are disabled
Reduce Current	TRUE	If TRUE - current will be reduced to 60% from the maximum after the time specified by Reduce Current Timeout parameter (see 0)
Power Off and Make a Whole Step	TRUE	If TRUE step motor will be moved to the whole step position and power will be off
Emergency Power Off	FALSE	Power will be off without moving to the whole step position. As the result accurate position will be lost.
Limit switch 1 True	FALSE	Limit switch 1 true logical state

State		
Limit switch 2 True State	FALSE	Limit switch 2 true logical state
Rotary Transducer True State	TRUE	Rotary transducer true logical state
Swap Limit switches	TRUE	If TRUE Limit switches are swapped
Limit switch 1 Operation Enabled	TRUE	If TRUE step motor will be stopped when Limit switch 1 in a true logical state
Limit switch 2 Operation Enabled	TRUE	If TRUE step motor will be stopped when Limit switch 2 in a true logical state
Rotary Transducer Operation Enabled	FALSE	If TRUE rotary transducer will be used (note: one full revolution is needed before error can be detected).
Rotary Transducer Operation Select	TRUE	If TRUE step motor will cease moving if rotary transducer is in true logical state
Button 1 True State	FALSE	Button 1 true logical state
Button 2 True State	FALSE	Button 2 true logical state
Swap Buttons		If TRUE buttons will be swapped
Reset Rotary Transducer	FALSE	Reset rotary transducer logical machine
Enable Output Synchronization	FALSE	If TRUE output synchronization enabled
Reset Output Sync Counter	FALSE	Reset output synchronization counter
Sync In operation	FALSE	If TRUE step motor will move one time (after receiving of the synchronization signal) to the destination position specified by Start command (see 5.2.9), if FALSE step motor will move multiple times by difference between current position and the destination position.
Sync count	50	[u32] Number of steps after which synchronization output signal occurs. If Sync count=0 then synchronization output signal occurs when the motion start.
Set Power State	TRUE	Sets power of the controller on (TRUE) or off (FALSE).
Invert Sync Out Polarity	FALSE	Inverts the polarity of the signals from the Synchronization Out pin.
Use Encoder	FALSE	If TRUE encoder will be used to determine the precise position of the step motor. (Note! If you set this bit to TRUE you should set Rotary Transducer Operation Enabled to FALSE)
Invert Encoder Direction	FALSE	Invert Encoder Counter Direction
Reset Encoder to 0	FALSE	Sets current encoder position to zero.
Reset SM Enc Pos to Enc Pos	FALSE	Sets current step motor position (in encoder units) to encoder position.

6.2.2 Parameters

Title	Default	Description
Acceleration Time	294	[u16] Acceleration time (in ms). Possible range: 1 - 24990.

Deceleration Time	294	[u16] Deceleration time (in ms). Possible range: 1 - 24990.
Reduce Current Timeout	1000	[u16] – Time (in ms) after which current will be reduced to 60% of maximum
Buttons Timeout 1	500	[u16] – Time (in ms) after which speed of step motor rotation will be equal to the one specified at Buttons Speed 1 field (see below). (This parameter is used when controlling step motor with buttons)
Buttons Timeout 2	1000	[u16] – Time (in ms) after which speed of step motor rotation will be equal to the one specified at Buttons Speed 2 field. (This parameter is used when controlling step motor with buttons)
Buttons Timeout 3	1500	[u16] – Time (in ms) after which speed of step motor rotation will be equal to the one specified at Buttons Speed 3 field. (This parameter is used when controlling step motor using buttons)
Buttons Timeout 4	2000	[u16] – Time (in ms) after which speed of step motor rotation will be equal to the one specified at Buttons Speed 4 field. (This parameter is used when controlling step motor using buttons)
Buttons Timeout Reset	2000	[u16] – Time (in ms) after which reset command will be performed
Buttons Timeout DblClick	294	[u16] – Time (in ms) after which double click command will be performed
Reset Speed	64	[u16] – Speed (Full Steps/sec) while performing reset operation. (This parameter is used when controlling step motor with buttons)
Buttons Speed 1	3	[u16] – Speed (Full Steps/sec) after Buttons Timeout 1 time has passed. (This parameter is used when controlling step motor with buttons)
Buttons Speed 2	13	[u16] – Speed (Full Steps/sec) after Buttons Timeout 2 time has passed. (This parameter is used when controlling step motor with buttons)
Buttons Speed 3	32	[u16] – Speed (Full Steps/sec) after Buttons Timeout 3 time has passed. (This parameter is used when controlling step motor with buttons)
Buttons Speed 4	64	[u16] – Speed (Full Steps/sec) after Buttons Timeout 4 time has passed. (This parameter is used when controlling step motor with buttons)
Max loft	1024	[u16] Value in ticks that will be used performing backlash operation
RTDelta	12800	[u16] - Revolution distance – number of ticks per one full revolution of stepper motor shaft. RTDelta must be divisible to 256!
RTMinError	320	[u16] - Number of ticks missed to raise the error flag
Maximum Temperature	70	[dbl] Temperature (Celsius) after witch controller will automatically turn off feeding power
Sync duration	5	[u16] - Duration of the output synchronization pulse in steps (or microsteps, depending on resolution) between the output synchronization pulses.
Loft Speed	3	[u16] - Speed of the last phase of the backlash operation.

EncVSCP	10	[u16] – It is equal to number of encoder steps per one full step of stepper motor multiply by 4.

6.2.3 State

Title	Description
Current position	[dbl] - Current position of the step motor
CW/CCW	Indicates step motor rotation direction relatively (clockwise / counterclockwise)
Speed	[u16] – 0 – step motor is not moving 1 – step motor is accelerating/decelerating 2 – step motor is moving at full speed
Temperature	[dbl] - Current temperature (Celsius) of power driver
Power	[u16] – 0 – no power 1 – half power (in current reduction mode) 2 – full power
Limit switch 1	Indicates limit switch 1 logical state
Limit switch 2	Indicates limit switch 2 logical state
After Reset	Indicates if device have been reset (true). This indicator is switched to the false state after Set Current Position command (see 5.2.2)
Step	[u8] Step size: 0 – 1/8 step mode 1 – ¼ step mode 2 - ½ step mode 3 – full step mode
Loft	Indicates backlash operation status
RT Error	Indicates rotary transducer error flag (true – error, false – no error)
RT State	Indicates rotary transducer current logical state
Sync In	Indicates the state of the input synchronization pin on the board
Sync Out	Indicates the state of the output synchronization pin on the board
Ext Disconnect	Indicates emergency disable button logical state (true – external power off)
USB powered	Reserved
Voltage	[dbl] – Power supply voltage (Volts)
Working	If TRUE – controller is functioning properly (only for firmware 24.03+)
SM Position in Encoder Units	[dbl] – Step motor position in displayed in units of the encoder
Encoder Current Position	[dbl] – Current position of the encoder

6.2.4 StartPlus

Start plus structure contains information required to start the rotation of the stepper motor as well as the additional information about the system as a whole. Optional fields (the parameters that will not affect the movement of the stepper motor directly) are marked as italics. They are used at the SMCVieW application and can be imported with the help of "Load Profile (uSMC).vi.

Title	Default	Description
Serial Number	*0000000000000001*	*[string] – Serial Number of the controller*
Axis Name	*XYZ*	*[string] – User defined name of the controller or axis*
Loft	0	[u16] – 0 – no backlash operation 1 - carry backlash operation clockwise 2 - carry backlash operation counterclockwise
Wait For Input Signal To Start	FALSE	If TRUE controller will wait for input synchronization pulse to start rotation
Use Slow Start/Stop	TRUE	If TRUE slow start/stop is enabled
Units Name	*steps*	*[string] – User defined name of the units of the real world*
Maximum Speed	*5000*	*[dbl] – User defined maximum speed in the units of the real world*
Speed	1500	[dbl] – Speed of rotation in the units of the real world.
Step	3	[u8] Step size: 0 – 1/8 step mode 1 – ¼ step mode 2 - ½ step mode 3 – full step mode
Point 1 Ticks	*0*	*[i32] – 1st gauge point in ticks*
Point 2 Ticks	*1*	*[i32] – 2nd gauge point in ticks*
Point 1 Units	*0*	*[dbl] – 1st gauge point in units of the real world*
Point 2 Units	*64*	*[dbl] – 1st gauge point in units of the real world*
Limit switch 1 Standoff	*0*	*[dbl] – position from the limit switch to the zero position (used at the SMCVieW "Reset" operation) see section 5.2.7 on page 49.*
Zero Value	*0*	*[dbl] – Position of the controller after the completion of the SMCVieW "Reset" operation, see section 5.2.7 on page 49*
Destination position	0	[dbl] - Destination position
Force Loft	FALSE	If TRUE and destination position is equal to the current position and backlash operation is carried out
Power Off On Stop	*FALSE*	*If TRUE – power will go off after time specified at Power Off Delay (used at SMCVieW, see section 5.2.5 at page 48)*
Power Off Delay	*600*	*[dbl] Time in seconds after witch power will go off in case controller is idle.*
Positioner Name	*Linear*	*[string] – User defined name of the positioner or stepper motor connected to controller*
R, Ohm	*0*	*[dbl] – value of the current sense resistors that should be used in the controller*
I, mA	*0*	*[dbl] – minimal current, required for proper rotation of stepper motor or positioner*

Positioner's Type	*0*	*[u16] – type of the positioner:* *0 – linear* *1 – rotational*
Reset Direction	FALSE	Direction of the "Reset" operation. FALSE – towards maximal value, TRUE – towards minimal value
Manufacturer	*Unknown*	*[string] – name of the positioner's manufacturer*
Model	*Unknown*	*[string] – name of the stepper motor*
Scheme	*4Wire*	*[string] – name of the connection scheme*
Firmware Version	*unknown*	*[string] – version of the controllers firmware obtained by Get Version (uSMC).vi*

6.2.5 Position

Title	Default	Description
Cur Pos Units	0	[dbl] – Current position of the controller in the units of the real world
Set Cur Pos Units	0	[dbl] – Position that will be used by Set Current Position (uSMC).vi in the units of the real world
Lo Limit Units	-33554428	[dbl] – Lowest possible position of the controller in the units of the real world
Hi Limit Units	33554427	[dbl] – Highest possible position of the controller in the units of the real world
Cur Pos Ticks	0	[i32] – Current position of the controller in tics
Set Cur Pos Ticks	0	[i32]– Position that will be used by Set Current Position (uSMC).vi in tics
Lo Limit Ticks	-2147483648	[i32]– Lowest possible position of the controller in tics
Hi Limit Ticks	2147483647	[i32]– Highest possible position of the controller in tics
Minimal Value	-33554428	[dbl] – Minimal value that can be achieved with current experimental set-up in the units of the real world
Maximal Value	33554427.78	[dbl] – Maximal value that can be achieved with current experimental set-up in the units of the real world

6.3 Examples

See Examples on CD in \Development Kit\VIs folder.

7 Dynamic link library USMCDLL.dll for Win2000/XP

Note: This DK package comes with new USMCDLL.dll and its MicroSMC driver (see paragraph 4.1.2). New USMCDLL.dll uses ezusb.sys instead of NI VISA. Note that only one of these drivers can be installed for the one device (see paragraph 4.5).

Note: Makes sure that you use 8SMC1-USBh controllers with firmware usmc2503.usm or higher! In other case update it before programming (see paragraph 5.4.5).

7.1 New version information

MicroSMC development kit now uses MicroSMC driver based on ezusb.sys driver instead of NI VISA. It is capable of controlling up to 32 "8SMC1-USBh" devices (per computer). The new USMCDLL.dll library have some new functions allowing one to perform refreshing of driver when devices are being plugged or unplugged on the fly.

It also allows to use safely multiple processes (programs) controlling one or more devices simultaneously. In that case any "8SMC1-USB" device is shared between these programs.

New DK is much faster than preceding USMCDLL.dll which used NI VISA and LabVieW parts.

USMCDLL.dll is placed in windows/system32 directory so you can now run you program which uses it from any location. First program performing Init function will initiate launch of MicroSMC.exe process that will become that will become available for any other programs.

Last information sent to each controller is saved in the corresponding shared file describing parameters of each device. Any GetXXX function will fill structures with the parameters last sent to the device by any process. These parameters are saved in file that is persistent between sessions.

7.2 General information

Dynamic link library USMCDLL.dll includes all basic commands for stepping motors control. Using it, one can write his own programs for windows to control step motor driver. Example of console interface in c++ code is also enclosed.

All described structures and functions declarations are contained in **USMCDLL.h** C/C++ header file.

7.3 File list

- **USMCDLL.dll** – Windows dynamic link library containing all functions used to control step motor. It is placed in %win%/%sys32% directory
- **%MicroSMC%\USMCDLL.lib** – "lib" file, describing DLL functions, which can be used as input to linker
- **%MicroSMC%\USMCDLL.h** – C/C++ header file with functions and structures declarations

- **%MicroSMC%\MicroSMC.exe** – Shared process file. It is placed in product installation directory. See description bellow.
- **%MicroSMC%\Test** – Sample source code and executable written in C++
- **%MicroSMC%\Driver\ –** Copy of driver and inf file
- **%MicroSMC%\Data\ –** Directory containing parameters of every device ever connected to this computer

7.4 MicroSMC.exe

This application is used to control any interaction with devices so the multiple processes can access single device simultaneously. This executable is also used to restart or close its process from command line. There are two shortcuts available in installation directory:

- **Restart MicroSMC.lnk** – Restart MicroSMC.exe process (>MicroSMC.exe reload)
- **Close MicroSMC.lnk** – Close MicroSMC.exe process (>MicroSMC.exe close). You can also close MicroSMC.exe process using USMC_Close function.

7.5 Functions

7.5.1 General information

All functions except USMC_GetLastErr return error code which is equal to zero value if no error occurred and is nonzero value otherwise. USMC_GetLastErr function is used to retrieve string, describing last error.

Most functions receive some of their arguments in form of reference type variables. These functions can perform corrections of its input variables according to the existing boundaries and precision.

7.5.2 Functions list

Return value	Name / Argument structure	USB Operation	Description on page
Error code	USMC_Init USMC_Devices	IN	72 79
Error code	USMC_GetState USMC_State	IN	72 81
Error code	USMC_SaveParametersToFlash	SETUP/OUT	73
Error code	USMC_SetCurrentPosition	SETUP/OUT	77
Error code	USMC_GetMode USMC_Mode	–	73 80
Error code	USMC_SetMode USMC_Mode	OUT	74 80
Error code	USMC_GetParameters USMC_Parameters	–	74 79
Error code	USMC_SetParameters USMC_Parameters	OUT	75 79

Error code	USMC_GetStartParameters USMC_StartParameters	–	75 80
Error code	USMC_Start USMC_StartParameters	OUT	76 80
Error code	USMC_Stop	SETUP/OUT	76
Error code	USMC_GetEncoderState USMC_EncoderState	IN	78 81
Error code	USMC_Close	–	79
void	USMC_GetLastErr	–	78

7.5.3 USMC_Init

The USMC_Init function initializes dll and/or searches for connected devices. It must be called before any other function of the dll.

```
DWORD USMC_Init( USMC_Devices &Str );
```

Arguments list:

USMC_Devices &Str – OUT – Structure describing connected devices

Return value:

The USMC_Init function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

You need to execute this function at least once before using any other functions of USMCDLL.dll.

The USMC_Init function is used to obtain number of connected devices and their serial numbers. The value between zero and NOD member of USMC_Devices structure minus 1 is a handle to the particular device.

When the function is executed firs time from particular process it initializes dll data. Only if the MicroSMC.exe process is not running or no devices were connected earlier it performs the refresh of connected devices. Any consequent launch of USMC_Init will perform the refresh. This behavior is needed when multiple processes are using multiple devices so the new process will not refresh the device list which is being used by other processes.

If one of the other functions returns error indicating that the particular device is disconnected, one should run the USMC_Init function in order to refresh the device list.

7.5.4 USMC_GetState

The USMC_GetState function reads current state from step motor controller.

```
DWORD USMC_GetState( DWORD Device, USMC_State &Str  );
```

Argument list:

DWORD Device – IN – Device number

USMC_State &Str – OUT – Structure describing current device state

Return value:

The USMC_GetState function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_GetState and USMC_GetEncoderState are the only functions that receive information from controller. They are used to monitor current state of step motor.

The USMC_GetState returns all available information on SM controller except encoder feedback.

7.5.5 USMC_SaveParametersToFlash

The USMC_SaveParametersToFlash function saves previously sent parameters in the flash memory of controller.

```
DWORD USMC_SaveParametersToFlash( DWORD Device  );
```

Argument list:

DWORD Device – IN – Device number

Return value:

The USMC_SaveParametersToFlash function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_SaveParametersToFlash function saves all elements of USMC_Parameters structure and some elements of USMC_Mode structure previously sent to controller using corresponding functions. If no values were sent nothing happens.

7.5.6 USMC_GetMode

The USMC_GetMode function returns USMC_Mode structure that was sent to device or default structure.

```
DWORD USMC_GetMode( DWORD Device, USMC_Mode &Str );
```

Argument list:

DWORD Device – IN – Device number

USMC_Mode &Str – OUT – Structure containing some of parameters (see 7.6.4 on page 80) that can be sent to controller

Return value:

The USMC_GetMode function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_Mode structure is filled with values last sent by the USMC_SetMode function or with default values for this device. Default values can be changed to the desired

one by SMCVieW interface ("SETUP" button) and are stored in \Data\Saves\ directory for any connected device.

7.5.7 USMC_SetMode

The USMC_SetMode function is used to send device parameters contained in USMC_Mode structure to 8SMC1-USBh controller.

DWORD USMC_SetMode(DWORD Device, USMC_Mode &Str);

Argument list:

DWORD Device – IN – Device number

USMC_Mode &Str – IN/OUT – Structure containing some of parameters (see 7.6.4 on page 80) that can be sent to controller

Return value:

The USMC_SetMode function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_SetMode function is used to set some Boolean parameters (see 7.6.4 on page 80) of controller and one non-Boolean field. Some of the parameters describe electro/mechanical construction and are changed rarely, but some of them describe current controller operation.

In order to maintain changes made by this function after controller restart, you should call the USMC_SaveParametersToFlash function after parameters are sent.

Example:

If you want to turn power of the device off, you can check ResetD bit of USMC_Mode structure and call USMC_SetMode function (all other fields received by USMC_GetMode function in that case should be unchanged):

```
USMC_GetMode(Dev, Mode) ;
Mode.ResetD =TRUE;                    // Turn power off
USMC_SetMode(Dev, Mode);
```

If you want to turn power of the device on:

```
USMC_GetMode(Dev, Mode) ;
Mode.ResetD = FALSE;                  // Turn power on
USMC_SetMode(Dev, Mode);
```

7.5.8 USMC_GetParameters

The USMC_GetParameters function returns USMC_Parameters structure last sent to device or default structure.

DWORD USMC_GetParameters(DWORD Device, USMC_Parameters &Str);

Argument list:

DWORD Device – IN – Device number

USMC_Parameters &Str – OUT – Structure containing some of parameters (see 7.6.2 on page 79) that can be sent to controller

Return value:

The USMC_GetParameters function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_Parameters structure is filled with values last sent by the USMC_SetParameters function or with default values for this device. Default values can be adjusted by SMCVieW interface ("SETUP" button) and are stored in \Data\Saves\ directory for all connected device.

7.5.9 USMC_SetParameters

The USMC_SetParameters function sets device parameters contained in USMC_Parameters structure.

```
DWORD USMC_SetParameters( DWORD Device, USMC_Parameters &Str  );
```

Argument list:

DWORD Device – IN – Device number

USMC_Parameters &Str – IN/OUT – Structure containing some of parameters (see 7.6.2 on page 79) that can be sent to controller

Return value:

The USMC_SetParameters function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_SetParameters function is used to set rarely changed parameters (see 7.6.2 on page 79) into controller. Values sent to controller are corrected beforehand according to their limits and precision. You can track these changes.

In order to maintain changes made by this function after controller restart one should call USMC_SaveParametersToFlash function after parameters are sent.

7.5.10 USMC_GetStartParameters

The USMC_GetStartParameters function returns USMC_StartParameters structure last sent to device by means of calling USMC_Start function or default structure.

```
DWORD USMC_GetStartParameters( DWORD Device,
                               USMC_StartParameters &Str  );
```

Argument list:

DWORD Device – IN – Device number
USMC_StartParameters &Str – OUT – Structure containing start parameters

Return value:
The USMC_GetStartParameters function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:
The USMC_StartParameters structure is filled with values last sent by the USMC_Start function or with default values for this device. Default values can be adjusted by SMCVieW interface ("SETUP" button) and are stored in \Data\Saves\ directory for any connected device.

7.5.11 USMC_Start

The USMC_Start function sets start parameters and initiates step motor rotation instantly or delayed (if WSyncIN bit of USMC_StartParameters structure is TRUE) .

```
DWORD USMC_Start( DWORD Device,  int DestPos,  float &Speed,
                  USMC_StartParameters &Str );
```

Argument list:
DWORD Device – IN – Device number
int DestPos – IN – Absolute or relative (see remarks) destination position in micro steps
float &Speed – IN/OUT – Speed of rotation in [units] (steps / ½–steps / ¼–steps / 1/8–steps) per second
USMC_StartParameters &Str – OUT – Structure containing start parameters

Return value:
The USMC_Start function returns error code. Zero value for normal execution and nonzero value otherwise.
Remarks:
The USMC_StartParameters structure describes parameters of step motor rotation. If WSyncIN bit of USMC_StartParameters structure is TRUE then rotation is initiated after next impulse on synchronization input, otherwise rotation is initiated immediately.
DestPos is absolute destination position of controller except the case when WSyncIN bit of USMC_StartParameters structure is TRUE. In that only case DestPos value determines relative to current position of step motor distance.
All reference type values sent to controller are corrected beforehand according to their limits and precision. You can track these changes.
Any call to this function when step motor is rotating causes it to stop immediately and only then perform requested start operation.

7.5.12 USMC_Stop

The USMC_Stop function stops the step motor if it is rotating.

```
DWORD USMC_Stop( DWORD Device  );
```

Argument list:

DWORD Device – IN – Device number

Return value:

The USMC_Stop function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

If slow start/stop mode is enabled (SlStart bit of USMC_StartParameters structure) the USMC_Stop function initiates deceleration like when step motor is automatically stopped when it reaches the destination position. In that case current position will be changed by corresponding deceleration distance; no step loss or step error shall arise.

Step motor is finally stopped when the RUN bit of USMC_State structure is FALSE.

If immediately stepping motor stop required one should use EMReset bit of USMC_Mode structure; step loss in this case is quite possible.

7.5.13 USMC_SetCurrentPosition

The USMC_SetCurrentPosition function overrides current position value of controller.

```
DWORD USMC_SetCurrentPosition( DWORD Device, int Position  );
```

Argument list:

DWORD Device – IN – Device number

int Position – IN – New current position in 1/8 steps (5 least significant bits are ignored)

Return value:

The USMC_SetCurrentPosition function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_SetCurrentPosition function is used to maintain current position value of step motor after controller reset (logical power supply off – don't confuse with powering off the step motor by command from interface or by the USMC_SetMode function, in latter case current position value is persistent). It should be called when the AReset bit of USMC_State structure is TRUE. The argument Position should be equal to last read CurPos field of USMC_State structure when the step motor's power is already off (again AReset bit of USMC_State structure is TRUE).

5 least significant bits (corresponds to 4 full steps) of argument Position are ignored because they correspond directly to the electro–mechanical state of step motor and thus cannot be persistent during step motor power off. When one turns step motor's power off, it makes minimal required rotation to the nearest "zero–state". Because of that if maintaining of the step motor's precise position is required one should follow procedure described above (or just use SMCVieW).

7.5.14 USMC_GetEncoderState

The USMC_GetEncoderState function returns structure representing current position of encoder and additionally the synchronized position of SM in encoder units.

```
DWORD USMC_GetEncoderState( DWORD Device,
                            USMC_EncoderState &Str  );
```

Argument list:

DWORD Device – IN – Device number

USMC_EncoderState &Str – IN/OUT Structure containing encoder state

Return value:

The USMC_GetEncoderState function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_GetState and USMC_GetEncoderState are the only functions that receive information from controller. They are used to monitor current state of step motor.

The USMC_GetEncoderState function returns position of encoder and SM read at the same time. These values can be used to determine error of SM position if any.

The values are presented as integers in a half of encoder step precision. To make them comparable with USMC_State.CurPos value one should divide them by USMC_Parameters.EncMult parameter.

RotTrErr parameter of USMC_State structure is set (when encoder is enabled) if the difference between USMC_GetEncoderState parameters exceeds the USMC_Parameters.RTMinError parameter. In such cases SM will behave the same way as when rotary transducer error occurs.

7.5.15 USMC_GetLastErr

The USMC_GetLastErr function returns string containing last error (if any) description.

```
void USMC_GetLastErr( char *str,  size_t len );
```

Argument list:

*char *str* – OUT – Output string buffer (for single byte characters)

size_t len – IN – Output string buffer length in bytes

Return value:

The USMC_GetLastErr function has no return values.

Remarks:

The USMC_GetLastErr function fills the buffer with null terminated string (if length is enough). Don't forget to allocate corresponding (to len argument) amount of memory.

7.5.16 USMC_Close

The USMC_Close function returns string containing last error (if any) description.

DWORD USMC_Close(void);

Argument list:

The USMC_Close function has no arguments.

Return value:

The USMC_Close function returns error code. Zero value for normal execution and nonzero value otherwise.

Remarks:

The USMC_Close closes MicroSMC.exe process. MicroSMC.exe will be automatically restarted on the next execution of USMC_Init function.

7.6 Structures

7.6.1 USMC_Devices

The USMC_Devices structure describes devices connected to computer.

Type	Name	Description
DWORD	NOD	Number of devices connected to computer
char **	Serial	Array of pointers to 16–byte ASCII strings of length – NOD
char **	Version	Array of pointers to 4–byte ASCII strings of length – NOD

7.6.2 USMC_Parameters

Type	Name	Description
float	AccelT	Acceleration time (in ms)
float	DecelT	Deceleration time (in ms)
float	PTimeout	Time (in ms) after which current will be reduced to 60% of normal
float	BTimeout1	Time (in ms) after which speed of step motor rotation will be equal to the one specified at BTO1P field in this structure
float	BTimeout2	Time (in ms) after which speed of step motor rotation will be equal to the one specified at BTO2P field in this structure
float	BTimeout3	Time (in ms) after which speed of step motor rotation will be equal to the one specified at BTO3P field in this structure
float	BTimeout4	Time (in ms) after which speed of step motor rotation will be equal to the one specified at BTO4P field in this structure
float	BTimeoutR	Time (in ms) after which reset command will be performed (see 5.4.7 at page 53)
float	BTimeoutD	This field is reserved for future use
float	MinP	Speed (steps/sec) while performing reset operation
float	BTO1P	Speed (steps/sec) after BTIMEOUT1 time has passed (see 5.4.8 at page 54)
float	BTO2P	Speed (steps/sec) after BTIMEOUT2 time has passed (see 5.4.8 at page 54)
float	BTO3P	Speed (steps/sec) after BTIMEOUT3 time has passed (see 5.4.8 at page 54)

float	BTO4P	Speed (steps/sec) after BTIMEOUT4 time has passed (see 5.4.8 at page 54)
WORD	MaxLoft	Value in full steps that will be used performing backlash operation
DWORD	StartPos	Current Position saved to FLASH. Refer to test.cpp for implementing this functionality. Should be set to 0 for correct reloading of current position in SMCVieW program
WORD	RTDelta	Revolution distance – number of full steps per one full revolution
WORD	RTMinError	Number of full steps missed to raise the error flag
float	MaxTemp	Maximum allowed temperature (centigrade degrees)
BYTE	SynOUTP	Duration of the output synchronization pulse (see 5.4.13 at page 57)
float	LoftPeriod	Speed (steps/sec) of the last phase of the backlash operation
float	EncMult	Encoder step multiplier. Should be <Encoder Steps per Revolution> / <SM Steps per Revolution> and should be integer multiplied by 0.25

7.6.3 USMC_StartParameters

Type	Name	Description
BYTE	SDivisor	Step is divided by this factor (1,2,4,8)
BOOL	DefDir	Direction for backlash operation (relative) (see 5.4.15 at page 59)
BOOL	LoftEn	Enable automatic backlash operation (works if slow start/stop mode is off)
BOOL	SlStart	If TRUE slow start/stop mode enabled
BOOL	WSyncIN	If TRUE controller will wait for input synchronization signal to start
BOOL	SyncOUTR	If TRUE output synchronization counter will be reset
BOOL	ForceLoft	If TRUE and destination position is equal to the current position backlash operation will be performed

7.6.4 USMC_Mode

Type	Name	Masked *	Saved to flash	Description
BOOL	PMode	–	YES	Turn off buttons (TRUE - buttons disabled)
BOOL	PReg	–	YES	Current reduction regime (TRUE - regime is on)
BOOL	ResetD	–	YES	Turn power off and make a whole step (TRUE - apply)
BOOL	EMReset	–	–	Quick power off (see 5.4.6 at page 52)
BOOL	Tr1T	–	YES	Limit switch 1 TRUE state (TRUE : +3/+5B; FALSE : 0B)
BOOL	Tr2T	–	YES	Limit switch 2 TRUE state (TRUE : +3/+5B; FALSE : 0B)
BOOL	RotTrT	–	YES	Rotary Transducer TRUE state (TRUE : +3/+5B; FALSE : 0B)
BOOL	TrSwap	–	YES	If TRUE, Limit switches are treated to be swapped
BOOL	Tr1En	–	YES	If TRUE Limit switch 1 Operation Enabled
BOOL	Tr2En	–	YES	If TRUE Limit switch 2 Operation Enabled
BOOL	RotTeEn	–	YES	If TRUE Rotary Transducer Operation Enabled
BOOL	RotTrOp	–	YES	Rotary Transducer Operation Select (stop on error if TRUE)
BOOL	Butt1T	–	YES	Button 1 TRUE state (TRUE : +3/+5B; FALSE : 0B)
BOOL	Butt2T	–	YES	Button 2 TRUE state (TRUE : +3/+5B; FALSE : 0B)
BOOL	ResetRT	YES	–	Reset Rotary Transducer Check Positions (need one full revolution before it can detect error)
BOOL	SyncOUTEn	–	YES	If TRUE output synchronization enabled
BOOL	SyncOUTR	YES	–	If TRUE output synchronization counter will be reset
BOOL	SyncINOp	–	YES	Synchronization input mode: TRUE - Step motor will move one time to the DestPos FALSE - Step motor will move multiple times by DestPos microsteps as distance

DWORD	SyncCount	–	YES	Number of steps after which synchronization output signal occurs
BOOL	SyncInvert	–	YES	Set this bit to TRUE to invert output synchronization polarity
BOOL	EncoderEn	–	YES	Enable Encoder on pins {SYNCIN,ROTTR} - disables Synchronization input and Rotary Transducer
BOOL	EncoderInv	–	YES	Invert Encoder Counter Direction
BOOL	ResBEnc	YES	–	Reset <EncoderPos> and <ECurPos> to 0
BOOL	ResEnc	YES	–	Reset <ECurPos> to <EncoderPos>

* – These Boolean values will be automatically cleared by USMC_GetMode (to FALSE)

7.6.5 USMC_State

Type	Name	Description
int	CurPos	Current position (in 1/8 steps)
float	Temp	Current temperature of the power driver
BYTE	SDivisor	Step is divided by this factor
BOOL	Loft	Indicates backlash status
BOOL	FullPower	Full power if TRUE
BOOL	CW_CCW	Current direction of rotation (relatively to some direction – dependent on step motor circuits connection and on its construction)
BOOL	Power	If TRUE then Step Motor power is ON
BOOL	FullSpeed	If TRUE then full speed. Valid in "Slow Start" mode only
BOOL	AReset	TRUE After Device reset, FALSE after "Set Position"
BOOL	RUN	TRUE if step motor is rotating
BOOL	SyncIN	Logical state directly from input synchronization PIN (pulses treated as positive)
BOOL	SyncOUT	Logical state directly from output synchronization PIN (pulses are positive)
BOOL	RotTr	Indicates current rotary transducer logical press state
BOOL	RotTrErr	Indicates rotary transducer error flag (reset by USMC_SetMode function with ResetRT bit – TRUE)
BOOL	EmReset	Indicates state of emergency disable button (TRUE – Step motor power off)
BOOL	Trailer1	Indicates Limit switch 1 logical press state
BOOL	Trailer2	Indicates Limit switch 2 logical press state
float	Voltage	Power supply voltage (Volts)

7.6.6 USMC_EncoderState

Type	Name	Description
Int	EncoderPos	Current position measured by encoder
Int	ECurPos	Current position (in Encoder Steps) - Synchronized with request call

7.7 Examples

7.7.1 C++ example

The example placed in **%MicroSMC%\Test** folder.

This sample program is written on C++ language as console application. It demonstrates usage of all described functions. Moreover for all members of all structures there is a detailed description in form of formatted string output with function "printf".

8 Dynamic link library USMCDLL.dll for WM

8.1 General information

DLL is designed for control and monitor 8SMC1-USBh controllers using mobile devices with Microsoft Windows Mobile 5.0 or higher operation system. It includes all basic commands for stepping motors control. Using it, one can write his own programs for Windows Mobile to control 8SMC1-USBh controllers. Example of window-based interface in c++ code is also enclosed. Functionality of USMCDLL.dll for Microsoft Windows Mobile 5.0 is just the same as USMCDLL.dll for Microsoft Windows XP (see chapter 7).

Note: Makes sure that you use 8SMC1-USBh controllers with firmware usmc2504.usm or higher! In other case update it before programming (see paragraph 5.4.5).

8.2 File list

8.2.1 Host PC files

- **%MicroSMC for WM%\USMCDLL.lib** – "lib" file, describing DLL functions, which can be used as input to linker.
- **% MicroSMC for WM %\USMCDLL.h** – C/C++ header file with functions and structures declarations.
- **% MicroSMC for WM %\CEAppMgrSetup.exe** – executable file, needed for invoke Microsoft ActiveSync application manager. Application manager installs driver files from **Mobile** subdirectory onto your mobile device.
- **% MicroSMC for WM %\ReadMe.rtf** – rich text file containing installation notes.
- **% MicroSMC for WM %\Test** – Sample source code and executable on C++.
- **% MicroSMC for WM %\Driver\ –** Copy of driver and USMC interface dll (CtrlDrv.dll and USMCDLL.dll).
- **% MicroSMC for WM %\Mobile\ –** Windows Mobile 5.0 driver installation files.

8.2.2 Mobile device files

- **%win%\USMCDLL.dll** – Windows Mobile 5.0 dynamic link library containing all functions used to control step motor.
- **%win%\CtrlDrv.dll** – Windows Mobile 5.0 dynamic link library, driver itself.
- **%MicroSMC%\USMCDLLTest.exe** – Windows Mobile 5.0 executable file, compiled sample project.

8.3 Functions

All functions and structures are just the same as in USMCDLL.dll for Microsoft Windows XP. It description see in chapter 7.

8.4 Test Applications

There are two test applications in this package for control 8SMC1-USBh with mobile devices. It is placed in **%MicroSMC%** folder on your mobile device. USMCTest.exe demonstrates general possibilities of USMCDLL.dll, USMCPresentation.exe is a complete application for cycle motion of motorized translation stage with two limit switches.

Note that applications may not work as expected if they're launched both at one time.

8.5 Examples

8.5.1 C++ example

The example source code placed in **%MicroSMC for WM%\Test** folder on host PC, executable files in "**%Program Files%\%MicroSMC%"** on mobile device.

This sample program is written on C++ language as dialog-based application. It demonstrates usage of all USMCDLL.dll functions. Moreover for all members of all structures there are detailed descriptions in form of formatted string output ("sprintf" function).