

# TRINAMIC 芯片快速使用指导

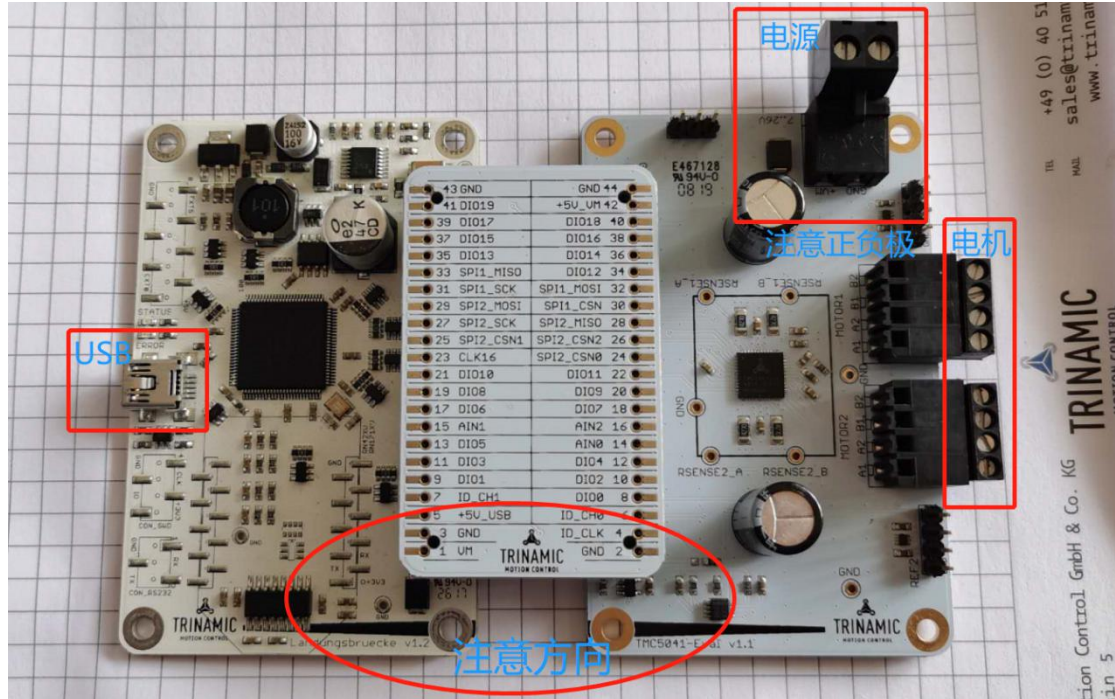


## 目 录

一：硬件部分	3
二：TMCL-IDE 软件部分	
1. TMCL-IDE 软件下载安装	3
2. 电流设置	4
3. 速度控制	5
4. 位置控制	7
5. 示波器功能	12
6. 不同斩波模式切换	12
7. 力矩模式 Stallguard 配置	15
8. 节能 Coolstep 配置	20
9. 静音 Stealthchop 配置	28
(1) 第一代 Stealthchop	28
(2) 第二代 Stealthchop	32
10. 高速 Spreadcycle 配置	35
11. 如何移植开发板代码	36
12. 视频	38

## 一：硬件准备

1. 请将 MCU 板，转接板和 TMC5041-EVAL 按照下图装配一起，请注意方向
  2. 将电机线，USB 线接好
  3. 正确连接电源线，确认好正负极，**电源接反会烧坏开发板**
- 上电之后 MCU 板显示闪烁绿灯。

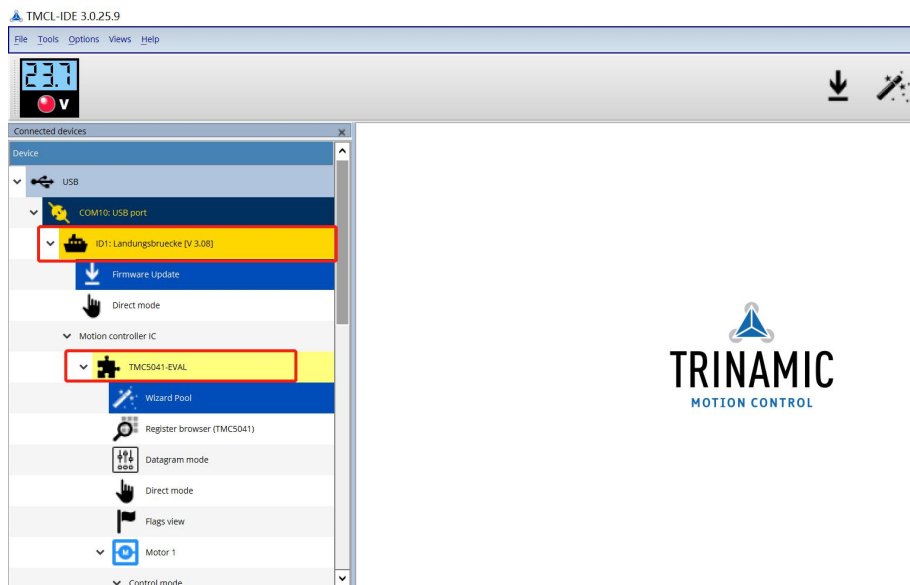


## 二：TMCL-IDE 软件

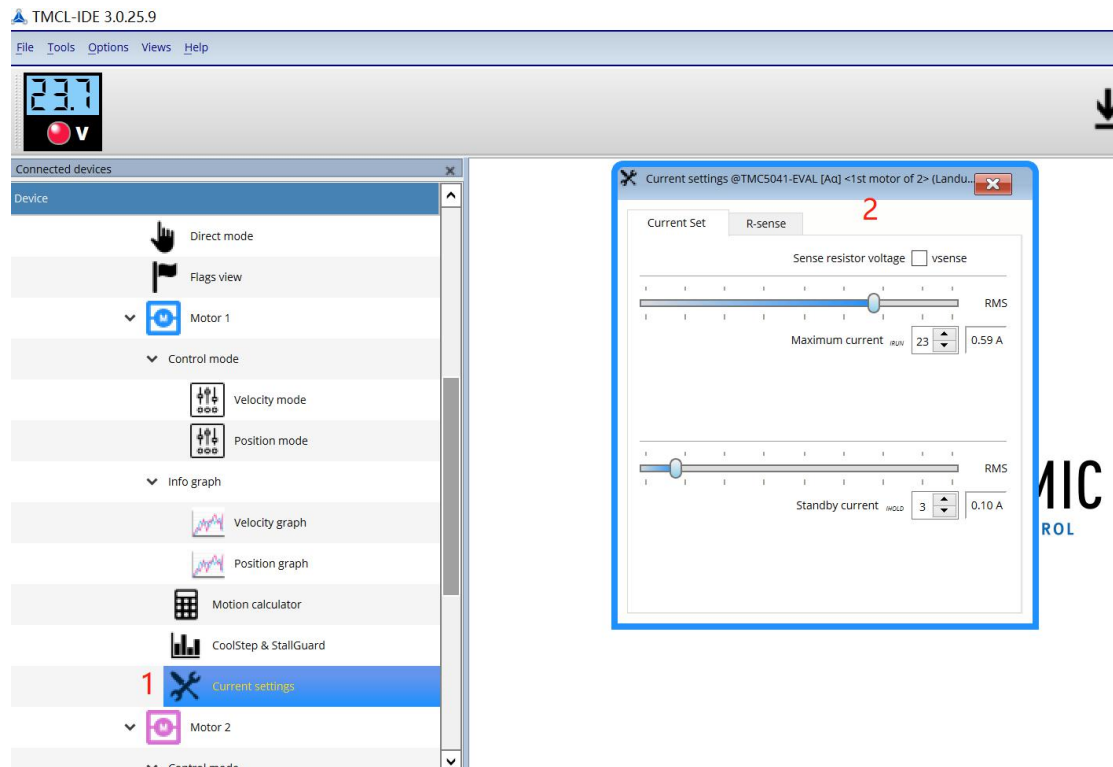
安装 TMCL-IDE 软件，软件下载链接

<https://www.trinamic.com/support/software/tmcl-ide/>

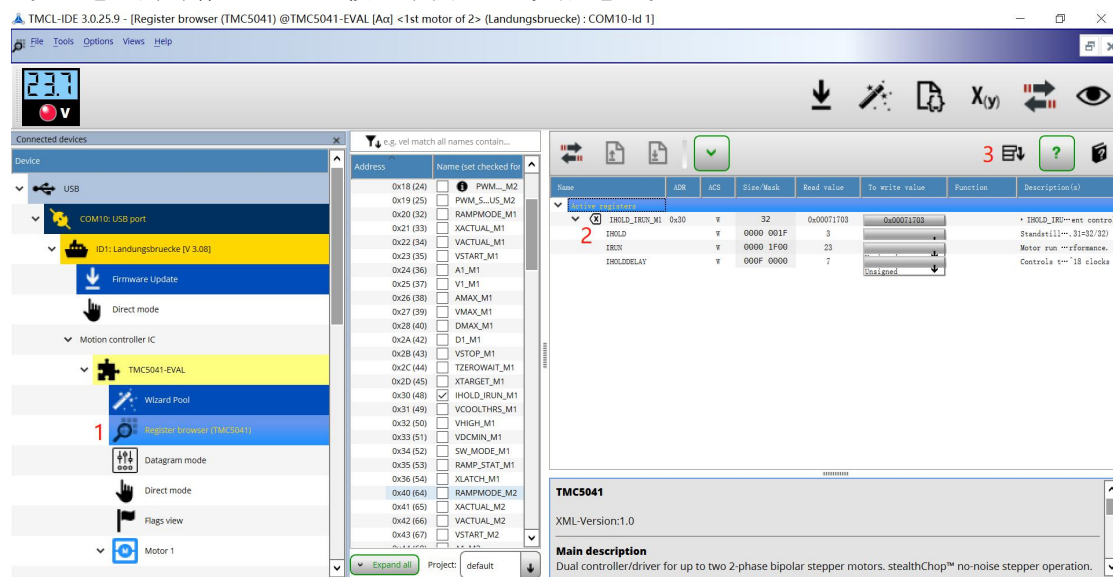
1. 开发板的 USB 驱动会自动安装，打开 TMCL-IDE 软件后会自动识别所连接的型号，如下



2. 设置运行电流和静止电流，根据电机的电流设置运行电流 Maximum Current 和静止电流 Standby Current 如下图



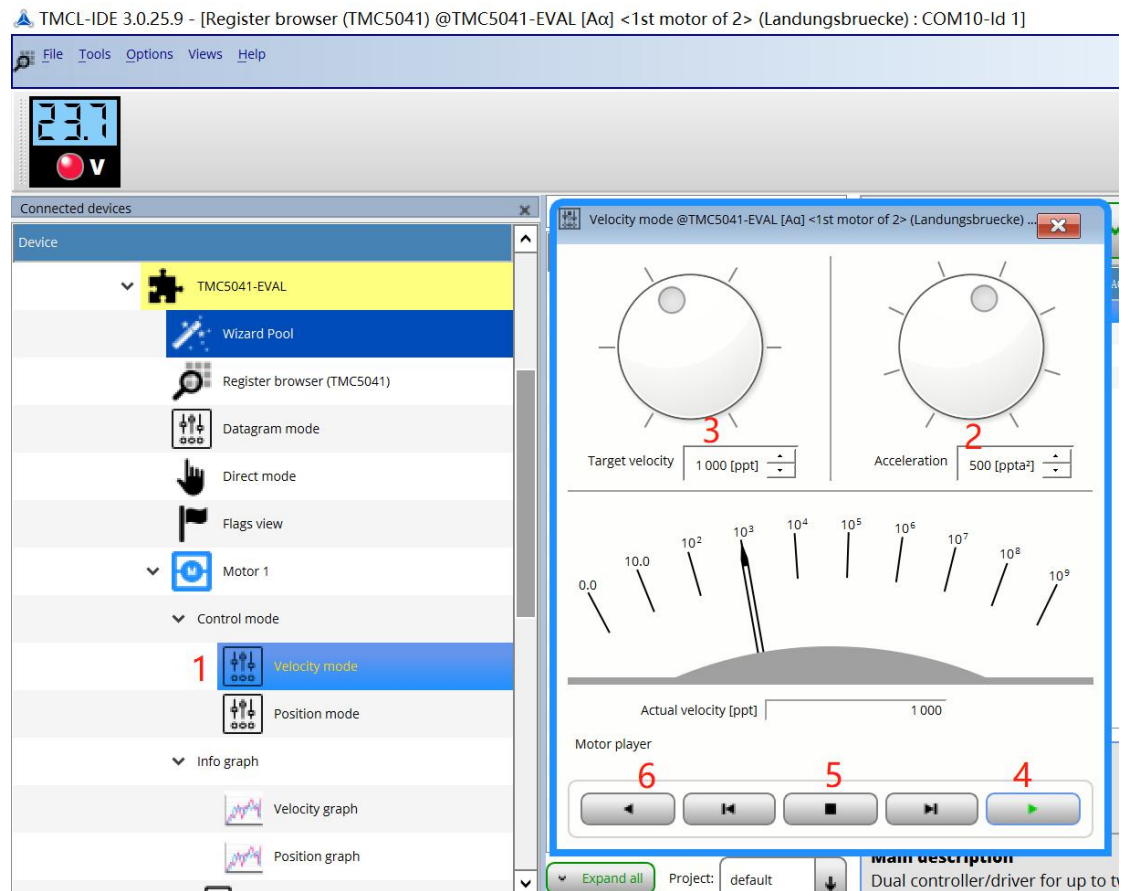
设置电流的寄存器 **0x30**: 按照下图 1-2-3 完成电流设置。



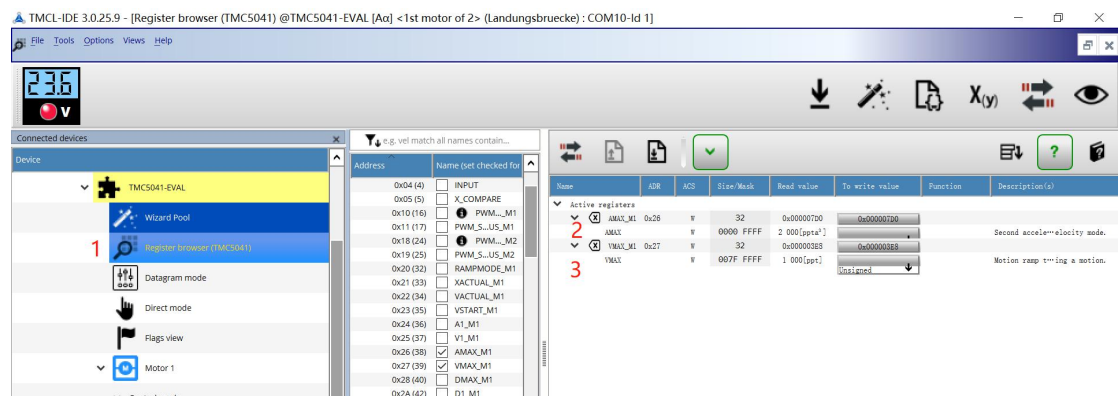
以上二者操作其中一个即可完成配置，建议两者配合方便熟悉寄存器。

### 3. 控制电机运行-速度模式

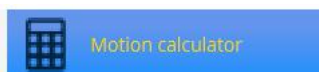
- 在 Control mode 中快速操作，按照以下步骤 1-2-3-4-5，设置最大加速度和最大速度，方向。



- 如下为在寄存器中配置速度模式  
配置加速度寄存器 **0x26 :AMAX** ;最大速度寄存器 **0x27: Vmax**

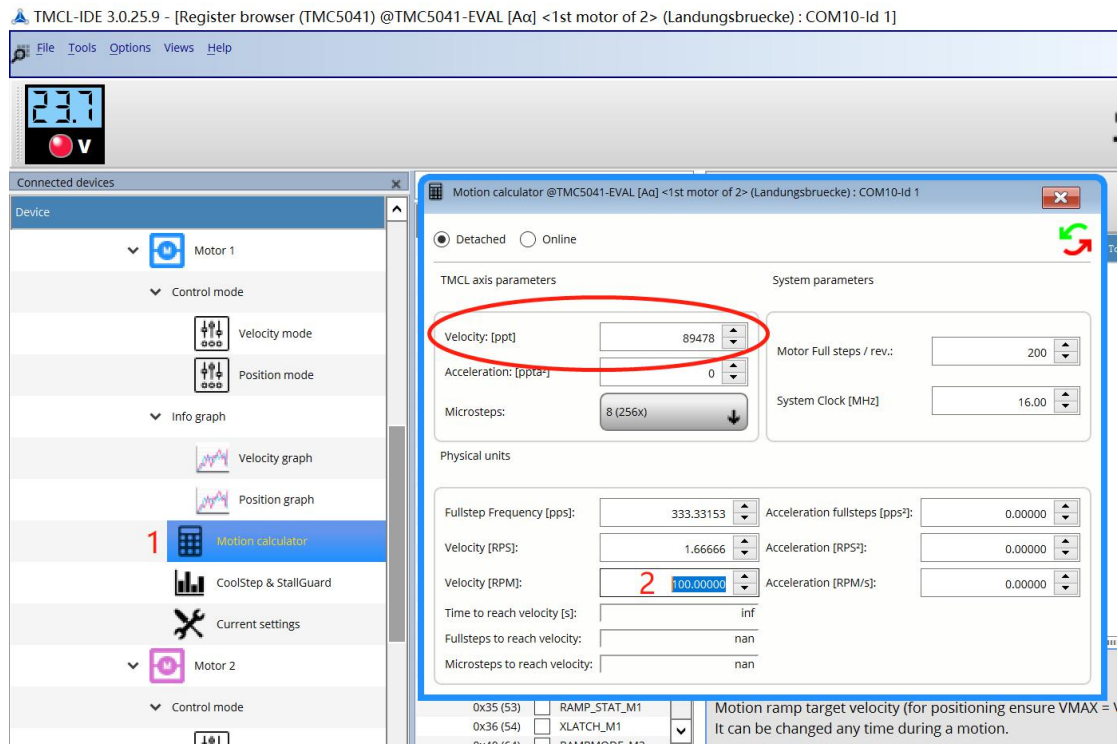


- 换算实际速度快速方法  
单击

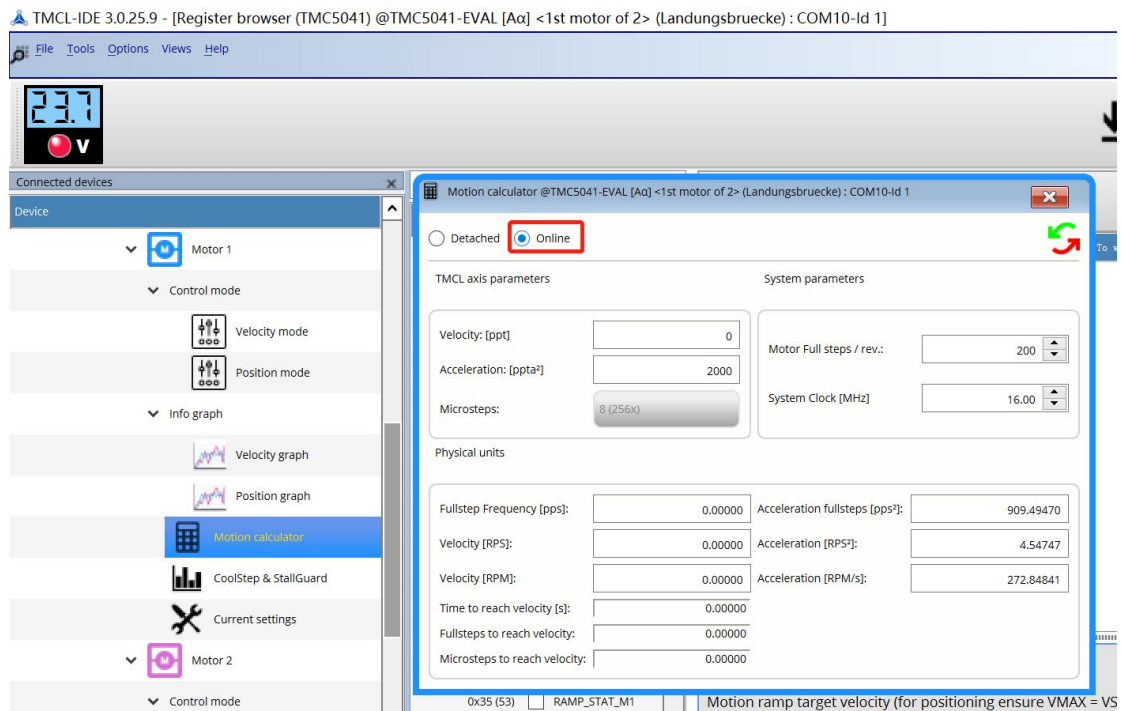


## 选择细分

可以从上面的 TMCL Axis Parameters 的配置推导到下面的实际速度 Physical units 也可以在 Physic unitss 设置实际目标速度推到上上面的速度和加速度配置，然后将推导出来的设置设到最大速度或加速度寄存器里面



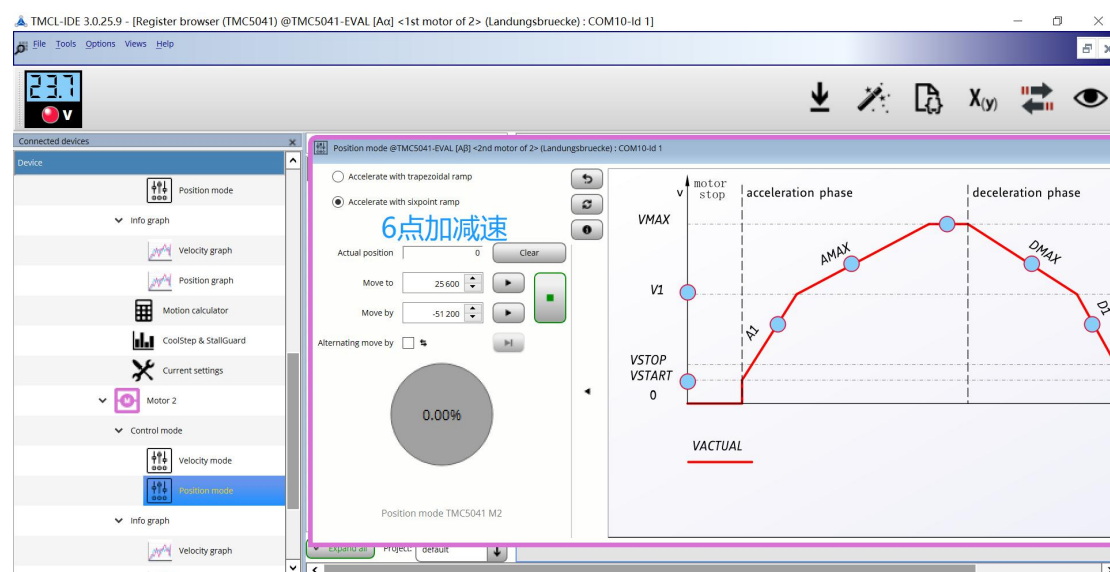
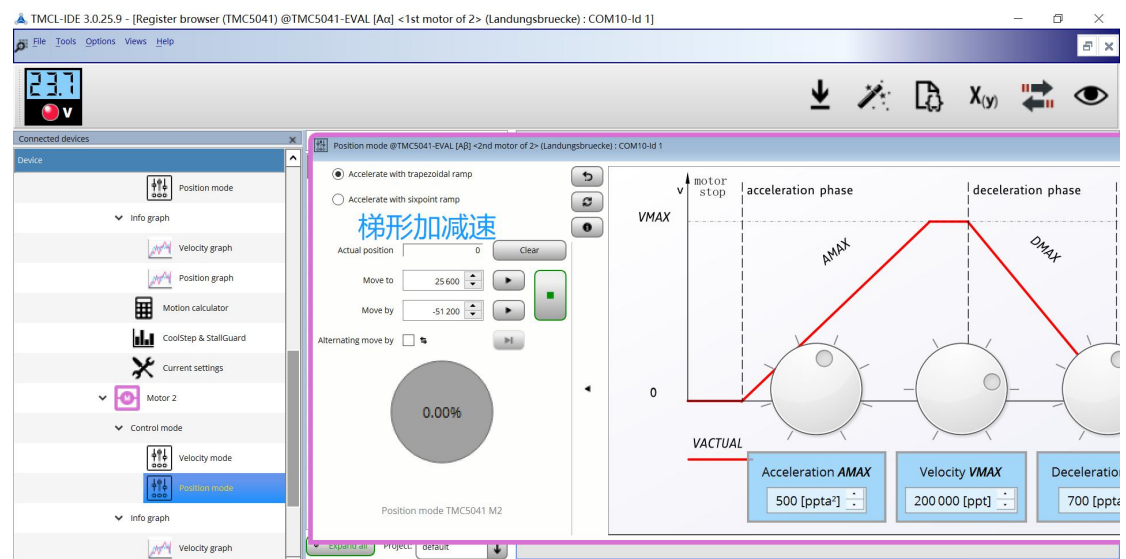
当选择了 Online 之后，会在电机运行过程中实时显示速度





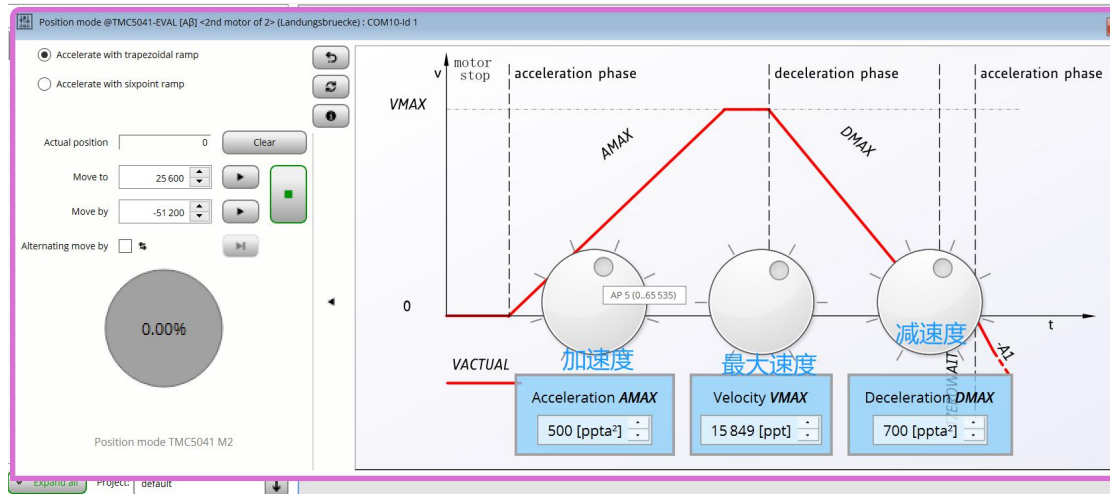
#### 4. 控制电机运行-位置模式

TRINAMIC 的带有运动控制的芯片如 TMC5041/5072/5130/5160/5161.....在位置模式中有 2 种轨迹曲线 梯形加减速和 Six Point 六点加减速



### - 快速配置梯形加减速:

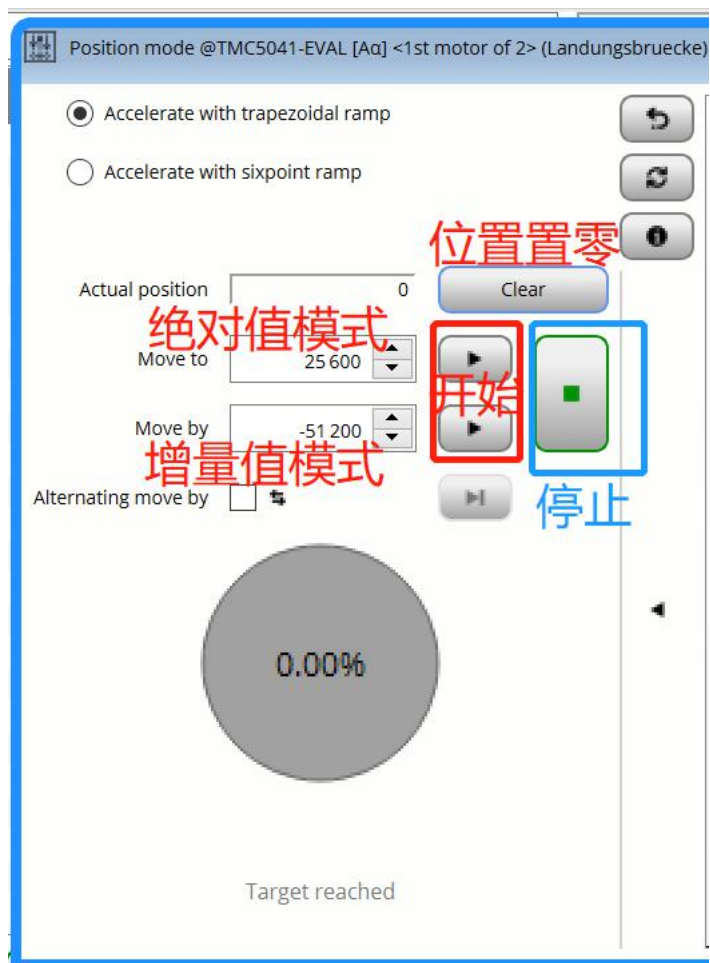
通过手动选择最大速度，加速度和减速度，如下



通过左侧设置目标位置

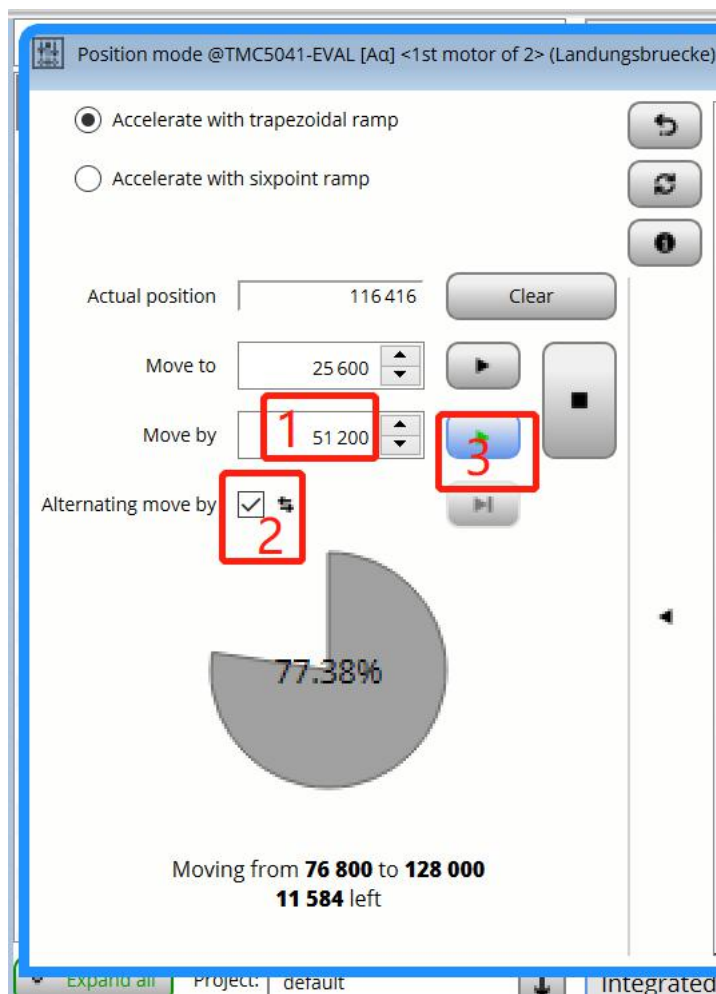
**Move to:** 绝对值位置模式 后面的数值相对零点来运行

**Move By:** 增量位置模式，后面的所在位置的起点相对于上一个位置结束点来说





如果选择以下 1-2-3 配置后电机来回重复运行。



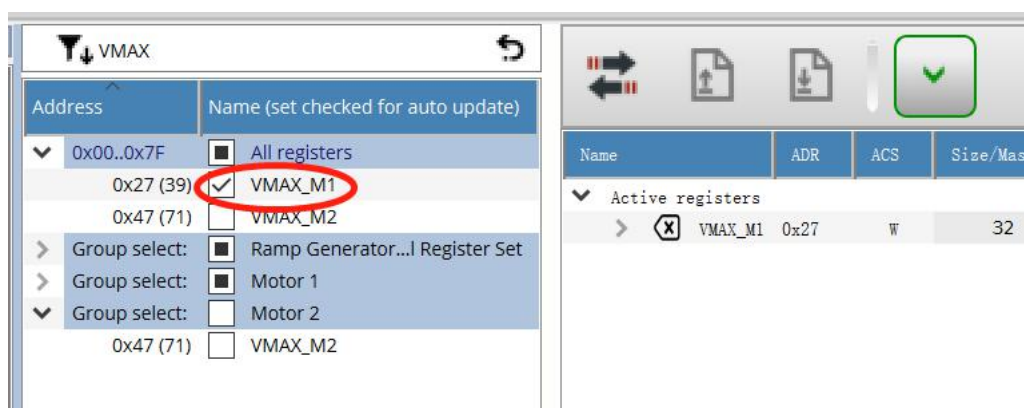
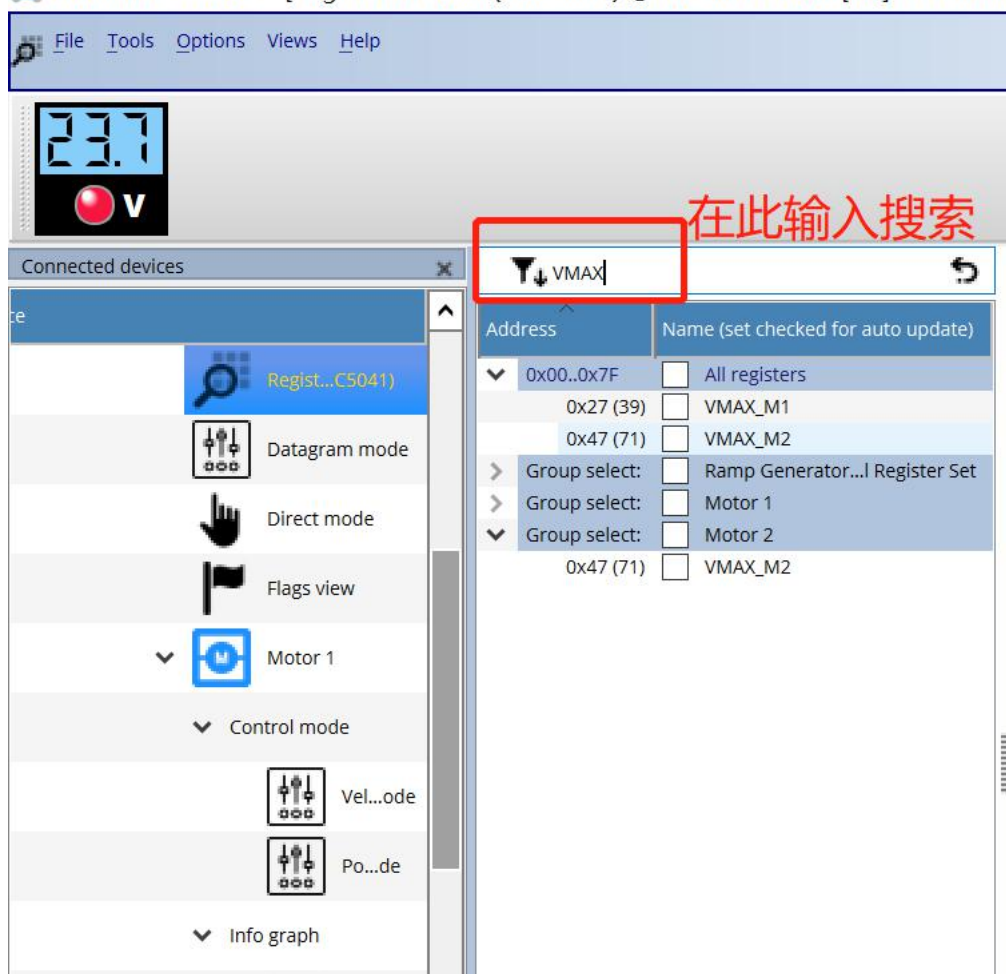
- 在寄存器中配置梯形加减速：

在寄存中找到最大速度 Vmax,加速度 AMAX 和 DMAX

VMAX:0x27; 加速度 AMAX:0x26;减速度 DMAX:0x28

可以在下图中快速搜索，将搜索的想要配置的寄存器前面选择上✓

TMCL-IDE 3.0.25.9 - [Register browser (TMC5041) @TMC5041-EVAL [Aα] &lt;1st moto



VMAX:0x27; 加速度 AMAX:0x26;减速度 DMAX:0x28

Name	ADR	ACS	Size/Mask	Read value	To write value	Function	Description(s)
Active registers							
AMAX_M1	0x26	W	32	0x000007D0	0x000007D0		
AMAX		W	0000 FFFF	2 000[pppt <sup>2</sup> ]			Second acceleration between V1 and V2 value for velocity mode.
VMAX_M1	0x27	W	32	0x000007D0	0x000007D0		
VMAX		W	007F FFFF	2 000[pppt]			Motion ramp target velocity for any time during a motion.
DMAX_M1	0x28	W	32	0x00001388	0x00001388		
DMAX		W	0000 FFFF	5 000[pppt <sup>2</sup> ]			Deceleration between VMAX and V1 (unsigned)

设置完上述的速度，加减速之后，在目标位置寄存器设置目标位置之后电机就会按照上述的速度和加减速运行到目标位置，

目标位置寄存器是 XACTUAL 0x21

Name	ADR	ACS	Size/Mask	Read value	To write value	Function	Description(s)
Active registers							
XACTUAL_M1	0x21	RW	32	0x00006B9D	0x00004E20		
XACTUAL		RW	FFFF FFFF	27.549			Actual motor position
AMAX_M1	0x26	W	32	0x000007D0	0x000007D0		
AMAX		W	0000 FFFF	2 000[pppt <sup>2</sup> ]			Second acceleration between V1 and V2 value for velocity mode.
VMAX_M1	0x27	W	32	0x000007D0	0x000007D0		
VMAX		W	007F FFFF	2 000[pppt]			Motion ramp target velocity for any time during a motion.
DMAX_M1	0x28	W	32	0x00001388	0x00001388		
DMAX		W	0000 FFFF	5 000[pppt <sup>2</sup> ]			Deceleration between VMAX and V1 (unsigned)

XACTUAL

Hex: 0x00004E20

Dec: 20 000

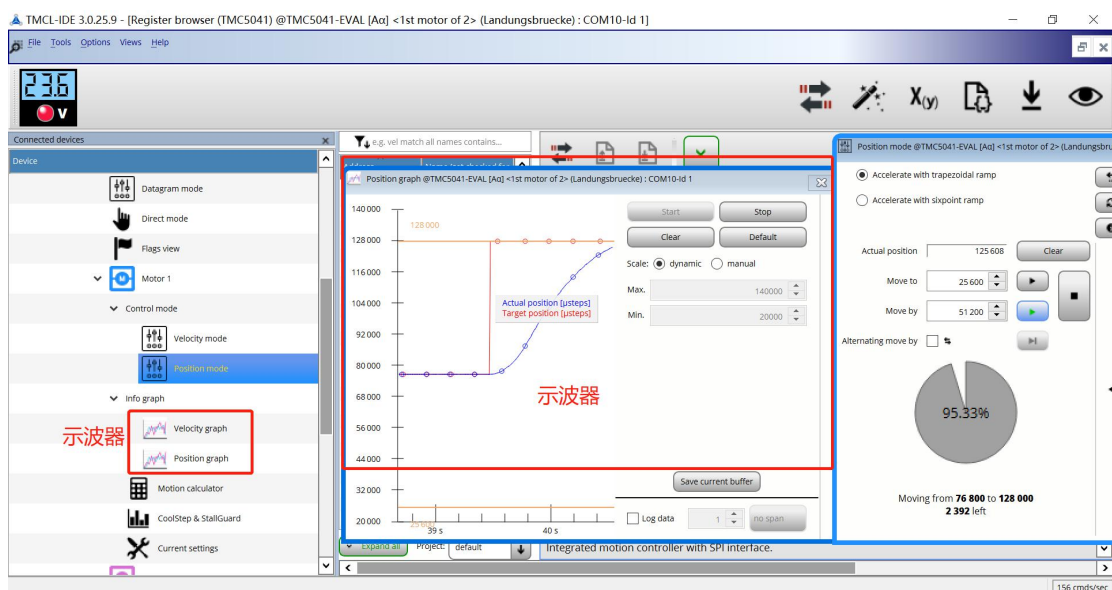
Unsigned

0000 0000 0000 0000 0100 1110 0010 0000

以上介绍了梯形加减速的位置控制的 2 种配置模式，对于 Six Point 加减速曲线模式方法一样，在这里就不多叙述。

## 5. 示波器功能:

在左侧的 Info graph 中打开示波器功能，可以实时显示电机的速度或者位置，而且可以将数值导出。

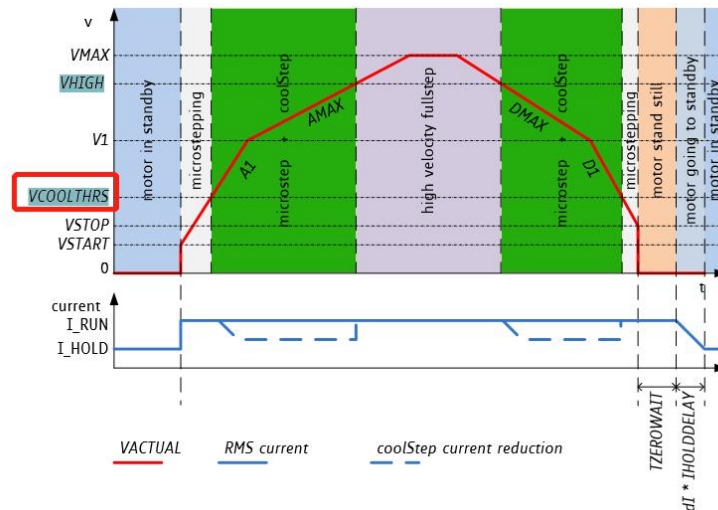


## 6. 不同斩波模式的切换

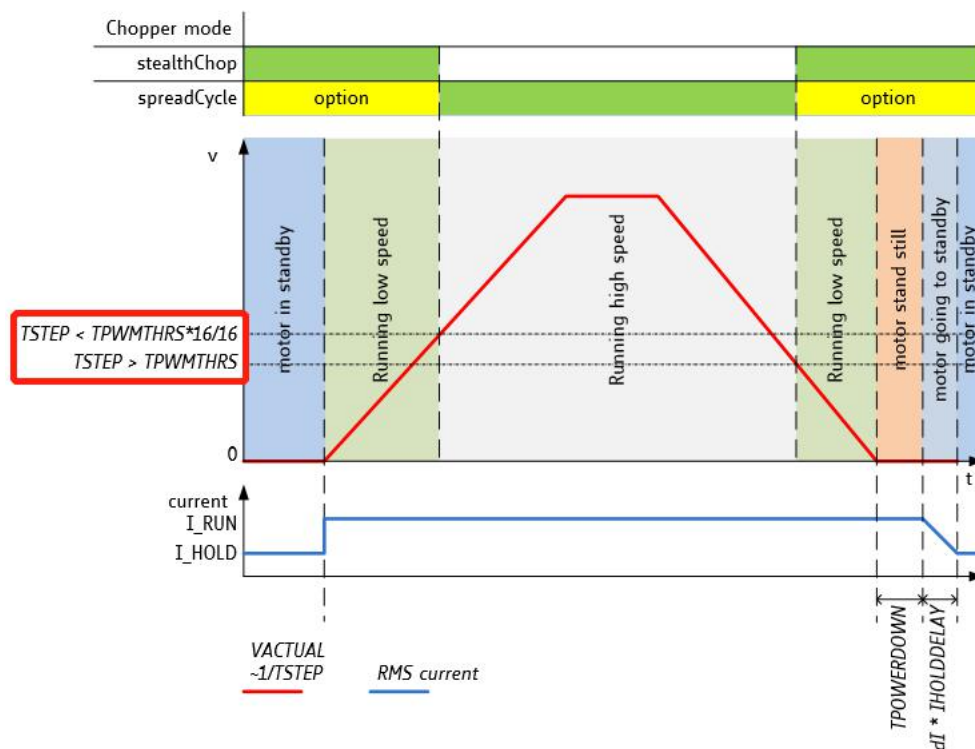
TMC 的芯片会有三种斩波模式低速 Stealthchop, 中速 Spreadcycle(适应 Coolstep) 和高速模式 (High speed 整步模式), 模式的切换是通过配置速度自动完成, 不同芯片会有差异

(1) 如 TMC5041/TMC5072/TMC5130 是通过配置速度参数 **VCOOLTHRS** 和 **VHIGH** 实际速度  $V_{ACT} \leq V_{COOLTHRS}$  时 Stealthchop 才能在配置之后起作用  
当  $V_{HIGH} \geq |V_{ACT}| \geq V_{COOLTHRS}$  时, Spreadcycle 起作用, 只有在这个范围内才可以启动 Coolstep, Stallguard(注第四代的 Stallguard 和 Coolstep 不受这些限制, 可以和 Stealthchop 同时使用比如 TMC2209 等)  
当实际速度  $V_{ACT} > V_{HIGH}$  时 电机工作在高速模式, 此时是整步模式运行。

Name	ADR	ACS	Size/Mask	Read value	To write value	Function	De
Active registers							
▼ VCOOLTHRS_M1	0x31	W	32	0x00000000	0x00000000		
VCOOLTHRS		W	007F FFFF	0 [spt]	Unsigned		



- (3) 如 TMC5160/5161/2208/2209 是通过配置时间参数 **TPWMTHRS** 和 **THIGH** **TPWMTHRS** 用来控制 Stealthchop 模式的最高速度阈值  
**THIGH** 用来控制 Stallguard, coolstep 模式的最高速度阈值和高速 High speed 的开始阈值



将电机以速度模式运行到所希望的转速此时读去 **TSTEP** 寄存器（两个细分之后微步的时间值）

当  $TSTEP \geq TPWMTHRS$  如果 Stealthchop 模式被配置之后就会起作用

当  $TCOOLTHRS \geq TSTEP \geq THIGH$  CoolStep 起作用，如果被配置之后；Stealthchop 不起作用

当  $TCOOLTHRS \geq TSTEP$  当 Stallguard 被配置后在过载时电机自动停止，并且有信号输出

当  $TSTEP \leq THIGH$  时 CoolStep 和 Stealthchop 不起作用，如果 **vhighS** 被配置电机工作会在整步模式

Parameter	Description	Setting	Comment
<i>stst</i>	This flag indicates motor stand still in each operation mode. This occurs $2^{*}20$ clocks after the last step pulse.	0/1	Status bit, read only
<i>TPOWER DOWN</i>	This is the delay time after stand still ( <i>stst</i> ) of the motor to motor current power down. Time range is about 0 to 4 seconds.	0...255	Time in multiples of $2^{*}18 t_{CLK}$
<i>TSTEP</i>	Actual measured time between two $1/256$ microsteps derived from the step input frequency in units of $1/f_{CLK}$ . Measured value is $(2^{*}20)-1$ in case of overflow or stand still.	0...1048575	Status register, read only. Actual measured step time in multiple of $t_{CLK}$
<i>TPWMTHRS</i>	$TSTEP \geq TPWMTHRS$ <ul style="list-style-type: none"> <li>- stealthChop PWM mode is enabled, if configured</li> <li>- dcStep is disabled</li> </ul>	0...1048575	Setting to control the upper velocity threshold for operation in stealthChop
<i>TCOOLTHRS</i>	$TCOOLTHRS \geq TSTEP \geq THIGH$ : <ul style="list-style-type: none"> <li>- coolStep is enabled, if configured</li> <li>- stealthChop voltage PWM mode is disabled</li> </ul> $TCOOLTHRS \geq TSTEP$ <ul style="list-style-type: none"> <li>- Stop on stall and stall output signal is enabled, if configured</li> </ul>	0...1048575	Setting to control the lower velocity threshold for operation with coolStep and stallGuard
<i>THIGH</i>	$TSTEP \leq THIGH$ : <ul style="list-style-type: none"> <li>- coolStep is disabled (motor runs with normal current scale)</li> <li>- stealthChop voltage PWM mode is disabled</li> <li>- If <i>vhighchm</i> is set, the chopper switches to <i>chm=1</i> with <i>TFD=0</i> (constant off time with slow decay, only).</li> <li>- If <i>vhighfs</i> is set, the motor operates in fullstep mode and the stall detection becomes switched over to dcStep stall detection.</li> </ul>	0...1048575	Setting to control the upper threshold for operation with coolStep and stallGuard as well as optional high velocity step mode

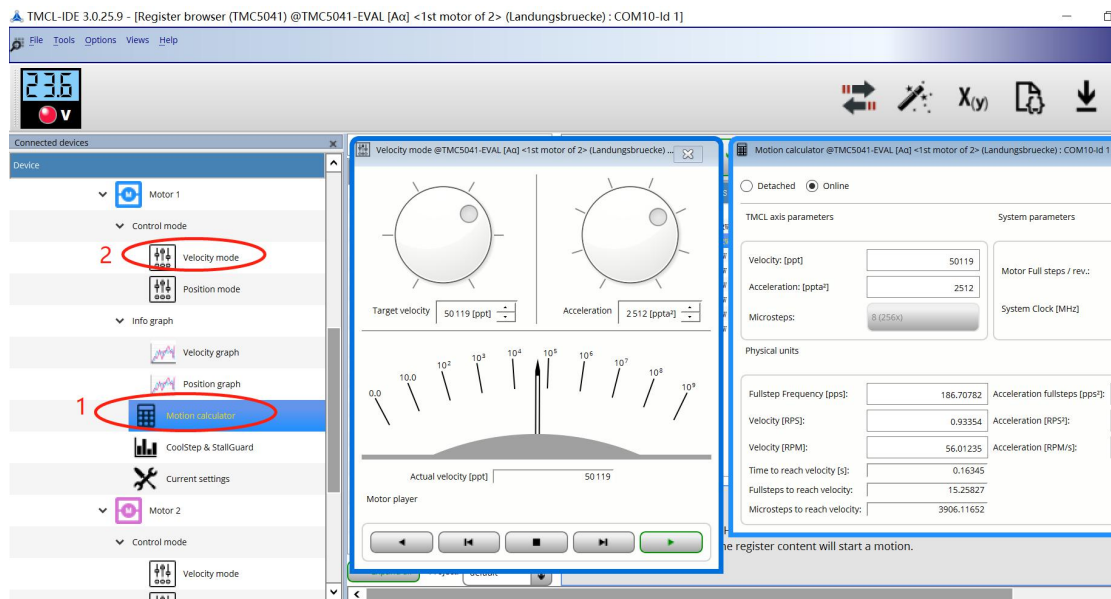


7. **Stallguard** 无传感器负载检测功能，由于 **Stallguard** 是利用电机运行起来之后的反电势推算外部负载，因此 **Stallguard** 不适应电机的速度太低或者太高情况。

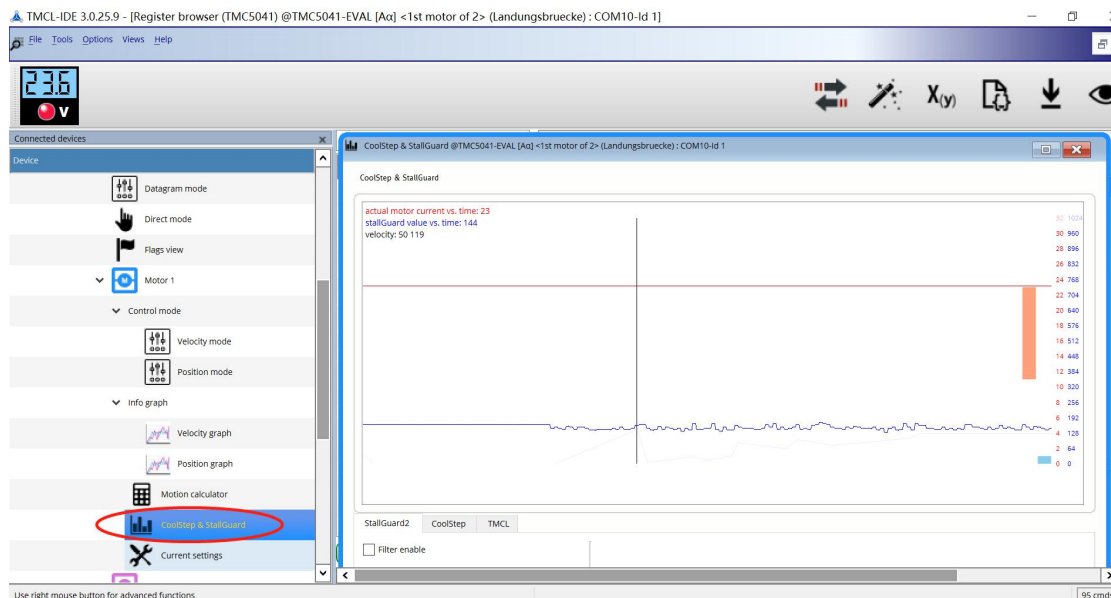
**Stallguard** 的配置步骤以及方法如下：

#### 7-1: 设置目标速度

打开  和  **Velocity mode** 使电机工作在目标的速度模式



#### 7-2: 单击 Coolstep & StallGuard

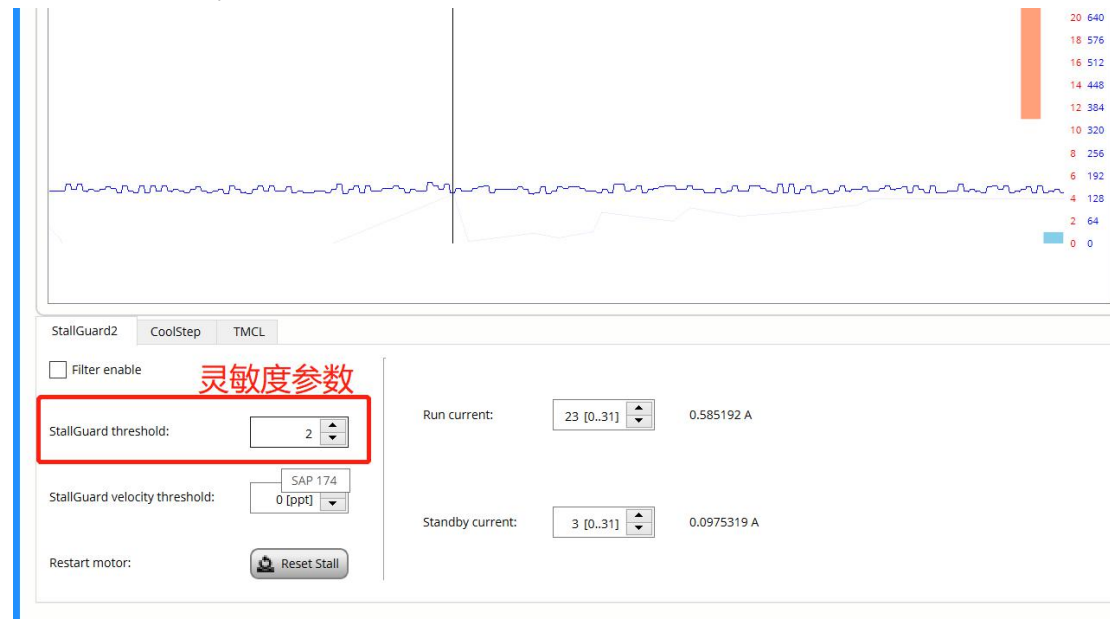


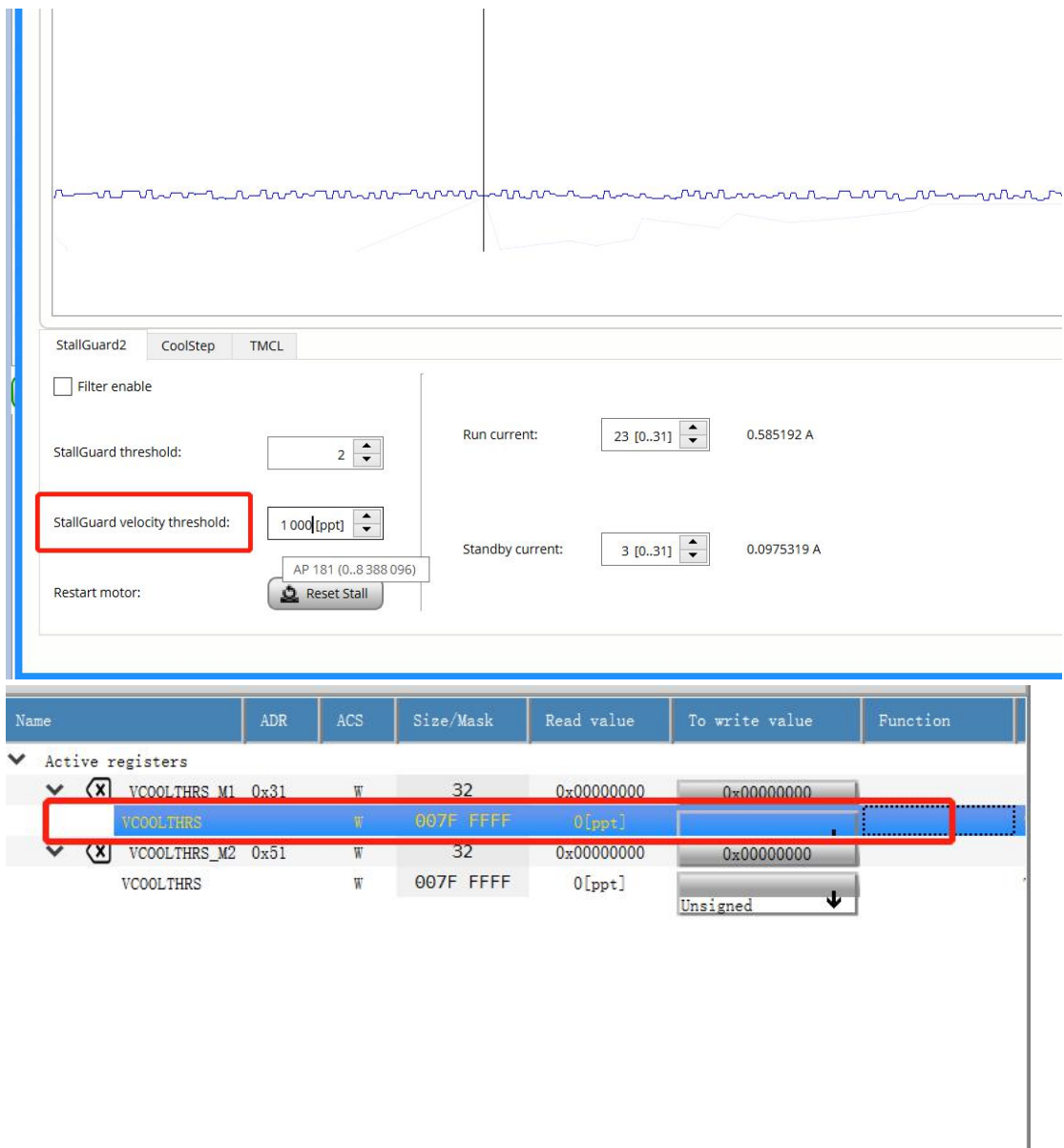
**Stallguard** 的原来的是设置一个对外部力的敏感度 通过 **Stallguard threShold(寄存器 COOLCONF\_M1 06D sgt)** 该数值越小灵敏度越高,该数值越大灵敏度越低 区别就是外部

增大负载之后，Stallguard Value 数值变化，注意：Stallguard Value(寄存器 0x6F SG\_RESULT) 的反馈数值和实际外部负载力相反，也就是外部力越大的时候，Stallguard value 数值越接近于 0，Stallguard threshold 数值越小的时候 越灵敏 Stallguard Value 越接近于 0；  
当下面中 Stallguard Velocity threshold 中数值为非 0 的时候，而外部的负载大到使 Stallguard value 等于 0 时候，此时电机自动停止。

Stallguard Velocity threshold 是启动自动停止的速度阈值 当速度超过这个速度的时候，Stallguard Value=0 的时候 电机才会停止，否则电机不会停止。

SatllGuard Velocity Threshold 由 VCOOLTHRS 0x31 配置





				of 218 clocks
				This is the lower threshold velocity for switching on smart energy coolStep and stallGuard feature. Further it is the upper operation velocity for stealthChop. (unsigned)
				Set this parameter to disable coolStep at low speeds, where it cannot work reliably. The stop on stall function (enable with <i>sg_stop</i> when using internal motion controller) becomes enabled when exceeding this velocity. It becomes disabled again once the velocity falls below this threshold. This allows for homing procedures with stallGuard by blanking out the stallGuard signal at low velocities (will not work in combination with stealthChop).
W	0x31 0x51	23	VCOOLTHRS	$VHIGH \geq  VACT  \geq VCOOLTHRS$ : <ul style="list-style-type: none"><li>- coolStep and stop on stall are enabled, if configured</li><li>- Voltage PWM mode stealthChop is switched off, if configured</li></ul> (Only bits 22..8 are used for value and for comparison)

详情参考各个芯片的 dataSheet

Parameter	Description	Setting	Comment
SGT	This signed value controls the stallGuard2 threshold level for stall detection and sets the optimum measurement range for readout. A lower value gives a higher sensitivity. Zero is the starting value working with most motors. A higher value makes stallGuard2 less sensitive and requires more torque to indicate a stall.	0	indifferent value
		+1... +63	less sensitivity
		-1... -64	higher sensitivity
sfilt	Enables the stallGuard2 filter for more precision of the measurement. If set, reduces the measurement frequency to one measurement per electrical period of the motor (4 fullsteps).	0	standard mode
		1	filtered mode
Status word	Description	Range	Comment
SG	This is the <i>stallGuard2 result</i> . A higher reading indicates less mechanical load. A lower reading indicates a higher load and thus a higher load angle. Tune the SGT setting to show a SG reading of roughly 0 to 100 at maximum load before motor stall.	0... 1023	0: highest load low value: high load high value: less load

*In order to use stallGuard2 and coolStep, the stallGuard2 sensitivity should first be tuned using the SGT setting!*

#### 6.2.2.1 SW\_MODE – Reference Switch & stallGuard2 Event Configuration Register

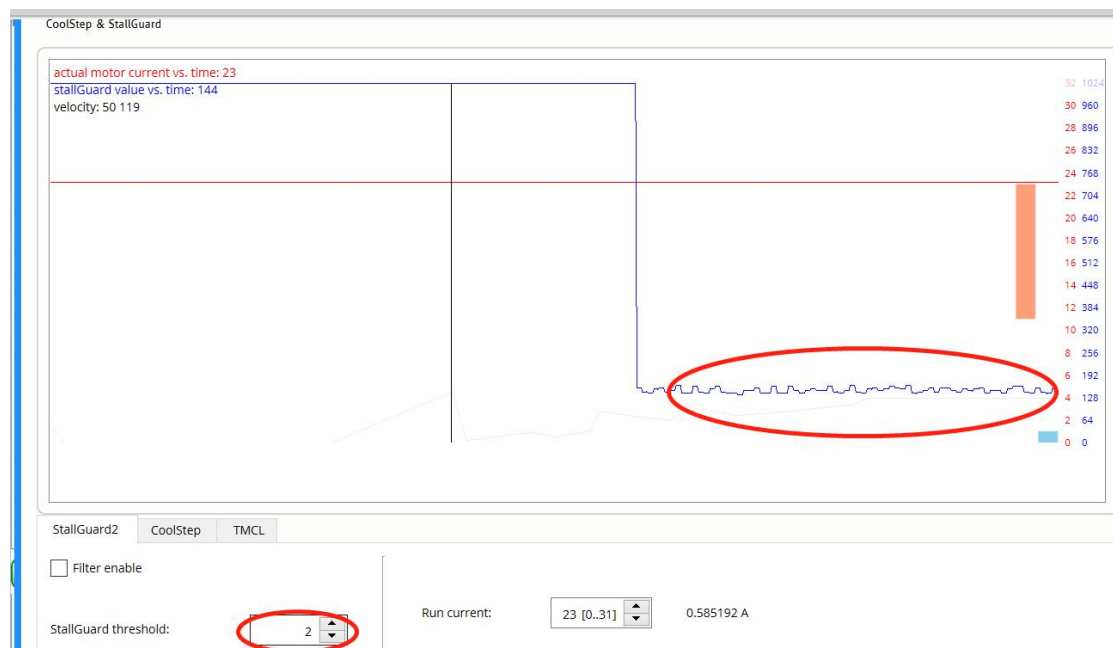
0x34, 0x54: SW_MODE – REFERENCE SWITCH AND STALLGUARD2 EVENT CONFIGURATION REGISTER		
Bit	Name	Comment
11	en_softstop	<p>0: Hard stop 1: Soft stop</p> <p>The soft stop mode always uses the deceleration ramp settings DMAX, V1, D1, VSTOP and TZEROWAIT for stopping the motor. A stop occurs when the velocity sign matches the reference switch position (REFL for negative velocities, REFR for positive velocities) and the respective switch stop function is enabled.</p> <p>A hard stop also uses TZEROWAIT before the motor becomes released.</p> <p><i>Attention: Do not use soft stop in combination with stallGuard2.</i></p>
10	sq_stop	<p>1: Enable stop by stallGuard2. Disable to release motor after stop event.</p> <p><i>Attention: Do not enable during motor spin-up, wait until the motor velocity exceeds a certain value, where stallGuard2 delivers a stable result, or set VCOOLTHRS to a suitable value.</i></p>

把速度设置为目标速度之后可以调整灵敏度参数 Stallguard threshold 来配置外部需要多少力才可以让电机停止 也就是使用 Stallguard value=0

下图为同一速度 不同 Stallguard threshold 下的 Stallguard value(蓝色曲线反应外部力情况, 负载越大值越小)



Stallguard threshold=12



上图为 Stallguard threshold=2 此时只要外部施加少许力电机就会停止（因为此时的 Stallguard value 更接近于 0）。

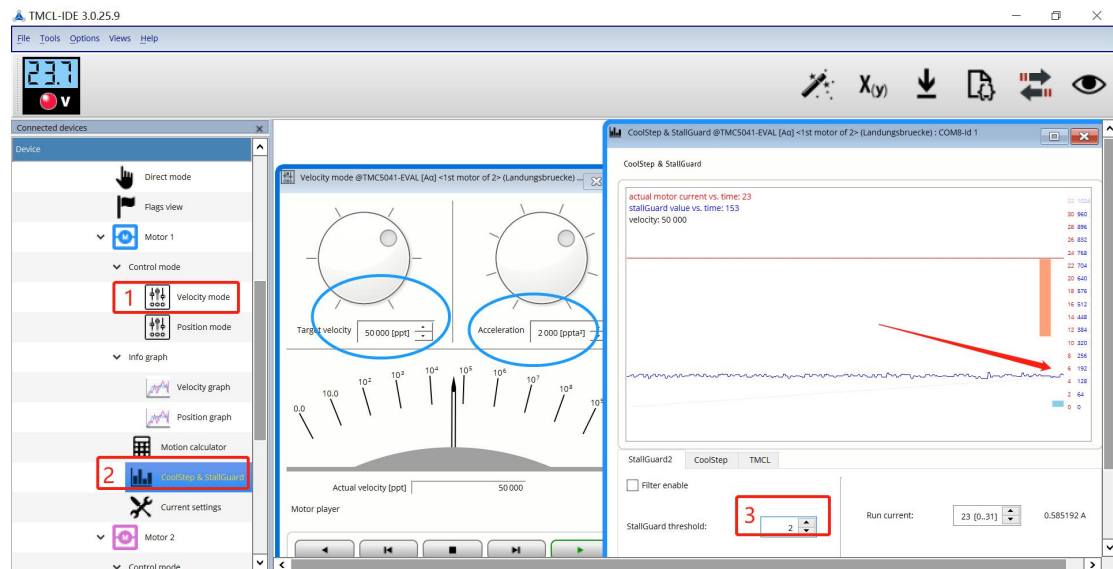
Stallguard 功能可以实现不需要力传感器情况下的力控制，比如压力控制，无传感器回零等。



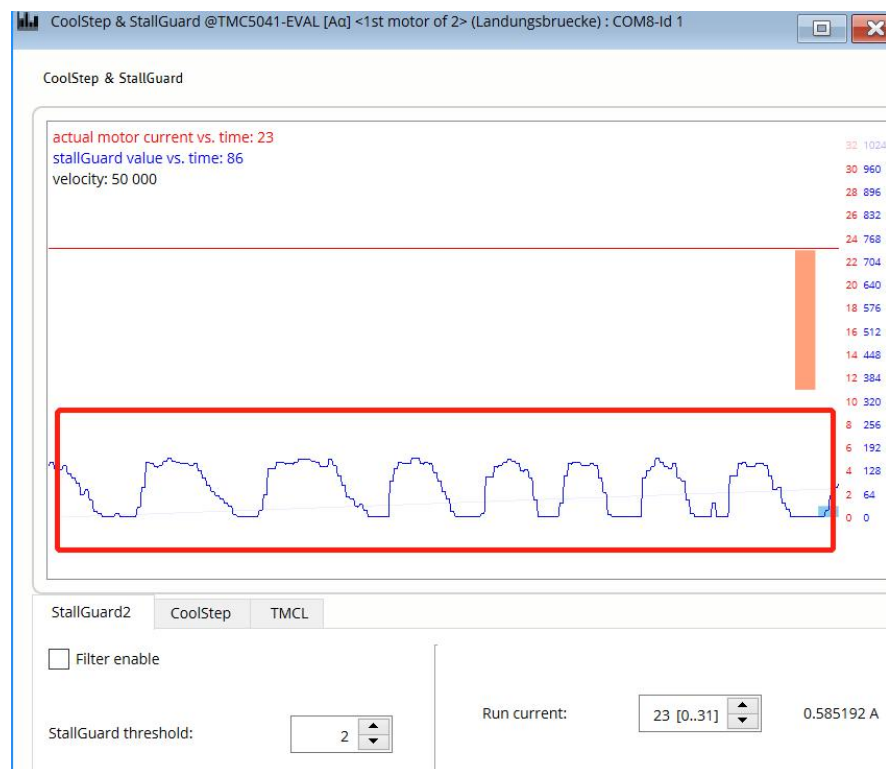
**8. Coolstep 配置** 相比传统的步进电机是恒流驱动，为了不使电机丢步，只能把驱动电流设置的很高，这样会带来功耗比较大，电机容易发热，容易产生剩磁震动等问题。Coolstep 是基于电机的反电动势动态控制输出电流，负载大的时候电流自动增加，负载小的时候电流自动下降，相比传统的恒流驱动 Coolstep 可以节省 80% 的能量。

Coolstep 是在调节好 Stallguard 的基础上来配置的，需要首先配置 SGT Stallguard threShold(寄存器 COOLCONF\_M1 0x6D sgt)

8-1. 通过下图的 1-2-3 设置加速度，速度和灵敏度参数 StallGuard threshold 寄存器 0x6D 下图为 2



保证增大负载时候 Stallguard value(寄存器 0x6F SG\_RESULT)下将，负载减小时 Stallguard Value 上升





## 8-2

切换到 Coolstep 栏



## 8-2-1

CoolStep Velocity threshold 设置 Coolstep 功能起作用的速度阈值，当实际速度高于该数值的时候 Coolstep 不起作用，该寄存器由 0x31 VCOOLTHRS 决定

CoolStep velocity threshold: 0 [ppt]

Name	ADR	ACS	Size/Mask	Read value	To write value	Function
Active registers						
VCoolThrs_M1	0x31	W	32	0x00000000	0x00000000	
VCoolThrs		W	007F FFFF	0 [ppt]		
VCoolThrs_M2	0x51	W	32	0x00000000	0x00000000	
VCoolThrs		W	007F FFFF	0 [ppt]	Unsigned	

例如目前电机速度是 50000 当 Coolstep velocity threshold 大于 50000 时 Coolstep 不起作用

8-2-2 Current minimum 设置电流的最小值（当负载小的时候 电流到达的数值为最大电流的 1/2 或 1/4）

StallGuard2 CoolStep TMCL

Current minimum: 0 - 1/2

Current down step: 3 - very fast

Current up step: 3 - large

Hysteresis width: 1

Hysteresis start: 2

Current Minimum 所对应的寄存器如下 0x6D-seimin

Active registers						
COOLCONF_M1	0x6D	W	32	0x00000000	0x00000000	
semin		W	0000 000F	0		minimum step off
seup		W	0000 0060	id: 0	1	Current value
semax		W	0000 0F00	0		stall energy
sedn		W	0000 6000	id: 0	n=32	Current by one
seimin		W	0000 8000	id: 0		1/2 of setting minimum control
sgt		W	007F 0000	0		stall. enable
sfilt		W	0100 0000	id: 0		

seimin	Hex: 0x0	Dec: 0	Idx: 1/2 of current setting
0	0	1/2 of current setting	
1	1	1/4 of current setting	

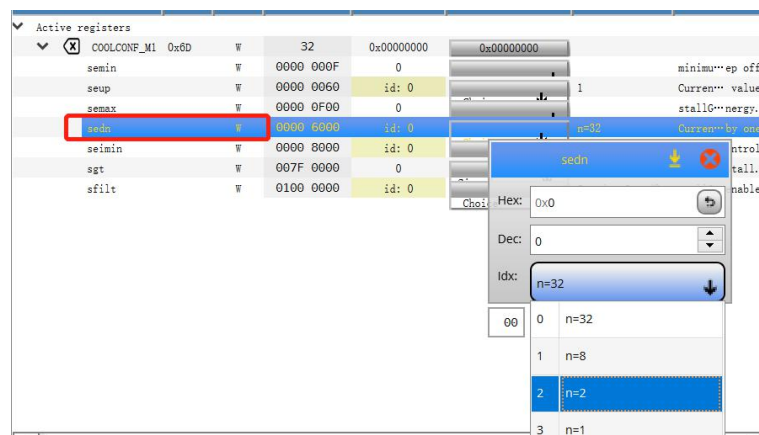
下图为 Current minimum 为 1/2 和 1/4 最小电流的对比



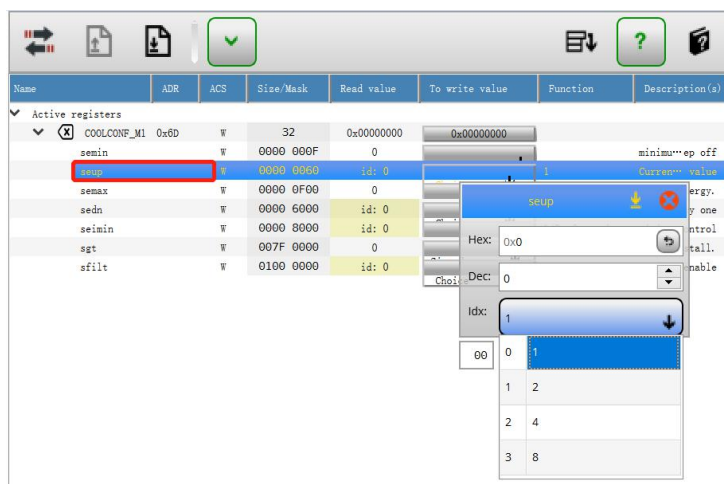
8-2-3: current down step: 电流下降过程的幅度  
Current up step: 电流上升的幅度

The screenshot shows the CoolStep configuration interface. The 'Current minimum' is set to '1 - 1/4'. The 'Current down step' is set to '3 - very fast' and the 'Current up step' is set to '3 - large'. The 'Hysteresis width' is set to '1' and the 'Hysteresis start' is set to '2'.

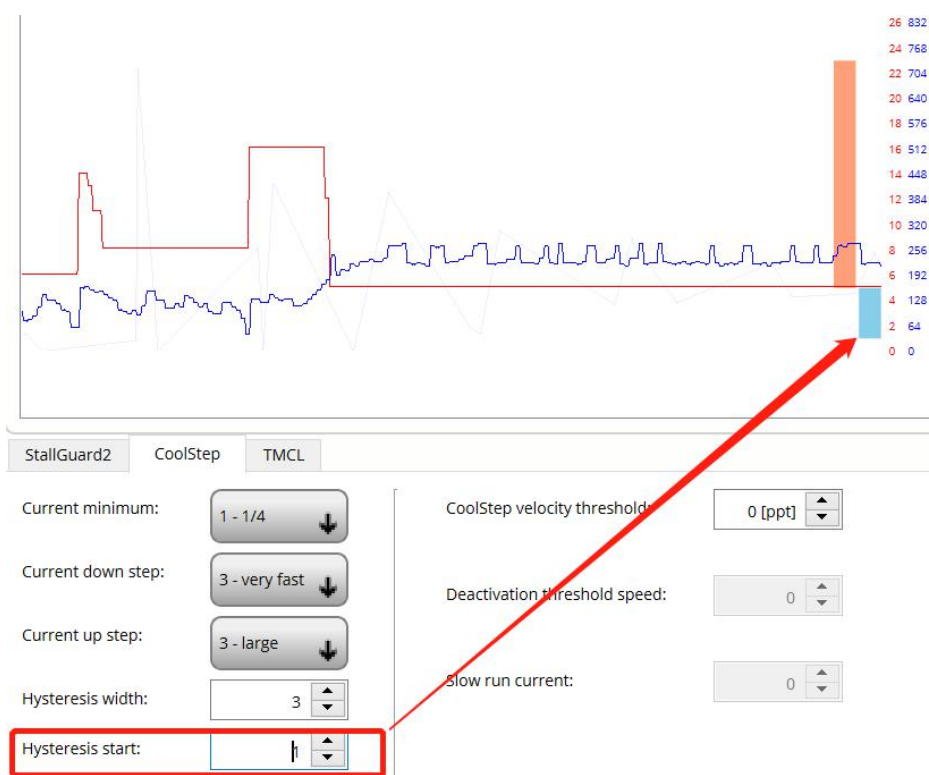
Current down step 对应的寄存器为 COOLCONF\_M1 - 0x6D-sedn 如下图



Current up step 对应的寄存器为 COOLCONF\_M1 - 0x6D-seup 如下图



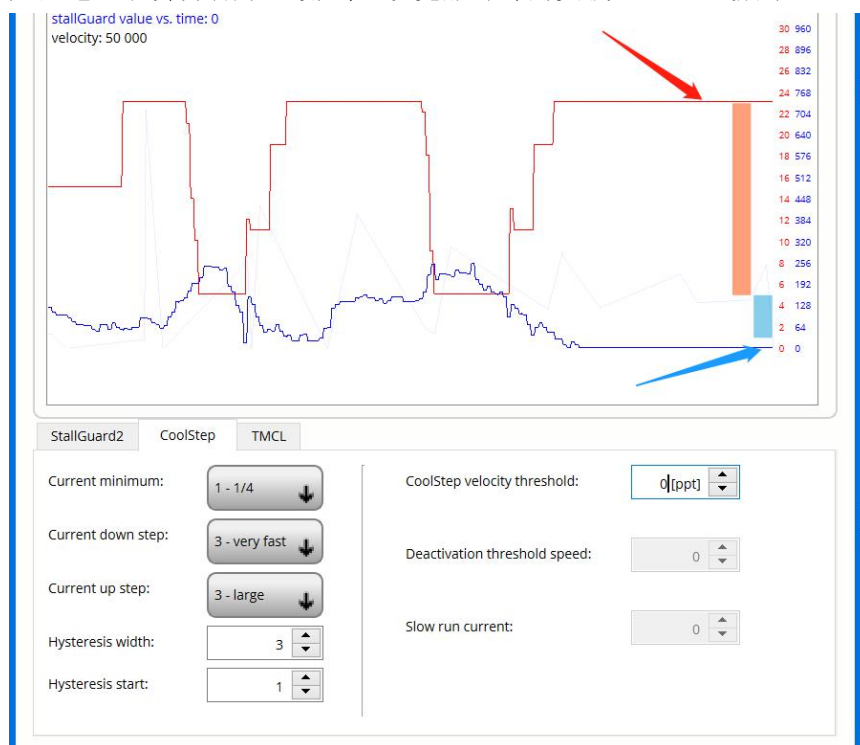
8-2-4: Hysteresis Start: 当 Stallguard Value(寄存器 0x6F SG\_RESULT) 下图蓝色曲线低于该数值的时候电流增加至最大电流。



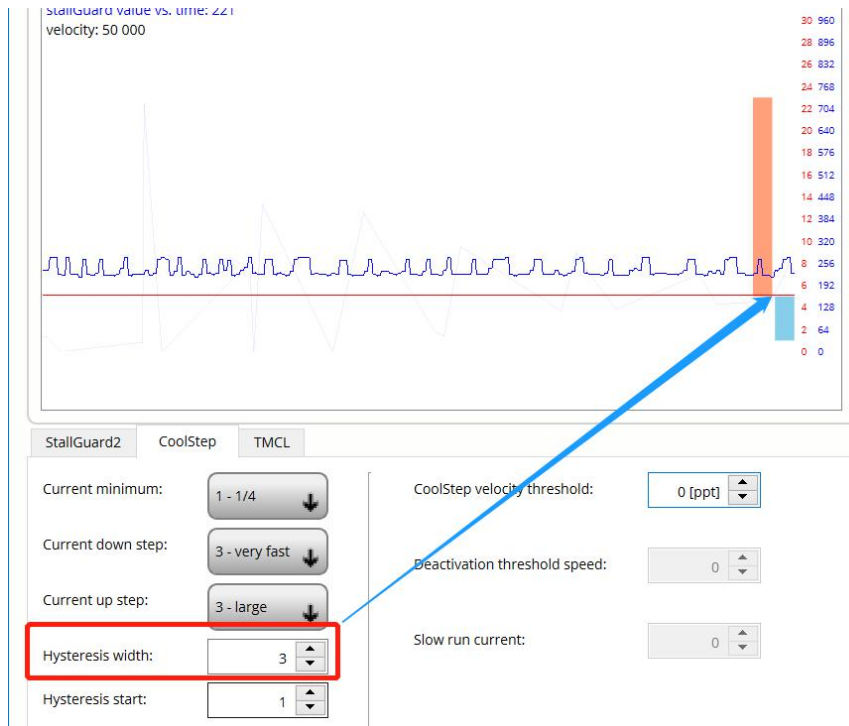
Hysteresis Start 对应的寄存器为 COOLCONF\_M1 - 0x6D-semin 如下图

Name	ADR	ACS	Size/Mask	Read value	To write value	Function	Description
Active registers							
COOLCONF M1	0x6D	W	32	0x00000000	0x00000001		
semin		W	0000 000F	0			minimum speed of
seup		W	0000 0060	id: 0			valu
semax		W	0000 0F00	0			ergy
sedn		W	0000 6000	id: 0			y on
seimin		W	0000 8000	id: 0			ontro
sgt		W	007F 0000	0			stall
sfilt		W	0100 0000	id: 0			enabl

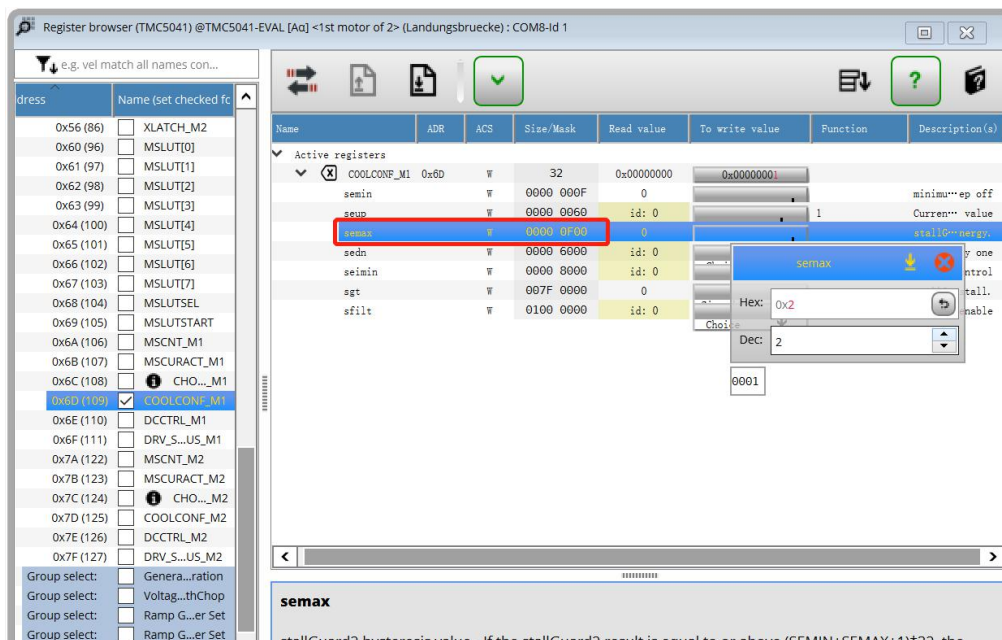
如下图所示当 Stallguard Value 蓝色曲线低于 Hysteresis Start 蓝色柱型底部，表示外部负载很大；电流就会自动升至最大值 以克服外部的负载 红色曲线所示。



Hysteresis width: 当 Stallguard Value (寄存器 0x6F SG\_RESULT) 下图蓝色曲线等于或高于该数值（蓝色柱型顶部）的时候电流会自动下降至最小电流值。

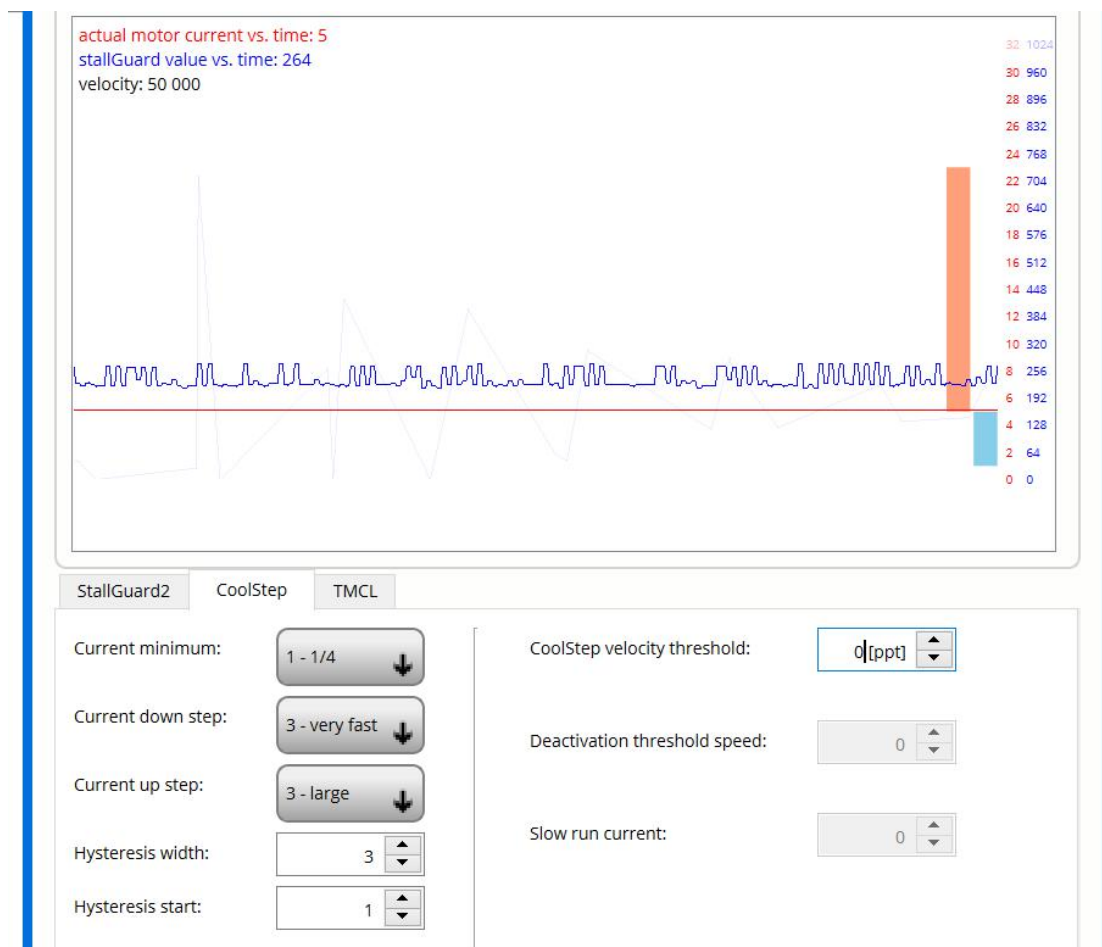


Hysteresis width 对应的寄存器为 **COOLCONF\_M1 - 0x6D-semax** 如下图

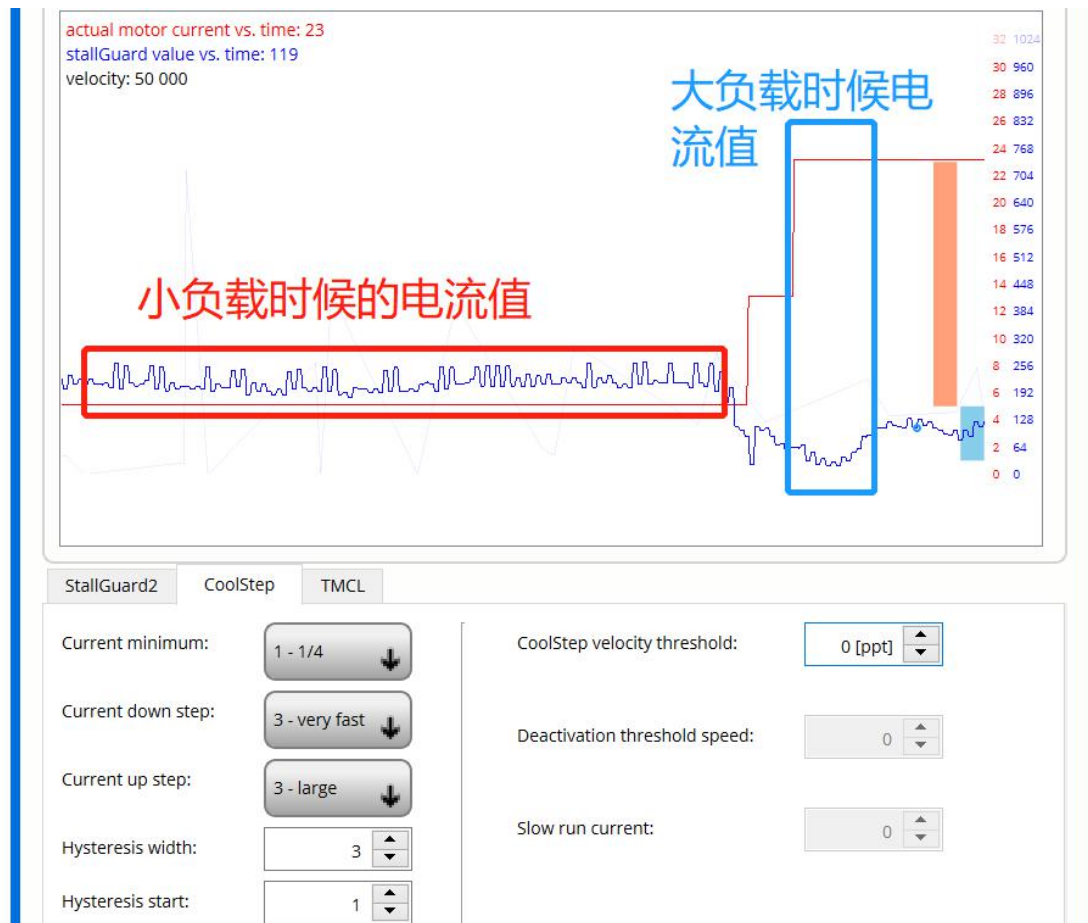


如下图当 StallGuard value 高于或等于 Hysteresis width (蓝色柱状顶部) 表示外部的负载很小, 此时电流会自动下降至最小电流





如下图电流会根据 Stallguard 的数值（反应的是外部负载）自动调节电流，以实现最佳能耗



## 9. Stealthchop 配置

TRINAMIC 的静音斩波 Stealthchop 到目前一共有 2 代

早期的 TMC5130/2100/2130/5041/5072 是第一代 Stealthchop

之后的 TMC5160/5161/2208/2209/2160/2225 是第二代的，区别是第二代的 Stealthchop 可以自动配置 AUTOTUNING

### (1) 第一代 Stealthchop

Stealthchop 是电压的斩波模式 可实现低速静音，平稳运行。高速时候切换到 Spreadcycle 模式的切换是通过不同的速度来实现，VCOOLTHRS(0x31)寄存器用来选择模式

该参数是使能 Coolstep 的最低速度值，是使能 Stealthchop 的最高速度值

如果 Coolstep 配置的话当  $V_{HIGH} \geq |V_{ACT} \text{ 实际速度}| \geq V_{COOLTHR}$  时 Coolstep 起作用

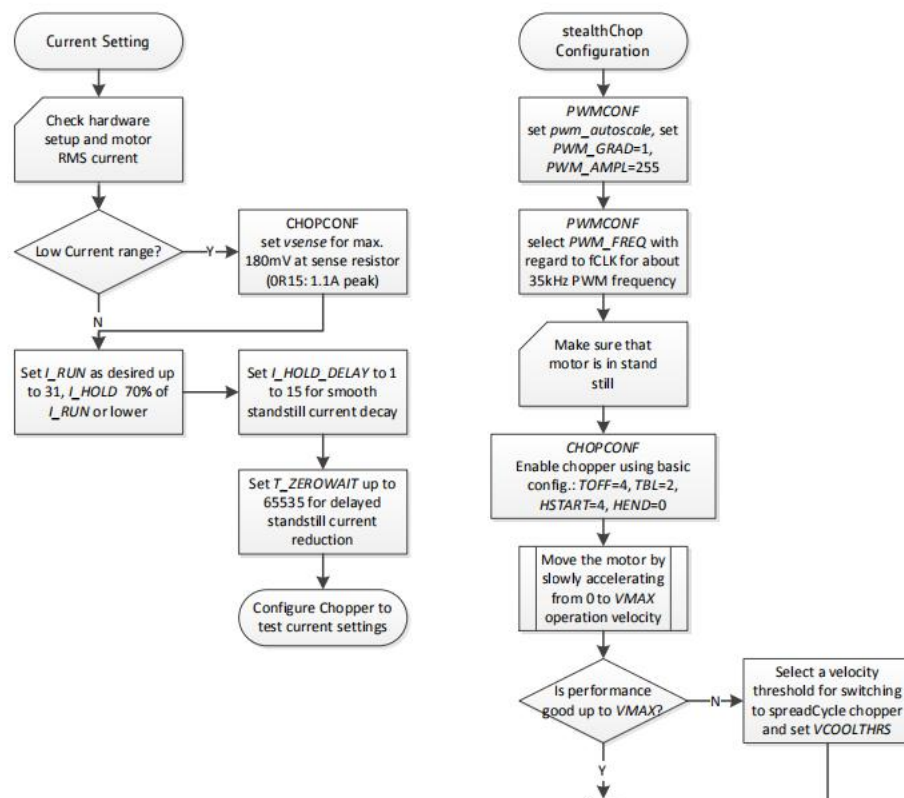
W	0x31 0x51	23	VCOOLTHRS	<p>This is the lower threshold velocity for switching on smart energy coolStep and stallGuard feature. Further it is the upper operation velocity for stealthChop. (unsigned)</p> <p>Set this parameter to disable coolStep at low speeds, where it cannot work reliably. The stop on stall function (enable with <i>sg_stop</i> when using internal motion controller) becomes enabled when exceeding this velocity. It becomes disabled again once the velocity falls below this threshold. This allows for homing procedures with stallGuard by blanking out the stallGuard signal at low velocities (will not work in combination with stealthChop).</p> <p><b>VHIGH ≥  VACT  ≥ VCOOLTHRS:</b></p> <ul style="list-style-type: none"> <li>- coolStep and stop on stall are enabled, if configured</li> <li>- Voltage PWM mode stealthChop is switched off, if configured</li> </ul> <p>(Only bits 22..8 are used for value and for comparison)</p>
---	--------------	----	-----------	--

当 Stealthchop 配置时， $|VACT| \leq VCOOLTHR$  时 Stealthchop 起作用  
每款芯片的 datasheet 会包括 Quick Configuration Guide 里面有详细如何逐步配置 Stealthchop 和 Spreadcyc

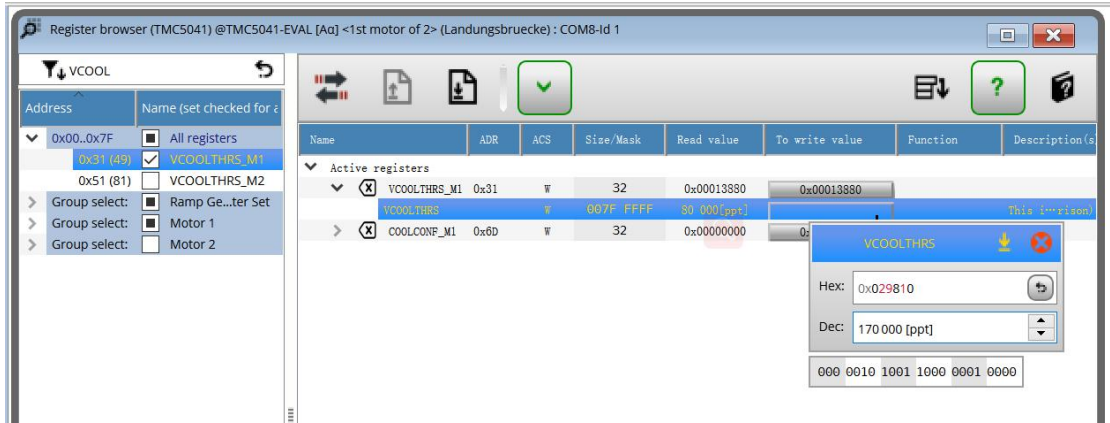
## 14 Quick Configuration Guide

This guide is meant as a practical tool to come to a first configuration and do a minimum set of measurements and decisions for tuning the driver. It does not cover all advanced functionalities, but concentrates on the basic function set to make a motor run smoothly. Once the motor runs, you may decide to explore additional features, e.g. freewheeling and further functionality in more detail. A current probe on one motor coil is a good aid to find the best settings, but it is not a must.

### CURRENT SETTING AND FIRST STEPS WITH STEALTHCHOP



9-1: 将 VCOOLTHRS 0x31 设为 1700000 因此只有实际速度低于 1700000pps Stealthchop 才起作用。



9-2: 配置 PWM\_Autoscale

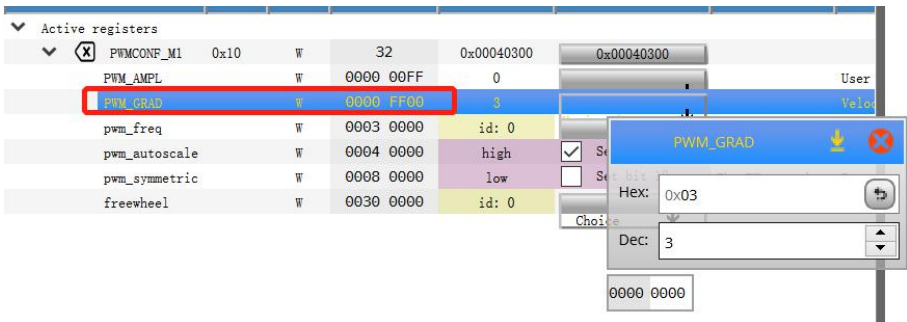
19	-	reserved	Set to 0	
18	<i>pwm_autoscale</i>	PWM automatic amplitude scaling	0	User defined PWM amplitude. The current settings have no influence.
			1	Enable automatic current control <i>Attention: When using a user defined sine wave table, the amplitude of this sine wave table should not be less than 244. Best results are obtained with 247 to 252 as peak values.</i>

将 PWM\_Autoscale 设为 high

Name	ADR	ACS	Size/Mask	Read value	To write value	Function
Active registers						
<input checked="" type="checkbox"/> PWMCONF_M1	0x10	W	32	0x00040000	0x00040000	
PWM_AMPL		W	0000 00FF	0		
PWM_GRAD		W	0000 FF00	0		
pwm_freq		W	0003 0000	id: 0		2/1024 f CLK
pwm_autoscale		W	0004 0000	high	<input checked="" type="checkbox"/> Set bit 18	Enable...alues.
pwm_symmetric		W	0008 0000	low	<input type="checkbox"/> Set bit 19	The PW... cycle
freewheel		W	0030 0000	id: 0		Normal...ration

9-3: 配置 PWM\_GRAD 此处设置为 3

PWM_GRAD	Global enable and regulation loop gradient when <i>pwm_autoscale</i> =1.	0	Do not use stealthChop
		1 ... 15	stealthChop enabled



注意：当 PWM\_GRAD=0 的时候 Stealthchop 不起作用

9-4: 配置 PWM\_AMPL 用于配置 Stealthchop 速度变化时候由于相序跳变时候带来的电流过流

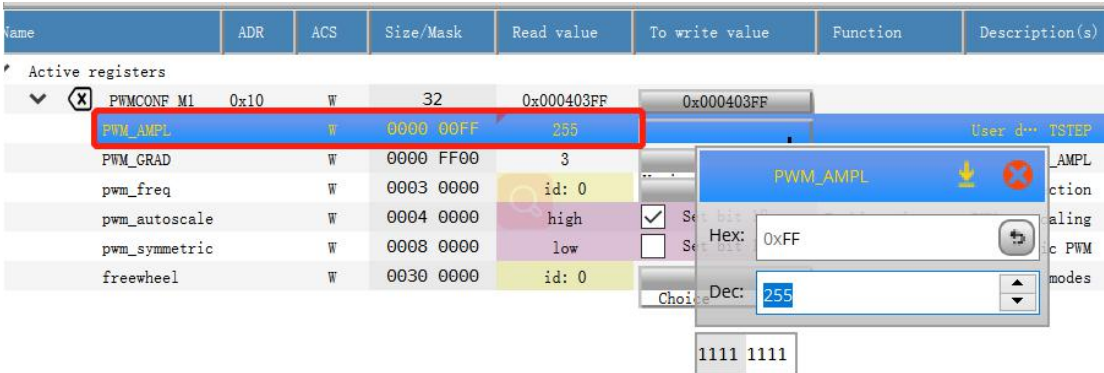
7.2.2 PWM\_AMPL for Using stealthChop and spreadCycle

When combining stealthChop with spreadCycle or constant off time classic PWM, a switching velocity can be chosen using *VCOOLTHRS*. With this, stealthChop is only active at low velocities. Often, a very low velocity in the range of 1 to a few 10 RPM fits best. In case a high switching velocity is chosen, special care should be taken for switching back to stealthChop during deceleration, because the phase jerk can produce a short time overcurrent. (Refer to chapter 7.4 for more details about combining stealthChop with other chopper modes.)

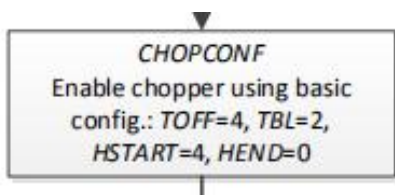
To avoid a short time overcurrent and to minimize the jerk, the initial amplitude for switching back to stealthChop at sinking velocity can be determined using the setting *PWM\_AMPL*. Tune *PWM\_AMPL* to a value which gives a smooth and safe transition back to stealthChop within the application. As a thumb rule,  $\frac{1}{2}$  to  $\frac{3}{4}$  of the last *PWM\_SCALE* value which was valid after the switching event at rising velocity can be used. For high resistive steppers as well as for low transfer velocities (as set by *VCOOLTHRS*), *PWM\_AMPL* can be set to 255 as most universal setting.

**Note**  
The autoscaling function only starts up regulation during motor standstill. After enabling stealthChop and setting all parameters, be sure to wait until *PWM\_SCALE* has reached a stable state before starting a motion. Failure to do so will result in zero motor current!  
In case the automatic scaling regulation is instable at your desired motion velocity, try modifying the chopper frequency divider *PWM\_FREQ*. Also adapt the blank time *TBL* and motor current for best result.

对于高阻值的电机 PWM\_AMPL 可以设置为 255

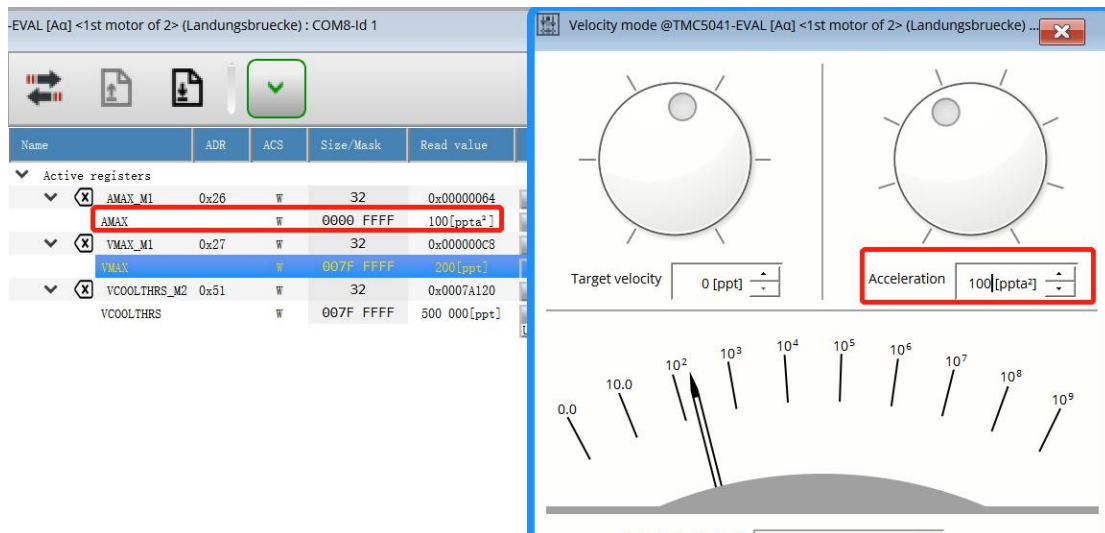


9-5: 配置按照下图配置 CHOPCONF 对于寄存器 0x6C



9-6: 在这之前请保证电机是静止状态。  
在调试 Stealthchop 的时候原则是先设置一个很小的加速度然后慢慢提高速度





(2) 第二代 Stealthchop 用 TMC5161 来测试，如果有条件的话最好配备一个电流探头使用示波器检测线圈电流，每款芯片的 datasheet 里面都会有如下快速配置指导

#### CURRENT SETTING AND FIRST STEPS WITH STEALTHCHOP

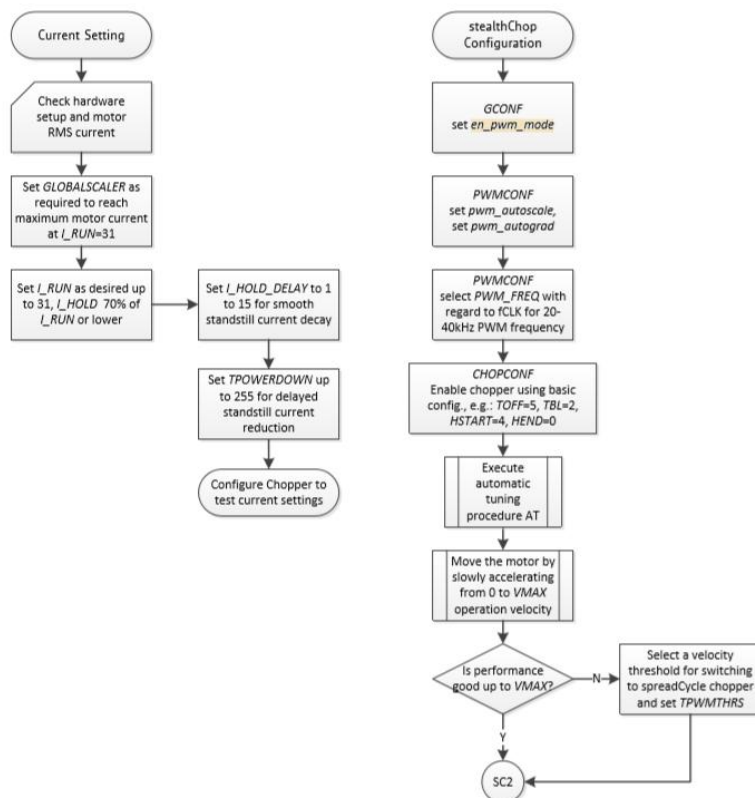
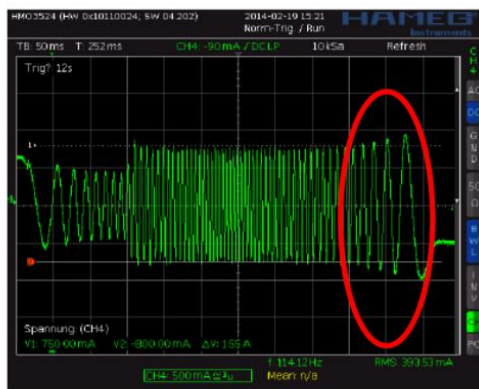


Figure 22.1 Current setting and first steps with stealthChop

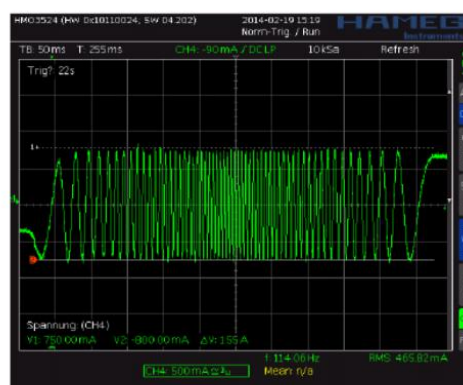


用电流探头检测电机线圈电流如果电流波形如第一个图的时候表示 Stealthchop 没有调试好，电流在加速和减速的时候有滞后现象，容易引起过流保护。下图的 After Tuning 是 Stealthchop 配置成功后的电流波形。

Before Tuning



After Tuning



配置步骤如下：

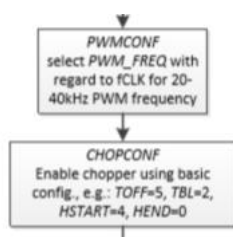
(1)使能 En\_PWM\_mode 为 High 对应 0x00 寄存器 如下图

Name	ADR	ACS	Size/Mask	Read value	To write value	Function
Active registers						
▼ <input checked="" type="checkbox"/> GCONF	0x00	RW	32	0x0000000C	0x0000000C	
recalibrate		RW	0000 0001	<input type="checkbox"/> low	<input type="checkbox"/> Set bit 0	
faststandstill		RW	0000 0002	<input type="checkbox"/> low	<input type="checkbox"/> Set bit 1	Normal...clocks
<b>en_pwm_mode</b>		<b>RW</b>	<b>0000 0004</b>	<input checked="" type="checkbox"/> high	<input checked="" type="checkbox"/> Set bit 2	<b>1: ste... only</b>
multistep_filt		RW	0000 0008	<input checked="" type="checkbox"/> high	<input checked="" type="checkbox"/> Set bit 3	Enable... source

(2)使能 PWM\_autoscale 和 PWM\_autograd 为 High 对应 0x70 寄存器，如下图

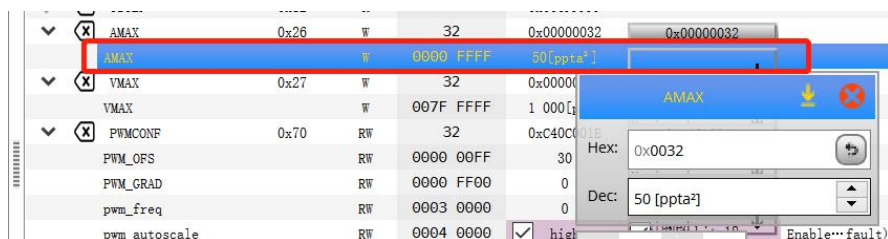
▼ <input checked="" type="checkbox"/> PWMCONF	0x70	RW	32	0xC40C001E	0xC40C001E	
PWM_OFS		RW	0000 00FF	30		User d...RUN).
PWM_GRAD		RW	0000 FF00	0		Veloci..._GRAD.
pwm_freq		RW	0003 0000	0		%00: f...=2/410
<b>pwm_autoscale</b>		<b>RW</b>	<b>0004 0000</b>	<input checked="" type="checkbox"/> high	<input checked="" type="checkbox"/> Set bit 18	<b>Enable...fault)</b>
<b>pwm_autograd</b>		<b>RW</b>	<b>0008 0000</b>	<input checked="" type="checkbox"/> high	<input checked="" type="checkbox"/> Set bit 19	<b>Automa...otion.</b>
freewheel		RW	0030 0000	0		Stand ...ivers.

(3) 配置下面 2 个寄存器

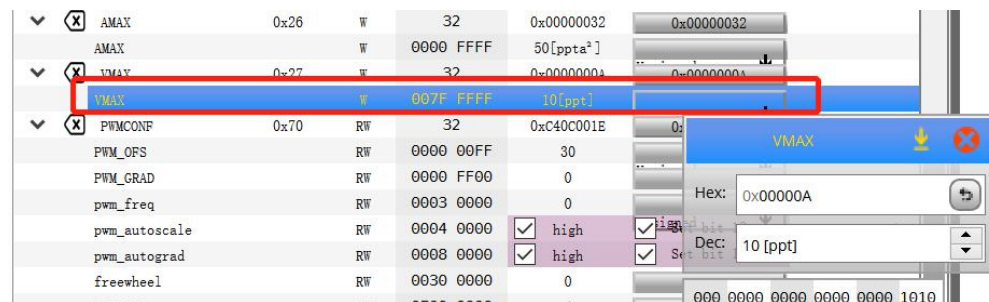


(4)开始 AutoTuning

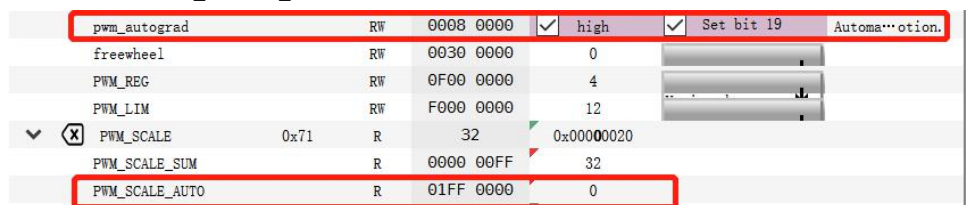
在电机停止时候（需要电机待机情况下超过 130ms），设置非常小的加速度 AMAX 0x26 设置为 50 设置更小



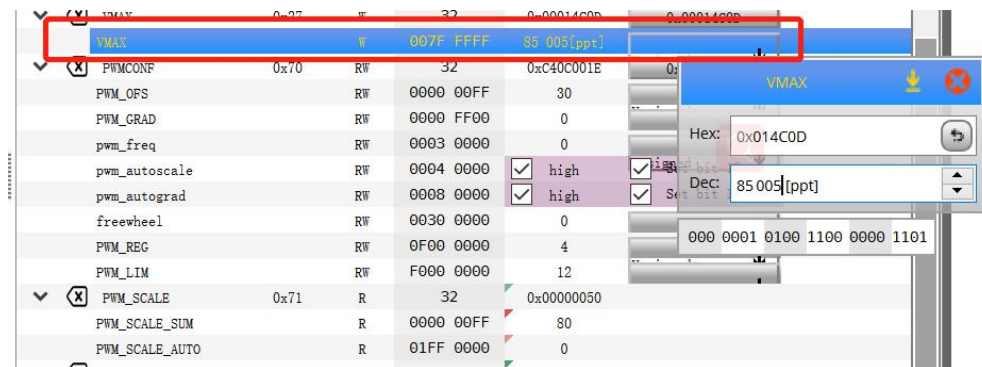
设置非常小的速度 VMAX 对应 0x27 寄存器设置为 10



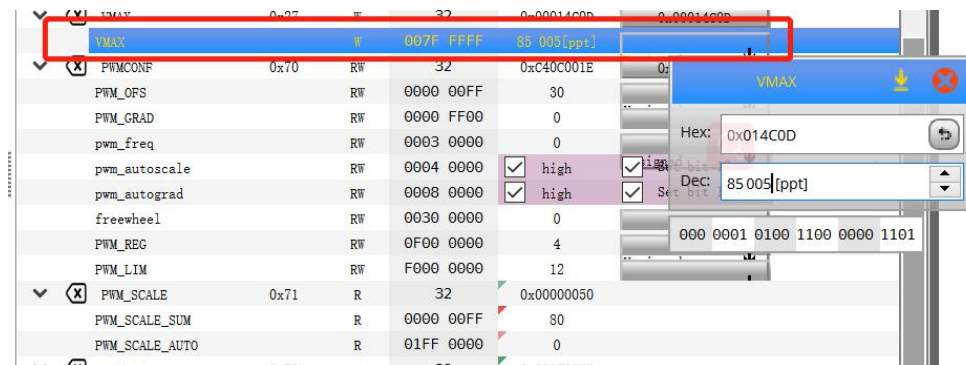
在电机运动过程中确认 PWM\_SCALE\_AUTO 等于 0, 如果等于 0 的话 PWM\_SCALE\_AUTO 会被自动写入 PWM\_GRAD\_AUTO



然后逐步提供最大速度 Vmax 和重复确认 PWM\_SCALE\_AUTO



直至加速到某个速度下电机出现抖动, 此时需要配置读取 TSTEP 参数将其设置到 TPWMTHRS 寄存器, 目的是切换至 Spreadcycle 模式



可以在每次开机回原点过程完成 AUTO TUNING.

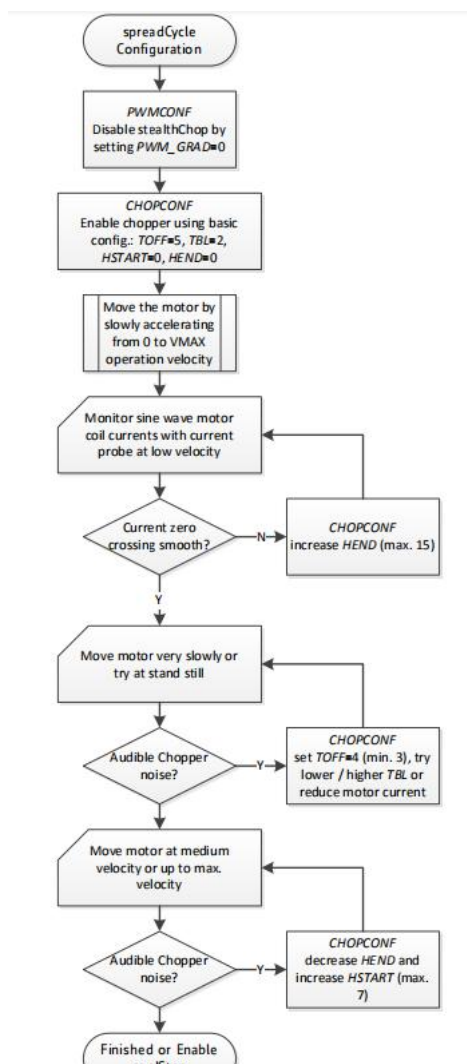
## 10. Spreadcycle 是高速运行模式

如果 Stealthchop 和 Spreadcycle 一起用的话, 通过 VCOOLTHRS 来切换, 低速时候用 Stealthchop 高速时候用 Spreadcycle 模式

W	0x31 0x51	23	VCOOLTHRS	<p>This is the lower threshold velocity for switching on smart energy coolStep and stallGuard feature. Further it is the upper operation velocity for stealthChop. (unsigned)</p> <p>Set this parameter to disable coolStep at low speeds, where it cannot work reliably. The stop on stall function (enable with <i>sg_stop</i> when using internal motion controller) becomes enabled when exceeding this velocity. It becomes disabled again once the velocity falls below this threshold. This allows for homing procedures with stallGuard by blanking out the stallGuard signal at low velocities (will not work in combination with stealthChop).</p> <p><b><math>V_{HIGH} \geq  V_{ACT}  \geq V_{COOLTHRS}</math>:</b></p> <ul style="list-style-type: none"> <li>- coolStep and stop on stall are enabled, if configured</li> <li>- Voltage PWM mode stealthChop is switched off, if configured</li> </ul> <p>(Only bits 22..8 are used for value and for comparison)</p>
---	--------------	----	-----------	--

如果是单独工作在 Spreadcyc 模式的话

参照以下步骤, 详情参照 Datasheet 第 14 章节



## 11. 如何将开发板的配置快速移植到自己的 MCU 里？

TMCL-IDE 3.0.25.9 - [Register browser (TMC5041) @TMC5041-EVAL [Aa] <1st motor of 2> (Landungsbruecke) : COM8-Id 1]

File Tools Options Views Help

23.7 V

Connected devices

Device: TMC5041-EVAL

Wizard Pool

Register (TMC5041)

Datagram mode

Direct mode

Flags view

Motor 1

Control mode

Veloc... mode

Posit... mode

Info graph

Address Name (set checked for auto u)

0x22 (34) VACTUAL\_M1

0x23 (35) VSTART\_M1

0x25 (37) V1\_M1

0x27 (39) VMAX\_M1

0x2B (43) VSTOP\_M1

0x31 (49) VCOOLTHRS\_M1

0x32 (50) VHIGH\_M1

0x33 (51) VDCMIN\_M1

0x42 (66) VACTUAL\_M2

0x43 (67) VSTART\_M2

0x45 (69) V1\_M2

0x47 (71) VMAX\_M2

0x4B (75) VSTOP\_M2

0x51 (81) VCOOLTHRS\_M2

0x52 (82) VHIGH\_M2

0x53 (83) VDCMIN\_M2

0x6F (111) DRV\_STATUS\_M1

0x7F (127) DRV\_STATUS\_M2

Group select: Ramp Gener...gister Set

Group select: Ramp Gener...gister Set

Group select: Driver Register Set

Group select: Motor 1

Group select: Motor 2

Export or import register content. Edit a comment in the last column To write value

Active registers

gCONF 0x00 RW 32 0x00000000 0x00000000

gSTAT 0x01 RW 32 0x00000000 0x00000000

IFCNT 0x02 R 32 0x00000000 0x00000000

TEST\_SEL 0x03 W 32 0x00000000 0x00000000

INPUT 0x04 R 32 0x00000007 0x00000000

COMPARE 0x05 W 32 0x00000000 0x00000000

PMCONF\_M1 0x10 W 32 0x000403FF 0x000403FF

PM\_STATUS\_M1 0x11 R 32 0x00000000 0x00000000

PMCONF\_M2 0x18 W 32 0x00000000 0x00000000

PM\_STATUS\_M2 0x19 R 32 0x00000000 0x00000000

RAMPMODE\_M1 0x20 RW 32 0x00000001 0x00000001

YACTUAL\_M1 0x21 RW 32 0x00000000 0x00000000

YACTUAL\_M2 0x22 R 32 0x00000000 0x00000000

YSTART\_M1 0x23 W 32 0x00000000 0x00000000

Y1\_M1 0x24 W 32 0x00000000 0x00000000

Y1\_M2 0x25 W 32 0x00000000 0x00000000

YMAX\_M1 0x26 W 32 0x00000004 0x00000004

AMAX 0000 FFFF 100[ppr4]

TMC5041

Export/Import

TMC5041: actual and saved writeable register values

No ini file loaded

ADR	Name	ACS	Value read	Value written	Value from file	Add a comment
0x64	MSLUT[4]	W	0xF8FFFFFF			
0x65	MSLUT[5]	W	0xB5B8777D			
0x66	MSLUT[6]	W	0x49295556			
0x67	MSLUT[7]	W	0x0404222			
0x68	MSLUTSEL	W	0xFFFF8056			
0x69	MSLUTSTART	W	0x00F70000			
0x6C	CHOPCONF_M1	RW	0x00010044	0x00010044		
0x6D	COOLCONF_M1	W	0x00000000			
0x6E	DCCTRL_M1	W	0x00000000			
0x7C	CHOPCONF_M2	RW	0x000101D5			
0x7D	COOLCONF_M2	W	0x00000000			
0x7E	DCCTRL_M2	W	0x00000000			

Export

Export conversion

Import/Export INI: actual register values

Import/Export INI: actual register values

Export C: register addresses as #defines

Export C: actual register values

Export/Import Options

add prefix {EXPRESSION} add suffix

Save to File

Export



将下面 4 的代码复制到自己的 MCU 里面

Export C: register addresses as #defines

## Export C: register addresses as #defines

This function generates C style defines of the IC register addresses. They may help during firmware development and make code more readable

```
// =====//
// ADDRESS DEFINES FOR TMC5041 (created: 2020/03/03 21:13:06) //
// =====//

#define TMC5041_GCONF 0x00 // (Address: 0)
#define TMC5041_TEST_SEL 0x03 // (Address: 1)
#define TMC5041_X_COMPARE 0x05 // (Address: 2)
#define TMC5041_FWMCONF_M1 0x10 // (Address: 3)
#define TMC5041_FWMCONF_M2 0x18 // (Address: 4)
#define TMC5041_RAMPMODE_M1 0x20 // (Address: 5)
#define TMC5041_XACTUAL_M1 0x21 // (Address: 6)
#define TMC5041_VSTART_M1 0x23 // (Address: 7)
#define TMC5041_A1_M1 0x24 // (Address: 8)
#define TMC5041_V1_M1 0x25 // (Address: 9)
#define TMC5041_AMAX_M1 0x26 // (Address: 10)
#define TMC5041_VMAX_M1 0x27 // (Address: 11)
#define TMC5041_DMAX_M1 0x28 // (Address: 12)
#define TMC5041_D1_M1 0x2A // (Address: 13)
#define TMC5041_VSTOP_M1 0x2B // (Address: 14)
#define TMC5041_TZEROWAIT_M1 0x2C // (Address: 15)
#define TMC5041_XTARGET_M1 0x2D // (Address: 16)
#define TMC5041_IHOLD_IRUN_M1 0x30 // (Address: 17)
```

或

将下面格式的代码复制到自己的 MCU 里面

Export conversion

Export C: actual register values

## Export C: actual register values

This function allows to generate C style SPI write function calls like:

```
FUNCTIONNAME (ADDRESS, VALUE); //COMMENT
```

With the editable textfields it's possible to add a prefix, which will be the function name in this case. The suffix terminates the function.

```
// =====//
// ACTUAL SETTINGS FOR TMC5041 (created: 2020/03/03 21:14:16) //
// =====//

TMC5041_SPIWriteInt(0x00, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 0 = 0x00(GCONF)
TMC5041_SPIWriteInt(0x03, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 1 = 0x03(TEST_SEL)
TMC5041_SPIWriteInt(0x05, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 2 = 0x05(X_COMPARE)
TMC5041_SPIWriteInt(0x10, 0x000403FF); // writing value 0x000403FF = 263167 = 0.0 to address 3 = 0x10(FWMCONF_M1)
TMC5041_SPIWriteInt(0x18, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 4 = 0x18(FWMCONF_M2)
TMC5041_SPIWriteInt(0x20, 0x00000001); // writing value 0x00000001 = 1 = 0.0 to address 5 = 0x20(RAMPMODE_M1)
TMC5041_SPIWriteInt(0x21, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 6 = 0x21(XACTUAL_M1)
TMC5041_SPIWriteInt(0x23, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 7 = 0x23(VSTART_M1)
TMC5041_SPIWriteInt(0x24, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 8 = 0x24(A1_M1)
TMC5041_SPIWriteInt(0x25, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 9 = 0x25(V1_M1)
TMC5041_SPIWriteInt(0x26, 0x000003E9); // writing value 0x000003E9 = 1000 = 0.0 to address 10 = 0x26(AMAX_M1)
TMC5041_SPIWriteInt(0x27, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 11 = 0x27(VMAX_M1)
TMC5041_SPIWriteInt(0x28, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 12 = 0x28(DMAX_M1)
TMC5041_SPIWriteInt(0x2A, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 13 = 0x2A(D1_M1)
TMC5041_SPIWriteInt(0x2B, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 14 = 0x2B(VSTOP_M1)
TMC5041_SPIWriteInt(0x2C, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 15 = 0x2C(TZEROWAIT_M1)
TMC5041_SPIWriteInt(0x2D, 0x00000000); // writing value 0x00000000 = 0 = 0.0 to address 16 = 0x2D(XTARGET_M1)
TMC5041_SPIWriteInt(0x30, 0x00071703); // writing value 0x00071703 = 464643 = 0.0 to address 17 = 0x30(IHOLD_IRUN_M1)
```

12. 相关视频链接:

- ① TMC5072 视频介绍: <https://v.qq.com/x/page/l0747nirhdb.html?>
- ② 如何从 TMC4671 开发板导出 C 代码: <https://v.qq.com/x/page/t3074g6n06a.html?>
- ③ 节能技术 Coolstep: <https://v.qq.com/x/page/d30741v23rz.html?>
- ④ 无传感 Stallguard: <https://v.qq.com/x/page/d3014mlftmz.html?>  
<https://v.qq.com/x/page/f083363nh01.html?>
- ⑤ TMC2208Stealthchop 和 Spreadcycle 切换: <https://v.qq.com/x/page/f0923nuiowi.html?>
- ⑥ 第二代 Stealthchop TMC5160 <https://v.qq.com/x/page/n300655p5qp.html?>  
[https://v.youku.com/v\\_show/id\\_XNDM2MzgONTgwMA==.html](https://v.youku.com/v_show/id_XNDM2MzgONTgwMA==.html)



Wrote By:Gao Wei

T:138-1261-7052

