Virtual Encoder

Virtual encoder emulation

Application Note

www.agito-akribis.com

Member of Akribis Systems group

Revision History

Version	Description	Date
1.0	Initial Release	11 May 2022

Contact Information

Manufacturer	Agito Akribis Systems Ltd., Member of Akribis Systems Group
Address	6 Yad-Harutsim St., P.O.Box 7172, Kfar-Saba 4464103
Telephone	+972-9-8909797
Website	www.agito-akribis.com

Copyright Notice

Products Rights

AGDx, AGCx, AGMx, AGAx, AGIx, and AGLx are products designed by Agito Akribis Systems Ltd. in Israel. Sales of the products are licensed to Akribis Systems Pte Ltd. under intercompany license agreement.

Agito Akribis Systems Ltd. has full rights to distribute above products worldwide.

Disclaimer

This product documentation was accurate and reliable at the time of its release.

Agito Akribis Systems Ltd. reserves the right to change the specifications of the product described in this manual without notice at any time.

Trademarks

Agito PCSuite is a trademark of Agito Akribis Systems Ltd..

Warranty

This product is warranted to be free of defects in material and workmanship and conforms to the specifications listed in this manual, for a period of 12 months from the shipment date from factory.

Contents

1 Introduction

1.1 Background

In many applications, such as pulse and direction control, remote control and many more, the position control loop need to receive encoder signal from motor drive.

Arguably, the simplest and most cost-effective solution would be using virtual encoder to transfer encoder reading from motor drive to controller.

Agito controllers have built-in features that allow the user to use virtual encoder function.

1.2 Scope

This application note seeks to introduce the gist of how Virtual Encoder can be implemented for standard applications. For the user's convenience, this document will pull in relevant definitions and details from other manuals; this document alone should be sufficient reference for the user to implement force control for the standard application.

However, for the full technicalities, the user should still refer to the relevant manuals.

This application note will take the approach of going through the implementation process of one of the more common applications, pulse and direction control and virtual encoder.

The steps of the pulse and direction control example will be as such,

1. From initial position, move motor to target position with pulse and direction control.
2. Motor drive generate virtual encoder and send back to controller.

The application note will also cover some basic information about the hardware setup.

2 Setup

2.1 Equipment and overview

The typical setup topology is presented in the following figure:

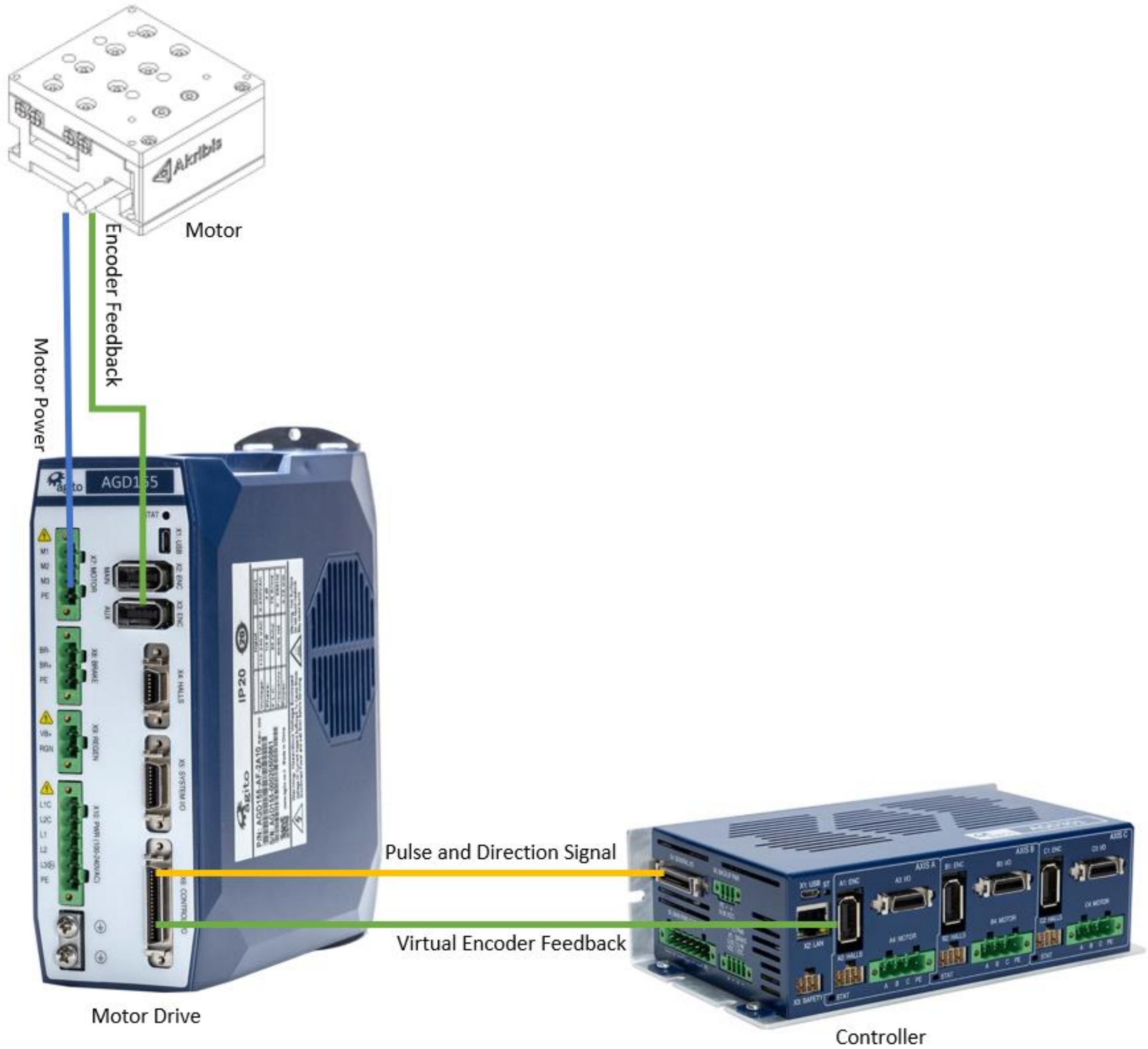

Figure 1. Standard setup topology

The example setup includes:

1. AGD155 motion controller with integrated drive, which generates motion profile, supplies power to the motor and generates virtual encoder signal.
2. AGD301 motion controller with integrated drive, which receives virtual encoder signal.
3. Motor with feedback (could be any rotary, linear or DC motor).

2.2 Wiring

AGD155: A4 – Motor Power

Motor Phase A	Phase A, M1	1	Motor Power
Motor Phase B	Phase B, M2	2	Motor Power
Motor Phase C	Phase C, M3	3	Motor Power, NC for voice coils
PE	PE	4	Motor PE

AGD155: X2 – Main Encoder Port (Example shows pinout for sincos encoder)

Function	Pin Name	Pin #	Remarks
5V	5V	1	Power for encoder
GND	GND	2	Power return for encoder
SIN+	Encoder_2+	5	Encoder signal input
SIN-	Encoder_2-	6	Encoder signal input
COS+	Encoder_3+	7	Encoder signal input
COS-	Encoder_3-	8	Encoder signal input
Z+	Encoder_4+	9	Encoder signal input
Z-	Encoder_4-	10	Encoder signal input

AGD155: X6 – Control I/O port

Function	Pin Name	Pin #	Remarks
Emulated encoder A+	Differential_Output_1+	1	Differential output 1+ | Emulated encoder A+
Emulated encoder A-	Differential_Output_1-	2	Differential output 1- | Emulated encoder A-
GND	GND	5	GND for the differential I/Os
Direction input+	Differential_Input_2+	8	Differential input 2+ | Direction input+ | Encoder follower B+ | Handwheel B+
Direction input-	Differential_Input_2-	9	Differential input 2- | Direction input- | Encoder follower B- | Handwheel B-
Emulated encoder B+	Differential output 2+	19	Differential output 2+ | Emulated encoder B+
Emulated encoder B-	Differential output 2-	20	Differential output 2- | Emulated encoder B-
Pulse input+	Differential_Input_1+	24	Differential input 1+ | Pulse input+ | Encoder follower A+ | Handwheel A+
Pulse input-	Differential_Input_1-	25	Differential input 1- | Pulse input- | Encoder follower A- | Handwheel A

AGD301: A1 – Encoder Port (Example shows pinout for Incremental encoder)

Function	Pin Name	Pin #	Remarks
5V	5V	1	Power for encoder
GND	GND	2	Power return for encoder
A+	Encoder_2+	5	Encoder signal input
A-	Encoder_2-	6	Encoder signal input
B+	Encoder_3+	7	Encoder signal input
B-	Encoder_3-	8	Encoder signal input
Z+	Encoder_4+	9	Encoder signal input
Z-	Encoder_4-	10	Encoder signal input

Note – Wiring for other controllers

This example uses Agito controller, AGD301 for the example. Wiring information for other controllers or encoder protocols can be found in their respective Product Manuals.

2.3 Differential Output Setting

As virtual encoder generated by AGD155 is send out from differential output, we need to do setting on two differential output ports.

In PCSuite, navigate to IO Tab 

, then the Digital Menu 

and then the Discrete Outputs Discrete Outputs.

We are going to use two differential outputs on X6 port, Pin 1,2 and 19,20.

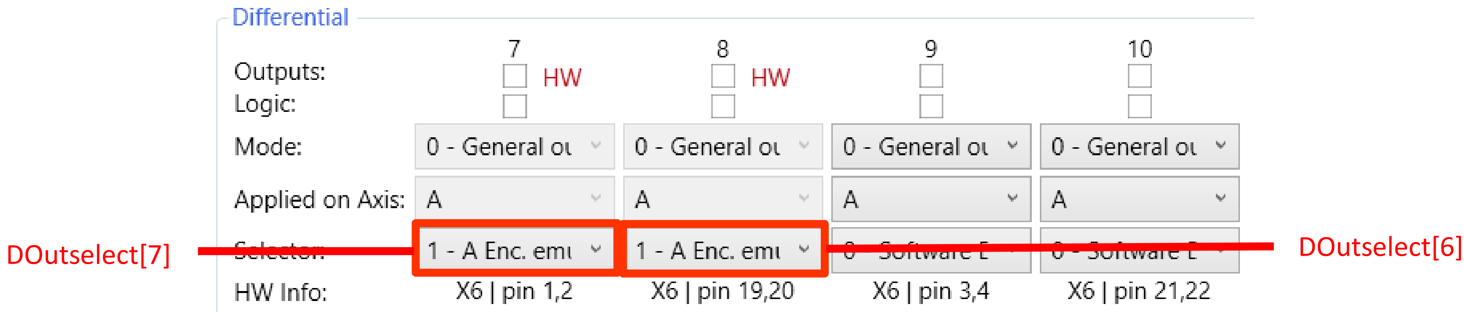

Figure 2. Screen capture of Digital Menu page on AGD155.

As in Figure 2, the parameters are set such that the virtual encoder signal will be sent out from two differential outports 1,2 and 19,20.

Keyword	Description

<table>
<tr><td>DOutselect[]</td><td>DOutMode defines special functionality that is given to a given output bit and is implemented by the controller software.
<table>
<tr><th>Value</th><th>State</th></tr>
<tr><td>1</td><td>Reserved or Pre-Defined function at differential outputs as following:
Differential output 1(DOutPort bit 4) – Encoder Emulation A for Axis A
Differential output 2(DOutPort bit 5) – Encoder Emulation B for Axis A
Differential output 3(DOutPort bit 6) – Encoder Emulation Z for Axis A
Differential output 4(DOutPort bit 7) – Encoder Emulation Pulse for Axis A</td></tr>
</table>
</td></tr>
</table>

2.4 Virtual Encoder Parameter Setting

As now the PCSuite GUI for virtual encoder parameter setting haven't been finished yet, we need to do these setting in Terminal window.

Please follow the keyword table to finish the virtual encoder parameter setting.

<table>
<tr><th>Keyword</th><th>Description</th></tr>
<tr><td>VEncOn</td><td>Decides whether to send anything to the FPGA to output.
<table>
<tr><th>Value</th><th>State</th></tr>
<tr><td>0</td><td>Virtual encoder is OFF</td></tr>
<tr><td>1</td><td>Virtual encoder is ON</td></tr>
</table>
</td></tr>
<tr><td>VEncSrc</td><td>Defines which variable to emulate using the keyword's CAN code.
E.g:
VEncSrc = 2 (CAN code of Pos) will result in outputting the delta between measured encoder positions.
VEncSrc = 24 (CAN code of PosRef) will result in outputting the delta between reference encoder positions.</td></tr>
<tr><td>VEncType</td><td>Defines in what protocol to output the signals, in AqB (Quadrature) or PD (Pulse and Direction).
<table>
<tr><th>Value</th><th>State</th></tr>
<tr><td>0</td><td>PD</td></tr>
<tr><td>1</td><td>AqB</td></tr>
</table>
</td></tr>
</table>

VEncDelay	The time in microseconds to define the delay. To allow for a smooth direction change, it is not desirable to have changes in either of the 2 differential signals being outputted happen simultaneously. So, when a direction change is commanded, the following happens at the next interrupt: 1.There is a "VEncDelay" in which nothing changes. 2.One of the differential signals starts 3.Following an additional "VEncDelay", the other signal commences as well.
VEncFact	VEncFact is the virtual encoder factor. E.g: To enable the use of fractions VEncFact is multiplied by 65536. For an emulation ratio of 1. For an emulation ratio of 0.5 VEncFact = 32768. For an emulation ratio of 3, VEncFact = 65536*3 = 196608.

2.5 Controller Motor and Feedback Setting

Here we set "VEncType =1" on AGD155, so the virtual encoder signal generated is AqB signal.

In this way, we need to set AqB encoder feedback on AGD301.

In PCSuite, navigate to CONFIG Tab 

, then the MOT Menu 

.

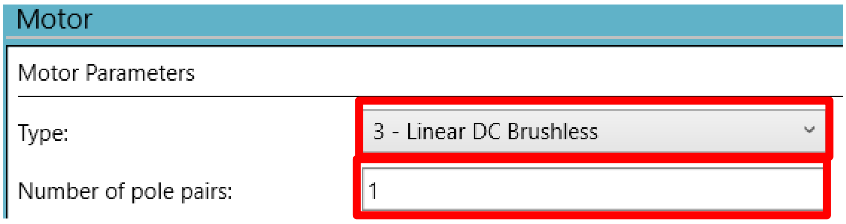

Figure 3. Screen capture of Motor page on AGD155.

Figure 3. Screen capture of Motor Menu page.

As in Figure 3, the motor type we choose 3 – Linear DC Brushless. The Number of pole pairs we set as 1.

Navigate to CONFIG Tab 

, then the FDBK Menu 

.

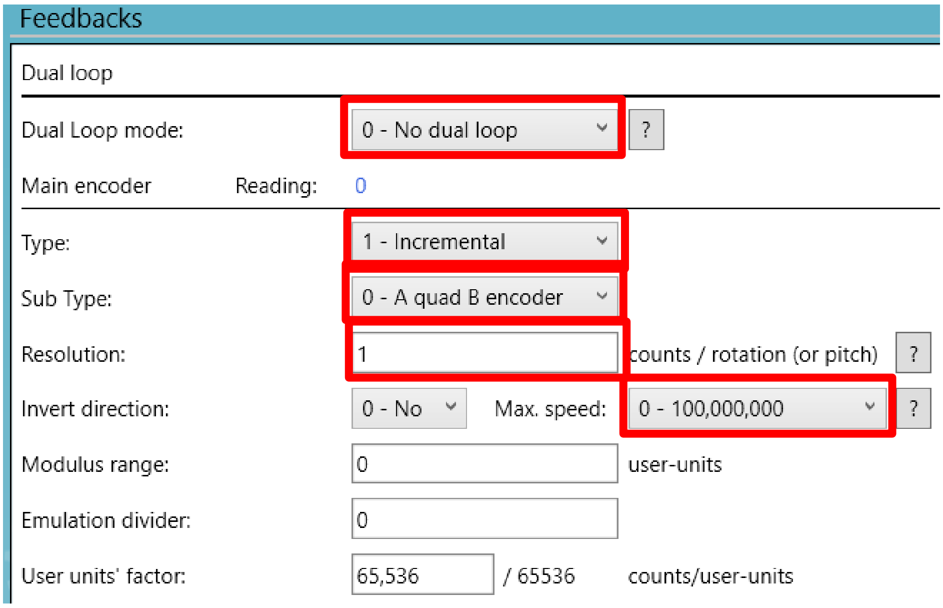

Figure 4. Screen capture of Feedback Menu page on AGD155.

As in Figure 4, we choose dual loop mode as 0 – No dual loop. Encoder type need to be set as 1 – Incremental, 0 – A quad B encoder. The resolution needs to be set to 1. And please choose the max. speed depends on the motion speed you need.

2.6 Pulse and Direction Signal Setting

AGD301 is used to generate pulse and direction signal to control AGD155.

In PCSuite, navigate to IO Tab 

, then the Digital Menu 

and then the Discrete Outputs Discrete Outputs.

We are going to use two differential outputs on X4 port, Pin 1,2 and 19,20.

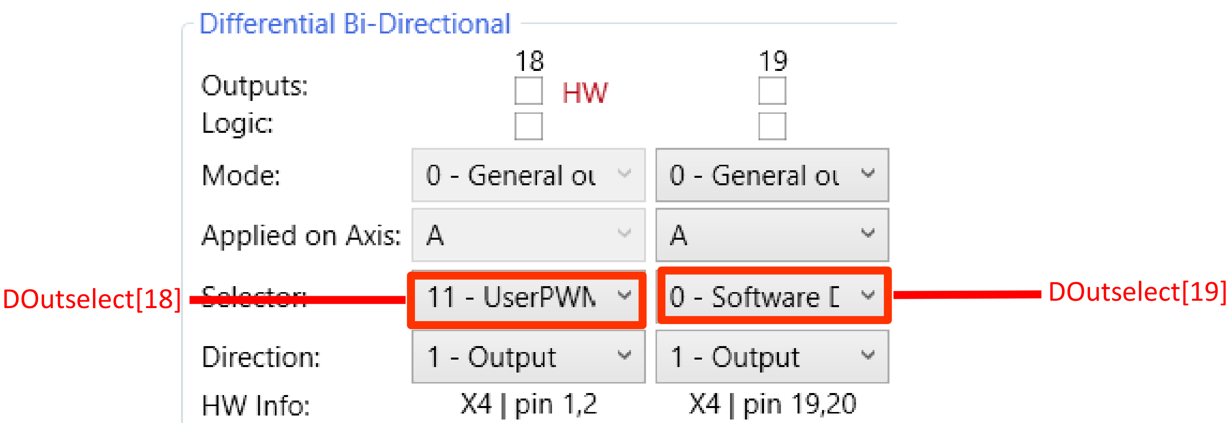

Figure 5. Screen capture of Digital Menu page on AGD301.

As in Figure 2, the parameters are set such that the virtual encoder signal will be sent out from two differential outports 1,2 and 19,20.

<table>
<tr><th>Keyword</th><th>Description</th></tr>
<tr><td>DOutselect[]</td><td>DOutMode defines special functionality that is given to a given output bit and is implemented by the controller software.
<table>
<tr><th>Value</th><th>State</th></tr>
<tr><td>0</td><td>Software DoutPort</td></tr>
<tr><td>11</td><td>UserPWM 1</td></tr>
</table>
</td></tr>
</table>

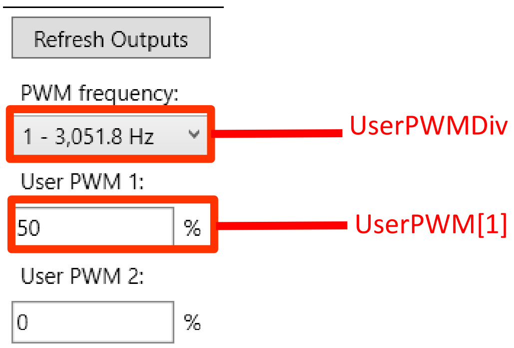

Figure 6. Screen capture of Digital Menu page on AGD301.

For the PWM frequency setting,

Keyword	Description
UserPWMDiv	Please refer to the UserPWM manual page for full description of the UserPWM feature. Example: AUserPWMDiv=3; Then PWM clock Frequency=80/2^ (3 + 1) MHz=5 MHz and the resulted PWM frequency will be: 5MHz / 4096 = 1.22 KHz.
UserPWM	UserPWM controls the duty cycle of each user PWM output. This is the compare value such that when the internal counter < UserPWM[N] the output value is 1 else the output value is 0. This allows us to generate a PWM signal where we can vary the duty cycle with resolution of 1/4096(12bit).

3 Virtual Encoder Application Example

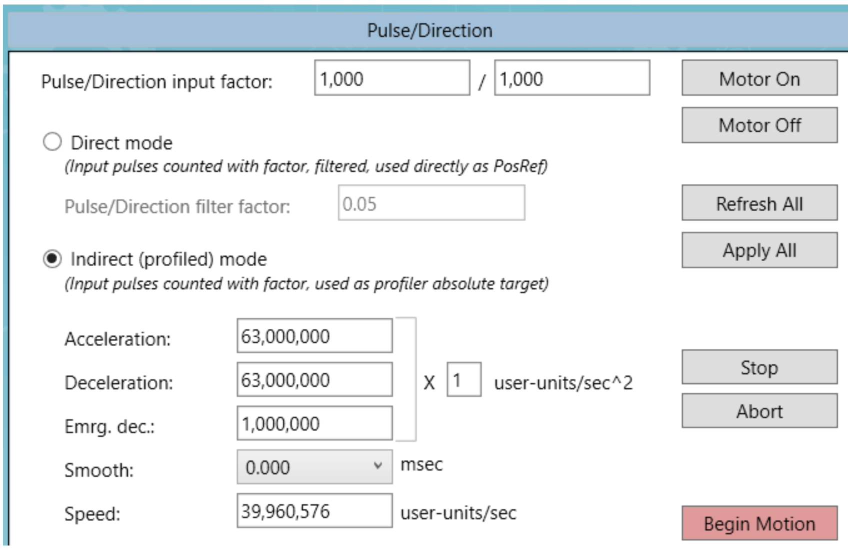

Figure 7. Motion profile on AGD155.

Before starting motion on AGD155, we set the position to zero on AGD155 and AGD301.

Then follow Figure 5, we set target to 100,000,000 counts.

For the VEncFact, we set to 4096, so the ratio of virtual encoder is 4096/65536=0.0625.

For the VEncSrc, we set to 2, means we use AGD155 position reading as virtual encoder source.

Now we move AGD155 to position to 100,000,000.

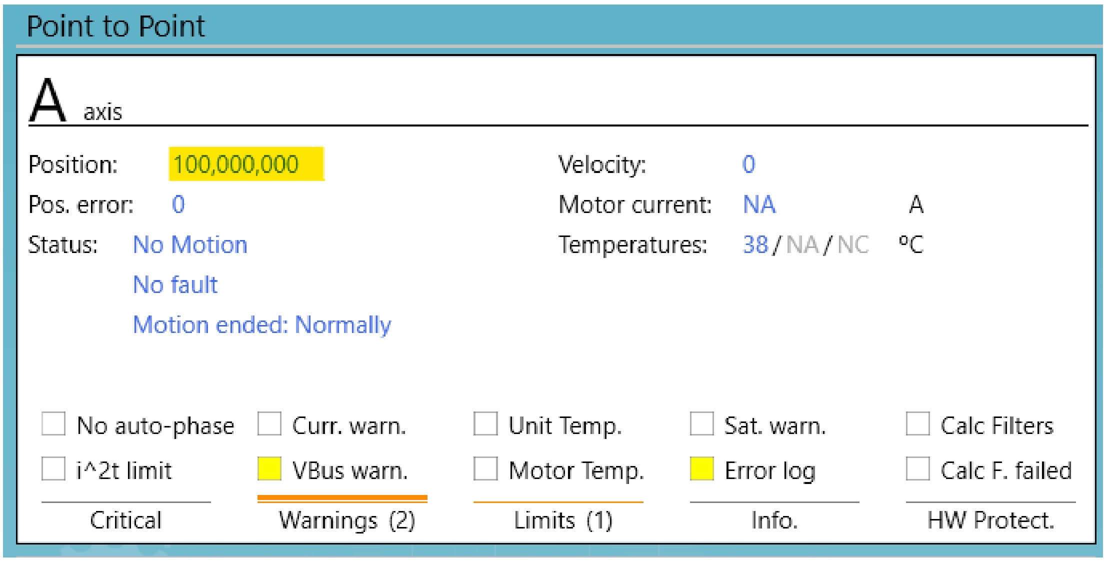

Figure 8. Position reading on AGD155.

Check the position reading on AGD301, it shows 6,250,000 = 100,000,000 * 0.0625.

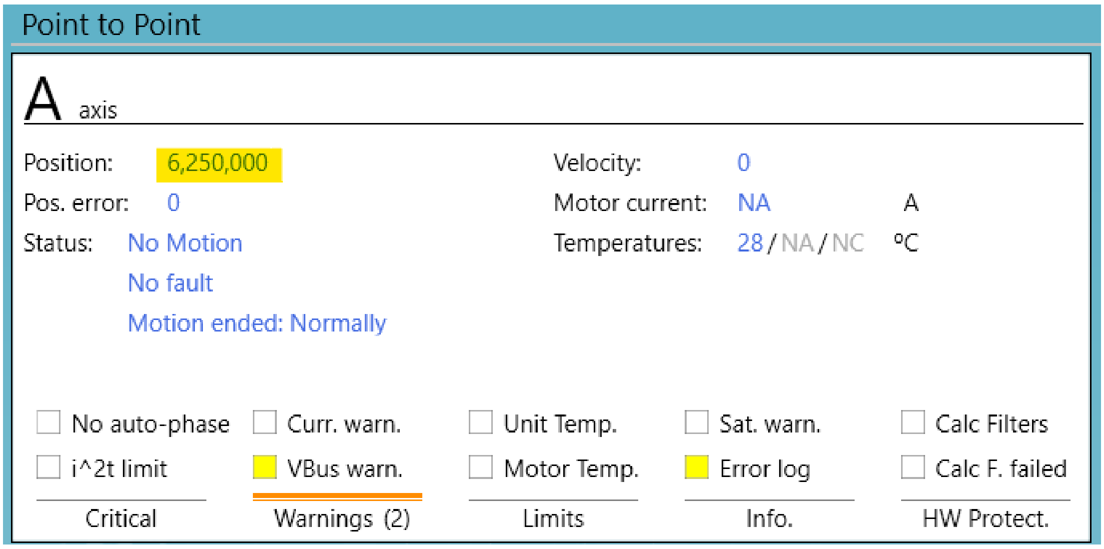

Figure 9. Position reading on AGD301.