Using SimplexMotion integrated motors for simple standalone applications

This document describes how to use the SimplexMotion integrated motor units for simple motion tasks where the motor is manually controlled. The control interface can use buttons, switches, potentiometers and encoders.
In these applications the communication interface of the motor (USB or RS485) is only used for configuring the device, and not during actual operation.
A number of different control schemes are discussed and parameter files for these configurations are available at the web site for direct download to the motor unit.

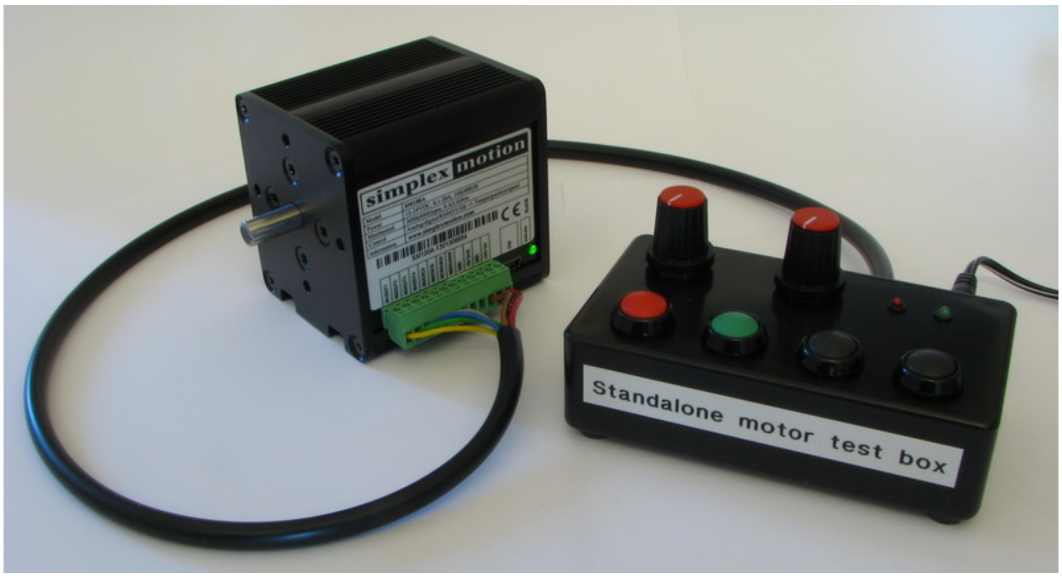


Table of contents:

1 The SimplexMotion integrated motor

The SimplexMotion integrated motor is a complete servo motor with a brushless motor, position feedback sensor and control electronics, all integrated in the same enclosure. This makes for a cost efficient motion control system that is easy to setup and use.

The interfaces to the motor consist of a USB port and a serial port using the RS485 standard with the Modbus RTU protocol. These interfaces can be used for motor configuring and for continuous control from a top level system. But to allow standalone operation there is also a number of analog inputs and digital inputs/outputs that can be configured for custom use.

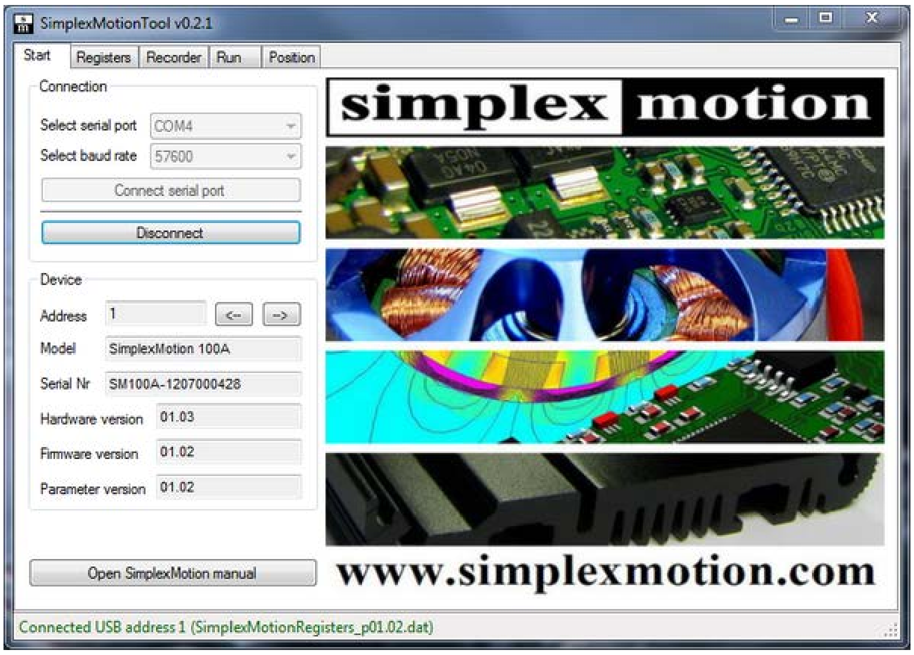
Both speed and position control modes are supported by the motor. Ramp control is also available so that the position or speed is changed in a smooth fashion when it is changed from one setting to a new one.

The motor position sensor measures the motor position with a resolution of 4096 position per shaft revolution. The motor position is stored as a 32 bit integer value in the motor unit.

The configuration and behavior of the motor is entirely controlled by its internal registers. These registers can be read and written using the USB or RS485/Modbus interfaces.

1.1 Software tool for motor configuration

A support software called 'SimplexMotionTool' is available at www.simplexmotion.com to allow access to the set of registers in the motor unit that defines its behavior.



Select serial port COM4
Select baud rate 57600
Address 1
Model SimplexMotion 100A
Serial Nr SM100A-1207000428
Hardware version 01.03
Firmware version 01.02
Parameter version 01.02

Connected USB address 1 (SimplexMotionRegisters_p01.02.dat)

The ‘register’ tab of the software is continuously updated with the register contents once the communication has been established using the ‘start’ tab. When the registers are updated by editing the numeric values they are immediately updated in the working registers in the motor unit. But make sure to select ‘Write to memory’ when finished as the register contents will be lost when the power is turned off otherwise. There are also buttons to read the register contents from a file, or to write all parameters to a file for backup.

The ‘run’ tab in the software is useful for general motor testing, but please note that this will change the registers and the motor settings, which will affect the current configuration. The same goes for the ‘position’ tab.
Please consult the software manual for further information on how to use the software.

Using supplied parameter files requires the following workflow with the software:
1. Get the parameter file to use from www.simplexmotion.com.
2. Start the SimplexMotionTool software.
3. Connect the motor unit to the PC with a USB cable.
4. On the ‘Start’ tab, select ‘Connect USB’. The text in the bottom status field should show the connection in green text.
5. On the ‘Register’ tab, select ‘Read from File’. Select the parameter file to download to the motor unit.
6. Select ‘Write to memory’ to permanently store the new parameters.

2 Support for standalone control

This application note will focus on standalone operation, and we will now look at the registers in the motor that are used to do the necessary configuration. For full understanding it may be necessary to also read the motor unit datasheet.

Since there is a lot of flexibility, the configuration also becomes time consuming. For simple setups it is possible to use the example parameter files instead of understanding all the configuration settings. Please see chapter 3 for examples of use.

2.1 Control of speed or position

The motor can either control speed or position. The selection is done by setting the <Mode> register. Since many standalone applications change the input control value in steps it is important to use control modes with ramp control. This means that the actual target value will be changed in a smooth fashion by using a set acceleration/deceleration parameter.
The motor control regulator handles the actual closed loop motion control, and the regulator always use motor position for control. When speed control mode is in action, the motor position for the regulator is continuously calculated from the target speed.

The following table shows the important registers to configure:

Register	**Description**
<Mode>	This register selects the motor control mode. 21 = Position with ramp control. Will set the motor position according to the input target value. 33 = Speed with ramp control. Will set the motor speed to be according to the input target value.
<MotorTorqueMax>	This register sets the maximum torque allowed by the motor. The value is in units of [0.001Nm].
<RampSpeedMax>	Sets the maximum speed allowed by the motor. The unit is [positions/second * 1/16]. Value = RPM * 256 / 60.
<RampAccMax>	For ramp control we need to set the acceleration to use. The unit is [positions/second^2 * 1/256]. Value = RPM/s^2 / 225. Typical values are 100-1000.
<RampDecMax>	This registers sets the deceleration to use. Sometimes it is important to have separate settings for acceleration and deceleration. Unit is same as for <RampAccMax>. Please note that high decelerations with high inertia loads can cause overvoltage as the rotational energy is converted to electrical energy by the motor. This is potentially harmful to the motor unit.

For absolute position control it is necessary to establish a reference position at system startup. The motor position stored in the motor is lost when the power is switched off. To establish an absolute position reference it is common to do a ‘homing sequence’ at startup, where the motor is rotated in one direction until an external switch is activated to signal that the reference position has been reached. The SimplexMotion motor unit has a homing feature that can be configured to accomplish this, see the datasheet for more information.

2.2 Target value

Once the control mode has been decided the next step is to configure the source of target control values. The target value is used to set the wanted position or speed, depending on selected control mode. The motor unit supports several different sources for the target value, and there are features for manipulating the target value before use.

The following table summarizes the target setup registers:

Register	Description
<TargetInput>	Target value for regulator. This value is used when <TargetSelect> = Register. Signed 32 bit value.
<TargetSelect>	Sets the target source according to:

Value	Name	Description
0	Register	Target is set by a register content. The value to use is written to register <TargetInput>.
1	AIN1	Analog value from AIN1 is used as target. Value 0..65535
2	AIN2	Analog value from AIN2 is used as target.
3	AIN3	Analog value from AIN3 is used as target.
4	AIN4	Analog value from AIN4 is used as target.
5	Encoder	Encoder interface is used for target values. The encoder can be set for quadrature encoder input or Step/Dir interface for step motor emulation.
6	Pulse	A digital input pulse length is used to set target values. Compatible with RC servo pulses. Not yet implemented.

Register	Description
<TargetMul>	Value to multiply with input target value before used by the regulator.
<TargetDiv>	Value to multiply with input target value before used by the regulator.
<TargetOffset>	Value to add to the input target before it is used by the regulator. The Offset is applied after TargetMul and TargetDiv.
<TargetMin>	Minimum allowed value for target value
<TargetMax>	Maximum allowed value for target value
<TargetHysteresis>	Hysteresis value to remove noise from target values. This is typically useful when the target source is an analog input. Applied after Mul/Div/Offset. Typical values 0..100.
<TargetFilter>	Allows filtering of target values to reduce noise and limit rate of change. This is useful when the target source is an analog input. 0 = no filtering, increasing values allows more filtering. Typical values 0..7.
<TargetPresent>	The current target value as it is sent to the regulator. Useful for troubleshooting.

For standalone operation it is possible to use any of the target sources, depending on the application. The most straightforward one might be to use an analog input with a potentiometer as a voltage divider to set the target value. But if we want simple buttons to control the target value we need to use the ‘Register’ setting and use another feature to manipulate the <TargetInput> register value.

2.3 Events

The motor unit supports a feature called ‘event’. These events can be used to manipulate registers from other register contents. This feature can for example implement a digital input connected to a button to start the motor or set a certain motor speed.
Events are made up of two parts; A triggering condition and an action part consisting of a register manipulation. The triggering condition is set up using a specified register, an operator and a specified value. The trigger is active if the result from the trigger operation is non-zero.
The action part is very similar and uses a specified register, an operator, a specified value and a destination register for the result. There are 20 different events available for configuration. Each event is evaluated for each regulator period, currently at 2kHz rate.

The following registers configure one event:

Register	Description
<EventControl>	Control register for event.

Bits	Description
0..3	Trigger operation Used to determine if trigger condition is met.
4..7	Trigger filter Allows filtering of trigger condition. Values 0..15 corresponds to filter delay times of 1, 2, 4, 8, 16, 32, 64, 128, … 32767 regulator periods.
8..9	Trigger condition

	<table><tr><td></td><td>0 = Active, 1 = Edge, 2 = Repeat.</td></tr><tr><td>10..13</td><td>Data operation Used to manipulate register when event is executed.</td></tr></table>
<EventTrgReg>	Trigger register number.
<EventTrgData>	Trigger data value. 16-bit value to use with trigger register and operator.
<EventSrcReg>	Source register number.
<EventSrcData>	Source data value. 16-bit value to use with source register and operator.
<EventDstReg>	Destination register to write the event execution result to.

The <EventControl> register configures the main options for each event. Setting the register to 0 disables the event. The available operators are the same for both the trigger condition and the action:

Value:	**Operator:**	
0		Always true
1	=	Equal
2	!=	Not equal
3	<	Less than
4	>	Greater than
5	or	Bitwise or
6	nor	Bitwise not or
7	and	Bitwise and
8	nand	Bitwise not and
9	xor	Bitwise exclusive or
10	nxor	Bitwise not exclusive or
11	+	Add
12	-	Subtract
13	*	Multiply
14	/	Divide
15	Literal	Takes literal value only

The trigger filtering makes it possible to remove noise from inputs and also introduces a delay in the triggering. The later can be useful to allow a constantly active triggering condition to cause the event to repeatedly execute at a set rate.

There are also a number of different trigger types, as shown in the following table:

Trigger type	**Description**
0	Active. Event is performed each time the filtered trigger condition is true.
1	Edge. Event is only performed the first time the filtered event becomes true. The condition has to become false before next trigger can occur.
2	Repeat. Event is performed repeatedly while the trigger condition is true, but the filter is reset each time so that the filter creates a time delay between event executions.

The event feature is quite useful for many custom applications, but can be somewhat cumbersome to get used to. See the motor unit datasheet for a more detailed description.

3 User control methods

Now we will look closer to different user control schemes for typical standalone applications. We will cover some typical setups and there are available parameter files for these so that users do not have to get deeply involved with event configuring etc. The parameter files can also form a good starting point for further customization.

3.1 Fixed control

The most basic configuration is a fixed setup that defines the motor behavior when the power is switched on. The only available control will be power on/off.
Note that the <Mode> register is loaded from the <ModeStartup> register value at startup, so the correct operating mode should be entered into the <ModeStartup> register.

A power switch is used to turn the motor on/off

Parameter file	Description
Param_SM100A_StandaloneSpeedFixed	Configuration for constant speed at 1000rpm. No input control other than power on/off.

3.2 Control using buttons

In many applications it is convenient to use normal buttons to control the motion. Here we explore common such configurations.

Buttons can be connected to any of the inputs IN1-IN6. There are configurable internal pull up resistors, so buttons can simply be connected between the inputs and GND.

3.2.1 Multiple fixed control

Here we show a configuration with several fixed target values. 4 buttons at IN1-4 are used to select value. Events are used to set the <TargetInput> register when triggered by button presses.

The necessary events for the speed selection are shown in this table:

Description	Trigger	Filter	Condition	Action	<EventControl>
Button 1 sets speed = 0 rpm	<Input> AND 0x0001	3	Edge	<TargetInput> = 0	0xF137
Button 2 sets speed = 1000 rpm	<Input> AND 0x0002	3	Edge	<TargetInput> = 4267	0xF137
Button 3 sets speed = 2000 rpm	<Input> AND 0x0004	3	Edge	<TargetInput> = 8533	0xF137
Button 4 sets speed = 3000 rpm	<Input> AND 0x0008	3	Edge	<TargetInput> = 12800	0xF137

The available parameter files are:

Parameter file	Description
Param_SM100A_StandaloneSpeedSelect	Configuration for constant speed at selectable 0/1000/2000/3000 RPM.
Param_SM100A_StandalonePosSelect	Configuration for fixed positions at 0/10/20/30 shaft revolutions

3.2.2 Up/Down control

In this configuration we use 4 buttons according to:

Button	Input	Description
Off	IN1	Turns motor off
On	IN2	Turns motor on
Down	IN3	Decrease position/speed
Up	IN4	Increase position/speed

4 buttons are used: Off, On , Speed down, Speed up

Events are used to set the <TargetInput> register. The event trigger filtering feature is used to cause a delay that allows the up/down buttons to be held pressed constantly for stepwise changes. Events are also implemented to limit the speed range to 10-2000 rpm. Limiting is possible with <TargetMin> and <TargetMax> registers, but that will not limit the <TargetInput> contents, so using events provides a better solution.

The necessary events for the speed control setup:

Description	Trigger	Filter	Condition	Action	<EventControl>
Button 1 turns motor off	<Input> AND 0x0001	3	Edge	<Mode> = 0	0xF137
Button 2 turns motor on	<Input> AND 0x0002	3	Edge	<Mode> = 33	0xF137
Button 3 decreases speed	<Input> AND 0x0004	3	Edge	<TargetInput> = <TargetInput> - 100	0xC137
Button 4 increases speed by 50 rpm	<Input> AND 0x0008	3	Edge	<TargetInput> = <TargetInput> + 100	0xB137
Button 3 held decreases speed in repeated steps of 50 rpm	<Input> AND 0x0004	9	Repeat	<TargetInput> = <TargetInput> - 213	0xC297
Button 4 held increases speed in repeated steps of 50 rpm	<Input> AND 0x0008	9	Repeat	<TargetInput> = <TargetInput> + 213	0xB297
If speed < 50 rpm, set to 50 rpm	<TargetInput> < 213	0	Active	<TargetInput> = 213	0xF003
If speed > 2000 rpm, set to 2000 rpm	<TargetInput> > 8533	0	Active	<TargetInput> = 8533	0xF004

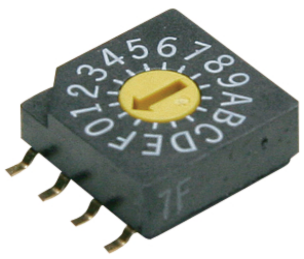
And the available parameter files for up/down control:

Parameter file	Description
Param_SM100A_StandaloneSpeedUpDown	Speed can be changed up/down by 50 rpm in the range 50-2000 rpm.
Param_SM100A_StandalonePosUpDown	Position can be increased/decreased in steps of 0.1 shaft revolution in a range of 0-10 revolutions.

3.3 Control using rotary switch

With a binary coded rotary switch we can easily select between a larger number of fixed control values. By using a 16 position switch and 4 inputs IN1-4 we can use 16 fixed target values.
The rotary switch (or 4 separate switches) should be connected according to:

IN4	IN3	IN2	IN1	Position
Off	Off	Off	Off	1
Off	Off	Off	On	2
Off	Off	On	Off	3
Off	Off	On	On	4
Off	On	Off	Off	5
Off	On	Off	On	6
Off	On	On	Off	7
Off	On	On	On	8
On	Off	Off	Off	9
On	Off	Off	On	10
On	Off	On	Off	11
On	Off	On	On	12
On	On	Off	Off	13
On	On	Off	On	14
On	On	On	Off	15
On	On	On	On	16



Some kind of binary coded rotary switch is used

An event is constantly triggered and masks out the IN1-4 inputs from the <Input> registers and puts the result in the <TargetInput> register. Then the <TargetMul> register can be used to scale the input properly. Offsetting the scale can be accomplished with the <TargetOffset> register.

The events used:

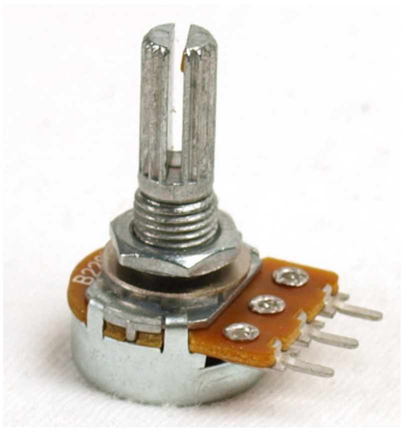
Description	Trigger	Filter	Condition	Action	<EventControl>
Takes binary code from inputs and masks out first 4 bits	Always	0	Active	<TargetInput> = <Input> AND 0x000F	0x7000

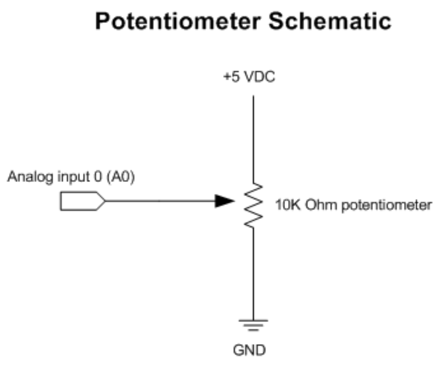
Available parameter files implementing binary coded control:

Parameter file	Description
Param_SM100A_StandaloneSpeedBinary	Speed can be set to 16 values 0 – 1500 rpm in steps of 100 rpm. <TargetMul> = 1280 and <TargetDiv> = 3 provides exact scaling of 100 rpm / step.
Param_SM100A_StandalonePosBinary	Position can be set to 16 values covering one shaft revolution, with steps of 1/16 of a revolution.

3.4 Control using potentiometer input

In some applications it is more convenient to use an analog input for speed or position control, since fixed positions may be limiting. An easy way to do this is to set the <TargetSelect> register to use an analog input directly as the target value. In this example we use the IN1 for analog control with a potentiometer connected as a voltage divider between +5V and GND. We use the IN3 input for motor off and the IN4 input as motor on buttons.
Note that the value from the analog input is in the range 0-65535, and this needs scaling to be useful for target speed setting.





A typical potentiometer and the required connections

The events for the speed control version:

Description	Trigger	Filter	Condition	Action	<EventControl>
Button 3 turns motor off	<Input> AND 0x0004	3	Edge	<Mode> = 0	0xF137
Button 4 turns motor on	<Input> AND 0x0008	3	Edge	<Mode> = 33	0xF137

And the available parameter files:

Parameter file	Description
Param_SM100A_StandaloneSpeedAnalogOnOff	Speed can be set by analog value in the range 0-3000 rpm + On/Off control
Param_SM100A_StandalonePosAnalogOnOff	Position can be set by analog value across 100 revolutions + On/Off control

3.5 Control using encoder input

The motor unit has an encoder interface that is compatible with incremental quadrature encoders. The input position from the encoder can be used as target value directly. This is a simple way to follow the rotation of another mechanical part or motor. Electronic gearing can be accomplished by the <TargetMul> and <TargetDiv> registers.
The encoder interface can also be configured for step/direction inputs, which is common for step motor control systems. See separate application notes on CNC applications where this feature is used.

A typical incremental encoder

In this case we do not need any events, just configure the encoder interface (see the datasheet) and select it by setting the <TargetSelect> register.

The available parameter files:

Parameter file	Description
Param_SM100A_StandaloneSpeedEncoder	Speed is changed by use of encoder
Param_SM100A_StandalonePosEncoder	Position is changed by use of encoder

4 Troubleshooting

With the SimplexMotionTool software it is possible to inspect the internal registers in the motor while it is in operation. This is very useful for troubleshooting. Some examples are:

Check	Description
Button inputs	The buttons are binary coded into the register <Input>. Activating the IN1 input should show up as the least significant bit in the <Input> register. The polarity of the inputs can be changed by the <InputPolarity> threshold (Normal setting = 15, which means that IN1-IN4 is active low, which is appropriate for buttons connected to GND). For IN1-IN4 the threshold level that discriminates between low and high input level can be set using the <InputPolarity> register.
Analog input	The analog values on IN1-IN4 can be read from registers <Analog1> - <Analog4>. The range is 0-65535 for a voltage of 0-5V.
Target value manipulation	To verify that the events used cause the correct effect on the <TargetInput> register, the register content can be viewed in the 'register' tab of the SimplexMotionTool software while buttons are pressed. The <TargetPresent> register shows the target value after manipulation, and this is the actual value used by the motor control system.
Actual motor RPM	By using the 'run' tab in the SimplexMotionTool software it is possible to see the real time speed of the motor in rpm units. The actual motor speed is also available in the <MotorSpeed> register, but then in the motor native u nits of [positions/s * 1/16].

5 Further information

For a deeper understanding of the SimplexMotion100A product please consult the datasheet available at:
http://simplexmotion.com/support/documentation/
At the same place the manual for the SimplexMotionTool software can be found.

There are other application notes available for other application areas at:
http://simplexmotion.com/support/application-notes/