VLC-25-13

USER MANUAL

Version 1.4

SMAC
Moving Coil Actuators

Disclaimer

The contents of this user manual are intended to be as accurate as possible, but may be subject to change without prior notification. SMAC shall not be liable for any damages that may arise as a consequence of the use of information presented in this user manual.

Document Version	Note	By	Date
1.0	1st release.	RZ	8/14/2020
1.1	Added low pass filter variables, added an example on current mode, fixed the I^2T example program.	RZ	10/16/2020
1.2	General updates (typos, missing information, etc.), added a command summary in the appendix.	RZ	2/2/2021
1.3	Revisions: P/N from VLC-25H13 to VLC-25-13, JR command argument range, added data capture system variables, RM command description.	RZ	6/28/2021
1.4	Added a more specific description of the VLC in the introduction, added an information on the SQ value duty cycle limit under the SQ command description, updated the phasing program examples with an automatic current sensing offset adjustment routine, added an appendix with information for first-time users of VLCs.	RZ	1/10/2022

Contents

1 Introduction

The VLC-25-13 is a 2-axis standalone integrated controller/servo drive that is used for the control of brushed/brushless industrial linear actuators and motors. A typical use of VLC-25-13 is in the control of SMAC's linear-rotary actuator, which involves coordinated tasks between the 2 actuator axes. The VLC-25-13 has current ratings of 10 A continuous and 13 A peak. It is equipped with 4 opto-isolated digital I/O pairs, 5 analog inputs and 2 analog outputs, as well as STO (Safe Torque Off) channels. The housing of VLC-25-13 enables it to be mounted on its side or base part with screws. By request, a clip for DIN rail mounting can be made available.

The VLC-25-13 implements a mnemonic type command instruction set via a standard RS-232 serial communication interface. These commands can be executed directly or used to create programs that can be stored in the on-board flash memory.

Figure 1.1 depicts the VLC-25-13, for which the specifications are presented in **Table 1.1**.

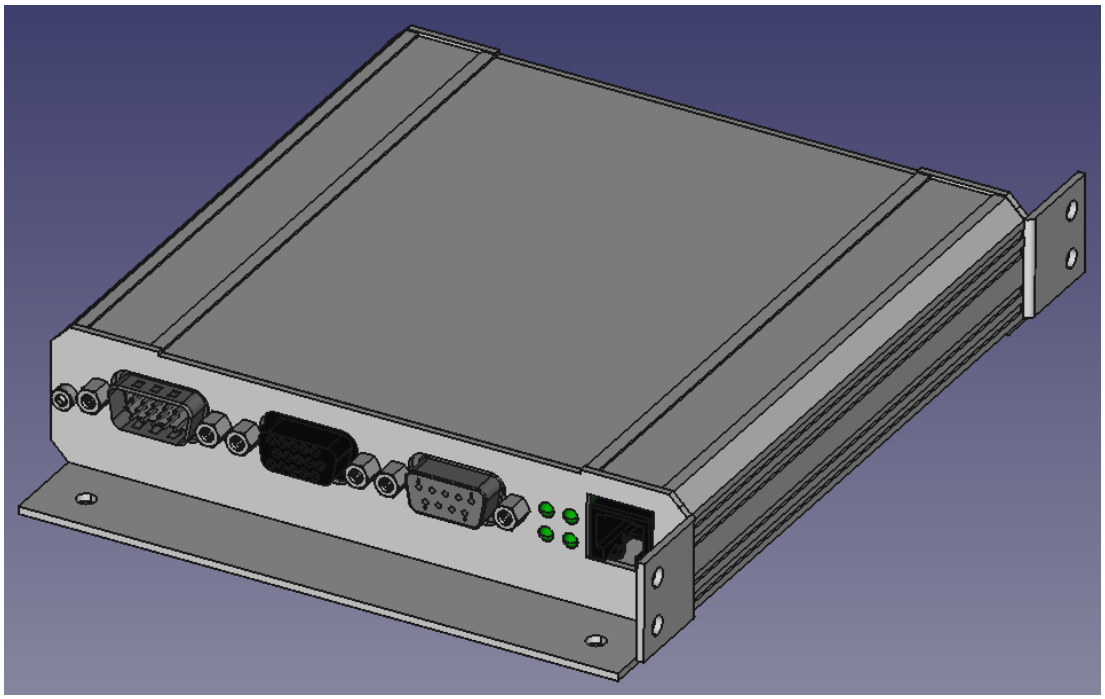


Figure 1.1. VLC-25-13.

Table 1.1. VLC-25-13 specifications.

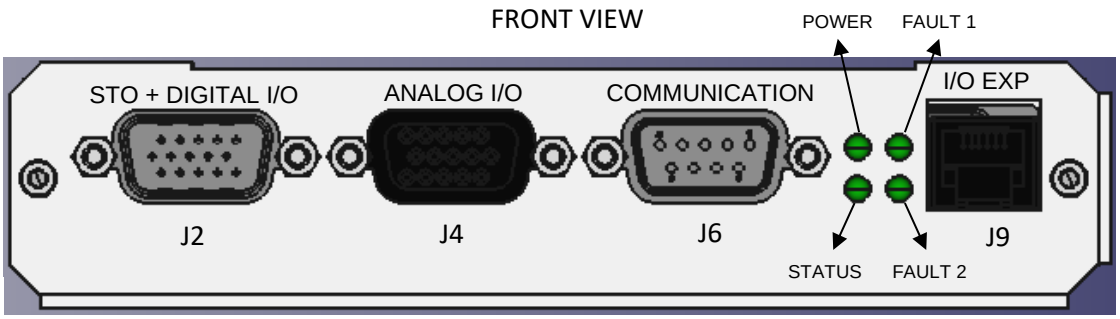
Description	Stand-Alone 2 Axis Servo Motor Controller / Driver
Operating Modes	Position, Velocity, Torque
Filter Algorithm	PID
Max. Servo Loop Rate	100 µS
Trajectory Generator	Trapezoidal, electronic gearing
Servo Position Feedback	Incremental Encoder with Index
Output	PWM (space-vector-modulated), 10 A Continuous and 13 A Peak.
Motor Type	3-Phase Brushless, DC Brushed, DC Linear Actuator
PWM Frequency	20.0 KHz

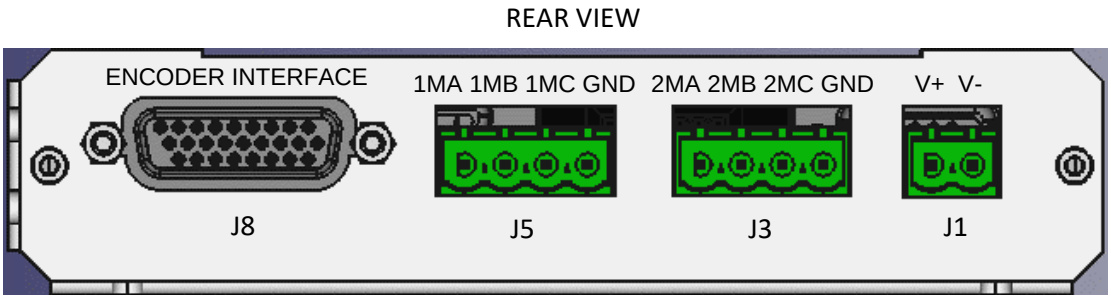
Current resolution	6.35 mA (approximate)
Encoder and Index Input	Differential
Encoder Supply Voltage	5 VDC
Encoder Input Voltage	5.5 VDC Max., -0.1 VDC Min.
Encoder Count Rate	40 million encoder counts per second
Position Range	31 Bits
Velocity Range	31 Bits
Acceleration Range	31 Bits
General Purpose Digital I/O	4x opto-isolated digital inputs, 5V to 24 V max 4x opto-isolated digital outputs, 60V, 200 mA max
Dedicated Digital I/O	2x opto-isolated coarse home inputs
STO (Safe Torque Off)	2x STO opto-isolated digital inputs, 5V to 24 V max 1x STO opto-isolated feedback output, 60V, 200 mA max
Analog Inputs	2x 12-bit pseudo-differential analog inputs, 0 to +/- 10V range 3x 12-bit analog inputs, 0V to 10V range (0V to <10V optional)
Analog Outputs	2x 12-bit analog outputs, 0V to 10V range (0V to 5V optional)
LEDs	1x Power on LED 1x Status LED 2x Fault LEDs
Serial Interface	RS-232 non-isolated, 9600 baud default, selectable between 2400 - 460800
Supply Voltage	+8 To +48 VDC
Protections	> Reversed power supply polarity connection > Driver overtemperature > I^2T > (excessive) servo position error
Program space	> Macro storage: 56286 bytes > Maximum number of macros: 256 (from firmware v1.53: 512) > Maximum number of program registers: 2048

2 Hardware and Software

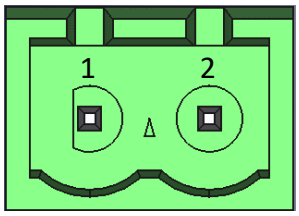
2.1 Hardware

2.1.1 Power/signal/communication connectors





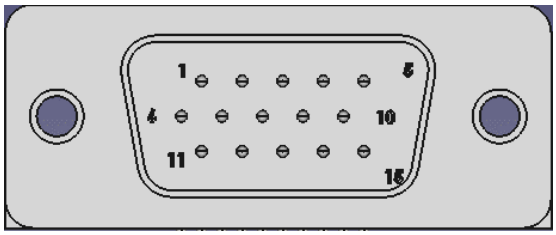
J1 - Power interface



2 pin terminal block header, 5.08 mm pitch.

Pin number	Signal	Description
1	V+	Power supply positive
2	V-	Power supply return / ground

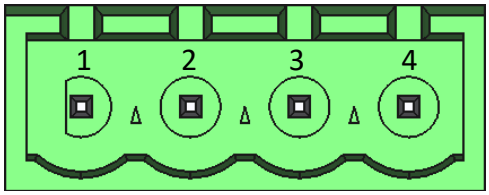
J2 – General purpose + dedicated digital opto-isolated I/O



D-SUB 15 connector, high-density, male.

Pin number	Signal	Description
1	GPI0	General purpose digital input 0
2	GPI1	General purpose digital input 1
3	GPI2	General purpose digital input 2
4	GPI3	General purpose digital input 3
5	GPI_COM	Common terminal for general purpose digital inputs
6	GPO0	General purpose digital output 0
7	GPO1	General purpose digital output 1
8	GPO2	General purpose digital output 2
9	GPO3	General purpose digital output 3
10	GPO_COM	Common terminal for general purpose digital outputs
11	+5V	+5V power for external circuitry
12	STO2	STO input 2
13	STO1	STO input 1
14	STO_FB	STO feedback output
15	STO_COM	Common terminal for STO's

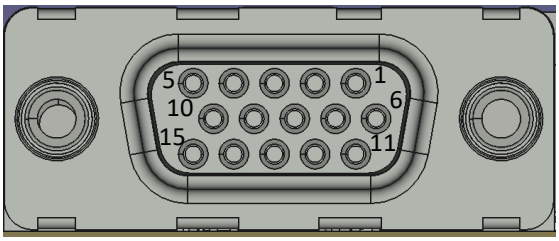
J3 – Axis 2 Motor Output



4 pin terminal block header, 5.08 mm pitch.

Pin number	Signal	Description
1	2MA	Axis 2 motor phase A/U (positive for single-phase actuators)
2	2MB	Axis 2 motor phase B/V (negative for single-phase actuators)
3	2MC	Axis 2 motor phase C/W
4	GND	Ground

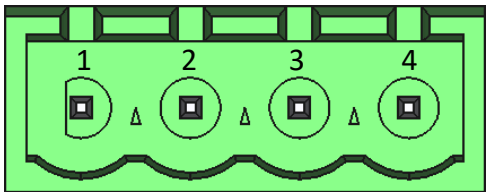
J4 – Analog I/O



D-SUB 15 connector, high-density, female.

Pin number	Signal	Description
1	AN_IN2	Analog input 2, single-ended
2	AN_OUT0	Analog output 0
3	AN_OUT1	Analog output 1
4	AN_IN0+	Analog input 0, differential input +
5	AN_IN1+	Analog input 1, differential input +
6	AN_IN3	Analog input 3, single-ended
7	GND	Ground
8	GND	Ground
9	AN_IN0-	Analog input 0, differential input -
10	AN_IN1-	Analog input 1, differential input -
11	AN_IN4	Analog input 4, single-ended
12	+5V	+5V power for external circuitry
13	+5V	+5V power for external circuitry
14	GND	Ground
15	GND	Ground

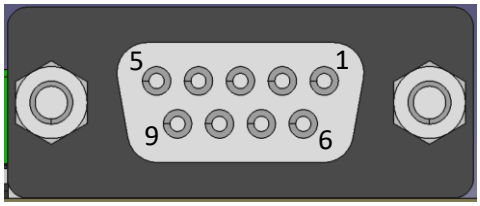
J5 – Axis 1 Motor Output



4 pin terminal block header, 5.08 mm pitch.

Pin number	Signal	Description
1	1MA	Axis 1 motor phase A/U (positive for single-phase actuators)
2	1MB	Axis 1 motor phase B/V (negative for single-phase actuators)
3	1MC	Axis 1 motor phase C/W
4	GND	Ground

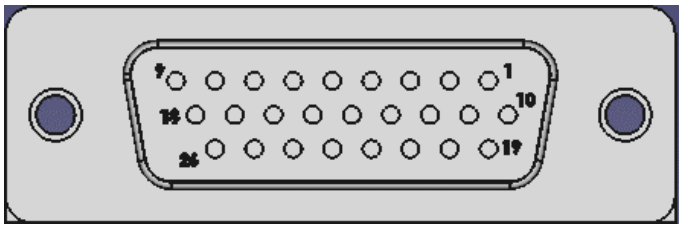
J6 – Communication (RS232)



D-SUB 9 connector, female.

Pin number	Signal	Description
1	+5V	+5V power for external circuitry
2	TX	RS232 Transmit
3	RX	RS232 Receive
4	N/C	Not connected
5	GND	Ground
6	N/A	Reserved, do not connect
7	N/A	Reserved, do not connect
8	N/A	Reserved, do not connect
9	N/A	Reserved, do not connect

J8 – Incremental encoder interface

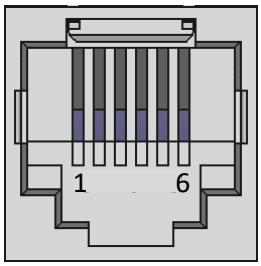


D-SUB 26 connector, high-density, female.

Pin number	Signal	Description
1	N/C	Not connected
2	N/C	Not connected
3	N/C	Not connected
4	+5V	+5V power for external circuitry
5	+5V	+5V power for external circuitry
6	2A-	Axis 2 encoder phase A-
7	2A+	Axis 2 encoder phase A+
8	1A-	Axis 1 encoder phase A-
9	1A+	Axis 1 encoder phase A+
10	N/C	Not connected
11	N/C	Not connected
12	1HOM	Axis 1 home input
13	GND	Ground
14	GND	Ground
15	2B-	Axis 2 encoder phase B-
16	2B+	Axis 2 encoder phase B+
17	1B-	Axis 1 encoder phase B-
18	1B+	Axis 1 encoder phase B+

19	2HOM	Axis 2 home input
20	N/C	Not connected
21	GND	Ground
22	GND	Ground
23	2Z-	Axis 2 encoder phase index-
24	2Z+	Axis 2 encoder phase index+
25	1Z-	Axis 1 encoder phase index-
26	1Z+	Axis 1 encoder phase indx+

J9 – Expansion I/O

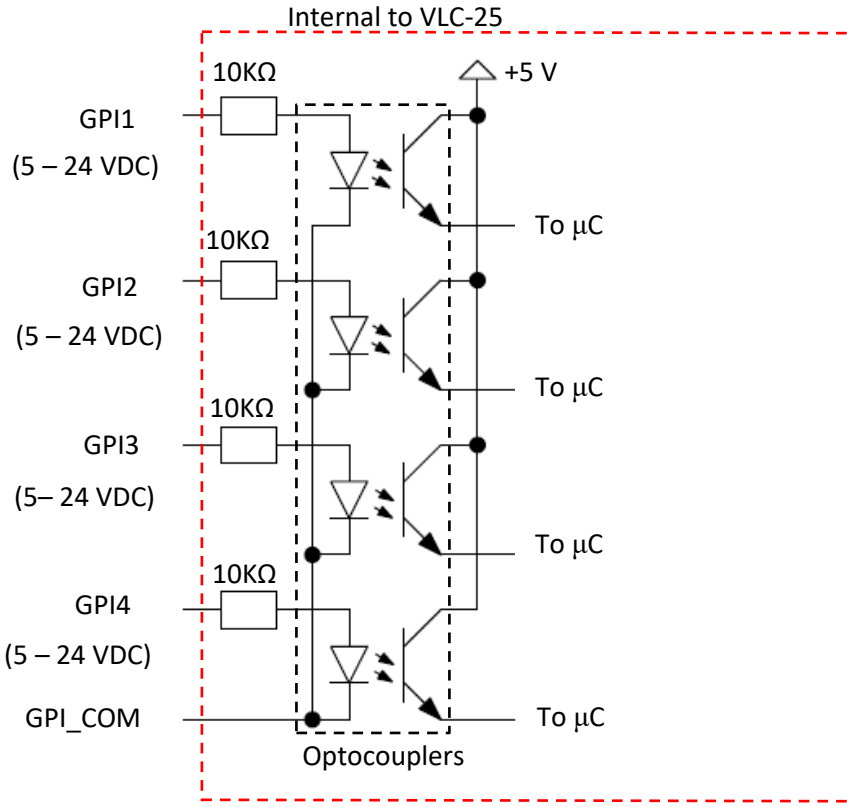


RJ-25 socket connector.

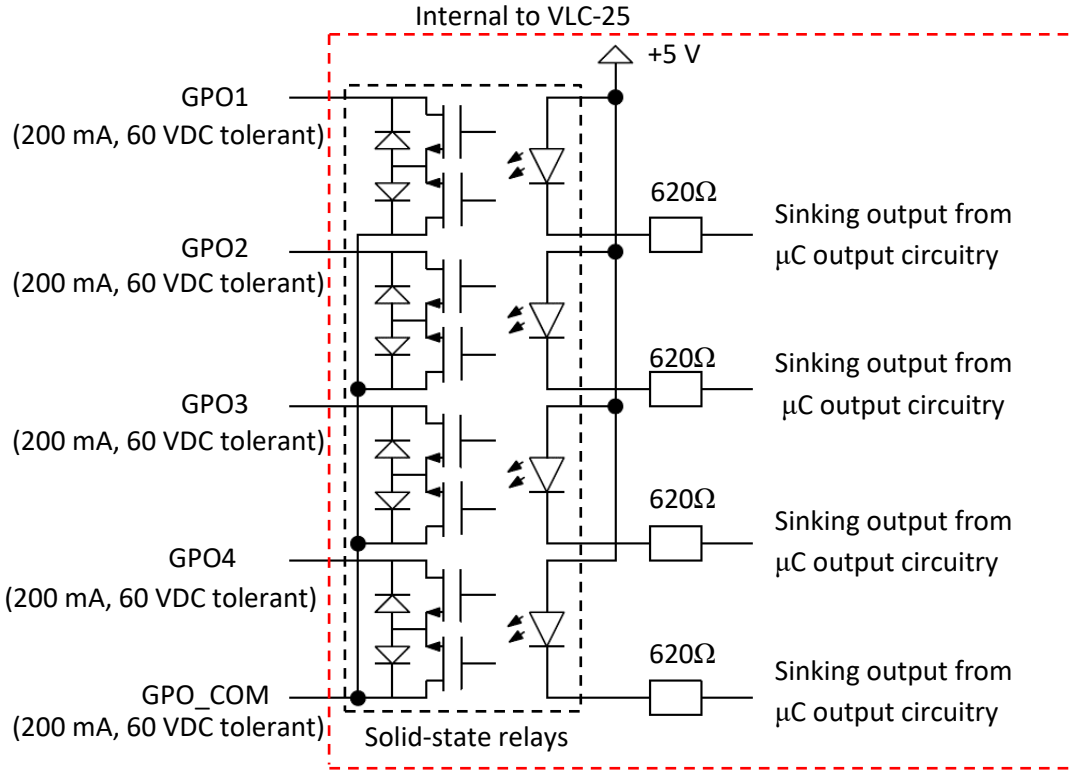
Pin number	Signal	Description
1	RX-	Receive data- input
2	TX-	Transmit data- input
3	RX+	Receive data+ input
4	TX+	Transmit data+ input
5	CLK+	Clock+ input
6	CLK-	Clock- input

2.1.2 I/O electrical schematics

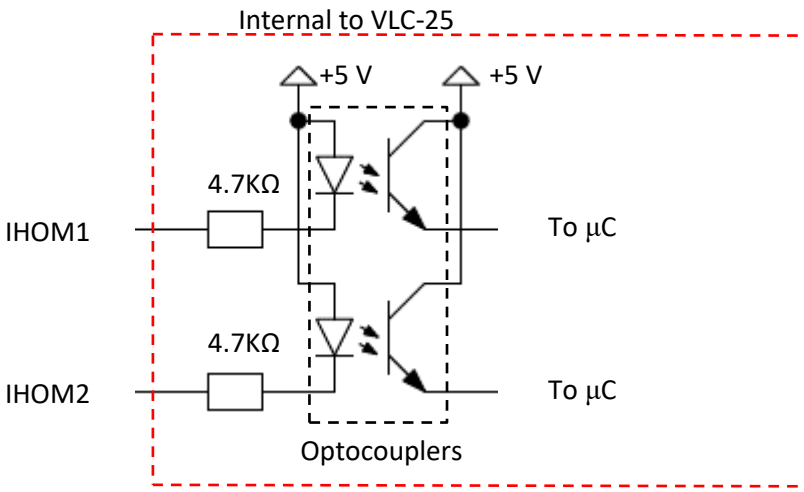
Digital inputs



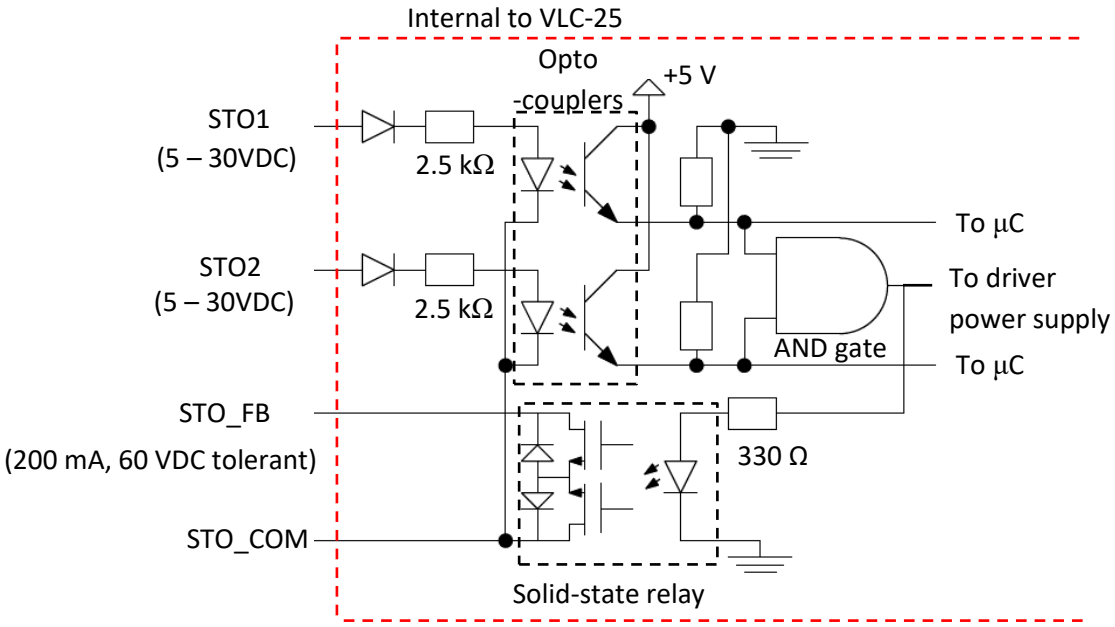
Digital outputs



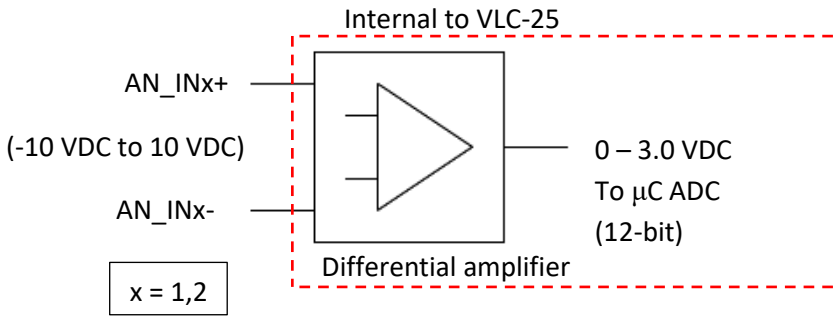
Home inputs



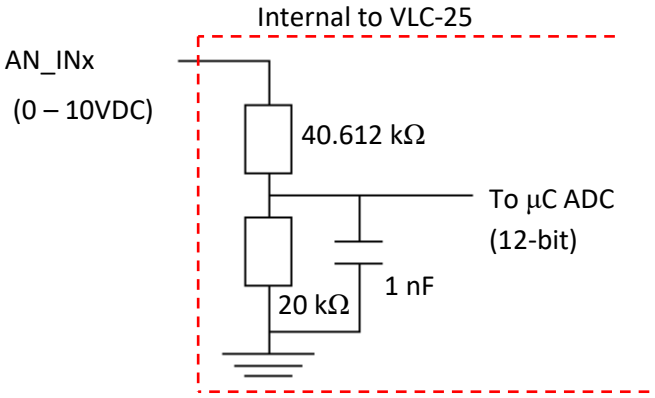
STO



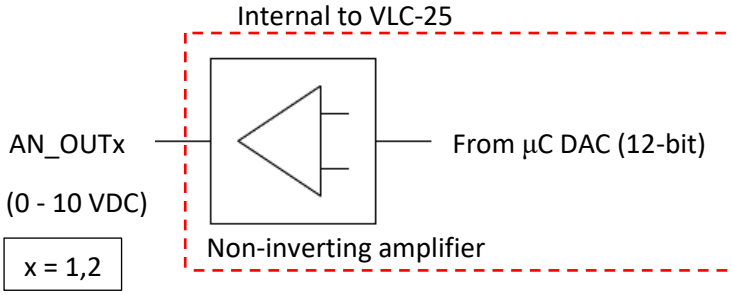
Analog input (differential)



Analog input (single-ended)



Analog output



2.1.3 LEDs

There are 4 LEDs in VLC-25-13, namely power, status, axis 1 fault and axis 2 fault. The location of these LEDs can be seen in the front view of VLC-25-13 in section 2.1.1.

- Power LED turns ON when the specified DC voltage is supplied into the controller
- Status LED blinks if an incorrect program command is executed
- Axis fault LED turns ON when the corresponding servo axis experiences fault such as driver overtemperature, I^2T trip and excessive servo position error (SE command).

2.2 Software

Remark: First-time users of VLCs are highly recommended to read Appendix E. Users who are familiar with the LAC-25 controller are reminded to be aware of programming differences with the VLC-25, which are further described in Appendix B.

2.2.1 Software environment

The VLC-25 communicates with a host PC via an RS-232 serial interface. Generic serial terminal software (e.g. Tera Term, Termite) or SMAC LAC-X editor (downloadable from https://www.smac-mca.com/lac-single-axis-controller-p-127.html?cPath=1_10) with a GUI screenshot shown in **Figure 2.1** can be used to program VLC-25. The serial baud rate is user selectable from 2400 bps – 460800 bps, with 9600 bps being the default. Characters are fixed at 8 bits in length with 1 stop bit and no parity.

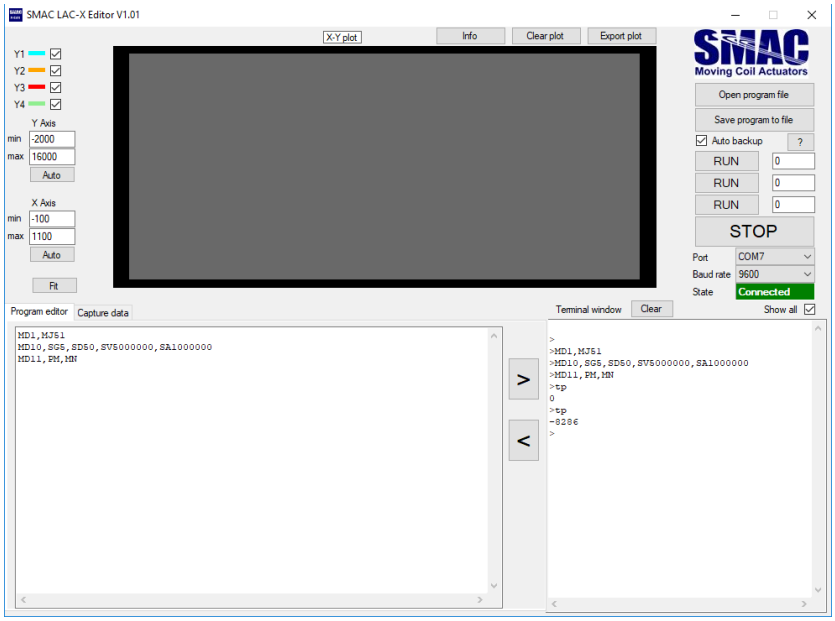


Figure 2.1. SMAC LAC-X Editor software environment.

2.2.2 Programming concept

Immediately after power-up, VLC-25 is ready to accept commands from the serial terminal software. To verify this, hit the ESC key and if the controller power, communication connection and serial communication setting are properly configured, the following sign should appear

>

After that, commands (further discussed in **section 3**) can be entered with the following syntax

`[Axis#]Command[argument],Command[argument],…,Command[argument]`

which can go up to a maximum line length of 127 characters.

Next, by hitting the RETURN key (typically corresponds to a Carriage Return), the previous commands will be downloaded to the controller and executed. If any of the commands or arguments is incorrect, the following will appear

```
? <error code>
```

and the commands will not be executed. A list of the error codes is given in **appendix A**.

Otherwise, after the commands have been completely executed, the following sign will appear again

```
>
```

If the command execution needs to be stopped, hit the ESC key.

The axis number is normally specified as being from "1" to "2", with "0" being used to refer to both axes at the same time. Once an axis has been specified, the same one will remain in effect until another is specified. For example, if the following were entered:

```
1SG10,SD100,SV100000<CR>
```

Or,

```
1SG10<CR>
SD100<CR>
SV100000<CR>
```

The SG command specifies the axis number, so the subsequent SD and SV commands are performed on the same axis. If the following command were issued:

```
0TP<CR>
```

It would have the same effect as issuing these commands:

```
1TP<CR>
2TP<CR>
```

2.2.2.1 Macros

The sequence of commands can be stored in a macro, with the following syntax

```
MD<macro number>,[Axis#]Command[argument],…,Command[argument]
```

This way, a given macro could be programmed by the user to represent a certain part of a motion program, such as configuration, initialization, point-to-point move, etc.

Macros that have been stored can be executed by executing the following command corresponding to the macro number

```
MS<macro number>
```

To store macros in the Non-Volatile Memory of VLC-25, execute the following command

```
PS
```

Note that macro number: 0, is automatically executed when VLC-25 powers up.

From FW version 1.53, the maximum number of macros is 512. With 56286 bytes of macro storage, over 9000 commands are allowed, averaging about 18 commands per macro.

Further discussions on commands associated with macros are given in **section 3**.

2.2.2.2 Commenting program codes

If a command line is ended by character ";" and a comment

```
SG100,SD1000 ; set filter gains.
```

Then the ";" and anything following to the end of the line is treated as a comment and will be ignored by VLC-25. This feature could be useful for documentation purposes when the user prepares the VLC-25 program macros in a separate text editor, whose content could be copy-and-pasted to the serial terminal software.

2.2.3 Macro Interrupt System

The VLC-25 employs a "Macro Interrupt System" to provide additional versatility in programming the VLC-25. This system comprises 11 interrupt sources with corresponding vectors. When an interrupt's source is enabled for operation and then becomes active, the current macro being executed is saved to a so called macro stack and execution of the macro specified by that interrupt's vector table entry begins. This happens to be similar procedure to that which the Macro Call (MC) command follows.

2.2.3.1 The Interrupt Vector Table

The Interrupt Vector Table consists of an entry for each interrupt source and each entry will correspond to that interrupt's level (level 0 = entry 0, level 1 = entry 1, etc.). A particular table entry must be loaded with the number of a valid macro to be executed should that interrupt source become active. The method for loading a vector table entry is provided by the Load Vector (LV) command. The user must first use the Accumulator Load (AL) command to set the number of the macro for a vector. The LV command is then used to transfer the low 8-bits of the accumulator to the vector table entry specified by the LV command. If an interrupt is generated and that vector table entry has not been defined (equal to 0) then the interrupt will not be executed. Note that this implies that macro "0" cannot be used as an interrupt macro. If an interrupt is generated and its vector table entry has been defined but the macro it specifies has not, then an error will be reported.

2.2.3.2 Enabling and Disabling Interrupts

Loading a vector table entry will not enable an interrupt for operation. The Enable Vector (EV) command must be used for this purpose. When the EV command is used, it will enable the interrupt source (specified with the command) to function. In the event that it is necessary to disable an interrupt source, there is a Disable Vector (DV) command that functions in a similar manner as the EV command.

In order to prevent multiple or continuous interrupts, **as an interrupt is taken it is automatically disabled.** This means that the user must re-enable that interrupt using the EV command before it will occur again.

2.2.3.3 Interrupt Sources

The following table lists all the possible interrupt sources.

Interrupt Source	Level / Vector	Interrupt Source	Level/Vector
Axis 0 Error	31	(NB command)	15
Axis 1 Error	30	Reserved	14
Reserved	29	Reserved	13
Reserved	28	Reserved	12
Axis 0 Fault	27	Reserved	11
Axis 1 Fault	26	Reserved	10
Reserved	25	Reserved	9
Reserved	24	Reserved	8
Reserved	23	Reserved	7
Reserved	22	Reserved	6
Reserved	21	Reserved	5
Reserved	20	Reserved	4
Axis 0 IP/IR	19	Digital input 3	3
Axis 1 IP/IR	18	Digital input 2	2
Reserved	17	Digital Input 1	1
Reserved	16	Digital Input 0	0

The Axis Error interrupt indicates that the position following error for the axis has exceeded the limit set by the Set Error (SE) command. Normally, when this limit is exceeded, the servo is disabled and the "Error" bit in that axis' status word is set. **If the interrupt for this condition is enabled, the "Error" bit will still be set but the servo will not be disabled.**

The Axis Fault interrupt indicates that a fault condition (usually an over-temperature condition) has arisen. Normally, when this condition is detected, the servo is disabled and the "Fault" bit in the status word is set. **If the interrupt for this condition is enabled, the "Fault" bit will still be set but the servo will not be disabled.**

Digital Inputs 0 - 1 provide 2 levels of undedicated, user definable interrupts. The interrupt for a given input will be active when that input is active.

2.2.3.4 Interrupt Priority

If more than one interrupt source becomes active at the same time, then the source with the higher level will be executed first. Level/vector 31 has the highest priority and level/vector 0 has the lowest priority.

2.2.3.5 Interrupt Completion

Once an interrupt macro (or set of macros) has finished executing, a Return from Call (RC) command or an undefined macro may be used to cause a return from the interrupting macro back to the interrupted macro where command execution will continue from where it was interrupted (see MS command). In cases where it is undesirable to return to the interrupted macro, the Unpush Macro (UM) command can be used to remove the previously pushed macro from the macro stack. This

command can also be used to completely reset the macro stack in order that the user program can be restarted.

2.2.3.6 Interrupt Latency

Interrupt sources are sampled before each command in a macro is executed. This means that the amount of time that an interrupt is held off before execution (also known as interrupt latency) depends on how long it takes the previous command to complete. For most commands this delay will be imperceptible to the user.

Commands such as Wait (WA), Wait for Edge (WE), Wait for Stop (WS), Wait for Off (WF), Wait for On (WN) and Wait for Index (WI) would normally be a source of unacceptable delay in that they can quite often be indeterminate in length. This problem has been avoided by making these instructions interruptable. For example, if a WA10000 command (a 10 second delay) is currently in progress and an interrupt comes along, the remaining delay period will be saved and then returned to after the interrupt has completed. If the interrupt were to take 3 seconds to execute, then the total wait time of the WA10000 command would be extended to 13 seconds.

The Position Mode (PM), Torque Mode (QM), Velocity Mode (VM), Wait for Position Absolute (WP) and Wait for Position Relative (WR) commands and any command that uses the serial communications link are all commands that could cause unacceptable interrupt latency. Therefore, their usage should be carefully considered where interrupts are possible.

2.2.4 Servo operating modes

The VLC-25 has four servo operating modes. The following subsections describe them further and present the associated VLC-25 commands and axis variables (more information is in subsection 3.4.1 of this manual).

2.2.4.1 Position mode

Figure 2.2 depicts the control loop diagram that is used when the position mode is enabled through the PM command, whereas **Table 2.1** presents the associated commands and axis variables.

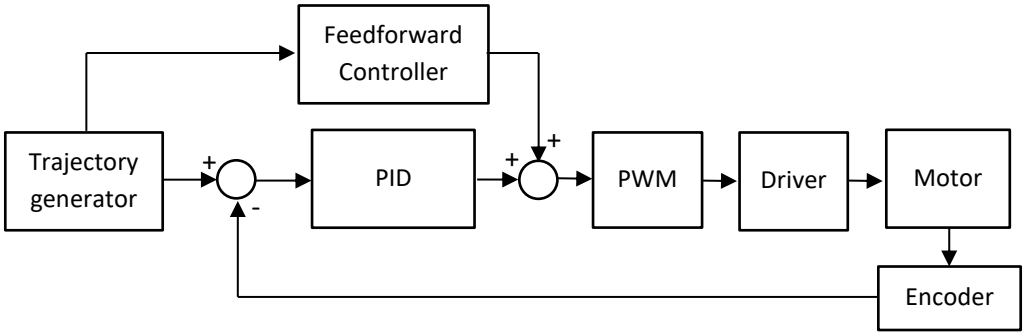


Figure 2.2. Control loop diagram for position and velocity modes.

Table 2.1. Commands and axis variables associated with position mode.

	Associated command	Associated axis variable
Trajectory generator	Input: MR, MA, SV, SA Output: TT, TO	Desp, PV, MPV, Ack, Carp
PID	Input: DB, SE, SS, SG, SI, SD, FR, IL Output: TF, TG, TI, TD	PGAIN, IGAIN, DGARIN, IL, INTRVL, SMPCNT, IINTRVL, ISMPCNT, PERR, OERR, MAXERR, DBAND, DERR
Feedforward controller	Input: FA, FV	FAGAIN, FVGAIN
Driver	Input: SQ, OO Output: TQ	THRO, BIAS, TLMTPL, TLMTMI
Motor	N/A	IMON
Encoder	Output: TP, TV	Curp, V

2.2.4.2 Velocity mode

Figure 2.2 depicts the control loop diagram that is used when the velocity mode is enabled through the VM command, whereas **Table 2.2** presents the associated commands and axis variables.

Table 2.2. Commands and axis variables associated with velocity mode.

	Associated command	Associated axis variable
Trajectory generator	Input: DI, SV, SA Output: TT, TO	PV, MPV, Ack
PID	Input: DB, SE, SS, SG, SI, SD, FR, IL Output: TF, TG, TI, TD	PGAIN, IGAIN, DGARIN, IL, INTRVL, SMPCNT, IINTRVL, ISMPCNT, PERR, OERR, MAXERR, DBAND, DERR
Feedforward controller	Input: FA, FV	FAGAIN, FVGAIN
Driver	Input: SQ, OO Output: TQ	THRO, BIAS, TLMTPL, TLMTMI
Motor	N/A	IMON
Encoder	Output: TP, TV	Curp, V

2.2.4.3 Torque mode, voltage-based

Figure 2.3 depicts the control diagram that is used when the voltage-based torque mode is enabled through the QM0 command, whereas **Table 2.3** presents the associated commands and axis variables.

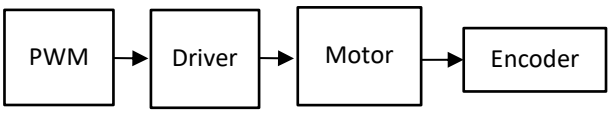


Figure 2.3. Control diagram for torque mode, voltage-based.

Table 2.3. Commands and axis variables associated with torque mode, voltage-based.

	Associated command	Associated axis variable
Driver	Input: SQ, OO Output: TQ	THRO, BIAS, TLMTPL, TLMTMI
Motor	N/A	IMON
Encoder	Output: TP, TV	Curp, V

2.2.4.4 Torque mode, current-based

Figure 2.4 depicts the control loop diagram that is used when the current-based torque mode is enabled through the QM1 command, whereas **Table 2.4** presents the associated commands and axis variables.

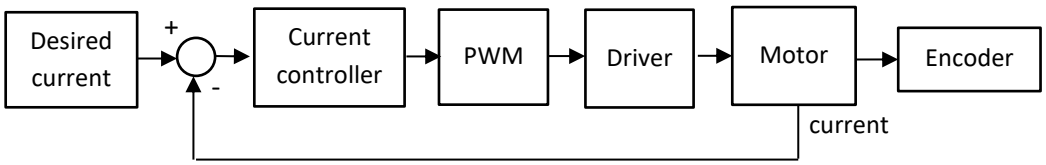


Figure 2.4. Control loop diagram for torque mode, current-based.

Table 2.4. Commands and axis variables associated with torque mode, current-based.

	Associated command	Associated axis variable
Desired current	Input: SQ	QCMD, IMON
Current controller	Input: SC	N/A
Driver	Input: SQ, OO Output: TQ	THRO, BIAS, TLMTPL, TLMTMI
Motor	N/A	IMON
Encoder	Output: TP, TV	Curp, V

3 Program commands

3.1 Parameter commands

The parameter setting commands are considered to be those for setting the operating conditions of the servo system (i.e. PID filter gains, velocity, acceleration and etc.).

Command: aDBn -- Dead-Band --

Argument: 0 <= n <= 16383

Default: 0

This command sets the position following error dead-band for servo axis ‘a’. The purpose for the DB command is to allow an acceptable static position error for which there will be no restoring force. This has the effect of reducing or eliminating "hunting" which is the continuous movement at or about a position in trying to seek that position. This is useful for applications that cannot tolerate this condition. Please note that the DB command is only in effect when the servo is not in motion (when the Trajectory Complete bit is set in the servo status word).

Related Commands: TF

Command: aFAn -- Feed-forward, Acceleration --

Argument: 0 <= n <= 32767

Default: 0

This command allows for the adjustment of the PID digital filter acceleration feed-forward term for servo axis ‘a’. During the course of a Position Mode (PM) or Velocity Mode (VM) move, at any point during acceleration or deceleration (with a consistent load), the ideal required value of the servo output is fairly consistent and somewhat predictable.

During acceleration or deceleration:

OUTPUT = (VELOCITY × FV_CONSTANT) + (ACCELERATION × FA_CONSTANT)

During constant velocity:

OUTPUT = (VELOCITY × FV_CONSTANT)

If this value can be dynamically predicted and summed with the output of the PID digital filter, in effect, it reduces the burden of the PID filter to make lead/lag corrections based of the following error, thereby enhancing performance.

Related Commands: FV, OO

Command: aFRn-- Set Derivative Sampling Period --

Argument: 0 <= n <= 127

Default: 0

This command allows for the adjustment of the derivative sampling interval for servo axis ‘a’. The period of this interval can be calculated by the following:

T = (n+1) × S × 0.000100

where "T" is the period in seconds, "n" is the FR command argument and "S" is the sample period count specified by the Servo Speed (SS) command. For example, if the value previously set by the SS command is 10 and the value set by the FR command is 1, then the derivative sample period will be:

(1+1) × 10 × 0.000100 = .002000 S or 2 mS

This command is useful in tuning the PID servo loop to the inertial properties of the system.

Related Commands: RI, SS

Command: aFVn -- Feed-forward, Velocity --

Argument: 0 <= n <= 32767

Default: 0

This command allows for the adjustment of the PID digital filter velocity feed-forward term for servo axis ‘a’.

During the course of a Position Mode (PM) or Velocity Mode (VM) move, at any point along the way (with a consistent load), the ideal required value of the servo output is fairly consistent and somewhat predictable.

During acceleration or deceleration:

OUTPUT = (VELOCITY × FV_CONSTANT) + (ACCELERATION × FA_CONSTANT)

During constant velocity:

OUTPUT = (VELOCITY × FV_CONSTANT)

If this value can be dynamically predicted and summed with the output of the PID digital filter, in effect, it reduces the burden of the PID filter to make lead/lag corrections based of the following error, thereby enhancing performance.

Related Commands: FA, OO

Command: aGRn -- Gear Ratio –

Argument: -8388607 <= n <= 8388607

Default: 0

This command sets the electronic gearing ratio for axis ‘a’. A negative argument will cause a direction reversal in the electronic gearing. The argument to this command is the desired gearing ratio scaled by 65536.

Examples:

```
GR65536 ; Slave gear ratio is 1:1
GR131072 ; Slave gear ratio is 2:1
GR32768 ; Slave gear ratio is .5:1
GR6554 ; Slave gear ratio is .1:1 (actual gear ratio
is .100006103516:1).
GR-65536 ; Slave ratio is 1:1 with direction reversal
```

Related Commands: EG

Command: aILn -- Set Integration Limit –

Argument: 0 <= n <= 16383

Default: 0

This command clamps the level of influence that the PID digital filter integral term can use to reduce the static position error thereby reducing integral “wind up” of servo axis ‘a’. When properly adjusted, this can enhance loop stability and operation. The Integral Limit (IL) and Set Integral Gain (SI) must both be set to a non-zero value in order for the integral term to have any effect.

Related Commands: SI

Command: aOMn -- Output Mode –

Argument: 0 <= n <= 255

Default: 0

This command allows the user to determine what data gets sent to the D/A analog output for a given axis. The upper four bits of the argument are for redirection of data and **determine from which axis the D/A channel will get its data**. This allows both D/A channels to output data from the same axis. If no redirection is specified, the default data used is that of the current axis.

Note: When outputting the servo output command, if no redirection is specified, then the output command as phase adjusted by the PH command will be output. If redirection is specified, then the normalized data as reported by the TQ command will be output.

'n'	Value output	Range
0	Servo output command	-32767 to 32767 (0 to 10 V)
1	Servo following error	-2047 to 2047 (0 to 10 V)
2	Servo following error * 64	-131008 to 131008 (0 to 10 V)
3	Variable USER1	-2047 to 2047 (0 to 10 V)
4	Low bits of encoder position	-2047 to 2047 (0 to 10 V)

'n+'	Redirect Channel
0	Default
16	1
32	2

Examples:

```
1OM0 ; Send axis 1 servo output command to D/A channel 1.
1OM16 ; Send axis 1 servo output command to D/A channel 1.
2OM0 ; Send axis 2 servo output command to D/A channel 2.
2OM32 ; Send axis 2 servo output command to D/A channel 2.
1OM0 ; Send axis 1 servo output command to D/A channel 1.
2OM17 ; Send axis 1 following error to D/A channel 2.
```

Command: aOOn -- Output Offset –

Argument: -32767 <= n <= 32767

Default: 0

This command allows the user to set a continuous output for servo axis 'a'. In certain applications, such as an overhanging load, there will be a continuous burden placed upon a servo axis. In cases like these, where there is a predictable load, the OO command can be used to provide a continuous restoring force that will be combined with the output of the PID digital filter. This has the effect of improving the performance of the PID digital filter in that because it is not saturated with static load, it has a better dynamic response to load disturbances.

Related Commands: FA, FV, OM

Command: aPHn -- Set Servo Phasing –

Argument: 0 <= n <= 63

Default: 0

This command is used to set the output polarity, encoder phasing and index input sense for servo axis ‘a’. The polarity of the output will determine whether the servo is driven in a direction that reduces or increases position error. The encoder phase will determine whether the position count will increase or decrease for a valid encoder input sequence. The Index sense determines what logic edge will cause the Index input to be active.

To determine the argument to be used with the PH command, use the follow table and add the required values together.

	Add to 'n'
Output Phase Normal	0
Output Phase Reversed	1
Encoder Phase Normal	0
Encoder Phase Reversed	2
Index Active Level Low	0
Index Active Level High	4
Home Sense Active ON	0
Home Sense Active OFF	8

Example:

If it were necessary to reverse the encoder phasing and to set the Home sense active “OFF”, then n = 2 + 8 = 10.

Related Commands: SP

Command: aRIn -- Sampling Rate of Integral –

Argument: 0 <= n <= 127

Default: 0

This command allows for the adjustment of the PID digital filter integral sampling interval for servo axis ‘a’. The period of this interval can be calculated by the following:

$$T = (n+1) \times S \times 0.000100$$

where "T" is the period in seconds, "n" is the RI command argument and "S" is the sample period count specified by the Servo Speed (SS) command. For example, if the value previously set by the SS command is 10 and the value set by the RI command is 1, then the integral sample period will be:

(1+1) × 10 × 0.000100 = .002000 S or 2 mS

This command is useful in tuning the PID servo loop to the inertial properties of the system.

Related Commands: FR

Command: aSAn -- Set Acceleration –

Argument: 0 <= n <= 1073741823

Default: 0

This command sets the acceleration rate for servo axis ‘a’. The 32 bit argument to this command is scaled by 65536. This number determines how much the servo's velocity will be altered by each servo loop interval (determined by the Servo Speed "SS" command) while it is accelerating or decelerating. If this command is executed during a Position Mode move, it will be ignored.

Example:

Encoder: 500 lines or 2000 Counts/Rev

Desired Acceleration: 75 Rev/Sec^2

Servo Loop Interval: 1,000 Hz

9830.4 = ((75 Rev/Sec^2 × 2000 Counts/Rev) / (1000 Hz)^2) × 65536

To achieve an acceleration of 75 Rev/Sec2, the command SA9830 would be issued. A simpler way to calculate the acceleration argument would be to determine a constant for your application by which to calculate desired acceleration.

131.072 = K = ((1 Rev/Sec^2 × 2000 Counts/Rev) / (1000 Hz)^2) × 65536

9830.4 = 75 Rev/Sec^2 × K

Please note that if the Set Acceleration (SA) command is used with an argument of "0", then you have commanded the velocity to change in steps of zero which means if the servo is stopped it will not be able to move, and if the servo is moving it will not be able to change velocity.

Related Commands: SS, SV

Command: aSCn -- Set Current Mode Gain –

Argument: 0 <= n <= 32767

Default: 0

This command sets the "current mode" gain for servo axis ‘a’ used by Torque Mode (QM1 only, see QM command). This allows the response of the current error integrator to be adjusted to suit a given application. A low SC value will provide slow response to load changes while a high SC value will provide quick response to load changes. If SC is set is too low then inadequate control may occur. If the SC setting is too high then the system may be unstable. A good "general" value for SC is about 8000.

Related Commands: QM, SQ

Command: aSDn -- Set Derivative Gain –

Argument: 0 <= n <= 32767

Default: 0

This command sets the derivative gain term of the PID digital filter loop for servo axis ‘a’.

Related Commands: SG, SI, IL, FR

Command: aSGn -- Set Proportional Gain –

Argument: 0 <= n <= 32767

Default: 0

This command sets the proportional gain term of the PID digital filter loop for servo axis ‘a’.

Related Commands: SI, SD, IL

Command: aSIn -- Set Integral Gain –

Argument: 0 <= n <= 32767

Default: 0

This command sets the integral gain term of the PID digital filter loop for servo axis ‘a’. The Set Integral Gain (SI) and Integral Limit (IL) must both be set to a non-zero value in order for the integral term to have any effect.

Related Commands: SG, SD, IL

Command: aSQn -- Set [Maximum] Torque Level –

Argument: -32767 <= n <= 32767 in Torque Mode, voltage-based (QM0)

-2047 <= n <= 2047 in Torque Mode, current-based (QM1)

0 <= n <= 32767 in Position Mode (PM)

0 <= n <= 32767 in Velocity Mode (VM)

Default: 32767

For servo axis 'a', this command sets the maximum output level when in Position Mode (PM) or Velocity Mode (VM) and sets the desired output level when in Torque Mode (QM). When this command is issued in Position Mode (PM) or Velocity Mode (VM), its limiting effect will remain, in all modes of operation, until again changed. Note that an argument less than zero is allowed only in Torque Mode (QM).

For voltage-based torque mode, position mode and velocity mode, SQ value of 0 - 32767 linearly corresponds to 0 V – (VLC25 power supply voltage). Note however that in VLCs, SQ value is effectively clamped at 86.6% of the power supply duty voltage. Meaning that when SQ is set to a value that exceeds 0.866 x 32767 = 28376 (minus or plus), the actual SQ value will be limited to that value.

For current-based torque mode, SQ value of 0 - 2047 linearly corresponds to 0 A – 13 A, unless an optional non-standard current resolution mod is implemented upon request on the VLC.

Examples:

```
>PM,SQ20000<CR>; Set maximum output to +-20000.
>VM,SQ10000<CR>; Set maximum output to +-10000.
>QM,SQ20000<CR>; Set output to forward 20000.
>QM,SQ-10000<CR>; Set output reverse 10000.
>PM,SQ10000<CR>; Set maximum output to +-10000.
>QM0,SQ15000<CR>; Set output to forward 15000 but
                 ; will only achieve output of 10000
                 ; due to previous command.
```

Related Commands: PM, QM, SC, VM

Command: SSn -- Set Servo Speed –

Argument: 1 <= n <= 62

Default: 2

This command sets the servo loop interval. The period of this interval can be calculated by the following:

$T = (n) \times 0.000100$

where "T" is the period in seconds and "n" is the SS command argument.

For example, if the value set by the SS command is 10 then the servo loop period will be:

10 × 0.000100 = .001000 S or 1 mS

Although this command will accept a parameter of 0 or 1, it will be clamped to a minimum servo loop rate of 100 µS.

Using the SS command affects the Set Velocity (SV) and Set Acceleration (SA) commands in that they both are specified in terms of servo loop intervals. For example, if the servo loop rate is doubled, the velocity and acceleration would appear to have been halved. It should also be noted that the "SS" command will most likely affect the required settings of the PID digital filter.

Related Commands: SA, SV

Command: aSVn -- Set Maximum Velocity –

Argument: 0 <= n <= 1073741823

Default: 0

This command sets the desired velocity (in quadrature counts per servo loop interval) for servo axis ‘a’. The 32 bit argument to this command consists of a 16 bit integer part and a 16 bit fractional part.

Example:

Encoder: 500 lines, 2000 Counts/Rev

Desired Velocity: 40 Rev/Sec

Servo Loop Interval: 1,000 Hz

5242880 = ((40 Rev/Sec × 2000 Counts/Rev) / 1000 Hz) × 65536

To achieve a velocity of 40 Rev/Sec, the command SV5242880 would be issued. A simpler way to calculate the velocity would be to determine a constant for your application by which to calculate desired velocity.

131072 = K = ((1 Rev/Sec × 2000 Counts/Rev) / (1000 Hz)) × 65536

5242880 = 40 Rev/Sec × K

Related Commands: SA, SS

3.2 Reporting commands

The reporting commands output data relevant to the operating status of the VLC-25. Numerical output will be in the mathematical base determined by the Decimal Mode (DM) / Hexadecimal Mode (HM) commands and output will be followed by a carriage return and linefeed.

Command: TAn -- Tell Analog To Digital Channel 'n' –

Argument: 0 <= n <= 15

This command will cause a conversion on A/D channel 'n' and will display the result. 'n' refers to different analog input channels presented in the table below.

n	Signal	Range
0	Axis 1direct motor current (Id)	0 to +/- 2047 (0 to +/- 13 A)
1	Axis 1 phase A/U motor current (Ia)	0 to +/- 2047 (0 to +/- 13 A)
2	Axis 1 Phase B/V motor current (Ib)	0 to +/- 2047 (0 to +/- 13 A)
3	Axis 1 Quadrature motor current (Iq)	0 to +/- 2047 (0 to +/- 13 A)
4	Axis 2 direct motor current (Id)	0 to +/- 2047 (0 to +/- 13 A)
5	Axis 2 phase A/U motor current (Ia)	0 to +/- 2047 (0 to +/- 13 A)
6	Axis 2 Phase B/V motor current (Ib)	0 to +/- 2047 (0 to +/- 13 A)
7	Axis 2 Quadrature motor current (Iq)	0 to +/- 2047 (0 to +/- 13 A)
8	User analog input 0 (differential)	0 to +/- 2047 (+/-10 V)
9	User analog input 1 (differential)	0 to +/- 2047 (+/-10 V)
10	User analog input 2 (single-ended)	0 to 4095 (0 to +10 V)
11	User analog input 3 (single-ended)	0 to 4095 (0 to +10 V)
12	User analog input 4 (single-ended)	0 to 4095 (0 to +10 V)
13	Bus voltage monitor (VMON)	0 to 4095 (0 to 48 V)
14	Axis 1 power stage thermistor value	Same as axis variable TTEMP
15	Axis 2 power stage thermistor value	Same as axis variable TTEMP

Related Commands: GA

Command: aTB -- Tell Breakpoint –

Default: "None"

This command reports the last programmed breakpoint value specified by the Interrupt on Position Absolute (IP) or the Interrupt on Position Relative (IR) commands for servo axis 'a'. If no breakpoint value has yet been specified then the word "NONE is returned.

Related Commands: IP, IR

Command: TCn -- Tell State of I/O Channel ‘n’ –

Argument: 0 <= n <= 63

Default: “Off”

This command reports the current state of the I/O channel specified by 'n' and what type of logic the channel is (either active "on" or active "off"). The display is formatted as follows:

/xx = aaa

The presence of the "/" character indicates that a channel is active in the "off" state. A lack of the "/" character indicates that the channel is active in the "on" state. The "xx" indicates the channel number and the "aaa" indicates the channel's current state, either "ON" or "OFF". The following examples show all the possible combinations:

```
01 = ON ; Channel 1 is "ON" in the active "ON” state.
/01 = ON ; Channel 1 is "ON" in the active "OFF" state.
01 = OFF ; Channel 1 is "OFF" in the active "ON" state.
/01 = OFF ; Channel 1 is "OFF" in the active "OFF" state.
```

Related Commands: CF, CH, CL, CN

Command: aTD -- Tell Derivative Gain –

This command reports the derivative gain value of the PID digital filter for servo axis ‘a’.

Related Commands: TG, TI, TL

Command: TE -- Tell Error –

This command reports the last error code caused by any command or macro. If no error has occurred then a "0" will be reported. Once the TE command has been issued, then the error code will be reset to zero. The TE command is useful for determining what error occurred in the case that no display was connected at the time.

Command: aTF -- Tell Following Error –

This command reports the following error for servo axis ‘a’. This value is the difference between the current desired temporal position (or that which is reported by the Tell Optimal (TO) command) of the trajectory generator and the servo’s current real position (or that which is reported by the Tell Position (TP) command).

Related Commands: DB

Command: aTG -- Tell Proportional Gain –

This command reports the proportional gain value of the PID digital filter for servo axis ‘a’.

Related Commands: TI, TD, TL

Command: aTI -- Tell Integral Gain –

This command reports the integral gain value of the PID digital filter for servo axis ‘a’.

Related Commands: TG, TD, IL

Command: aTKn -- Tell (K) Constants –

Argument: 0 <= n <= 1

This command will display a number of internal settings depending on the value specified by 'n'. If 'n' is "0" or is not specified, then various parametric values for servo axis ‘a’ will be displayed. If 'n' is "1", then various system parameters will be displayed.

Example display for the command "1TK0":

```
>TK0

Parameter Values for Axis [1]

Proportional Gain ---------- (SG) = 0
Integral Gain ------------- (SI) = 0
Derivative Gain ------------ (SD) = 0
Integral Limit ------------- (IL) = 0
Current Gain --------------- (SC) = 0
Velocity Feed-forward Gain - (FV) = 0
Accel. Feed-forward Gain --- (FA) = 0
Output Offset -------------- (OO) = 0
Position Error Dead-Band --- (DB) = 0
Maximum Following Error ---- (SE) = 16383
Integral Sample Rate ------- (RI) = 0
Derivative Sample Rate ----- (FR) = 0
Phase and Sense Settings --- (PH) = 0
Maximum Velocity ----------- (SV) = 0
Acceleration --------------- (SA) = 0
Desired Direction ---------- (DI) = 0
Torque (output) Limit ------ (SQ) = 32767
Counts/Electrical Cycle ---- (EC) = 0
Commutation Phasing -------- (SP) = 0
```

Example display for the command "TK1":

```
>TK1

System Parameter Settings (group 1).

Axis 1 Enabled ------------- (EA) = Yes
Base 16 Input & Output -- (HM/DM) = Off
Character Echo ---------- (EN/EF) = On
Handshake --------------- (HN/HF) = Off
Fail -------------------- (FN/FF) = Off
Servo Loop Rate ------------ (SS) = 2
Input Debounce/Delay ------- (ID) = 0
Phase and Sense Settings --- (CV) = 0
Intr. Vector Enable, HIGH (EV/DV) = 0
Intr. Vector Enable, LOW  (EV/DV) = 0
Firmware Revision ---------- (VE) = 1.52
Macro Memory Remaining ---------- = nnnn
```

Command: aTL -- Tell Integral Limit –

This command reports the integration limit value of the PID digital filter for servo axis ‘a’.

Related Commands: TG, TI, TD

Command: TMn -- Tell Macros –

Argument: -2 <= n <= 255

The Tell Macro (TM) command will display the commands which make up any macros that have been defined. If 'n' >= 0 and 'n' <= 255, then the macro specified by 'n' be displayed. If 'n' = - 1, then all the macros will be displayed preceded by the individual macro number. If 'n' =-2, then all macros will be displayed and will be preceded by the letters "MD" and the individual macro number. This is useful when downloading programs from the VLC-25 to a computer in that you need not re-enter the Macro Define (MD) command and number at the beginning of each macro.

Related Commands: MD, RM

Command: aTO -- Tell Optimal Position –

This command reports the desired position for servo axis ‘a’. This value may be different from the position reported by the TP command if the servo is moving or the controller is unable to drive the servo.

Related Commands: TT, TP

Command: aTP -- Tell Real Position –

This command reports the absolute position of servo ‘a’. It may be used to monitor motion during both the Motor On (MN) and the Motor Off (MF) states.

Related Commands: TT, TO

Command: aTQ -- Tell Torque –

This command reports the current output torque being commanded by the servo axis ’a’. This value will be either that which is generated by the PID output or which is set by the user via the Torque Mode (QM).

Command: TRn -- Tell Contents of Register ‘n’ –

Argument: 0 <= n <= 2047

This command reports the value contained by Register 'n'.

Related Commands: Register Commands

Command: aTS -- Tell Status Word –

This command reports the operating status word of servo axis ‘a’. The response is coded into a single 32 bit value. The meaning of each bit is listed below:

Bit	Purpose
0	**Servo Enabled**. This bit will be set to “1” when the servo is enabled via the Motor On (MN) command. A “0” indicates that the servo has been disabled via one of the following reasons: a Motor Off (MF) command, a servo error due to excessive following error or over temperature condition, an external fault.
1	**Servo Error.** A “1” in this bit position indicates that the maximum servo following error has been exceeded.
2	**Over Temperature / Fault.** A “1” in this bit position indicates that an over-temperature condition has occurred or the external fault input has been activated.

3	**Breakpoint Reached.** A "1" in this bit position indicates that a previously defined breakpoint has been reached. This bit will be reset by any of the following commands: Motor On (MN), Interrupt on Position (IR), Interrupt on Relative Position (IR).
4	**Trajectory Complete.** A "1" in this bit position indicates that the servo has completed a commanded move. A "0" indicates that the servo is busy executing a commanded move.
5	**Servo Stopping.** A "1" in this bit position indicated that the servo has been commanded to stop. Upon stopping, this bit is then cleared.
6	**Current Direction.** This bit indicated the current direction of travel for the servo. (0 = Positive, 1 = Negative)
7	**Desired Direction.** This bit indicates the direction commanded by the Direction (DI) command. (0 = Positive, 1 = Negative)
8	**Reserved.**
9	**Output Phasing.** This bit indicates the output phasing as set by the Phase (PH) command. (0 = Normal, 1 = Reversed)
10	**Looking for Index.** A "1" in this bit position indicates that the controller is currently watching for an index pulse to occur as commanded by the Find Index (FI) command.
11	**Looking for Edge.** A "1" in this bit position indicates that the controller is currently watching for the coarse home input to go active as commanded by the Find Edge (FE) command.
12	**Reserved**
13	**Coarse Home Input Active.** A "1" in this bit position indicates that the coarse home input is active.
14	**Capture Index Flag.** A "1" in this bit position indicates that the Find Index (FI) command is trying to capture the position on occurrence of an index pulse as opposed to initializing the position.
15	**Bad Input.** A "1" in this bit position indicates that invalid input has been entered via the Variable Input (VI) command. A "0" indicates that the last data obtained via the VI Command was valid.
16	**Accelerating.** A "1" in this bit position indicates that the servo is currently accelerating.
17	**Position Mode.** A "1" in this bit position indicates that the servo is currently operating in position mode.
18	**Velocity Mode.** A "1" in this bit position indicates that the servo is currently operating in velocity mode.
19	**Torque Mode.** A "1" in this bit position indicates that the servo is currently operating in (voltage) torque mode.
20	**Current Mode.** A "1" in this bit position indicates that the servo is currently operating in (current) torque mode).
21	**Reserved.**
22	**Electronic Gearing Mode.** A "1" in this bit position indicates that the servo is currently operating in electronic gearing mode.
23-31	**Reserved.**

Command: aTT -- Tell Target Position –

This command reports the current target position of servo axis ‘a’. This is the absolute position to which the servo was last commanded to move. It may have been specified directly with the Move to Position (MP) or Move Absolute (MA) commands or indirectly with the Move Relative (MR) command. If the servo axis is in Velocity Mode (VM), then the target position will track the current optimal position (or that which is reported by the Tell Optimal (TO) command).

Related Commands: TO, TP

Command: aTV -- Tell Current Velocity –

This command reports the trajectory generator current velocity for servo axis ‘a’. The value reported has the same units as the Set Velocity (SV) command. See the description of that command for further details.

Related Commands: SV, SS

Command: VE -- Tell Version –

This command reports the firmware revision level. This revision level exists as a code in the internal RAM memory (see Register Commands). This code should be interpreted as the first byte being the major revision number and second byte being the minor revision number. This allows user programs to determine the firmware revision for compatibility purposes.

Related Commands: Register Commands

3.3 Motion commands

Motion Commands are those commands that involve or cause the actual movement of a servo. Any position mode (PM) based motion will be carried out using a trapezoidal velocity profile with the maximum velocity determined by the Set Velocity (SV) command. The acceleration and deceleration are the same and both are determined by the Set Acceleration (SA) command.

During the execution of a Position Mode based motion, **the target position (MA or MR) and the maximum velocity (SV) may be changed at any time, however, changes to the acceleration (SA) will be ignored.** If a velocity mode (VM) based motion is under way, the maximum velocity (SV) and the acceleration (SA) may be changed at any time. During this type of motion, the target position will continuously be set equal to the current optimal position. If a torque mode (QM) based motion is under way, the torque setting (SQ) may be changed at any time. During this type of motion, the target and optimal positions will continuously be set equal to the current real position. In either PM, VM or QM, a Stop (ST) or Abort (AB) command will cause motion for the specified servo axis to cease. If a servo axis is in Position Mode (PM) and is commanded into Velocity Mode (VM), the motion will continue (only with no target destination) at the current maximum velocity set by the Set Velocity (SV) command. If a servo axis is in PM or VM and commanded into Torque Mode (QM), the servo output will be reduced to zero, and the unit will be placed in QM. If a servo axis is in VM and commanded into PM, the servo will be stopped at the current acceleration rate and placed into the PM mode. If a servo axis is in QM and commanded into VM, it will attempt to instantly assume the last known maximum velocity and remain in VM. If a servo axis is in QM and commanded into PM, the output will be reduced to zero, and the servo will begin station keeping in PM.

There is another mode of operation referred to as "Electronic Gearing Mode". This mode is a supplement to the position based modes of operation. Electronic gearing causes the servo axis for which it is in effect (or the slave axis), to move proportionally to the servo axis it is tracking (or the master axis) in conjunction with it's own motion profile. The proportion of the slave axis it set by the Gear Ratio (GR) command.

When the EG command is issued, the relative position of the master axis (from the time of the EG command), is monitored by the slave axis. This relative motion is multiplied by the value set with the GR command and the product of which is added to the optimal temporal position of the slave axis. The result of this is to allow the motion profile of the master servo axis to be superimposed on the slave axis' motion profile. While the EG mode is primarily intended for use with axis' that are operating in position mode (PM), it will also work with axis' that are operating in velocity mode (VM).

Command: aAB -- Abort Motion –

This command causes an emergency stop on servo axis 'a'. The servo stops abruptly but leaves the servo control loop enabled. The target position is changed to be equal to the present position. The command is used for stopping an undesired motion.

Related Commands: ST

Command: aCI -- Capture on Index –

This command causes the position counter of servo axis ‘a’ to be stored in the internal variable HREG upon receipt of an index pulse. For more information on HREG, see the Register Commands and the listing of the internal variables.

Related Commands: FI, WI

Command: aDAn -- Define Absolute Home –

Argument: -1073741832 <=n <= 1073741832

Note: only relevant for 3-phase motors/actuators

For servo axis ‘a’, this command defines the commutation offset angle between electrical phase home and mechanical home, which can be used for phasing of 3-phase motors. To produce meaningful torque, the offset angle should be equal to 90 electrical degree, therefore

$$DA = (EC/4) + (SP/65536) \times EC$$

This command is used as a part of the phasing. First, the shaft position has to be “locked” somewhere along the stroke using the command

SPx,QM0,MN,SQy

Where x and y values are to be determined by the user, such that the above shaft condition is sufficed. Next, DA value based on the previous equation is set and also EC value has to be set properly. Afterwards, the phasing is done and the motor is ready to be operated.

Related commands: EC, SP

Command: aDHn -- Define Home –

Argument: -2147483647 <=n <= 2147483647

This command causes the current position of servo axis ‘a’ to be defined as 'n'. From then on, all positions used and reported for the servo axis will be relative to that physical position.

Related Commands: FE, FI

Command: aDIn -- Set Direction –

Argument: n = 0 for positive, n = 1 for negative

Default: 0

This command sets the move direction for servo axis ‘a’ when in velocity mode. Issuing this command with an argument of "0" will cause the servo to move in a positive direction while an argument of "1" will cause it to move in a negative direction.

Related Commands: SA, SV, VM

Command: aEAn -- Enable Axis for Operation –

Argument: n = 0 for axis enable, n = -1 for axis disable

Default: 0

This command will enable/disable operation of servo axis ’a’, depending on the ‘n’ value. If an axis is not enabled for operation, no system resources will be expended on it. For example, this means that if an axis is going to be used for it’s encoder interface only, it must still be enabled for operation via the EA command, however, the undedicated encoder interface is always enabled and does not apply here.

Before using the EA command, one should be sure that the Set Servo Speed (SS) command has been used such that the servo loop time is long enough to service the newly enabled axis as well as any previously enabled axis.

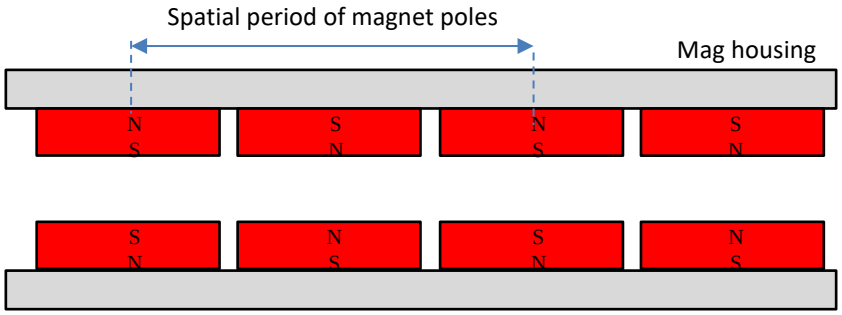
Command: aECn -- Electronic Commutation Counts –

Argument: 6 <= n <= 4294967295

Note: only relevant for 3-phase motors. For 1-phase motors, EC must be equal to 0.

For servo axis 'a', this command sets the number of encoder counts for 1 electrical cycle of the electronic commutation. EC value is to be set after phasing has been performed. Whenever EC $\neq$ 0, SP value will automatically change depending on the motor shaft position, as a result of the electronic commutation.

For 3-phase linear actuators:
EC = spatial period of magnet poles (in encoder counts). See below.



Below are EC values of selected SMAC 3-phase linear actuators (for SMAC actuators not listed below, please contact SMAC for the EC value):

Actuator	5 micron encoder resolution	1 micron encoder resolution
LCR13/LCR16	3780	18900
LCA25	10400	52000
LCA50	21000	105000
LBL25	5960	29800

For 3-phase rotary motors:
EC = (Encoder counts per revolution)/(number of magnet pole pairs)

If the EC calculation of 3-phase rotary motors results in a non-integer value (for instance, EC = 800/6 = 133.33), EC value has to be set according to the following equation:
EC = ((number of pole pairs-1) <<28) + (encoder counts per revolution)

Where "<<" is a bitwise left shift operator. Note that due to 32-bit math limitations:

1. Maximum number of pole pairs = 16
2. The limit on the maximum absolute encoder position is 2^31 divided by the number of pole pairs
3. The maximum encoder counts per revolution is 268435456 (before firmware version 1.53, the limit goes up to 32767)

Related commands: DA, SP

Command: aEGn -- Electronic Gearing Mode –

Argument: -2 <= n <= 2

Default: Position Mode

This command causes (slave) servo axis ’a’ to begin to move proportionally to (master) servo axis ’n’. If ’n’ is "0" then the electronic gearing mode for axis ’a’ will be disabled. If ’n’ is positive then the slave axis will track the master axis’ optimal position (or that which is reported by the TO command). If ’n’ is negative then the slave axis will track the master axis’ real position (or that which is reported by the TP command). It is preferable that the EG command only be used when the target (or slave) axis is not in motion.

Related Commands: GR

Command: aFEn -- Find Edge of Coarse Home Input –

Argument: -2147483647 <=n <= 2147483647

This command is used to initialize servo axis ’a’ at a given position. It will remain in effect until the Home input goes active. At that time, the current position of the servo will be defined as ’n’. This command will neither start nor stop any servo motion. It is up to the user to initiate servo motion before issuing the command and to stop any motion after the command is completed.

Related Commands: WE

Command: aFIn -- Find Edge of Index –

Argument: -2147483647 <=n <= 2147483647

This command is used to initialize servo axis ‘a’ at a given position. It will remain in effect until the Index input goes active. At that time, the current position of the servo will be defined as 'n'. This command will not start or stop any servo motions; that is up to the user. Since an index pulse may occur at numerous points of the servo's travel (once per revolution in rotary encoders), a typical application will require a home signal to establish a coarse position reference before the index pulse can be used to fine tune that reference.

Related Commands: CI, WI

Command: aGH -- Go Home –

This command causes servo axis ‘a’ move to absolute position 0. This is equivalent to a Move Absolute (MA) command when the destination is 0. This command will initiate motion; therefore, a Go (GO) command is not required.

Related Commands: MA, GO

Command: aGO -- Go (start motion) –

This command causes servo axis ‘a’ to begin motion. When in the Velocity Mode (VM), the axis will accelerate to a constant velocity. When in the Position Mode (PM) the axis will begin to seek the specified target position. The servo must be in the "on" state for motion to occur.

Related Commands: MA, MR, PM, VM

Command: aMAn -- Move Target Absolute –

Argument: -2147483647 <=n <= 2147483647

This command sets the target position of servo axis ‘a’ to absolute position 'n'. The absolute position is relative to the point of initialization (home or 0). As the Move Absolute (MA) command will not initiate a motion, a Go (GO) command must be used. The servo's target position will be adjusted whether the servo is on or off.

Related Commands: GO, MR, PM

Command: aMF -- Motor Off –

This command is used to place servo axis ‘a’ in the "off" state. The servo's output will go to the null level. This command can be used to prevent unwanted motion or to allow manual positioning of the unit. When the servo is turned off, the target and the optimal positions will follow the real position.

Related Commands: MN

Command: aMN -- Motor On –

This command is used to place the servo in the "on" state. When the servo is turned on, its target position is set to its current position so that the servo is not inclined to move. When this command is issued, it will disable the Index interrupt sources and will reset the Error, Fault, Breakpoint reached, Looking for Edge and Looking for Index bits in the status word for axis 'a'.

Related Commands: LM, LN, MF

Command: aMRn -- Move Target Relative –

Argument: -2147483647 <=n <= 2147483647

This command generates a relative target position of 'n' counts for servo axis ‘a’. Since the Move Relative (MR) command will not initiate a motion, a Go (GO) command must be used. The servo's target position will be adjusted whether the servo is on or off.

Related Commands: GO, MA, PM

Command: aPM -- Enter Position Mode –

Default: Position Mode

This command causes servo axis ‘a’ to enter the position mode of operation. In this mode, the servo can be commanded to move to a specific target position. The moves will be executed using a trapezoidal velocity profile based upon parameters set by the Set Velocity (SV) and the Set Acceleration (SA) commands.

Related Commands: EG, QM, VM

Command: aQMn -- Enter Torque Mode –

Argument: 0 for voltage mode, 1 for current mode

Default: Position Mode

This command places servo axis ‘a’ in the Torque Mode of operation. For mode 0 (or QM0), the output PWM duty cycle (or analog output for appropriate models) can be manually controlled by the user program. Once QM0 has been entered, the Set Torque (SQ) command can be used to set or change the servo output (see the SQ command). Bear in mind, that if the output was being limited by an SQ command before entering torque mode (while in PM or VM), it will remain limited while in QM and cannot be changed until the unit is in PM or VM.

For mode 1 (or QM1), the VLC-25 will use one of its A/D channels to monitor servo output current and attempt to maintain that output current at a steady level as commanded by the user program (see the SQ and SC commands). In that the A/D converter has 12-bit resolution, this will result in a value from 0 to +/- 2047 which represents the instantaneous output current with 0 being approximately 0 Amp and +/-2047 being approximately +/- 13 A: for VLC-25-13, with intermediate values being somewhat linear.
In the case of a motor (for example), when an output command of 397 is set (or about 2.5 amps for VLC-25-13), the PWM output begins to ramp up until the desired output current is reached. If the motor is not loaded enough to require 2.5 amps then the PWM will ramp up to 100% putting the maximum supply voltage to the motor. If at this point the motor is suddenly loaded, the PWM will ramp down to a level sufficient to supply 2.5 amps to the motor.

Mode QM1 will not function with the D/A module option.

Related Commands: EG, PM, VM

Command: aSEn -- Set Maximum Following Error –

Argument: 0 <= n <= 16383

Default: 16383

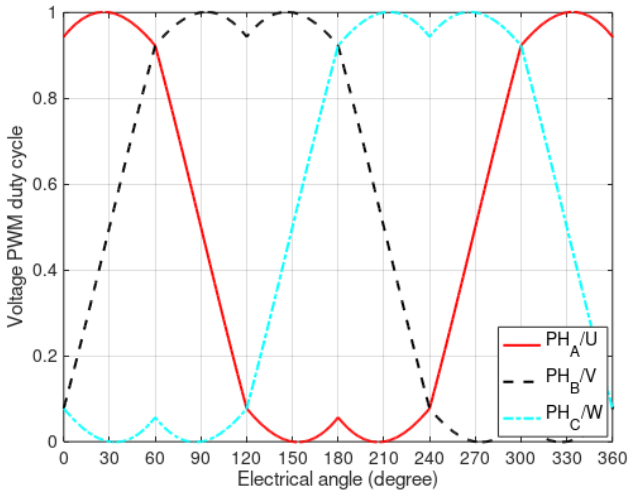
This command is used to set the maximum allowable following error (difference between the actual position and the optimal position) for servo axis ‘a’. If the following error exceeds the programmed value, the servo will be turned off (except in the case of the related interrupt being enabled) and the Error bit in the status word for axis ’a’ will be set. This bit will remain set until the servo is turned back on with the Motor On (MN) command.

Related Commands: DB, TF

Command: aSPn -- Set Commutation Phase --

Argument: 0 <= n <= 65535

For servo axis 'a', this command manually sets the commutation phase angle, where n = 0 and n = 65535 corresponds to 0 and 360 electrical degrees, respectively, following the voltage PWM duty cycle waveform (generated by the space vector modulation) shown below. Due to the 120^0 lag of phase B relative to phase A, increasing the SP value while energizing the coil will cause a retracting motion if standard SMAC actuator cabling is used. Therefore, PH1 command has to be executed to reverse this behavior.



Note that the behavior of SP command is also determined by the EC command. If EC = 0, as in the case of 1-phase motors or during the phasing process of 3-phase motors, SP can be used to set the commutation phase angle. As soon as EC $\neq$ 0, SP value is automatically adjusted by the commutation algorithm in the controller.

Examples for a 1-phase motor:

Use SP27307 (corresponding to the maximum PWM duty cycle or 150^0 electrical angle in the above graph) for positive SQ to output from B to A, resulting in a positive shaft displacement for QM0 mode, for standard SMAC actuators.

Use SP60075 (corresponding to the maximum PWM duty cycle or 330^0 electrical angle in the above graph) for positive SQ to output from A to B, resulting in a negative shaft displacement for QM0 mode, for standard SMAC actuators.

Positive SQ must produce a positive encoder position change. If it does not then the output phasing needs to be reversed with the PH command.

Brief phasing example using an LAR30-025-55C (1-phase motor)
For: Coil+ to PH_A_Out, Coil- to PH_B_Out, Encoder A to A, B to B

Extend causes increasing encoder count
SQ+ output is from PH_B_Out to PH_A_Out
```
>mf,ec0,sp27307,ph0,da0,dh0,qm0,mn,sq-6000,wa1000,tp,da,sq0
```

Brief phasing examples using an LCS25-150-15-6-3 (3-phase motor)
For: Coil_U to PH_A_Out, Coil_V to PH_B_Out, Coil_W to PH_C_Out
Encoder A to A, B to B
Extend causes increasing encoder count
```
>mf,ec0,sp0,ph1,qm0,mn,sq6000,wa500,da0,dh0,sp5461,wa500,tp,sp0,wa500,sq0
```

Related commands: DA, EC, PH

Command: aST -- Stop Motion –

This command is used to terminate motion on servo axis ‘a’. This command differs from the Abort (AB) command in that the servo will decelerate at its preset rate instead of stopping abruptly. When in Torque Mode, the outputs will be set to null.

Related Commands: AB, WS

Command: aVM -- Enter Velocity Mode –

Default: Position Mode

This command places servo axis ‘a’ in the Velocity Mode of operation. In this mode, the servo can be commanded to move in either direction to a maximum velocity. The servo will move in that direction until commanded to stop. When using Velocity Mode, the user must specify the direction for the servo to move using the Direction (DI) command. After specifying the desired maximum velocity and the desired direction and placing the servo in velocity mode, the servo can be started by issuing the Go (GO) command. While the servo is moving in Velocity Mode, the user can change the velocity by issuing new Direction (DI) and/or Set Velocity (SV) commands. The acceleration rate at which the servo's velocity will change, is determined by the Set Acceleration (SA) command. The acceleration can also be changed at any time.

Related Commands: EG, QM, PM

3.4 Register commands

The VLC-25 uses part of its flash memory to create a 2048 by 32 bit general purpose register space with Register "0" being referred to as the Accumulator.

The registers have many uses including storing data and parameters, performing mathematical operations and controlling command execution. The registers can be manipulated by several commands and can also be used to replace the argument in most commands. For example, if register "6" contains the value "-12000" and the following command is used...

```
MA@6,GO
```

it would use the contents of register "6" as the argument thus giving the same result as if the following command was issued...

```
MA-12000
```

Note that the use of the "@" character is what caused the command to assume a register argument (or indirect argument) instead of a direct argument. If the value following the "@" is not in the range of the 2048 registers (0 to 2047), an error will be reported. If the value contained by the register of the indirect argument is out of range, an error will also be reported.

As stated earlier, because the registers are within the flash memory, they are non-volatile and can be used as such. For example, if a register is incremented once every user program cycle, it can be used as an ongoing cycle counter for maintenance purposes, production accounting and etc.

3.4.1 Internal Variables

In certain applications, the user may find it necessary to use data pertinent to the internal operation of the VLC-25. This may be accomplished via the use of the Read Byte (RB), the Read Word (RW) and the Read Long (RL) commands which copy the VLC-25's internal RAM memory to the accumulator and the Write Byte (WB), the Write Word (WW) and the Write Long (WL) commands which copy the accumulator to the VLC-25's internal RAM memory. The listings below tell the location of and describe the internal variables that may be of use to the user.

WARNING: Randomly modifying these variables or other internal RAM not listed will most likely affect the operation of the VLC-25 resulting in unpredictable behavior or possible damage.

Axis Variable Descriptions

Status	Status word (TS command).
PV	Peak velocity (SV command).
MPV	Negative peak velocity (SV command).
V	Temporal velocity (TV command).
Desp	Desired position (TT command).
Carp	Temporal desired position (TO command).
Ack	Acceleration constant (SA command).
Curp	Current virtual position (TP command).
HREG	Holding register (CI,IP,IR commands).
IPPOS	Interrupt on position reference.
QI	Integral of current mode error (QM1 command).
PGAIN	Proportional term of PID filter (SG command).
IGAIN	Integral term of PID filter (SI command).
DGAIN	Derivative term of PID filter (SD command).
IL	Integral limit of PID filter (IL command).
CGAIN	Current mode gain (QM1 command).
FVGAIN	Velocity feed-forward of PID filter (FV command).
BIAS	Output offset (OO command).
THRO	Current servo output (TQ,SQ command).
QCMD	Current command (QM1,SQ command).
TLMTPL	Maximum positive servo output (SQ command).
FAGAIN	Acceleration feed-forward of PID filter (FA command).
PERR	Last calculated servo position error (TF command).
OERR	Old servo position error.
MAXERR	Maximum servo position error (SE command).
I	Servo error integral.
DERIV	Servo error derivative.
IMON	Servo output current monitor.
INTRVL	PID derivative sample interval (FR command).
SMPCNT	PID derivative sample counter.
IINTRVL	PID integral sample interval (RI command).
ISMPCNT	PID integral sample counter.
AXIS	Axis number lookup index.
ATYPE	Axis output type (OM command).
PHASE	Phase, sense and polarity information (PH command).
GMAxis	Gear mode "gear to" axis (EG command).
DBAND	Position error dead-band (DB command).
DERR	Position error with dead-band.
GMOFF	Gear mode initial position offset (EG command).
RATIO	Gear mode gear ratio (GR command).
USER1	User variable #1 (See OM command).
FCNT	Fault system counter.
FCMP	Fault system comparator.
TLMTMI	Maximum negative servo output (SQ command).
LPG1[1]	IA/IB ADC current input low-pass filter gain. Default value: 65535.
LPG2[1]	IMON low-pass filter gain. Default value: 32767.
CMPG[1]	Current mode (QM1) proportional filter gain. Default value: 0.

RCurp	Current real position.
TLMTMI	Maximum negative servo output (SQ command).
POff	Virutal position offset. Difference between RCurp and Curp
CPEC	Encoder Counts Per Electrical Cycle
COMPH	Current Commutation Phase. Set by SP or automatically.
IAOff	Phase A current monitor ADC Offset (2048 default).
IBOff	Phase B current monitor ADC Offset (2048 default).
i2t_NOM	i2t Nominal Current (range 0 to 2047, default is 2047).
i2t_TRIP	i2t Trip Level (range 0 to 2^{31}-1, default is 2^{31}-1).
i2t_INT	i2t Integrator (range 0 to 2^{31}-1, initializes to 0).
i2tINTRVL	i2t sample interval (range 0 to 255 with 0 being default).
i2tSMPCNT	i2t sample counter.
IA	Phase A output current (+/-2047)*.
IB	Phase B output current (+/-2047)*.
ID	dq0 Direct current (+/-2047)*.
IQ	dq0 Quadrature current (+/-2047)*.
TTRIP	Thermistor over-temperature trip point (default is 3880 = 70 C).
TTEMP	ADC Thermistor value (see the table in section 4.1).

Note: [1] only for firmware version 1.55 and above , * Equivalent to +/- 13A.

System Variable Descriptions

RecRate	Data recorder sample rate.
RecAddr	Data recorder sample address source.
RecSize	Data recorder sample data size.
AIN0	Axis 1 ID. (After TA0,GA0 command). +/-2047*
AIN1	Axis 1 IA. (After TA1,GA1 command). +/-2047*
AIN2	Axis 1 IB. (After TA2,GA2 command). +/-2047*
AIN3	Axis 1 IQ. (After TA3,GA3 command). +/-2047*
AIN4	Axis 2 ID. (After TA4,GA4 command). +/-2047*
AIN5	Axis 2 IA. (After TA5,GA5 command). +/-2047*
AIN6	Axis 2 IB. (After TA6,GA6 command). +/-2047*
AIN7	Axis 2 IQ. (After TA7,GA7 command). +/-2047*
AIN8	AN_IN0. (After TA8,GA8 command). -10V to +10V = -2047 to +2047
AIN9	AN_IN1. (After TA9,GA9 command). -10V to +10V = -2047 to +2047
AIN10	AN_IN2. (After TA10,GA10 command). 0V to +10V = 0 to +4095
AIN11	AN_IN3. (After TA11,GA11 command). 0V to +10V = 0 to +4095
AIN12	AN_IN4. (After TA12,GA12 command). 0V to +10V = 0 to +4095
AIN13	VMON. (After TA13,GA13 command). V+ Voltage monitor. 0 - 48V = 0 - 4095
AIN14	1OTW. (After TA14,GA14 command). Same as TTEMP.
AIN15	2OTW. (After TA15,GA15 command). Same as TTEMP.
VERSION	Version number for program use. (VE command).
LST_ERR	Last error code (TE command).
LTIMER	1 Hz LED timer.
ADSEMA	A/D Interrupt system semaphores flags.
RecNE	Number of data recorder array elements
RecE1A	Element 1 address
RecE1S	Element 1 size
RecE2A	Element 2 address
RecE2S	Element 2 size
RecE3A	Element 3 address
RecE3S	Element 3 size
RecE4A	Element 4 address
RecE4S	Element 4 size
STO_STAT	STO Status (Bit 0=STO_IN_1, 1=STO_IN_2, 2=STO_Active), 3=STO_Fault).
STO_A_CMP	STO Active comparator. Default is 10 (ms).
STO_F_CMP	STO Fault comparator. Default is 1600 (ms).
STO_A_TMR	STO Active timer.
STO_F_TMR	STO Fault timer.
SYSSTAT	System status word.
IPEND0	User interrupt pending level 0 -15 (EV,DV command).
IPEND1	User interrupt pending level 16 - 31 (EV,DV command).
LCNT	Servo update rate (SS command).
SCLOCK	Servo loop counter.
RCLOCK	1 mS Real time clock/counter.
IO_DELAY	User digital input delay (ID command).

Note: * Equivalent to +/- 13A.

SYSSTAT Variable Bit Definitions

Bit	Purpose
0	**Axis 0 Enabled.** This bit is set to "1" when axis 0 is enabled.
1-4	**Reserved.**
5	**Pause.** This bit is set to "1" when the space bar is used to pause a user program.
6	**SXOff.** This bit is set to "1" when the VLC-25 receives an XOFF code.
7	**HexMod.** This bit is set to "1" by the HM command and set to "0" by the DM command.
8	**Echo.** This bit is set to "1" by the EN command and set to "0" by the EF command.
9	**HandSh.** This bit is set to "1" by the HN command and set to "0" by the HF command.
10-13	**Reserved.**
14	**Fail.** This bit is set to "1" by the FN command and set to "0" by the FF command.
15	**BadInp.** This bit is set to "1" by the VI command to indicate an error in user input.

Axis Variable Locations

Variable Name	Type	Axis #1 Address	Axis #2 Address
Status	LONG	01C0H/0448	0290H/0656
PV	LONG	01C6H/0454	0296H/0662
MPV	LONG	01CAH/0458	029AH/0666
V	LONG	01CEH/0462	029EH/0670
Desp	LONG	01E0H/0480	02B0H/0688
Carp	LONG	01E6H/0486	02B6H/0694
Ack	LONG	01EAH/0490	02BAH/0698
Curp	LONG	01EEH/0494	02BEH/0702
HREG	LONG	01F8H/0504	02C8H/0712
IPPOS	LONG	01FCH/0508	02CCH/0716
QI	LONG	0200H/0512	02D0H/0720
PGAIN	WORD	0204H/0516	02D4H/0724
IGAIN	WORD	0206H/0518	02D6H/0726
DGAIN	WORD	0208H/0520	02D8H/0728
IL	WORD	020AH/0522	02DAH/0730
CGAIN	WORD	020CH/0524	02DCH/0732
FVGAIN	WORD	020EH/0526	02DEH/0734
BIAS	WORD	0210H/0528	02E0H/0736
THRO	WORD	0212H/0530	02E2H/0738
QCMD	WORD	0214H/0532	02E4H/0740
TLMTPL	WORD	0216H/0534	02E6H/0742
FAGAIN	WORD	0218H/0536	02E8H/0744
PERR	WORD	021AH/0538	02EAH/0746
OERR	WORD	021CH/0540	02ECH/0748
MAXERR	WORD	021EH/0542	02EEH/0750

I	WORD	0220H/0544	02F0H/0752
DERIV	WORD	0222H/0546	02F2H/0754
IMON	WORD	0224H/0548	02F4H/0756
INTRVL	BYTE	0226H/0550	02F6H/0758
SMPCNT	BYTE	0227H/0551	02F7H/0759
IINTRVL	BYTE	0228H/0552	02F8H/0760
ISMPCNT	BYTE	0229H/0553	02F9H/0761
AXIS	WORD	022AH/0554	02FAH/0762
ATYPE	BYTE	022CH/0556	02FCH/0764
PHASE	BYTE	022EH/0558	02FEH/0766
GMAxis	BYTE	022FH/0559	02FFH/0767
DBAND	WORD	0230H/0560	0300H/0768
DERR	WORD	0232H/0562	0302H/0770
GMOFF	LONG	0234H/0564	0304H/0772
RATIO	LONG	0238H/0568	0308H/0776
USER1	WORD	0240H/0576	0310H/0784
FCNT	WORD	0242H/0578	0312H/0786
FCMP	WORD	0244H/0580	0314H/0788
TLMTMI	WORD	0246H/0582	0316H/0790
LPG1[1]	WORD	0248H/0584	0318H/0792
LPG2[1]	WORD	024AH/0586	031AH/0794
CMPG[1]	WORD	024CH/0588	031CH/0796
RCurp	LONG	0250H/0592	0320H/0800
POff	LONG	0254H/0596	0324H/0804
CPEC	LONG	0258H/0600	0328H/0808
COMPH	WORD	025CH/0604	032CH/0812
IAOff	WORD	025EH/0606	032EH/0814
IBOff	WORD	0260H/0608	0330H/0816
i2t_NOM	WORD	0262H/0610	0332H/0818
i2t_TRIP	LONG	0264H/0612	0334H/0820
i2t_INT	LONG	0268H/0616	0338H/0824
i2tINTRVL	BYTE	026CH/0620	033CH/0828
i2tSMPCNT	BYTE	026DH/0621	033DH/0829
IA	WORD	026EH/0622	033EH/0830
IB	WORD	0270H/0624	0340H/0832
ID	WORD	0272H/0626	0342H/0834
IQ	WORD	0274H/0628	0344H/0836
TTRIP	WORD	0276H/0630	0346H/0838
TTEMP	WORD	0278H/0632	0348H/0840

Note: [1]Firmware version 1.55 and above

System Variable Locations

Variable Name	Type	Address
RecRate	WORD	01A6H/422
RecAddr	WORD	01A8H/424
RecSize	WORD	01AAH/426
AIN0	WORD	05F4H/1524
AIN1	WORD	05F6H/1526
AIN2	WORD	05F8H/1528
AIN3	WORD	05FAH/1530
AIN4	WORD	05FCH/1532
AIN5	WORD	05FEH/1534
AIN6	WORD	0600H/1536
AIN7	WORD	0602H/1538
AIN8	WORD	0604H/1540
AIN9	WORD	0606H/1542
AIN10	WORD	0608H/1544
AIN11	WORD	060AH/1546
AIN12	WORD	060CH/1548
AIN13	WORD	060EH/1550
AIN14	WORD	0610H/1552
AIN15	WORD	0612H/1554
VERSION	WORD	0616H/1558
LST_ERR	BYTE	0619H/1561
LTIMER	WORD	061AH/1562
ADSEMA	WORD	061CH/1564
RecNE	WORD	0640H/1600
RecE1A	WORD	0642H/1602
RecE1S	WORD	0644H/1604
RecE2A	WORD	0646H/1606
RecE2S	WORD	0648H/1608
RecE3A	WORD	064AH/1610
RecE3S	WORD	064CH/1612
RecE4A	WORD	064EH/1614
RecE4S	WORD	0650H/1616
STO_STAT	WORD	06F4H/1780
STO_A_CMP	WORD	06F6H/1782
STO_F_CMP	WORD	06F8H/1784
STO_A_TMR	WORD	06FAH/1786
STO_F_TMR	WORD	06FCH/1788
SYSSTAT	WORD	0712H/1810
IPEND0	WORD	0718H/1816
IPEND1	WORD	071AH/1818
LCNT*	BYTE	071EH/1822
SCLOCK	LONG	0722H/1826
RCLOCK	LONG	0726H/1830
IO_DELAY	BYTE	073EH/1854

Note: * only readable. Writing into this variable can only be done through the SS command

---

Command: AAn -- Add ‘n’ to Accumulator (Signed) –

Argument: -2147483647 <=n <= 2147483647

This command adds the value ‘n’ to the accumulator.

Command: AC -- Accumulator Complement –

This command causes a (bit-wise) 1’s complement of the accumulator.

Command: ADn -- Accumulator Divide by ‘n’ (Signed) –

Argument: -2147483647 <=n <= 2147483647

This command performs a 64 bit by 32 bit signed division with a 64 bit quotient and a 32 bit remainder. **The low order 32 bits of the numerator must be in the accumulator and the high order 32 bits must be in Register 1.** The divisor is specified by 'n'. The lower 32 bits of the quotient will be stored in the accumulator and the upper 32 bits will be stored in Register 1. The remainder will be in Register 2.

Related Commands: AM

Command: AEn -- Exclusive-Or Accumulator with ‘n’ –

Argument: -2147483647 <=n <= 2147483647

This command causes the result of an exclusive-OR of the accumulator with 'n' to reside in the accumulator.

Command: ALn -- Load Accumulator with ‘n’ (Signed) –

Argument: -2147483647 <=n <= 2147483647

This command causes the accumulator to be loaded with the value ‘n’

Command: AMn -- Multiply Accumulator by ‘n’ (Signed) –

Argument: -2147483647 <=n <= 2147483647

This command does a signed multiply of the 32 bit contents of the accumulator and 'n', leaving the lower 32 bits of the 64 bit product in the accumulator and the upper 32 bits in Register 1.

Related Commands: AD

Command: ANn -- And Accumulator with ‘n’ –

Argument: -2147483647 <=n <= 2147483647

This command causes the result of the accumulator AND-ed with ‘n’ to reside in the accumulator.

Command: AOn -- Or Accumulator with ‘n’ –

Argument: -2147483647 <=n <= 2147483647

This command causes the result of the accumulator OR-ed with ‘n’ to reside in the accumulator.

Command: ARn -- Copy Accumulator to Register ‘n’ –

Argument: 0 <=n <= 2047

This command causes the value in the accumulator to be copied to Register ‘n’.

Related command: RA

Command: ASn -- Subtract ‘n’ from Accumulator (Signed) –

Argument: -2147483647 <=n <= 2147483647

This command causes the value ‘n’ to be subtracted from the accumulator.

Command: RAn -- Copy Register ‘n’ to Accumulator –

Argument: 0 <=n <= 2047

This command causes the value in Register ‘n’ to be copied to the accumulator

Related command: AR

Command: RBn -- Read Byte (8 bits) at RAM location ‘n’ –

Argument: 0 <=n <= 2047

This command reads the 8 bit value at internal RAM location 'n' and copies it to the low 8-bits of the accumulator while clearing the upper 24 bits.

Related Commands: RW, RL, WB, WL, WW

Command: RLn -- Read Long (32 bits) at RAM location ‘n’ –

Argument: 0 <=n <= 2046

This command reads the 32 bit value at internal RAM location 'n' and copies it to the accumulator. The value specified by 'n' must be evenly divisible by 2.

Related Commands: RB, RW, WB, WL, WW

Command: RWn -- Read Word (16 bits) at RAM location ‘n’ –

Argument: 0 <=n <= 2046

This command reads the 16 bit value at internal RAM location 'n' and copies it to the low 16-bits of the accumulator while clearing the upper 16 bits. The value specified by 'n' must be evenly divisible by 2.

Related Commands: RB, RL, WB, WL, WW

Command: SLn -- Shift Accumulator Left ‘n’ bits –

Argument: 0 <= n <= 31

This command causes the accumulator to be shifted left ‘n’ bits while filling the low order bits with zero.

Related command: SR

Command: SRn -- Shift Accumulator RIght ‘n’ bits –

Argument: 0 <= n <= 31

This command causes the accumulator to be shifted right ‘n’ bits while filling the high order bits with zero.

Related command: SL

Command: TRn -- Tell Contents of Register ‘n’ –

Argument: 0 <= n <= 2047

This command reports the value contained by Register ‘n’.

Command: WBn -- Write Byte (8 bits) to RAM location ‘n’ –

Argument: 0 <= n <= 2047

This command copies the low 8-bits of the accumulator to the internal RAM location ‘n’.

Related Commands: RB, RL, RW, WL, WW

Command: WLn -- Write Long (32 bits) to RAM location ‘n’ –

Argument: 0 <= n <= 2046

This command copies all 32 bits of the accumulator to the internal RAM location 'n'. The value of 'n' must be evenly divisible by two.

Related Commands: RB, RL, RW, WB, WW

Command: WWn -- Write Word (16 bits) to RAM location ‘n’ –

Argument: 0 <= n <= 2046

This command copies the low 16-bits of the accumulator to the internal RAM location 'n'. The value of 'n' must be evenly divisible by two.

Related Commands: RB, RL, RW, WB, WL

3.5 Sequence commands

The VLC-25 includes commands that provide for conditional sequence command execution based on the register data, I/O state and etc. These Sequence Commands are illustrated by the following general forms:

- If the condition is true, command execution will continue normally. If the condition is false, the next two commands in the command line or the macro will be skipped.
- If the condition is true, command execution will continue normally. If the condition is false, the rest of the command line or the macro will be skipped.
- If the condition is true, command execution will continue normally. If the condition is false, command execution will be suspended until the condition becomes true.

Command: DFn -- Do If I/O Channel is "Off" –

Argument: 0 <= n <= 1

This command will cause command execution to continue if I/O channel 'n' is "Off"; otherwise, the rest of the command line or the macro will be skipped.

Related Commands: CF, CN, DN

Command: DNn -- Do If I/O Channel is "On" –

Argument: 0 <= n <= 1

This command will cause command execution to continue if I/O channel 'n' is "On"; otherwise, the rest of the command line or the macro will be skipped.

Related Commands: CF, CN, DF

Command: EP -- End Program –

This command will cause program execution to cease and return the ">" prompt.

Related commands: RC

Command: IBn -- If Accumulator is Below 'n' (Signed) –

Argument: -2147483647 <=n <= 2147483647

If the contents of the accumulator is less than (signed comparison) the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: ICn -- If Bit 'n' of Accumulator is Clear (0) –

Argument: 0 <= n <= 31

If bit 'n' of the accumulator is clear (equals 0), command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: IEn -- If Accumulator Equal to 'n' –

Argument: -2147483647 <=n <= 2147483647

If the accumulator is equal to the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: IU

Command: IFn -- If I/O Channel is "Off" –

Argument: 0 <=n <= 63

If the I/O channel specified by 'n' is in the "off" state, command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: CF, CN, IN

Command: IGn -- If Accumulator is Greater than 'n' (Signed) –

Argument: -2147483647 <=n <= 2147483647

If the contents of the accumulator is greater than (signed comparison) the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Command: INn -- If I/O Channel is "On" –

Argument: 0 <=n <= 63

If the I/O channel specified by 'n' is in the "on" state, command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: CF, CN, IF

Command: aIPn -- Interrupt on Absolute Position 'n' –

Argument: -2147483647 <=n <= 2147483647

This command is used to determine when servo axis 'a' has achieved the specified real position 'n' referenced by the home position (or zero). When that position has been reached, the "breakpoint reached" flag in the status register will be set. This command can be issued before or after the servo has been commanded to move.

Related Commands: IR, TB

Command: aIRn -- Interrupt on Relative Position 'n' –

Argument: -2147483647 <=n <= 2147483647

This command is used to determine when servo axis 'a' has achieved the specified real position 'n' referenced by the current position when this command is executed. When that position has been reached, the "breakpoint reached" flag in the status register will be set. This command can be issued before or after the servo has been commanded to move.

Related Commands: IP, TB

Command: ISn -- If Bit 'n' of Accumulator is Set (1) –

Argument: 0 <=n <= 31

If bit 'n' of the accumulator is set (equals 1), command execution will continue; otherwise, the next two commands in the command line the or the macro will be skipped.

Related Commands: IC

Command: IUn -- If Accumulator Unequal to 'n' –

Argument: -2147483647 <=n <= 2147483647

If the accumulator is unequal to the value 'n', command execution will continue; otherwise, the next two commands in the command line or the macro will be skipped.

Related Commands: IE

Command: RPn -- Repeat –

Argument: 0 <=n <= 65535

This command causes the command line to repeat ’n’ more times. If ’n’ is not specified or is "0", the command line is repeated indefinitely.

Command: WAn -- Wait ‘n’ milliseconds –

Argument: 0 <=n <= 65535

This command causes a wait period of ’n’ milliseconds before going on to the next command.

Command: aWEn -- Wait for Edge of Coarse Home Input –

Argument: 0 or 1

This command waits until the Coarse Home Input Active bit in the status word of servo axis ‘a’ is at the logic state specified by ’n’ before continuing command execution. If ’n’ is not specified or is 0, it will wait for the Home input to go inactive. If ’n’ is 1, it will wait for the Home input to go active.

Related Command: FE

Command: WFn -- Wait for I/O Channel “Off” –

Argument: 0 <= n <= 63

Default: “Off”

This command waits until I/O channel ’n’ is in the "off" state before continuing command execution.

Related Commands: CF, CN, WN

Command: aWI -- Wait for Index –

This command will wait for the occurrence of an Index input signal on servo axis ‘a’ before continuing to the next command. If a Find Index (FI) command has not been issued prior to this command, no waiting will occur.

Related Commands: CI, FI

Command: WNn -- Wait for I/O Channel “On” –

Argument: 0 <= n <= 63

Default: “Off”

This command waits until I/O channel ’n’ is in the "on" state before continuing command execution.

Related Commands: CF, CN, WF

Command: aWPn -- Wait for Absolute Position ‘n’ –

Argument: -2147483647 <=n <= 2147483647

This command is used to determine when servo axis ‘a’ has achieved the specified real position ’n’ referenced by the home position (or zero). Until that position has been reached, command execution will be suspended. This command can be issued before or after the servo has been commanded to move; however, if the servo is not moving, then command execution may be suspended indefinitely.

Related Commands: WR

Command: aWRn -- Wait for Relative Position ‘n’ –

Argument: -2147483647 <=n <= 2147483647

This command is used to determine when servo axis ‘a’ has achieved the specified real position ’n’ referenced by the current position when this command is executed. Until that position has been reached, command execution will be suspended. This command can be issued before or after the servo has been commanded to move; however, if the servo is not moving, then command execution might be suspended indefinitely.

Related Commands: WP

Command: aWSn -- Wait for Stop –

Argument: 0 <= n <= 65535

This command will wait until servo axis ‘a’ trajectory has stopped moving for ’n’ milliseconds before continuing to the next command.

Related Commands: ST

3.6 Learned position storage commands

The VLC-25 uses part of its flash memory to create a 1792 item table for storing positions.

The LPS table is accessed by three commands: Learn Position (LP), Learn Target (LT) and Move to Position (MP). The purpose for the LPS table is to allow the user to store pre-determined positions for later use (such as in contouring) as the flash memory will retain data even when powered down.

The LPS table overlaps registers 256 - 2048 in the general purpose register space so that LPS entry 0 is the same as register 256. This allows for the calculation of predefined positions that can be accessed via the LPS commands.

Command: aLPn -- Learn Current Position –

Argument: 0 <= n <= 1791

This command causes the current real position of servo axis ‘a’ (that position reported by the Tell Position (TP) command) to be stored in location ’n’ of the Learned Position Storage table. This command is useful for "teaching" positions to the VLC-25.

Related Commands: LT, MP

Command: aLTn -- Learn Target Position –

Argument: 0 <= n <= 1791

This command causes the current ideal target position of servo axis ‘a’ (that position reported by the Tell Target (TT) command) to be stored in the location ’n’ of the Learned Position Storage table. This command is useful for setting up a table of target positions via a downloading procedure.

Related Commands: LP, MP

Command: aMPn -- Move to Index Table Position ‘n’ –

Argument: 0 <= n <= 1791

This command causes the position contained by the Learned Storage Position table location ’n’ to become the new target position. This command will not initiate a motion; therefore, a Go (GO) command may be required.

Related Commands: LP, LT

3.7 Macro commands

Command instructions can be entered directly and executed immediately, but the VLC-25 also has the capability of using commands to form other commands called "macros". These macros can be stored in the flash memory and can be executed automatically. Macros are created by stringing together one or more commands (with arguments), separated by commas, and indicating where they are to be stored. It should be noted that macros can use indirect as well as direct arguments. The VLC-25 allows for the creation of 256 macros consisting of a total of over 8900 command instructions. Macro calls via the Macro Call (MC) command or the interrupt system, may be nested up to 25 calls deep.

An example of a macro might be...

```
>MD5,SV1000000,SA10000,MA25000,GO,WS100
```

Once this command line is entered, it will become the definition for macro 5. When macro

5 is used via the Macro Call (MC), Macro Sequence (MS) or Macro Jump (MJ) commands, the commands contained within the macro will be executed automatically.

There are a few necessary restrictions when creating macros. When using the Macro Define (MD) command, it must be placed first in the command line and it may be used only once. This also implies that it cannot be used as part of a macro. The Reset Macro (RM) command is similar in that it too cannot be used as part of a macro.

Another feature of the VLC-25 is the ability to automatically execute macro "0" on power-up or after a Reset (RT) command. If macro "0" is not defined, the user will receive the ">" prompt, and the VLC-25 will wait for manual input. If macro “0” is defined, the VLC-25 will automatically generate a “MS0” command thereby executing macro “0”.

Command: DVn -- Disable Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command disables interrupt vector 'n'. This prevents any possibility of an interrupt being caused by that source.

Related Commands: EV, LV

Command: EVn -- Enable Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command enables interrupt vector 'n'. This allows the possibility of an interrupt being caused by that source.

Related Commands: DV, LV

Command: JPn -- Jump to Command, Absolute --

Argument: 0 <= n <= 31

This command causes execution of a macro to jump to the absolute command specified by ’n’. Commands are numbered sequentially starting from 0. If ’n’ is specified whereas the command would attempt to jump past the end of the macro, command execution will resume as if the macro had ended.

Related Commands: JR

Command: JRn-- Jump to Command, Relative --

Argument: -31 <= n <= 31

This command causes execution of a macro to jump to the command specified by 'n' relative to the current command location. If 'n' is specified as zero, then this command will jump to itself. If 'n' is specified such that the command would attempt to jump past the end of the macro, command execution will resume as if the macro had ended. If 'n' is specified such that the command would attempt to jump to a point before the beginning of the macro, an error will be reported.

Related Commands: JP

Command: LVn -- Load Interrupt Vector 'n' --

Argument: 0 <= n <= 31

This command loads interrupt vector table entry 'n' with the contents of the low 8-bits of the accumulator. The following program fragment will cause execution of macro "10" upon the activation of digital input 0 (while a macro program is running).

```
MD1,AL10 ; Load accumulator with number for macro "10".
MD2,LV0 ; Load that number to interrupt vector 0.
MD3,EV0 ; Enable interrupt vector 0.
MD10,MG"Input 0 was just activated.",RC
```

When input 0 is activated, the line "Input 0 was just activated." will be displayed.

Related Commands: DV, EV

Command: MCn -- Macro Call --

Argument: 0 <= n <= 255

This command allows a previously defined macro specified by 'n' to be called like a subroutine. When this command is used, the current macro being executed is pushed to the macro stack and execution of macro 'n' begins. If macro 'n' has not been defined, then an error will be reported. After execution of the defined macro, command execution will continue immediately after the Macro Call (MC) command. The Macro Call (MC) command can be used any place in a macro. Macro calls may be nested up to 25 times; however, it is NOT advisable for a macro to call itself.

Related Commands: RC, UM

Command: MDn -- Macro Definition --

Argument: 0 <= n <= 255

This command is used to define a new macro. Any duplication of numbers will simply result in the loss of the previously defined macro using that number and the loss of the memory that it used. The Macro Define (MD) command must be the first command in the command line or an error will be reported.

Related Commands: RM, TM

Command: MJn -- Macro Jump --

Argument: 0 <= n <= 255

This command may be used to "Jump" to a previously defined macro command. Once the VLC-25 begins executing the new macro, it has no record of how it got there. This means that any commands that appear after the Macro Jump (MJ) command will not be executed. If there is no macro defined by the number 'n', an error will be reported. Once the end of the macro is encountered, macro execution stops (See the Macro Sequence (MS) command). The Macro Jump command can be used any place in a command string or macro command. It is also acceptable for a macro to jump to itself.

Related Commands: MC, MS

Command: MSn -- Begin Macro Sequence --

Argument: 0 <= n <= 255

This command will cause macros to be executed sequentially beginning with macro 'n' until an undefined macro or an End Program (EP) command is encountered. This command can be used anywhere in a command string or macro command. If the Macro Jump (MJ) or Macro Call (MC) commands are encountered, macro execution will still continue to execute sequentially.

Related Commands: MC, MJ

Command: NBn -- Interrupt on Bit –

Argument: 0 <= n <= 32767

This command allows interrupt level/vector 15 to share a function that monitors a single bit level in the "internal variables" memory space. When that bit is observed in the desired state, it will cause a level 15 interrupt.

The command's parameter is constructed such that it specifies the bit to be monitored as well as its desired state. To build the parameter, the byte address must be multiplied by 16. To that result you must add the bit address (within the byte) multiplied by 2. Finally, to that result the desired bit state can be added, either 0 or 1. If the parameter is set to '"0", then the function is disabled.
The following example will cause an interrupt to macro 20 if the "Servo Error" status bit becomes "1":

```
; 32-bit STATUS variable = 448 (RL448)
; STATUS variable is made up of byte addresses:
;   Bits  0 -  7 = 448
;   Bits  7 - 15 = 449
;   Bits 16 - 23 = 450
;   Bits 24 - 31 = 451
; "Servo Error" bit = STATUS bit 1 or byte 448, bit 1
; Parameter 'n' = (byte address * 16) + (bit address * 2) + bit state
; n = 7171 = (448 * 16) + (1 * 2) + 1
;
>MD20,MG"Servo Error Detected",EP
>MD21,AL20,LV15,EV15,NB7171
>MD22,MG"Waiting for Servo Error”
>
>MS21
 ( ... waiting for Servo Error ... )
Servo Error Detected
>
```

Related Commands: DV, EV, LV

Command: RC -- Return from Macro Call --

When executed, this command will cause immediate return to the calling macro (assuming there was one).

Related Commands: MC, interrupts

Command: RMn -- Reset Macro(s) –

Argument: 0 <= n <= 255

This command is used to delete one or more macros. If an argument is used, the macro specified by 'n' will be deleted and the memory it used will be lost. If no argument is used, all macros will be deleted, and all macro memory will be recovered. A Reset Macro command (RM) should be used before entering or downloading a new set of macro commands.

When multiple macros are to be erased through RM, in order to make sure that the macros are all erased, allow a time period of approximately 500 milliseconds before executing the next command. One way to do this is to make use of semicolons illustrated by the following example:

```
MF
RM ; or ZF123
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
MD0,NO,NO,NO
MD1, ..... etc......
```

Note that PS command is automatically executed after the execution of RM command.

Related Commands: MD

Command: TMn -- Tell Macros --

Argument: -2 <= n <= 255

The Tell Macro (TM) command will display the commands which make up any macros that have been defined. If 'n' >= 0 and 'n' <= 255, then the macro specified by 'n' be displayed. If 'n' = - 1, then all the macros will be displayed preceded by the individual macro number. If 'n' =-2, then all macros will be displayed and will be preceded by the letters "MD" and the individual macro number. This is useful when downloading programs from the VLC-25 to a computer in that you need not re-enter the Macro Define (MD) command and number at the beginning of each macro.

Related Commands: MD

Command: UMn -- Un-push Macro(s) --

Argument: 0 or 1

This command is used for controlling the macro subroutine call and return stack. If this command is used with no argument or an argument of "0" then one macro will be removed from the macro stack. If there is no macro to be removed then a MACRO_STACK_UNDERFLOW error will be reported. If the argument is "1" then the macro stack will be completely reset. The UM command is used in the event that a macro is paused due to a program interrupt and it is not desirable to return to that macro, or under some circumstances, a program may need to completely reset itself.

Related Commands: MC, interrupts

3.8 Input/Output commands

The user is able to manipulate the VLC's I/O channels via the use of several commands. These include setting or clearing outputs, reading inputs and altering the logic type of both.

Command: BIn – Bulk Input from I/O Port ‘n’ --

Argument: 0 <= n <= 7

This command reads the value at the 8-bit digital input port and copies it to the low 8-bits of the accumulator with the lower channel being the lowest bit in the accumulator. The unused bits of the accumulator will be set to 0. The state of the inputs will be determined by the Channel High (CH) and Channel Low (CL) commands and the inputs will have been debounced if the Input Debounce (ID) command is in effect.

Related Commands: BO

Command: BOn -- Bulk Output to I/O Port 'n' --

Argument: 0 <= n <= 7

This command copies the low 8-bits of the accumulator to the digital output port. The state of the outputs will be determined by the Channel High (CH) and Channel Low (CL) commands. Unused bits in the accumulator are ignored.

Related Commands: BI

Command: CFn -- Turn I/O Channel ’n’ "Off" --

Argument: 0 <= n <= 63

Default: "OFF"

This command will cause I/O channel ’n’ to assume the "OFF" state. The actual output state will depend on whether the channel is set active HIGH (CH command) or active LOW (CL command).

Related Commands: CH, CL, CN

Command: CHn -- Make I/O Channel ’n’ Active High --

Argument: 0 <= n <= 63

Default: Active HIGH

This command causes I/O channel ’n’ to assume an active HIGH mode.

Related Commands: CL

Command: CLn -- Make I/O Channel ’n’ Active Low --

Argument: 0 <= n <= 63

Default: Active HIGH

This command causes I/O channel ’n’ to assume an active LOW mode.

Related Commands: CH

Command: CNn -- Turn I/O Channel ’n’ "On" --

Argument: 0 <= n <= 63

Default: "OFF"

This command will cause I/O channel ’n’ to assume the "ON" state. The actual output state will depend on whether the channel is set active HIGH or active LOW.

Related Commands: CF, CH, CL

Command: IDn -- Input Debounce ’n’ milliseconds --

Argument: 0 <= n <= 7

Default: 0

This command determines the length of debounce time, if any, that is applied to the digital inputs. The digital inputs are sampled once every millisecond. What this means is that an input must remain in a given state for ’n’ samples before it is considered valid. If the argument ’n’ is "0", then no input debouncing in performed. This debouncing applies to the following commands: BI, DF, DN, IF, IN, WF, WN.

3.9 Serial communications and miscellaneous commands

These commands control operation of the serial communication's interface and cover the balance of the VLC-25 functions not fitting in the other categories.

Command: BK -- Break –

This command will cause the rest of the command line or macro to be skipped. This command is used along with the Sequence Commands for conditional command execution.

Command: BRn – Set Baud Rate –

Argument: 2400, 4800, 9600, 14400, 19200, 38400, 57600, 62500, 78125, 115200, 125000, 128000, 156250, 230400, 460800

Default: 9600

This command allows the user to change the baud rate at which the serial communication's interface operates. Once this command has been issued with a valid argument, the communication's interface will then immediately operate at the specified baud rate. This baud rate will remain in place, even after power cycling the unit, until it is changed again. **Please note that the default baud rate is 9600.**

Note: To help protect against communication lock-out, baud rates above 19200 are not non-volatile and must be included in a macro for automatic use. They also include a 2-second delay before execution - this gives the user an opportunity to hit 'ESC' and stop macro execution before the new baud rate is implemented. This feature is included because not all terminal programs will support all the baud rate available to the VLC-25 user.

Command: CDn – Capture Data –

Argument: 0 <= n <= 16383

This command allows for the capture and recording of data from an internal variable with the user being able to later download and plot this data for analysis.

The argument determines the number of samples to record. If an argument is used that is greater than the allocated storage space, then the argument will be truncated to fit that space. All data sizes are extended to 32 bits before they are stored.

Before using the CD command, a few things must be taken care of. The first of these is to define an area to store the data. The Capture Storage (CS) command is used for this in that it allocates a storage area in the macro memory to where the data can be recorded. The next thing to do is to specify the location and size of the data to be recorded. The internal 16-bit variable "RecAddr" (see Register commands) is used to specify the location (or address) of the variable to be recorded. The internal 16-

bit variable “RecSize” is used to specify the data size. If bit 0 of “RecSize” is equal to “1” (“AL1”) then 8 bit data samples are recorded. If bit 1 of “RecSize” is equal to “1” (“AL2”) then 16 bit data samples are recorded. If bits 0 and 1 of “RecSize” are both “0” (“AL0”) then 32 data samples are recorded. The last thing that needs to be specified is the sample rate at which the data is captured. This is set by writing the appropriate value to the 16-bit variable “RecRate”. The sample rate will be whatever the servo sample rate is (see SS command) divided by “RecRate”.

After using the CD command, the Dump Data (DD) command can be used to display the data that was captured.

Example:

To make 2000 records of the actual position of axis 2 you would issue the following commands:

```
CS1000 ; Reserve space for 1000 samples.
;
;
;
AL4,WW422 ; Servo rate = 1mS so Data rate = 4mS.
AL638,WW424 ; Set address of axis 2 Curp.
AL0,WW426 ; Set for 32-bit variable size.
```

The CS command should be used only once. After that, the CD and DD commands can be used repeatedly.

Simultaneous data capture of up to 4 variables is also possible. By setting “RecNE (Record Number Element)” to a non-zero value, the controller will capture multiple, synchronized variables each sample loop, as many as directed by “RecNE” variable.

The below example will record the SCLOCK variable, every servo loop cycle, in 32-bit samples (standard operation):

```
>CS16383
>
>md10,al0,ww1600 ; RecNE=0 (this line might not be necessary as RAM
is set
; to 0's on power-up)
>md11,al1,ww422 ; RecRate=1
>md12,al1826,ww424 ; RecAddr=1826 (SCLOCK)
>md13,al0,ww426 ; Rec=Size=0 (32-bit)
>CD30
>DD30.....
```

The next example will do the same thing, using “RecNE”:

```
md20,al1,ww422 ; RecRate=1
md21,al1,ww1600 ; RecNE=1
md22,al1826,ww1602 ; RecE1A=1826 (SCLOCK)
md23,al0,ww1604 ; RecE1S=0 (32-bit)
>CD30
>DD30.....
```

The next example will record SCLOCK, RCLOCK, SCLOCK and RCLOCK each sample period:

```
md30,al1,ww422 ; RecRate=1
```

```
md31,al4,ww1600 ; RecNE=4
md32,al1826,ww1602 ; RecE1A=1826 (SCLOCK)
md33,al0,ww1604 ; RecE1S=0 (32-bit)
md34,al1830,ww1606 ; RecE2A=1830 (RCLOCK)
md35,al0,ww1608 ; RecE2S=0 (32-bit)
md36,al1826,ww1610 ; RecE3A=1826 (SCLOCK)
md37,al0,ww1612 ; RecE3S=0 (32-bit)
md38,al1830,ww1614 ; RecE4A=1830 (SCLOCK)
md39,al0,ww1616 ; RecE4S=0 (32-bit)
>CD30
>DD30.....
```

Related Commands: CS, DD

Command: CSn – Capture Storage –

Argument: 0 <= n <= 16383

This command is used to allocate storage space for the data recorder commands. The argument determines the maximum number of samples that can be recorded. **The number of samples that can be recorded depends on the amount of macro memory available at the time that the Capture Storage (CS) command is issued.** If an argument is used that is greater than the macro memory available, then an error will be reported. Because information pertaining to this command is stored in non-volatile memory, once this command is issued, the storage space will remain, even after power-cycling. To recover the macro memory used by this command, use the Reset Macros (RM) command.

Related Commands: CD, DD

Command: DDn -- Dump Data --

Argument: 0 <= n <= 16383

This command is used to dump data that has been previously recorded by the Capture Data (CD) command to the display. The argument determines the number of recorded samples to display. If an argument is used that is greater than the allocated storage space, then the argument will be truncated to fit that space.

Related Commands: CD, CS

Command: DM -- Decimal Mode --

Default: Decimal Mode

This command causes all numerical input and output to be interpreted as decimal or base 10. Numbers will be output with a leading "-" if they are less than zero.

Related Commands: HM

Command: EF -- Echo Off --

Default: Echo On

This command suppresses the echoing of characters received by the serial communication's interface. It is normally used in the half-duplex mode of operation in serial communications.

Related Commands: EN

Command: EN -- Echo On --

Default: Echo On

This command causes characters received from the serial communication's interface to be echoed as they are received. It is normally used in the full-duplex mode of operation in serial communications.

Related Commands: EF

Command: GAn -- Get A/D Channel

Argument: 0 <= n <= 15

This command will cause a conversion on A/D channel 'n' and will store the result in the bottom 12 bits of the accumulator. Unused bits will be set to 0. See the description of command TA for the list of accessible analog signals.

Example – displaying the analog input voltage:

```
GA3,AM10000,AD4095,MG"AN_IN1 = ":0,MG"mV"
```

Related Commands: TA

Command: HM -- Hexadecimal Mode --

Default: Decimal Mode

This command causes all numerical input and output to be interpreted as hexadecimal or base 16. Numbers will be output as 2, 4 or 8 digits with leading 0's if they are positive and leading F's if they are negative.

Related Commands: DM

Command: MG[[""][:n][:N]] -- Display Message --

Argument: 0 <= n <= 511

This command allows for the display of an optional text string and / or an optional register variable with the additional option of inhibiting the carriage return / linefeed (CRLF) at the end.

The text string must be enclosed by quotes """ and may be up to 127 characters long. Additional parameters must be separated by a colon ":" .

The following are all the valid examples. Note that the "N" option inhibits a CRLF. Parameters must be given in specific order.

```
>MG                              ; Display CRLF only.
>MG"THE BLACK BEAR IS "          ; Display "THE BLACK BEAR IS " followed by
                                 ; CRLF.
>MG0                             ; Display contents of register 0 followed
                                 ; by CRLF.
>MGN                             ; This command displays nothing followed
                                 ; by nothing.
>MG"THE BLACK BEAR IS ":0        ; Display "THE BLACK BEAR IS " followed by
                                 ; the contents of register 0 followed by
                                 ; CRLF.
>MG"THE BLACK BEAR IS ":N        ; Display "THE BLACK BEAR IS " followed by
                                 ; nothing.
>MG0:N                           ; Display the contents of register 0
                                 ; followed by nothing.
>MG"THE BLACK BEAR IS ":0:N      ; Display "THE BLACK BEAR IS " followed by
                                 ; the contents of register 0 followed by
                                 ; nothing.
```

Command: NO -- No Operation --

This command does nothing. It can be used to cause short delays in command line executions or to fill out commands (see Sequence Commands).

Command: PS -- Parameter Save --

This command saves macros and register values into the Non-Volatile Memory of the VLC-25.

Related command: PL

Command: PL -- Parameter Load --

This command loads the previously saved macros and register values into the Non-Volatile Memory of the VLC-25. PL is executed Automatically at VLC-25 power-up.

Related command: PS

Command: RT-- Reset --

This command performs a complete restart of the VLC-25, including restoration of all default conditions, such as acceleration and velocity, and leaves all servo axis' in the "off" state.

Command: VI[[""][:n][:N]] -- Variable Input –

Argument: 0 <= n <= 255

This command allows for the display of an optional text string, an optional carriage return / linefeed (CRLF) and the entry of integer numeric operator input to a register variable.

The text string must be enclosed by quotes """ and may be up to 127 characters long. Additional parameters must be separated by a colon ":". Entered numbers will be interpreted (as decimal or hexadecimal) as determined by the current mode set by the DM or HM commands. If the value entered by the operator is in error (indeterminate), then the Bad Input bit (bit 15 of the variable SYSSTAT) is set, otherwise it is cleared. If no operator input is entered (only a carriage return is received), then no register will be altered. If no argument is given and operator input is received, then register 0 will be the default recipient of the input value.

The followingare all the valid examples. Note thatthe "N" option causes a CRLF.

Parameters must be given in specific order.

```
>VI                             ; Wait for operator input (if any) to register 0.
>VI"ENTER NUMBER: "             ; Display "ENTER NUMBER: " and wait for
                              ; operator input (if any) to register 0.
>VI5                            ; Wait for operator input (if any) to
                              ; register 5.
>VIN                            ; Display a CRLF and wait for operator input
                              ; (if any) to register 0.
>VI"ENTER NUMBER: ":5           ; Display "ENTER NUMBER: " and wait for
                              ; operator input (if any) to register 5.
>VI"ENTER NUMBER: ":N           ; Display "ENTER NUMBER: " followed by a
                              ; CRLF and wait for operator input (if any)
                              ; to register 0.
>VI5:N                          ; Display a CRLF and wait for operator input
                              ; (if any) to register 5.
>VI"ENTER NUMBER: ":5:N     ; Display "ENTER NUMBER: " followed by a CRLF
                              ; and wait for operator input (if any) to
                              ; register 5.
```

Command: ZDn -- Read Capture Data Buffer into Register 0 --

Argument: 0 <= n <= 16383

This command will copy the 32-bit data from capture data buffer index 'n' into register 0. If the value of 'n' is larger than the size of the buffer it will be truncated to that maximum size. This command allows the user's program to analyze the buffer information without having to download it to an external computing device.

Related Commands: CD, CS, DD

Command: ZF -- Format Flash Memory --

Argument: n = 123

This command re-formats and re-initializes the VLC's flash memory. This means that any macros will be deleted and their space recovered, register contents will be set to 0 and the baud rate (among other things) will be set to their default values. Evidence of the baud rate change will not be immediate but will occur when the unit is either power cycled or the Reset (RT) command is issued. This command is intended for use by the manufacturer to place the unit in a known state after testing but is sometimes useful as a "when all else fails" means under certain circumstances. This command must include an argument "key code" of "123" in order to function. This is to reduce the possibility of accidental execution.

Note that PS command is automatically executed after the execution of ZF command.

Command: ZZ-- Dump Memory --

Argument: 0 <= n <= $3FFFF

This command does a display dump of the memory starting at address 'n'. This command is intended for use by the manufacturer for diagnosing purposes.

3.10 Bit commands

A running macro program could check if it should ask the user to input new data without stopping the program (as in the case of VI command) by means of the “Enquiry” function. This can be done by typing “Ctrl-E”, which will set the user semaphore bit 0. There are four commands that are used to manipulate a bit variable area consisting of 64 “user semaphore” bits. The new commands are Set Bit (SB), Clear Bit (CB), If Bit Set (BS) and If Bit Clear (BC). These all have a parameter of 0 to 63 to indicate on which bit to operate. The first 8 of these semaphore bits are reserved for system use with bit 0 being the “Enquiry” bit that signals that a Ctrl-E key has been received and bit 1 being the “IOError” bit that signals that the last I/O module communication sequence has failed. The remaining 56 bits are at the user’s discretion.

The following example illustrates how the “Enquiry” function is used:

```
MD1,CB0
MD2,BS0,MJ3,NO,MG"Waiting for Ctrl-E...",WA500,MJ2
MD3,MG"!!! Received Ctrl-E !!!",MJ1
```

Command: BCn -- If Bit Clear --

Argument: 0 <= n <= 63

If user semaphore bit 'n' is clear (equals 0), command execution will continue; otherwise, the next two commands in the command line, or the macro will be skipped.

Related Commands: BS, CB, SB

Command: BSn -- If Bit Set --

Argument: 0 <= n <= 63

If user semaphore bit 'n' is set (equals 1), command execution will continue; otherwise, the next two commands in the command line, or the macro will be skipped.

Related Commands: BC, CB, SB

Command: CBn -- Clear Bit --

Argument: 0 <= n <= 63

Clear user semaphore bit 'n' to 0.

Related Commands: BC, BS, SB

Command: SBn -- Set Bit --

Argument: 0 <= n <= 63

Set user semaphore bit 'n' to 1.

Related Commands: BC, BS, CB

---

4 Software Configurations for Driver Overtemperature Protection, I²T, and STO

4.1 Driver Overtemperature Protection

The VLC-25 uses thermistors planted on the PCB near the power stage to monitor driver temperature. The voltages generated by these thermistors are measured and converted to a digital value by the ADC system and stored in Ain14 and Ain15 depending on the axis. If the value exceeds TTRIP (see the axis variable Table in section 3.4.1) then the servo output will be disabled and an error will be signalled in the Status word. Below is a table showing the relation between temperature and ADC value from the thermistor.

Temp (°C)	ADC Value
25	2829
30	3042
35	3225
40	3381
45	3510
50	3617
55	3704
60	3776
65	3834
70	3880
75	3918
80	3949
85	3973
90	3994
95	4010
100	4024
105	4035
110	4044
115	4052
120	4059
125	4064

4.2 I^2T

The I^2T protection mechanism provides a means to configure the actuator overloading characteristic in terms of peak current and time period. This can be used to protect the motor from overheating. There are three internal variables (see the axis variable Table in section 3.4.1) related to I^2T:

- i2t_NOM: the nominal/continuous current that can be sustained indefinitely (in ADC units)

- i2t _TRIP: a parameter to set the overloading characteristics, where

$$i2t\ \_TRIP = ((IOL)^2 - (i2t\ \_NOM)^2) \times \Delta t_{OL}$$

  where IOL is the expected overload current (in ADC units) and Δt_{OL} is the overload time period (in milliseconds)

- i2tSMPCNT: sample counter

- i2tINTRVL: sample interval (value ranges between 0 and 255, with 0 being the default)

- i2t _INT: the integral of power that exceeds the value set by i2t _NOM, calculated as follow at every millisecond

  if(--i2tSMPCNT<0)
  {
  i2tSMPCNT = i2tINTRVL;
  i2t_INT= i2t _INT + $(IMON)^2$ – $(i2t\ \_NOM)^2$, limit 0 to 2^31-1
  if(i2t_INT > i2t_TRIP), Fault = 1;
  }

  where IMON is the measured motor current (in ADC units). When i2t _INT> i2t _TRIP, a fault is generated and the drive is shutdown.

Example 1:

Given that the VLC-25-13 has an output current rating of 10 A Cont./13 A Peak, where ADC counts 0 to +/-2047 represents 0 to +/-13 A, the actuator continuous current is 3 A and an overload of 9 A for 1 seconds is allowed, then

i2t _NOM = (3 A/13 A) × 2047 = 472

IOL = (9 A/13 A) × 2047 = 1417

i2t _TRIP = $(1417^2 - 472^2) \times 1000$ = 1,785,105,000

Example 2:

Same conditions as in example 1, except that the overloading period is 3 seconds, then

i2t _NOM = (3 A/13 A) × 2047 = 472

IOL = (9 A/13 A) × 2047 = 1417

i2t _TRIP = (1417^2-472^2) × 3000 = 5,355,315,000

The variables i2t_TRIP and i2t_INT have each a maximum value of 2^{31}-1. Meanwhile, the above calculation results in i2t_TRIP value that exceeds the maximum limit. In this case, i2t_TRIP value has to be calculated at a slower time interval, in other words, i2tINTRVL has to be set to a different value. Assuming that i2t_INT is to be calculated every 10 milliseconds (instead of the default of 1 millisecond), i2tINTRVL should be set to 9 and therefore i2t _TRIP can be calculated as follow

i2t _TRIP = (1417^2-472^2) × 300 = 535,531,500

4.3 STO

The two STO inputs (STO1 and STO2, see section 2.1.2) have to be supplied with the specified DC voltage in order to enable the VLCI's driver power stage to operate the actuator. If the external means of supplying the DC voltage is not considered, the VLCI's on-board +5V supply can be used to supply STO1 and STO2, and together with connecting the STO_COM with the GND, the power stage is enabled. This is shown in Figure 4.1. When both STO1 and STO2 are energized, the STO_FB output becomes active to indicate the drive is ready to be operated.

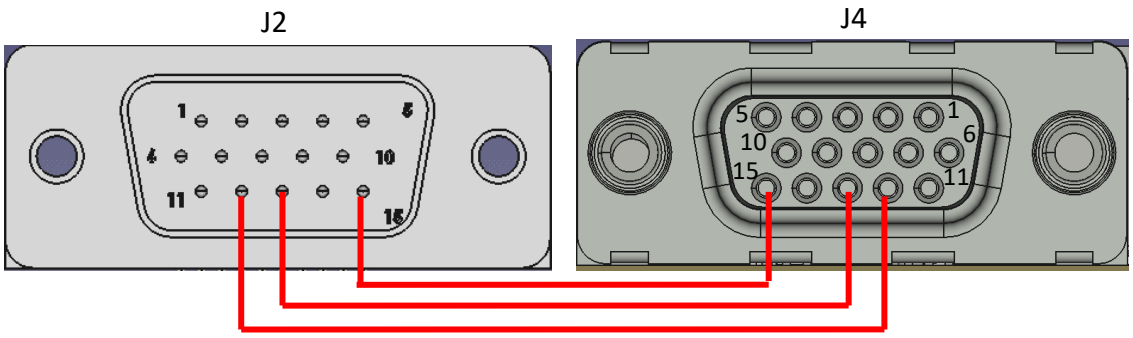


Figure 4.1. Disabling STO in VLC-25 with the on-board +5V supply.

The states of STO1 and STO2, together with the associated system variables STO_A_CMP (RW/WW1782), STO_F_CMP (RW/WW1784), STO_A_TMR (RW/WW1786) and STO_F_TMR (RW/WW1788), determine the STO_STAT (RW/WW1780) bit value, based on the table below

Bit	Purpose
0	**STO1 state.** If STO1 is energized, this bit is set to '1' and vice versa.
1	**STO2 state.** If STO2 is energized, this bit is set to '1' and vice versa.
2	**STO_A state (STO activity indicator).** If one or both STO inputs have been de-energized for 10 ms (default) or more, this bit is set to '1', meaning that STO is active and power stage is disabled. The de-energized timer is accessible through trip timer variable STO_A_TMR, and the default time period can be modified through the trip comparator variable STO_A_CMP.
3	**STO_F state (STO fault indicator).** If the two STO inputs have been in different states for 1600 ms (default) or more, this bit is set to '1', meaning there is an abnormal behavior in the way the STO inputs are energized. The timer value is accessible through the trip timer variable STO_F_TMR and the default time period can be modified through the trip comparator variable STO_F_CMP.

If either bit 2 or 3 is set to '1', the VLCI will also enter a fault mode and the fault indication LED turns ON. The MN command will reset STO_A bit but will not do the same for STO_F. The STO_F bit is reset either by power-cycling the VLCI or by writing '0' into the STO_STAT variable.

A VLC-25 Error Code Definitions

1 - ARGUMENT ERROR.

This error indicates that a command argument was not given or was specified out of the permissible numerical range.

2 - INVALID COMMAND.

This error indicates that an invalid or unrecognized command was specified in the command line.

3 - INVALID MACRO COMMAND.

This error indicates that an invalid or unrecognized command was used in defining a macro.

4 - MACRO ARGUMENT ERROR.

This error indicates that a command argument was not given or was specified out of the permissible numerical range in defining a macro.

5 - MACRO NOT DEFINED

This error indicates that execution of an undefined macro was attempted.

6 - MACRO OUT OF RANGE.

The VLC-25 allows for a maximum of 256 macros (numbered 0 to 255). This error indicates that an attempt was made to access a macro out of this boundary.

7 - OUT OF MACRO SPACE.

The VLC-25 allows for a maximum of 256 macros with up to 256 commands per macro and approximately 53983 bytes of macro storage space. Each macro command requires 6 bytes of macro storage memory and each macro requires an overhead of 1 byte. This error indicates that so many macros/macro commands have been defined that there is no remaining memory to define more.

8 - CAN'T DEFINE MACRO IN A MACRO.

This error indicates that an attempt was made to define a macro from another macro that is currently being executed. This is not allowed on the VLC-25 as to prevent loss of program control due to possible "nesting".

9 - CAN'T DEFINE MACRO WHILE SERVO ENABLED.

This error indicates that attempt was made to define a macro while a servo axis was enabled (i.e.: "MN"). This is not allowed on the VLC-25 as to prevent loss of program and/or servo control due to macro memory space definition contention.

10 - MACRO JUMP ERROR.

This command indicates that an attempt was made to jump ("MJ" command) to a command within a macro that does not exist.

11 - OUT OF MACRO STACK SPACE.

When a macro is called by another macro (via the "MC" command or a macro interrupt), the return macro and macro command number must be saved along with other internal variables. This error indicates that the memory space set aside for this purpose has been exhausted and no more "calls" may be attempted. The VLC-25 is capable of macro calls nested 25 deep.

12 - MACRO MUST BE FIRST COMMAND.

When defining a macro, the "MD" command must be the first command in the command line. This error indicates that this requirement was not met.

13 - STRING ERROR

When using a MG or VI command, no closing quote was encountered.

14 - MACRO STRING ERROR

When using a MG or VI command in defining a macro, no closing quote was encountered.

15 - SYNTAX ERROR

Indicates the improper usage of the MG or VI commands.

16 - MACRO SYNTAX ERROR

Indicates the improper usage of the MG or VI commands while defining a macro.

17 - AXIS RANGE ERROR

The axis specified is out the possible numerical range.

18 - INTERRUPT MACRO NOT DEFINED

During the course of interrupt processing, at attempt was made to go to an undefined macro.

19 - INTERRUPT MACRO STACK ERROR

Indicates that the macro stack has run out of space during interrupt processing.

20 - MACRO STACK OVERFLOW

The macro stack has run out of space.

21 - MACRO STACK UNDERFLOW

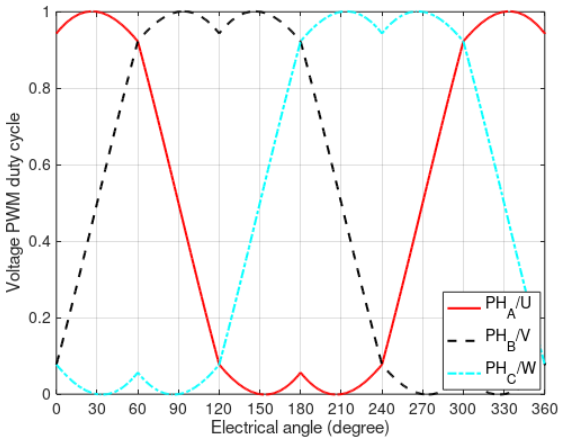
An attempt was made to "pop" a macro off of the macro stack when there was no macro to pop.

B Programming differences between VLC-25 and LAC-25

Apart from new commands that are solely relevant for 3-phase motors such as EC and DA as well as the new axis variables, there are additional commands that affect the way the software works as compared to LAC-25:

- **PS command**
  In LAC-25, macros that are defined through the MD command are automatically stored into the NVRAM (Non-Volatile Random Access Memory), whereas in VLC-25 this is not the case and the user has to execute the PS command to store the defined macros into the flash memory. Additionally, the PL command is used to load the stored macros in the flash memory. The PL command is automatically executed at VLC-25 power-up.

- **SP command**
  The voltage PWM duty cycle behaviour at the three motor terminals (U, V, W) on the VLC-25 follows the below graph, where the electrical angle 0^o - 360^o corresponds to SP value 0 – 65535. The voltage waveform shape is a result of the Space Vector Modulation (SVM), which is used in the control of 3-phase/brushless motors. During the phasing, SP value is manipulated to create actuator motion, such that the commutation angle can be determined.



It is important to note that SP is also relevant in the control of 1-phase/brushed motors. Based on the below graph, the default SP value (zero) corresponds to electrical angle of 0^o, resulting in maximum duty cycle values of approximately 0.94 and 0.08 for phases A and B, respectively. Therefore, the effective maximum voltage between motor terminals A and B is only 0.86 x (power supply voltage). Moreover, in this case, the positive voltage value corresponds to a decreasing encoder count for a positive SQ value.

In order to get the maximum voltage as well as increasing encoder count for a positive SQ value, the electrical angle has to be 150^o, corresponding to SP value of 27307. Alternatively, if PH1 is used (only in Firmware v1.51 and above), the electrical angle has to be 330^o, corresponding to SP value of 60075.

- **DA command**
  Since DA command is now used to define the absolute home position, to disable an axis (formerly `aDA`) the command is `aEAn` where n = -1, i.e. "`EA-1`".

C Program Examples

IMPORTANT NOTICE - due to the SP command behavior, execute the following command after the VLC-25 is powered up:

- *1-phase motors: aSP27307 (see the description on the SP command and appendix B for more information). Set 'a' according to the intended servo axis.*
- *3-phase motors: aPH1 (see the description on the PH command for more information). Set 'a' according to the intended servo axis.*

First-time users are also highly recommended to read Appendix E prior to trying out the program examples. Furthermore, make sure that the STO inputs are supplied by DC voltage following the schematic depicted in section 2.1.2. See section 4.3 for an alternative to realize this using the on-board +5VDC supply.

C.1 Scalar mode driving of 3-phase motors

With the scalar mode driving, 3-phase sinusoidal voltages with a fixed frequency are supplied to the motor. This creates a rotating/moving magnetic field in the stator that attracts the rotor/mover with a sufficient voltage, creating a motion. This type of check is sometimes required to verify the motor coil connection and encoder counting direction.

The following is a program example for the scalar mode driving (applied to axis 2):

```
MD1,2QM,MN,SQ5000
MD2,AL0
MD3,SP@0,AA256,WA1,RP255
MD4,MJ2
```

By executing Macro 1 (using the command `MS1`), motion can be generated. SQ value in the above program can be modified to deal with motors with high friction or heavy load. For a proper phasing and commutation of the motor, the generated motion should be in the positive direction (increasing encoder count). Otherwise, motor connection or encoder phase needs to be modified either mechanically by swapping the wires or by software through the PH command.

C.2 Phasing algorithm for 3-phase motors

The following interactive program can be used to perform phasing for 3-phase motors for both axis:

```
; AXIS 1
MD200,VI"ENTER NO. OF POLE PAIRS: ":199:N,VI"ENTER ENCODER INCREMENTS: ":201:N,VI"ENTER SQ VALUE:
":200:N
MD201,MC245,RA201,AD@199,AR197,AM@199,AR198,RA201,IE@198,RA197,MJ202,RA199,AS1,SL28,AA@201
MD202,AR202,RA197,AD4,AR196,AM11,AD10,AR195,RA196,AM9,AD10,AR194,1MF,EC0,SP16383,QM0,MN,SQ@200
MD203,WA500,RL494,AR203,SP0,WA500,RL494,AR204,AS@203,AR205
MD204,AM-1,IG@194,IB@195,MJ205,NO,MG"PHASING: FAILED!",SQ0,MF,EP
MD205,RA197,AD4,AR206,DA@206,EC@202,SQ0,MG"PHASING: SUCCESSFUL!",MF,EP
; MD245-248: current sensing offset adjustment, automatically called when phasing is executed
MD245,AL2048,WW606,WW608,AL0,AR400,AR401
MD246,GA1,AA@400,AR400,GA2,AA@401,AR401,WA1,RP999
MD247,RA400,AM-1,AD1000,AA2048,WW606
MD248,RA401,AM-1,AD1000,AA2048,WW608,RC
```

```
; AXIS 2
MD210,VI"ENTER NO. OF POLE PAIRS: ":299:N,VI"ENTER ENCODER INCREMENTS: ":301:N,VI"ENTER SQ VALUE:
":300:N
MD211,MC250,RA301,AD@299,AR297,AM@299,AR298,RA301,IE@298,RA297,MJ212,RA299,AS1,SL28,AA@301
MD212,AR302,RA297,AD4,AR296,AM11,AD10,AR295,RA296,AM9,AD10,AR294,2MF,EC0,SP16383,QM0,MN,SQ@300
MD213,WA500,RL702,AR303,SP0,WA500,RL702,AR234,AS@303,AR305
MD214,AM-1,IG@294,IB@295,MJ215,NO,MG"PHASING: FAILED!",SQ0,MF,EP
MD215,RA297,AD4,AR306,DA@306,EC@302,SQ0,MG"PHASING: SUCCESSFUL!",MF,EP
; MD250-253: current sensing offset adjustment, automatically called when phasing is executed
MD250,AL2048,WW814,WW816,AL0,AR402,AR403
MD251,GA5,AA@402,AR402,GA6,AA@403,AR403,WA1,RP999
MD252,RA402,AM-1,AD1000,AA2048,WW814
MD253,RA403,AM-1,AD1000,AA2048,WW816,RC
```

By executing the phasing program example above, the user will be asked to specify:

- Number of pole pairs (for linear actuators: 1, for rotary motors: enter accordingly)
- Encoder increments (for linear actuators: same as EC value, for rotary motors: encoder counts/revolution). For more information, see the discussion on EC command in **Section 3.3** of this manual.
- SQ value: for a start, use 10000. Higher values may be necessary in the presence of high friction or a heavy load.

The above phasing program is rather simple and would work for linear actuators without a spring attached to its shaft or rotary motors without a mechanical limit. A more robust phasing program, which would also work on linear actuators with a spring attached, is given as follow

```
; AXIS 1
MD200,AL0,AR214,VI"ENTER NO. OF POLE PAIRS: ":199:N,VI"ENTER ENCODER INCREMENTS:
":201:N,VI"ENTER SQ VALUE: ":200:N
MD201,MC245,RA201,AD@199,AR197,AM@199,AR198,RA201,IE@198,RA197,MJ202,RA199,AS1,SL28,AA@201
MD202,AR215,1MF,EC0,AL32767,AR202,AL8192,AR204,AM@197,IG0,AD65536,MJ203,RA197,AD65536,AM@204
MD203,AR205,AM9,AD10,AR206,RA205,AM11,AD10,AR207
MD204,SP0,QM0,MN,SQ@200,WA100,MC212,AL1,AR229,MC209
MD205,AL0,AA@204,AR209,SP@209,WA100,MC212,AL3,AR229,MC209,AL1,AR229,MC210
MD206,AL65535,AS@204,AR209,SP@209,WA100,MC212,AL5,AR229,MC209,AL4,AR229,MC210
MD207,SP@202,WA100,MC212,AL1,AR229,MC209,AL1,AR229,MC211,AL3,AR229,MC209,AL1,AR229,MC210
MD208,AL4,AR229,MC211,AL5,AR229,MC209,AL4,AR229,MC210,MJ216
MD209,RL494,JR@229,AR203,JR4,AR208,JR2,AR228,RC
MD210,JR@229,RA208,AS@203,JR3,RA203,AS@228,IG@206,IB@207,MJ213,RC,RC
MD211,JR@229,RA202,AA@204,JR3,RA202,AS@204,AR209,SP@209,WA100,MC212,RC
MD212,RL494,AR211,WA10,RL494,AS@211,IE0,NO,RC,RP
MD213,RW604,IB0,AA65535,AR210,AR210
MD214,RA197,AD4,AR212,RA210,AM@197,AD65535,AA@212,AR213,DA@213,EC@215,SQ0
MD215,MG"PHASING: SUCCESSFUL!",AL1,AR214,UM1,MF,EP
MD216,MG"PHASING: FAILED! MAKE SURE THE PARAMETERS ARE CORRECT!",AL2,AR214,SQ0,MF,EP
; MD245-248: current sensing offset adjustment, automatically called when phasing is executed
MD245,AL2048,WW606,WW608,AL0,AR400,AR401
MD246,GA1,AA@400,AR400,GA2,AA@401,AR401,WA1,RP999
MD247,RA400,AM-1,AD1000,AA2048,WW606
MD248,RA401,AM-1,AD1000,AA2048,WW608,RC


; AXIS 2
MD220,AL0,AR314,VI"ENTER NO. OF POLE PAIRS: ":299:N,VI"ENTER ENCODER INCREMENTS:
":301:N,VI"ENTER SQ VALUE: ":300:N
MD221,MC250,RA301,AD@299,AR297,AM@299,AR298,RA301,IE@298,RA297,MJ222,RA299,AS1,SL28,AA@301
MD222,AR315,2MF,EC0,AL32767,AR302,AL8192,AR304,AM@297,IG0,AD65536,MJ223,RA297,AD65536,AM@304
MD223,AR305,AM9,AD10,AR306,RA305,AM11,AD10,AR307
MD224,SP0,QM0,MN,SQ@300,WA100,MC232,AL1,AR329,MC229
MD225,AL0,AA@304,AR309,SP@309,WA100,MC232,AL3,AR329,MC229,AL1,AR329,MC230
```

```
MD226,AL65535,AS@304,AR309,SP@309,WA100,MC232,AL5,AR329,MC229,AL4,AR329,MC230
MD227,SP@302,WA100,MC232,AL1,AR329,MC229,AL1,AR329,MC231,AL3,AR329,MC229,AL1,AR329,MC230
MD228,AL4,AR329,MC231,AL5,AR329,MC229,AL4,AR329,MC230,MJ236
MD229,RL702,JR@329,AR303,JR4,AR308,JR2,AR328,RC
MD230,JR@329,RA308,AS@303,JR3,RA303,AS@328,IG@306,IB@307,MJ233,RC,RC
MD231,JR@329,RA302,AA@304,JR3,RA302,AS@304,AR309,SP@309,WA100,MC232,RC
MD232,RL702,AR311,WA10,RL702,AS@311,IE0,NO,RC,RP
MD233,RW812,IB0,AA65535,AR310,AR310
MD234,RA297,AD4,AR312,RA310,AM@297,AD65535,AA@312,AR313,DA@313,EC@315,SQ0
MD235,MG"PHASING: SUCCESSFUL!",AL1,AR314,UM1,MF,EP
MD236,MG"PHASING: FAILED! MAKE SURE THE PARAMETERS ARE CORRECT!",AL2,AR314,SQ0,MF,EP
; MD250-253: current sensing offset adjustment, automatically called when phasing is executed
MD250,AL2048,WW814,WW816,AL0,AR402,AR403
MD251,GA5,AA@402,AR402,GA6,AA@403,AR403,WA1,RP999
MD252,RA402,AM-1,AD1000,AA2048,WW814
MD253,RA403,AM-1,AD1000,AA2048,WW816,RC
```

A failed phasing could be caused by incorrect values of SQ, EC or an improper motor/encoder connection, which can be verified using the scalar mode driving program example in **Appendix C.1** of this manual.

C.3 Open loop torque mode

With this mode, a constant voltage level is supplied to the motor. This check is typically done to verify motor connection or as a simple torque/force check (applied to axis 1):

```
MD1,1QM,MN,SQ10000
```

By executing Macro 1, a voltage level corresponding to (SQ value/32767)*(applied DC bus voltage) is supplied to the motor. Note that to execute this mode in the case of 3-phase motors, phasing needs to be performed successfully beforehand.

C.4 Homing for a linear actuator

Suppose the linear actuator encoder count needs to be set to 0 at the fully retracted positions, the following program example could be used (applied to axis 1):

```
MD1,1SG10,SD100,SV100000,SA100000 ;PID and motion profile config
MD2,DI1,VM,MN,GO,WA100      ;Velocity move to negative direction
MD3,RW538,IB-500,NO,MJ4,RP  ;Check if following error < -500
MD4,ST,DH,MF,DI0            ;If the above is true, stop, define home
```

Note that to execute the above, the user needs to verify if the PID and motion profile configurations are proper for the actuator being used. Also, the following error detection level may need to be modified depending on the actuator orientation, presence of load, encoder resolution, etc.

C.5 Repetitive point-to-point position move of 2 axes

A simple repetitive point-to-point position move of 2 servo axes can be realized using the following program example

```
MD10,1SG5,SD50,2SG8,SD50,1SV2000000,SA2000,2SV50000,SA500 ;PID and motion
profile configs of both axes

MD11,0PM,MN; Enable position mode and turn on both motors

MD12,1MA0,GO,2MA0,GO,0WS50,1MA20000,GO,2MA4000,GO,0WS50,RP ;repetitive
position moves
```

C.6 Current mode

When it is required that a certain level of current (which also corresponds to torque based on the motor's torque constant), current mode of operation can be used. Suppose that 1 A needs to be supplied to the motor, the equivalent IMON value would be (1 A/13 A) × 2047 = 157, the following example realizes it (applied to axis 1)

```
MD1,1SC1000,QM1,MN,SQ157
```

Note that SC value is to be determined experimentally. If a better current tracking performance is desired, the CMPG variable (RW/WW588) can also be tuned to improve the performance. Additionally, variables LPG1 and LPG2 can be modified to reduce current measurement noise.

C.7 Configuring I2T parameters

Consider the following case (see **section 4.2** for explanations on the I2T parameters):

- Axis 1:
  - Continuous current: 3 A → i2t_NOM = (3A/13A) × 2047 = 472
  - Peak current: 6 A, allowed for 1 second → IOL = (6 A/13 A) × 2047 = 945, i2t _TRIP = (945^2-472^2) × 1000 = 670,241,000
- Axis 2:
  - Continuous current: 3 A → i2t_NOM = (3A/13A) × 2047 = 472
  - Peak current: 9 A, allowed for 2 seconds → IOL = (9 A/13 A) × 2047 = 1417, i2t _TRIP = (1417^2-472^2) × 2000 = 3,570,210,000
  - Since i2t _TRIP exceeds the maximum limit (2^{31}-1), evaluation of i2t_INT has to be done at a slower interval. Assume that the evaluation is to be done every 10 milliseconds, therefore i2tINTRVL = 9 , furthermore, i2t _TRIP becomes (1417^2-472^2) × 200 = 357,021,000

The following is an example on implementing the above parameters in a program

```
;axis 1 i2t parameters
MD20,AL472,WW610,AL670241000,WL612

;axis 2 i2t parameters
MD21,AL472,WW818,AL9,WB828,AL357021000,WL820
```

C.8 I2T and STO fault management through interrupt

An axis drive power is disabled when a fault occurs, which can be triggered by overcurrent, over-temperature and i2t events on the respective drive, as well as the STO becoming active. In certain cases, instead of letting the drive power to be disabled, the user can also choose for an action to be taken by the controller, by means of an interrupt.

In this example, the program detects whether I2T or STO or an unknown fault occurs. In case of an I2T fault, the program outputs a corresponding notification on the command prompt and set the max. torque (SQ value) to 10000 and stop the axis motion. As for the STO case, the program outputs a notification and turns on a digital output.

```
;axis 1 i2t parameters. i2t_TRIP value is copied to register 21.
MD20,AL472,WW610,AL640241000,WL612,AR21

;axis 2 i2t parameters. i2t_TRIP value is copied to register 22.
MD21,AL472,WW818,AL9,WB828,AL357021000,WL820,AR22

;set axis fault interrupt-triggered macro calls and activate them for both
axis.
MD22,AL31,LV27,AL35,LV26,EV27,EV26

;repetitive position move for both axis.
MD23,0PM,MN
MD24,1MA0,GO,2MA0,GO,0WS50,1MA20000,GO,2MA4000,GO,0WS50,RP

;macro to be called when axis 1 interrupt becomes active
MD31,RW1780,IS2,MJ40,NO,RL616,IG@21,MC32,NO,MJ33

;axis 1 i2t fault action
MD32,MG"axis 1 i2t fault",1SQ10000,EP

;axis 1 unknown fault indication
MD33,MG"unknown fault at axis 1",EP

;macro to be called when axis 2 interrupt becomes active
MD35,RW1780,IS2,MJ40,NO,RL824,IG@22,MC36,NO,MJ37

;axis 2 i2t fault action
MD36,MG"axis 2 i2t fault",2SQ10000,EP

;axis 2 unknown fault indication
MD37,MG"unknown fault at axis 2",EP

;STO fault indication (occurs on both axis)
MD40,MG"STO fault",CN0,EP
```

Notes:

- The macro interrupt system only works when a macro is executing.
- The macro interrupt system is reinitialized when MS, MJ or MC is executed from the prompt. Therefore, the interrupt has to be enabled (through EV command) within a sequence of macro jumps/calls towards the main program.

- After the interrupt macro call, the program will return to the main one that was running when the interrupt got triggered, unless an EP or MJ command is executed at the end of the interrupt macro.
- To clear the fault, use the MN command. Prior to this, the motor has be in an OFF state. Consider to execute MF in the interrupt macro.

D Command Summary

Parameter Commands		
DB: Set Dead Band	MR: Move Relative	**Learned Pos. Storage Commands**
FA: Feed-forward, Acceleration	PM: Position Mode	LP: Learn Current Position
FR: Derivative Sampling Freq.	QM: Torque Mode	LT: Learn Target Position
FV: Feed-forward, Velocity	SE: Set Max. Following Error	MP: Move To Position
GR: Gear Ratio	SP: Set Commutation Phase	
IL: Integration Limit	ST: Stop Motion	**Macro Commands**
OM: Output Mode	VM: Velocity Mode	DV: Disable Interrupt Vector
OO: Output Offset		EV: Enable Interrupt Vector
PH: Phase	**Register Commands**	JP: Jump Absolute
RI: Sampling Rate of Integral	AA: Accumulator Add	JR: Jump Relative
SA: Set Acceleration	AC: Accumulator Complement	LV: Load Interrupt Vector
SC: Set Current Gain	AD: Accumulator Divide	MC: Macro Call
SD: Set Derivative Gain	AE: Accumulator Exclusive-OR	MD: Macro Definition
SG: Set Proportional Gain	AL: Accumulator Load	MJ: Macro Jump
SI: Set Integral Gain	AM: Accumulator Multiply	MS: Macro Sequence
SQ: Set Torque	AN: Accumulator And	NB: Interrupt on Bit
SS: Set Servo Speed	AO: Accumulator Or	RC: Return from Call
SV: Set Velocity	AR: Copy Accum. to Register	RM: Reset Macro
	AS: Accumulator Subtracted	TM: Tell Macro
Reporting Commands	RA: Copy Register to Accum.	UM: Unpush Macro
TA: Tell A/D Channel	RB: Read Byte	
TB: Tell Breakpoint	RL: Read Long	**I/O Commands**
TC: Tell Channel	RW: Read Word	BI: Bulk I/O Input
TD: Tell Derivative Gain	SL: Accumulator Shift Left	BO: Bulk I/O Output
TE: Tell Last Command Error	SR: Accumulator Shift Right	CF: Turn Channel “Off”
TF: Tell Following Error	TR: Tell Register	CH: Make Channel Active “On”
TG: Tell Position Gain	WB: Write Byte	CL: Make Channel Active “Off”
TI: Tell Integral Gain	WL: Write Long	CN: Turn Channel “On”
TK: Tell Constants	WW: Write Word	ID: Input Debounce Delay
TL: Tell Integration Limit		
TM: Tell Macros	**Sequence Commands**	**Serial and Misc. Commands**
TO: Tell Optimal Position	DF: Do if Channel “Off”	BK: Break
TP: Tell Real Position	DN: Do if Channel “On”	BR: Baud Rate
TQ: Tell Torque	EP: End Program	CD: Capture Data
TR: Tell Register	IB: If Accumulator Below	CS: Capture Storage
TS: Tell Status	IC: If Accumulator Bit is Clear	DD: Dump Data
TT: Tell Target Position	IE: If Accumulator Equal to ‘n’	DM: Decimal Mode
TV: Tell Current Velocity	IF: If Channel “Off”	EF: Echo Off
VE: Tell Version	IG: If Accumulator is > ‘n’	EN: Echo On
	IN: If Channel is “On”	GA: Get A/D Channel
Motion Commands	IP: Interrupt on Absolute Pos.	HM: Hexadecimal Mode
AB: Abort Motion	IR: Interrupt on Relative Pos.	MG: Display Message
CI: Capture Index	IS: If Accumulator Bit is Set	NO: No Operation
DA: Define Absolute Home	IU: If Accum. Unequal to ‘n’	PL: Parameter Load
DH: Define Home	RP: Repeat	PS: Parameter Save
DI: Set Direction	WA: Wait	RT: Reset
EA: Enable Axis	WE: Wait for Edge (coarse home)	VI: Variable Input
EC: Electronic Commutation	WF: Wait for Channel “Off”	ZD: Read Capture Data Buffer
EG: Electronic Gearing	WI: Wait for Index	ZF: Format Flash Memory
FE: Find Edge (Home)	WN: Wait for Channel “On”	ZZ: Dump Memory
FI: Find Edge (Index)	WP: Wait for Absolute Position	
GH: Go Home	WR: Wait for Relative Position	**Bit Commands**
GO: Go (start motion)	WS: Wait for Stop	BC: If Bit Clear
MA: Move Absolute		BS: If Bit Set
MF: Motor Off		CB: Clear Bit
MN: Motor On		SB: Set Bit

E For first time users of VLCs: PLEASE READ THIS SECTION

IMPORTANT NOTICE - By default, all actuator safety-related parameters to avoid overtemperature that could damage the actuator (such as I^2T and `SE` *command) are* ***not pre-configured****. The user will have to configure these parameters as required by the application.*

Prior to operating an actuator (and possibly testing the program examples in Appendix C), a few checks can be performed on the VLC and actuator to ensure that the whole system (including cabling) behaves properly for each axis:

- Check whether the position value can be read by executing this command: `TP`
- If possible, move the actuator shaft manually while executing `TP` a few times and verify the following:
    - For linear actuators: by default, a forward movement (shaft extension) has to result in an increasing position value. Otherwise, `PH2` command has to be executed to reverse the encoder counting direction by software.
    - For rotary actuators: this could either be in clockwise or counter-clockwise direction. Execute `PH2` if necessary to get an increasing position value in the direction required by the application.
- Check if an open-loop voltage driving of the actuator results in a movement, verify the following:
    - For single-phase linear actuators and rotary motors: execute the command sequence `SP27307,QM,MN,SQ5000` and observe if the shaft moves in a positive direction (increasing position value). `SQ` value corresponds to the voltage duty cycle fed to the actuator, set the value wisely to produce a sufficient level of force to move the actuator shaft, but not too high such that it heats up the actuator excessively if the actuator power is not switched OFF (by executing `MF`) for a long time. See the description of `SQ` command for more information.
    - For three-phase linear actuators and rotary motors: perform a similar observation as the above, but instead of the command sequence above, create the following macro sequence and execute `MS1`

      ```
      MD1,QM,MN,SQ5000
      MD2,AL0
      MD3,SP@0,AA256,WA1,RP255
      MD4,MJ2
      ```

      Remember to type in the ESC button to stop the execution of the above macro sequence and `MF` to switch off the actuator power. To slow-down the movement, increase `WA` value in `MD3` above.

- If the generated movement from the previous step is in the negative direction, motor wires will have to be swapped physically or through software at the VLC terminal:
    - For single-phase linear actuators and rotary motors: Swap the MOT+ and MOT- (typically red and black coloured) wires, or, execute PH1 (or PH3, if PH2 in the has been executed previously).
    - For three-phase linear actuators and rotary motors: Swap any two of the U, V and W (typically red, black and white coloured) wires, or, execute PH1 (or PH3, if PH2 in the has been executed previously).
- Based on the previous checks, make sure that:
    - For single-phase actuators and rotary motors: SP27307 is executed after the controller is powered-up (possibly in macro 0 or any other macros prior to the execution of the initialization/homing procedure).
    - For all actuator/motor types: PH command with the corresponding value based on the previous checks, is executed after the controller is powered-up (possibly in macro 0 or any other macros prior to the execution of the initialization/homing procedure).
- If any of the suggested solutions above does not solve the issue, contact SMAC Tech Support.

After the previous checks, the user can try out the program examples in Appendix C of this manual. Be mindful about the following:

- The smaller the actuator size, the more prone it is to overtemperature damage. Make sure that SQ value is not set too high when executing the QM mode. One could also limit the SQ value that is applied internally by the VLC during a position or velocity move. Simply add SQ<desired value> right after PM or VM command.
- During the initial motion program testing, be ready to hit the ESC button and execute MF command, to stop the actuator execution and shut it off, in the event that the actuator exhibits signs of overheating (by sensing the temperature of the actuator housing with hand or a temperature sensor, when possible) or control instability (unstable movements and/or high-pitched noise coming out of the actuator). Alternatively, depending on the VLC type, remove the voltage supply connected to the STO inputs to immediately shut-off the power to the actuator.
- Safety measures: It is always safer to start with lower values of SE or I^2T parameters, although these may be too low that the fault/servo error would be triggered prematurely. Increase the parameter values gradually until a stable operation (based on the application requirement) is achieved.

F Command Index