16 Sine-Wave Look-up Table

The TMC2160 driver provides a programmable look-up table for storing the microstep current wave. As a default, the table is pre-programmed with a sine wave, which is a good starting point for most stepper motors. Reprogramming the table to a motor specific wave allows drastically improved microstepping especially with low-cost motors.

16.1 User Benefits

Microstepping	–	extremely improved with low cost motors
Motor	–	runs smooth and quiet
Torque	–	reduced mechanical resonances yields improved torque

16.2 Microstep Table

In order to minimize required memory and the amount of data to be programmed, only a quarter of the wave becomes stored. The internal microstep table maps the microstep wave from 0° to 90°. It becomes symmetrically extended to 360°. When reading out the table the 10-bit microstep counter *MSCNT* addresses the fully extended wave table. The table is stored in an incremental fashion, using each one bit per entry. Therefore only 256 bits (*ofs00* to *ofs255*) are required to store the quarter wave. These bits are mapped to eight 32 bit registers. Each *ofs* bit controls the addition of an inclination *Wx* or *Wx*+1 when advancing one step in the table. When *Wx* is 0, a 1 bit in the table at the actual microstep position means "add one" when advancing to the next microstep. As the wave can have a higher inclination than 1, the base inclinations *Wx* can be programmed to -1, 0, 1, or 2 using up to four flexible programmable segments within the quarter wave. This way even negative inclination can be realized. The four inclination segments are controlled by the position registers *X1* to *X3*. Inclination segment 0 goes from microstep position 0 to *X1*-1 and its base inclination is controlled by *W0*, segment 1 goes from *X1* to *X2*-1 with its base inclination controlled by *W1*, etc.

When modifying the wave, care must be taken to ensure a smooth and symmetrical zero transition when the quarter wave becomes expanded to a full wave. The maximum resulting swing of the wave should be adjusted to a range of -248 to 248, in order to give the best possible resolution while leaving headroom for the hysteresis-based chopper to add an offset.

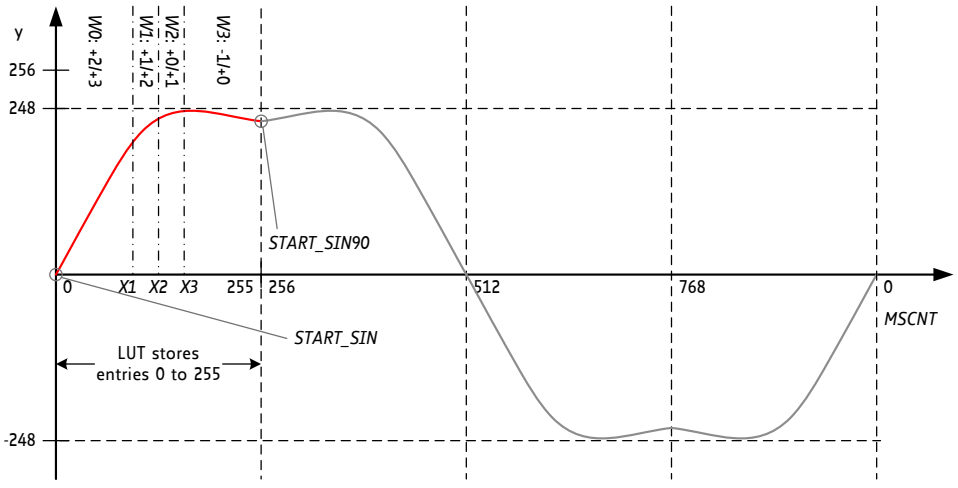

Figure 16.1 LUT programming example

When the microstep sequencer advances within the table, it calculates the actual current values for the motor coils with each microstep and stores them to the registers *CUR_A* and *CUR_B*. However, the incremental coding requires an absolute initialization, especially when the microstep table becomes modified. Therefore *CUR_A* and *CUR_B* become initialized whenever *MSCNT* passes zero.

Two registers control the starting values of the tables:

- As the starting value at zero is not necessarily 0 (it might be 1 or 2), it can be programmed into the starting point register *START_SIN*.
- In the same way, the start of the second wave for the second motor coil needs to be stored in *START_SIN90*. This register stores the resulting table entry for a phase shift of 90° for a 2-phase motor.

Hint
Refer chapter 5.3 for the register set and for the default table function stored in the drivers. The default table is a good base for realizing an own table.
The TMC2160-EVAL comes with a calculation tool for own waves.

Initialization example for the default microstep table:

MSLUT[0]= %10101010101010101011010101010100 = 0xAAAAB554
MSLUT[1]= %01001010100101010101010010101010 = 0x4A9554AA
MSLUT[2]= %00100100010010010010100100101001 = 0x24492929
MSLUT[3]= %00010000000010000010000010000010 = 0x10104222
MSLUT[4]= %11111011111111111111111111111111 = 0xFBFFFFFF
MSLUT[5]= %10110101101110110111011101111101 = 0xB5BB777D
MSLUT[6]= %01001001001010010101010101010110 = 0x49295556
MSLUT[7]= %00000000010000000100001000100010 = 0x00404222

MSLUTSEL= 0xFFFF8056:
X1=128, *X2*=255, *X3*=255
W3=%01, *W2*=%01, *W1*=%01, *W0*=%10

MSLUTSTART= 0x00F70000:
START_SIN_0= 0, *START_SIN90*= 247

17 Emergency Stop

The driver provides a negative active enable pin ENN to safely switch off all power MOSFETs. This allows putting the motor into freewheeling. Further, it is a safe hardware function whenever an emergency-stop not coupled to software is required. Some applications may require the driver to be put into a state with active holding current or with a passive braking mode. This is possible by programming the pin DCIN to act as a step disable function. Set GCONF flag *stop_enable* to activate this option. Whenever DCIN becomes pulled up, the motor will stop abruptly and go to the power down state, as configured via *IHOLD*, *IHOLD_DELAY* and stealthChop standstill options. Disabling the driver via ENN will require three clock cycles to safely switch off the driver.