

Configuration tuning and running of Servostar drives – *Configuration* is the basic stage of setting parameters so that the drive performs the functionality we would like it to have. Usually this is not an iterative stage and once we know the application and are familiar with the drive this is a straight forward thing to do.

Tuning relates to parameter values which cannot be figured out just by knowing the application and we do need an iterative session of changing parameters value and re testing, this is called tuning, and the process is made much easier if we have guidelines for the tuning process.

Running the drive finally is the actual final proof testing of what we have made in the configuration and tuning phases.

In certain cases we have only limited information about the application (motor, etc.), and we need to perform additional tests to figure out certain configuration values.

The software which helps us in performing the above operations are:

- Terminal – talking directly to the drive.
- MotionLink – performs conversions and arranging of data for the user convenience, including a graphic display (program is a little old).
- Labview – it is possible to create UI and serial communication incorporated in user screens.
- FutureLink – yet to come.

Q1: Based on your personal professional or every day life experience give a few examples of configuration and tuning processes. Can you give an example for a process which can enter both categories, depending on the conditions.

Mnemonics – the vocabulary of Servostar (~307 for 4.1.4), the command LIST returns the complete mnemonics set. There is a distinction between variables and commands, variable sets or displays a quantity, command performs an action.

Q2: using the [online reference manual](#) determine which of the following are variables and which are commands? what is the difference between EN and J, in terms of the distinction between command and variable? what is the difference between VE and OPMODE in terms of this distinction.

EN, J, GP, V, VE, OPMODE

Personality – the variables which define the drive behavior when it is powered. All variables which are stored in the EEPROM are personality variables.

NIE – Not In Effect, variables which are currently not effecting the behavior of the drive. There are a lot of them for every given application.

Q3: give at 10 personality variables, which relate current mode operation ([use Excel sheet](#))

Q4: for COMPMODE=4 find 5 variables which are not in effect – NIE, are there more ?

Model variables and Design procedures – variables which describe a quantity which is outside the drive, and are used by the drive to design its controls. "Design" is a background computing task performed by the drive, sometimes invoked by a special command and sometimes by a change of variable.

VBUS , MLMIN – are model variables used for the design of the current controller.
 CONFIG – invokes the design procedure, for designing a current controller.

On the fly changes – Variables that can be changed without interrupting drive operation.
 BW , KVI – they may invoke a design procedure, that after run will update the drive internal parameters.

Q5*: how do VBUS and MLMIN and effect the current controller gain and why?

Q6: find additional 5 parameters that can be changed on the fly.

Current loop configuration from scratch

The basic 4 quantities

In order to begin running a motor there are 4 quantities which **must** be known (they cannot be measured).
 They are:

MIPEAK – maximal peak current.
 MICONT – maximal continuous current.
 MAX Voltage – maximal bus voltage that can be used.
 MSPEED – maximal mechanical speed rated for the motor.

Without knowing these parameters damage can be caused the motor:

Exceeding MIPEAK – demagnetization of the permanent magnets
 Violating MICONT – motor over heats and insulation melting.
 Exceeding max voltage – motor insulation may give and short circuit can appear
 Exceeding MSPEED – damage to the ball bearings of the motor.

Measuring motor inductance

Motor is not connected to the drive.
 Use an inductance meter preferably at the 1Khz range.
 Connect meter to two motor leads.
 Rotate the motor.
 Take the **minimal** value found within motor revolution.
 Set MLMIN:

$$\text{MLMIN} = \text{Inductance [mH]} \times 100$$

Finding MPOLES with a power supply

After obtaining the 4 basic parameters it is possible find MPOLES or the number of electrical cycles per rev.
 Set a power supply with current limit of:

$$\text{LIMIT} = \text{MICONT}/100/\text{sqrt}(2) \text{ [Amp]}$$

Motor is not connected to the drive.
 Connect two leads to the power supply, the motor will lock.

Rotate the motor one full revolution and count how many "stations" you pass. The number of stations is the number of electrical cycles per motor revolution. To set MPOLES use:

$$\text{MPOLES} = \text{Stations} \times 2$$

Finding MPOLES with ZERO command

This is an alternative method for finding MPOLES without an external power supply. In order for this to work it is required to have:

1. The 4 basic parameters
2. The motor inductance MLMIN (which can also be measured).

The ZERO command applies constant current between phase C and phase B hence locking the motor similar to the test with the power supply. However in order for activating the drive basic configuration should be obtained.

– Send command CLREEPROM and cycle power

– Set the basic variables according to their real value (VBUS has to be set according to actual bus voltage).

```
MIPEAK=
MICONT=
VBUS=
MSPEED=
```

– Set motor inductance MLMIN

```
MLMIN=
```

– Set initial values for the following variables:

```
MJ=100;           Arbitrary value
MBEMF=20;         Arbitrary value
MBEMFCOMP=0;      Arbitrary value
MPOLES=2;         Arbitrary value
MENCRES=1000;     Arbitrary value
MENCTYPE=6;       Allow all kinds of encoders
MHINVA=1;         Create a legal hall state when no halls
exist
ILIM=IMAX;        Read IMAX and set ILIM
VLIM=VMAX;        Read VMAX and set VLIM
VOSPD=VMAX*1.2;   Compute VOSPD and set it
THERMODE=3;       Ignore motor thermal switch
LIMDIS=1;         Ignore limit switches
```

– Issue a CONFIG command

– Set

```
IZERO=10;          = 10% x MICONT/10 [Amp r.m.s]
```

- Activate the ZERO command by setting

ZERO=1

- Enable the drive
- Repeat the measurement described in previous section ([go there](#))

Measuring motor BEMF

In order to measure directly the BEMF of the motor it is required to move the motor at a fairly constant speed (hand rotation is usually enough).

The MPOLES parameter is required as found in previous tests ([with supply](#) , [with ZERO command](#))

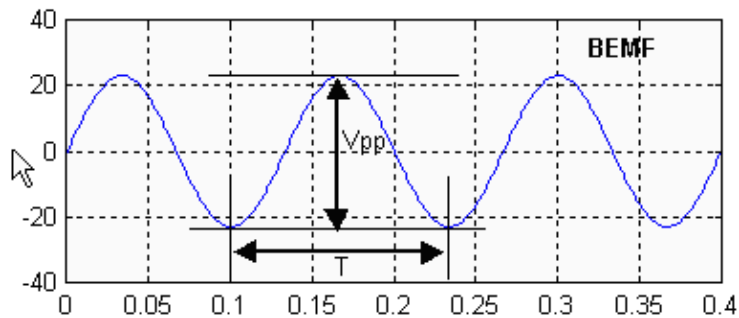
Motor is not connected to the drive!

- Connect two motor leads to an Oscilloscope (grounding one lead is o.k.)
- Rotate the motor and capture the voltage.

Check for:

- Signal is roughly a sin wave (with varying amplitude).
- At least one cycle is with constant amplitude.

The picture which is obtained is:



Using the scope cursors measure "Vpp" and "T" as shown on the plot.

Computing:

$$U_{eff} = V_{pp} / 2 / \sqrt{2} \text{ [Volt r.m.s]}$$

$$Vel = (1/T) / (MPOLES/2) \times 60 / 1000 \text{ [Krpm]}$$

$$MBEMF = U_{eff} / Vel \text{ [Volt r.m.s / Krpm]}$$

Locked rotor current test

In order to test the static behavior of the current loop it is possible to conduct a locked rotor test, in this test the motor is brought to "zero" commutation location and locked either by a vice grip or by a large inertia which practically holds it in place. The internal STEP generator is used to inject short step sequence directly to the current loop. The response is best viewed with a current probe placed on Phase-A.

A basic configuration is required in order to conduct this test and activate the drive. A possible setting is:

MIPEAK=

MICONT=

VBUS=

```

MSPEED=
MLMIN=
MPOLES=
MBEMF=
MBEMFCOMP=0      Disable Bemf compensation
MJ=100;          Arbitrary value
MENCRES=1000;    Arbitrary value
MENCTYPE=6;      Allow all kinds of encoders
MHINVA=1;        Create a legal hall state when no halls
exist
ILIM=IMAX;       Read IMAX and set ILIM
VLIM=VMAX;       Read VMAX and set VLIM
VOSPD=VMAX*1.2;  Compute VOSPD and set it
THERMODE=3;      Ignore motor thermal switch
LIMDIS=1;        Ignore limit switches
CONFIG;          Current loop design
IZERO=25;        = 25% x MICONT/10 [Amp r.m.s] (or higher
for accurate location)
ZERO=1;          Lock rotor
EN;              activate the drive

```

At this stage the rotor is locked either by a vice grip, or by a large inertia, and then the ZERO command is cancelled (ZERO=0). In order to inject the step command directly to the current loop a transparent velocity controller is set:

```

opmode 0
compmode 0
gv=1537
gvi=0
filtmode=0
convert
vh 0 0 0 0 0 0 0 0; cancel the feedback
compmode 3

```

These settings mean:

- VCMD is directly connected to ICMD (in UNITS=0).
- Velocity feedback is disabled

Running the step command:

```

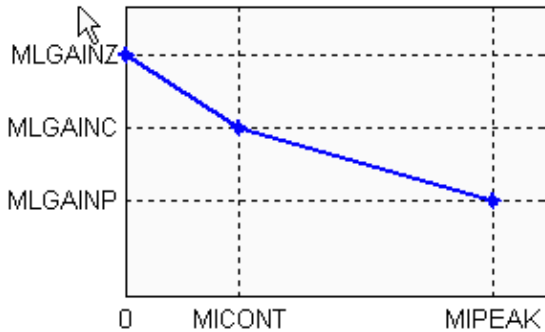
en
step 7 200 7 -200
#delay 50
k

```

Only a short test is needed with short step times since the typical response time of the current loop is 1mSec. The results are best viewed with a DC high bandwidth current probe connected to a scope. In factory configuration the locked rotor test is used for setting the adaptive gains:

MLGAINZ=10
 MLGAINC=9
 MLGAINP=8

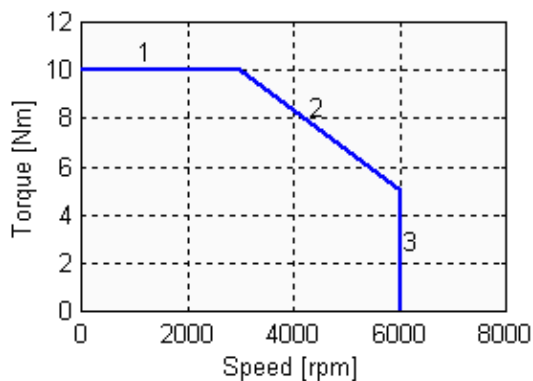
The adaptive gains are needed to compensate for inductance changes which depend on phase current. The above is a typical setting for a general purpose motor, Kollmorgen Goldline and Goldline XT series require a special setting, since they have significant inductance for different current values. The verification is made by applying steps at different amplitudes and observing the step response at the whole range.



The above diagram demonstrates the adaptive gain lookup table with the two linear slopes.

Phase advance and Bemf compensation

Phase advance allows to extend the motor operating envelope to produce higher torque and reach higher speeds, Bemf compensation is used for exactly the same purpose. The optimization of these parameters can only be performed after the motor has been commutated properly, so initial values will be assumed and further discussion will appear later.



Above is a schematic torque velocity curve for intermittent operation (using peak current). The limits consists of 3 parts as is shown in the figure:

1. Current limit of the motor or drive.
2. Bus voltage limit.
3. Maximal speed of the motor.

In actual applications (a) the curve may be smooth with no corners but still follow the general scheme (b) one or two of the segments may reduce to zero. (for example bus voltage may be the only limit) (c) The optimization of phase advance and Bemf compensation allows to push further segment "2" and allow higher torque especially at the high speeds.

Resolver system commutation by ZERO command

Resolver is an absolute sensor so there is only the initial offset to determine in order to operate it correctly. This can be done using the ZERO command. (in order to have an accurate test it is better to use high values of IZERO , upto 100).

If the motor and resolver is aligned correctly the motor will lock at one of the following locations:

$$PRD_{lock} = n * (65536 / (MPOLES/2)) ; \quad n = 0,1,.. (MPOLES/2-1)$$

Obviously one of the locations which is easy to work with is $PRD_{lock}=0$. If there is a deflection from the location, then the motor isn't aligned correctly.

$$MPHASE = - PRD / 65536 * (MPOLES/2) * 360$$

MPHASE should be truncated by 360deg multiples so that: $0 < MPHASE < 359$.

A deflection of MPHASE by less than 5 deg from the zero location usually has negligible effect on performance.

For cases where this doesn't work check out the [next section](#).

Phasing of resolver motor

In order to verify phasing of a resolver motor first check the direction in which PRD increases. If this is not the desired positive direction there are two options to invert it:

Software : set MFBDIR=3 this will invert the treatment of the feedback direction both for commutation and for velocity control.

Hardware : swap between the sin and cos of the resolver, or invert one of the signals (swap high/low). Doing two changes will maintain the initial direction. Inverting the reference will not cause any direction change.

Once the resolver positive direction has been established, establish the right MPHASE using the [previous test](#). try to rotate the motor at torque mode with small torque command (start with T=10 and increase). If the motor locks in place, swap phases B and C add or subtract 180 from MPHASE and try again (make sure ZERO=0 when trying to rotate the motor).

Changing MPHASE from 0 to 120 or to 240 is equivalent to cyclic changing motor leads:

$$A-B-C \implies C-A-B \implies B-C-A$$

swapping two leads is equivalent to inverting the motor direction of rotation

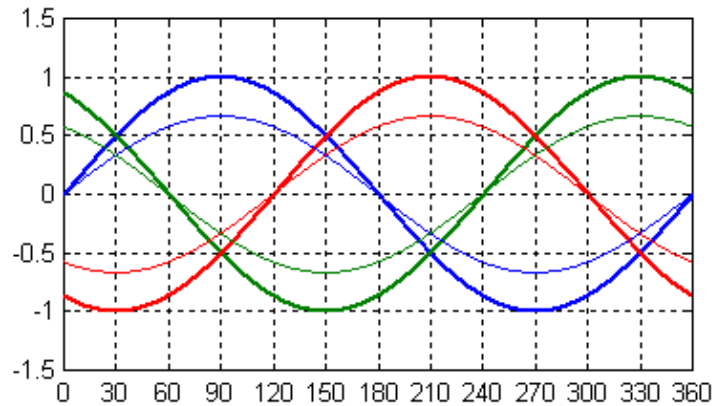
$$A-C-B \implies C-B-A \implies B-A-C$$

Changing within each group can be emulated by MPHASE changes of 120 deg, however to switch between the two groups is not possible by software means.

Various cases of commutation miss alignment
Proper alignment,

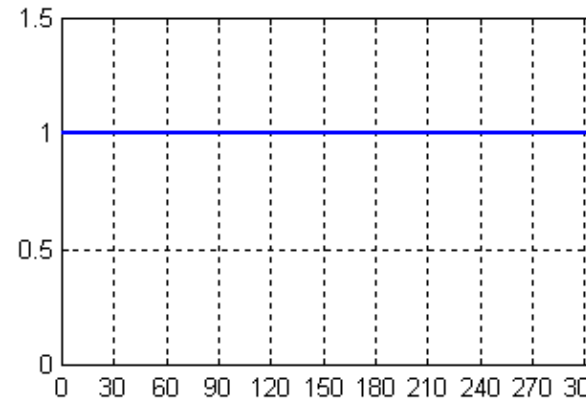
Phase Kt and current plots

Proper alignment and resulting torque ==> nominal torque production

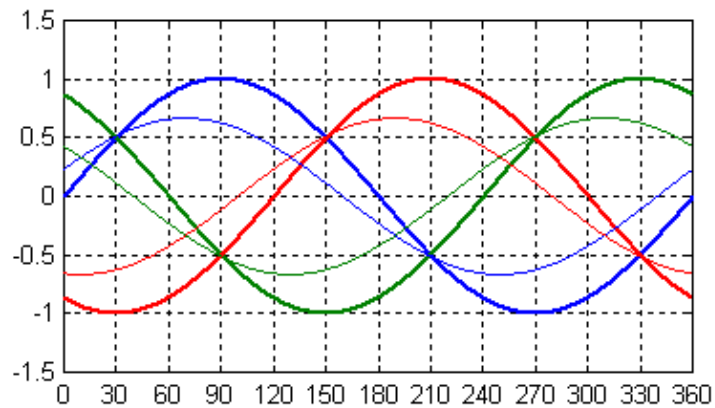


- **Wide line** – phase Kt
- *Narrow line* – Phase current
- **Blue** – phase A
- **Red** – phase B
- **Green** – phase C

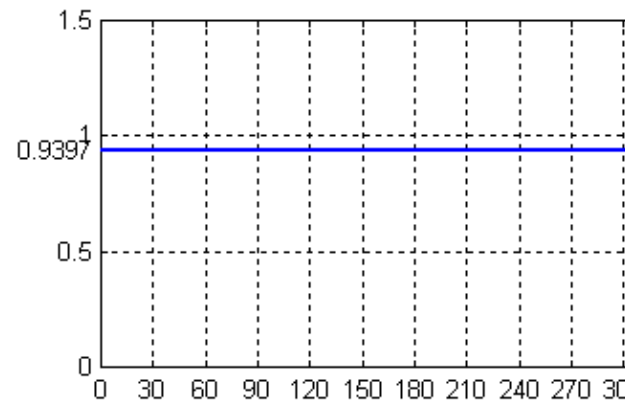
Motor torque production



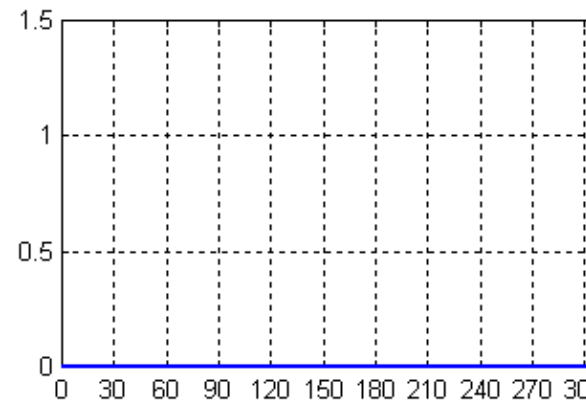
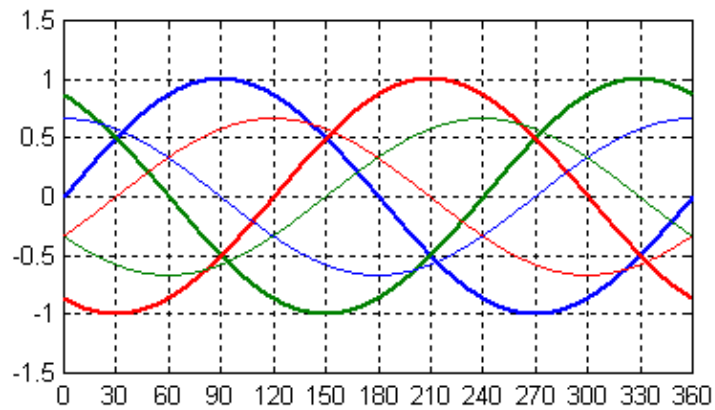
Misalignment of 20 deg ==> torque drop of 6%



- **Wide line** – phase Kt
- *Narrow line* – Phase current
- **Blue** – phase A
- **Red** – phase B
- **Green** – phase C

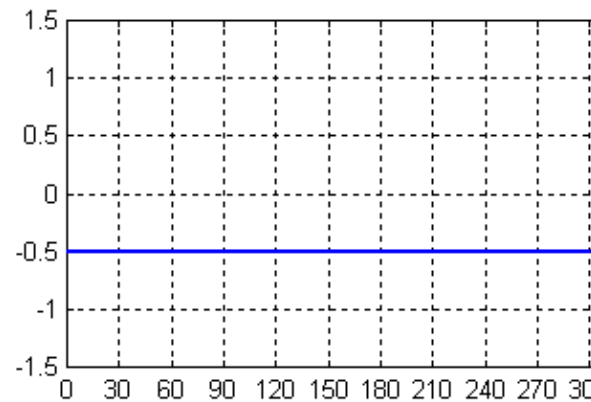
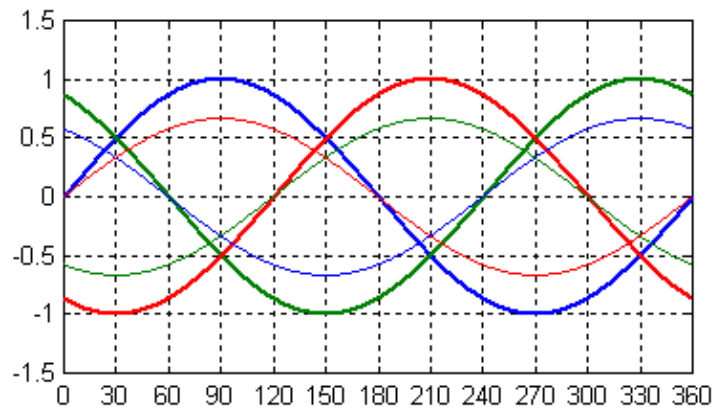


Misalignment of 90 deg ==> no Torque



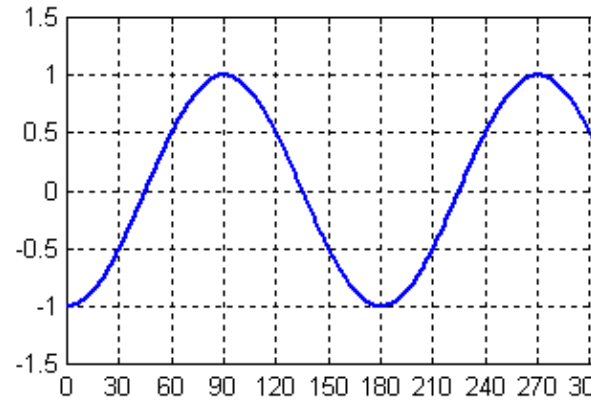
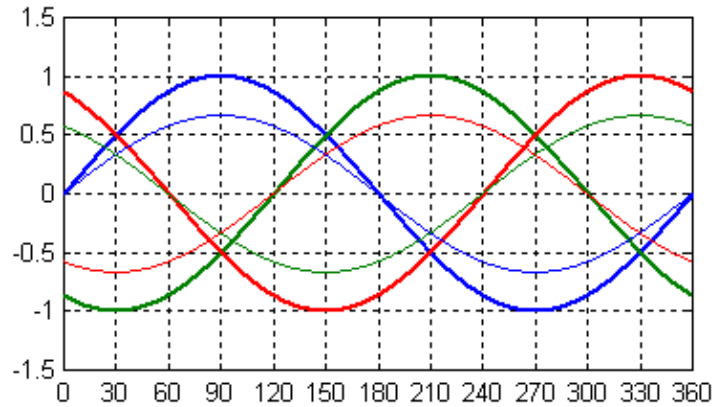
- **Wide line** – phase Kt
- *Narrow line* – Phase current
- **Blue** – phase A
- **Red** – phase B
- **Green** – phase C

Phase cyclic rotation ==> torque is in opposite direction and half in size



- **Wide line** – phase Kt
- *Narrow line* – Phase current
- **Blue** – phase A
- **Red** – phase B
- **Green** – phase C

Phase swap B-C ==> motor locks no rotation



- **Wide line** – phase Kt
- *Narrow line* – Phase current
- **Blue** – phase A
- **Red** – phase B
- **Green** – phase C

Encoder system commutation sequence

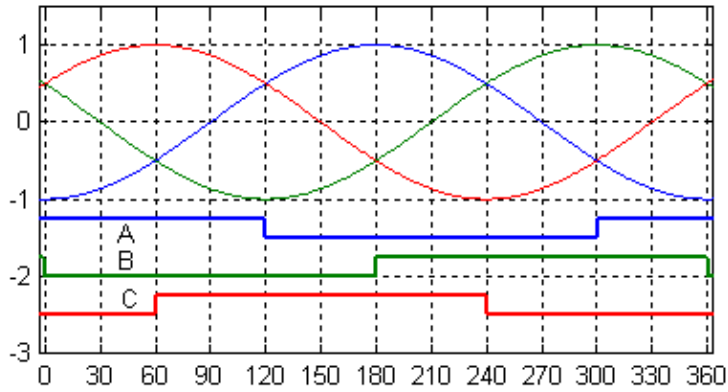
Encoder systems vary with regard to the components included in the system. A and B square wave quadrature signals must be present in order for the system to operate. Since A and B produce only incremental reading (we can only know our position relative to the position at power up). This is not enough to establish commutation and therefore there are additional means of obtaining the commutation angle.

Index pulse – an additional channel having just one pulse in motor revolution and hence defining an absolute position. The difficulty with index pulse is that we only benefit from it when we cross it, and for that we have to establish motion capability.

Hall sensor – additional 3 channels of digital information indicating the crossing between phase Kt functions. This serves as a rough absolute sensor in terms of commutation (not in terms of motor revolution). If capturing the hall's transition location a more accurate position indication can be obtained.

Wake and shake – if no halls are present, and we need to establish commutation then a process of applying currents to the motor takes place for identifying the electrical position, and starting the commutation.

Halls command and phase alignment



Electrical Cycle (CW)	Hall states A B C		
0 – 60	1	0	0
60 – 120	1	0	1
120 – 180	0	0	1
180 – 240	0	1	1
240 – 300	0	1	0
300 – 360	1	1	0

Note that when typing the HALLS command, the Reading is C-B-A

Drive operation for MENCTYPE=0,6

1. Checks hall state and establishes commutation as if the motor is at the center of the hall segment.
2. Waits for hall transition and when this occurs updates commutation angle more accurately.
3. Waits for index pulse transition and updates commutation according to MENCOFF definition.

Remark (1) – if index is encountered before hall transition (3 before 2), then (2) is skipped.

Remark (2) – if no index is present then stage (3) is skipped. (MENCTYPE=6).

Drive operation for MENCTYPE=1,2,3,4

For these types of systems there is no absolute indication of position at power up and hence a small "test" has to be performed to check the response of the motor to currents. The test is called ENCSTART and is activated either by a command (MENCTYPE=1,3), or automatically at enable of the drive (MENCTYPE=2,4). There are 3 methods of doing the process, and they are controlled by INITMODE variable.

In general the process consists of two stage:

1. Establish commutation by wake and shake procedure according to INITMODE.
2. Wait for index pulse crossing and update the commutation angle.

Remark : if no index is present (MENCTYPE=3,4) then the procedure terminates after step (1).

System components and possible initialization algorithms:

System	Marker pulse	Hall Effects	Recommended MENCTYPE	Possible MENCTYPE's
ABZH	√	√	0	0-4,6
ABZ	√	×	1-2	1-4
ABH	×	√	6	3-4,6
AB	×	×	3-4	3-4