TI Spins Motors

Quick Start Guide:
Evaluating InstaSPIN-MOTION (F2806xM) using the Graphical User Interface

Version 1.0.5 *Motor Solutions*

Introduction

Welcome and thank you for selecting TI's InstaSPIN solutions. This document will guide you through a quick evaluation of InstaSPIN-MOTION using a simple Graphical User Interface (GUI). When you complete this initial evaluation, proceed to the InstaSPIN-FOC and InstaSPIN-MOTION User Guide. The User Guide provides example projects (labs) intended for you to not only experiment with InstaSPIN but to also use as reference for your design. The User Guide will help you develop a successful product.

Product Overview

InstaSPIN-MOTION is a cascaded motion + speed + torque three phase motor control solution featuring a disturbance-rejecting speed controller that proactively compensates for system errors. The controller uses single-parameter tuning to significantly slash setup time and shorten design cycles. The single tuning parameter is effective across varying speeds and loads.

InstaSPIN-MOTION also includes a motion engine, which calculates the smoothest trajectory based on user-defined parameters, and a motion sequence planner that makes it easy to design complex motion sequences. The motion sequence planner is included in the User Guide, and will not be covered as part of the initial evaluation through the GUI.

The demonstration GUI and primary example projects use InstaSPIN-MOTION in a sensorless (self-sensored) system, taking advantage of the FAST™ software sensor and the InstaSPIN-FOC torque controller. The User Guide provides an example project for sensored feedback as well, and of course you can customize the inner torque control and modulation as you wish.

Supported Evaluation Kits

The InstaSPIN-MOTION (F2806xM) GUI allows you to quickly evaluate InstaSPIN™-MOTION using TI's evaluation kits and the TI provided motors or your own motor.

The GUI supports the following evaluation kits:

- TMDSCNCD28069MISO Piccolo F28069M (ROM) controlCARD paired with
- 3-phase Inverter
  - Low Voltage / Low Current: DRV8312_REVD
    - PN: DRV8312-69M-KIT
  - Low Voltage / High Current: DRV8301_REVD
    - PN: DRV8301-69M-KIT
    - Do not try to use with BOOSTXL-DRV8301
  - High Voltage: hvmtrkit_r1p1 [Yellow PCB with black enclosure]
    - PN: TMDSHVMTRINSPIN
    - Do NOT use TMDX or TMDSHVMTRKIT5X [Yellow PCB with light grey enclosure] with CNCD28069MISO

The Evaluation Process

The GUI is designed to quickly guide you through the InstaSPIN-MOTION evaluation process. The Evaluation process is depicted in Figure 1. Once you determine that InstaSPIN-MOTION is right for your application, use the MotorWare-based projects, in conjunction with the InstaSPIN-FOC and InstaSPIN-MOTION User's Guide to design your project and conduct performance testing.

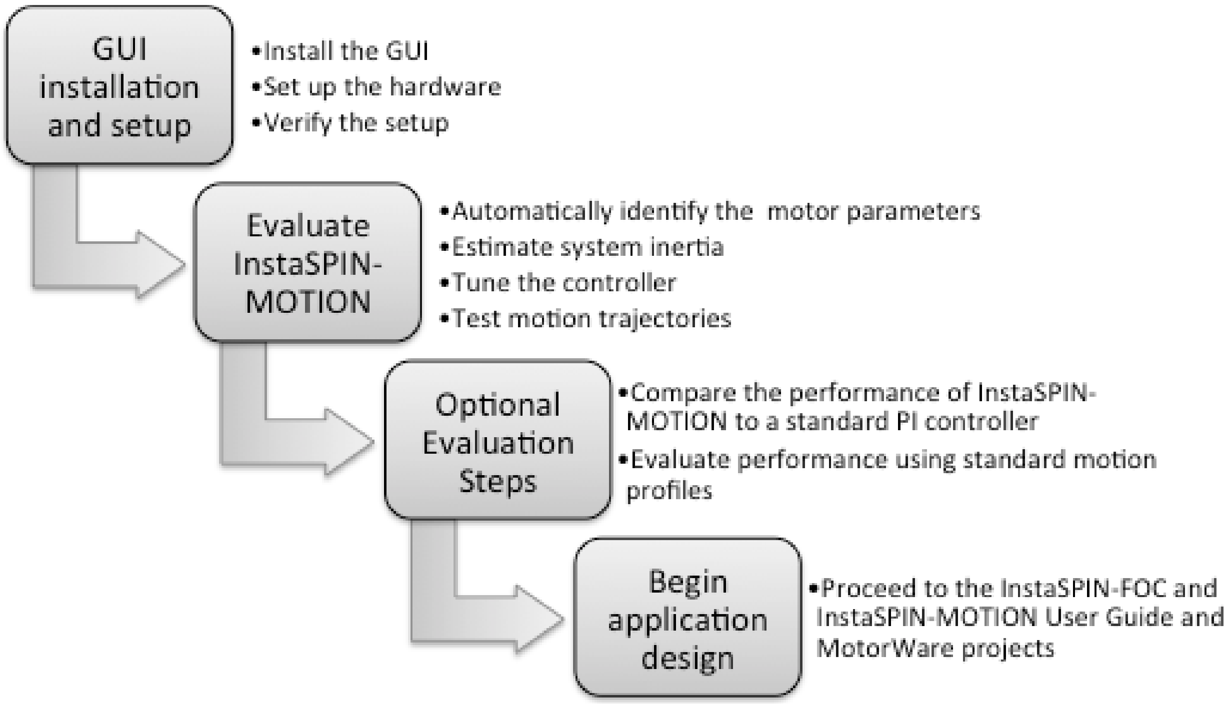


Figure 1 - InstaSPIN-MOTION Evaluation Process

------------------ NOTICE ----------------------

The C code project (CCS developed target binary) that runs on these kits is FROZEN. It is at a fixed PWM & control frequency and fixed user scalings which are NOT appropriate for all motors. It also does not include the CRITICAL updates we have made to the solution since introduction in early 2013. You MUST use MotorWare projects to get these important features.

This GUI should only be used for general demonstration. Any performance testing should be done using the MotorWare based projects with careful consideration paid to setting up hardware and software scaling for proper motor ID and best performance.

------------------ NOTICE ----------------------

Version: 1.0.5

Revision History:

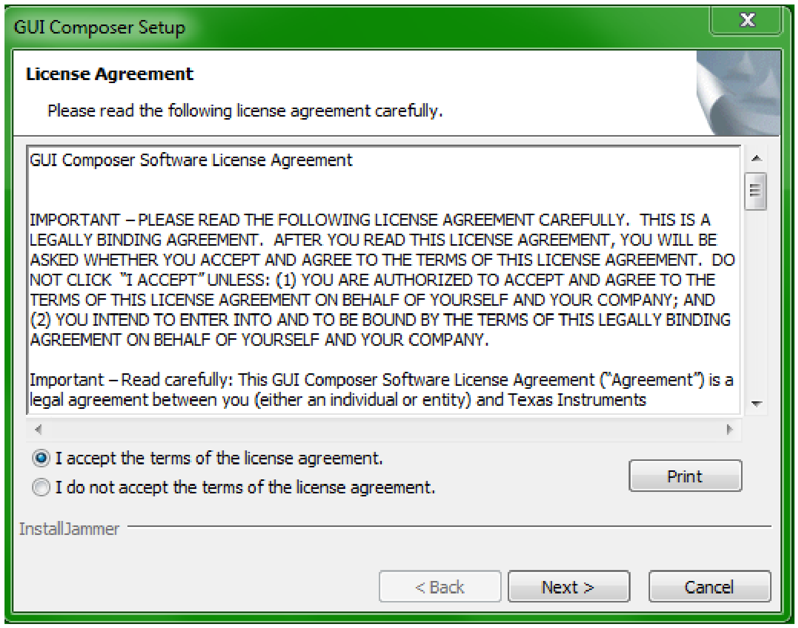
1.0.5	**April 23, 2014**	**Updates regarding Warnings, Scalings, and direction for using MotorWare projects instead of the GUI.**
1.04	**May 9, 2013**	**Updated appProgram.out to support ADC Clock fix from MotorWare 1_01_00_07** **Changed name of GUI installer that supports both the –FOC and –MOTION GUIs to GUI_Composer_InstaSPINs_F2806xM_install.exe** **Latest GUI installer includes patch to support PCs with multiple cores**
1.03	**April 5, 2013**	**Revised for InstaSPIN-MOTION release**
1.0.2	**March 21, 2013**	**Added Revision History**
1.0.1	**February 26, 2013**	**First release**

Table of Contents

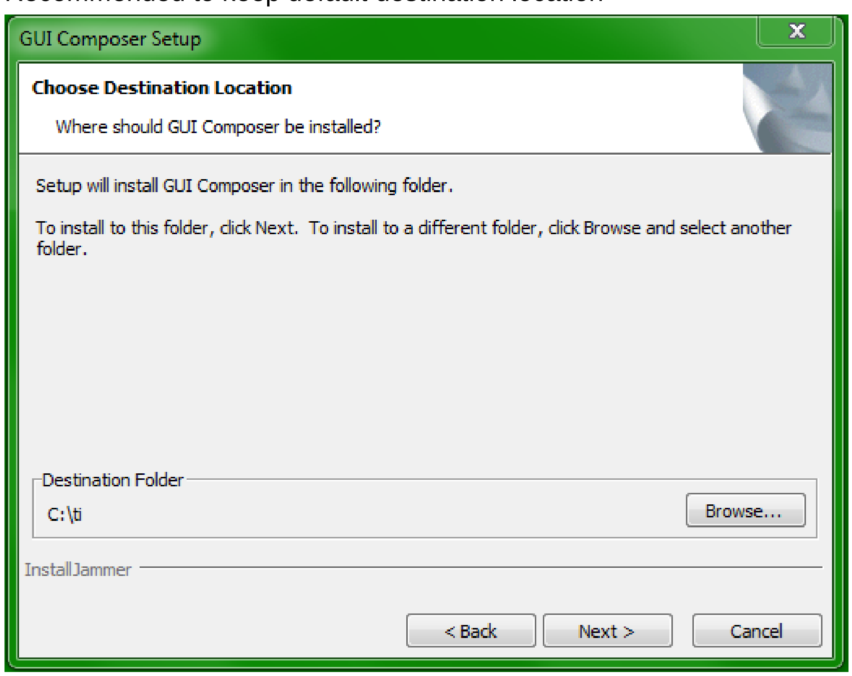
GUI Installation and Setup

Installation

1. Obtain the latest "kit contents package" from http://www.ti.com/tool/motorkitscncd69miso
2. If this is the first time to install a GUI composer project, run GUI_Composer_InstaSPINs_F2806xM_install.exe
    a. Accept the license agreement



    b. Recommended to keep default destination location



3. Once the installer has been run, for any future updates to the GUI (available through MotorWare or at the kit contents download) you may simply copy the updated folder "InstaSPIN_MOTION_F2806xM" to C:\ti\guicomposer\webapps

Hardware Set-up

1. All new kits should arrive with proper settings for all jumpers and switches
2. Please review the Quick Start Guides and hardware documentation – available through MotorWare - for details
3. Install F28069M controlCARD into the controlCARD slot
4. Check Critical Board Settings
    a. TMDSCNCD28069MISO
        i. SW1: UP-UP
        ii. SW2: DOWN-DOWN
        iii. SW3: UP-UP
    b. DRV8312_REVD
        i. JP1: VR1 - Middle
        ii. JP2: Populated
        iii. M1: High-Middle
        iv. All three toggle switches in the middle (MCU) position
        v. Motor Phase Wires to MOA, MOB, MOC
        vi. DC Voltage to J9 or PVDDIN
    c. DRV8301_REVD
        i. JP2: Populated
        ii. Motor Phase Wires to OUTA, OUTB, OUTC
        iii. DC Voltage to PVDD and GND
    d. HVMTR_r1p1
        i. Populate: J3, J4, J5, J8,
        ii. Populate: M3-J5
            1. (turns OFF on-kit emulation so you can use controlCARD emulation)
        iii. J7: Right-Middle (towards capacitors)
        iv. Motor Phase Wires to U, V, W
        v. J2
            1. For DC bus input (recommended): Do NOT Populate J2 and

a. connect 50-350V DC supply from BS6 to BS5

2. For AC input: Populate J2 Bridge-Middle

   a. connect banana cable from BS5 to BS1

   b. connect AC power cord to P1

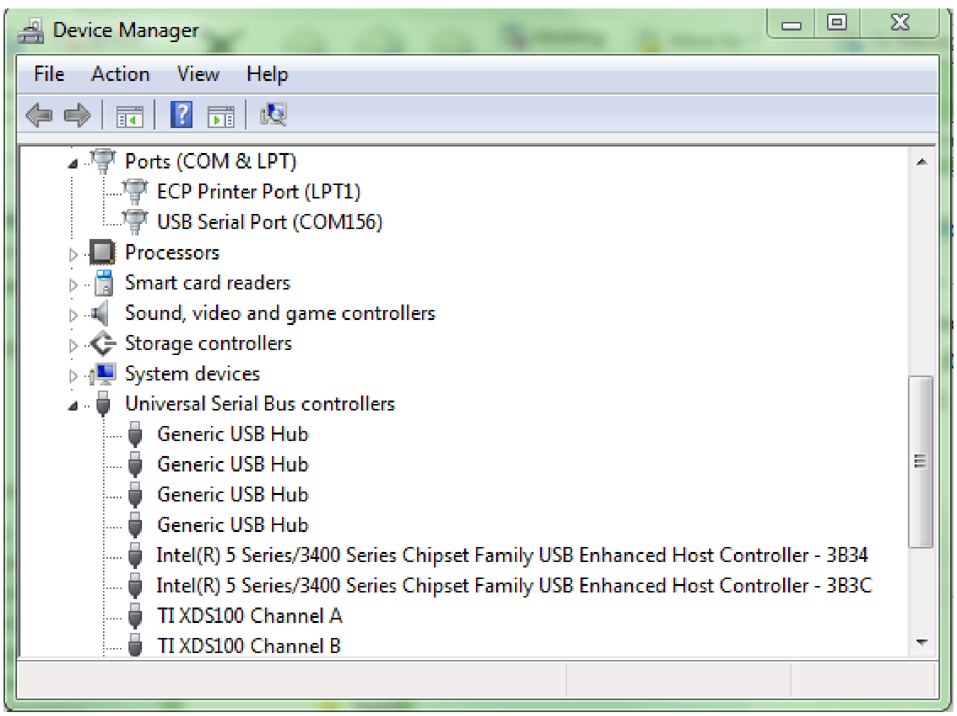
3. DO NOT APPLY HIGH VOLTAGE AC OR DC UNTIL ALL CONNECTIONS ARE CHECKED, VERIFIED, AND YOU HAVE LAUNCHED THE GUI

vi. 15V DC power supply to JP1 and SW1 ON

Launch the GUI

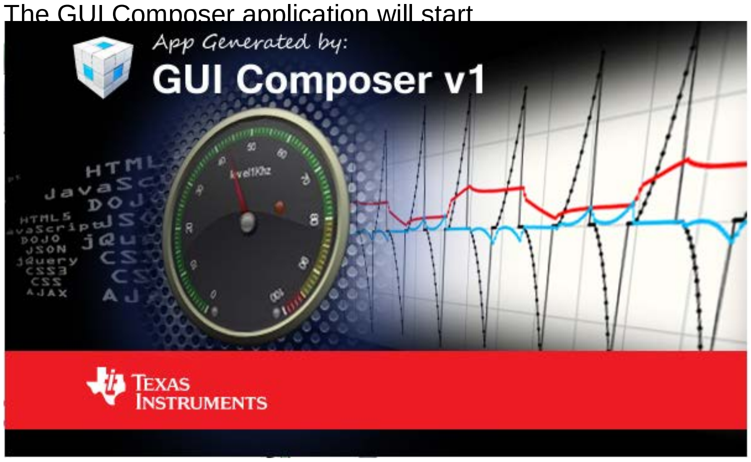
1. With low voltage DC powered, connect USB from your PC to J1 on controlCARD

   a. You can verify that you connected to the FTDI XDS100v2 emulator by checking your Windows Device Manager for Ports: USB Serial Port (COMxxx) and TI XDS100 Channel A and B



2. Run: **[C:\ti\guicomposer\webapps\InstaSPIN_MOTION_F2806xM\InstaSPIN_MOTION_F2806xM.exe]**

a. The GUI Composer application will start



b. GUI Composer will initialize, connect, then load \appProgram.out into the memory of the Piccolo F28069M.

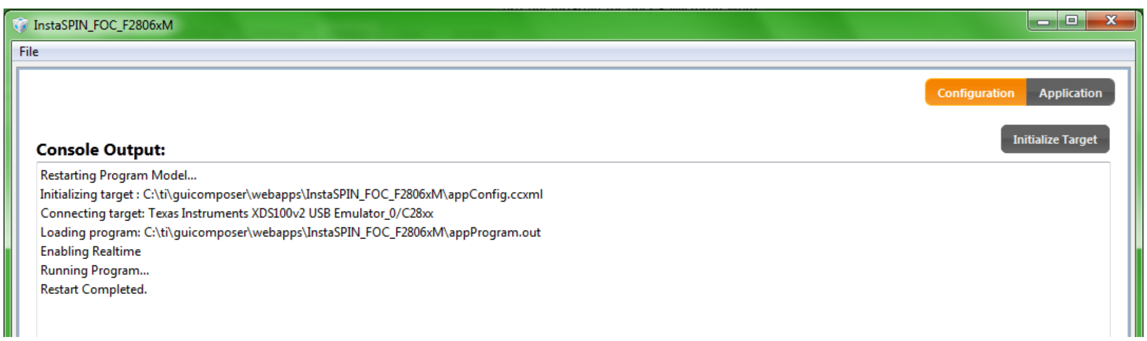


Figure 2 - Console Output for a Successful GUI Launch

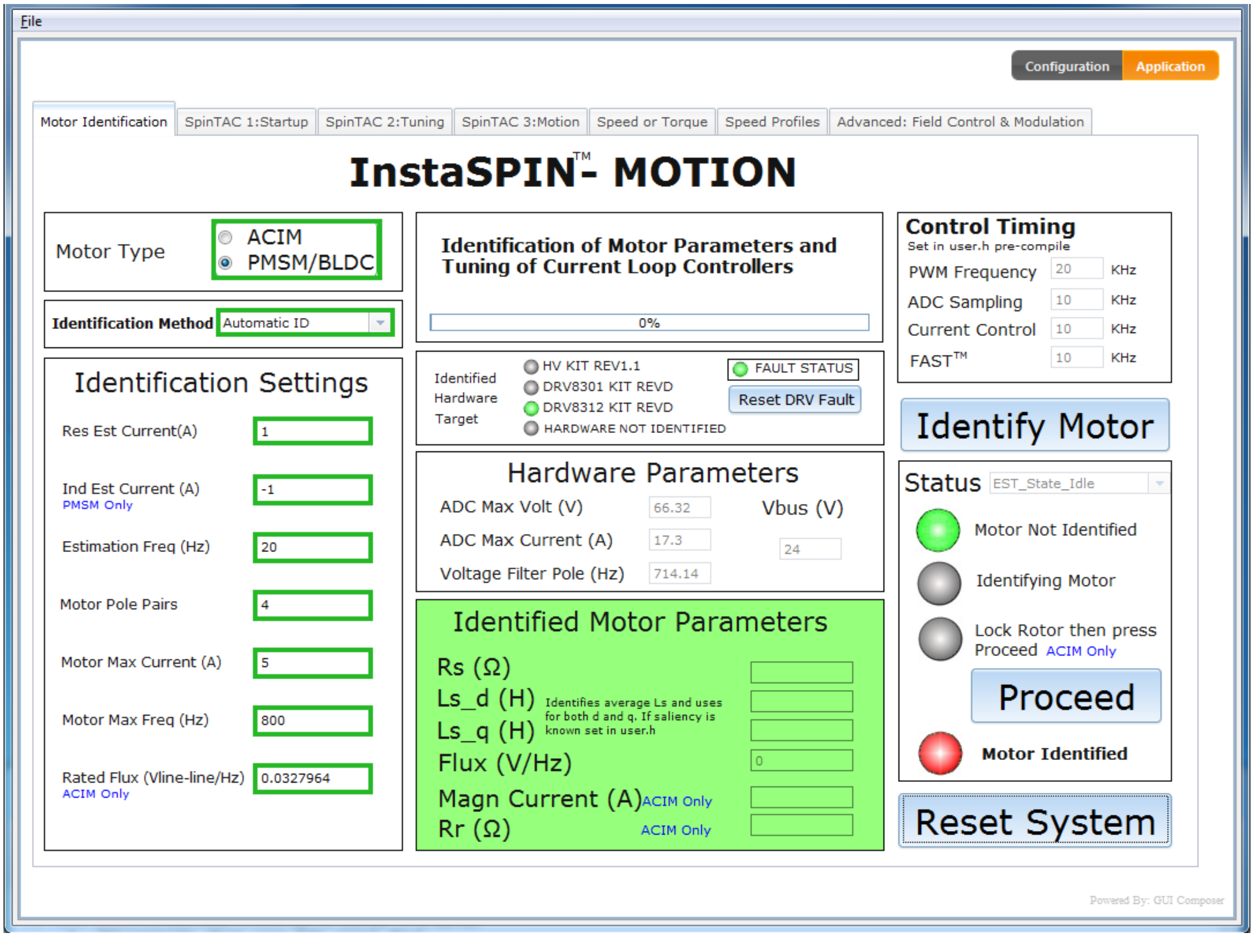


Figure 3 - Initial GUI Screen Display

c. If using the high voltage kit you can now energize the high voltage AC input (110-220Vac) or high voltage DC Bus (50-350Vdc)

Evaluating InstaSPIN-MOTION

Both InstaSPIN-FOC and InstaSPIN-MOTION are sensorless FOC solutions that identify, tune and control your motor in minutes. Both solutions feature:

- The FAST unified software observer, which exploits the similarities between all motors that use magnetic flux for energy transduction. The FAST estimator measures rotor flux (magnitude and angle) in a sensorless FOC system.
- Automatic torque (current) loop tuning with option for user adjustments
- Automatic or manual field weakening and field boosting
- Bus voltage compensation

InstaSPIN-MOTION combines this functionality with SpinTAC™ components from LineStream Technologies. SpinTAC features:

- A disturbance-rejecting speed controller, which proactively estimates and compensates for system errors. The controller offers single-parameter tuning that typically works over the entire operating range.
- Trajectory planning for easy design and execution of complex motion sequences (Note: this feature will not be exercised through the GUI. See the InstaSPIN-FOC and InstaSPIN-MOTION User Guide and MotorWare projects to exercise SpinTAC Plan).
- A motion engine that insures that your motor transitions from one speed to another as smoothly as possible.

Motor Identification

InstaSPIN measures and keeps track of motor parameters, eliminating the need for a known motor model. Although identifying the motor is not a must for all applications, it provides an easy and better out of the box experience to run any given motor sensorlessly. Other algorithms in the marketplace require an intensive tuning process upfront, even before running the motor in closed loop.

This section describes InstaSPIN's motor identification process. Significant effort has been spent ensuring both the algorithms and steps describe will successfully identify a large number of motor types. But, one should not expect that the algorithms and steps described will always successfully identify all motors or motor types. If you experience difficulty identifying a motor, please reference the Motor Identification Troubleshooting section of the InstaSPIN-FOC and InstaSPIN-MOTION User Guide and **plan to use the latest software projects in MotorWare, which has many significant updates for motor identification that are NOT included in this FROZEN embedded project that works with the GUI**.

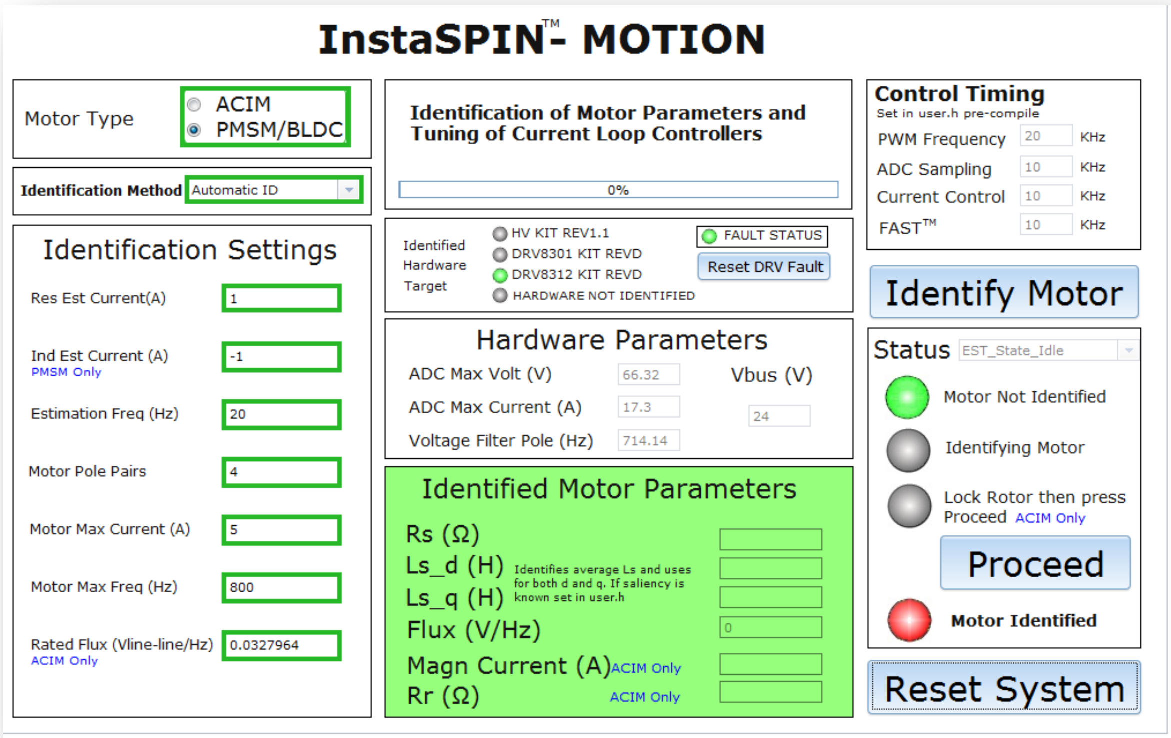
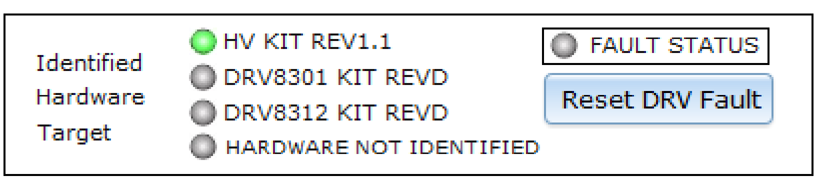


Figure 4 - Motor Identification Tab

Identified Hardware Target

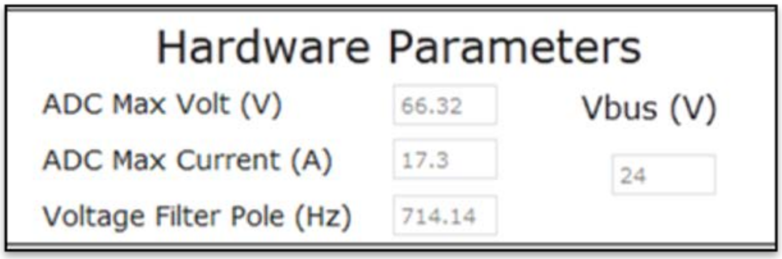
The GUI should automatically recognize the evaluation kit. Only proceed if the GUI has properly identified the target hardware you are using.



- Fault Status
  - The fault status light will be GREEN if NO FAULTS are present only on the DRV8312 or DRV8301
  - The fault status light will only turn RED if a FAULT is present on the DRV8301
  - Reset DRV Fault works only for DRV8301
  - DRV8312 includes a fault LED (/FAULT) on the hardware, and must be cleared using the toggle switches (move RSTA from MCU to RESET to MCU).

Hardware Parameters

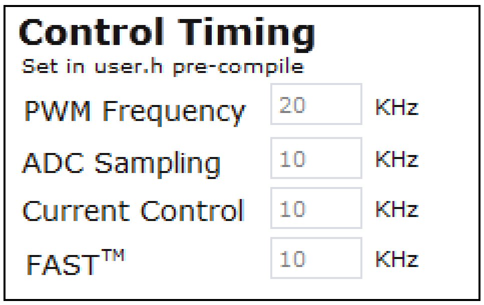
Based on the identified hardware, this displays the hardware settings



- ADC Max Voltage, is scaled by the evaluation kit to be seen as 3.3V input of the ADC
    - Note that FAST observer performance – especially low speed – is dependent on resolution of the measured phase voltage signals. In your own design you will find better performance by designing your hardware to exactly match the expected range of voltage measurements (based on the larger of the bus voltage or EMF of the motor at maximum expected frequency).
- ADC Max Current, is scaled by the evaluation kit to to be seen as 3.3V input of the ADC
    - Note that InstaSPIN-MOTION current control performance – especially low load – is dependent on resolution of the measured shunt current signals. In your own design you will find better performance by designing your hardware to exactly match the expected range of current measurements (based on the current/torque of the motor at maximum expected load).
- Voltage Filter Pole, hardware RC filter on phase voltage readings is critical to performance of the FAST observer.
- Vbus: Real voltage reading of the DC bus of the inverter
    - Insure that you are providing enough bus voltage for proper performance evaluation.
    - Especially with the High Voltage kit, most applicable motors are specified for 280-350Vdc. If AC entry is used the rectification from a 110Vac line is too low (ex: 165Vdc) to meet speed and torque expectations.

Control Timing

For the GUI, these default setting are chosen to work across all evaluation kit hardware platforms and most of the target applications for this technology. These settings have not been optimized for each application. Some motor types will require changes.



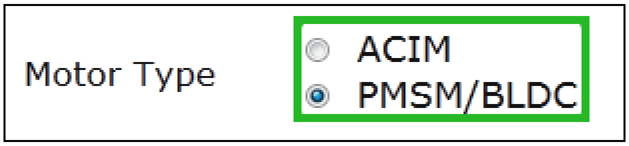
- The control timing is fixed in the user.h file used to compile this FROZEN project
  - **For some motors the PWM, current, or FAST estimaton rates are too low for proper control. You MUST use the MotorWare projects and update the user.h file accordingly.**
- For the GUI these setting are chosen to work across all hardware platforms and most of the target applications for this technology. **It is not optimized for each application and there will be motor types which require changes.**
  - PWM Frequency: 20 KHz default
    - Typically want to use the lowest possible to reduce switching losses but 8 – 20 KHz is the most typical range
    - Lower inductance (often higher current) motors typically require higher PWM frequencies (30 – 100 KHz) to effectively control switching
  - ADC Sampling: 10 KHz default
    - ADC sampling should be synchronized to the higher of the Current Control (typical) or FAST rate.
    - In higher PWM applications where the current control rate will run slower anyways (to allow the control system to run on < 100 MHz MCUs) the ADC can be decimated by using the Piccolo ePWM ADC SOC hardware (recommended) through the user.h software decimation.
    - The ADC conversions done signal starts the main control interrupt to run the sensorless FOC control system
  - Current Control: 10 KHz default
    - Synchronized to the ADC in this example
    - In extremely high dynamics (speed or load changes), extremely high speeds, or when controllng low inductance motors with large current ripple you want to run as often as possible (15-20 KHz) for maximum performance

- In lower dynamic applications 5 KHz is often acceptable

- FAST: 10 KHz default
  - Synchronized to the Current frequency in this example,
  - For extremely high dynamics (speed or load changes) or extremely high speeds you would want to run as often as possible (15-20 KHz) for maximum performance
  - In lower dynamic applications 2.5 KHz or 5 KHz is often acceptable

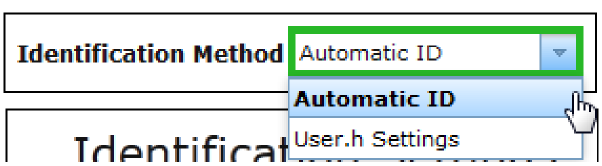
- The Speed and Trajectory Controller is set to be run at 1 KHz, but can be modified as desired in user.h
  - Excellent results have been seen by running the speed controller at the FAST rate due to the accuracy of the speed estimation from FAST

Motor Type



- Select the three phase motor you wish to identify
  - Asynchronous (AC Induction)
  - Synchronous (permanent magnet or brushless DC)
    - Interior Permanent Magnet (IPM) salient motors are also supported by the InstaSPIN-MOTION solution, but with a caveat on the motor identification
      - During identification the motor is treated as a non-salient motor and an average stator inductance is identified.
    - To maximize efficiency of salient (IPM) motors
      - For most accurate operation of the FAST observer
        - Ls-d and Ls-q must be user provided in user.h
        - Ls-d and Ls-q will change across motor loading, so in a full system a look-up table technique to load new Ls values to FAST should be considered
      - For most efficient FOC operation
        - A special maximum torque per amp algorithm may need to be considered
        - This will adjust the torque command output of the speed controller to the Iq and Id reference inputs of the InstaSPIN-MOTION torque controller
        - An example will be provided in the future

Identification Method



- Automatic ID
  - Select “Automatic ID” to use the Motor Identification portion of InstaSPIN-MOTION to identify the required motor parameter of your motor
- User.h Settings

- Select “User.h Settings” to bypass the Identification process and load the values and Identification Settings stored under USER_MOTOR in the user.h file that was used during project compile.
    - **Re-compiling of this project will NOT be offered**
    - **You must move to the MotorWare projects to save your USER_MOTOR settings in user.h**
- The values stored are for the DRV8312 EVM and the included 24V Anaheim/Telco BLDC motor only
- Default settings from user.h (for Anaheim / Telco NEMA17 24V BLDC)

#define Anaheim_BLY172S 102

#define USER_MOTOR Anaheim_BLY172S

#elif (USER_MOTOR == Anaheim_BLY172S)

#define USER_MOTOR_TYPE MOTOR_Type_Pm

#define USER_MOTOR_NUM_POLE_PAIRS (4)

#define USER_MOTOR_Rr (NULL)

#define USER_MOTOR_Rs (0.4110000000)

#define USER_MOTOR_Ls_d (0.0007092811)

#define USER_MOTOR_Ls_q (0.0007092811)

#define USER_MOTOR_RATED_FLUX (0.0327964)

#define USER_MOTOR_MAGNETIZING_CURRENT (NULL)

#define USER_MOTOR_RES_EST_CURRENT (1.0)

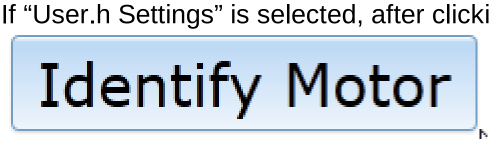
#define USER_MOTOR_IND_EST_CURRENT (-1.0)

#define USER_MOTOR_MAX_CURRENT (5.0)

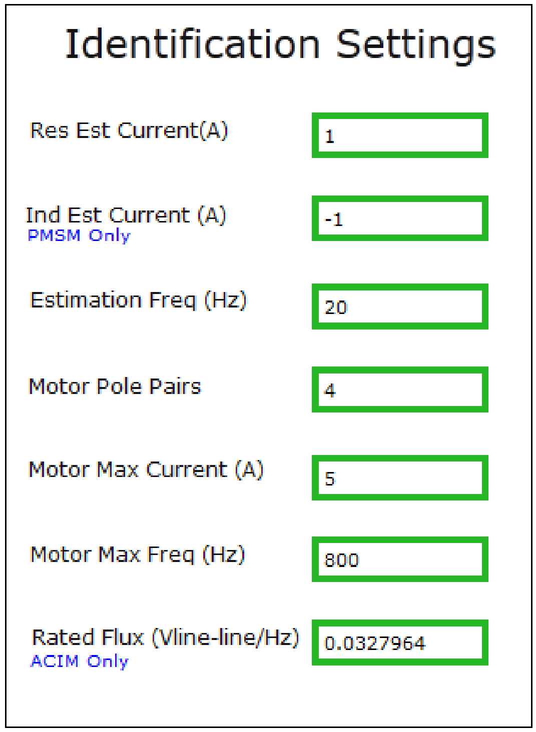
#define USER_MOTOR_FLUX_EST_FREQ_Hz (20.0)

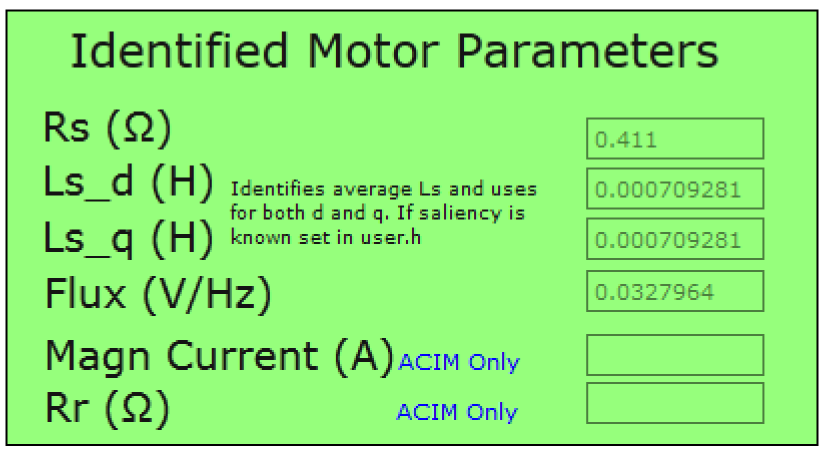
- The **BLACK** settings from user.h are displayed in the GUI under Identification Settings
- The **RED** settings from user.h are what are identified (for synchronous motors)

* If “User.h Settings” is selected, after clicking

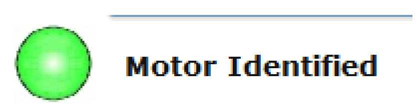


the **RED** values from user.h will be pulled and displayed as the Identified Motor Parameters





- And Motor Identified will turn green



- Note that when loading from user.h it will also load the hardware current and voltage offsets. Ex:

#define USER_DRV8312_KIT_REVD_I_A_offset (0.7386139631)

#define USER_DRV8312_KIT_REVD_I_B_offset (0.7363774185)

#define USER_DRV8312_KIT_REVD_I_C_offset (0.7402219176)

#define USER_DRV8312_KIT_REVD_V_A_offset (0.1301432252)

#define USER_DRV8312_KIT_REVD_V_B_offset (0.1301434636)

#define USER_DRV8312_KIT_REVD_V_C_offset (0.1303436756)

Identification Settings

With this FROZEN GUI project:
Most Induction Motors will NOT identify correctly, use MotorWare proj_lab02a or 2b for Motor ID.
Most high speed, low Ls, or low Flux motors will NOT identify correctly, use MotorWare proj_lab02c for Motor ID.

The **IQ_FULL_SCALE_VOLTAGE and_CURRENT** variables are not made accessible through the GUI - only adjustable through the user.h file and recompilation - but it is important to understand their impact on the system.

	IQ_FULL_SCALE _VOLTAGE_V (0-MAX)	**ADC_FULL_SCALE _VOLTAGE_V (V) (peak-peak)**	**IQ_FULL_SCALE _CURRENT_A (0-MAX)**	**ADC_FULL_SCALE _CURRENT_A (peak-peak)**
DRV8312	**48.0**	**66.32**	**10.0**	**17.3**
DRV8301	**48.0**	**66.32**	**40.0**	**82.5**
HVMTR	**450.0**	**409.9**	**10.0**	**19.89**

Unlike the ADC_FULL_SCALE values, which are defined by the specific hardware sensing circuit, the IQ_FULL_SCALE values are adjustable on any given project and are used to normalize all of the voltage/current terms to a per unit value (pu).

- For the current this is easy to define as you will never have a current variable value higher than what the hardware can measure.
  - Note however that the resolution is impacted as you attempt to cover a broader range of current. This lower resolution for the DRV8301 will affect Motor ID and run-time performance when used with a low current motor.
- For voltage you must also consider the Bemf of the motor and the calculations occurring inside the algorithms. These voltages inside the motor can be greater than the input voltage itself in cases where the motor is operated in field weakening or where the motor is operated beyond its rated speed.
  - When selecting this value to be used for Motor ID you would like to maximize your resolution, so choosing a low enough value is critical.

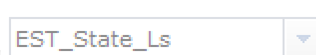
- When selecting this value for full operational run-time you MUST insure that the IQ_VOLTAGE value is greater than any internal voltages of the motor (back EMF). If not, a variable overflow will occur, leading to unexpected results from the estimator.
- For this GUI project values were selected for good resolution during Motor ID and normal operation and evaluation of most motors.
  - **Due to this, large Bemf motors may NOT perform as expected with this GUI project at highest speeds and field weakening regions. This value must be adjusted in those cases. See the User's Guide for more information and use MotorWare projects.**

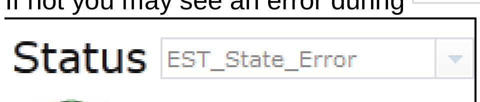
Res Est Current: Constant current used during resistance testing (positive number). This value should typically be ~10% maximum rated current – or current needed to produced rated torque - of the motor. It must be large enough (may be >10%) to insure that the motor starts spinning during the Estimation State RampUp though. Avoid too high of a current which will cause overheating of the motor during identification.

Ind Est Current: Constant current (negative) used during inductance testing.

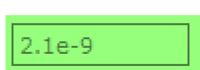
- **PMSM**: This value should be ~10% (negative) of the rated current. In general this value is the negative value of the current used for resistance estimation.
- **ACIM**: When identifying ACIM motors, this inductance estimation current is ignored and not used. Set to 0.

Estimation Freq: Maximum target frequency used during estimation, ~10% of rated speed.

- **PMSM:** 20Hz typical works for most motors. Exceptions
  - if the motor's inductance is known to be in the tens of micro Henry (µH), this frequency should be increased to 40 Hz, 60 Hz, or even as high as 150Hz until a reliable value is obtained.
  - When using the DRV8301 kit with any lower voltage/current motors, 20 Hz may not provide an accurate value for the inductance.
    - If not you may see an error during 



or values that are obviously too low



    - This is due to the resolution of the feedback as scaled by the board's hardware ADC scaling and the software IQ scaling.
    - Use as much voltage as possible and increase the Estimation Frequency, starting at 40 Hz until reliable results are obtained

  - **Use MotorWare proj_lab02c for high speed, low Ls,, or Low flux motors with attention paid to IQ_VOLTAGE scaling**

- **ACIM:** Estimation frequency should always be 5Hz for 50/60 Hz motors.
    - **It is recommended to use MotorWare for all induction motors as Motor ID was significantly improved starting with MotorWare 1_01_00_12**

Motor Pole Pairs: Used to calculate

- User friendly RPM from Electrical Frequency (RPM = Hz * 120 / poles)
- Accurate estimation of the shaft torque in N.m.

Motor Max Current: This is the maximum value allowed as the output of the Speed Controller (input to the Iq PI controller); it is NOT a safety setting for total current through the motor and should be set no higher than the maximum current rating of the motor.

- This value is also directly used for calculation of the initial Speed PI control Kp and Ki gain values as a "rule of thumb" that generally provides a stable – though not tuned – speed control output to a torque control system. Double the Motor Max Current and you will notice that the speed Kp and Ki also double.

Motor Max Freq: Used for per unit scaling of all frequencies in the system, this should be slightly higher than the highest electrical frequency the motor will target. This is in electrical frequency, so if the absolute maximum speed of a motor is 10,000 RPM on a 2 pole pair motor, this frequency should be (10,000/120)*4 = 333.33Hz. Some headroom is recommended, so in this case a Motor Max Frequency of 500Hz could be used

- **Even for low frequency motors it is recommended to keep the max frequency at a minimum of 500 Hz. Larger values do not negatively affect the control system.**
- Motor Max Freq must be < 4 * the Hardware Voltage Filter Pole (with some buffer)

DRV8301_revD	335.648 Hz	Motor Max Freq Max = 1200 Hz
DRV8312_revD	714.14 Hz	Motor Max Freq Max = 2500 Hz
HVMTR_rev1p1	372.5 Hz	Motor Max Freq Max = 1300 Hz

- Note that the Motor Max Frequency also subsequently affects another portion of the system, the speed at which the ForceAngle feature switches off and FAST estimations are used.
- The frequency of this switching point is set by Motor Max Frequency multiplied by USER_ZEROSPEEDLIMIT (from user.h). In this GUI project USER_ZEROSPEEDLIMIT is set to 0.002 and cannot be changed unless you re-compile. Note how this effects the lowest operating speed at which FAST estimations will be used as you change the Max Frequency

Motor Max Freq (Hz)	Switch from ForceAngle to. FAST (Hz)
200 Hz	0.4 Hz
500 Hz	1.0 Hz

2000 Hz	4.0 Hz

- For the default settings that load from user.h, the ForceAngle to FAST switch frequency = 800 * 0.002 = 1.6 Hz; 1.6 Hz * 120 / 8 poles = 24 RPM. FAST will be active over +/- 24 RPM, and inside this range the ForceAngle will be active unless disabled.

- **Note that the ForceAngle feature is used during motor identification, so this effective frequency of switching from ForceAngle to FAST must < than Estimation Freq or FAST will not be activated, hence it will not provide the necessary feedback for proper ID.**

- While FAST is capable of tracking at <1 Hz speeds, the lowest frequency for any system is completely dependent on the hardware and software scaling, quality of sensing, and Bemf (flux * speed_Hz) of the motor. You would also need to insure that the ForceAngle was set below the speed you want to run or is disabled during run-time. See the User's Guide further details.

    - For demonstrations on the TI motor kits at normal operating voltages 1-5 Hz for FAST is a reasonable expectation.

    - While you can't directly change this ForceAngle setting for a given Max Frequency through this GUI interface, keep this in mind as you are evaluating low speed performance

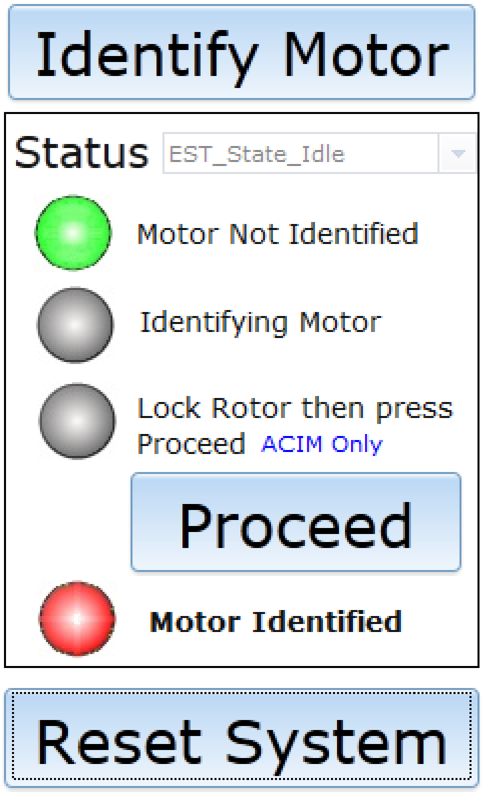
- It should be noted that motors that are designed t to run at over 1.5 KHz may not be fully supported by InstaSPIN-MOTION at this time. Please see the User's Guide for discussion on this topic and use MotorWare proj_lab02c for proper identification.

Rated Flux:

- **ACIM:** This value should be: SQRT(2/3) * Rated Voltage from line to line / Rated Frequency. For example, a 220VAC/60Hz ACIM motor is to be identified, the value to be entered for Rated Flux is: 2.99.

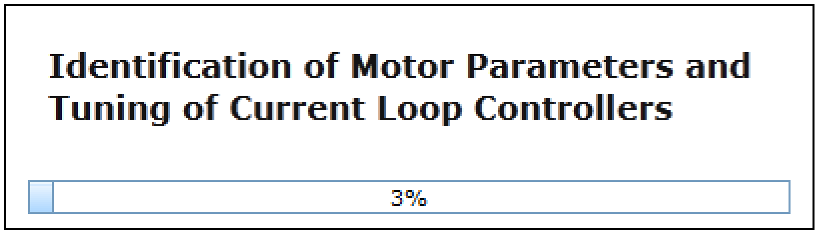
- **PMSM:** This setting is ignored for PMSM motors as it is a value identified.

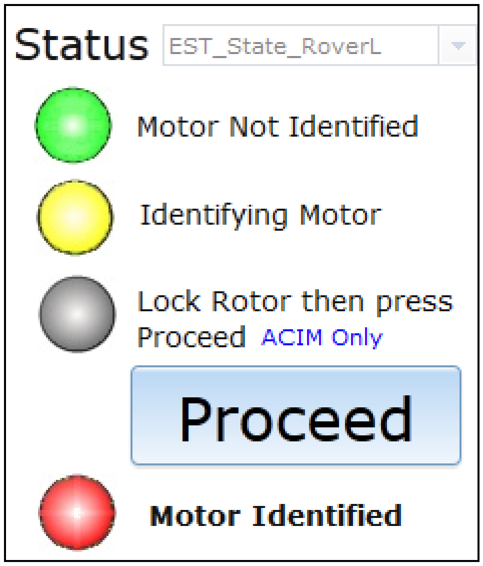
Motor Identification



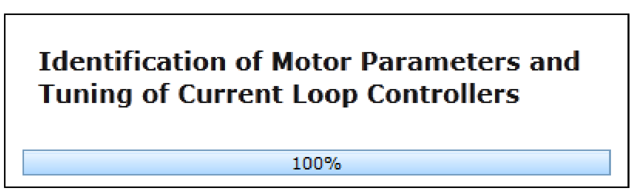
- The motor identification process, state machine, hardware scaling, choosing of PI control gains, and details of the tests being performed are documented in the User's Guide
    - While many synchronous motors will identify perfectly, there are known corner cases - especially for particular motor types and conditions -where the motor ID settings accessible through the GUI project will not allow for successful identification. Use MotorWare to adjust the scaling and use the updated proj_lab## for best ID.
    - It should be noted that a successful Motor ID is not necessary to use FAST or InstaSPIN-MOTION. Motor parameters can be identified by hand measurement or through a motor datasheet and updated in the user.h file for compile. Use MotorWare projects.

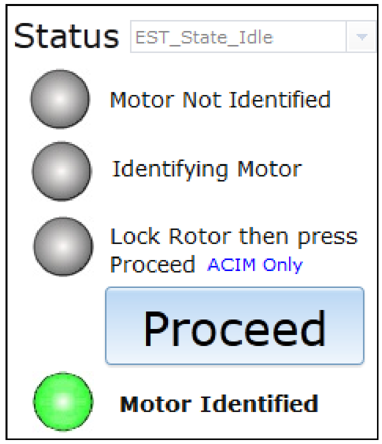
- Click Identify Motor to begin the process
    - You will note that the Identifying Motor lamp illuminates, the Status moves to a new state, and the Percentage bar begins to move



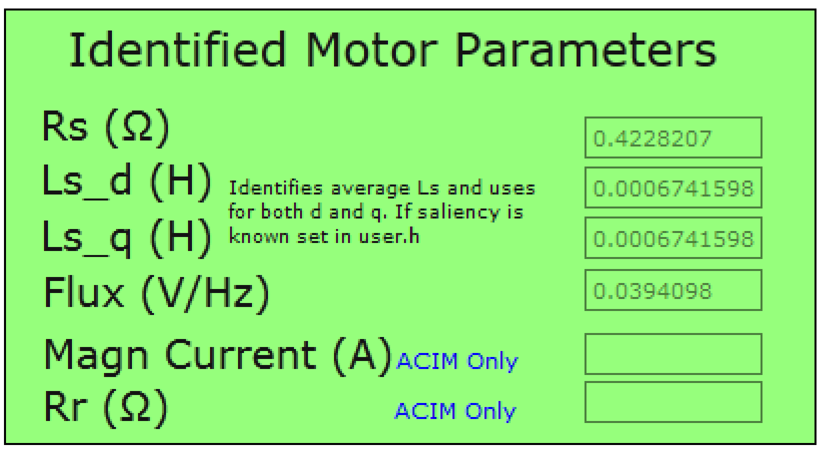


- Status States
  - Defined in sw\modules\est\src\32b\est.h and thoroughly explained in the User's Guide
  - For ACI, the ID will pause at EST_State_LockRotor
    - At this point the Lock Rotor lamp will illuminate
    - Mechanically lock the rotor (using a wrench for small motors) for the duration of the testing and press Proceed
- Identification is completed when the display reads 100%, the Motor Identified lamp lights, and the parameters are displayed



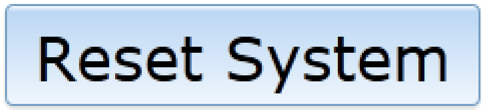


- Example for Anaheim/Telco NEMA17 supplied with DRV8312 Kit
  - Note that your Anaheim/Telco motor may have different values, although the Ls and Flux values tend to be pretty consistent based on our testing. The Rs is typically 0.38 – 0.45 and is temperature dependent.



- Identification Issues
  - EST_State_Error
    - If this is displayed an error has occurred
    - Possible causes
      - Choosing ACIM with a PMSM/BLDC Motor or choosing PMSM/BLDC with an ACIM
      - Incorrect settings, particularly
        - Res Est Current (need large enough to spin during ramp-up)
        - Estimation Frequency (especially for high speed motors or when the scale of the motor signals are much lower than the scale of the hardware)

      - PWM Frequency (may need faster switching for very low inductance motors)
      - Incorrect Rated Flux (for ACIM)
  - To return to initial conditions click

    - If you reset the system during Identification or after an identification error the Est_State Status and the lamps will NOT reset until you next click Identify Motor.

- It is also possible that the motor identification completes without an Error but that the parameters identified are incorrect.
  - This occurs most frequently with very low inductance, high speed PMSM/BLDC motors (such as hobby motors used for helicopters and remote control cars)
  - During identification be sure to watch the motor. From EST_State_RampUp the motor should **begin spinning** up to the Estimation Freq (Hz) and should **continue until the identification process completes.**
    - If it does not spin, then the parameters identified are NOT correct.
  - If motor did not spin through completion, possible causes include
    - Res Est Current needs to be set higher to allow the motor to start-up
    - Estimation Frequency too low, increase in 20 KHz increments
    - PWM frequency too low to properly control current ripple of ultra low inductanc motors (typicaly require 30-100 KHz)
    - Scaling of IQ vs. ADC vs. Motor values are incompatible
      - Will need to move from GUI project to MotorWare project and consult the User's Guide
  - For very low inductance motors the identified inductance Ls may not be precise due to the identification at too low of an Estimation Frequency. Try raising the Estimation Frequency to 40 Hz and 60 Hz and note if it changes significantly.
  - The smallest valid Ls is ~1e-6 H (1uH). Any lower should be considered an incorrect motor ID.

Estimate System Inertia

Inertia is the resistance of an object to rotational acceleration around an axis. The greater the system inertia, the greater the torque needed to accelerate or decelerate the motor. The SpinTAC™ speed controller uses the system's inertia value to provide the most accurate system control.

There is a common misunderstanding that inertia is equivalent to load. Load usually has two components that contribute to the system inertia:

- Load inertia
  - The mass that will spin simultaneously with the motor rotor
  - Examples: direct couplers and belt pulleys with the mass rigidly mounted to the load shaft
- Load torque
  - Appears as an external force applied on the motor rotor shaft
  - Examples: gravity of a mass applied to one side of the motor rotor shaft, distributed clothes in a washing machine drum during the spin cycle; fluid viscosity of a pump.

SpinTAC estimates the load inertia of the system.
Eliminate or minimize the load torque before estimating the system inertia.

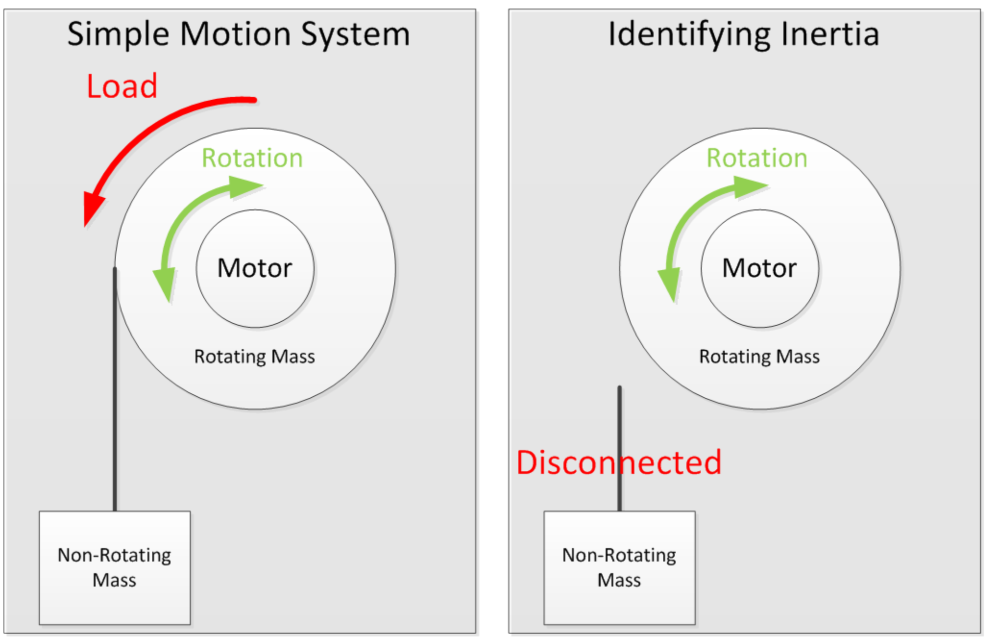


Figure 5: Example of inertia identification in a simple motion system

Figure 5 shows an example of a simple motion system. In this system, the Rotating Mass is rigidly coupled with the Motor. This means that the Rotating Mass rotates along with the motor and is considered as part of the inertia. The Non-Rotating Mass is not rigidly coupled with the motor and is considered as part of the load. During the inertia identification process, this Non-Rotating Mass should be detached from the motor. Table 1 lists common application system configurations during inertia identification for.

Table 1: System configuration for identifying inertia on common applications

Application	System Configuration for Identifying Inertia
Washing Machine	Drum should be attached to motor and free of clothes or water
Pump / Compressor	Motor should be connected to pumping / compressing apparatus, but system should be at lowest possible load. In a compressor, the pressure should be minimized.

Conveyor Belt	Motor should be attached to conveyor, but the conveyor should be free of objects.
Fan	Fan blades should be attached to motor

SpinTAC 1: Startup Tab

The Startup tab is used to identify the system inertia and enable SpinTAC

- You must be in Speed + Torque mode to identify the intertia.
    - o This is the default mode after Motor ID, but you can check this on the “Speed or Torque” tab
    - o If you are loading values from user.h, be sure that you are using the correct ADC Offsets for your hardware. You can also run the motor initially on “Speed or Torque” tab with Offset Recal Enabled to insure you are using the correct values before you begin the InstaSPIN-MOTION Inertia ID.
- Inertia estimation can start with the motor at 0 RPM, any speed, or with the drive Disabled (Start/Stop, Motor Not Running)
- Unlike in the Motor Identification tab where we wanted all coupling removed from the shaft of the motor, for Inertia Estimation we actually want any couplings or sources of inertia – besides variable loads – to be included.

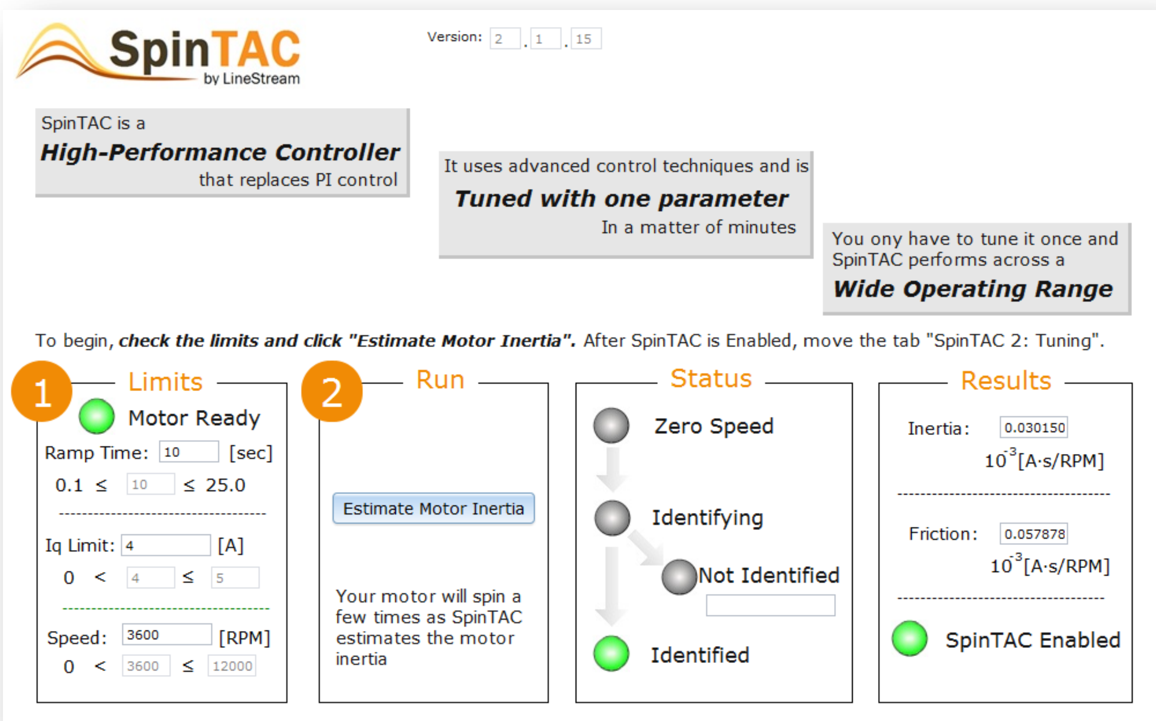


Figure 6 - SpinTAC 1: Startup Tab

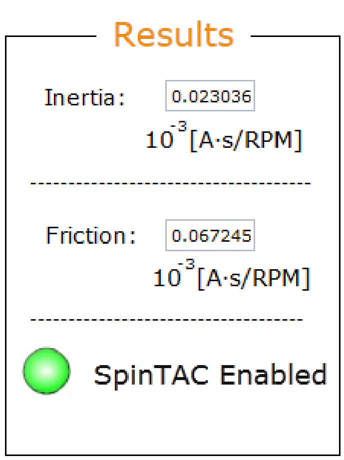
Limits

- Ramp Time
  - Rate at which torque is applied during identification. Typically this should be set to the default. For motors with large cogging force or high friction, this value should be decreased in order to increase the torque more quickly to correctly finish the identification process.
- Iq Limit:
  - Maximum torque command during identification. In general this should be as low as possible while insuring that the motor spins from 0 speed to the Speed Limit. Default is 80% of the MAX CURRENT value from the Motor Identification Tab to cover high cogging torque motors, but for low cogging torque motors this can be lowered to anything that insures the motor spins up to the Speed target
- Speed
  - Target speed where the inertia measurement will be taken. 10% rated speed is typically a good target.
  - Make sure the target speed is within normal speed range (do not try inertia ID at a speed that requires maximum modulation or field weakening)

Run

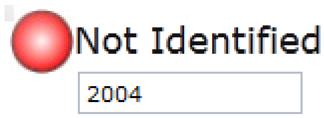
- Click Estimate Motor Inertia to begin inertia estimation

  Zero Speed then Identifying will light, and within a few seconds you should see Identified, your results, and SpinTAC will be Enabled.



Status

- If you receive an error
    - 2003: this is typically due to the ramp rate of the torque controller during ID. Decrease the Ramp Time parameter by 1.0 and try the inertia identification again.
    - 2004: this is due to a time-out, with root cause typically being a lack of start-up current (increase Iq Limit) or too high of a speed target (lower speed target)



Once Identified and the results indicate that SpinTAC is enabled, move to SpinTAC 2: Tuning tab.

Tune the SpinTAC Speed Controller

The industry standard PI speed controller has a number of inherent deficiencies:

- Tuning parameters are interdependent and create tuning challenges.
- PI controllers may need to be tuned multiple times to operate effectively at different speed and load points (referred to as gain staging)

The SpinTAC™ speed controller solves these challenges. SpinTAC™ features Active Disturbance Rejection Control (ADRC), which actively estimates and compensates for system disturbances in real time. ADRC automatically compensates for undesired system behavior caused by:

- Uncertainties (e.g. - resonant mode)
- Nonlinear friction
- Changing loads
- Environmental changes

SpinTAC™ presents better disturbance rejection and trajectory tracking performance than a PI controller, and can tolerate a wide range of inertia change. This means that SpinTAC™ improves accuracy and minimizes mechanical system duress.

The SpinTAC™ speed controller also features a single tuning parameter – bandwidth – which determines the stiffness of the system and dictates how aggressively the system will reject disturbances. Once tuned, the controller typically works over a wide range of speeds and loads.

SpinTAC 2: Tuning Tab

The SpinTAC Tuning tab is used to tune and test the SpinTAC controller.

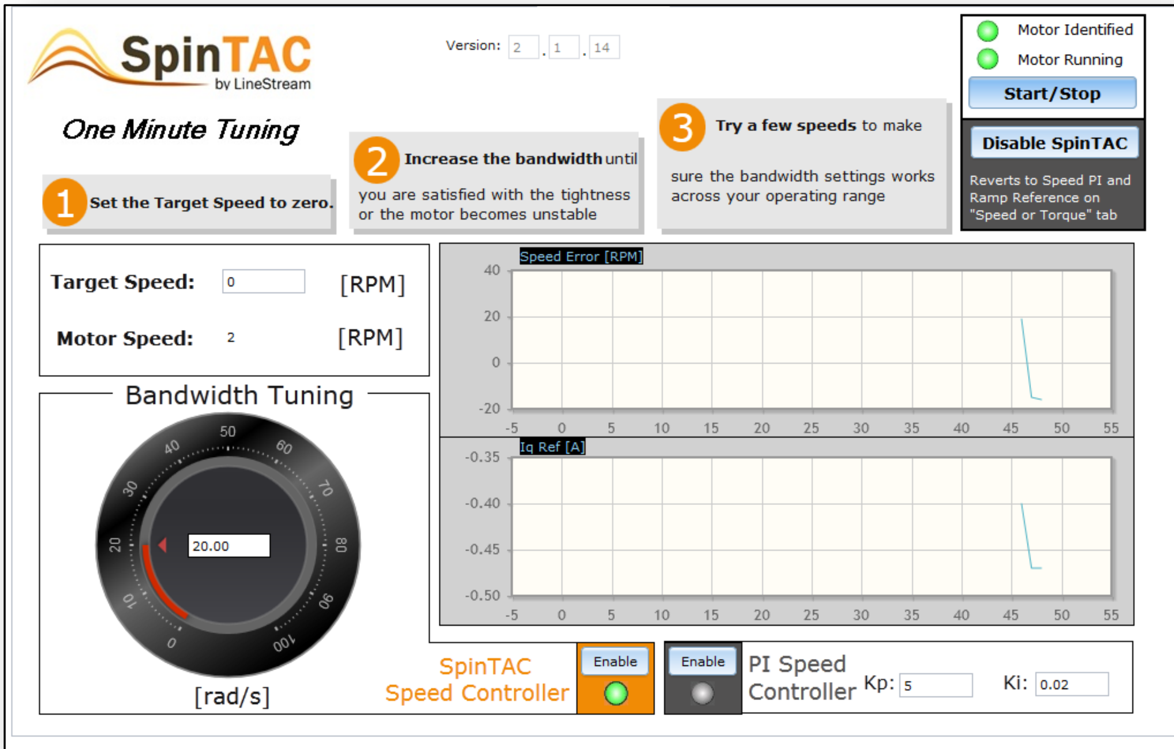


Figure 7 - SpinTAC 2: Tuning Tab

- Start with a Target Speed of 0 RPM. Note that the sensorless FAST-based control system is attempting to control to zero speed.
- Manually rotate the motor shaft to feel the response of the motor at 0 RPM
    - For motors where the shaft is inaccessible, set a series of speed references and watch the response of the motor to the changing speed references

- Increase the bandwidth to give a stiffer response, without becoming unstable. This should be your upper limit.

Compare the Speed Error (RPM) in Figure 8 to the Speed Error (RPM) in Figure 9. Notice that the speed Error (RPM) in Figure 9 is smaller. The bandwidth has been increased so that the controller has a stiffer response. At the higher bandwidth, the system will respond more aggressively to disturbance, and will more quickly compensate for system disturbances.

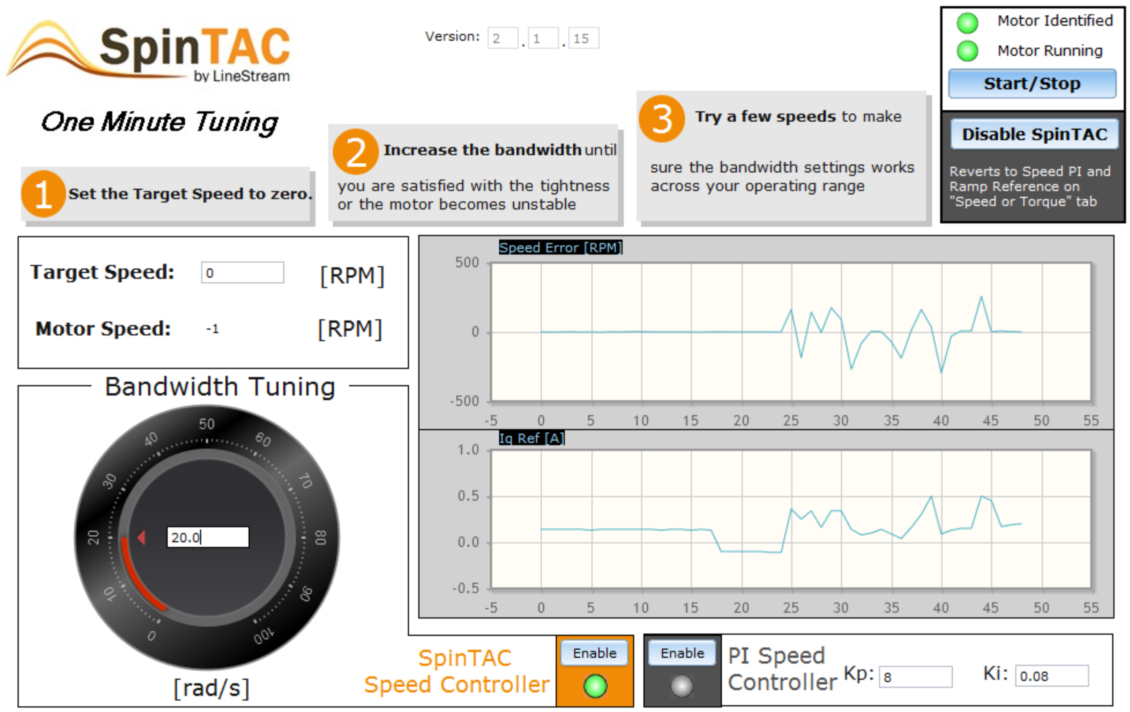


Figure 8 - Speed Error at Zero Speed with Low Bandwidth

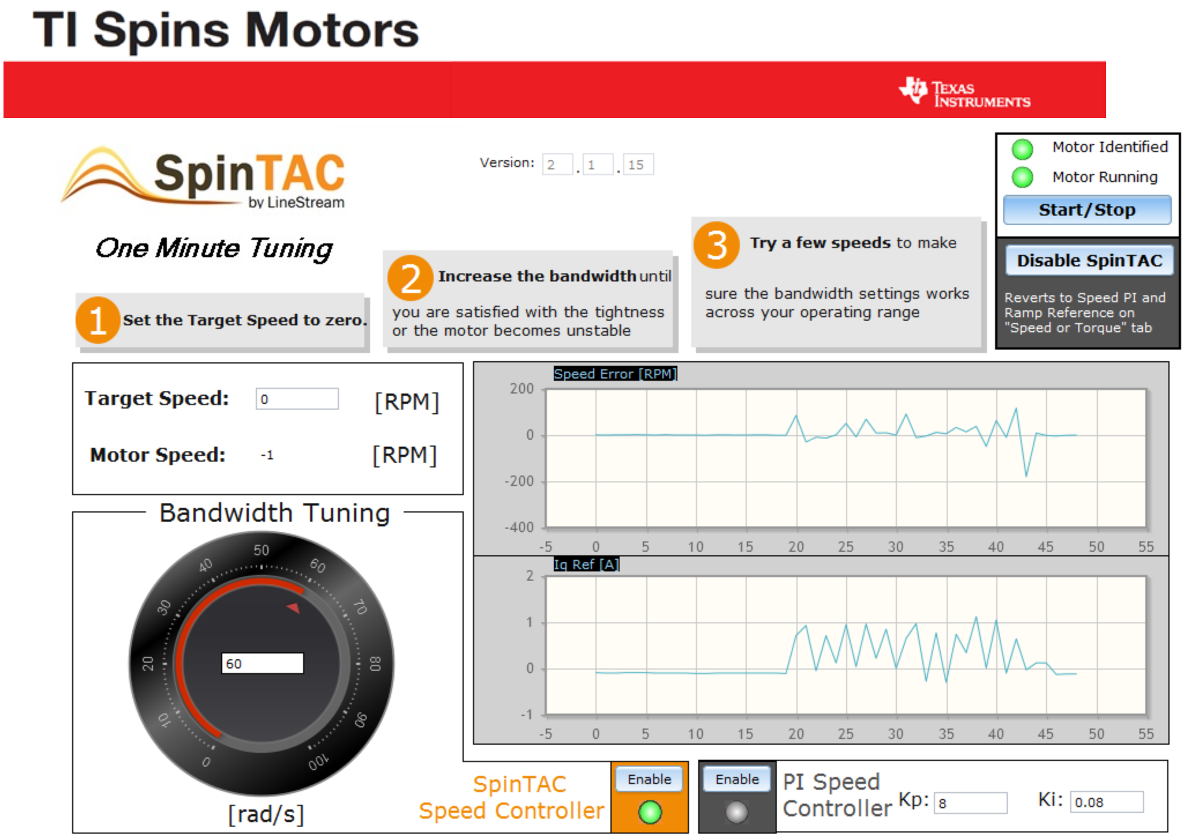
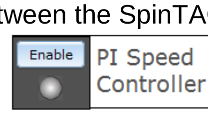


Figure 9 - Speed Error at Zero Speed with High Bandwidth

- Try different speeds to see if the bandwidth needs to be decreased to meet dynamic constraints.
  - o Some bandwidths will work very well at zero speed, but will become unstable at higher speeds. If this is observed, decrease the bandwidth by 10-15% and repeat the testing.

Compare SpinTAC Tuning to PI Tuning

Now you should compare the tuning process for the SpinTAC speed controller with tuning a PI speed controller. The default Kp and Ki gains have been provided. In general, these defaults allow stable operation over a wide range of motors. However, DO NOT consider the speed controller tuned. You will need to tune the speed controller based on the system inertia and and the required dynamic performance of your system, considering different speeds, motions, loads, etc.

- To switch between the SpinTAC and PI controller, buttons have been provided.
  - o Click 

to enable the PI speed controller
  - o Click 

to enable the SpinTAC speed controller
  - o NOTE: it takes approximately 3 seconds to switch between the two controllers
- Set the maximum speed reference to your motor to see if the default Kp and Ki are stable for your motor
  - o If you notice oscillation, these gains should be reduced in order to run in a stable condition at this speed.
- Try running the motor at a couple of different speed points in order to judge how well your PI controller is tuned
  - o If the motor is not being very responsive to speed changes, than the Kp and Ki should be increased.

- The InstaSPIN-FOC and InstaSPIN-MOTION User Guide includes a section on Tuning Regulators. This section provides a detailed process for calculating PI gains for your system.

You should notice that the SpinTAC speed controller takes much less time to achieve the optimal tuning setting for the entire speed range than the PI speed controller.

Test Motion Trajectories

SpinTAC™ provides a velocity profile generator that computes the fastest path between Speed A and Speed B, accounting for the user defined acceleration and jerk bounds. Jerk represents the rate of change of acceleration. A larger jerk will increase the acceleration at a faster rate. Steps, or sharp movement between two speeds, can cause systems to oscillate. The bigger the step, the greater this tendency. Control over jerk can round the velocity corners, reducing oscillation. As a result, acceleration can be set higher. Controlling the jerk in your system will lead to less mechanical stress on your system components and can lead to better reliability and less failing parts.

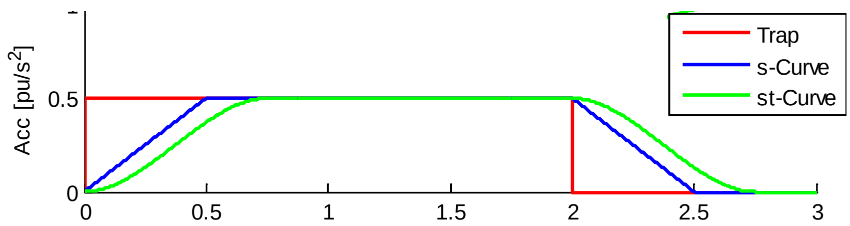


Figure 10 - Motion Curve Comparison

SpinTAC supports the following curve types:

- trapezoidal ramp
  - constant acceleration
- advanced s-curve
  - smoother starting and ending accelerations
- advanced st-curve
  - Linestream Proprietary
  - Features a continuous jerk to provide additional smoothing of the trajectory.
  - The smoothest motion, which is critical for systems that are sensitive to sharp acceleration changes.

Figure 9 compares the acceleration profiles for trapezoidal, s- and st-curves. Notice the smooth acceleration of the st-curve.

SpinTAC 3: Motion Tab

The SpinTAC 3: Motion tab allows you to try different motion trajectories (move from speed A to speed B) with the SpinTAC speed controller.

The SpinTAC trajectory planning feature enables easy creation of multiple motion states based on events, time, speeds, etc. This feature is covered in the InstaSPIN-FOC and InstaSPIN-MOTION User Guide, and is included in the MotorWare projects.

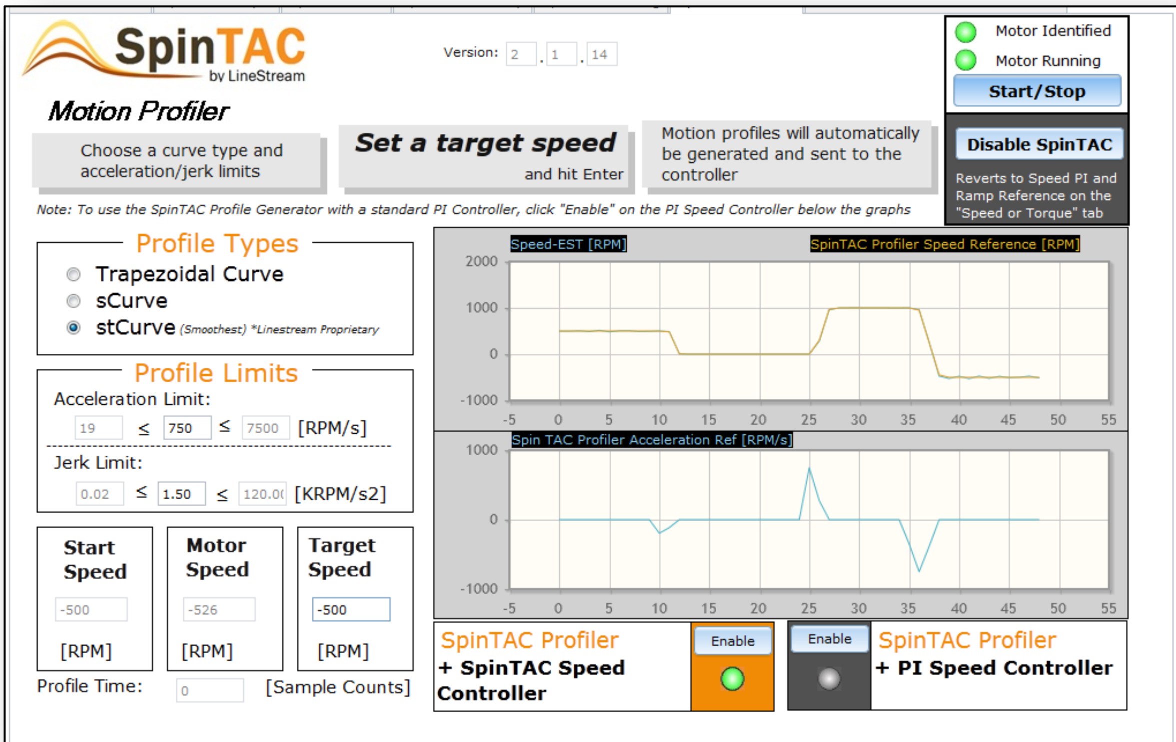


Figure 11 - SpinTAC 3: Motion Tab

Profile Types

- Select the motion profile for your application. For most applications the st-curve is the optimal motion profile.
  - Trapezoidal –Constant acceleration
  - s-curve – Smooth acceleration
  - st-curve – Smoothest acceleration with continuous jerk (optimal choice for most applications)

Profile Limits

- Acceleration Limit will be the maximum acceleration/deceleration used in the profile. Must fall within the low-high bounds displayed.
- Jerk Limit will be the maximum jerk allowed in the profile. Must fall within the low-high bounds displayed.

Speed Settings

- Start Speed: Speed at start of profile (the current speed as you enter a profile)
- Motor Speed: Real-time speed as estimated from FAST
- Target Speed: Target speed for end of profile
- Profile Time: The calculated number of control cycles to execute the Start to Target profile

Experiment

Trying some of the extreme settings will give you a feeling of how the parameters influence the curves. For example, the following are recommended for use with the DRV8312 and the Telco NEMA17 motor. Try all curve types and positive to negative speed.

- Large Acceleration 7000 + Large Jerk 100
- Large Acceleration 7000 + Small Jerk 0.05
- Small Acceleration 20 + Large Jerk 100
- Small Acceleration 20 + Small Jerk 0.05

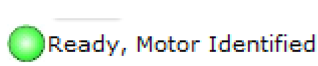
Optional Exercises

Speed or Torque Tab

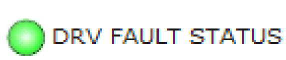
The Speed or Torque tab is used for initial evaluation and tuning of a traditional PI speed and torque controller. Motors can be run in torque only mode (user sets or varies a torque input, controller maintains that constant torque) or speed + torque mode (user varies speed input, speed controller commands necessary torque across acceleration and load to maintain speed).

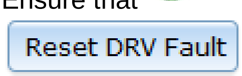
Figure 12 - Speed or Torque Tab

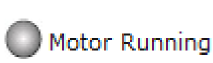
Status Lamps

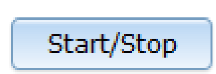
- Ensure that 

is GREEN, else return to Motor Identification Tab, RESET SYSTEM, and re-identify

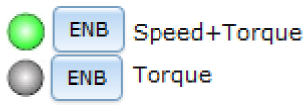
- Ensure that 

is GREEN, else return to Motor Identification Tab to 

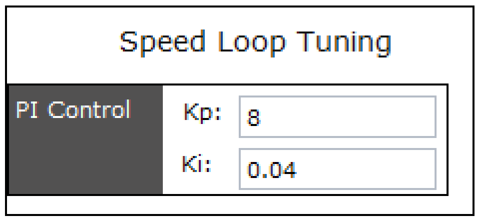
- 

will turn green once the drive is enabled using 

Mode Options & Lamps

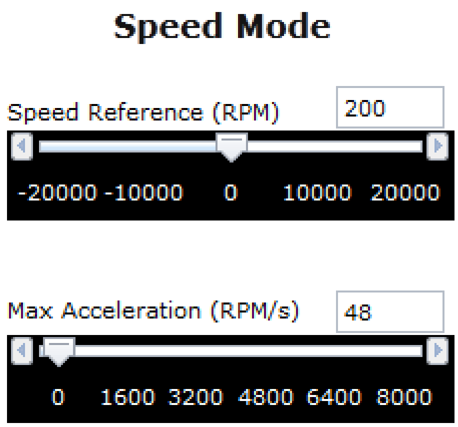


- Speed + Torque: Enables the ROM implementation of FAST used in an InstaSPIN-FOC torque controller with cascaded speed controller
    - Speed Controller

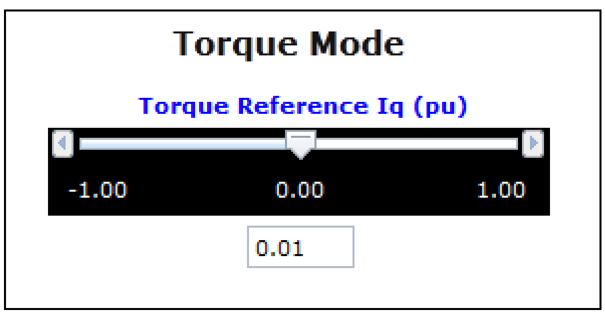
    - Default Speed Controller is the PI included in ROM as part of InstaSPIN-FOC
        - The Kp and Ki gains are calculated from user.h settings
            - Kp = 0.02 * Motor Max Freq * Motor Max Current (pu)
            - Ki = 2.00 * Motor Max Freq * ctrlPeriod_sec * Motor Max Current (pu)
        - Do NOT consider the speed controller tuned. This is a simple “rule of thumb” calculation that generally allows stable operation over a wide range of motors
            - The default values provided by this “rule of thumb”
                - Are typically too low for larger inertia motors
                - Are typically too high for smaller inertia motors
        - User will need to tune the speed controller based on their system inertia and required dynamic performance, often across different speeds, motions, loads, etc.
            - As a general example, with the DRV8312 kit and motor
                - For very low speed operation with load, the gains need to be increased; try some values like 3x
                - For very high speed and into field weakening the gainsneed to be decreased, try some values like /4
        - The User’s Guide includes a speed control implementation section.
    - The user input into the control system is a SpeedRef command in +/-RPM along with a maximum acceleration rate in RPM/s
        - max speed and acceleration (Hz, Hz/s) is limited in user.h and dependent on the motor, power supply, load, etc.

        - Higher speeds may be possible with your motor
        - The 20 KRPM and 8 KRPM/s noted in the GUI may NOT be possible with your motor



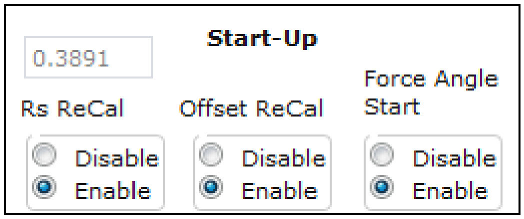
    - The output of the speed controller is an Iq torque command in per unit [-1.0 to 1.0], with 1.0 representing the Max Current setting on the first tab.

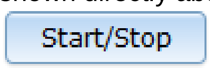
- Torque: Enables the ROM implementation of FAST used in an InstaSPIN-FOC Torque controller WITHOUT any speed controller capability
    - The input into the control system is a TorqueRef command in per unit [-1.0 to 1.0]

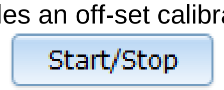


    - **Caution: When in Torque mode you are commanding a constant torque. You should always apply a load to the motor or you risk even a small constant torque command creating a motor that quickly goes as fast as mechanically possible, which can create a fault and damage the inverter, motor, or user.**
    - When in Torque Mode, even though the speed controller output is no longer an input to the control system, a **non-zero, sign matching** target SpeedRef must still be entered for the direction and ForceAngle Start functionality. Ex:
        - Torque of + 0.05 pu, Speed Reference = +200
        - Torque of -0.05 pu, Speed Reference of -200

Start-Up Options

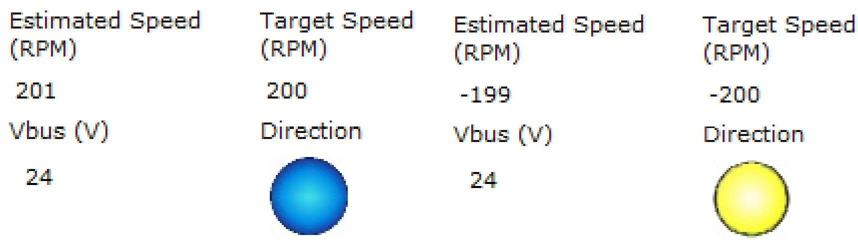


- Rs ReCal
  - Enables an ~8 second re-identification of the Stator Resistance (existing value shown directly above) each time the drive moves from disabled to enabled using 

  - Can be disabled if you just performed an ID
  - May be enabled periodically to view changes in Rs
  - Enable at least once if the motor parameters were loaded from the user.h file
  - See User's Guide for details of Rs On-Line (continous run-time) re-estimation
- Offset ReCal
  - Enables an off-set calibration of the ADC channels when the drive is enabled using 

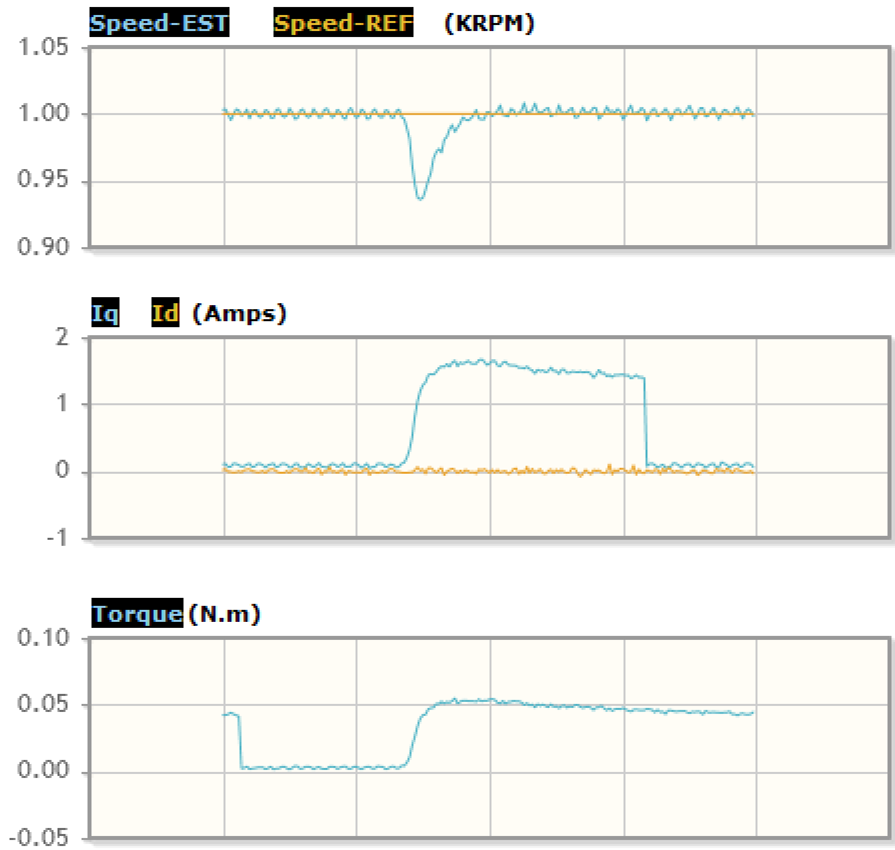
  - After the first time Offset ReCal runs (including during Motor ID), the values are stored for re-use, but it is recommended you enable when possible
  - If loading from user.h be sure your offset values are also stored. If not, enable Offset ReCal at least once
- Force Angle Start
  - Enables a full torque start-up from zero speed routine before the FAST based feedback is fed to the FOC torque system. The frequency at which FAST takes over is discussed in the Identification Settings – Motor Max Freq section.
  - Typically Enable when first starting from zero speed
  - Disable after start-up if you plan to move through zero speed
  - Details on the Force Angle feature and how to calculate and set acceleration and Force Angle to FAST transition points are covered in the User's Guide

Display Panel



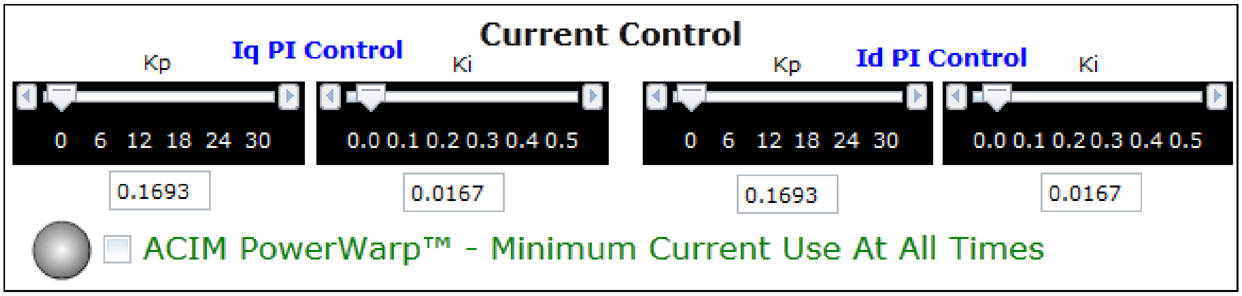
- The Display Panel shows
  - Estimated Speed (RPM) from FAST
  - Target Speed (RPM) going to the speed controller (output of the motion profile)
  - Voltage Bus (V)
  - Direction: Blue (+) or Yellow (-)
    - Note, sometimes you will see the direction lamp erroneously change color. This is a bug in the GUI tool and not the direction variable that is coming from FAST.

Graphs

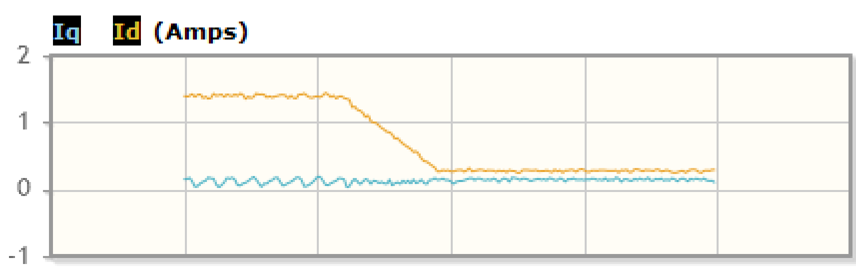


- The Graphs display real-time data on
  - Speed-EST vs. Speed-REF (KRPM)
  - Iq (Torque Component) vs. Id (normally 0) (Amps)
  - Shaft Torque (Newton Meters)
- The graphs automatically rescale to fit the dataset

Current Control



- The current (Iq and Id) PI controllers can be changed at any time
- The Kp and Ki settings are determined from the motor parameters and other scaling and settings in user.h. The formulas used are documented in the User's Guide.
- The values are set slightly soft to give good performance over a wide variety of hardware and motors. To stiffen the control, increase the values of Kp and Ki. At very high frequencies (speeds of 1 KHz+) stiffer control up to multiples of 4x the original values may be required.
    - This 4x is NOT required when using MotorWare projects that take advantage of the angle compensation algorithm.
- You may choose to run specific step response tests to ideally tune for the specific current dynamic performance you require
- ACIM PowerWarp is a mode for three phase induction (asynchronous) motors only. When enabled it will dynamically adjust the Id current to the lowest possible while still maintaining tracking of the rotor and good speed control, providing best efficiency for variable load applications. This will occur only as high torque is not required, in what is known as "low or partial load" operating conditions. It also is a way to automatically do Field Weakening on induction motors as the Id current will be reduced to meet speed target while still maintaining stability.



    - When in PowerWarp mode - because the magnetic field is reduced - as torque is required (load increases) the dynamic response of the system will be lower (it will take longer to react and provide the needed torque vs. PowerWarp disabled). If the load disturbance is great enough it is possible that the motor will stall, but assuming it is not an overload condition FAST will continue to track the angle and the motor will recover and reach the speed reference once again.
    - If Enabled for a synchronous (BLDC/PMSM) motor there will be no change to the control system or to performance

Speed Profiles Tab: Evaluate Controller Performance Using Standard Motion Profiles

The Speed Profiles tab is provided with two different state machine motion profile examples to enable *evaluation* of common variable speed applications. The first is a Staircase Spin Profile, and the second is an agitation profile with speed reversal.

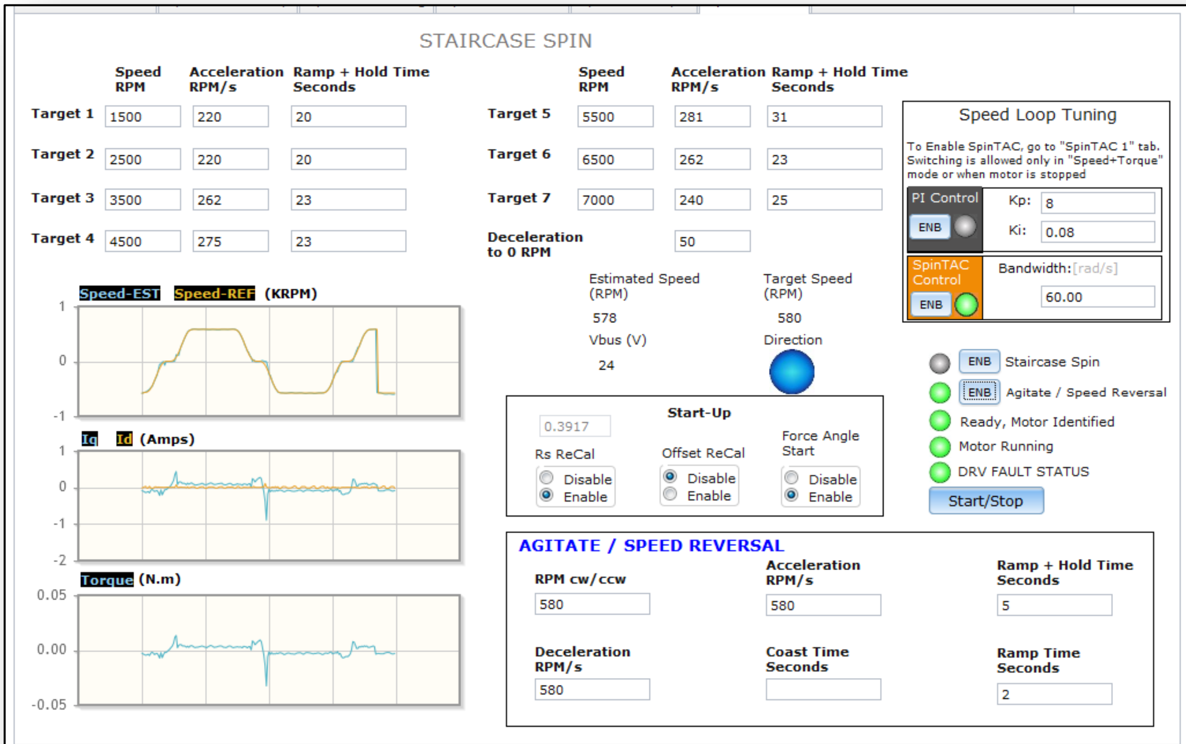
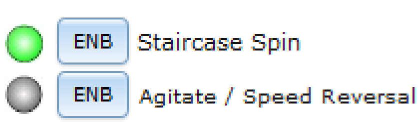
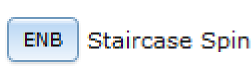


Figure 13 - Speed Profiles Tab

Staircase Spin



- Enabled by selecting 

TI Spins Motors

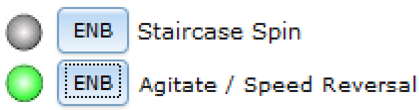
STAIRCASE SPIN

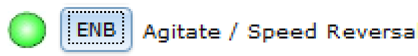
	Speed RPM	Acceleration RPM/s	Ramp + Hold Time Seconds		Speed RPM	Acceleration RPM/s	Ramp + Hold Tim Seconds
Target 1	1500	220	20	Target 5	5500	281	31
Target 2	2500	220	20	Target 6	6500	262	23
Target 3	3500	262	23	Target 7	7000	240	25
Target 4	4500	275	23	Deceleration to 0 RPM		50	

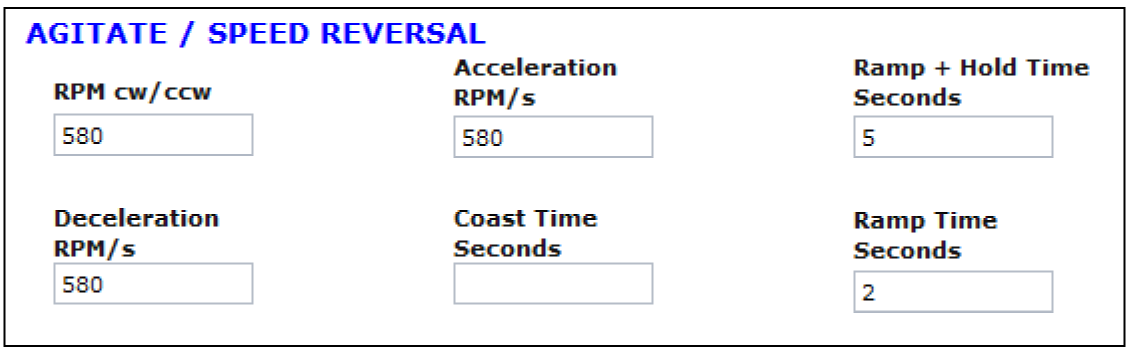
- This Motion Profile sets a SpeedRef (RPM) to the controller using the following logic
  - 0 RPM Start ramp to Target 1 Speed @ Target 1 Acceleration
  - At Target 1 Time, ramp to Target 2 Speed @ Target 2 Acceleration
  - After additional Target 2 Time, ramp to Target 3 Speed @ Target 3 Acceleration
  - After additional Target 3 Time, ramp to Target 4 Speed @ Target 4 Acceleration
  - …
  - After additional Target 7 Time, ramp to 0 RPM at Deceleration
    - **Motors at high speed are creating a large Bemf voltage. Caution should be taken during Deceleration to do so slowly or an over voltage situation will occur resulting in a fault and possibly damaging the inverter or the motor.**
- Speed can be positive or negative
- Acceleration/deceleration should always be entered as a positive number
- Capability is heavily dependent on the motor, mechanics, inertia, power supply, and load. Do not assume that a motor will track any motion profile that is entered.
- Disable RsReCal for proper operation of the counters during Speed Profiles
  - The timers start when the run-time controller is in on the on-line state. If Enabled, RsReCal is run during the on-line state and the first ramp + hold time will be offset by ~8 seconds
- Enable Force Angle Start every time you start the drive
  - After Start-Up
    - Enable/Disable does not matter for a profile that does not cross zero speed
    - Disable if your motion includes transitions straight through zero, or short transition durations at zero speed
    - Leave Enabled if your motion includes longer periods near and at zero speed. FAST continues to track well moving through zero, and with larger Bemf motors and good sensing it can hold zero for some amount

of time. Evaluate if the performance Enabled / Disabled is better for your application.

Agitate / Speed Reversal Mode



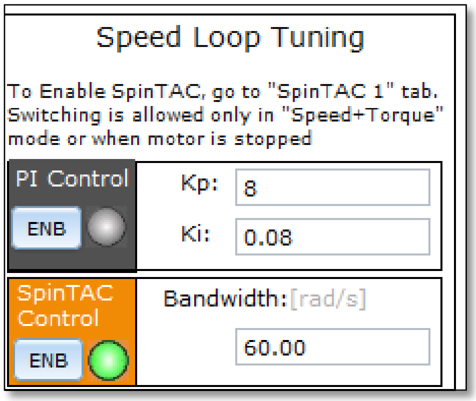
- Enabled by selecting 



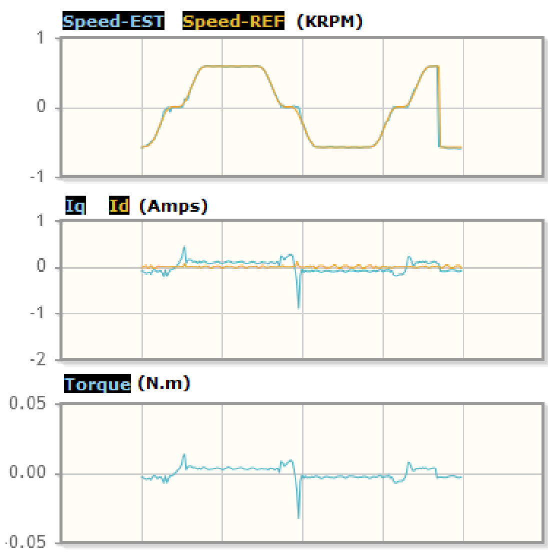
- This Motion Profile sets a SpeedRef (RPM) to the controller using the following logic
  - 0 RPM Start, ramp to RPM cw/ccw @ Acceleration
  - After Ramp + Hold Time
    - If desired a Coast Time (motor freewheels) can be entered
    - Typically 0
  - After Coast Time
    - If desired a Ramp Time can be entered
      - Ramp towards 0 RPM @ Deceleration
      - If time remains, Hold @ 0 RPM
    - If 0 is entered for Ramp Time, or a value lower than necessary to bring the motor to 0 speed due to inertia, load, or deceleration setting, the motion profile will immediately repeat in the opposite direction
  - Repeat in opposite direction (- RPM cw/ccw)
  - Repeat forever
- RPM can be positive or negative (depending on initial starting direction)
- Acceleration/Deceleration should always be entered as a positive number

- Capability is heavily dependent on the motor, mechanics, inertia, power supply, and load. Do not assume that a motor will track any motion profile that is entered.
- Disable RsReCal for proper operation of the counters during Speed Profiles
  - The timers start when the run-time controller is in on the on-line state. If Enabled, RsReCal is run during the on-line state and the first ramp + hold time will be offset by ~8 seconds
- Enable Force Angle Start every time you start the drive
  - After Start-Up
    - Disable if your motion includes transitions straight through zero, or short durations at zero speed
    - Leave Enabled if your motion includes longer periods near and at zero speed
    - FAST continues to track well moving through zero, and with larger Bemf motors and good sensing it can hold zero for some amount of time. Evaluate if the performance Enabled / Disabled is better for your application.
    - Low speed agitation with load is one of the most challenging control scenarios. In a real application the currents are often quite large, causing Rs values to increase dramatically. Please see the User's Guide for discussion and implementation of on-line continuous Rs recalibration.

You can switch between PI and SpinTAC control to evaluate performance:



- Examine the PI speed performance. Introduce a disturbance (such as grabbing the motor shaft). Notice how the PI controller responds.



- Enable the SpinTAC Profiler
- Examine the SpinTAC controller performance. Introduce a disturbance (such as grabbing the motor shaft). Notice how the SpinTAC controller responds quicker and with less overshoot to the introduced disturbance.

Advanced: Field Control & Modulation Tab

The Advanced Field Control tab is meant to be used in conjunction with the other tabs, typically Speed+Torque, Staircase Spin (for field weakening), or Torque Mode (for field boosting). Once operating in those modes you can use this tab to further control the field.

This should only be used with those of knowledge of field weakening. Using field weakening improperly may permanently damage your motor and/or inverter.

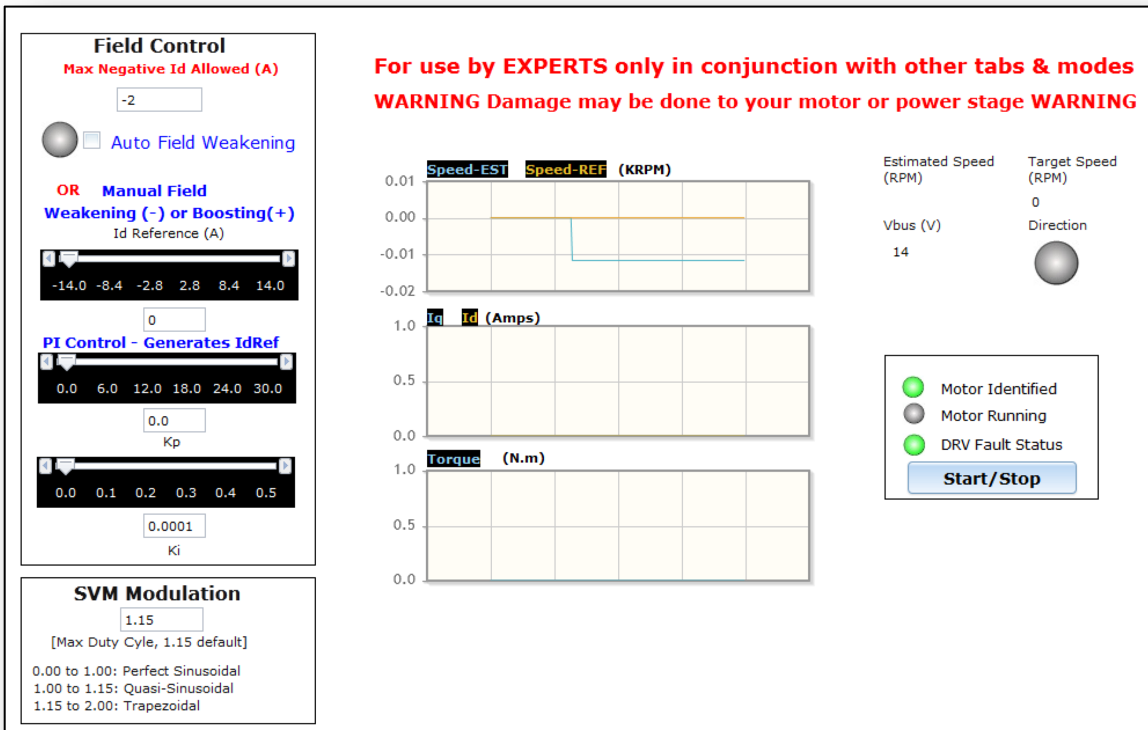


Figure 14 - Advanced: Field Control & Modulation Tab

Field Control

- Max Negative Id Allowed
    - Safety feature active ONLY for Automatic Field control
        - BE CAREFUL USING MANUAL CONTROL
    - Too large of –Id can demagnetize a motor. Insure that sqrt(Id^2 + Iq^2) < Irated or your motor will demagnetize.

- Auto Field Weakening
    - Enabling will increase the –Id until Estimated speed reaches Target speed, or –Id reaches Max Negative Id Allowed
        - In low voltage motors you may see the –Id go straight to the maximum allowed. If this is the case you may want to use manual field weakening or investigate alternative automatic field weakening algorithms.
    - Once Auto Field Weakening applies a –Id value,
        - As you reduce your speed below the field weakening RPM you will need to manually increase your –Id towards 0.
        - This should be done incrementally and slowly, not all at once or you risk an over voltage condition on the bus
    - This is just one implementation of automatic field weakening to be shown as an example. There are many additional ways to implement field weakening that may work better for your application.
- Manual Field
    - Always active/available if Auto Field Weakening is NOT selected
    - This is the standard way field weakening is usually performed, where a look-up table is created with a set of Id values for a given speed and load.
    - Id Reference manually sets the end Target Id (typically zero for standard FOC) to be controlled
        - (-) makes for field weakening, allowing higher speeds but less torque
        - (+) makes for field boosting, allowing for higher torque for induction machines
- PI Control
    - Controls the gains of the controller whose output is the Id reference
    - When an Id is entered you do not want this value to be a step input to the control system or an instability may occur
    - The PI controller lets you tune the response of adding field control
- Note that during field weakening the speed loop often becomes a larger control challenge. You will need to re-tune (soften) the speed controllers to avoid against oscillation. Review the User's Guide for guidance on tuning PI and SpinTAC speed controllers.

SVM Over-Modulation

- Use caution changing this setting when the drive is enabled, especially if the duty cycle is operating near the modulation limit. Recommend to change when the drive is disabled to avoid faults.
- This setting changes the maximum duty cycle and hence maximum speed.
    - 0.00 to 2.00 valid range

        - Entries < 0.00 will clip to 0.0, >2.00 will clip to 2.00 internally (will still display the entered value)
    - Typical values:
        - 0.8667 = Standard Space Vector with no over modulation, 87% duty cycle
        - 1.15 = Space Vector with slight over modulation to achieve 100% peak duty cycle
        - 2.00 = 100% duty cycle with trapezoidal waveform
    - Note: in MotorWare a different scale range is used for modulation indices and an updated SVM module is available. This uses a scale of 1.0 for pure sinewave and 1.333 as the maximum trapezoidal modulation.
- The User's Guide includes detailed description of this topic.

Example performance

- DRV8312-69M-KIT with provided Telco NEMA17 BLDC

Modulation	Max RPM Typical (no load)
0.1	500
0.5	2570
0.85	4400
1.15	5700
1.50	6155
1.75	6285
2.00	6360

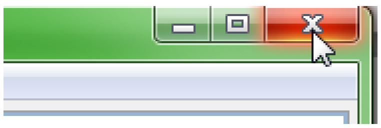
Field Weakening	Max RPM Typical (no load, 1.15 Modulation)
0	5700
-0.2	5850
-0.5	6090

-1.0	6500
-1.50	6995
-2.0	7540

Shutting Down

- Stop the Drive

- Close the GUI



- Remove bus power

WARNING

Do not close the GUI until the drive has been stopped. Failure to do so will leave the program running or put the processor into an unknown state, causing the system to continue to draw current, possibly damaging the controlCARD, board, host computer, motor and posing a fire hazard. After proper shut-down always disconnect the power supplies and remember that **capacitors are charged and will take time to dissipate!**