



Document details

ID and title	D3.5
Description	API documentation - 1
Work package	WP3
Type	Other
Confidentiality	PUBLIC
Actual date of delivery	2019-03-31
Contractual date of delivery	2019-03-31
Project name	iHand
Project number	801945
Call	H2020 SME Instrument

Document history

Version	Date	Status	Changes	Editor
V0.1	2019-03-02	First draft	-	Martin Ewaldsson
V1.0	2019-03-28	Reviewed	-	Martin Remning Wahlstedt

Contributors

Acronym	Full Name	Person
BIO	Bioservo Technologies	Martin Ewaldsson
BIO	Bioservo Technologies	Emma Sviestins
BIO	Bioservo Technologies	Martin Remning Wahlstedt

Table of Contents

1	Introduction.....	3
1.1	Purpose of this document	3
2	General description	4
2.1	Settings	4
3	Command line interface	5
3.1	Line editing	5
3.2	Commands.....	5
3.2.1	Logging in to the system.....	5
3.3	Variables	6
3.3.1	Setting variables	6
3.3.2	Getting variables.....	7
3.3.3	Displaying variables.....	7
3.3.4	Logging variables	7
4	Controlling the iHand system externally	9
4.1	Variables for control.....	9
4.1.1	Examples of external control of the actuators.....	9
4.2	Special control states	10
4.2.1	Examples.....	10

1 Introduction

1.1 Purpose of this document

This document describes the interface (API) that is used to communicate with and control the iHand glove. This is a first version of the document. Further improvements will be made and a final version will be submitted before the end of the project.

2 General description

The interface builds upon a serial interface using ASCII communication. The interface can be used to monitor the system and to activate the system externally.

2.1 Settings

The table below shows the recommended settings.

Baudrate	921600
New line received	CR
New line transmitt	CR+LF
Terminal emulation type	VT100

3 Command line interface

3.1 Line editing

The iHand emulates a VT100 terminal. A few of the last lines of input can be retrieved using the arrow keys. The line used can be edited by moving in the line using arrow keys. The editing is inspired by readline.

- **arrow-up** previous input line
- **arrow-down** next input line
- **arrow-left** move left in the line
- **arrow-right** move right in the line
- **<ctrl-l>** redraws the line and put cursor at end
- **<ctrl-d>** clears the screen and gets the system out of any escaped state.
- Use **tab** to complete the line
- Any line of input cannot be longer than 80 characters

3.2 Commands

Command	Short	Purpose
reset	Ctrl-D	Resets the terminal.
help	-	Prints a list of all available commands
help <command>	-	Prints help about the command specified. For example, help set prints information about the command set.
set	s	Used to set variable values
get	g	Used to get variable values
display	d	Starts the display
log	-	Logs variables on display to the terminal. Stops at key press
logging	-	Logs variables on display to the terminal. Stops at Ctrl-D
login		Used to log in to the iHand system

3.2.1 Logging in to the system

The iHand system has several access levels which a user can log in to using the command **login**.

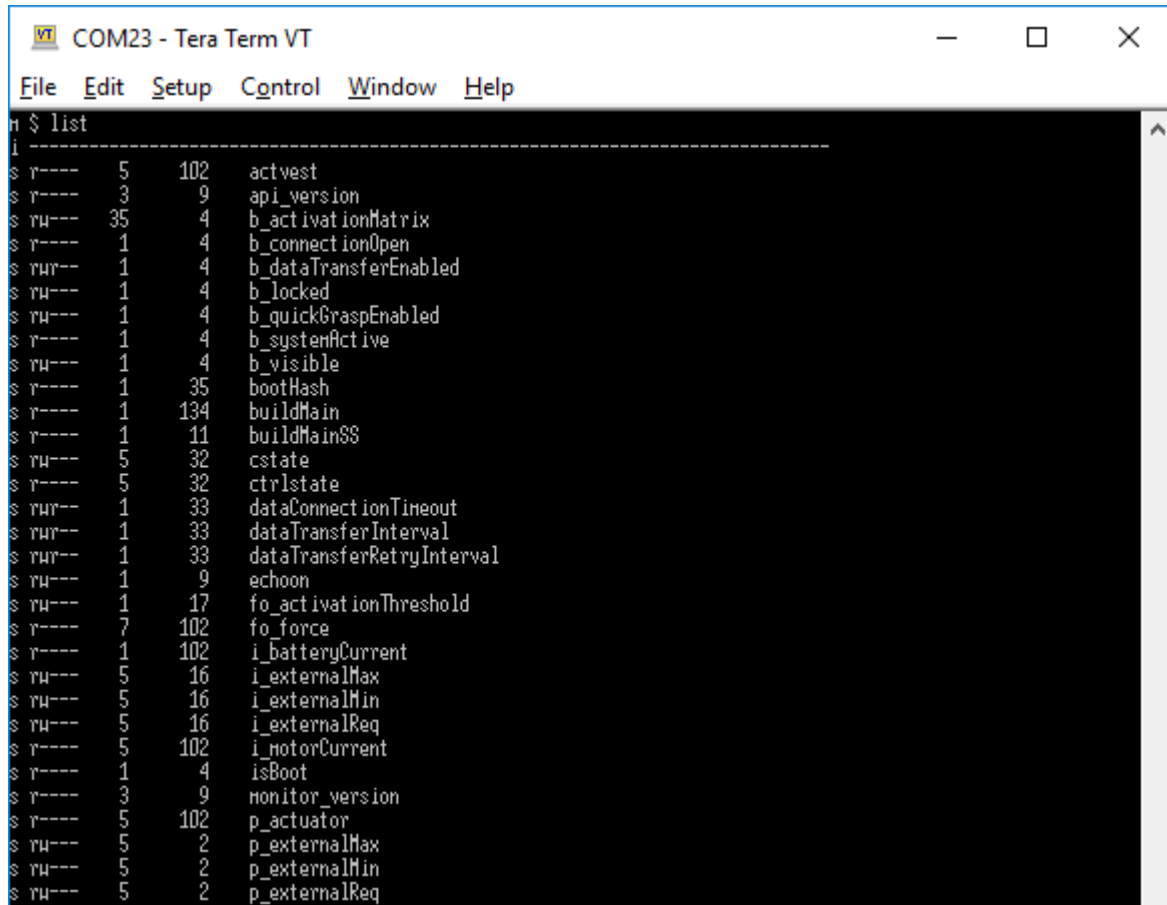
Level name	Password	Comment
all		Lowest access level. Logged in at startup.
api	api	Access to all API functionality
debug	<confidential>	Access to internal debug functionality
highest	<confidential>	Access to functionality of highest level

Example to log in to api level:

- **\$login api**
- **\$api**

3.3 Variables

The iHand system has many internal variables. Some of them are read-only and some of them are read-write. The command list is used to print a list of all available variables.



```

COM23 - Tera Term VT
File Edit Setup Control Window Help
M $ list
i
-----
$ r---- 5 102 actvest
$ r---- 3 9 api_version
$ rw--- 35 4 b_activationMatrix
$ r---- 1 4 b_connectionOpen
$ rwr-- 1 4 b_dataTransferEnabled
$ rw--- 1 4 b_locked
$ rw--- 1 4 b_quickGraspEnabled
$ r---- 1 4 b_systemActive
$ rw--- 1 4 b_visible
$ r---- 1 35 bootHash
$ r---- 1 134 buildMain
$ r---- 1 11 buildMainSS
$ rw--- 5 32 cstate
$ r---- 5 32 ctrlstate
$ rwr-- 1 33 dataConnectionTimeout
$ rwr-- 1 33 dataTransferInterval
$ rwr-- 1 33 dataTransferRetryInterval
$ rw--- 1 9 echoon
$ rw--- 1 17 fo_activationThreshold
$ r---- 7 102 fo_force
$ r---- 1 102 i_batteryCurrent
$ rw--- 5 16 i_externalMax
$ rw--- 5 16 i_externalMin
$ rw--- 5 16 i_externalReq
$ r---- 5 102 i_motorCurrent
$ r---- 1 4 isBoot
$ r---- 3 9 monitor_version
$ r---- 5 102 p_actuator
$ rw--- 5 2 p_externalMax
$ rw--- 5 2 p_externalMin
$ rw--- 5 2 p_externalReq

```

Figur 1 Example of result of the command list

3.3.1 Setting variables

Some variables available are read-write variables. These variables can be set using the command **set** or **s** followed by the variable name and the array values separated by spaces.

Example:

- **\$set tn_externalReq 5000 5000**

This line sets the first two elements of the variable `tn_externalRequest` to 5000

Example:

- **\$s tn_externalReq * 10000**

This line sets all elements in the array to 10000 mN by using the `*`-operator.

Example:

- **\$s tn_externalReq [2] 2500**

This line sets the third element in the `tn_externalReq` array variable to 2500 mN. Array/vector variables are accessed with index 0 as the first element index.

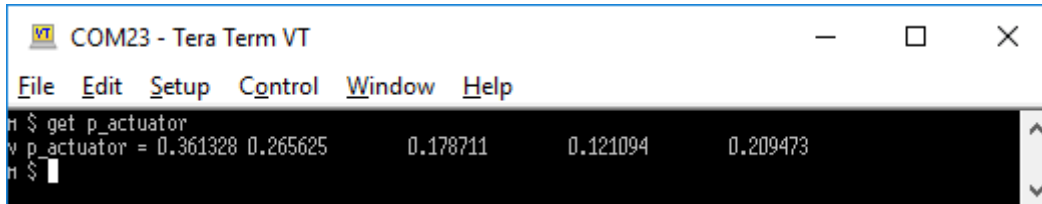
3.3.2 Getting variables

All variables can be read using the **get** or **g** command followed by the variable name.

Example:

- **\$get p_actuator**

This can result in the response **v p_actuator 2.00 15.00 2.00** which means that the three actuators are positioned at 2, 15 and 2 mm respectively.



```
COM23 - Tera Term VT
File Edit Setup Control Window Help
M $ get p_actuator
v p_actuator = 0.361328 0.265625      0.178711      0.121094      0.209473
M $
```

Figur 2 Example of usage of the command `get`

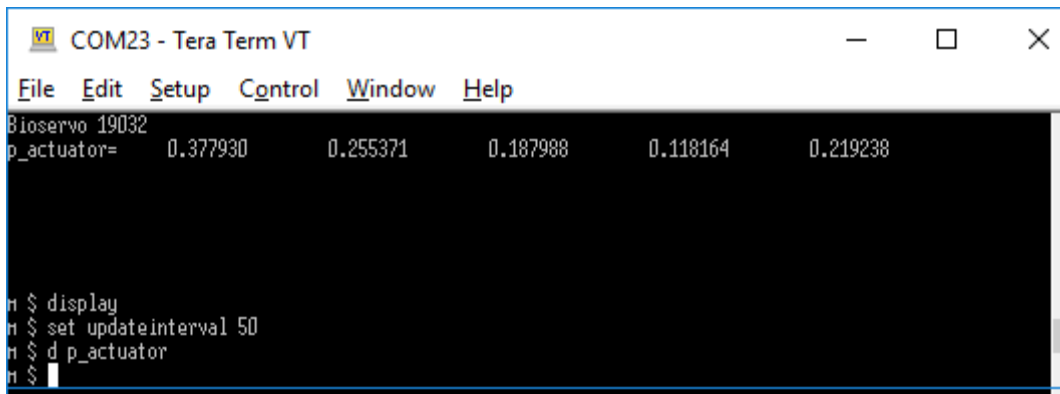
3.3.3 Displaying variables

To get real time update of selected variables the command **display** or **d** can be used. When display is active the system time is automatically displayed and it is the time in milliseconds since start up.

Example:

- **\$display**
- **\$set updateinterval 50**
- **\$d p_actuator**

This set of commands will display the variable `p_actuator` with an updateinterval of 50ms.



```
COM23 - Tera Term VT
File Edit Setup Control Window Help
Bioservo 19032
p_actuator= 0.377930      0.255371      0.187988      0.118164      0.219238
M $ display
M $ set updateinterval 50
M $ d p_actuator
M $
```

Figur 3 Example of using the command `display`

3.3.4 Logging variables

With the command **log** variables that are on display are dumped on the terminal, line after line. This can be used to log and store data for analysis in an external tool. The logging is terminated by any character received.

Example:

- **\$display**
- **\$d p_actuator**
- **\$log**

```
COM23 - Tera Term VT
File Edit Setup Control Window Help
h $
??Time p_actuator
l 119216 0.378418 0.255859 0.187988 0.118164 0.219238
l 119266 0.378418 0.255859 0.187988 0.118164 0.219238
l 119316 0.378418 0.255859 0.187988 0.118164 0.219238
l 119366 0.378418 0.255859 0.187988 0.118164 0.219238
l 119416 0.378418 0.255859 0.187988 0.118164 0.219238
l 119466 0.378418 0.255859 0.187988 0.118164 0.219238
l 119516 0.378418 0.255859 0.187988 0.118164 0.219238
l 119566 0.378418 0.255859 0.187988 0.118164 0.219238
l 119616 0.378418 0.255859 0.187988 0.118164 0.219238
l 119666 0.378418 0.255859 0.187988 0.118164 0.219238
l 119716 0.378418 0.255859 0.187988 0.118164 0.219238
l 119766 0.378418 0.255859 0.187988 0.118164 0.219238
l 119817 0.378418 0.255859 0.187988 0.118164 0.219238
l 119867 0.378418 0.255859 0.187988 0.118164 0.219238
l 119917 0.378418 0.255859 0.187988 0.118164 0.219238
l 119967 0.378418 0.255859 0.187988 0.118164 0.219238
l 120017 0.378418 0.255859 0.187988 0.118164 0.219238
h $
```

Figur 4 Example of using the command log

4 Controlling the iHand system externally

There are three main types of variables used to control the motors of the system externally:

- Requests – sets desired level
- Max limiters – sets upper limit
- Min limiters – sets lower limit

Requests never override max limiters. If several max limiters are set, the one resulting in the lowest motor voltage is in control. Analogously the inverse for the min limiters.

4.1 Variables for control

The system can be activated and limited externally using the variables in the table below.

Variable	Unit	Function	Range
u_externalReq	mV	Motor voltage request	[-20000, 20000]
u_externalMax	mV	Motor voltage upper limit	[-20000, 20000]
u_externalMin	mV	Motor voltage lower limit	[-20000, 20000]
i_externalReq	mA	Motor current request	[-10000, 10000]
i_externalMax	mA	Motor current upper limit	[-10000, 10000]
i_externalMin	mA	Motor current lower limit	[-10000, 10000]
tq_externalReq	μNm	Motor torque request	[-100000, 100000]
tq_externalMax	μNm	Motor torque upper limit	[-100000, 100000]
tq_externalMin	μNm	Motor torque lower limit	[-100000, 100000]
p_externalReq	mm	Actuator position request	[-10, 200]
p_externalMax	mm	Actuator position upper limit	[-10, 200]
p_externalMin	mm	Actuator position lower limit	[-10, 200]
v_externalReq	mm/s	Actuator position request	[-300, 300]
v_externalMax	mm/s	Actuator position upper limit	[-300, 300]
v_externalMin	mm/s	Actuator position lower limit	[-300, 300]
tn_externalReq	mN	Tension request	[-100000, 100000]
tn_externalMax	mN	Tension upper limit	[-100000, 100000]
tn_externalMin	mN	Tension lower limit	[-100000, 100000]

Even though variables can be set to any value within the range it is not guaranteed that the system can deliver that value, it might be limited by other limitations, physical or programmed in firmware. There are several safety max and min limiters that are always active that cannot be altered through the API. No settings done through the API can harm the system in any way. The API cannot remove limitations, only further limit the system.

4.1.1 Examples of external control of the actuators

Example:

- **\$set p_externalMax 55**
- **\$set v_externalReq 150**

This set of commands will make the actuator travel at 150 mm/s up to the position 55 mm where it stops.

Example:

- **\$set i_externalMax 500**
- **\$set v_externalMax 100**
- **\$set tn_externalRequest 20000**

These commands will result in that the system tries to apply 20000 mN of tension force. The speed never exceeds 100 mm/s and the current never exceeds 500 mA. If 500 mA is not enough to reach 20000 mN, the requested tension force will not be reached.

4.2 Special control states

In normal operation the iHand is activated by pressing the finger sensors. When external commands are entered to control the system, the normal operation functionality can come in conflict with the external requests. To disable the normal operation the system can be told to enter special control states using the variable **cstate**. The variable **cstate** is a vector variable with one element for each finger. That means that fingers can be set to be in different states if desired.

Value	State
0	No request (system is disabled)
1	Motor voltage request
2	Speed request
3	Tension request
4	Normal operation (default state)

The external max and min limits can always be used in any of the special control states.

4.2.1 Examples

Example:

- **\$set cstate * 1**
- **\$set i_externalMax * 1000**
- **\$set u_externalReq * 15000**

This set of commands puts all actuators into motor voltage request state (**cstate** = 1) and an upper limit of the motor current to 1 A (1000 mA). The voltage is then set to 15 V which is applied but is limited if the motor current approaches 1 A. Since the system is voltage request state nothing will happen when the finger sensors are pressed.

Example:

- **\$set cstate * 2**
- **\$set tn_externalMax * 20000**
- **\$set v_externalRequest * 100**

The system will activate, and the actuators start moving in 100 mm/s and stop when the tension is approaching 20000 mN.

Example:

- **\$set cstate * 3**

- **\$set p_externalReq * 100**

In this case nothing happens since the system is in tension request state and no tension request is provided.