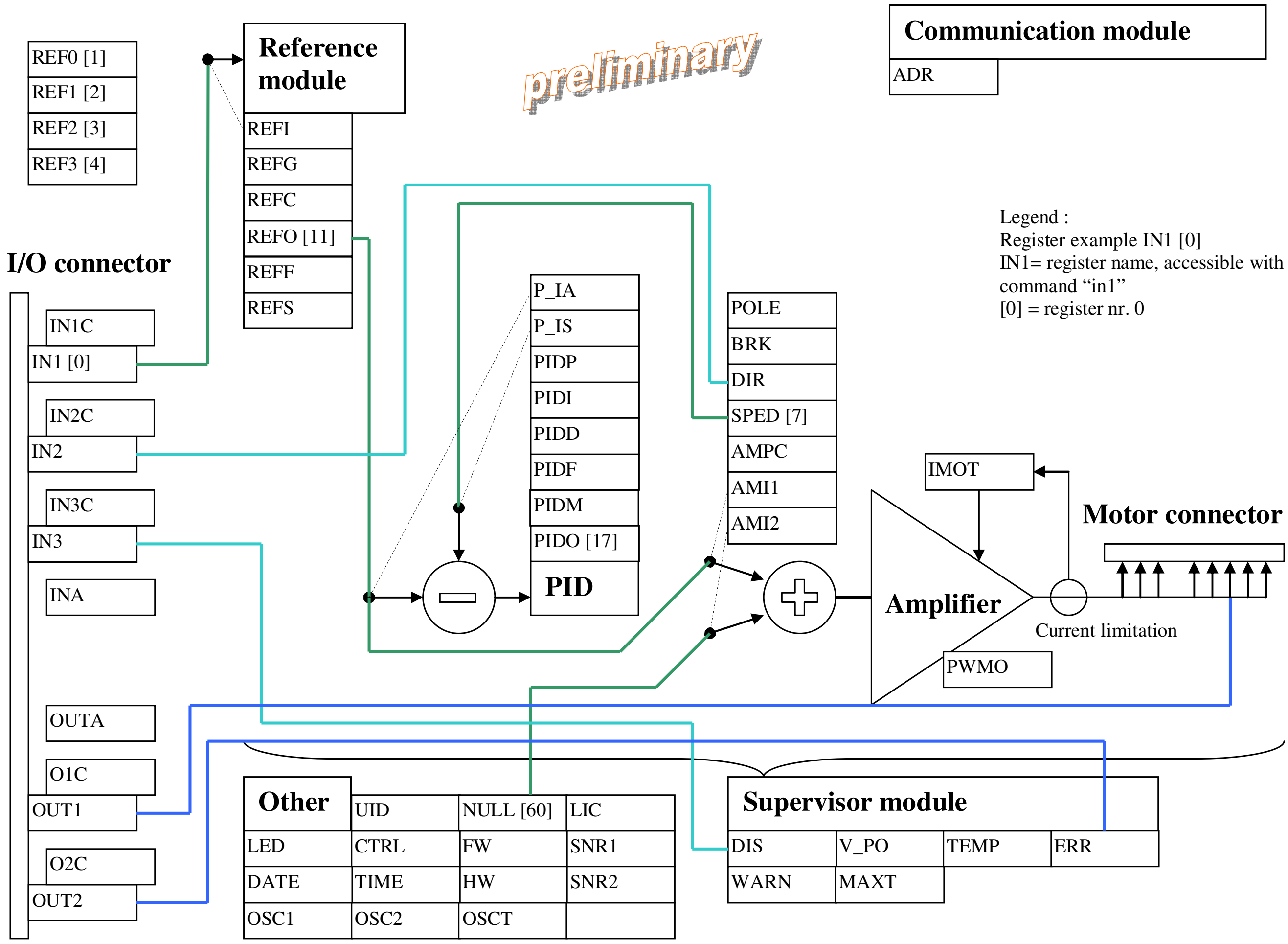
Tinaxis-L-BLDC-150-sensored BLDC driver 9-30V, 5A, digital

Software manual



Legend :
Register example IN1 [0]
IN1= register name, accessible with command "in1"
[0] = register nr. 0

Figure 1, block diagram, with lines showing default configuration

Registers

The registers are 16 bit sign numbers.

Nr.	Register	Description	status
0	In1	Input 1, analog value in mV Range is 0 to 27V, accuracy ±0.15V ±5%	RO
24	In2	Input 2, digital, threshold ~9V By default this input is linked to DIR (please see IN2C register)	RO
25	In3	Input 3, digital, threshold ~9V By default this input is linked to DIS (please see IN2C register)	RO
28	Inc1	Input 1 config	RW *flash*
29	Inc2	Input 2 config Bit 7= invert polarity Bit 0 to 2: configuration: 0: not linked 1: linked to DIR (rotation direction) 2: linked to DIS (disable) 3: linked to BRK (brake) 4: linked to REFS bit 0	RW *flash*
30	inc3	Input 3 config Bit 7= invert polarity Bit 0 to 2: configuration: 0: not linked 1: linked to DIR (rotation direction) 2: linked to DIS (disable) 3: linked to BRK (brake) 4: linked to REFS bit 1	RW *flash*
23	ina	Input 1, 2 and 3 paked together as digital	RO
26	out1	ouput 1, digital, open collector with 100 Ohm serial resistor.	RW
27	out2	ouput 2, digital, open collector with 100 Ohm serial resistor.	RW
31	oc1	Output 1 config Bit 7= invert polarity Bit 0 to 2: configuration: 0: not linked 1: linked to Hall sensor 2: linked to ERR (Error register) 3: linked WARN (Warning register)	RW *flash*
32	oc2	Output 2 config Bit 7= invert polarity Bit 0 to 2: configuration: 0: not linked 1: not available 2: linked to ERR (Error register) 3: linked WARN (Warning register)	RW *flash*
	err	Error: bit 0= temperature error bit 1= external STOP function ("br" command)	Read, write only 0
	warn	Warning: bit 0= motor not mooving	Read, write only 0
1	ref0	Register available to store reference The value is free, from -32767 to + 32767.	RW

		Mostly used to store 1 of the 4 possible reference speed memory, and activated with IN2/IN3	
2	ref1	same as ref0	RW *flash*
3	ref2	same as ref0	RW *flash*
4	ref3	same as ref0	RW *flash*
36	refi	Reference module input selector. It is the nr. of the register to take the value from. Example: if REFI=0 the value is taken from IN1, if REFI= 3, the value is taken from REF2.	RW *flash*
37	refc	Reference module config 0: REFI is freely modifiable by the used. By default it’s value is 0, that mean that reference module takes it’s reference from register 0 (IN1) 1: REFI is automatically updated according IN2 and IN3; using REFS register. REFI= REFS+1 in order to select 1 of the 4 reference hold in REF0 to REF3 registers	RW *flash*
37	refs	Reference module source (selected by inputs) REFS is automatically updated according IN2 and IN3; if INC2 and/or INC3 is configured to give value to ref module. The REFS value is therefore 0 to 3	RO
9	refg	Reference gain. Value 1024=100%	RW *flash*

11	refo	REFO is calculated with this formula:
		REFO=value[REFI]*REFG/1024
		Example:
		REF3=1000
		REFG=2048
		REFI=4
		
		Then value[REFI]=REF3=1000
		REFO=1000*2048/1024=2000
		
		REFO is generally used as input value for the PID setpoint or amplifier value.
		
		REFO refresh rate is 1kHz
			RO

10	reff	Reserved for compatibility with other products (filter parameter)	RW *flash*
47	p_ia	Register number used to get the non inverted input of the PID	RW *flash*
48	p_is	Register number used to get the inverted input of the PID	RW *flash*
13	pidp	P factor (output of PID proportional to this value) 1024= gain 1:1	RW *flash*
14	pidi	I factor of PID (integral value) 1024= gain of 1/1024 each ms	RW *flash*
15	pidd	Derivator factor 1024= gain 1 of difference between actual sample and previous sample. Generally unused	RW *flash*
21	pidf	PID feed forward: output= value[P_IA]*PIDF/1024	RW *flash*
17	pido	PID output; holds the result of the PIDPID. Generally used as input for the amplifier.	RO
16	pidm	PID minimum value threshold limit value to enable the PID. Below this value, only the feed-forward is used. By default: 0 (PID always running)	RW *flash*
7	sped	Speed measured. This value is zero when rotation speed is lower than ~900RPM (1 pair of poles) Precision: ±2%	RO
52	pole	Motor Pole count. example: 1 pair of poles, write 2	RW *flash*

44	ami1	Amplifier input Register number used to get the input of the amplifier. By default it is 11, to get value of REFO register. Often used with value 17, to get the value of PIDO	RW *flash*
45	ami2	Amplifier input It is added to value[AMI1]. Generally unused, that mean it's value is 60 to take value 0 from the NULL register	RW *flash*
50	brk		RW
33	dir		RW
60	null	NULL register (parking for any vector), it's value is always 0	RO
41	led	Reserved for compatibility with other products (LED status/configuration)	

55	date	Date of firmware. Format: YYMMI
		YY= year coded in 2 digit
		MM=month coded in 2 digit
		I= index incremented if more than 1 version is provided during a month
			RO

53	fw	Firmware version	RO
54	hw	Hardware version	RO
8	time	Time counter, incremented at 1khz	RW
56	snr1	Reserved for compatibility with other products (serial nr.)	RO
57	snr2	Reserved for compatibility with other products (complementary serial nr.)	RO
58	uid	Reserved for compatibility with other products (unique id)	
59	lic	Reserved for compatibility with other products (license nr)	
38	osc1	Reserved for compatibility with other products (oscilloscope)	
39	osc2	Reserved for compatibility with other products (oscilloscope)	
40	osct	Reserved for compatibility with other products (oscilloscope)	
	maxt	Maximum allowed temperature; above it generates an error in "err" register and stop the motor	
	temp	Temperature measured near power bridge (°C)	

Legend:
RW= read and write possible
RO= read only register
flash= register saved in flash (using "save" command).

Commands

Register	Description
save	Save the configuration When the registers are set correctly, sending this command will record the registers to flash memory. Therefore after power OFF->ON, the regiters will be the same.
pc	Show all registers and command
sr	Ping command. (return FD)
help	Get help of link to help on internet
wiz	Wizard: some predefined set of parameters can be uploaded directly. Use "save" command at the end. 0: Open loop 1: closed loop speed 100: simple rotation (test)
rst	Reset the board
br	Stop motor (toogle)
tm	Demo move

Syntax

Description	*Examples*
asking the content of a register: type the register	out1
modifying a register: type the register name + space + value	out1 1 ref0 -4250
save only 1 register: type the register name+ ## + value	ami1##3
get the saved value: type the register name+ ##	ami1##

Embedded memories

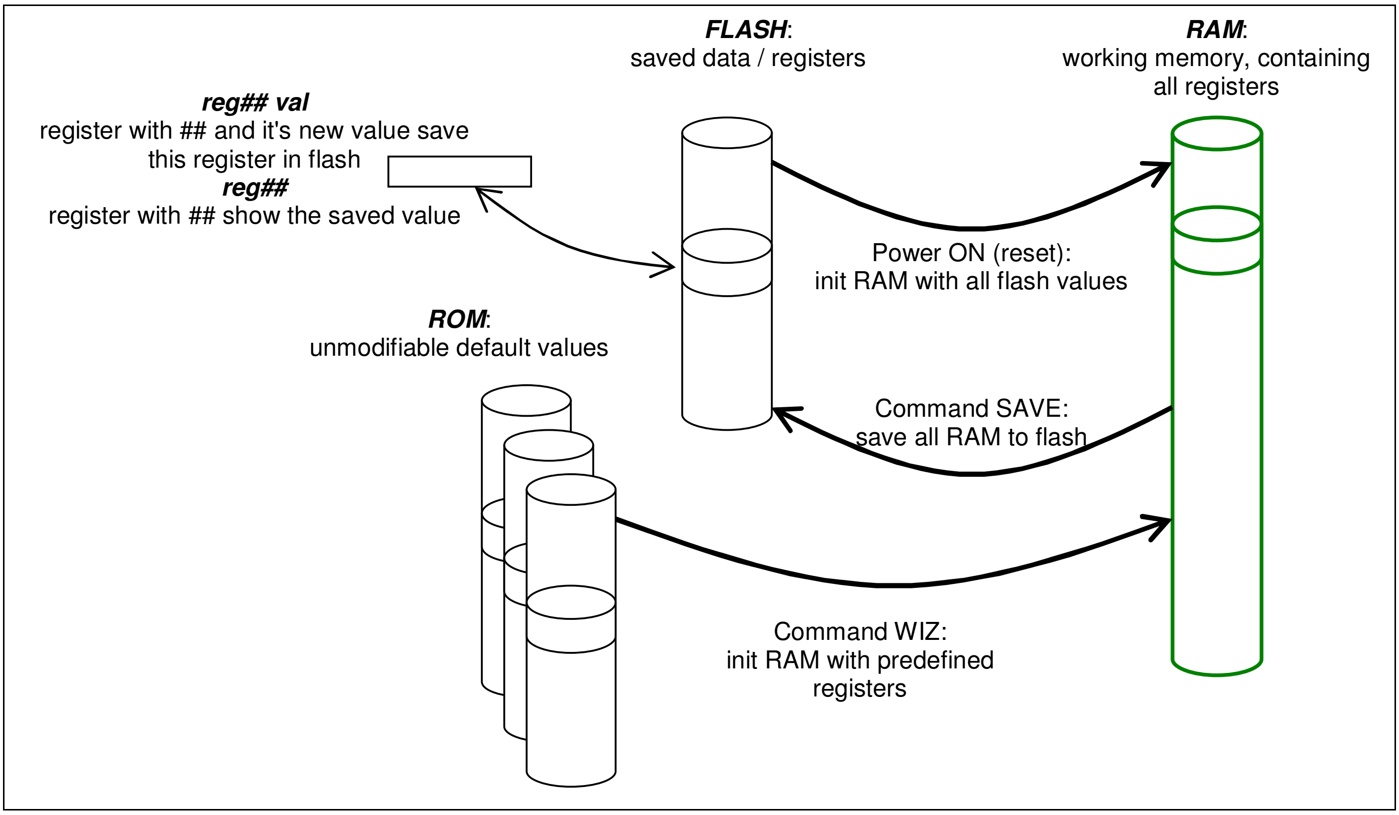

FLASH:
saved data / registers
RAM:
working memory, containing all registers
reg## val
register with ## and it's new value save this register in flash
reg##
register with ## show the saved value
Power ON (reset):
init RAM with all flash values
ROM:
unmodifiable default values
Command SAVE:
save all RAM to flash
Command WIZ:
init RAM with predefined registers

Figure 2, memory organization

How to configure

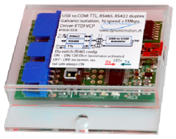
To configure the board, there is 2 options:

1. Modifying the registers
2. Ask the manufacturer to adapt the registers before shipping

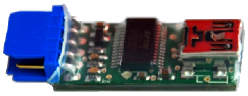
Necessary hardware

The required hardware to make any change into registers is an interface USB to Tinaxis

Example: P000-034 (high-end version, with galvanic isolation)



or low cost version P000-036 (without galvanic isolation)



Any other virtual com port adapter with TTL outputs is suitable
RS232 to TTL adapter based on MAX232 chip or similar is also suitable

3 pins are necessary: GND, TX and RX. See P000-034 documentation for more details

Software environment

The requires software is freely available at our web portal:
www.dynamicmotion.ch PRODUCT->DOWNLOAD, then select "setup dynamic motion.exe". It's an installer containing software and documentation for most of our boards.

Driver for USB interface:
the USB interface uses a chipset from FTDI manufacturer and require a driver for virtual com port functionality. On some systems (Windows 7 32bit), when connecting the interface the first time, windows takes 1-3 minutes to locate the driver on the web and install it automatically. On other systems, the driver must be installed manually (from the web, search "FTDI VCP" and download the correct driver from ftdi.com; from the install directory, by default under "c:\programmes files\dynamic motion programming suite\usb drivers\ftdi"), locate the driver and install it.

Example

Step 1: download and install environment on your Windows based computer

Step 2: Open "Dynamic Motion Communication Tool"

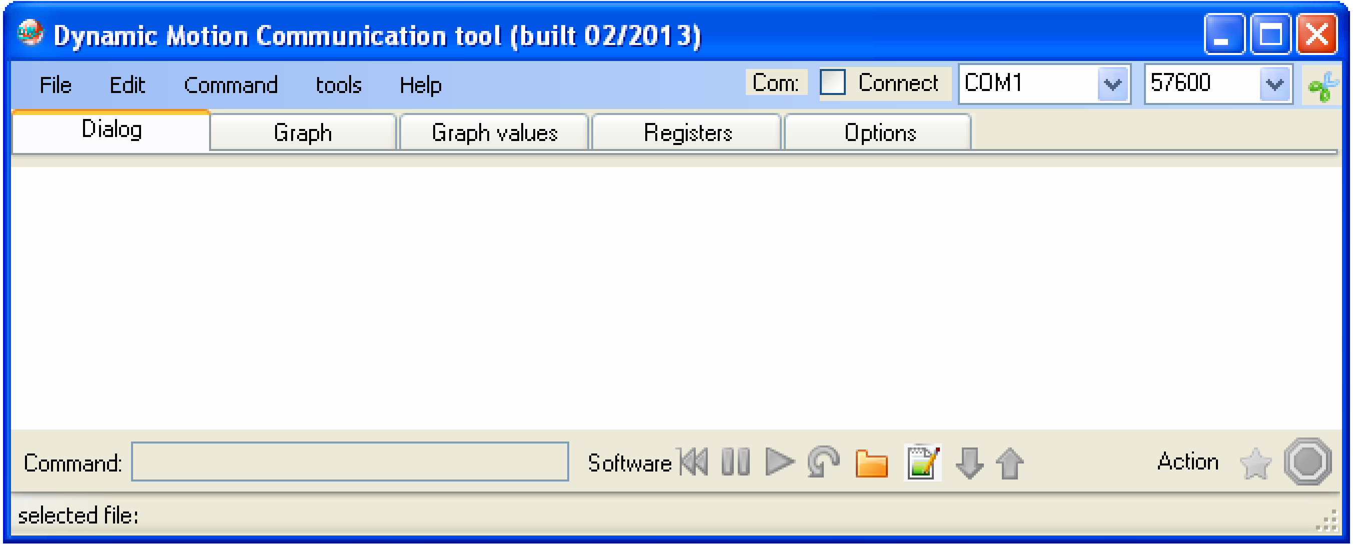


Figure 3, software when successfully installed

Step 3: Connect the board

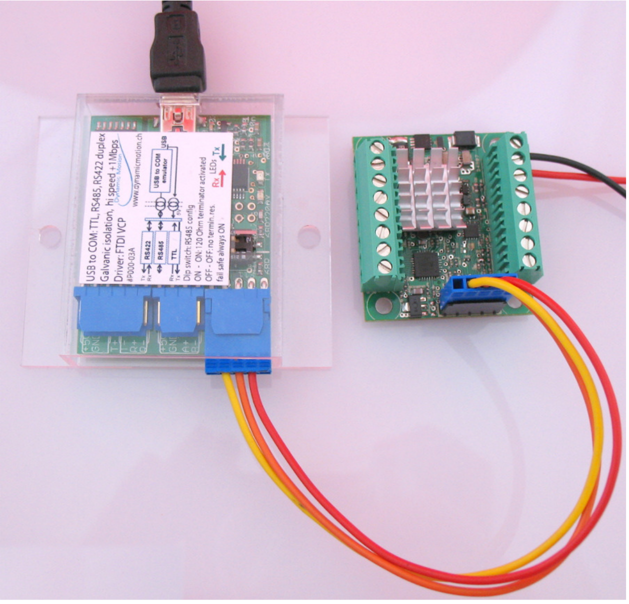


Figure 4: Supply the board with an external source and connect the interface

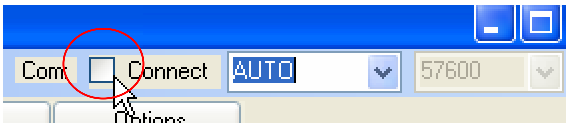


Figure 5, open communication between computer and board: click on connect

If the USB interface is correctly recognized by the computer and the board is powered, connection should occur.

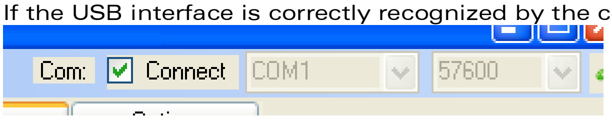
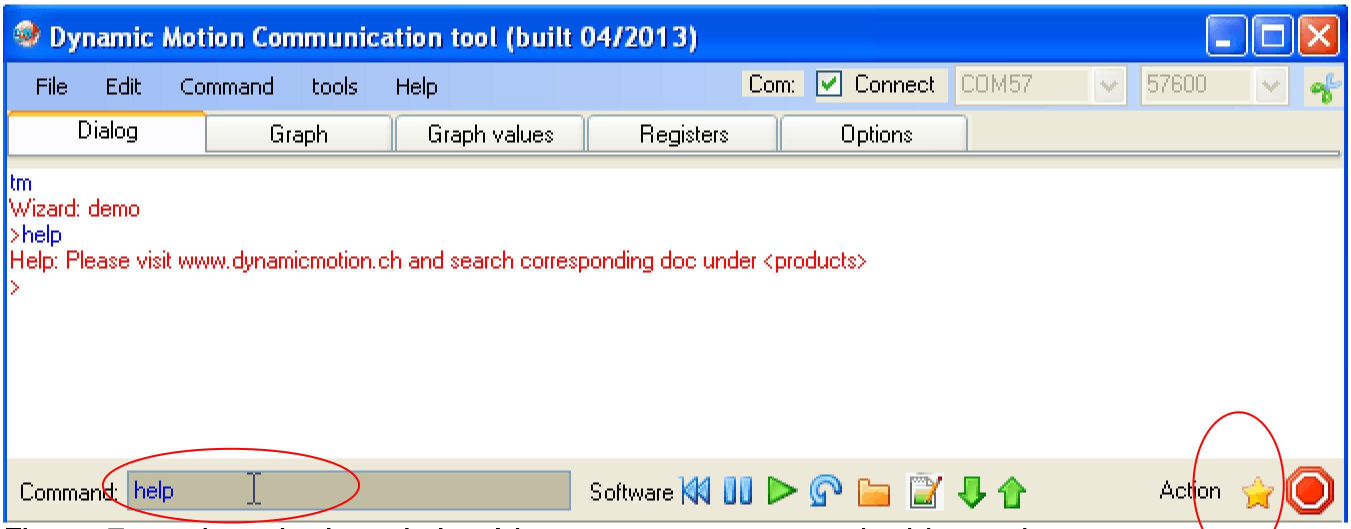


Figure 6, when connection is done

Check:
click on the star button or type a command (validate the command with):

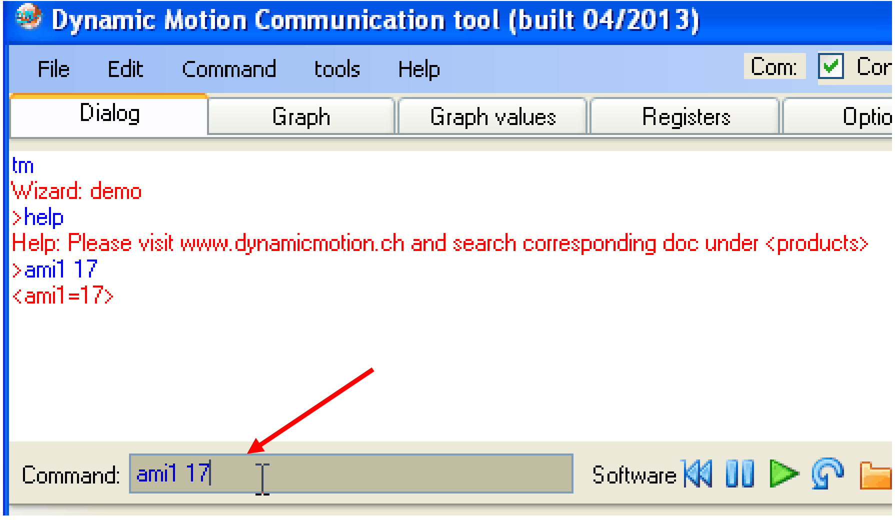


```
tm
Wizard: demo
>help
Help: Please visit www.dynamicmotion.ch and search corresponding doc under <products>
>
```

Command: help

Figure 7, testing: the board should answer to a command with a red text.

Step 4: configure as required

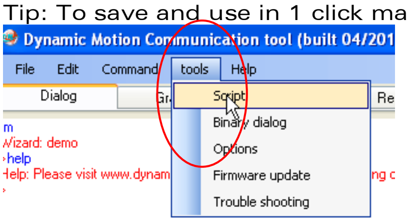


```
tm
Wizard: demo
>help
Help: Please visit www.dynamicmotion.ch and search corresponding doc under <products>
>ami1 17
<ami1=17>
```

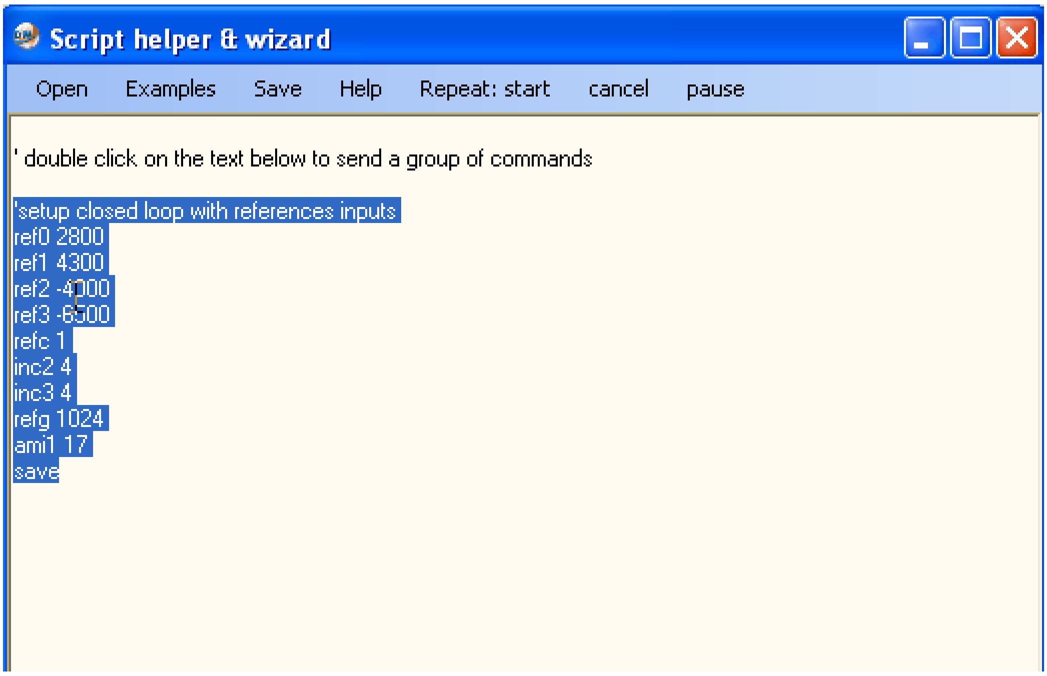
Command: ami1 17

Figure 8, example: configure a register, here "ami1" set to value 17. Use the command box.

Tip: To save and use in 1 click many registers, the software propose a tool named "script herper"



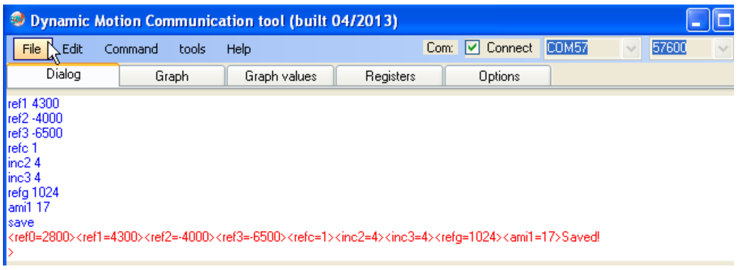
```
m
Wizard: demo
>help
Help: Please visit www.dynam
>
```



```
' double click on the text below to send a group of commands

'setup closed loop with references inputs
ref0 2800
ref1 4300
ref2 -4000
ref3 -6500
refc 1
inc2 4
inc3 4
refg 1024
ami1 17
save
```

Figure 9, script helper: paste in the script helper window your list of commands and double click on it to execute



```
ref1 4300
ref2 -4000
ref3 -6500
refc 1
inc2 4
inc3 4
refg 1024
ami1 17
save
<ref0=2800><ref1=4300><ref2=-4000><ref3=-6500><refc=1><inc2=4><inc3=4><refg=1024><ami1=17>Saved!
>
```

Figure 10, result after double click

Example of text file, can be downloaded here:
http://www.dynamicmotion.ch/download/example_config_file_for_Tinaxis-L.txt

Or can be copy-paste from below

```
' double click on the text below to send a group of commands

'setup closed loop with references inputs
ref0 2800
ref1 4300
ref2 -4000
ref3 -6500
refc 1
inc2 4
inc3 4
refg 1024
ami1 17
save
```

Adjusting the PID

The best way to adjust the PID is a practical test:

1. connect everything and mount the final load on the motor
2. set 0 to PIDF, PIDI, PIDD registers
3. search the highest value for PIDP that gives smooth result in all conditions
4. increase the PIDI value for compensation of the offset
5. PIDD can normally be omitted. Testing its best value can be done here
6. save!