

Comau Robotics Instruction Handbook



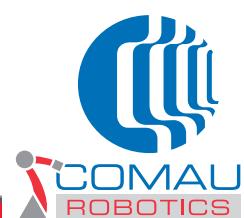
comau.com/robotics

C5G Controller Unit

CONTROL UNIT USE System Software Rel. 2.30

Access to system (power-on, shutdown, login, logout), User Interfaces, TP5 Teach Pendant, System Commands, WinC5G, Start and Stop Cycle, I/O configuration, Software Options.

CR00758046_en-00/2014.04



The information contained in this manual is the property of COMAU S.p.A.

Reproduction of text and illustrations is not permitted without prior written approval by COMAU S.p.A.

COMAU S.p.A. reserves the right to alter product specifications at any time without notice or obligation.

Copyright © 2008-2013 by COMAU - Date of publication 04/2014

SUMMARY

PREFACE19
Symbols used in the manual	19
Reference documents	20
Modification History	21
1. GENERAL SAFETY PRECAUTIONS25
Responsibilities	25
Safety Precautions	26
Purpose	26
Definitions	26
Applicability	27
Operating Modes	28
Performance	34
2. USER INTERFACE35
Introduction	35
Teach Pendant (TP5).	35
Front view	36
Rear view	37
Interface on Personal Computer (WinC5G)	38
3. SYSTEM POWER ON AND SHUT DOWN39
Foreword	39
Main switch	39
System power-on	40
STARTUP Procedure	42
System shut down	43
4. ACCESS TO THE CONTROL (LOGIN/LOGOUT)45
Introduction	45
Login	45
User profile	46
Administrator	47
Default	47
Programmer	48
Maintenance	48

Service	48
Technology	48
Summary table of access rights	48
Logout	50
Automatic Timed Logout	51
5. TP5 TEACH PENDANT	52
Introduction	52
Modal selector switch	53
Keys, Pushbuttons and LEDs	54
Keys	54
Blue keys	55
Special keys	55
General purpose keys	57
Navigation general keys	57
Black keys	58
BACK	59
COORD	59
ARM	60
% + e -	60
JOG keys	61
JPAD	61
Other colours keys	62
RESET (white)	63
HOLD (yellow)	63
START (green)	63
Keys to restart the Teach Pendant	63
Pushbuttons and LEDs	64
Pushbuttons	64
Stop pushbutton	64
Enabling Device	64
LEDs	65
Green LED	66
Red LED	66
USB port	66
Display	67
Display areas	67
Status bar	67
Right Menu	70
Functional keys menu	71
Pages Area	72
User Interface Pages	72
General information	73
Field types	73
Numeric field	74
Textbox	74
Combobox	74
Combobutton	75
Checkbox	75

Checkgroup	75
Label.....	75
List	76
Cell of a table	76
TreeGrid	77
Keyboards	77
Numeric keyboard	78
Alphanumeric keyboard	79
PDL2 keyboard.....	80
Statement	81
Variables	81
Values	82
History.....	82
Connection Page	83
Dialogue and information window	83
Start Page	86
Basic information	86
Controller name	86
Arms name and type	87
Software version.....	87
Network address.....	87
Memory space	88
Current Login	88
Language selection	88
Date and time	89
Commands.....	90
Login and Logout	90
Login	90
Logout.....	91
Restart	91
Cold	92
Shutdown	93
MENU.....	94
QUICK Page	95
Overview	95
Viewing.....	95
Creating/Modifying	95
SYS_CALL	96
PDL2.....	96
MENU Page.....	98
Overview	98
Desktop	98
Active	99
Motion Page.....	101
Basic	101
Arm\SyncArm	101
COORD	102
Tool, Frame, Base	103
Tool and Frame	103
Base	104
Current arm position	104

Payload	104
Coop (optional feature)	105
Coop Status	105
COOP	105
Advanced	106
Incremental Jog mode	106
J-Pad	107
Status	108
Collision Detection (optional feature)	109
Palletizing (optional feature)	109
Override	110
Conveyor Tracking (optional feature)	110
Arm Override Page	113
Arm Settings Page	114
Alarm Page	115
Active	115
History	116
Prog Page	118
Memory	119
New program	120
Load	120
Load (with option)	120
Activate	120
with another .VAR (As)	120
.Var only (Var)	121
Another .Var only (Var As)	121
Convert conflicts	121
Dependencies	121
Save	121
Save data	121
Save code	121
Save variables (As)	121
Erase from execution memory	122
Edit	122
Select All	122
Deselect All	123
Invert Sel	123
Program	123
Open [ENTER]	124
Activate	125
Deactivate	125
Reset	125
Pause	126
Unpause	126
Step	126
View	127
Filters	127
Level	128
Details	128
Show variables	128
Callchain	130

Properties	130
IDE Page	132
Opening the IDE page	132
Description of the video screen	133
IDE status lines.....	133
Editor area	134
Colours	134
Cursors	134
Available functions description.....	136
Program	137
Save (cod, var)	137
Save all mod progs (cod,var)	137
Reset	137
Abort	137
Deactivate	137
Delete unused vars	138
Import	138
Step	139
Immediate execution	141
Edit	142
Select	142
Paste	143
Delete line	143
Undelete line	143
Goto	143
Line.....	143
Set execution cursor	144
Insert/remove break point	144
Bypass	144
Comment/Uncomment	144
Insert.....	145
REC	146
REC setup	146
MOD	147
Details.....	148
View variables in line	148
View variables in program	149
Close	149
Display Page	150
Elements	150
IO	150
Device.....	151
Device	153
Commands to handle Devices	155
Device (F1)	156
Remote Inp. (F2).....	156
Properties (F5)	157
Ports	157
Display	159
Set (SI and SO)	159
On.....	160
Off.....	160

Force (SIF... SOF...)	160
On	161
Off	161
Current	161
Unforce (SIU... SOU...)	161
View (SIV... SOV...)	161
Zoom in	162
Zoom out	162
Vertical	163
Horizontal	163
Format	164
Window	164
Close	165
Maximize	165
Restore	165
Next	165
Appl Page	166
Files Page	167
File (F1)	167
Open	168
Extract (FUCE)	169
Send to (FC)	169
New	169
Delete (FD)	169
Delete permanent (FD/P)	170
Rename (FR)	170
Attribute (FUA)	170
Properties	170
Select (F2)	170
Undo	171
Cut	171
Copy	171
Paste	171
Select all	171
Deselect all	172
Invert sel	172
Sel prompt	172
View (F3) (FV)	172
Small icons	172
Big icons	173
Details	173
Refresh	173
Sort by	173
Utility (F4)	173
Translate (FT)	174
Backup (FUB)	174
Backup of Saveset (FUB/S)	176
Restore (FUR)	177
Restore of Saveset (FUR/S)	179
Device Change	180
Device Properties	181
Recycle bin	181

Data Page	182
General information	182
Table.	183
Open	183
Close.	183
Apply	183
Save	184
Reload	184
History.	184
Modify.	184
Insert.	184
Copy	185
CellCopy	185
Paste.	185
Undo All	185
Set Active	185
Selection.	185
Line	185
Invert.	185
Select all	185
Deselect all	185
AutoApply	186
System tables.	186
BASE	188
TOOL	188
FRAME.	189
Accessing system tables from within a PDL2 program	189
Routine for Tool and Frame setting for a specified Arm	190
Routine for Base setting for a specified Arm	191
Routines to get/set/save the Tool, Frame and Base value	191
Using the routines - examples	195
Error codes	196
Application tables.	196
Tables created by the user.	197
Setup page.	198
Config. Function.	199
Properties.	199
Arm	200
Calib	200
Turn Set (CAT)	201
Calib (CAC).	202
Save (CARS)	204
Load (CARL)	204
Modify (F5)	205
Close (F6)	206
Mounting.	206
StrokeEnd.	207
Press	207
AUX_AXES	208
Positioners	208
SLIDE	213
iPortal	214
Gantry with 3 linear axes	214

Gantry with 2 linear axes	214
Trans-rotational column	215
Backup-Restore	215
Backup	216
Restore	218
Device	220
Controller (CCS)	221
Conveyor (optional feature)	222
Tracking Table editing procedure	224
Customer	226
Date-Time	227
FB_Util	230
Hand	230
One	232
Dual	232
Pulse	233
Step	234
Install (FUI)	234
IO_Clone	236
IO_Cnfg	236
IO_Map	236
IRegions (optional feature)	237
Interference Region Editing Procedure	239
Tips about special shape regions	245
Plane	245
Joint	247
Login	248
User	248
User	249
Startup (CCLS)	249
Automatic Logout	250
Network	251
Network	251
USB to Ethernet	251
Ethernet Annex	252
ReloadSw	252
Upgrading of System and TP	254
Upgrading of robot configuration	257
System configuration completion (.C5G)	259
Upgrading of TP only	261
Full loading of System and TP	262
ToolFrame	268
MGDSetup (optional feature)	268
Service Page	270
Properties	271
Arm	271
PageBattery	272
Display Var (MVV)	273
Add (F1)	274
Delete (F2)	279
Load... (F3)	279
Save... (F4)	280
Format (F5)	280

Close (F6)	281
PageExecute (E)	282
Moni	283
MULTI-ARM	286
PageSysInfo (SCV)	286
Trace	287
 Customizations on the TP	290
Additional User Pages	290
User table creation from DATA environment	291
Properties	292
Table (global) Properties	292
Field Properties	296
Field of a POSITION Properties	298
Variable <type>_signal	299
Variable <typename>validate_name_err	300
Sample program for a variable name validation	301
Example program for a table creation	302
Handling TP right menu	304
XML Tag Parameters	305
Softkey Pressure and Release events	306
Example of XML configuration file	307
Example of right menu configuration	308
Customizing the PDL2 Statements insertion, in IDE environment	310
Adding User Instructions to the PDL2 menu	310
Adding a Statement	310
Adding a Routine	311
Adding a Built-in Routine	312
Creating a “virtual” Numeric Keypad to insert User Instructions	313
Customizing I/O ports on the TP	315
Creating user pages in SETUP and SERVICE environment	316
Adding Projects/Applications, installable from within SETUP Page	318
 6. SYSTEM COMMANDS	320
Introduction	320
Commands sent from menu on TP (TPINT page) or from WinC5G (Terminal window)	321
Command selection	321
Typing-in the characters	322
Call-up of recently used parameters (History)	322
Program commands sent via SYS_CALL	322
Option /4 to view in the 40 columns	323
Wildcard use	323
Command options	323
Viewing and NOPAGE option	323
Access to the control (login)	324
Directories	324
Description of commands	324
CONFIGURE branch	331

Arm menu	331
CONFIGURE ARM CALIBRATE (CAC)	331
CONFIGURE ARM RETENTIVE MEMORY LOAD (CARL)	331
CONFIGURE ARM RETENTIVE MEMORY SAVE (CARS)	332
CONFIGURE ARM TURN_SET (CAT)	332
CONFIGURE ARM VIEW_CAL (CAV)	332
Cntrler Login menu	333
CONFIGURE CONTROLLER LOGIN ADD (CCLA)	333
CONFIGURE CONTROLLER LOGIN DELETE (CCLD)	333
CONFIGURE CONTROLLER LOGIN GROUP (CCLG)	333
CONFIGURE CONTROLLER LOGIN MODIFY (CCLM)	334
CONFIGURE CONTROLLER LOGIN STARTUP (CCLS)	334
CONFIGURE CONTROLLER LOGIN VIEW (CCLV)	335
Cntrler Restart menu	335
CONFIGURE CONTROLLER RESTART COLD (CCRC)	335
CONFIGURE CONTROLLER RESTART UPGRADE (CCRU)	335
CONFIGURE CONTROLLER RESTART SHUTDOWN (CCRS)	337
CONFIGURE CONTROLLER STARTUP (CCS)	337
CONFIGURE CONTROLLER TIME (CCT)	337
CONFIGURE CONTROLLER VIEW (CCV)	338
Load menu	338
CONFIGURE LOAD ALL (CLA)	338
CONFIGURE LOAD CATEGORY	338
Save menu	340
CONFIGURE SAVE ALL (CSA)	340
CONFIGURE SAVE CATEGORY	341
DISPLAY branch	342
Arm menu	343
DISPLAY ARM CURRENTS (DAC)	343
DISPLAY ARM DATA (DAD)	343
DISPLAY ARM FOLLOWING (DAF)	343
DISPLAY ARM JOINT (DAJ)	343
DISPLAY ARM POSITION (DAP)	343
DISPLAY ARM REVOLUTIONS (DAR)	343
DISPLAY ARM STATUS (DAS)	344
DISPLAY ARM TEMPERATURE (DAT)	344
Close menu	344
DISPLAY CLOSE ARM (DCA)	344
DISPLAY CLOSE INPUT (DCI)	344
DISPLAY CLOSE OUTPUT (DCO)	344
DISPLAY CLOSE PROGRAM (DCP)	344
DISPLAY CLOSE RESPLC (DCR)	344
DISPLAY CLOSE SELECT (DCS)	344
DISPLAY CLOSE TOTAL (DCT)	345
DISPLAY CLOSE VARS (DCV)	345
Input menu	345
DISPLAY INPUT AIN (DIA)	345
DISPLAY INPUT DIN (DID)	345
DISPLAY INPUT FMI (DIF)	345
DISPLAY INPUT GI (DIG)	346
DISPLAY INPUT IN (DII)	346
DISPLAY INPUT SYSTEM (DIS)	346
Output menu	346

DISPLAY OUTPUT AOUT (DOA)	346
DISPLAY OUTPUT DOUT (DOD)	347
DISPLAY OUTPUT FMO (DOF)	347
DISPLAY OUTPUT GO (DOG)	347
DISPLAY OUTPUT OUT (DOO)	347
DISPLAY OUTPUT SYSTEM (DOS)	347
DISPLAY PROGRAM (DP)	347
DISPLAY RESPLC (DR)	348
Vars menu	348
DISPLAY VARS BIT (DVB)	348
DISPLAY VARS WORD (DVW)	348
EXECUTE command (E)	349
FILER branch	350
FILER COPY (FC)	350
FILER DELETE (FD)	351
FILER EDIT	351
FILER PRINT (FP)	351
FILER RENAME (FR)	352
FILER TRANSLATE (FT)	353
UTILITY ATTRIBUTE menu	354
FILER UTILITY ATTRIBUTE ARCHIVE (FUAA)	354
FILER UTILITY ATTRIBUTE HIDDEN (FUAH)	354
FILER UTILITY ATTRIBUTE READONLY (FUAR)	354
FILER UTILITY ATTRIBUTE SYSTEM (FUAS)	354
Backup and Restore commands	354
FILER UTILITY BACKUP (FUB)	356
FILER UTILITY RESTORE (FUR)	358
Compressed files management commands	359
FILER UTILITY COMPRESS DELETE (FUCD)	359
FILER UTILITY COMPRESS EXTRACT (FUCE)	359
FILER UTILITY COMPRESS MAKE (FUCM)	360
FILER UTILITY COMPRESS VIEW (FUCV)	360
Directory management commands	360
FILER UTILITY DIRECTORY CHANGE (FUDC)	360
FILER UTILITY DIRECTORY DELETE (FUDD)	361
FILER UTILITY DIRECTORY MAKE (FUDM)	361
FILER UTILITY INSTALL (FUI)	361
FILER UTILITY PROTECT (FUP)	361
FILER UTILITY SEARCH (FUS)	361
FILER VIEW (FV)	362
MEMORY branch	363
MEMORY DEBUG	363
ERASE menu	363
MEMORY ERASE ALL (MEA)	363
MEMORY ERASE PROGRAM (MEP)	364
MEMORY ERASE VARIABLE(MEV)	364
MEMORY LOAD (ML)	364
MEMORY SAVE(MS)	365
MEMORY TEACH	365
VIEW menu	366
MEMORY VIEW DATABASE (MVD)	366
MEMORY VIEW PROGRAM (MVP)	366
MEMORY VIEW SYSTEM (MVS)	367

MEMORY VIEW TREE (MVR)	367
MEMORY VIEW TYPE (MVT)	367
MEMORY VIEW VARIABLE (MVV)	368
MEMORY VIEW VP2 (MVV)	368
PROGRAM branch	369
PROGRAM ACTIVATE (PA)	369
PROGRAM DEACTIVATE (PD)	369
PROGRAM EDIT	370
PROGRAM GO (PG)	371
ResPLC menu	372
PROGRAM ResPLC ACTIVATE (PRA)	372
PROGRAM ResPLC DEACTIVATE (PRD)	372
PROGRAM ResPLC RESTORE (PRR)	372
UTILITY menu	372
PROGRAM ResPLC UTILITY PROJDIR (PRUP)	372
PROGRAM ResPLC UTILITY SAVE (PRUS)	372
PROGRAM ResPLC UTILITY UNLOAD (PRUU)	372
PROGRAM ResPLC UTILITY VIEW (PRUV)	372
STATE menu	373
PROGRAM STATE BYPASS (PSB)	373
PROGRAM STATE PAUSE (PSP)	373
PROGRAM STATE UNPAUSE (PSU)	373
TEST branch	373
Break menu	373
PROFILE menu	374
STEP menu	375
PROGRAM VIEW (PV)	376
SET branch	377
SET ARM menu	378
SET ARM DISABLE (SAD)	378
SET ARM ENABLE (SAE)	378
SET ARM GEN_OVR (SAG)	378
SET ARM NOSTROKE(SAN)	378
SET ARM SIMULATE (SAS)	379
SET ARM TP_MAIN (SAT)	379
SET ARM UNSIMULATE	379
SET CNTRLR menu	379
SET CNTRL KEY_LOCK (SCK)	379
SET CNTRL LANGUAGE (SCL)	379
SET CONTROLLER VIEW (SCV)	380
SET CONTROLLER WIN_CLEAR (SCW)	380
SET INPUT menu	380
SET INPUT FORCE branch	381
SET INPUT REMOTE branch	381
SET INPUT UNFORCE branch	382
SET INPUT VIEW branch	382
SET LOGIN (SL)	383
SET OUTPUT menu	383
SET OUTPUT FORCE branch	383
SET OUTPUT UNFORCE branch	384
SET OUTPUT VIEW branch	385
UTILITY branch	385
UTILITY APPLICATION (UA)	385

UTILITY COMMUNICN menu	386
UTILITY COMMUNICN DISMOUNT (UCD)	386
UTILITY COMMUNICN MOUNT menu	386
UTILITY COMMUNICN PORT_CHAR (UCP)	387
UTILITY COMMUNICATION SET_DEF (UCS)	387
UTILITY COMMUNICATION VIEW (UCV)	387
UTILITY LOG branch	387
UTILITY LOG ACTION (ULA)	387
UTILITY LOG ERROR (ULE)	388
UTILITY LOG LATCH ACKNOWLEDGE (ULLA)	388
UTILITY LOG LATCH VIEW (ULLV)	388
Types of files available in the System	388
7. WINC5G PROGRAM - INTERFACE TO C5G ON PERSONAL COMPUTER	390
Foreword	390
Overview	390
WinC5G Activation	391
Connection to the Robot Control Unit	391
Ethernet point to point connection	391
Ethernet connection to the local network	394
Remote connection via Internet	394
Network configuration	395
VPN configuration in Windows XP environment	396
Remote PC configuration (server VPN)	396
Local PC configuration (client VPN)	399
Connection procedure	403
Remote PC side	403
Local PC side	403
FTP Server mode connection	404
Remote connection by Proxy	405
Chat Service	405
WinC5G configuration in Proxy mode	406
WinC5G configuration in Remote mode	406
Auto Config	407
Using the Chat Window	408
Disabling the Terminal	408
Connection by serial line	409
User interface	409
Tools panel	410
The Terminal	410
File Translation	411
Cause/Remedy	413
Errors Display and Actions	414
Files Manipulation	415
Axis check	417
Tool and Frame update	417
Linear Shift	418
Linear Rotation	418
Mirror	420

Calibration	422
Sysdata adjust	422
Turn difference	422
Dialogue box for File Manipulation settings	423
Viewing and editing of .UDB file (optional feature)	424
Directories Panel	425
Files Panel	426
Output Panel	427
WinC5G operating parameters set-up	427
Commands Menu	429
Files Menu	429
Edit Menu	430
View Menu	431
Manipulation Menu	432
Help Menu	432
Tools Menu	433
How to obtain a new licence for WinC5G	433
Most common problems	433
8. FILE MANAGEMENT BETWEEN PC AND CONTROL UNIT	436
View files on PC	436
Delete files on PC	436
File transfer from PC to Controller	436
File transfer from Controller to PC	437
Automatic file transfer from Controller to PC	437
Automatic file transfer from PC to Controller	437
9. SYSTEM OPERATING MODES AND STATES	439
Foreword	439
System operating modes	439
System states	440
HOLD status	441
AUTO status	442
PROGR status	442
ALARM status	442
Stand-by function	443
10. START AND STOP CYCLE	444
11. I/O CONFIGURATION PROGRAMS	445
IO_CNFG Program - I/O modules configuration	446

Activation of IO_CNFG program	446
Network (F1)	448
Powerlink	448
Add - (F1)	450
Remove (F2)	450
Fieldbus (F3)	450
X20 (F5)	458
Virtual	465
Add (F1)	465
Edit (F2)	466
Delete (F3)	466
COM0: serial Port	467
Serial (F2)	467
Acopos (F3)	468
Annex (F4)	469
Powerlink	469
Ethernet	470
Save (F5)	470
Exit (F6)	471
 FB_util Program - Fieldbus modules utility	472
Activation of FB_util	472
Profinet (F1)	473
Operational procedures	476
First installation procedure	476
Device replacement procedure	477
Profibus (F2)	479
DeviceNET (F3)	479
Quit (F6)	480
 Project of a Master Fieldbus Network	481
Creating a new project	481
Phase one - SYCON.net program	481
Phase two - WinC5G program	490
Differences in creating a Master DeviceNet network	491
Modifying a project already saved onto the Controller	495
Phase one - WinC5G	495
Phase two - SYCON.net	496
 IO_MAP Program - I/O ports mapping	498
Activation of IO_MAP program	498
Device (F1)	499
Modify (F1) - Device	500
Map or Modify (F1) - I/O points	504
Clear (F2) - I/O points	505
Direction (F3) - I/O points	505
Prev page and Next page (F4 and F5) - I/O points	505
Clear (F2) - Device	506
Move (F3) - Device	506
Prev page (F4), Next page (F5)	508
USER Devices	508
Port list (F2)	510
Category (F2) - Port list	511
Direction (F3) - Port list	511
Prev page (F4) and Next page (F5) - Port list	511

Profile (F3)	511
System Profile	512
System Profile - Comau	513
System Profile - Custom	517
Application Profile.	520
Application Profile - Comau	521
Application Profile - Custom	525
Link (F4)	527
Add (F1) - Link	528
Delete (F2) - Link	529
Prev page (F4) and Next page (F5) - Link	529
Save (F5)	529
Exit (F6)	529
IO_CLONE Program - I/O Configuration Export/Import	530
Activation of IO_CLONE	530
Export (F2)	532
Import (F1)	533
Save (F5)	535
Quit (F6)	535
12. APPENDIX - SOFTWARE OPTIONS	536

PREFACE

- Symbols used in the manual
- Reference documents
- Modification History

Symbols used in the manual

The symbols for **WARNING**, **CAUTION** and **NOTES** are indicated below together with their significance.



This symbol indicates operating procedures, technical information and precautions that if ignored and/or are not performed correctly could cause injuries.



This symbol indicates operating procedures, technical information and precautions that if ignored and/or are not performed correctly could cause damage to the equipment.



This symbol indicates operating procedures, technical information and precautions that it are important to highlight.

Reference documents

This document refers to the **C5G Control Unit**.

The complete set of manuals for the **C5G** consists of:

Comau	C5G Control Unit	<ul style="list-style-type: none">– Technical Specifications– Transport and installation– Maintenance– Control Unit Use.
-------	-------------------------	---

These manuals are to be integrated with the following documents:

Comau	Robot	<ul style="list-style-type: none">– Technical Specifications– Transport and installation– Maintenance
	Programming	<ul style="list-style-type: none">– PDL2 Programming Language– VP2 - Visual PDL2– Motion programming
	Applications	<ul style="list-style-type: none">– According to the required type of application.

Modification History

1. GENERAL SAFETY PRECAUTIONS



It deals with a general specification that apply to the whole Robot System. Due to its significance, this document is referred to unreservedly in any system instructions handbook.

In this chapter are shown the following topics:

- [Responsibilities](#)
 - [Safety precautions](#).
-

1.1 Responsibilities

- The system integrator is responsible for ensuring that the [Robotic system \(Robot and Control Unit\)](#) are installed and handled in accordance with the Safety Standards in force in the country where the installation takes place. The application and use of the protection and safety devices necessary, the issuing of declarations of conformity and any EC markings of the system are the responsibility of the Integrator.
- COMAU Robotics shall in no way be held liable for any accidents caused by incorrect or improper use of the [Robotic system \(Robot and Control Unit\)](#), by tampering with circuits, components or software, or the use of spare parts that are not included in the spare parts list.
- The application of these Safety Precautions is the responsibility of the persons assigned to direct / supervise the activities indicated in the [Applicability](#) section are to make sure that the [Authorised Personnel](#) is aware of and scrupulously follow the precautions contained in this document as well as the Safety Standards in addition to the Safety Standards applicable to [Robotic system \(Robot and Control Unit\)](#) in force in the Country where the system is installed.
- The non-observance of the Safety Standards could cause to the operators permanent injuries or death and can damage the [Robotic system \(Robot and Control Unit\)](#).



The installation shall be made by qualified installation Personnel and should conform to all National and Local standards.

1.1.1 Safety Fundamental Requirements applied and respected

The robotic system is composed of C5G Control Unit and Robot series SMART 5 considers as applied and respected the following Safety Fundamental Requirements, Annex 1 of Directive on Machinery 2006/42/CE: 1.1.3 – 1.1.5 – 1.2.1 – 1.2.2 – 1.2.3 – 1.2.4.3 – 1.2.5 – 1.2.6 – 1.3.2 – 1.3.4 – 1.3.8.1 – 1.5.1 – 1.5.2 – 1.5.4 – 1.5.6 – 1.5.8 – 1.5.9 – 1.5.10 – 1.5.11 – 1.5.13 – 1.6.3 – 1.6.4 – 1.6.5 – 1.7.1 – 1.7.1.1 – 1.7.2 – 1.7.4. In case that is provided only the Robot SMART 5 series, are considered applicable the requirements: 1.1.3 – 1.1.5 – 1.3.2 – 1.3.4 – 1.3.8.1 – 1.5.1 – 1.5.2 – 1.5.4 – 1.5.6 – 1.5.8 – 1.5.9 – 1.5.10 – 1.5.11 – 1.5.13 – 1.6.4 – 1.6.5 – 1.7.1 – 1.7.1.1 – 1.7.2 – 1.7.4

1.2 Safety precautions

1.2.1 Purpose

These safety precautions are aimed to define the behaviour and rules to be observed when performing the activities listed in the [Applicability](#) section.

1.2.2 Definitions

Robotic system (Robot and Control Unit)

Robotic system is the workable unit composed of: Robot, Control Unit, Teach Pendant and other options.

Protected Area

The protected area is the zone confined by the protection barriers and to be used for the installation and operation of the robot.

Authorised Personnel

Authorised personnel defines the group of persons who have been trained and assigned to carry out the activities listed in the [Applicability](#) section

Staff in charge

The staff in charge to direct or supervise the activities of the workers referred to in the paragraph above.

Installation and Putting into Service

The installation is intended as the mechanical, electrical and software integration of the Robot and Control System in any environment that requires controlled movement of robot axes, in compliance with the safety requirements of the country where the system is installed.

Operation in Programming Mode

Operating mode under the control of the operator, that excludes automatic operation and allows the following activities: manual handling of robot axes and programming of work cycles at low speed, programmed cycle testing at low speed and, when allowed, at the working speed.

Auto / Remote Automatic Mode

Operating mode in which the robot autonomously executes the programmed cycle at the work speed, with the operators outside the protected area, with the protection barriers closed and the safety circuit activated, with local (located outside the protected area) or remote start/stop.

Maintenance and Repairs

Maintenance and repairs are activities that involve periodical checking and / or replacement (mechanical, electrical, software) of Robot and Control System parts or components, and trouble shooting, that terminates when the Robot and Control System has been reset to its original project functional condition.

Putting Out of Service and Dismantling

Putting out of service defines the activities involved in the mechanical and electrical removal of the Robot and Control System from a production unit or from an environment in which it was under study.

Dismantling consists of the demolition and dismantling of the components that make up the Robot and Control System.

Integrator

The integrator is the professional expert responsible for the installation and putting into service of the Robot and Control System.

Misuse

Misuse is when the system is used in a manner other than that specified in the Technical Documentation.

Action area

The robot action area is the enveloping volume of the area occupied by the robot and its fixtures during movement in space.

1.2.3 Applicability

These Precautions are to be applied when carrying out the following activities:

- Installation and Putting into Service
- Operation in programming mode
- Auto / Remote Automatic Mode
- Robot axes brake
- Maintenance and Repair
- Putting Out of Service and Dismantling.

1.2.4 Operating modes

Installation and Putting into Service

- Putting into service is only possible when the Robot and Control System has been correctly and completely installed.
- The system installation and putting into service is exclusively the task of the authorised personnel.
- The system installation and putting into service is only permitted inside a protected area of an adequate size to house the robot and the fixtures it is outfitted with, without passing beyond the protection barriers. It is also necessary to check that under normal robot movement conditions there is no collision with parts inside the protected area (structural columns, power supply lines, etc.) or with the barriers. If necessary, limit the robot working areas with mechanical hard stop (see optional assemblies). If necessary, limit the robot working areas with mechanical hard stop (see optional assemblies).
- Any fixed robot control protections are to be located outside the protected area and in a point where there is a full view of the robot movements.
- The robot installation area is to be as free as possible from materials that could impede or limit visibility.
- During installation the robot and the Control Unit are to be handled as described in the product Technical Documentation; if lifting is necessary, check that the eye-bolts are fixed securely and use only adequate slings and equipment.
- Secure the robot to the support, with all the bolts and pins foreseen, tightened to the torque indicated in the product Technical Documentation.
- If present, remove the fastening brackets from the axes and check that the fixing of the robot fixture is secured correctly.
- Check that the robot guards are correctly secured and that there are no moving or loose parts. Check that the Control Unit components are intact.
- Install the Control Unit outside the protected area: the Control Unit is not to be used to form part of the fencing.
- Check that the voltage value of the mains is consistent with that indicated on the plate of the Control Unit.
- Before electrically connecting the Control Unit, check that the circuit breaker on the mains is locked in open position.
- Connection between the Control Unit and the three-phase supply mains at the works, is to be with a four-pole (3 phases + earth) armoured cable dimensioned appropriately for the power installed on the Control Unit. See the product Technical Documentation.
- The power supply cable is to enter the Control Unit through the specific cable entry and be properly clamped.
- Connect the earth conductor (PE) then connect the power conductors to the main switch.
- Connect the power supply cable, first connecting the earth conductor to the circuit breaker on the mains line, after checking with a tester that the circuit breaker terminals are not powered. Connect the cable armouring to the earth.

- Connect the signals and power cables between the Control Unit and the robot.
- Connect the robot to earth or to the Control Unit or to a nearby earth socket.
- Check that the Control Unit door (or doors) is/are locked with the key.
- A wrong connection of the connectors could cause permanent damage to the Control Unit components.
- The C5G Control Unit manages internally the main safety interlocks (gates, enabling pushbuttons, etc.). Connect the C5G Control Unit safety interlocks to the line safety circuits, taking care to connect them as required by the Safety standards. The safety of the interlock signals coming from the transfer line (emergency stop, gates safety devices etc.) i.e. the realisation of correct and safe circuits, is the responsibility of the Robot and Control System integrator.



In the cell/line emergency stop circuit the contacts must be included of the control unit emergency stop buttons, which are on X30. The pushbuttons are not interlocked in the emergency stop circuit of the Control Unit.

- The safety of the system cannot be guaranteed if these interlocks are wrongly executed, incomplete or missing.
- The safety circuit executes a controlled stop (IEC 60204-1 , class 1 stop) for the safety inputs Auto Stop/ General Stop and Emergency Stop. The controlled stop is only active under Automatic mode; under Programming mode the power is powered off immediately. The procedure for the selection of the controlled stop time (that can be set on SDM board) is contained in the Transport and Installation Manual of the Control Unit.
- When preparing protection barriers, especially light curtains and access doors, bear in mind that the robot stop times and distances are according to the stop category (0 or 1) and the weight of the robot.



Check that the controlled stop time is consistent with the type of Robot connected to the Control Unit. The stop time is selected using selector switches SW1 and SW2 on the SDM board.

- Check that the environment and working conditions are within the range specified in the specific product Technical Documentation.
- The calibration operations are to be carried out with great care, as indicated in the Technical Documentation of the specific product, and are to be concluded checking the correct position of the machine.

- To load or update the system software (for example after replacing boards), use only the original software handed over by COMAU Robotics. Scrupulously follow the system software uploading procedure described in the Technical Documentation supplied with the specific product. After uploading, always make some tests moving the robot at slow speed and remaining outside the protected area.
- Check that the barriers of the protected area are correctly positioned.

Operation in programming mode

- The robot is only to be programmed by the authorised personnel.
- Before starting to program, the operator must check the **Robotic system (Robot and Control Unit)** to make sure that there are no potentially hazardous irregular conditions, and that there is nobody inside the protected area.
- When possible the programming should be controlled from outside the protected area.
- Before operating inside the **Protected Area**, the operator must make sure from outside that all the necessary protections and safety devices are present and in working order, and especially that the hand-held programming unit functions correctly (slow speed, emergency stop, enabling device, etc.).
- During the programming session, only the operator with the Teach Pendant is allowed inside the **Protected Area**.
- If the presence of a second operator in the working area is necessary when checking the program, this person must have an enabling device interlocked with the safety devices.
- Activation of the motors (DRIVE ON) is always to be controlled from a position outside the range of the robot, after checking that there is nobody in the area involved. The Drive On operation is concluded when the relevant machine status indication is shown.
- When programming, the operator is to keep at a distance from the robot to be able to avoid any irregular machine movements, and in any case in a position to avoid the risk of being trapped between the robot and structural parts (columns, barriers, etc.), or between movable parts of the actual robot.
- When programming, the operator is to avoid remaining in a position where parts of the robot, pulled by gravity, could execute downward movements, or move upwards or sideways (when installed on a sloped plane).
- Testing a programmed cycle at working speed with the operator inside the protected area, in some situations where a close visual check is necessary, is only to be carried out after a complete test cycle at slow speed has been executed. The test is to be controlled from a safe distance.
- Special attention is to be paid when programming using the Teach Pendant: in this situation, although all the hardware and software safety devices are active, the robot movement depends on the operator.
- During the first running of a new program, the robot may move along a path that is not the one expected.
- The modification of program steps (such as moving by a step from one point to another of the flow, wrong recording of a step, modification of the robot position out of the path that links two steps of the program), could give rise to movements not envisaged by the operator when testing the program.

- In both cases operate cautiously, always remaining out of the robot's range of action and test the cycle at slow speed.

Auto / Remote Automatic Mode

- The activation of the automatic operation (AUTO and REMOTE states) is only to be executed with the [Robotic system \(Robot and Control Unit\)](#) integrated inside an area with protection barriers properly interlocked, as specified by Safety Standards currently in force in the Country where the installation takes place.
- Before starting the automatic mode the operator is to check the Robot and Control System and the protected area to make sure there are no potentially hazardous irregular conditions.
- The operator can only activate automatic operation after having checked:
 - that the Robot and Control System is not in maintenance or being repaired;
 - the protection barriers are correctly positioned;
 - that there is nobody inside the protected area;
 - that the Control Unit doors are closed and locked with the key;
 - that the safety devices (emergency stop, protection barrier devices) are functioning;
- Special attention is to be paid when selecting the automatic-remote mode, where the line PLC can perform automatic operations to switch on motors and start the program.

Robot axes brake

- In the absence of motive power, the robot axes movement is possible by means of optional release devices and suitable lifting devices. Such devices only enable the brake deactivation of each axis. In this case, all the system safety devices (including the emergency stop and the enable button) are powered off; also the robot axes can move upwards or downwards because of the force generated by the balancing system, or the force of gravity.



Before using the manual release devices, it is strongly recommended to sling the robot, or hook to an overhead travelling crane.

- Enabling the Brake releasing Module may cause the axes falling due to gravity as well as possible impacts due to an incorrect restoration, after applying the brake releasing module. The procedure for the correct usage of the Brake releasing Module (both for the integrated one and module one) is to be found in the maintenance manuals.

- When the motion is enabled again following the interruption of an unfinished MOVE, the track recovery typical function may generate unpredictable paths that may imply the risk of impact. This same condition arises at the next automatic cycle restarting. Avoid moving the Robot to positions that are far away from the ones provided for the motion restart; alternatively disable the outstanding MOVE programmes and/or instructions.

Maintenance and Repair

- At the COMAU works all robots are lubricated using products that do not contain any harmful substances. However, in some cases, repeated and prolonged exposure to such products may cause irritation of the skin and they may be harmful if swallowed.
- **First Aid Measures** In case of contact with the eyes or skin: rinse the affected areas with copious amounts of water; should irritation persist, seek medical advice. If swallowed, do not induce vomiting or administer anything by mouth; see a doctor as soon as possible.
- Maintenance, trouble-shooting and repairs are only to be carried out by authorised personnel.
- When carrying out maintenance and repairs, the specific warning sign is to be placed on the control panel of the Control Unit, stating that maintenance is in progress and it is only to be removed after the operation has been completely finished - even if it should be temporarily suspended.
- Maintenance operations and replacement of components or the Control Unit are to be carried out with the main switch in open position and locked with a padlock.
- Even if the Control Unit is not powered (main switch open), there may be interconnected voltages coming from connections to peripheral units or external power sources (e.g. 24 Vdc inputs/outputs). Power off external sources when operating on parts of the system that are involved.
- Removal of panels, protection shields, grids, etc. is only allowed with the main switch open and padlocked.
- Faulty components are to be replaced with others having the same Part number, or equivalent components defined by COMAU Robotics.



After replacement of the SDM module, check on the new module that the setting of the stop time on selector switches SW1 and SW2 is consistent with the type of Robot connected to the Control Unit.

- Trouble-shooting and maintenance activities are to be executed, when possible, outside the protected area.
- Trouble-shooting executed on the control is to be carried out, when possible without power supply.
- Should it be necessary, during trouble-shooting, to intervene with the Control Unit powered, all the precautions specified by Safety Standards are to be observed when operating with hazardous voltages present.
- Trouble-shooting on the robot is to be carried out with the power supply powered off (DRIVE OFF).
- At the end of the maintenance and trouble-shooting operations, all deactivated safety devices are to be reset (panels, protection shields, interlocks, etc.).

- Maintenance, repairs and trouble-shooting operations are to be concluded checking the correct operation of the **Robotic system (Robot and Control Unit)** and all the safety devices, executed from outside the protected area.
- When loading the software (for example after replacing electronic boards) the original software handed over by COMAU Robotics is to be used. Scrupulously follow the system software loading procedure described in the specific product Technical Documentation; after loading always run a test cycle to make sure, remaining outside the protected area
- Disassembly of robot components (motors, balancing cylinders, etc.) may cause uncontrolled movements of the axes in any direction: before starting a disassembly procedure, consult the warning plates applied to the robot and the Technical Documentation supplied.
- It is strictly forbidden to remove the protective covering of the robot springs.

Putting Out of Service and Dismantling

- Putting out of service and dismantling the Robot and Control System is only to be carried out by **Authorised Personnel**.
- Move the robot to transport position and mount the axes locking items (if present), following the instructions on the plate posted on the robot and its Technical Documents.
- Before stating to put out of service, the mains voltage to the Control Unit must be powered off (switch off the circuit breaker on the mains distribution line and lock it in open position).
- After using the specific instrument to check there is no voltage on the terminals, disconnect the power supply cable from the circuit breaker on the distribution line, first disconnecting the power conductors, then the earth. Disconnect the power supply cable from the Control Unit and remove it.
- First disconnect the connection cables between the robot and the Control Unit, then the earth cable.
- If present, disconnect the robot pneumatic system from the air distribution mains.
- Check that the robot is properly balanced and if necessary sling it correctly, then remove the robot securing bolts from the support.
- Remove the robot and Control Unit from the working area, following all prescriptions in the product Technical Documents; in case of lifting, check the eyebolts fastening and use only suitable slinging devices and equipment.
- Before starting dismantling operations (disassembly, demolition and disposal) of the Robot and Control System components, contact COMAU Robotics & Service, or one of its branches, who will indicate, according to the type of robot and Control Unit, the operating methods in accordance with safety principles and safeguarding the environment.
- The waste disposal operations are to be carried out complying with the legislation of the country where the Robot and Control System is installed.

1.2.5 Performance

The performances below shall be considered before installing the robotic system:

- Stop spaces
- Mission time (typical value).

Stop spaces

- With Robot in programming mode (T1), if you press the stop pushbutton (red mushroom-shaped one on WiTP) in category 0 (According to Standard EN60204-1), you will obtain:

Tab. 1.1 - Stopping spaces in programming mode (T1)

Mode	Expected speed	Case	Stopping distance	Stopping space
T1	250 mm/s	Nominal	120 ms	30 mm
		Limit	500 ms	125 mm

Tab. 1.2 - Safety electronics reaction time in programming mode (T1)

Mode	Expected speed	Case	Reaction time
T1	250 mm/s	For the safety inputs of the SDM module (e.g. stop pushbutton of TP in wired version)	150 ms
		For the stop stop and enabling device inputs from the TP in wireless version, when the safety wire transmission is active.	
		For the time-out of stop input and enabling device from TP in wireless version, when the safety wire transmission is lost or interrupted.	350 ms

- Considering the Robot in automatic mode, under full extension, full load and maximum speed conditions, if you press the stop pushbutton (red mushroom-shaped one on WiTP) in category 1 (according to norm EN60204-1) you will trigger the Robot complete stop with controlled deceleration ramp. Example: for Robot NJ 370-2.7 you will obtain the complete stop in about 85 ° motion, that correspond to about 3000 mm movement measured on TCP flange. Under the said conditions, the stopping time for Robot NJ 370-2.7 is equal to 1,5 seconds.
- For each Robot the limit stop distances can be required to COMAU Robotics.

Mission time (typical value)

- We remind you that the safety system efficiency covering is equal to 20 years (mission time of safety-related parts of control systems (SRP/CS), according to EN ISO 13849-1).

2. USER INTERFACE

2.1 Introduction

The C5G Robot Control Unit, for robot manual handling, to create, modify and run programs, to modify step-by-step movements, to supply system control and monitoring functions, requires the following user interfaces:

- Teach Pendant (TP5)
 - Interface on Personal Computer (WinC5G)
-

2.2 Teach Pendant (TP5)

The C5G Robot Control Unit Teach Pendant is used to manually control the robot movements, to program it and execute and modify step by step movements; it supplies system control and monitoring functions, as well as including the safety devices (enabling device and emergency stop pushbutton). It is user-friendly and is suitable for both right-hand and left -hand use.

It includes the following components:

- Display
- Keys
- Pushbuttons and LEDs
- USB port.



The TP5 Teach Pendant is connected by a cable to the Control Unit
For more detailed information about the Teach Pendant, see the following manuals:
– [C5G Control Unit - Technical Specification - par.3.3 Teach Pendant](#);
– [C5G Control Unit - Transport and Installation - cap.5 Teach Pendant: connection](#);
– [Cap.5. - TP5 Teach Pendant on page 50](#).

The following [Fig. 2.1](#) and [Fig. 2.2](#) illustrate the components of the Teach Pendant. For more detailed information about modal selector switch, keys, pushbuttons, LEDs, display and communication port, see the following [Cap.5. - TP5 Teach Pendant on page 50](#).

2.3 Front view

Fig. 2.1 - TP5 front view



2.4 Rear view

Fig. 2.2 - TP5 rear view



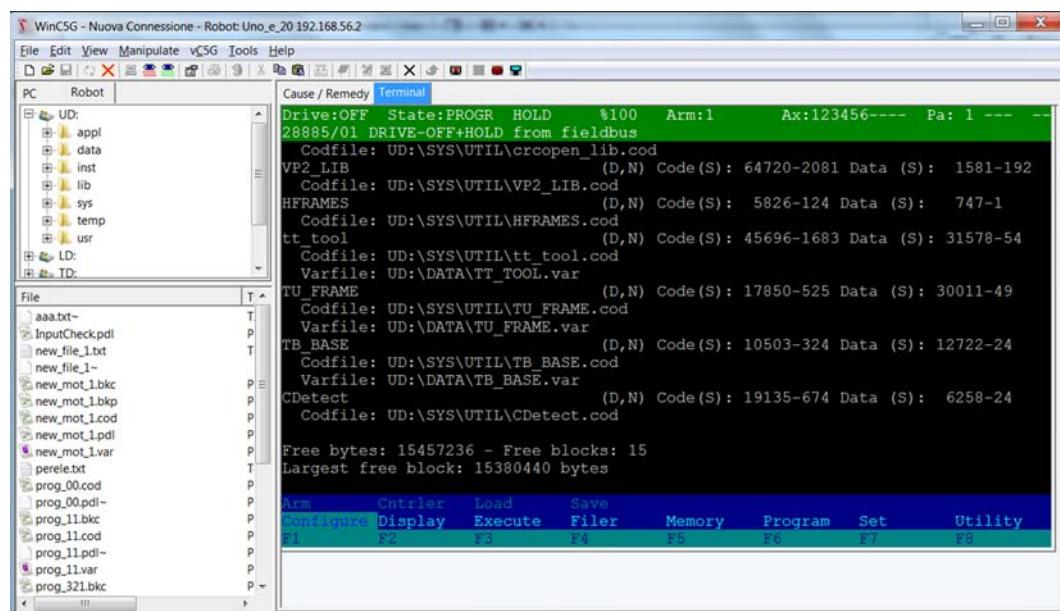
2.5 Interface on Personal Computer (WinC5G)

WinC5G program is the interface on Personal Computer to C5G Robot Control Unit.

It includes a set of functions, which are:

- display of files existing in the Control Unit,
- possibility of editing them, translating them into executable format (.COD) and running them,
- searching and displaying errors,
- possibility to issue commands to the Control Unit,
- conversion of already existing programs depending on new reference points.

For further details, see the [Cap. WinC5G Program - Interface to C5G on Personal Computer on page 364](#).



3. SYSTEM POWER ON AND SHUT DOWN

3.1 Foreword

This chapter provides all information needed to switch the power on/off to C5G Controller Unit (see [Fig. 3.1](#)):

Fig. 3.1 - C5G Controller Unit



Front view

Rear view

The following topics are described in detail:

- [Main switch](#)
- [System power-on](#)
- [System shut down](#)

3.2 Main switch

To Power on/off the System, it is needed to use the main switch. It is placed on the Controller Unit Cabinet door, upward on the left. The available versions are shown in the following [Fig. 3.2](#).

Fig. 3.2 - Main switch



Europe version



North America version

The main switch acts on the supply voltage of the Controller Unit.

3.3 System power-on

To power on the C5G Robot Control Unit, proceed as follows:

- a. close the Robot Controller Cabinet door.
- b. Check that the following cables are connected:
 - power supply cable
 - X10 and X60 cables towards the robot
 - X30 safety devices cable



Wait at least 30 seconds after a power-off, before powering the control unit on again.

- c. Turn on the electric power supply by turning the main switch to ON (see [par. 3.2 Main switch on page 37](#)).
Upon Control Unit activation, power is supplied to all the modules
- d. in order for the user to be able to use the Teach Pendant, the system activates TP5 display, executes the [STARTUP Procedure](#), displays the [Start Page](#) and makes the Teach Pendant ready to use



WARNING

If the TP software version does not match with the one existing onto the Controller, a message is displayed stating that the system will automatically align the two versions, within 15 seconds. Press 'Cancel' to avoid such an operation.
It is anyway recommended to align the software versions.

- e. If the execution of any robot movement is required (motion program activated by the STARTUP program) it is needed to power the drives on (see the description of DRIVE ON functionality in [par. 5.5.1.2 Right Menu on page 69](#)) and press **START** button (see the description of **START** (green) button, in [par. 5.3.1.3 Other colours keys on page 60](#)).

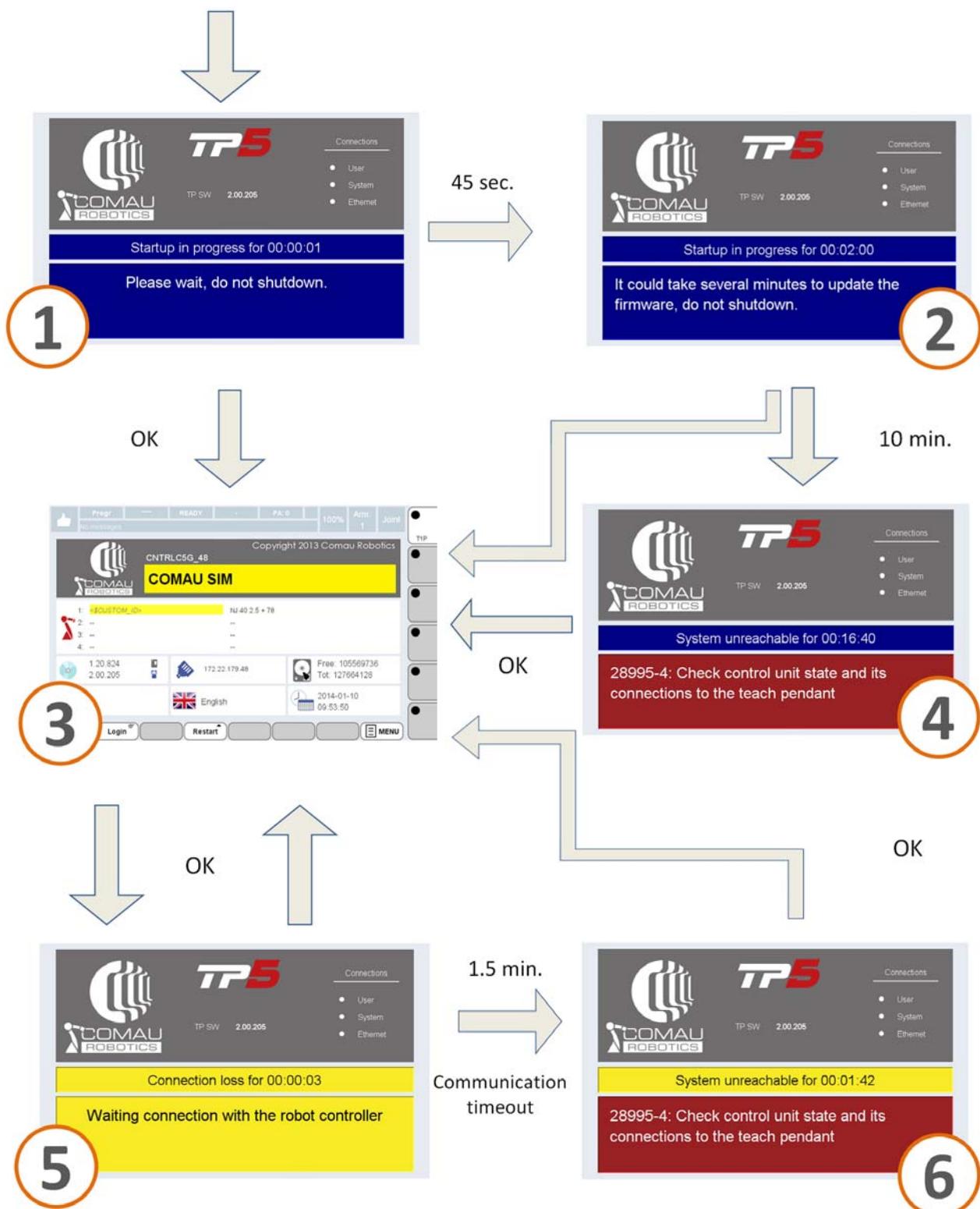


To handle automatic motion, please refer to [Cap.10. - Start and Stop Cycle on page 418](#).

3.3.1 STARTUP Procedure

Fig. 3.3 - STARTUP Procedure

Power Up



- At power on, the system displays the [Connection Page](#) shown in the above diagram (step 1). Do not shutdown the system until operations are completed.
- If the connection step takes more than 45 seconds, some update operations could be running for additional firmware modules. If so, the Teach Pendant displays a different message (step 2), and the user must wait for the operation to be accomplished, without powering the system off.
- If such a connection step takes more than 10 minutes, the system displays an error information on red background (step 4). Please, be sure that both the Controller has properly been started up and the connections towards the Teach Pendant are correct. In some cases, switching off and subsequent switching on the system could help fixing the situation. **Always refer to the Cause&Remedy information, related to the shown error message on the Teach Pendant display.**
- During all the described above situations, the Teach Pendant continuously tries to connect to the Controller. As soon as the connection is ok, the [Start Page](#) is displayed (step 3) on the Teach Pendant and the system is ready.



CONNECTION FAILURE

During any working phase, the communication between Control Unit and Teach Pendant could fail due to connection lost, upon communication time-out.

An alarm message is displayed on yellow background (step 5), indicating the elapsed time from connection lost.

If it is not restored within 1.5 minutes, an alarm message is displayed on red background (step 4 and step 6), with the error code, to ask the user to check both the Control Unit and the Teach Pendant connection states.

As soon as the proper working situation is restored, the [Start Page](#) (step 3) is displayed and the system is ready again.

3.4 System shut down

Even if the system can be shut down by just moving the [Main switch](#) to OFF, it is strongly recommended the use of the **software shutdown procedure**, in order to avoid unnecessary waste of UPS buffer battery cycles that would reduce the life of the battery:

- a. set the system in **DRIVE OFF**
(see DRIVE OFF function description in [par. 5.5.1.2 Right Menu on page 69](#))
- b. activate the software shutdown, according to one of the following modalities:
 - b.1 by means of the **TP5** Teach Pendant - in the [Start Page](#), issue [Restart](#), [Shutdown](#) command;
 - b.2 from a Personal Computer (**WinC5G** program) - issue [CONFIGURE CONTROLLER RESTART SHUTDOWN \(CCRS\)](#);
 - b.3 from within a **PDL2** program - use **SYS_CALL** built-in routine, by specifying **CCRS** command.



WARNING!

- In the case in which the command comes from within a program (by means of SYS_CALL), it is suggested to always check the SYS_CALL returned state in order to be able to handle possible anomalous situations.

- c. 5 seconds later, the TP5 screen is cleaned off. Wait for at least 15 seconds more.
- d. move the main switch to OFF.
(see [par. 3.2 Main switch on page 37](#)).

4. ACCESS TO THE CONTROL (LOGIN/LOGOUT)

4.1 Introduction

To access C5G Control Unit, regardless of the device from which dialogue with the Controller is desired (Teach Pendant or **WinC5G** program from PC), a [Login](#) is necessary, otherwise the only available commands are for viewing.

To terminate access to the Control Unit, so that unauthorised personnel cannot enter, it is necessary to execute a [Logout](#).

4.2 Login

If the user is working with the Teach Pendant, the Login functionality is available in the Teach Pendant [Start Page](#), either:

- by means of shortly touching the [Login](#) command, in the Function keys menu (highlighted in red in the following figure)



or

- by touching the Login field (highlighted in red in the following figure)



Using one of the listed above modalities, the system access environment (login/logout) is then activated.

When working on a PC (**WinC5G** program), the system asks the user to perform a Login operation at connection time towards the Controller; it is available the **SET LOGIN (SL)** command in the commands menu.

The following topics are now fully described:

- [User profile](#),
 - [Summary table of access rights](#).
-

4.2.1 User profile

It is always needed to specify a Username and a Password which have to be acknowledged by the Controller to be interfaced; these must be defined and saved beforehand in the Controller database.

There are six types of predefined and recognised by the control unit users:

- [Administrator](#)
- [Default](#)
- [Maintenance](#)
- [Programmer](#)
- [Service](#)
- [Technology](#)

They are described in the following sections.

The access rights are associated to each of these categories, and enable or disable the use of a certain command, also according to the system status.

These profiles definition is assigned to an **administrator** user who can declare them, after Logging in on the Controller, in the data base of the Controller, sending:

- [Add](#) command, available by pressing [Users](#) softkey in the [Setup page - Login](#) subpage on the Teach Pendant, or
- [CONFIGURE CONTROLLER LOGIN ADD \(CCLA\)](#) command and the associated option, from within the **WinC5G** Terminal window.

Then, it is necessary to execute

- [Configure](#) command, available in the [Setup page](#), son the Teach Pendant, or
- [CONFIGURE SAVE ALL \(CSA\)](#) command, from **WinC5G**, on PC

The C5G Control Unit is delivered to the customer with the following predefined users:

- **programmer** user
 - Username: pu
 - Password: pu
- **maintenance** user
 - Username: mu
 - Password: mu
- **administrator** user
 - Username: admin
 - Password: admin

Se gli utenti predefiniti (**programmatore** o **manutentore**) non incontrano i requisiti del cliente finale, è possibile definirne altri.

To do so, from **administrator** user, issue [Add](#) command, available by pressing [Users](#)

softkey in the [Setup page - Login](#) subpage on the Teach Pendant, or [CONFIGURE CONTROLLER LOGIN ADD \(CCLA\)](#) command and the associated option, from within the **WinC5G** on Personal Computer.

The **programmer** login is the default one that is held after a Restart Cold, since it is saved as Startup Login ([Startup](#) command from [Users](#) subpage, [Setup page](#) on the Teach Pendant).

If the predefined users (**programmer** or **maintenance**) don't meet the final client requirements, it is possible to define more profiles.

To do so, from **administrator** user, issue [Add](#) command, available by pressing [Users](#) softkey in the [Setup page - Login](#) subpage on the Teach Pendant, or [CONFIGURE CONTROLLER LOGIN ADD \(CCLA\)](#) command and the associated option, from within the **WinC5G** on Personal Computer.

4.2.1.1 Administrator

The only task of this user is to enter and/or delete the users in the data base (file .UDB) that gives access to the system; therefore many other commands are not enabled for the Administrator.

Each time it is wished to add (or remove) a user level, it is necessary to perform the following steps:

- a. enter the system as **administrator** user ([Login](#) key from the Teach Pendant [Start Page](#)). If a new profile for the **administrator** has not already been defined, the Username **admin** and the Password **admin** are predefined in the system;
- b. to add a new user, issue either [Add](#) command, available by means of [Users](#) key, in the [Setup page](#), [Login](#) subpage on the Teach Pendant or [CONFIGURE CONTROLLER LOGIN ADD \(CCLA\)](#) command and the associated option, from within **WinC5G**, on a PC
- c. to check that the insertion has taken place, select [Login](#) subpage, [Users](#) key in the [Setup page](#)
- d. perform the Logout to exit from the system as administrator: from the Teach Pendant [Start Page](#), select the [Logout](#) option by long touching the [Login](#) key
- e. Login again entering the newly defined Username and Password, to check it has been properly set;
- f. save the settings using one of these methods:
 - f.1 issue [Configure](#) command, available in the [Setup page](#), on the Teach Pendant, or
 - f.2 issue [CONFIGURE SAVE ALL \(CSA\)](#) command, from within **WinC5G**, on the PC.

4.2.1.2 Default

The type of user to be identified is the operator running a production line. The main operations he/she requires are to start and stop programs, to delete alarms, manual movements, override modification, restart and shut down of the Controller.

4.2.1.3 Programmer

The programmer user is enabled, mainly, to execute operations associated to the development, the verification and the setting up of the programs.

4.2.1.4 Maintenance

The type of user to be identified is the integrator. This user is more powerful than the programmer.

4.2.1.5 Service

This kind of user represents the After Sales Service and is enabled to execute operations connected to system updating, using commands for software loading and machine calibration.

Access to the Control Unit with this profile, is exclusive: if a user connects to the system as Service, it will be possible to connect to the Control Unit, from other devices, only specifying the same Username and the same Password.

4.2.1.6 Technology

The Technology user allows accessing to some functionalities related to the currently installed application, typical of such an application.

It implies the definition of some using levels depending on the application and the INTEGER value which has been specified as a parameter at the profile definition time. Please refer to the application manual to better understand which value is to be assigned to such a parameter (indicating the technology).

It can be either associated to a profile chosen among **Programmer**, **Service**, etc. or customized by means of the **.UDB** file, from within **WinC5G** (see [par. 7.5.1.6 Viewing and editing of .UDB file \(optional feature\) on page 398](#)).

4.2.2 Summary table of access rights

The following table indicates the current settings about access rights of the listed above predefined users. The codes in the table, associated to the users are: D for **Default**, A for **Administrator**, M for **Maintenance**, P for **Programmer** and S for **Service**.

As far as the **Technology** user, the access rights are the same of the associated profile (either **Programmer**, **Service** or **Maintenance**).

The corresponding command from SYS_CALL is indicated in brackets.

COMMAND	PROGR	LOCAL	REMOTE
Add a login to the database (CCLA)	A	A	A
Delete a login from the database	A	A	A
Assign the Startup Login (CCLS)	P,M,S	P,M,S	P,M,S
Assign the Startup program (CCS)	P,M,S	P,M,S	P,M,S
Assign the time (CCT)	M,S	M,S	M,S

COMMAND	PROGR	LOCAL	REMOTE
Jog an arm	D,P,M,S		
Modify the override (SAG). Set the active arm for the Teach Pendant (SAT)	D,P,M,S	D,P,M,S	D,P,M,S
Disable stroke ends errors (SAN)	P,M,S		
Turn Set operation on the axes (CAT)	D,P,M,S		
Load axis data from retentive memory (CARL)	M,S		
Save data in retentive memory (CARS)	M,S	M,S	M,S
Arm: enable (SAE), disable (SAD), simulation (SAS), simulated state removal (SAU).	S	S	S
Calibration of an arm (CAC)	S		
Controller restart (CCRC)	A,D,P,M,S	A,D,P,M,S	A,P,M,S
Controller shut-down (CCRS)	D,P,M,S	D,P,M,S	D,P,M,S
Load Controller system software (CCRR)	M,S		
Lock keyboard (SCK)	M,S	M,S	M,S
Load configuration file .C5G and User Data Base file .UDB (CLA,CLC)	M,S		
Save configuration file .C5G and User Data Base file .UDB (CSA, CSC)	A,P,M,S	P,M,S	P,M,S
Execution of an instruction (E)	P,M,S	P,M,S	S
Delete(FD), rename (FR), translate (FT) a file	A,P,M,S	P,M,S	P,M,S
Create (FUDM), delete (FUDD) a directory	A,P,M,S	P,M,S	P,M,S
Compressed files management (FUC)	A,P,M,S	P,M,S	P,M,S
Applications installation (FUI)	P,M,S	P,M,S	P,M,S
Copy file from external device (FUR)	P,M,S	P,M,S	P,M,S
Change attributes of a file (FUA)	M,S	M,S	M,S
Access to Memory Debug environment (MD)	P,M,S	P,M,S	
Program Edit and IDE environment	P,M,S	P,M,S	P,M,S
Load programs in memory (ML), Save in UD: (MS)	A,P,M,S	P,M,S	P,M,S
Delete programs (MEP), variables (MEV), both (MEA) from memory	A,P,M,S	P,M,S	M,S
Program activation (PA), deactivation (PD), loading and activation (PG)	A,D,P,M,S	D,P,M,S	M,S
Programs debug: temporary interruption of execution (PSP), resume execution (PSU), bypass on suspensive instruction (PSB), breakpoint insertion/deletion (PTB)	P,M,S	P,M,S	
Change program step (PTS)	P,M,S	P,M,S	P,M,S
Operations on Inputs (except access to privileged inputs) (Set Input...)	P,M,S	P,M,S	P,M,S

COMMAND	PROGR	LOCAL	REMOTE
Operations on Outputs (SetOutput...)	P,M,S	P,M,S	P,M,S
Forcing of privileged inputs (SIFP...) / outputs (SOFP.)	S	S	S
Start / interrupt a protocol (UCM, UCD...), set properties of a port (UCP), or the default port (UCS)	P,M,S	P,M,S	M,S
Access to all devices	S	S	S
Deactivate all programs through Cntrl Y	M,S		
Display contents of all log files	S	S	S
Table opening (from the TP Data Page)	A,M,P,S	A,M,P,S	A,M,P,S
Table modification (from the TP Data Page)	A,M,P,S	A,M,P,S	A,M,P,S
Program Editing (from the TP IDE Page) (*)	A,M,P,S	A,M,P,S	A,M,P,S
Program Viewing (from the TP IDE Page)	A,M,P,S	A,M,P,S	A,M,P,S

D: default, the user who is defined by [Setup page - System - Login - Users - Add](#) without options.
 A: Administrator, defined with [Setup page - System - Login - Users - Add](#) selecting /Admin profile
 M: Maintenance, defined with [Setup page - System - Login - Users - Add](#) selecting /Maintenance profile
 P: Programmer, defined with [Setup page - System - Login - Users - Add](#) selecting / Programmer profile
 S: Service, defined with [Setup page - System - Login - Users - Add](#) selecting / Service profile

(*) compliant with the Profile rights

It is also allowed to set a startup login used by the Controller at each restart (either [\(Setup page - System - Login - Startup](#) command on the Teach Pendant, or [CONFIGURE CONTROLLER LOGIN STARTUP \(CCLS\)](#)) from within [WinC5G](#), on Personal Computer).

4.3 Logout

If working with the Teach Pendant, apply a long touch on [Login](#) key in the [Functional keys menu](#) in the [Start Page](#), and choose [Logout](#) to start the logout operation.

If working on [WinC5G](#), press the disconnection key, or select [SET LOGIN \(SL\)/Logout](#) command from the command menu.

4.3.1 Automatic Timed Logout

This function has been implemented to prevent unauthorized people executing potentially hazardous operations during the system processing cycle.

The main purpose is that the system generates a Logout after the Teach Pendant user has remained inactive for a certain period of time.

This time is linked to the keyboard and touch screen activity: there is a timeout to shut down the Teach Pendant screen after a period of keyboard and touch screen inactivity (see [par. 5.5 Display on page 65](#)).

From that instant the time count starts, after which the system executes the automatic Logout.

When a user wishes to use the Teach Pendant **after an automatic Logout**, it is necessary to access again (Login).

When an automatic Logout takes place, some environments carry out special

operations. In particular:

- [Files Page](#) - the system resets the current directory to UD;
- [Data Page](#) - any open tables are "frozen": all modifications remain where they are at that moment (either in TP5 memory or in Controller memory). The system warns the user with an "Insufficient Rights" message. At the next Login the message is no longer present and the table is displayed again
- [IDE Page](#) - the system closes the current editing session. If such an operation requires viewing one or more dialogue windows towards the user (for example relating to the cursor position and modifications that have not been saved), the automatic Logout procedure is suspended until the user answers these questions.



For further information about configuring this functionality by the user, see [Autologout](#) in the [Setup page](#).

5. TP5 TEACH PENDANT

5.1 Introduction

The current chapter provides detailed information on how to use **TP5** Teach Pendant.



- | | |
|---------------------------|---------------------|
| 1 - Modal selector switch | 2 - Stop pushbutton |
| 3 - Display | 4 - Blue keys |
| 5 - Other colours keys | 6 - LEDs |
| 7 - JPAD | 8 - JOG keys |

The following subjects are dealt with:

- Modal selector switch
- Keys, Pushbuttons and LEDs
- USB port
- Display
- User Interface Pages

5.2 Modal selector switch

It allows selecting the command mode for the Control Unit (see Fig. 5.1):

- **T1** - The programmer operator can carry out editing sessions, teaching positions and checking the programs properly work. The robot moves at low speed, not more than 250 mm/s to flange centre, to tool centre point (TCP) and to any extreme tool. The operator can operate inside the cell.



Warning! The program ALWAYS starts at low speed, and the user, if wished, CAN increase it to the working speed. If jogkeys are used, the system will AUTOMATICALLY reduce the speed to below the limit of 250 mm/s at flange centre.



T1P Status

In addition to the standard T1 status which is suitable for programming, T1P status exists (which can be enabled/disabled by pressing the dedicated 'T1P' key in the [Right Menu](#)) to better execute the technological processes in programming mode, always complying with the safety speed constraint of 250 mm/s.

Note that T1P status DOES NOT refer to a different position of the modal selector switch!

Main functions and limitations:

- **switching from T1 to T1P and viceversa can be only performed with DRIVES OFF.**
- **In T1P mode the speed is limited to 250 mm/s, like it happens in T1 mode.**
- **The trajectories the robot executes in T1P status and the ones in Automatic cycle, are exactly the same, with the above mentioned maximum speed constraint.**
- **Jogging is NOT allowed in T1P status.**
- **Switching from T1P to Automatic (either AUTO or REMOTE) and switching back the selector to T1, the previous T1P setting is lost.**

- **AUTO** - The operator can check the program functioning at working speed, by activating the start commands from the Teach Pendant. The operator cannot work inside the cell.



This command mode is NOT available in North America versions.

- **REMOTE** - The program execution is controlled by external equipment (for example, line PLC or local control panels). The operator cannot work inside the cell.

In case of **TP5** Europe version, the modal selector key can be pull out in all the positions.

Fig. 5.1 - TP5 modal selector switch



Europe version

North America version

5.3 Keys, Pushbuttons and LEDs

- Keys
- Pushbuttons and LEDs

5.3.1 Keys

The Teach Pendant keyboard is basically arranged in the following manner:

- Blue keys
- Black keys
- Other colours keys.



5.3.1.1 Blue keys



- Special keys
- General purpose keys
- Navigation general keys.

5.3.1.1.1 Special keys



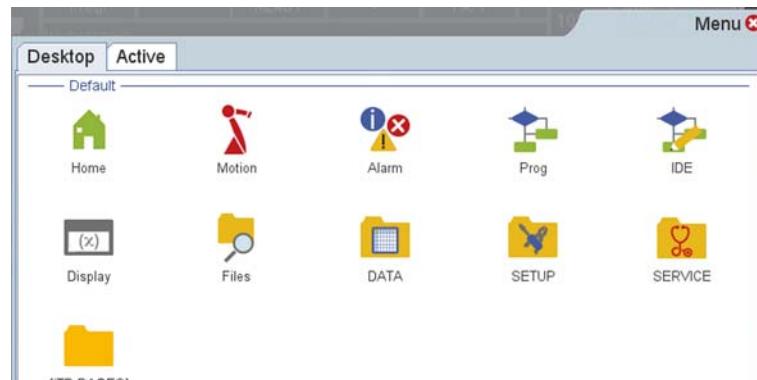
- MENU
- SPLIT
- TOGGLE
- QUICK.

MENU



Pressing this key allows accessing to the icons of all the environments available to the user.

Select the wished page by means of either the touch screen or the arrow keys.



Please refer to Chap. MENU Page on page 97 for a detailed description.

SPLIT

Reserved

TOGGLE

This key allows moving the cursor among the currently active windows on the TP screen.

QUICK



This key allows handling “short commands” created by the user.

When pressing it, the already existing short commands are made available and the user is allowed to create new ones.

In the following example, an already existing short command, called *ml_p1*, is displayed.



Please refer to Chap. QUICK Page on page 94 for a detailed description.

5.3.1.1.2 General purpose keys

- SHIFT
 - MORE.
-

SHIFT


Always in combination with other keys; the use changes according to the environment and the function of the key it is associated to.

MORE


When the **Right Menu** keys are more than 6, the **MORE** key allows scrolling of them all. They are displayed circularly.

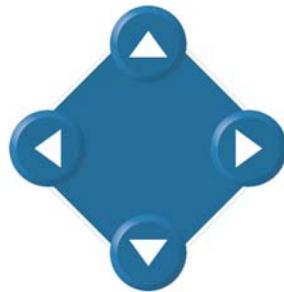
5.3.1.1.3 Navigation general keys

- ENTER and ESC
 - Cursor keys (up/down/right/left).
-

ENTER and ESC


- **ENTER** to confirm the current operation.
- **ESC** to go back, deleting the current operation.

Cursor keys (up/down/right/left)

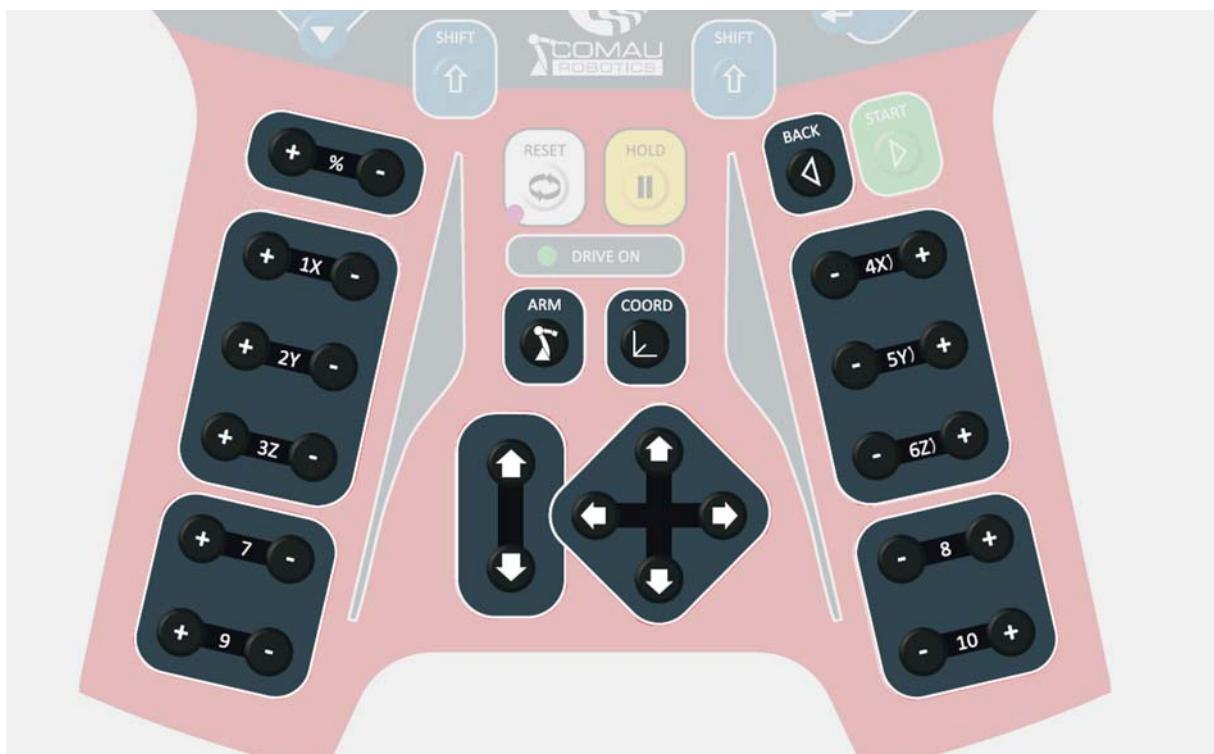


They allow moving among the different objects in the TP display; they can also be used in combination with the **SHIFT** keys.

5.3.1.2 Black keys

These keys are dedicated to the robot motion.

- BACK
- COORD
- ARM
- % + e -
- JOG keys
- JPAD.



5.3.1.2.1 BACK



Causes the movement backward, to the start position of the current movement, during step-by-step checking of a program.

For further information refer to [par. 5.15.3.1.8 Step on page 137](#), in [IDE Page](#).

5.3.1.2.2 COORD



Selects the Reference System:

- **JOINT** - JOINT mode. The keys are associated to each of the axes of the selected arm; any auxiliary axes, if present, follow those of the arm. Pressing one of such keys determines the movement of the corresponding axis in positive or negative direction, according to the direction indicated by the plate on the arm.
- **UFRAME** - linear movement mode, according to the User Reference Frame x, y, z (for example the frame of the being machined workpiece). The first three keys allow linear movements in the direction of the three axes of the user reference frame (defined by \$UFRAME predefined variable); the next three keys allow tool rotations around the same axes keeping the TCP position unchanged.
- **BASE** - linear movement mode according to the World Reference Frame x, y, z (the workshop reference frame). The first keys allow linear movement in the direction of the three axes of the world reference system, the next three keys allow the tool rotation around the same axes keeping the TCP position unchanged. Note that the World Frame is not directly defined by any system variable; in fact, it is the robot base that is represented referred to the world, through the \$BASE variable.
- **TOOL** - linear movement mode according to the Tool Reference Frame x, y, z (or TCP frame). The first three keys allow linear movement in the direction of the three axes of the Tool Reference Frame (defined by \$TOOL predefined variable); the next three keys allow tool rotation around the same axes keeping the TCP position unchanged (tool work point).
- **WR-BASE, WR-TOOL, WR-UFRAME** - if pressed together with the **SHIFT** key, the **COORD** key allows passing among Cartesian movement modes (BASE, TOOL, USER) and Wrist JNT (X,Y,Z and joints of axes 4,5,6).

For further information see [Chap.4. - Robot motion in Programming mode on page 36](#) - [Motion Programming](#) manual.

5.3.1.2.3 ARM



In **Multiarm** systems, it is used to manage the index of the main Arm and the synchronised Arm, with subsequent displaying in the [Status bar](#), in exactly the same way as by entering in the [Motion Page](#), sub-page [Basic](#) and modifying [ArmSyncArm](#) field.

In DRIVE-OFF - pressing **ARM** increases in circular mode the index of the main Arm, never altering the Arm quantity in the [Status bar](#) (if there is only one Arm, it remains one, if there are two Arms, two remain).

The new value is the first valid Arm index after an increment.

Pressing **SHIFT+ARM** increases in circular mode the index of the synchronized Arm. It is the only combination of keys able to change the quantity of Arms in the [Status bar](#). It always passes to two Arms when there is only one Arm in the [Status bar](#).

It returns to one Arm when the index of the second Arm becomes equal to the first Arm.

In DRIVE-ON - pressing **ARM** has two effects:

- if DRIVE-ON has taken place with **only 1 selected Arm** ($\$TP_SYNC_ARM[2]=0$), the Arm index is incremented, as for DRIVE-OFF;
- if DRIVE-ON has taken place with **2 selected Arms**, the selection is inverted for the Arm that will be moved with jog keys.

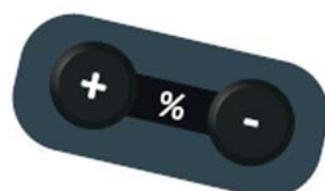
For example, if there are 3 Arms and the current situation is **Arm: 2<1**, where Arm 2 is the one moved by jog keys, the index is NOT incremented (it does NOT change to 3) pressing **ARM** key, but the situation becomes **Arm: 1<2**, i.e. the Arm moved by jog keys is now number 1.

The **first** displayed **number** is always the **Arm moved by jog keys**. **SHIFT+ARM** cannot be pressed.



A **LONG PRESSING** of the **ARM** key cyclically displays possible indexes of the main Arm: when the required index is reached, releasing the key makes this choice definite. While the key is pressed, the index is displayed in brackets. When the final choice is made, the brackets disappear.

5.3.1.2.4 % + e -

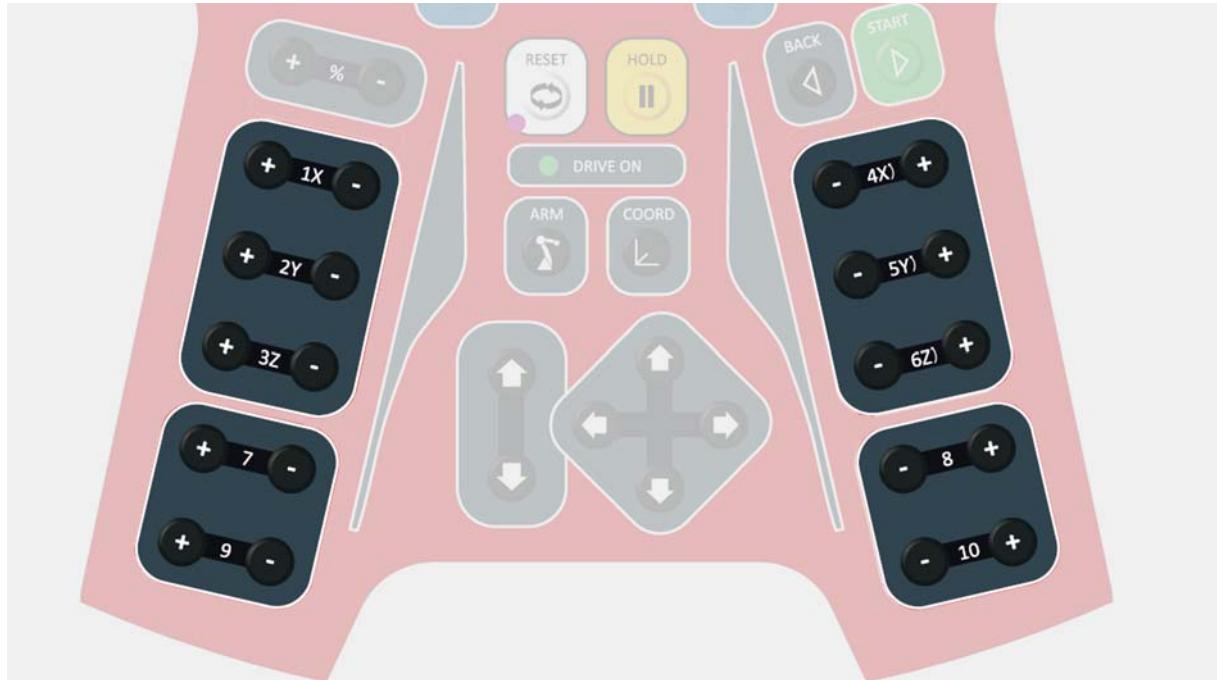


They can be used to change the OVERRIDE. Combined with **SHIFT** key the following

values can be obtained:

- **SHIFT -%** --> 25%
- **SHIFT +%** --> 100%

5.3.1.2.5 JOG keys

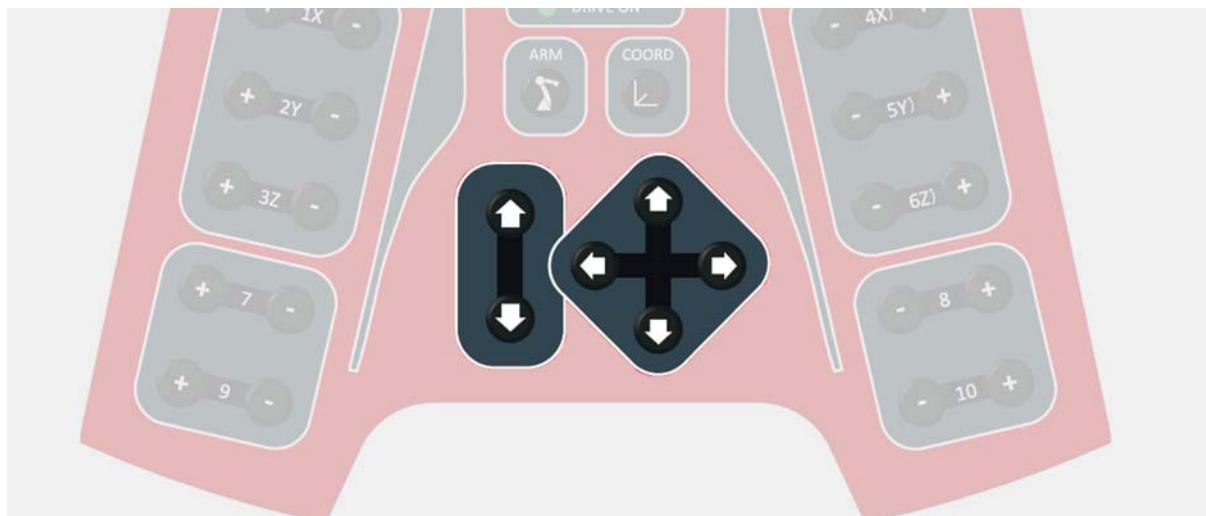


Such keys can be used to move the robot axes. Keys are available for all the axes the Control Unit can handle, i.e. 1 to 10.



Note that, in some particular scenarios, JOG keys could be disabled by the System. Please refer to either the **Robot** or the **Control Unit Maintenance** manual, to get more specific information.

5.3.1.2.6 JPAD



The JPAD group allows jogging referred to the **USER position**. Furthermore:

- the two arrow keys on the left refer to movements along axis z: upward and downward movements
- the four keys on the right are respectively for movements along x axis (upward arrow and downward arrow) and along y axis (left arrow and right arrow). The movements along x axis are for moving away, and for approaching the user; the movements along y axis are for moving left and right, always referred to the user.

These keys are NOT allowed in the following cases:

- when either a cooperative motion is active or with remote tool or conveyor,
- on disabled axes
- when a jog key is already pressed.



WARNING - For configuration with integrated Gantry with 3 linear axes, the JPAD is managed differently to normal operation.

- **The two arrow keys on the left refer to movements of the gantry auxiliary axis configured as Z**
- **The four keys on the right refer respectively to movements of gantry auxiliary axis configured as X (up arrow and down arrow) and to movements of auxiliary axis configured as Y (LH arrow and RH arrow).**

The position of the user does NOT have any influence on the JPAD functioning, and therefore the controls in [Advanced](#) sub-page of [Motion Page](#), DO NOT affect handling with JPAD.

Gantry movement with JPAD is the same as with [JOG keys](#) 7-8-9-10.



Please remember that any movement executed using JPAD is always according to user reference (\$UFRAME), except for movement of Gantry with 3 linear axes, which is always a joint movement.

The user position can be configured in the [Advanced](#) sub-page of the [Motion Page](#).

5.3.1.3 Other colours keys



- **RESET** (white)
- **HOLD** (yellow)
- **START** (green).

5.3.1.3.1 RESET (white)



Pressing this key allows resetting the ALARM state; in case in which the alarm has been caused at safety devices level (e.g.: TP5 mushroom button, external mushroom button, automatic line barrier, etc.), the alarm is not deleted until the cause has been removed. The corresponding [Red LED](#) is associated to the RESET key.

5.3.1.3.2 HOLD (yellow)



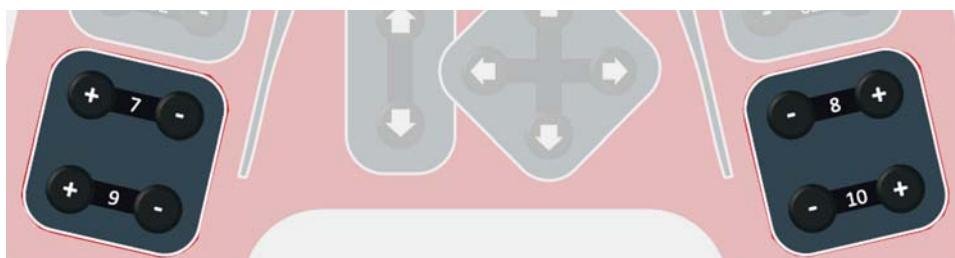
Pressing this key causes all the HOLDable programs and all the ARMs movements to be stopped. At the next pressure, the HOLD state is removed.

5.3.1.3.3 START (green)



in PROGR mode it is used to execute the current statement (from either edit/debug environment or EXECUTE command) for all the time it is kept pressed.
In LOCAL mode it runs the motion programs that are in READY state, waiting for this key to be pressed.

5.3.1.4 Keys to restart the Teach Pendant



If the Teach Pendant becomes locked in a condition from which autonomous re-enabling is not possible, or in any case a restart is necessary for other reasons, **simultaneously** press the four pushbuttons **7+, 8+, 9+ e 10+**.

5.3.2 Pushbuttons and LEDs

- Pushbuttons
- LEDs

5.3.2.1 Pushbuttons

- Stop pushbutton
- Enabling Device.

5.3.2.1.1 Stop pushbutton



On the front face of the Teach Pendant, at the top right position, there is the Stop pushbutton; the operating modes are as follows:

- activate by pressing
- release by screwing (clockwise).

5.3.2.1.2 Enabling Device



On the Teach Pendant rear there are two pushbuttons operating as the Enabling Device. They are inclined in respect to the Teach Pendant plane, in order to allow the thumb to

easily reach the front keys (see Fig. 5.2).

The right pushbutton and the left pushbutton operate in exactly the same way.

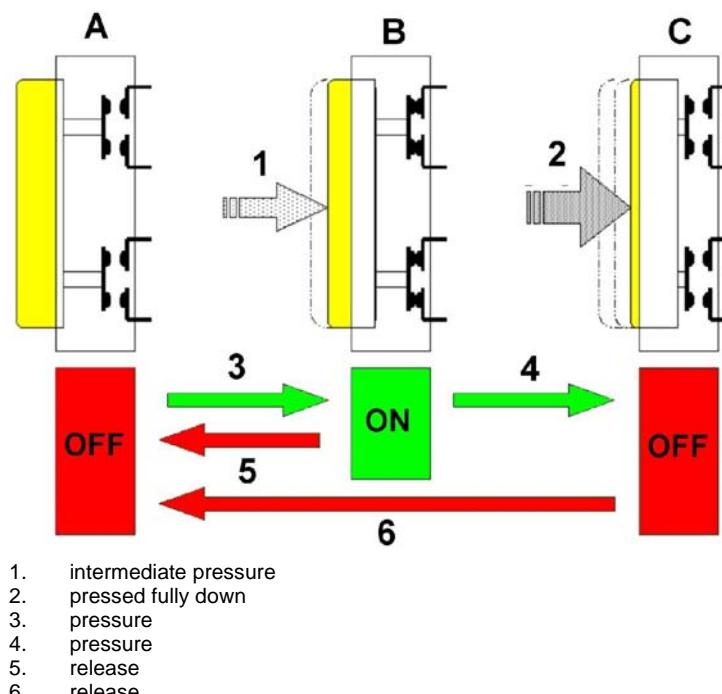
Each of these is a three-position safety device that is to be kept pressed in the intermediate position, to allow movements when the system is in Programming mode. When this pushbutton is pressed the motors are automatically activated (DRIVE ON).

The type of operation for each of them is as follows:

- released - **Drive OFF**
- intermediate pressure - **Drive ON**
- fully pressed - **Drive OFF** (anti-panic)
- pressing both these pushbuttons at the same time is interpreted as an error by the system, therefore only one at a time is to be used. It is anyway allowed to press them both at the same time, for a short predefined time interval, in order to be able to alternate hands.

A schematic diagram of the functioning is shown in Fig. 5.2.

Fig. 5.2 - Enabling Device operating mode



5.3.2.2 LEDs

The available LEDs on the Teach Pendant front are the following:



- Green LED
 - Red LED.
-

5.3.2.2.1 Green LED



When on, it means **DRIVE ON**.



As far as DRIVE ON and DRIVE OFF functionalities, please refer to [Enabling Device](#) and [Right Menu](#)

5.3.2.2.2 Red LED



This LED is associated to the [RESET \(white\)](#) key. When on, it means any type of **Alarm** (except Warning type indications).

5.4 USB port

In the front part of the Teach Pendant there is a USB port (vd.[Fig. 5.3](#)).

Fig. 5.3 - Porta USB



It is recommended to use the USB port to connect a storage device such as a Flash Disk USB one.



Only USB Flash Disk with a single FAT16 or FAT32 partition are supported. Multi-partitioned devices are not supported.

5.5 Display

Fig. 5.4 - Teach Pendant Display



The display of **TP5** Teach Pendant is a graphic 6.4" TFT colour display; resolution 640x480 pixel.

It is composed by:

- graphic colour display,
- analog resistive touch screen,
- 7" TFT wide screen,
- LED backlighting,
- resolution 800x480 pixel,
- WVGA.

After a few minutes that it is not used, the display switches off; to switch it on again just press any key.

5.5.1 Display areas

The Teach Pendant display can be considered as divided into the following areas:

- Status bar
- Right Menu
- Functional keys menu
- Pages Area.

5.5.1.1 Status bar



The Status bar is composed by

- Read-only fields and
- Fields for direct access to environments.

Read-only fields

- System state



Progr programming

T1P programming sub-state

Local local automatic

Remote remote automatic



- In PROGR and LOCAL states, an exclamation mark ‘!’ could be displayed preceding the status string. It indicates that some remote input signals (START, HOLD, DRIVE ON, DRIVE OFF) are disabled.
- When the system is in DRIVE ON state, “ON” is displayed too.
- State of Holdable programs and motion while programming



******** Indicates that no motion programs (holdable) are running

HOLD The HOLD pushbutton is pressed and therefore the execution of holdable programs and all arms movements are stopped; it is shown on an OCHRE background.

RUN At least one holdable program is running

- Motion current state



READY Everything is ready to move.

JOG A manual movement is in progress. OCHRE background.

FORW A FORWARD movement is in progress. OCHRE background.

BACK A BACKWARD movement is in progress. OCHRE background.



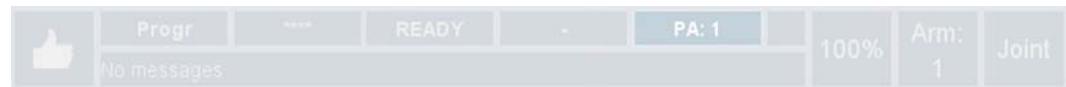
When the Enabling Device is pressed, “ED” is displayed too.
In error state, the background is RED.

- Arm state



- ‘-’ - arm properly configured and ready for moving
- ‘Turn’ - arm requires the turn-set operation
- ‘NotCal’ - not calibrated arm
- ‘Simu’ - simulated arm
- ‘Dis’ - disabled arm
- ‘Coop’ - arm-arm (‘An, Am’) or axis-arm (‘Ji, An’) cooperation
- ‘Pallet’ - active palletizing functionality
- ‘Res’ - arm needs to be resumed
- ‘Lock’ - locked arm.

– Total amount of currently active programs



– SHIFT key state



‘S’ is displayed when one of the **SHIFT** keys is pressed

– Error messages



This field displays the last occurred error/alarm message. The message includes the following information:

- signal code of 5 digits, which identifies each message in a unique way
- error/alarm severity level
- error/alarm message description string.

The background may be:

- transparent: no alarms/errors currently occurred;
- dark green: an INFORMATIONAL alarm is currently active. The system is NOT in ALARM state
- blue: WARNING message
- white: the text is black. This is a **Latched** type message, so the user must explicitly confirm it. To do so, from within the **Alarm Page**, remove the alarm cause and then press **Ack**
- ochre: this is an error generated by the application process. The system is not in ALARM state; the message disappears only when it is reset. To do so, from within the **Alarm Page**, remove the alarm cause and then choose one of the available commands in the **Functional keys menu**.
- red: this message is an ERROR message. The system is in ALARM state (an example is shown in the following figure).
- purple: this is a FATAL message. The system is in FATAL ERROR state.



For further details about the messages management, see the related [par. 5.13 Alarm Page on page 113](#)

- Arm

This field, second from right in the bar, indicates the current combination of Arm and SyncArm (if present).



In a **Multiarm** system with synchronised motion enabled, this field contains the number of the main Arm first, and then the one of the SyncArm; example:

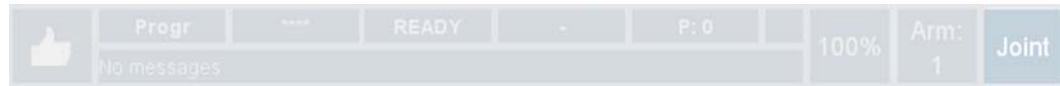
Arm:1<2 means **Arm 1** is the **main** Arm and **Arm 2** is the **synchronised** Arm.

For information regarding modification of the Arm indexes (main and synchronised) by means of ARM key, refer to [par. 5.3.1.2.3 ARM on page 58](#).

Touch this field to access the associated simplified environment in which the user can set the wished combination: [Arm Settings Page](#).

- Coord

This field, last field in the bar (see the following figure), displays the current modality used for the coordinates in Jog movements.



Touch this field to access the associated simplified environment in which the user can set the wished choice: [Arm Settings Page](#).

Fields for direct access to environments

- Alarm



Touch the leftmost field in the bar to directly access the [Alarm Page](#).

- GenOVR

This field, third from the right, displays the General Override current value for the main Arm, as well as enabled/disabled state for incremental movement. if this field displays the character 'I' (e.g.: '75%(I)'), it means that the system is in incremental movement mode. In this case the background color is ochre.



Touch this field to directly access the simplified page to handle such data: [Arm Override Page](#).

- Arm

This field, second from the right, indicates the currently selected combination between Arm and SyncArm if existing.



In case of **Multiarm** system, with enabled synchronized motion, this field first

displays the main Arm number and then the SyncArm one; example:

Arm:1<2 means that **Arm 1** is the **main** Arm and **Arm 2** is the **synchronized** Arm.

For further information about the Arm indexes modification (main and synchronized) by means of the ARM key, please refer to [par. 5.3.1.2.3 ARM on page 58](#).

Touching this field, the corresponding specified environment is accessed, in which it is allowed to set the wished combination: [Arm Settings Page](#).

- Coord

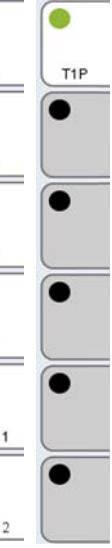
This field, last field on the right (see the following figure), displays the current modality to handle the Jog motion coordinates



Touching this field, the corresponding specified environment is accessed, in which it is allowed to set the wished combination: [Arm Settings Page](#).

5.5.1.2 Right Menu

This menu is dedicated to technological commands such as grippers opening or closings, welding parameters hardware settings and managing Technological Process status in Programming mode (**T1P** substate)

Local	Remote	Progr	Description
			<p>The meaning of the keys is different depending on the system state.</p> <ul style="list-style-type: none"> - in Local or Remote state, the 5th key is always dedicated to Drives on/off according to the following rules: <ul style="list-style-type: none"> • in Local state it represents DRIVE ON and DRIVE OFF commands; the corresponding LED is green when drives are ON; • in Remote state it represents DRIVE OFF command (DRIVE ON command comes from PLC); the corresponding LED is green when drives are ON. - In Progr state <ul style="list-style-type: none"> • U1..U4 keys are available to the user. • The 5th and 6th keys are dedicated to Hand1 and Hand2 functions for controlling tools 1 and 2, switching between OPEN and CLOSE. • The 7th key, T1P (available by pressing MORE key), acts on the T1P substate, according to well defined rules and limits, described in par. T1P Status on page 51.

Pressing one key causes the corresponding command to be executed and its state to be displayed (green led = active command; black led = NOT active command).



If the user wishes to create a customized Right Menu, it is necessary to write a PDL2 program as described in [par. 5.22.2 Handling TP right menu on page 283](#).

5.5.1.3 Functional keys menu

This menu is a group of 8 function keys, contextual to the selected item (current state and page, selected field, etc.). The shown functions correspond to the possible actions on the selected item.

In the figure below, the functional keys menu available for the [Start Page](#) is shown as an example.



Generally, when touching a functional key, the associated function is activated.

Anyway, some keys exist which are associated to a pull-up submenu: this is indicated by means of a small triangle in the topmost right angle of the key (e.g. the **Restart** key in the [Start Page](#) Functional keys menu).

Either use the touchscreen or move in the submenu by means of the cursor keys, then press **ENTER** to confirm, in order to make the wished choice.

A key can also be associated to a functionality with some optional settings: this kind of elements are identified by a small icon representing a clock, in the topmost right angle of the key (e.g. the **Login** key in the [Start Page](#) Functional keys menu).

A short touch on such a key, causes the function execution according to a default choice; a long touch opens a pull-up submenu for choosing the wished option.

The pull-up submenu is closed as soon as either a choice is made, or the **ESC** key is pressed (this deletes any action).

The meaning of the function keys is described in the corresponding [User Interface Pages](#).

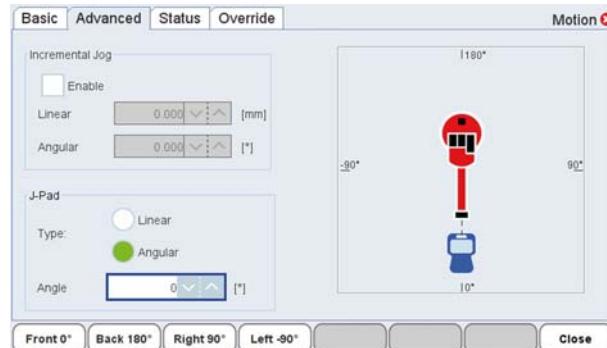
5.5.1.4 Pages Area

This is the part of the display for viewing the [User Interface Pages](#).

The name of the ACTIVE Page is written in the topmost right corner of the screen, besides the symbol for closing the page.

Each Page refers to specific functions in a precise ENVIRONMENT (for example Motion environment, I/O environment, alarms environment, etc.).

Fig. 5.5 - Page example





For a detailed description of the available pages and for any information related to their use, as well as the general internal navigation rules, see [User Interface Pages](#) section.

5.6 User Interface Pages

The purpose of this section is to give a detailed description of the pre-defined Pages available for the user, and how to use them in the User Interface.

In particular, information is given about:

- General information
- Field types
- Keyboards
- Connection Page
- Dialogue and information window
- Start Page
- QUICK Page
- MENU Page
- Motion Page
- Arm Override Page
- Arm Settings Page
- Alarm Page
- Prog Page
- IDE Page
- Display Page
- Appl Page
- Files Page
- Data Page
- Setup page
- Service Page
- Customizations on the TP.

5.6.1 General information

All the User Pages have some common features:

- the current page name is displayed in the topmost right corner of the screen



- to close the current page, the choice is between
 - touching symbol ‘x’ (white with red background) in the topmost right corner, besides the page name (see the above figure)
 - press **ESC** key
 - touch **Close** key
- in all the pages the user can navigate both by means of the touchscreen and the use of cursor keys plus **ENTER** and **ESC** keys available in the **Membrane Keyboard**. Special use are described in the specific environments paragraphs.
- The editing modalities of each field are different depending on the **Field types**. If the selected field is editable (white background), the corresponding edit tool is automatically opened. **TP5 Teach Pendant** provides some **Keyboards** of various type and functionality, in order to operate on each field type.

5.6.2 Field types

The available field types are:

- **Numeric field**
- **Textbox**
- **Combobox**
- **Combobutton**
- **Checkbox**
- **Checkgroup**
- **Label**
- **List**
- **Cell of a table**
- **TreeGrid**.

5.6.2.1 Numeric field



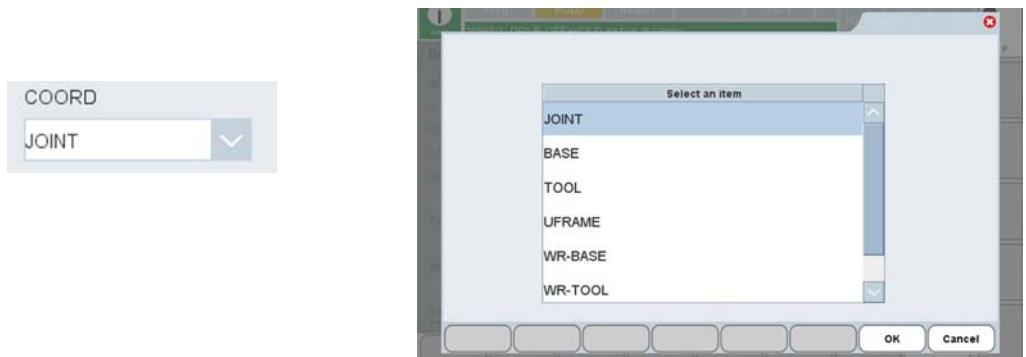
When this field type is selected, the system automatically opens the **Numeric keyboard**.

5.6.2.2 Texbox



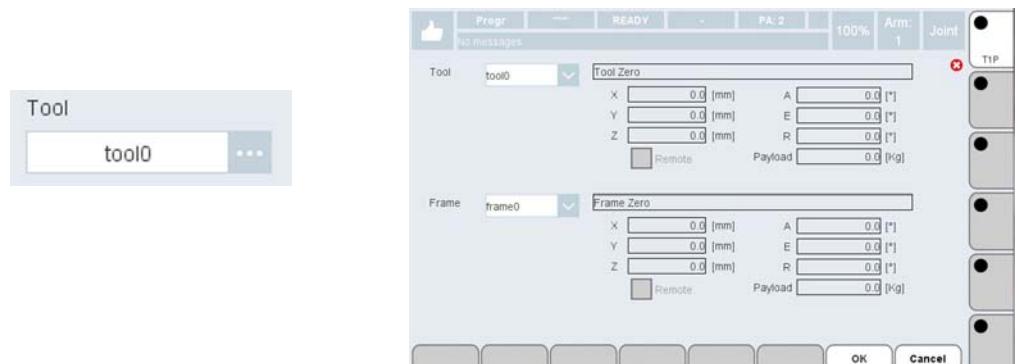
When this field type is selected, the system automatically opens the [Alphanumeric keyboard](#), unless the field includes or is intended to include a PDL2 statement: in such a case, the [PDL2 keyboard](#) is automatically opened.

5.6.2.3 Combobox



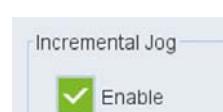
When this field type is selected, a screen is made available (see the above figure on the right) including a list of items among which the user can choose the wished one. Touch such an item and press **OK** to confirm.

5.6.2.4 Combobutton



It allows to open a further page (see the above figure on the right). The displayed text string (above figure on the left) depends on the current settings contained in such a page.

5.6.2.5 Checkbox



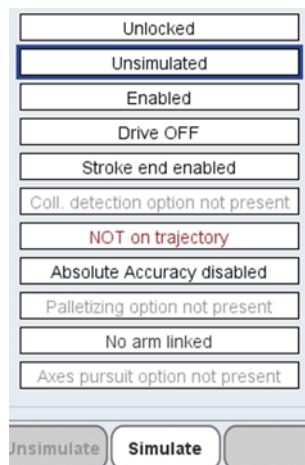
To switch the value in this field, just touch it.

5.6.2.6 Checkgroup



This field type is similar to the previous one ([Checkbox](#)); the only difference is that there are two or more “buttons” and just one at a time can be activated.

5.6.2.7 Label



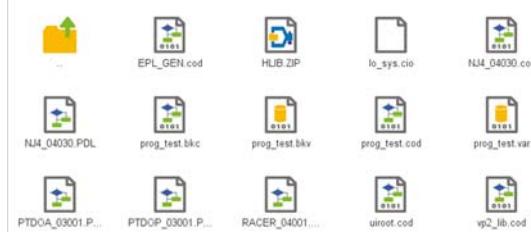
This field type can only be modified by means of the available key displayed in the [Functional keys menu](#), associated to the selected field.

When the background is GREY, the corresponding field can **not** be modified.

5.6.2.8 List

Lists can be displayed in the following ways:

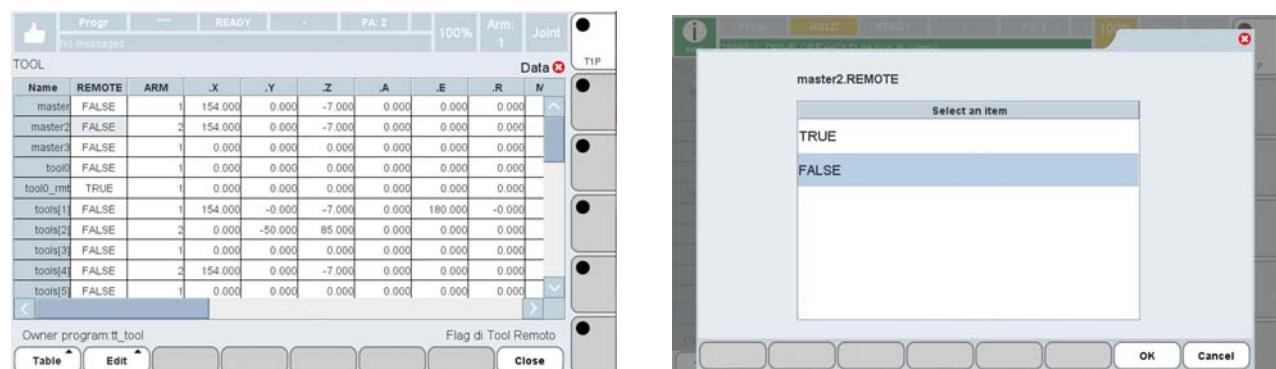
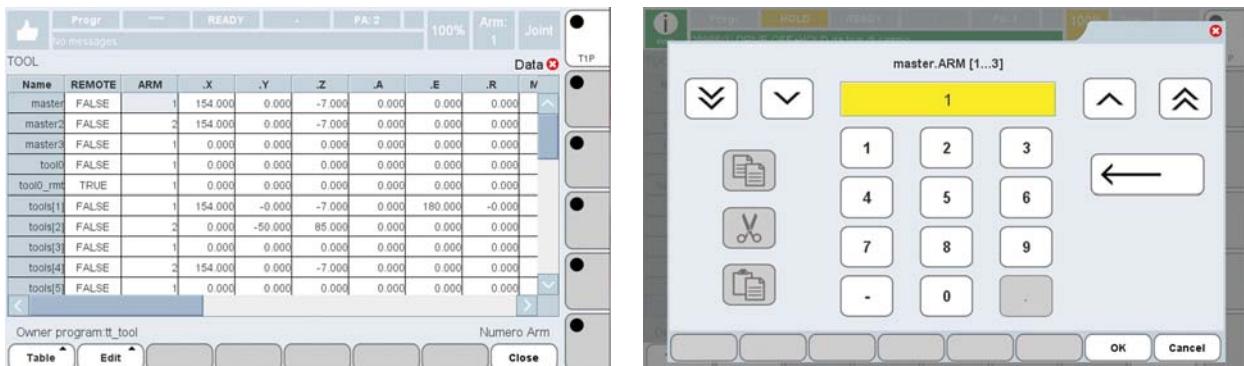
- large icons - displays the information with a large icon
- details - besides the name and the icon of each item in the list, further information is given (for example the size of the files and the date of the last change are listed).
- sort by
 - name
 - date
 - type
 - dimension.

large icons	details																								
	<table border="1"> <thead> <tr> <th>Name</th> <th>Size</th> <th>Time</th> </tr> </thead> <tbody> <tr> <td>disp_3.xml</td> <td>661</td> <td>2014-03-24 16:10:52</td> </tr> <tr> <td>disp_38549.xml</td> <td>1759</td> <td>2014-03-25 10:44:28</td> </tr> <tr> <td>double_var.xml</td> <td>3438</td> <td>2014-03-24 17:36:50</td> </tr> <tr> <td>example_1.xml</td> <td>600</td> <td>2014-03-25 16:42:24</td> </tr> <tr> <td>my_disp.xml</td> <td>1629</td> <td>2014-03-24 12:10:14</td> </tr> <tr> <td>new_disp.xml</td> <td>13298</td> <td>2014-03-24 15:42:54</td> </tr> <tr> <td>new_screen.xml</td> <td>1177</td> <td>2014-03-26 17:15:20</td> </tr> </tbody> </table>	Name	Size	Time	disp_3.xml	661	2014-03-24 16:10:52	disp_38549.xml	1759	2014-03-25 10:44:28	double_var.xml	3438	2014-03-24 17:36:50	example_1.xml	600	2014-03-25 16:42:24	my_disp.xml	1629	2014-03-24 12:10:14	new_disp.xml	13298	2014-03-24 15:42:54	new_screen.xml	1177	2014-03-26 17:15:20
Name	Size	Time																							
disp_3.xml	661	2014-03-24 16:10:52																							
disp_38549.xml	1759	2014-03-25 10:44:28																							
double_var.xml	3438	2014-03-24 17:36:50																							
example_1.xml	600	2014-03-25 16:42:24																							
my_disp.xml	1629	2014-03-24 12:10:14																							
new_disp.xml	13298	2014-03-24 15:42:54																							
new_screen.xml	1177	2014-03-26 17:15:20																							

Touch the wished list element, to select it. For multiple selection of the list elements, touch the first element of the selection and then the last element together with the **SHIFT** key.

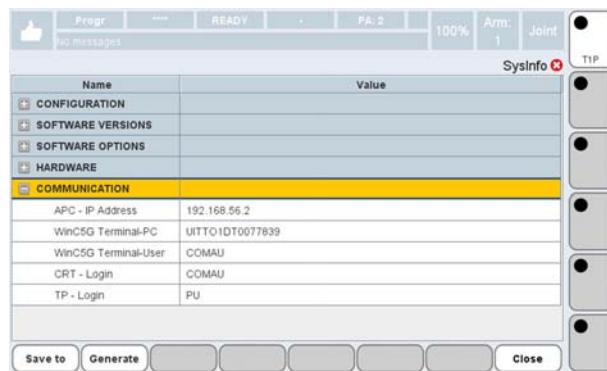
5.6.2.9 Cell of a table

Touch twice to open the associated edit tool (either the suitable keyboard or a list of items among which choosing the wished one), to edit a cell of a table.



5.6.2.10 TreeGrid

It is a special table structure displaying information in rows which are connected by a hierarchical relationship, like a tree (see the figure below). Touch one row of the table to open or close the “children” list of such an element.



5.6.3 Keyboards

Some general rules exist to be taken in account when using a keyboard:

- **Cursor keys (up/down/right/left)** can be used to move among the pages fields, instead of touch keys
- **SHIFT** keys are always to be combined together with other keys; its use depends on the environment and the functionality of the key it is associated to
- all keys existing on the touchscreen, i.e. “touchable” keys, activate the associated functions just tapping them
- simple tap, double tap and swipe are available to select/modify a text. Further details are provided in the following sections.

Available keyboards on **TP5** Teach Pendant are:

- **Membrane keyboard**
always available (for the functionalities related to this kind of keyboard, please refer to [par. 5.3.1.1 Blue keys on page 53](#))

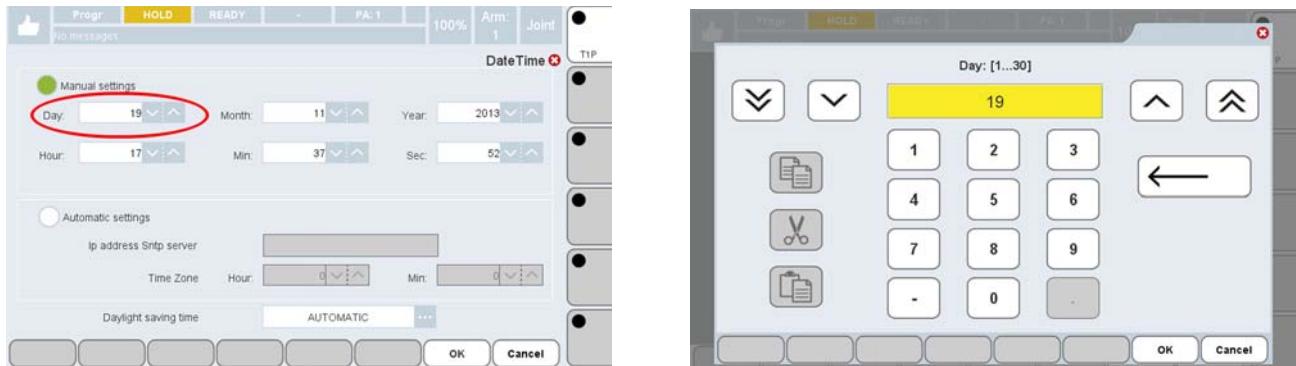
As far as these keys usage while editing values and texts, to substitute or help using the **Touchable keyboards**, please refer to the specific environments (see [par. 5.6.3.1 Numeric keyboard on page 76](#) and [par. 5.6.3.2 Alphanumeric keyboard on page 78](#)).

- **Touchable keyboards**
available on the **Display**, depending on the context.
They are classified as:
 - [Functional keys menu](#)
 - [Numeric keyboard](#)
 - [Alphanumeric keyboard](#)
 - [PDL2 keyboard](#).

5.6.3.1 Numeric keyboard

This keyboard is made available by the system, whenever editing a numeric field is required.

Fig. 5.6 - Numeric keyboard

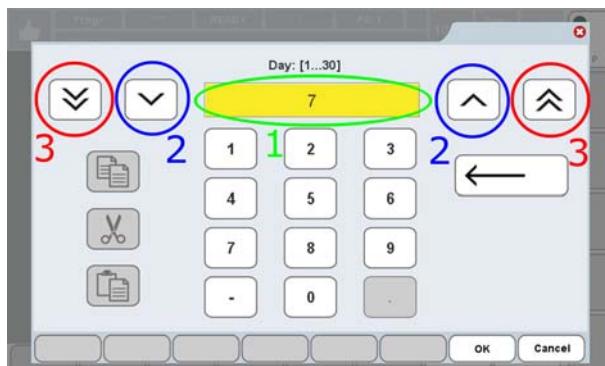


In the example shown in **Fig. 5.6**, the numeric keyboard (figure on the right) is activated by tapping the **Day** field (highlighted in red in the figure on the left - previous figures), from **DateTime** page.

The value in the selected field can be modified by means of the following functionalities, available from both the touchable keyboard and the **Membrane keyboard**:

- direct type the new value (**1** - highlighted in green in **Fig. 5.7**)
- decrement/increment (**2** - highlighted in blue in **Fig. 5.7**)
- big decrement/big increment (**3** - highlighted in red in **Fig. 5.7**)
- up cursor - increment
- down cursor - decrement
- right cursor - move the cursor of one character right
- left cursor - move the cursor of one character left
- SHIFT + up cursor - big increment
- SHIFT + down cursor - big decrement
- SHIFT + right cursor - select/unselect (depending on the current state) the first digit on the right of the cursor
- SHIFT + left cursor - select/unselect (depending on the current state) the first digit on the left of the cursor.

Fig. 5.7 - Editing a numeric value (touchable keyboard)



Press **OK** to confirm the new value. Press **Cancel** to exit without modifying.

5.6.3.2 Alphanumeric keyboard

Fig. 5.8 - Alphanumeric keyboard



This keyboard is made available by the system, whenever editing an alphanumeric field is required.

In the example of Fig. 5.8, the alphanumeric keyboard (on the right) is activated by tapping on the **Custom ID** field (highlighted in red in the figure on the left).

The value in the selected field can be modified tapping the keyboard touchkeys.

Press **OK** to confirm the new value. Press **Cancel** to exit without modifying.

Several options exist, available both from the touchable keyboard and the [Membrane keyboard](#), to insert/edit an alphanumeric text string:

- capability of choosing among the 15 last used items.

When tapping the highlighted in red key, in the left figure below, the history of the used items (up to 15) in the alphanumeric keyboard, is made available as shown in the right figure below.

The user is allowed to choose one of them to be inserted in the current field; then either tap **OK** or press **ENTER** to confirm.



- standard use of the Copy, Cut and Paste touch keys (highlighted in red in the following figure), for the currently selected part of text (e.g. 'SIM' text string)



- tap - select the tapped word
- double tap - select the whole text
- swipe - select the characters string between the first touched character and the last touched character, continuously swiping (never detach the finger)
- up cursor - move the cursor to the text beginning
- down cursor - move the cursor to the text end
- right cursor - move the cursor of one character right
- left cursor - move the cursor of one character left
- SHIFT + up cursor - select all, from the cursor current position to the text beginning
- SHIFT + down cursor - select all, from the cursor current position to the text end
- SHIFT + right cursor - select/deselect (depending on the current state) the first character on the right of the cursor
- SHIFT + left cursor - select/deselect (depending on the current state) the first character on the left of the cursor.

5.6.3.3 PDL2 keyboard

This keyboard is made available by the system, whenever editing a PDL2 statement is required.

Fig. 5.9 - PDL2 keyboard



Typically, this happens in IDE environment but also in other contexts, e.g. EXECUTE command or while creating a PDL2 macro in the [QUICK Page](#).

In the shown example (see Fig. 5.9), touching the command or the field for inserting the instruction, the **PDL2 keyboard** is opened by the system.

A statement can be directly typed in by means of the keyboard: the editing modality for text strings is exactly THE SAME as described for the [Alphanumeric keyboard](#).

The significant difference between the PDL2 keyboard and the Alphanumeric one consists of

- [dedicated touchable keys](#) (available in the Functional keys menu):
 - [Statement](#)
 - [Variables](#)
 - [Values](#)
 - [dedicated History function](#).
-

5.6.3.3.1 Statement

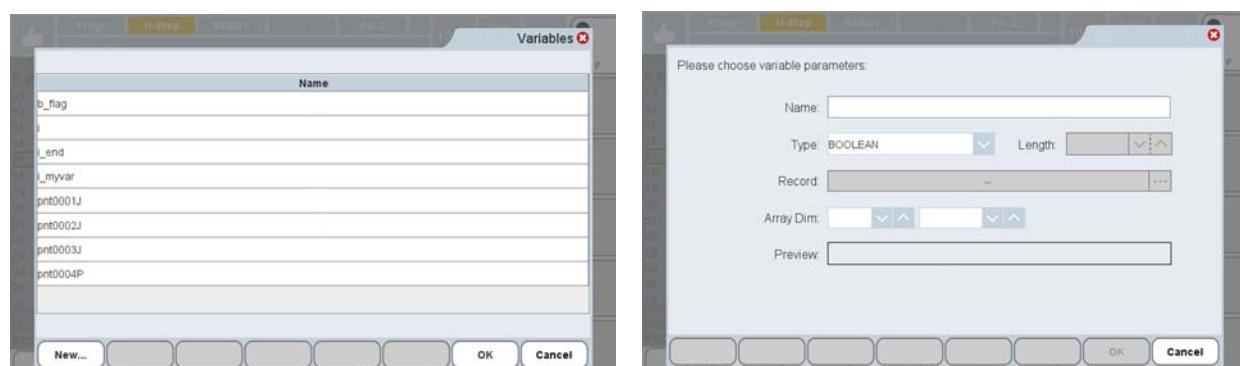
This functional key opens the menu containing the available PDL statements, among which the user can choose the wished one (see figure below, on the left).

Touch an icon (e.g. Flow) to cause the statements belonging to such a group to be displayed (see figure below, on the right); then touch the chosen statement twice or touch it once and then press **ENTER**, to finally insert the statement.



5.6.3.3.2 Variables

It opens the environment for viewing all existing variables (following figure on the left) and creating new ones (following figure on the right).



Touch the **New...** command to create a new variable. Then the dedicated page is opened (previous figure, on the right).

5.6.3.3.3 Values

This key is available just if the currently selected text is a PDL2 reserved word

representing a value; it allows accessing all the available values of such a type and selecting the wished one.

In the following example, selecting the reserved word '**TRUE**' activates the **Values** key; the pull up menu displays all the available choices for such a type. Touch the wished value to insert it into the statement.



5.6.3.3.4 History

Touch the symbol highlighted in red in the following figure (on the left), to open the History page associated to the PDL2 keyboard. It includes the following information:

- last 15 used STATEMENTS (Recent section)



- most used DEFAULT values (**PDL Default** section)
- most frequently used components in PDL2 programs: variables names, constants names, routines names, statements, etc. (**EXE Help** section).



Touch the wished element to select it, then either touch **OK** or press **ENTER** to confirm.

Such an item is inserted in the current program line.

5.6.4 Connection Page

The Connection Page is the environment in which all information is displayed, about the state of the Teach Pendant when trying to connect to the Control Unit.



For further details about the system behaviour in such situations, as well as about displayed messages with different severity, please refer to the [CONNECTION FAILURE](#) note, in [par. 3.3.1 STARTUP Procedure on page 39](#).

5.6.5 Dialogue and information window

When needed, the system opens a dedicated page in which the user is asked to confirm an operation or to enter a password, or to give information about the result of the required command.

Therefore, some suitable windows are available, having different background colour according to the message enclosed in the window:

- blue - request for confirmation or information by the system

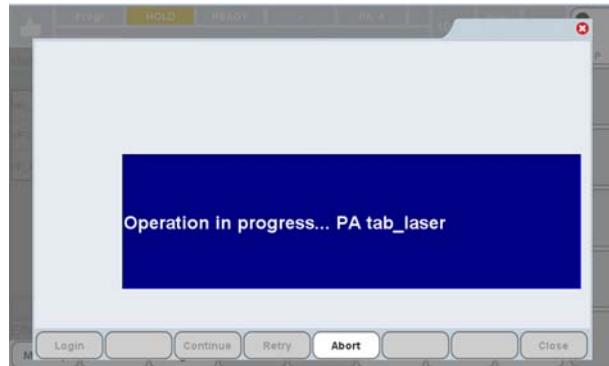


If the user does not wish to answer back to the system request, He/She should press the **ESC** key.

- yellow - system warning message
- red - command failure due to a serious error.

When the user asks for the execution of a certain command, the system opens a warning page containing “Operation in progress...” message and keep it displayed until the execution is finished.

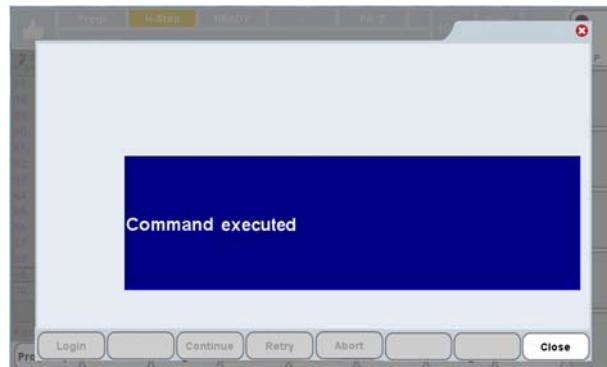
While running, the **Abort** command is made available to be able to permanently interrupt the command (see the figure below).



In most cases the execution is so fast that the blue page isn't even seen by the user.

According to the command execution status, the following situations may occur:

- the command has been successfully completed. The system could
 - no message displayed because this status is already obvious;
 - message displayed (blue background) indicating that the command has been completed; the **Close** key is available to exit from the command;



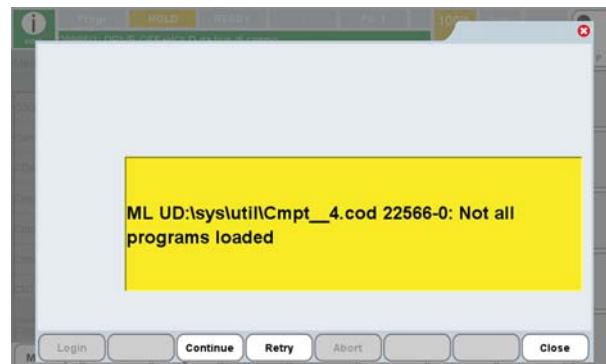
- message displayed (yellow background) indicating the completion of the command and the number of operations that have been skipped (by means of the **Continue** key); the **Close** key is available to exit from the command



- It has not been possible to execute the command - the system displays a message with a yellow background, to explain to the user the problems found. The following softkeys are available:
 - **Login** - if the error message is "Insufficient Rights", this means that the Login operation has not been performed yet. The user has to press **Login** key to do so. After the login operation is accomplished, the system will automatically execute the previously requested command again.



- It has not been possible to execute the command - the system displays a message with a yellow background, to explain to the user the problems found. The following softkeys are available:
 - **Continue** - if the command consists of a sequence of several operations, it may happen that one or more cannot be executed. In this case the system stops the command execution and informs the user: if the user wishes to skip the current operation and go on executing the following one, the **Continue** key must be pressed.



- **Retry** - if the system stops the command execution for any reason, the user can remove the problem and then press this key: the command will be run again.
- **Close** - exits from the current command. In case of sequence of operations, it does not run the subsequent commands and quits.

5.7 Start Page

Fig. 5.10 - Start Page



The Teach Pendant Home Page is called **Start Page**.

It includes:

- Basic information
- Commands.

5.7.1 Basic information

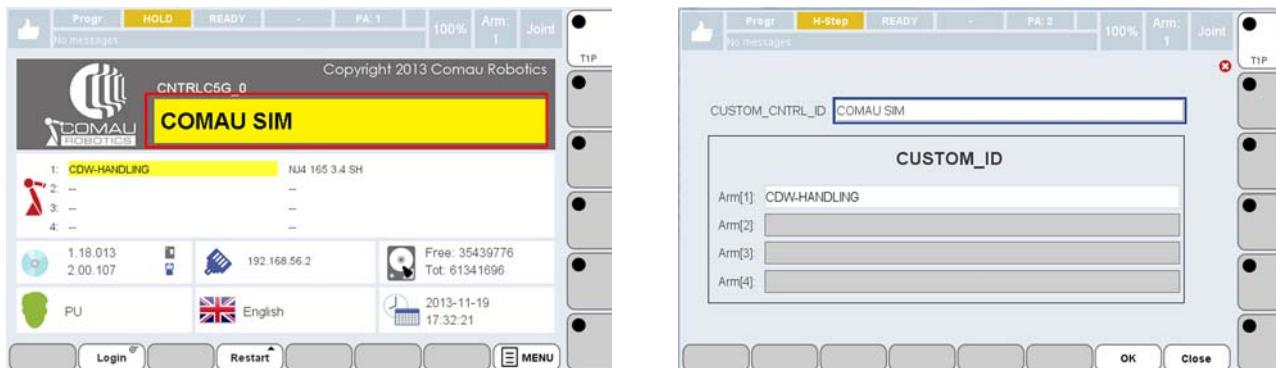
- Controller name
- Arms name and type
- Software version
- Network address
- Memory space
- Current Login
- Language selection
- Date and time

5.7.1.1 Controller name

It is the name chosen by the user and included into \$CUSTOM_CNTRL_ID predefined variable.

Touching this field (see following figures), this information can be inserted/modified by the User in the [Customer](#) subpage of the [Setup page](#).

Note that, if the Controller name has not been inserted yet, '\$CUSTOM_CNTRL_ID' string is displayed.



5.7.1.2 Arms name and type

These fields show the following choices made by the user:

- Arm name, CUSTOM_ID (field of \$ARM_DATA predefined variable), e.g. ‘Rob_1’, ‘Racer’, etc; like for the [Controller name](#), if the Arm name has not been inserted, the following string is displayed: ‘CUSTOM_ID’. This value can be inserted/modified by the User in the [Customer](#) subpage of the [Setup page](#), simply touching the [Controller name](#) field (highlighted in red in the above figure on the left);
- Type of associated Arm for **each of the 4 allowed Arms** (content of \$RB_NAME predefined variable); e.g. ‘NJ4 165 3.4 SH’.

5.7.1.3 Software version

This field displays two information:

- Controller software
- Teach Pendant software

they are displayed on two lines, each of them associated to the corresponding icon: **C5G** and **TP5**.

Touch this field to directly access the [Info](#) subpage in which System information is displayed (as shown in the below figure, on the right)



5.7.1.4 Network address

This field displays the address of the Controller on the factory network. Touch it to directly access the [Network](#) environment in the [Setup page](#) (see the following figures).



5.7.1.5 Memory space

This field displays two information about the memory space:

- currently free space, with realtime refresh;
- total space.



5.7.1.6 Current Login

This field displays the currently active login.



Touch it to directly open the system access environment ([Login and Logout](#)).

5.7.1.7 Language selection

This field displays the currently selected language (highlighted in red in the following figure, on the left) for the User Interface.



Touch it to access all the available languages:

- Chinese
- Czech
- Deutsch
- English (UK)
- English (US)
- Espanol
- Français
- Italiano
- Polish
- Portugues (for Brazil)
- Portogues (for Portugal)
- Russian
- Turkish.

Touch the an icon to select the corresponding language.

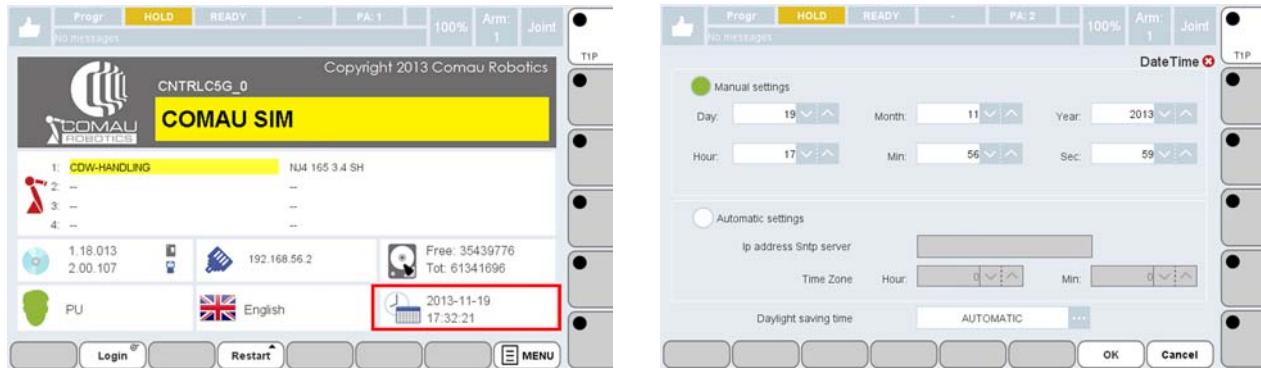


The difference between British English and American English is just in the format used for the date and the time.

5.7.1.8 Date and time

This field displays the current date and time, according to the specified time zone and the Daylight Saving Time functionality.

Touch this field to access the [Date-Time](#) subpage in the [Setup page](#) (as shown in the following figure).



5.7.2 Commands

The available functions for this page, in the [Functional keys menu](#), are as follows:

- [Login and Logout](#)
- [Restart](#)
- [MENU.](#)

5.7.2.1 Login and Logout



This command, depending on how the associated key is touched, allows to perform the following operations:

- [Login](#),
- [Logout](#).

5.7.2.1.1 Login

Short touch allows accessing the system by authorized personnel; the system asks for username and password for executing the wished login type (see the following figures).

Specify the needed data and either touch **OK** or press **ENTER**, to confirm.



For further information about LOGIN, refer to [Cap.4. - Access to the Control \(login/logout\) on page 43](#).

For user settings, open the [Login](#) subpage in the [Setup](#) page.

5.7.2.1.2 Logout

Long touch makes available the associated command to activate the Logout procedure (see the figure below, on the left).

A message informs the user when the logout operation has been successfully completed (see the figure below, on the right).



A specific Automatic Logout function is also available (see [par. 4.3.1 Automatic Timed Logout on page 48](#)).



5.7.2.2 Restart

Touching this key, a pull-up menu is opened as shown in the following figure.



The following commands are available:

- [Cold](#)
- [Shutdown](#)

Touch the wished item to choose.

5.7.2.2.1 Cold

This command performs the Control Unit complete Restart.



As far as the startup procedure, refer to [par. 3.3.1 STARTUP Procedure on page 39](#).

When selected, the system prompts the user for confirmation, before running the Restart Cold procedure (see the figure below, on the left)



Touch **OK** to confirm or touch **Cancel** to exit.

The acknowledgement of the command starts a count down (see above figure, on the right) which ends when both Controller and Teach Pendant restart.

In case of mismatch between memory content and disk content, a message is issued to prompt the user for continuing the Restart operation.



For further information about the Restart Cold operation, refer to [Cap. System Commands on page 295](#).

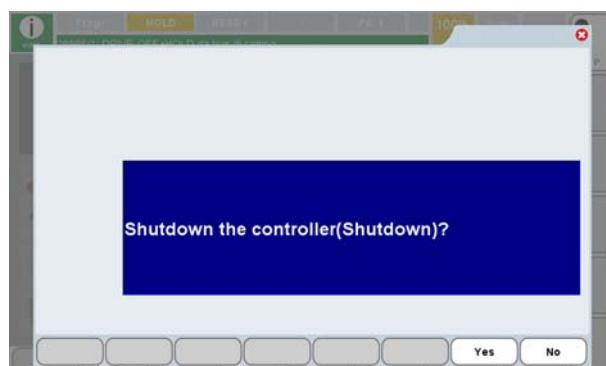


Note that, at the end of the Restart operation, the current directory is
UD:\USR

5.7.2.2.2 Shutdown

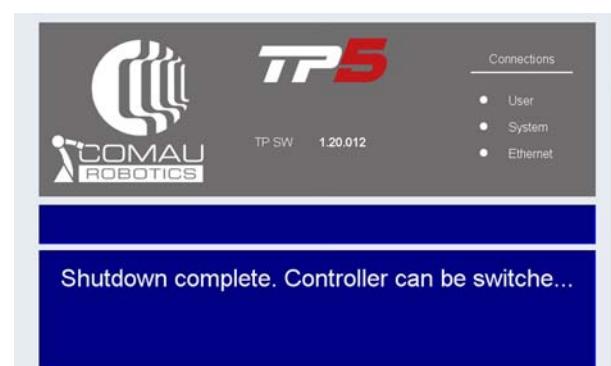
This command performs the Control Unit shutdown.

When selected, the system prompts the user for confirmation, before running the Shutdown procedure (see the figure below)



Touch **OK** to confirm or touch **Cancel** to exit.

The acknowledgement of the command starts a count down (see figure below on the left) which ends with a message suggesting the user to switch off the system (see figure below on the right).



5.7.2.3 MENU

Touch this key to access the [MENU Page](#), as shown in the below figures. It has the same functionality as the [MENU](#) key belonging to the [membrane keyboard](#) (see [par. 5.3.1.1.1 Special keys on page 53](#)).



For further details about such a page, refer to [par. 5.9 MENU Page on page 97](#).

5.8 QUICK Page

5.8.1 Overview

The **QUICK Page** allows accessing an environment for the use of “short commands” created by the user.

More in details, this page allows [Viewing/Creating/Modifying](#) such commands.

A “short command”, from now on referred to as “**macro**”, includes one or more repetitive or frequently used operations, associated to a name chosen by the user.

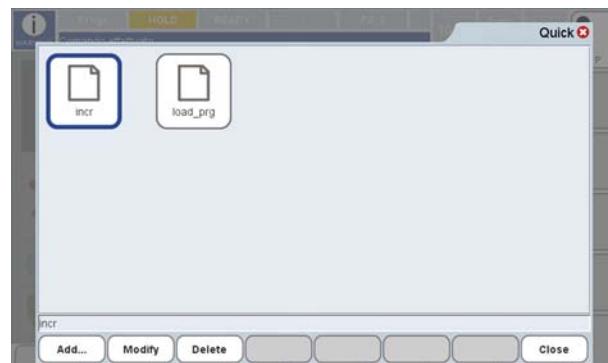
Press **QUICK** special key, belonging to the membrane keyboard, to access this page, as shown in the below figure.



5.8.2 Viewing

The already existing macros are displayed when accessing to the **QUICK Page**.

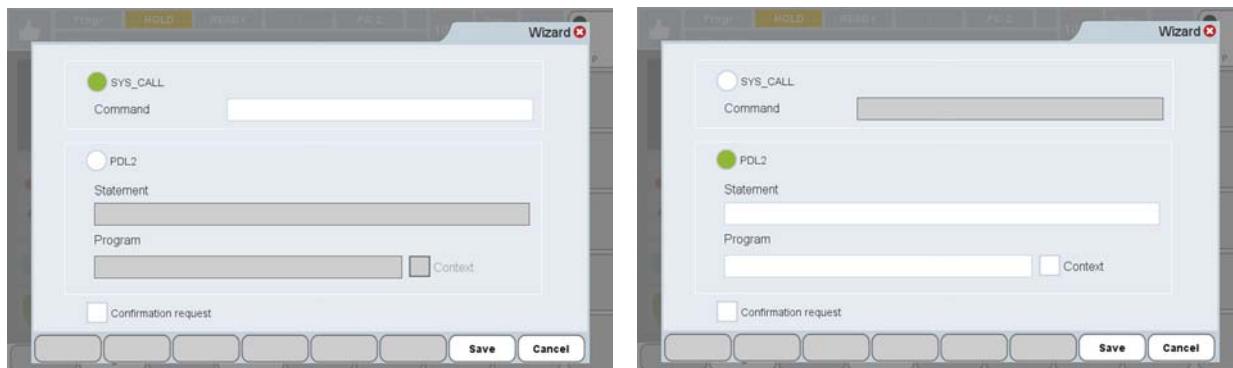
Fig. 5.11 - QUICK Page



In the shown above example, two macros are already present, referred to as *incr* and *load_prg*. To use them, just tap the corresponding icon.

5.8.3 Creating/Modifying

Tap **Add...** key to create a new macro; the wizard is activated, in which the user must choose whether it will include commands or PDL2 instructions.



To do it, the following options are available:

- **SYS_CALL**
- **PDL2**.

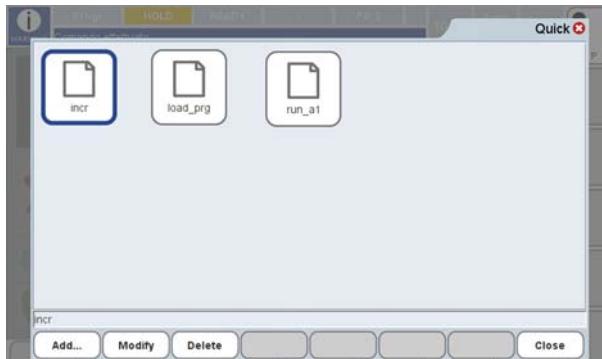
Please note that to modify an already existing macro, **Modify** function is available (see Fig. 5.11) which operates like **Add...** function which creates a new one.

5.8.3.1 **SYS_CALL**

It is to be used for macro including commands executed by means of **SYS_CALL** built-in routine.

Touching **Command** field, the **Alphanumeric keyboard** is displayed. Insert the wished command and either tap **OK** or press **ENTER** to confirm.

Tap **Save** key in the Wizard page, insert the name to be assigned to the newly created macro and confirm. In the example, shown in the following figure, the chosen name is *run_a1*.



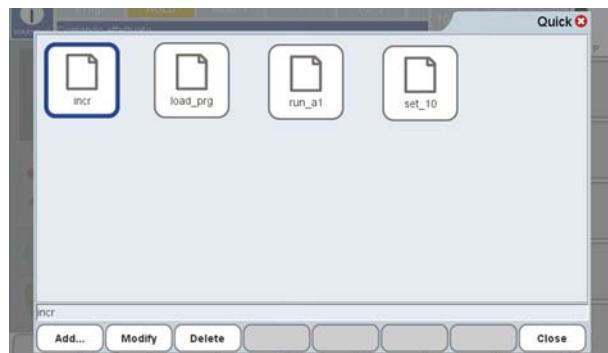
5.8.3.2 **PDL2**

This choice is to be used for creating macros which immediately execute PDL2 statements; thus, **EXECUTE** command rules have to be complied with.

Tapping **Statement** field, the **PDL2 keyboard** is displayed in order to allow the user to insert the wished PDL2 instruction.

Tap **OK** key or press **ENTER** to confirm.

Tap **Save** key in the Wizard page, insert the name to be assigned to the newly created macro and confirm. In the example, shown in the following figure, the chosen name is *set_10*.



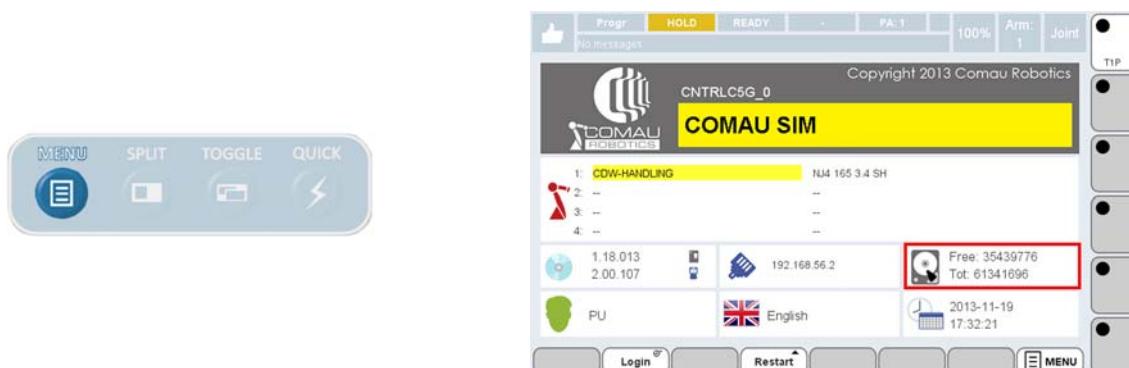
When executing a sequence of operations is needed, the user can create a routine containing them and then create a macro calling such a routine.

5.9 MENU Page

5.9.1 Overview

The **MENU Page** allows accessing to all the [User Interface Pages](#).

To enter this page, either press **MENU** special key belonging to the membrane keyboard, or touch **MENU** functional key in the [Start Page](#), as shown in the below figures.

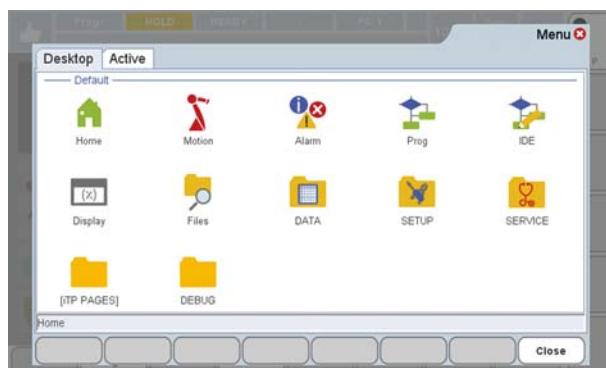


Two panels are available:

- [Desktop](#)
- [Active](#).

5.9.2 Desktop

Fig. 5.12 - MENU Page - Desktop Panel



This page includes all the icons allowing access to the [User Interface Pages](#). Either touch the corresponding icon to activate the wished environment, or use the cursor keys and then press **ENTER** to confirm.

Home icon returns back to the [Start Page](#). The same result can be obtained by closing the page (**Close** key or close page symbol in the topmost right corner).

In the [Functional keys menu](#) the **Property** key allows displaying the *Property notes* corresponding to the currently selected object.



As far as how to handle the [User Interface Pages](#), please refer to the specific chapters.

5.9.3 Active

Fig. 5.13 - MENU Page - Active Panel



This panel displays the currently active pages icons. It is always available when moving among the folders.

It is not available when inside an environment.

In Fig. 5.13 example, the currently active pages are the [Motion Page](#) and the [Prog Page](#), together with the [Start Page](#) which is always active, anyway.

5.10 Motion Page

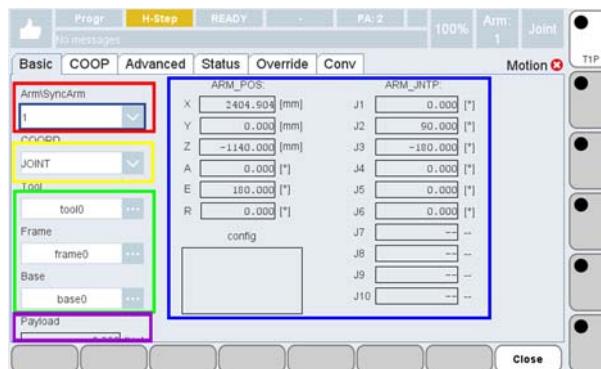
This User Page manages information regarding the robot motion environment and allows the user to display and change it.

It is divided into sub-pages, that can be selected through the corresponding labels

- Basic
- Coop (optional feature)
- Advanced
- Status
- Override
- Conveyor Tracking (optional feature).

5.10.1 Basic

Fig. 5.14 - Basic sub-page



The Basic sub-page contains the following fields:

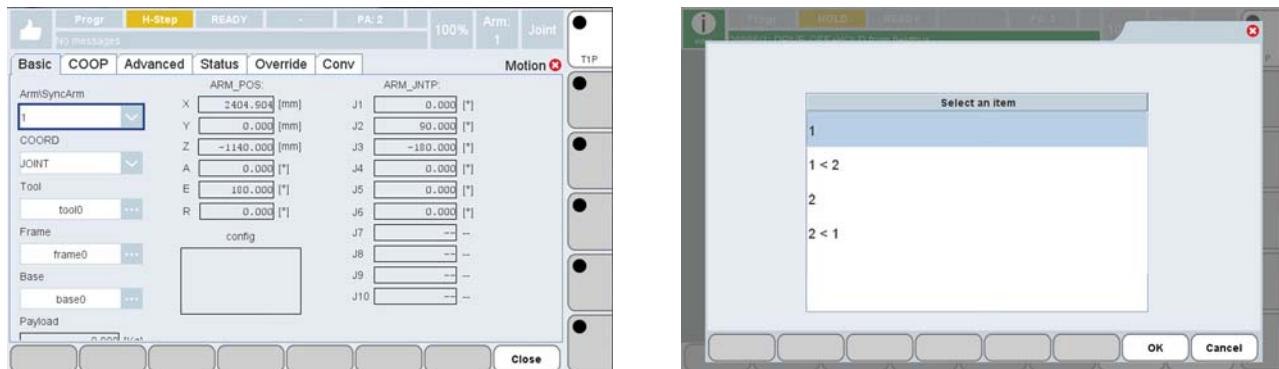
- **Arm\SyncArm** (red)
- **COORD**: Current coordinates system (yellow)
- **Tool, Frame, Base**: Current reference systems (green)
- **Current arm position** (blue)
- **Payload** (purple).

A detailed description of all fields follows.

5.10.1.1 Arm\SyncArm

It is a menu showing the Arm/Synchronized Arm combinations currently existing in the system. The User must open the menu and select the wished item.

Touch the field and make the choice by touching the wished item in the displayed list (see Fig. 5.15 - on the right).

Fig. 5.15 - Choosing Arm\SyncArm

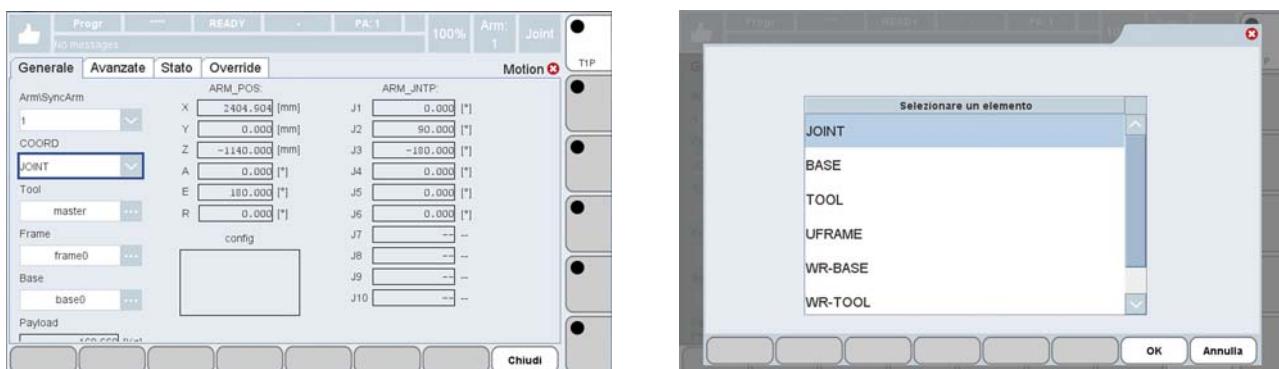
It is also allowed modifying this configuration from within the **ArmSettings** simplified environment, accessible touching **Arm** field in the **Status** bar.

5.10.1.2 COORD

This field indicates the current coordinates system according to which the arm moves will be made, in manual movement and in programming. The user can choose a coordinates system that is different to the current one, selecting from the following (as shown in Fig. 5.16):

- for **TP JOG** mode:
 - JOINT
 - BASE
 - TOOL
 - UFRAME
- for the **WRIST** mode (movement carried out to the robot wrist):
 - WR-BASE
 - WR-TOOL
 - WR-UFRAME

To make the choice, a menu is available to be opened by touching **COORD** field (as shown in Fig. 5.16 on the left). Touch the wished item in the displayed list (see Fig. 5.16 on the right).

Fig. 5.16 - Choosing the coordinates system

It is also allowed modifying this configuration from within the **ArmSettings** simplified environment, accessible touching **Arm** field in the **Status** bar.

5.10.1.3 Tool, Frame, Base

These three fields allow selecting the frames of reference to be used during manual motion and programming environment, choosing among the ones stored in the [System tables](#) of the [Data Page](#) (TOOL, UFRAME, BASE tables).

To act on the wished reference system, touch the corresponding field. The available sub-environments are as follows:

- [Tool and Frame](#)
- [Base](#).

5.10.1.3.1 Tool and Frame

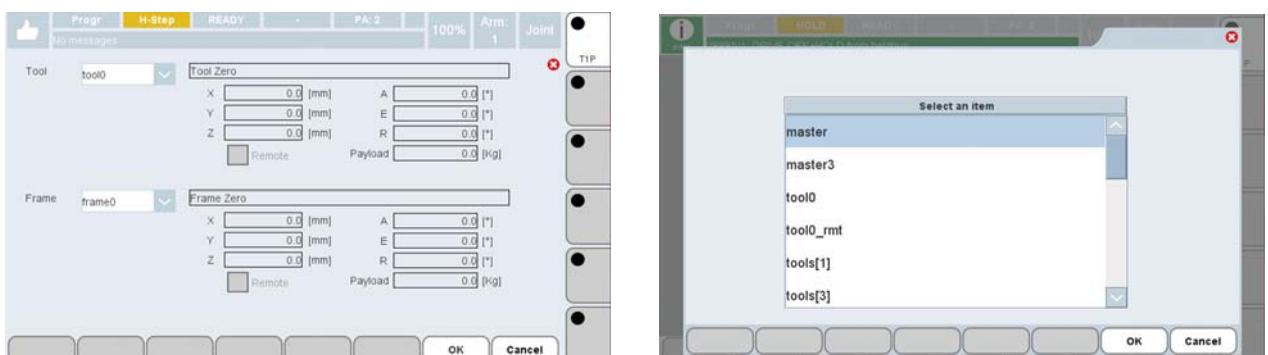


Note that the selected Tool must always be consistent with the selected Frame. If, for example, a remote Tool is to be chosen, the user must take care of choosing a Frame with the same attribute. At the end of the modifications, the needed check are performed and any invalid settings are refused.

When the user chooses to modify either Tool or Frame, the screen page shown in [Fig. 5.17](#) is displayed.

Open the suitable menu by touching either Tool or Frame. A list is displayed in which the user can choose by touching the wished item (the below example shows how to modify the Tool coordinates - see [Fig. 5.17](#) on the right).

Fig. 5.17 - Choosing Tool and Frame



Act in the same way to choose the other item, if needed, by either confirming the current value or selecting a different one.

Touch the **Cancel** key to exit from the sub-environments without making any modifications. If the system detects any inconsistency between the chosen Tool and Frame, the required modifications are refused.

Note that both for Tool and Frame, the following relevant read-only information is displayed:

- X, Y and Z cartesian coordinates, in millimeters
- A, E and R orientation angles, in degrees
- remote - indicates whether the corresponding Tool/Frame has got the Remote attribute
- payload - payload value associated to the Tool/Frame.

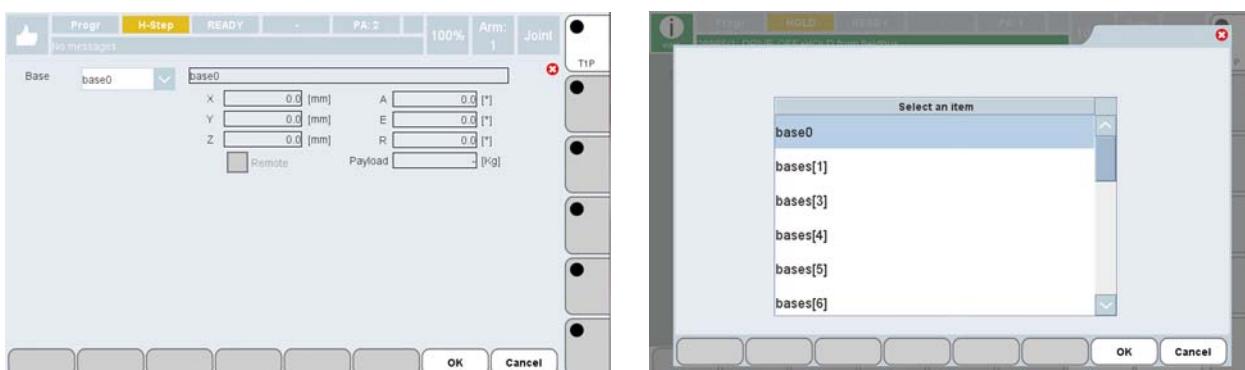
5.10.1.3.2 Base



Note that, unlike what happens for Tool and Frame, the Base can be autonomously selected, not related to the other two reference systems, so no consistency checks are performed.

When a Base selection is required, the dedicated screen page is displayed (see Fig. 5.18 on the left).

Fig. 5.18 - Choosing Base



Touch the **Base** field. The system opens a list in which the user can select the whished item by touching it (Fig. 5.17 - see figure on the right).

Touch the **Cancel** key to exit from the sub-environment without making any modifications.

Note that the following relevant read-only information about the Base, is displayed:

- X, Y and Z cartesian coordinates, in millimeters
- A, E and R orientation angles, in degrees.

5.10.1.4 Current arm position

This section of the Basic sub-page (light blue in Fig. 5.14) gives the current values of:

- **ARM_POS** - Cartesian position of the specified arm (expressed in millimetres for X, Y and Z, and in degrees for the orienting angles A, E and R)
- **ARM_JNTP** - joint position (expressed in degrees or in millimetres, according to the axis configuration)
- **config** - configuration string (attitude flag and multirev flag).



If the axis is NOT calibrated, or has lost the number of revolutions, these values are NOT available (and the '--' string is displayed).

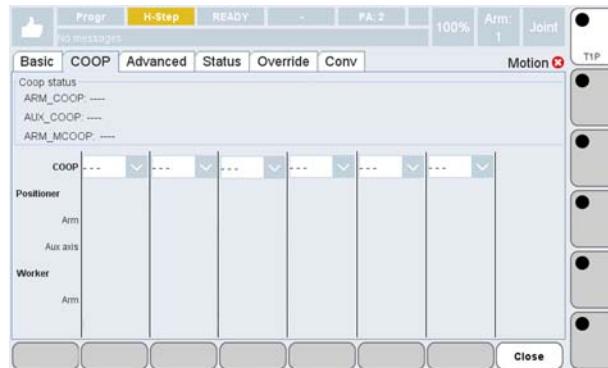
5.10.1.5 Payload

Indicates the payload value.

5.10.2 Coop (optional feature)

This subpage allows managing the cooperative motion (for further information, see [Cooperative Motion \(optional feature\)](#) in **Motion Programming** manual).

Fig. 5.19 - Coop subpage



The Coop subpage provides the following sections:

- [Coop Status](#) - cooperation status,
- [COOP](#) - using the cooperative motion commands.



For further information, see [par.5.12 Integrated Movement](#) in [Motion Programming Manual](#).

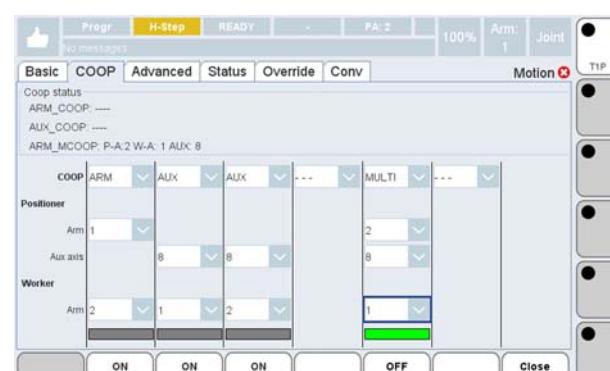
5.10.2.1 Coop Status



This section is read-only and displays information about the currently active cooperations status. See example in [Fig. 5.20](#).

5.10.2.2 COOP

Fig. 5.20 - Cooperation commands



In the current section some cooperation built-in routine calls (**ARM_COOP**, **AUX_COOP**, **ARM_MCOOP**) can be set up by the user, in order to activate the associated type of cooperation later on, when needed:

- open COOP menu and select the wished routine,
- insert the suitable values into the fields associated to such a routine,
- activate the wished cooperation type, by touching the corresponding **ON** key.

See the example shown in Fig. 5.20: 4 different calls to cooperation routines have been set up. The currently active cooperation type is the green one. The inactive ones are grey.

Furthermore, about the shown above example, the currently active status (for **ARM_MCOOP**) is displayed in the **Coop Status** section: the **Worker Arm** is Arm 2, the **Positioner Arm** is Arm 1 and the auxiliary axis of the Positioner is axis 8, as stated by the characters string

P-A:2 W-A:1 AUX: 8.

Obviously, nothing is active for the other two available cooperation states (**ARM_COOP** and **AUX_COOP**).



Also refer to:

- built-in **ARM_COOP**, **AUX_COOP** and **ARM_MCOOP** in [chap. BUILT-IN Routines list of PDL2 - Programming Language manual](#)
- [par. 5.12 Arm Settings Page on page 112](#) for **ARM_COOP** modality enabling/disabling.

5.10.3 Advanced

Fig. 5.21 - Advanced subpage



In the Advanced sub-page, data may be modified about

- [Incremental Jog mode](#)
- use of [J-Pad](#).

A detailed list of each one of them is given below.

5.10.3.1 Incremental Jog mode

In this section it is possible to modify, in millimetres or degrees, the distance that is covered each time an axis jog key is pressed (+ or -) - see [par. 5.3.1.2.5 JOG keys on](#)

page 59. Touch the **Enable** field to enable this mode.

In this way **Linear** and **Angular** fields can be accessed to make the suitable settings.

By either touching one of these fields or moving focus to each one of them, some functional keys (**small**, **med** and **large**) are made available in the **Functional keys menu** which allow more quickly entering values into these fields (Fig. 5.21).

5.10.3.2 J-Pad

For further information see also the **JPAD** keyboard description.

This section is divided into two parts, as can be seen in the figure below:

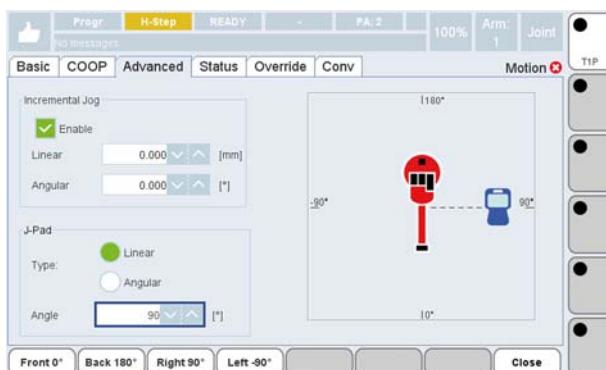


- **Type** (Linear/Angular)

These fields indicate whether, using **JPAD**, the robot makes a linear move (**Linear**) or a rotation (**Angular**).

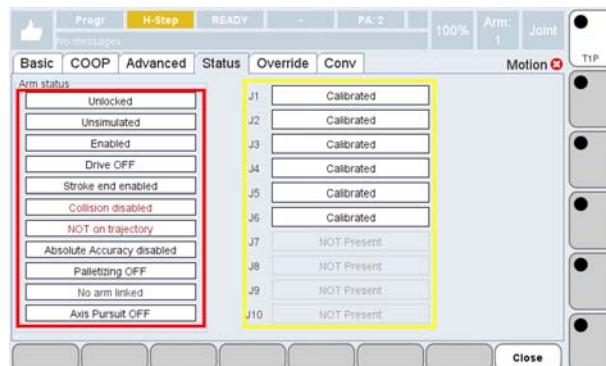
- **Angle**

This field indicates the user position in relation to the robot; such a position is pointed out by the Teach Pendant icon, which moves to the requested angular position, around the robot. Touching the **Angle** field or when the focus is on it, some commands are made available in the **Functional keys menu** (**front 0°**, **back 180°**, **right 90°**, **left-90°**) which allow quicker specifying the new User's position. However, the value may be modified directly touching the **Angle** field and typing the new value in the **Numeric keyboard** which is automatically opened by the system (see figure below on the right).



5.10.4 Status

Fig. 5.22 - Status subpage



The Status sub-page is related to the **Arm state** (red) and the **Axes state** (yellow).

For the **Arm state** (red) the available values are:

- Need for Unlock, Not in LOCK, Need to Resume
- Simulated, Unsimulated
- Mov. enabled, Mov. disabled
- DRIVE ON, DRIVE OFF - state of drives
- Stroke end enabled, Stroke end disabled.
- On trajectory, NOT on trajectory
- Absolute Accuracy enabled/disabled - the machine is/is NOT kinematically compensated.

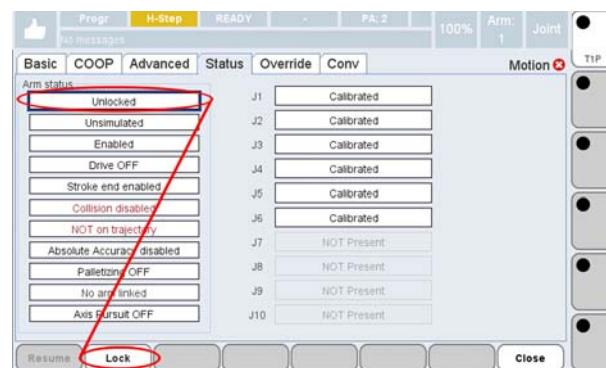


Furthermore, some optional fields are provided which are displayed only if the corresponding optional functionality has been purchased:

- **Coop (optional feature)**
- **Collision Detection (optional feature)**
- **Palletizing (optional feature).**

When navigating on these fields, in the **Functional keys menu** there are commands available, that correspond to the selected state, and allow it to be changed. For example:

Fig. 5.23 - Not in LOCK



For the **Axes state** (yellow in Fig. 5.22), the available values are:

- calibrated
- NOT calibrated
- NOT in Turn-set
- NOT connected, for example in case of electric gun change
- NOT Present.

Also for the Axes state, touching the different fields or moving focus onto them, some keys are shown in the [Functional keys menu](#) that correspond to the selected state, and that allow to modify it.

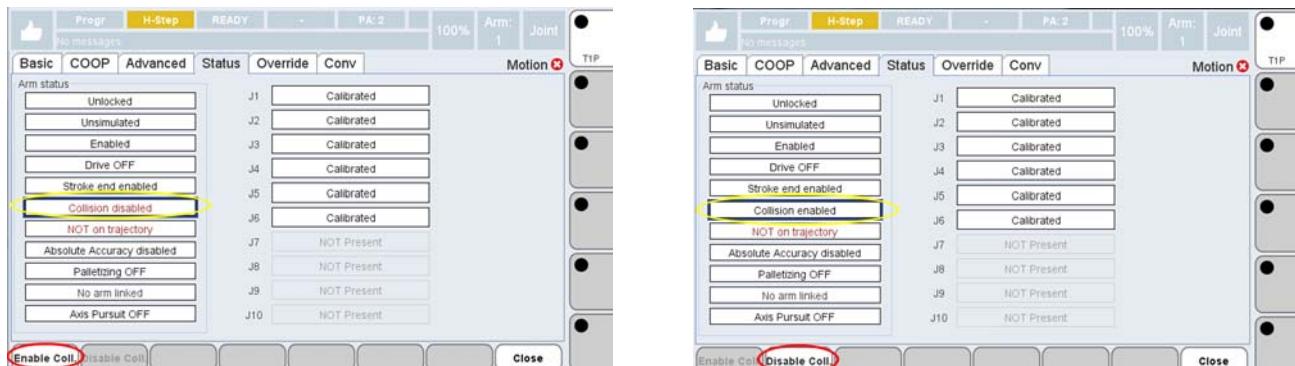
For both the state of the arm and of the axes, under normal operating conditions, all the text strings of the different fields are BLACK; otherwise in case of an anomalous arm condition or if the calibration or Turn-set data is missing, the colour is RED.

5.10.4.1 Collision Detection (optional feature)

If the System also includes the Collision Detection optional functionality, the Status subpage provides the following information (see Fig. 5.24 - yellow highlighted fields):

- **Collision Disabled** - Collision Detection functionality is disabled
- **Collision Enabled** - Collision Detection functionality is enabled

Fig. 5.24 - Collision Detection

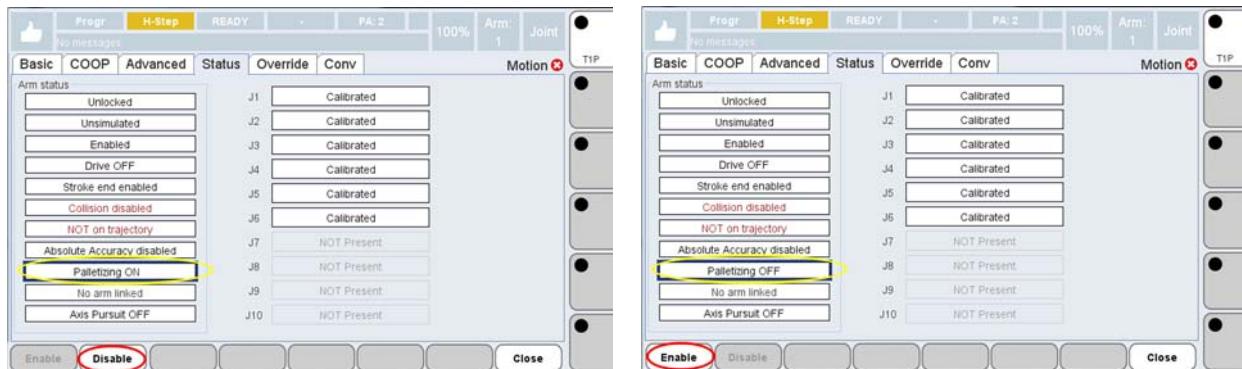


When the field representing the Collision Detection status either is touched or focus is on it, a command is made available in the [Functional keys menu](#), to allow **enabling (Enable Coll.)** or **disabling (Disable Coll.)** the functionality, depending on the current status (see Fig. 5.24 - red highlighted keys).

5.10.4.2 Palletizing (optional feature)

If the System also includes the Palletizing optional feature, the Status subpage provides the following information (see Fig. 5.25 - yellow highlighted fields):

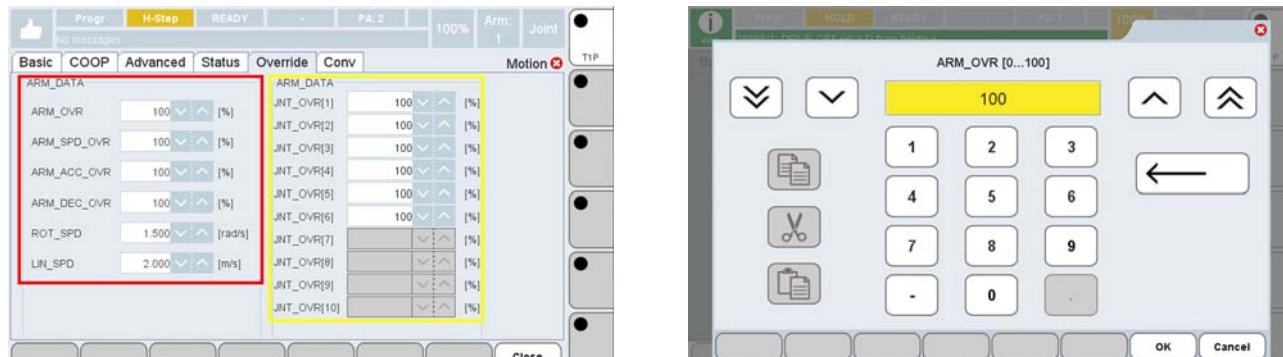
- **Palletizing ON** - Palletizing functionality is enabled
- **Palletizing OFF** - Palletizing functionality is disabled

Fig. 5.25 - Palletizing functionality

When the field representing the Palletizing status either is touched or focus is on it, a command is made available in the [Functional keys menu](#), to allow **enabling (Enable)** or **disabling (Disable)** the functionality, depending on the current status (see Fig. 5.25 - red highlighted keys)).

In the [Status bar](#), when the Palletizing functionality is enabled, the string '**Pallet**' is displayed.

5.10.5 Override

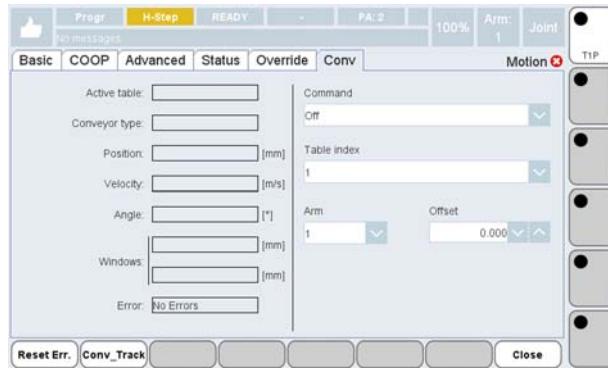
Fig. 5.26 - Override sub-page

The Override sub-page contains the overall pre-defined values (red) and single joints (yellow) relating to the current arm. All the information can be changed by the user by touching the wished field. The system automatically opens the [Numeric keyboard](#) (Fig. 5.26, on the right) and the user is allowed to insert the new value.

For further details about the above mentioned predefined variables, please refer to [PDL2 Programming Language Manual](#), chap.**PREDEFINED VARIABLES**.

5.10.6 Conveyor Tracking (optional feature)

This sub-page is available only if the System is provided with the Conveyor Tracking option. It is used to handle such an optional feature.



The left section in the above figure is read-only and displays the Conveyor current status:

- **Active table** - is the index of the currently active \$TRK_TBL table
- **Conveyor type**
 - 0: linear cartesian tracking;
 - 1: circular cartesian tracking;
 - 3: trans-rotational cartesian tracking (Robot controlling a bending-press).
 - 4: rail tracking in which the positive direction of the robot rail is the forward direction of the conveyor;
 - 5: rail tracking in which the negative direction of the robot rail is the forward direction of the conveyor.
- **Position** - is the conveyor current position, in mm
- **Velocity** - is the conveyor instantaneous velocity, in m/s
- **Angle** - only existing for conveyor of type 3 (trans-rotational cartesian tracking) and representing the V-opening in radians
- **Windows** - the first value represents the conveyor quote from which the robot must start with tracking; the second value represents the conveyor quote after which no tracking with the robot is allowed
- **Error** - indicates the conveyor error status. When red, resetting it is allowed by means of **Reset Err.** key.

The right section includes some menus to set up the **CONV_TRACK** built-in routine parameters:

- **Command** - this menu allows choosing among one of the following commands:
 - **Initialize** - initialize (-1)
 - **Initialize + On** - initialize and activate (1)
 - **On** - activate (2)
 - **Off** - deactivate (0)
- **Table Index** - is the index of the tracking table in which new settings are wished
- **Arm** - is the number of the Arm involved in the tracking functionality
- **Offset** - is the distance between the sensor and the being tracked object, at the conveyor initialization time; the default is 0.

When the **CONV_TRACK** built-in routine setup is completed, touch **Conv_Track** key to execute it.



For further information, please refer to [par. 5.20.2.3 Conveyor on page 214 - Setup page](#).

5.11 Arm Override Page



In this page the user is allowed to

- modify the general override for the active Arm
- enable/disable Incremental Jog.

To increment/decrement the *general override*, act in one of the following ways:

- tap + and - keys

The increment/decrement quantity is **5%**. If the starting value is less equal 5%, increment/decrement is **1%**.

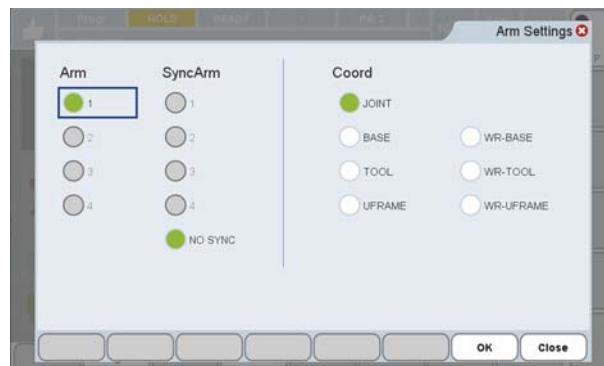
- tap the diagram bars

The override is set to the displayed value below the tapped bar.

In the following example, tapping the 75% bar comes to an override value set to 75%.



5.12 Arm Settings Page



This page allows setting the following choices:

- combination of **Arm** and **SyncArm** if existing - select the wished item in each one of the two **Checkgroups**
- modality for handling the jog motion coordinates. Select the wished modality among the ones displayed in the **Coord Checkgroup**:
 - **TP JOG** mode
 - JOINT
 - BASE
 - TOOL
 - UFRAME
 - **WRIST** mode (motion referred to the robot wrist)
 - WR-BASE
 - WR-TOOL
 - WR-UFRAME.
- enabling/disabling the **ARM_COOP** modality which is currently set in the **COOP** subpage of the **Motion Page**.

5.13 Alarm Page

This User Page allows managing information related to alarms/errors that have occurred in the system.

It consists of 2 sub-pages, which may be selected through their corresponding labels:

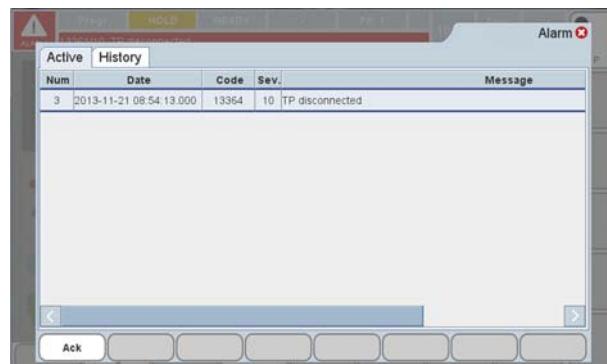
- [Active](#)
- [History](#).



Sometimes, some alarm messages could occur for which it is not so easy to understand the error cause and the corresponding remedy.

In such scenarios, to better decide what to do, it is suggested to use both the Controller Unit and the specific Robot [Maintenance](#) manuals.

5.13.1 Active



The **Active** sub-page displays the active alarms, i.e. that require **confirmation** from the user, and gives the following information:

- alarm ordinal number
- date and time
- error code and severity
- message text

Alarm code, severity and message text are also displayed in the [Status bar](#), as long as the alarm is active.

To confirm an alarm it has to be selected (see the following figure): press the down arrow to move the focus on the list of alarms, then choose the alarm with the cursor keys. At this point softkeys are available in the [Functional keys menu](#) to be able to perform some operations (e.g.: Confirm, Skip, etc.). The presence of one or several softkeys, or not, depends on the selected alarm type.

In particular, when the EXCL softkey is enabled, the pressure on this key enables the exclusion, for a limited time interval, of the effect of alarms such as those connected to an electrical limit switch.



When **EXCL** command is enabled (see the above figure), touching such a key allows excluding, for a limited time interval, the effect of alarms such as those related to an electrical stroke end.

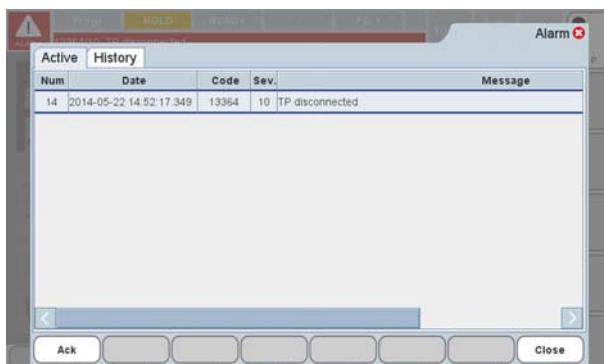
This command is available in Progr state only. As soon as the axis is moved beyond the electrical stroke end, an error message is generated and the robot is put to DRIVE OFF state. To exit from such a scenario, touch the **EXCL** key, switch the system to DRIVE ON state and jog the axis in the opposite direction in respect to the one of the stroke end.

To confirm an alarm, touch it to select it. In such a way the **Ack** command is made available in the **Functional keys menu** allowing to confirm the selected alarm (see the figure below).

The user should also use this page to confirm the so called **latched** alarms, to clear them from such status.

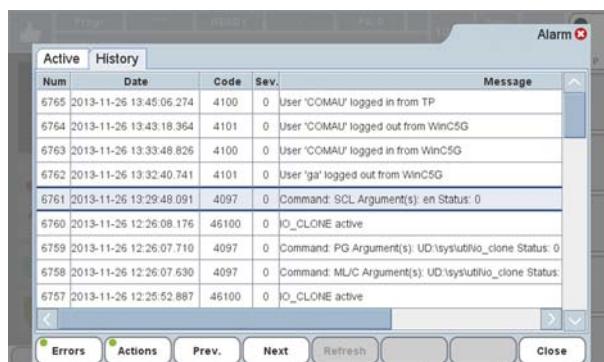


For further information about **Latched** alarms, refer to **PDL2 Programming Language**, manual, **\$LATCH_CNFG** predefined variable.



5.13.2 History

It is a unique log environment, including both trace of the occurred **Errors** and the performed **Actions** on the Control Unit.



By means of **Errors** and **Actions** functional keys, the user is allowed to filter the log messages viewing, by including the ones and/or the other ones, depending on his/her specific needs.

For each occurred error (errors log), the following information is displayed:

- alarm ordinal number
- date and time
- error code and severity
- message text.

Use the **cursor up/down** keys to scroll them all.



The errors that are stored in the errors LOG file are only those that change the system state

5.14 Prog Page

Purpose of this User Page is to handle the programs.



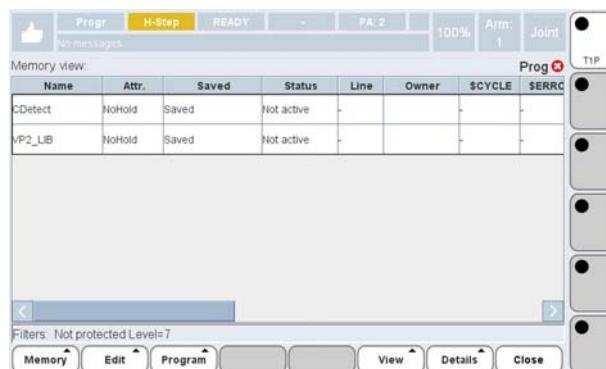
Programs Classification

Programs for the C5G Control Unit are written in PDL2 Programming language. We can classify them into two main categories, depending on the assigned holdable/non-holdable attribute:

- ‘holdable’ Programs (i.e with HOLD attribute) - their execution is controlled by the **HOLD (yellow)** and **START (green)** keys; a Program including motion statements MUST be a ‘holdable’ Program.
- ‘non-holdable’ Programs (i.e with NOHOLD attribute) - they are usually used for process controlling. They CANNOT include motion statements, even if they are allowed to use positional variables as well, for other purposes.

The default attribute is **HOLD**; if it is wished to write a ‘non-holdable’ Program, it is needed to explicitly insert the NOHOLD attribute.

In this page a table is displayed including the programs loaded in the execution memory.

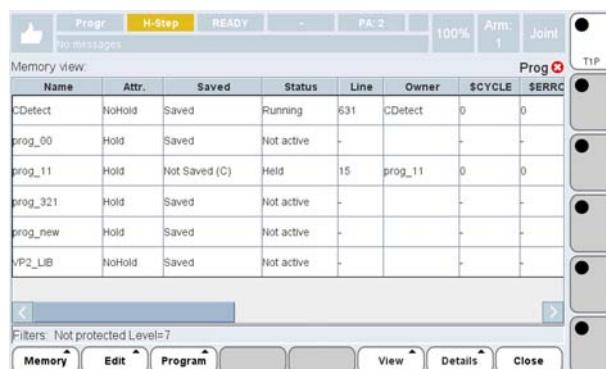


The screenshot shows the 'Memory view' table with the following data:

Name	Attr.	Saved	Status	Line	Owner	\$CYCLE	\$ERRC
CDetect	NoHold	Saved	Not active	-	-	-	-
vP2_LIB	NoHold	Saved	Not active	-	-	-	-

The following information is shown for each displayed item:

- **Name** - program name (32 characters of max. length)
- **Attr.** - program attribute (motion programs (Hold), no motion programs (NoHold))
- **Saved** - state of the program file (already saved (Saved), not saved yet (Not Saved))



The screenshot shows the 'Memory view' table with the following data:

Name	Attr.	Saved	Status	Line	Owner	\$CYCLE	\$ERRC
CDetect	NoHold	Saved	Running	631	CDetect	0	0
prog_00	Hold	Saved	Not active	-	-	-	-
prog_11	Hold	Not Saved (C)	Held	15	prog_11	0	0
prog_321	Hold	Saved	Not active	-	-	-	-
prog_new	Hold	Saved	Not active	-	-	-	-
vP2_LIB	NoHold	Saved	Not active	-	-	-	-

- **Status** - program execution state (interrupted (Held), executing (Running), programma non attivo (Not Active), etc.)

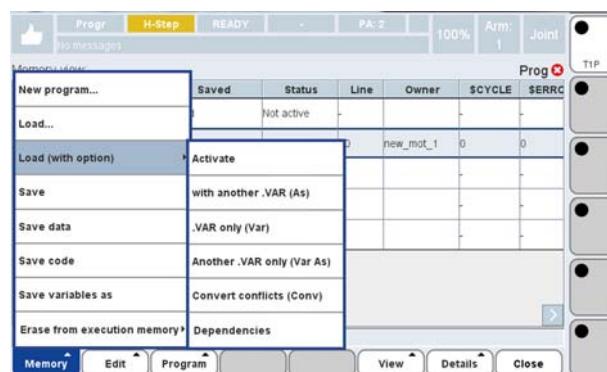
- **Line** - number of the being executed line
- **Owner** - name of the program that owns the currently being executed line (the line belongs to such a program code)
- **\$CYCLE** - value of \$CYCLE predefined variable belonging to the program
- **\$ERROR** - value of \$ERROR predefined variable belonging to the program
- **Arm** - program main arm
- **Step** - program step modality (Statement, Disable, etc.)
- **Path .COD** - directory path to reach the program .COD file
- **Time .COD** - creation date or last change of .COD
- **Size (KB)** - size of the program in kB
- **Path .VAR** - directory path to reach the program .VAR file
- **Time .VAR** - creation date or last change of .VAR

In the Prog Page **Functional keys menu** the following keys are available to allow the user operating on the currently selected program:

- [Memory](#)
- [Edit](#)
- [Program](#)
- [View](#)
- [Details](#)
- **Chiudi** - closes Prog Page.

5.14.1 Memory

Fig. 5.27 - Memory menu



Touch this key to open the following commands menu:

- [New program...](#)
- [Load...](#)
- [Load \(with option\)](#)
- [Save](#)
- [Save data](#)
- [Save code](#)

- Save variables (As)
 - Erase from execution memory.
-

5.14.1.1 New program...

Allows creating a new program to be developed in **IDE**. The following situations might occur:

- the requested program already exists in execution memory: the command does not take any effect
- the requested program only exists as a file in UD:. The command loads it in the execution memory.
- the requested program does not exist. The command creates a .COD file and loads it in the memory.

To open the created program, act in one of the following ways:

- touch the name to select it, then touch command **Open [ENTER]** available in **Program** menu, or
 - touch the program name twice, or
 - touch the name to select it, then press **ENTER**.
-

5.14.1.2 Load...

Loads the selected program(s) and the corresponding variables file(s).

After activating this command, the user is allowed to specify device, directory and file(s) as needed.

5.14.1.3 Load (with option)

The user is also allowed to load a program with the following options (see Fig. 5.27):

- Activate
- with another .VAR (As)
- .VAR only (Var)
- Another .Var only (Var As)
- Convert conflicts
- Dependencies.

A detailed list of each one of them is given below.

5.14.1.3.1 Activate

Touch this item to load the selected program into execution memory and activate it.

5.14.1.3.2 with another .VAR (As)

This option allows to load a program with variables file (.VAR) different from the .COD file. For example program file **PALLET.COD** can be loaded together with **PALLET_12.VAR** variables file.

5.14.1.3.3 .VAR only (Var)

Touch this item to load the selected variables file only.

5.14.1.3.4 Another .Var only (Var As)

Allows **only** a variables file (.VAR) to be loaded, associating it to a program entry with a **different** name. The program entry is to be selected from the list of the existing .COD. The name of the variables file is to be chosen from the .VAR list.



This command DOES NOT load the .COD.

5.14.1.3.5 Convert conflicts

This option is useful in case of conflicts detected during a loading operation, when they are solvable (e.g. a conflict of array or string size)..

5.14.1.3.6 Dependencies

In addition to the requested program, this option automatically loads code and vars of any programs that are called up by it.

If the referred files are in different directories than the loaded program, it is necessary to set \$DEPEND_DIRS predefined variable with the search path to be used by Load command.

5.14.1.4 Save

Save in UD: code and variables of a program, from the execution memory in a file on the default device.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

5.14.1.5 Save data

Saves the variables of the selected program

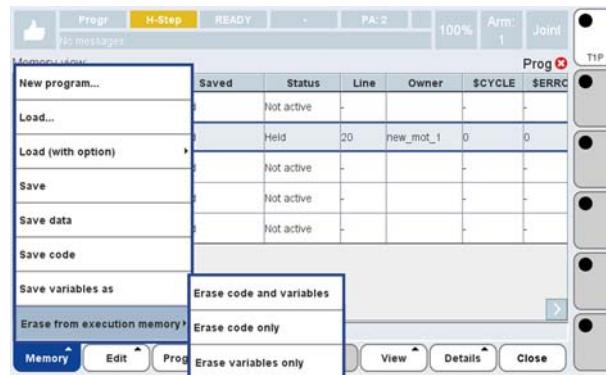
5.14.1.6 Save code

Saves the code of the selected program

5.14.1.7 Save variables (As)

Saves the program variables in a file with different a name than the program one. This option does not allow using the wildcard.

5.14.1.8 Erase from execution memory

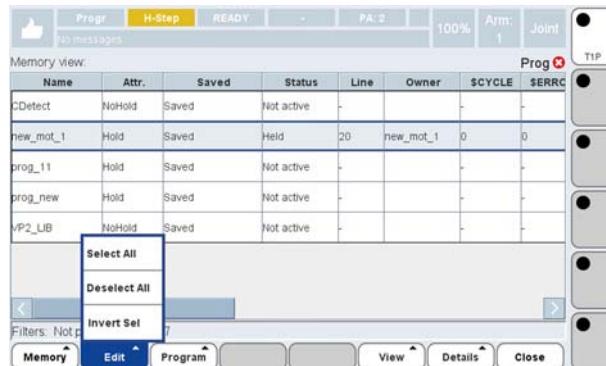


This submenu contains commands to delete a program and its variables from the execution memory. If other programs are associated to a routine or to variables of the being deleted program, an error occurs.

The available functions provided are as follows:

- **Erase code and variables** - Cancels both selected program code and variables.
- **Erase code only** - Cancels the selected program code only.
- **Erase variables only** - Cancels all the selected program variables.

5.14.2 Edit



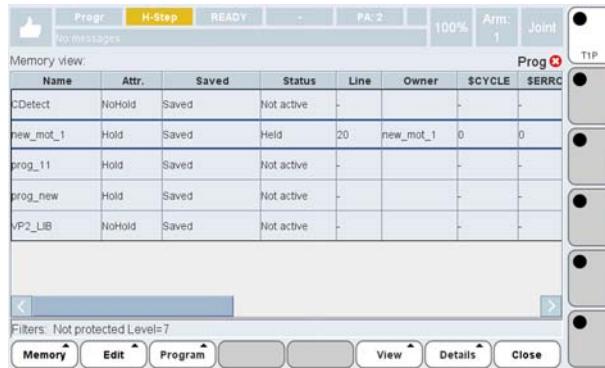
Touch this key to access a menu which operates on the displayed programs.

The available commands are as follows:

- **Select All**
- **Deselect All**
- **Invert Sel.**

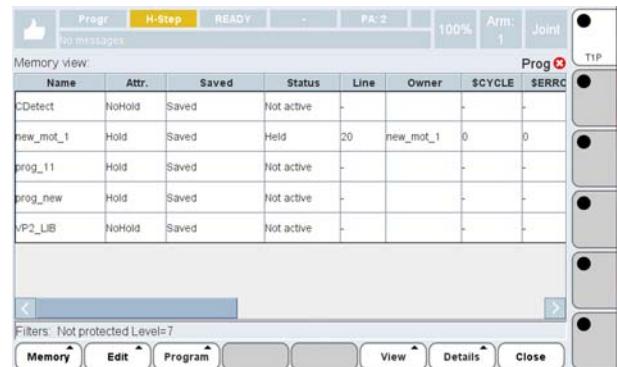
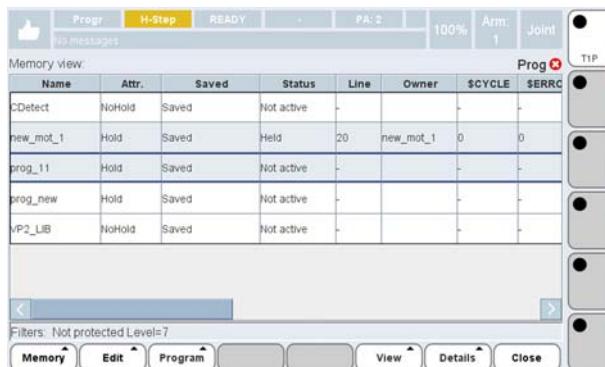
5.14.2.1 Select All

Touch this command to select all the currently displayed programs.



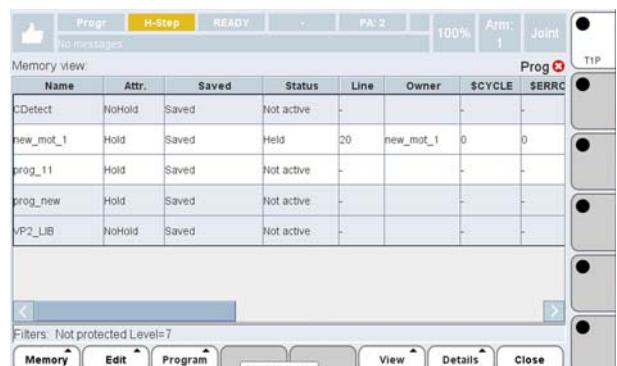
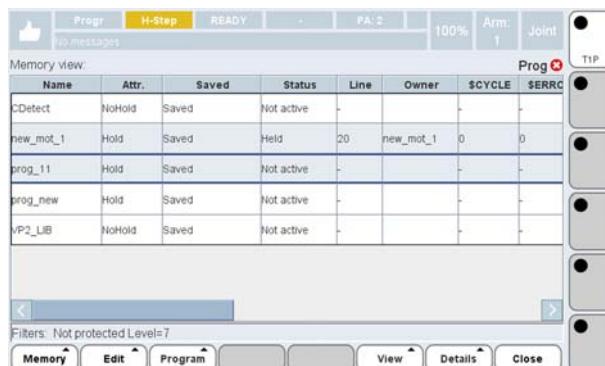
5.14.2.2 Deselect All

Touch this command to unselect all the currently selected programs. See the figures below for the situations before (on the left) and after (on the right) using such a command.



5.14.2.3 Invert Sel

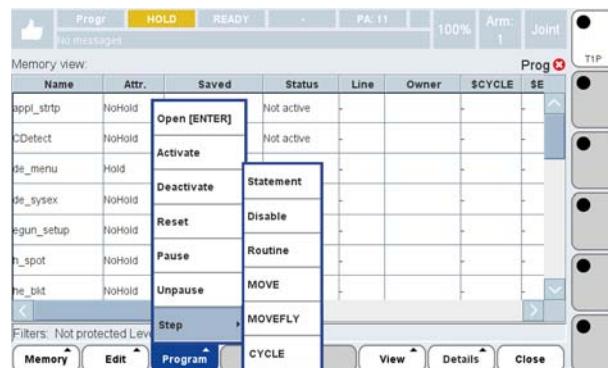
Touch this command to cause all the selected programs to be unselected and viceversa. See the figures below for the situations before (on the left) and after (on the right) using such a command.



5.14.3 Program

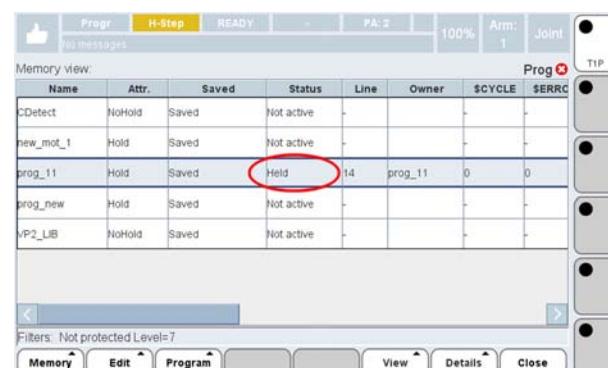
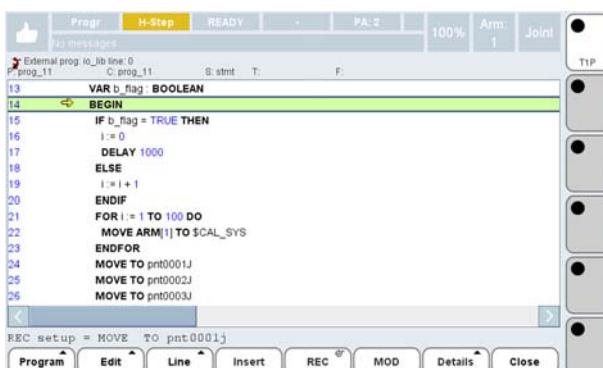
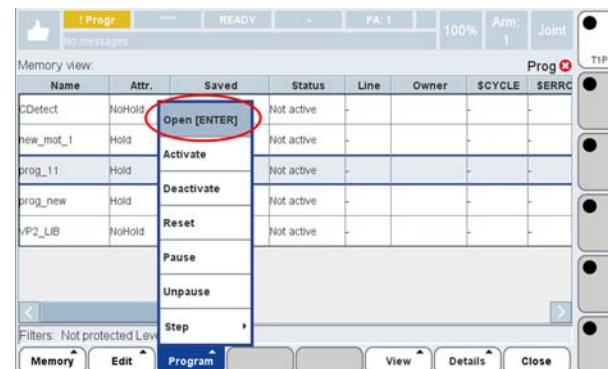
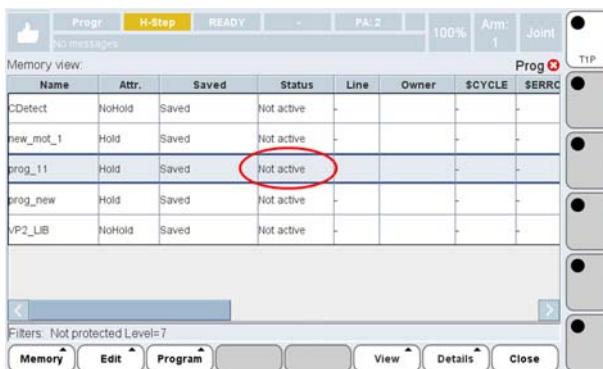
As shown in the figure below, the available choices are as follows:

- Open [ENTER]
 - Activate
 - Deactivate
 - Reset
 - Pause
 - Unpause
 - Step



5.14.3.1 Open [ENTER]

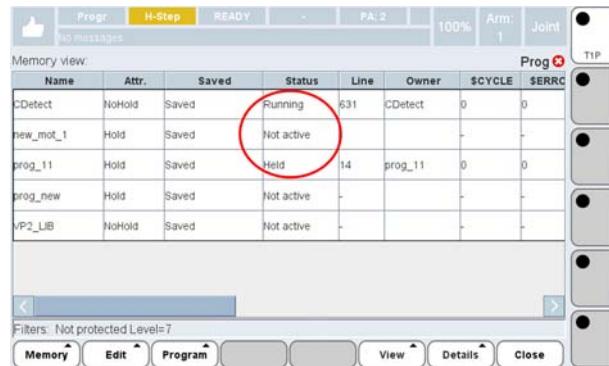
Touch this command to open the selected program in the Integrated Development Environment (**IDE**). If the program is not currently active, it is automatically activated and put into Held state.



5.14.3.2 Activate

Allows to start the execution of the selected program. If it is a holdable program, the command sets it in **Held** state; press **START (green)** key to start the execution.

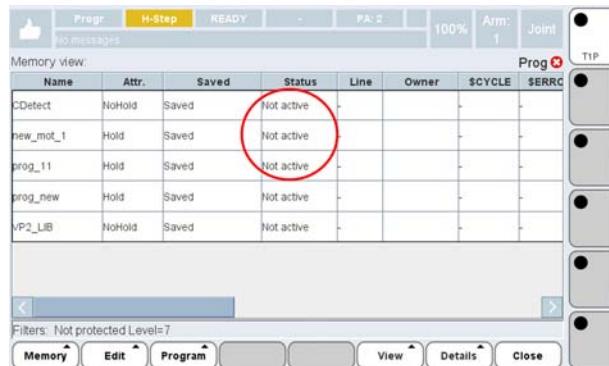
On a non-holdable program, the command has immediate effect and sets it directly into **Running** state.



Only one **holdable** program at a time can be activated, unless the DETACH attribute is specified in the header of the programs which are to be simultaneously executed.

Before the program is activated, it has to be already loaded into the execution memory ([Load...](#)).

5.14.3.3 Deactivate



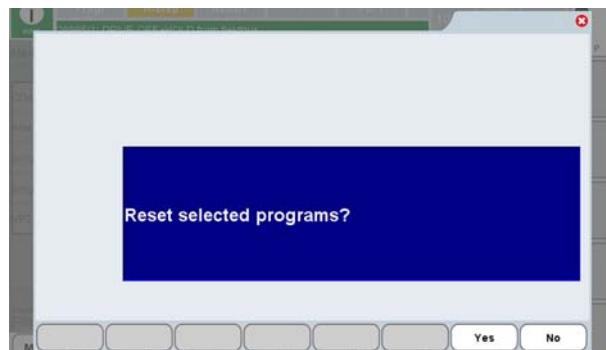
Deactivates the selected programs which, anyway, still remain in the execution memory.

5.14.3.4 Reset

This command allows deactivating the selected programs and then activating them again.

The system displays a suitable message to prompt the user for confirmation (see the figure below).

Answer **Yes** to make the command operational.



5.14.3.5 Pause

Name	Attr.	Saved	Status	Line	Owner	\$CYCLE	\$ERRC
CDetect	NoHold	Saved	Not active	-	-	-	-
new_mot_1	Hold	Saved	Not active	-	-	-	-
prog_11	Hold	Saved	Suspended	15	prog_11	0	0
prog_new	Hold	Saved	Not active	-	-	-	-
vP2_LIB	NoHold	Saved	Not active	-	-	-	-

Suspends the execution of the selected programs, setting them in **Suspended** state. To continue the execution it is necessary to issue the [Unpause](#) command

5.14.3.6 Unpause

Resumes the execution of the selected programs, removing them from the **Suspended** state. The program is set to the **Running** status provided that no other program suspension condition subsists. To resume the execution of holdable programs, [START \(green\)](#) key has to be pressed.

5.14.3.7 Step

This menu allows defining the program execution step.

It is mainly used upon the program debug.

Name	Attr.	Saved	Status	Line	Owner	\$CYCLE	\$ERRC
CDetect	NoHold	Open [ENTER]	Not active	-	-	-	-
new_mot_1	Hold	Activate	Not active	-	-	-	-
prog_11	Hold	Deactivate	Statement	15	prog_11	0	0
prog_new	Hold	Reset	Disable	-	-	-	-
vP2_LIB	NoHold	Pause	Routine	-	-	-	-
		Unpause	MOVE	-	-	-	-
		Step	MOVEFLY	-	-	-	-
			CYCLE	-	-	-	-

Usually, the program execution is continuous, i.e. without interruption. The execution

step allows defining the moment in which the program execution shall be interrupted.

A new execution step is started each time the **START** key is pressed.

The available commands are:

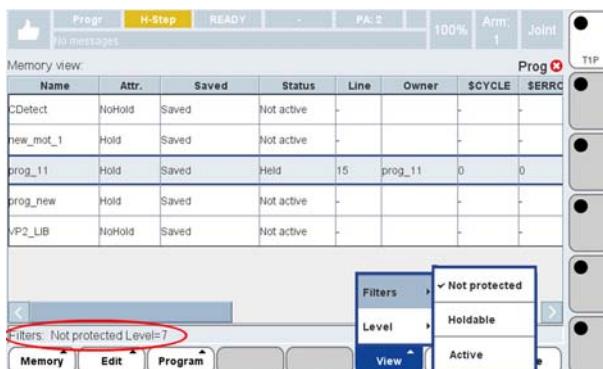
- **Statement** - The execution is interrupted at each statement. The protected Routines (whose body is not displayed) are run as they were a single statement; the execution of non protected Routines (whose body may be displayed) is interrupted at each statement.
- **Disable** - Stops the step-by step execution and hence the program is run in continuous mode.
- **Routine** - Similar to **Statement**, however routines are run as they were single statements.
- **MOVE** - The execution is interrupted at each individual movement. This is not allowed on non-holdable programs.
- **MOVEFLY** - Executes fly movements before interrupting the program execution. Similar to **MOVE**, however the program execution does not stop after the MOVEFLY statement. This is not allowed on non-holdable programs.
- **CYCLE** - Defines an individual program cycle, as the step option; the selected program must contain the CYCLE or BEGIN CYCLE statement.

5.14.4 View

This menu provides some commands to customise programs viewing, according to some wished [Filters](#) and [Levels](#).

The currently active filter and level types, are displayed in the field below the programs table (as highlighted in the below figure).

5.14.4.1 Filters



The following filter choices are available:

- **Not protected** - displays not protected programs only
- **Holdable** - displays programs with HOLD attribute only
- **Attivi** - displays currently active programs only.

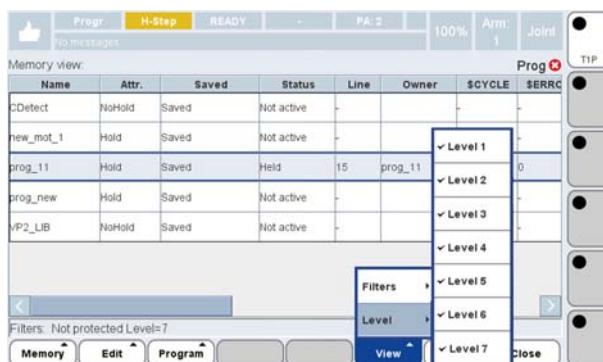
The current filter is displayed in the line above the [Functional keys menu](#) (see previous figure, highlighted in red).

5.14.4.2 Level

This command displays the allowed actions level for the selected program. Both **viewing actions** and **editing actions** are available.

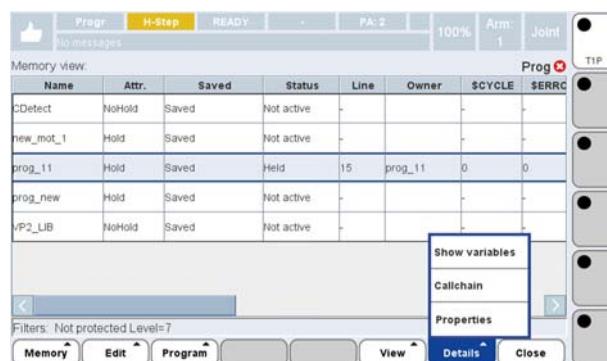
For further information about levels, refer to [par. HANDLING PROGRAMS EDITING AND VIEWING LEVELS on page 130](#).

The current level is displayed, together with the current filter, in the line above the **Functional keys menu** (see previous figure, highlighted in red).



5.14.5 Details

Allows the user to display some data related the selected program.



The available choices are as follows:

- [Show variables](#)
- [Callchain](#)
- [Properties](#)

A detailed description follows for each one of them.

5.14.5.1 Show variables

It displays the variables of the selected program, with the following information ([Fig. 5.28 - POS SHIFT command on page 127](#)):

- owning program (**Owner**)
- variable name (**Name**)
- **Value**
- **Comment**.

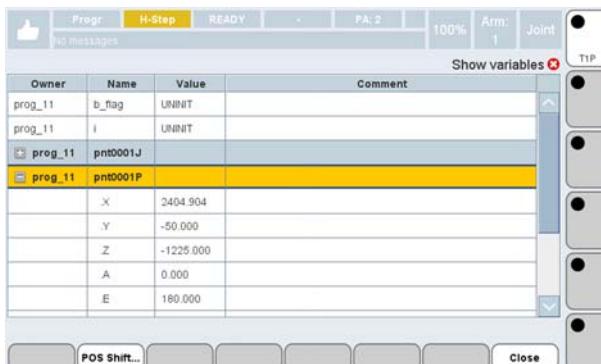
If the selected variable is a positional one of **POSITION** and/or **XTNDPOS** type, the **POS SHIFT** key is made available in the [Functional keys menu](#).

POS SHIFT

This function (see [Fig. 5.28](#)) allows “moving” in space positions that belong to the selected program, modifying its coordinates.

Typically, it is useful to translate the execution of an already taught path, by acting on the positions belonging to such a path.

Fig. 5.28 - POS SHIFT command



Touching this key, a page is opened in which the user is allowed to specify the wished offsets, related to the current components for **X**, **Y** and **Z** (the inserted values represents the translation millimeters along all the three directions of the cartesian coordinates system), as well as to select the wished reference frame, as shown in the below figure. If the selected variable is just one, the system displays its name and the corresponding modifications are just referred to it (see the highlighted area in the below figure).



Viceversa, when more than one variables are selected (as shown in the following example, figure on the left) the system displays the text “multiple selection” instead of the variable name (see the following figure on the right) and thus the translation is referred to all the selected variables.



Finally, touch **Apply** key to confirm the modifications.

At the end of the operation, the system prompts the user with a suitable message.
To touch **Close** key to quit the subpage.

5.14.5.2 Callchain

Displays the nesting of routines calls for the selected program (see [Fig. 5.29](#)).

The following information is given:

- **routine** name
- number of **line** where the call is made
- **status**
- program the routine belongs to (**owner**)
- program context (**type**): main, interrupt routine, routine, etc.

Fig. 5.29 - Callchain command



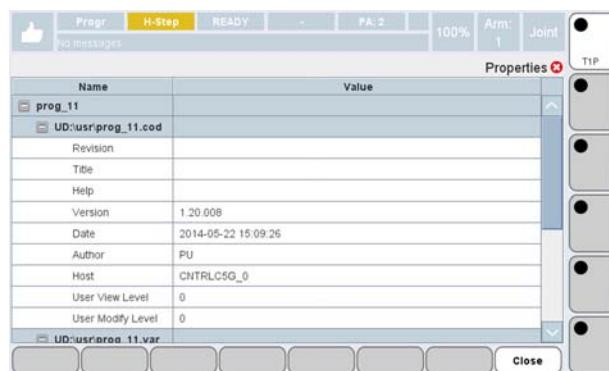
This command is allowed for active programs ONLY.

5.14.5.3 Properties

Allows displaying the Property Notes of the file including the selected program (see [Fig. 5.30](#)).

- **revision**
- **title**
- **help**
- **version**
- **date** - creation date
- **Author**
- **Host** - host device on which the file has been created
- **User View Level** - user level for viewing actions/livello utente per la modifica
- **User Modify Level** - user level for editing actions.

Fig. 5.30 - Properties



5.15 IDE Page



GLOSSARY

It is recommended to carefully read the following glossary, since in this paragraph the terms MODAL and NODAL are often referred to.

Such words are to be understood as follows:

- **modal - move and destination instructions only (example: MOVE TO pos1)**
- **nodal with clause WITH - settings that are only valid for the instruction where they are specified.**
- **customized nodal - very compact syntax, containing in a single line all the customising required by the customer. The description of this type of instruction is not contained in the standard manual, but only in the Nodal Moves manual, handed over to the customer for whom it has been developed. Therefore it is not dealt with herein.**

IDE environment (Integrated Development Environment) is used to develop motion programs (programs with HOLD attribute).

The following paragraphs give some information about:

- [Opening the IDE page](#)
- [Description of the video screen](#)
- [Available functions description](#)

5.15.1 Opening the IDE page

The IDE environment is used to edit a program which is **active, with HOLD attribute** (default).



HANDLING PROGRAMS EDITING AND VIEWING LEVELS

Two values are associated to each program file, which are included in the corresponding \$PROP_UVL and \$PROP_UML predefined variables (see [PDL2 Programming Language manual - chap.Predefined Variables List](#)); both an editing level and a viewing level are associated to each User; it is allowed to assign such levels by editing the .UDB file from within WinC5G program (see [par. 7.5.1.6 Viewing and editing of .UDB file \(optional feature\) on page 398](#)).

Such predefined variables values have the following meaning:

- **the User is allowed to edit the program, only if he/she has got an editing level greater than the file \$PROP_UML value;**
- **the User is allowed to view the program, only if he/she has got a viewing level greater than the file \$PROP_UVL value.**

Accessing the **IDE Page** is always related to a specific program; either pressing **MENU** special key and then choosing IDE icon, or touching **Open [ENTER]** command from **Prog Page**, the currently active program is opened. If no active programs are existing, an error message is displayed.



Use [New program...](#) command, available in the [Prog Page](#), to create a new program.

If the being opened program is not the active one, or if there is no active program, use the following procedure:

- a. activate the [Prog Page](#) by means of the **MENU** special key
- a.1 if the wished program is already present in the execution memory, go to step b.
- a.2 if the wished program is NOT present in the execution memory yet, load it by touching [Load...](#) command
- b. Select the wished program
- c. Either activate [Open \[ENTER\]](#) command or touch twice the wished program. It is then automatically activated. If another program is active, it is deactivated and the requested one is activated.

At the end of this procedure the system is in **IDE** environment.

5.15.2 Description of the video screen

When the IDE environment is active, the display screen of the Teach Pendant shows the following sections, which are peculiar to IDE:



The screenshot shows the Teach Pendant's IDE interface. At the top, there are several status indicators: 'Progr' (highlighted in red), 'H-Step', 'READY', 'PA: 2', '100%', 'Arm: 1', 'Joint', and 'T1P'. Below these are two status lines: 'External prog io.lib line: 0' and 'prog_11 C: prog_11 S: strmt T: F:'. The main area is a code editor with the following pseudocode:

```

14     VAR I:0..11 BOOLEAN
15 BEGIN
16     IF o_flag = TRUE THEN
17         I := 0
18         DELAY 1000
19     ELSE
20         I := I + 1
21     ENDIF
22     MOVE TO pnt0001P
23     MOVE TO pnt0002P
24     MOVE TO pnt0003P
25     FOR I := 1 TO 100 DO
26         MOVE ARM[1] TO $CAL_SYS
27     ENDFOR

```

At the bottom of the editor, there is a toolbar with buttons: Program, Edit, Line, Insert, REC (highlighted in blue), MOD, Details, and Close.

- IDE status lines
- Editor area.

5.15.2.1 IDE status lines

These are two areas of the display where the following information is shown:

- high status window (highlighted in red in the previous figure), containing
 - active program name (running program)
 - current program name (program displayed; for example it might be different than the program running if routines are called from other programs)
 - step mode:
 - statement - stmt (on single statement)
 - disable - dsbl (disabled step mode)
 - routine - rout (on routine)

- move - MOVE (on motion statement)
- fly - MOVEFLY (on motion in fly)
- cycle - CYCLE (on program cycle)
- active tool name
- active frame name
- if the system is executing a motion statement in an external program, the **MOTION CURSOR** symbol is shown, together with the name of the external program and the number of the line that contains such a motion statement;
- low status window (highlighted in blue in the previous figure), containing the current settings of the **REC** key.

5.15.2.2 Editor area

This is the window where the program statements are displayed. In this window, the user can edit and debug the active program, as well as follow its execution (see note about [Displaying the currently being executed statement](#)).

For an easy and efficient use of the editor area, the colours in which the program lines are displayed and the cursors that point to them have special meanings.

A detailed description follows of:

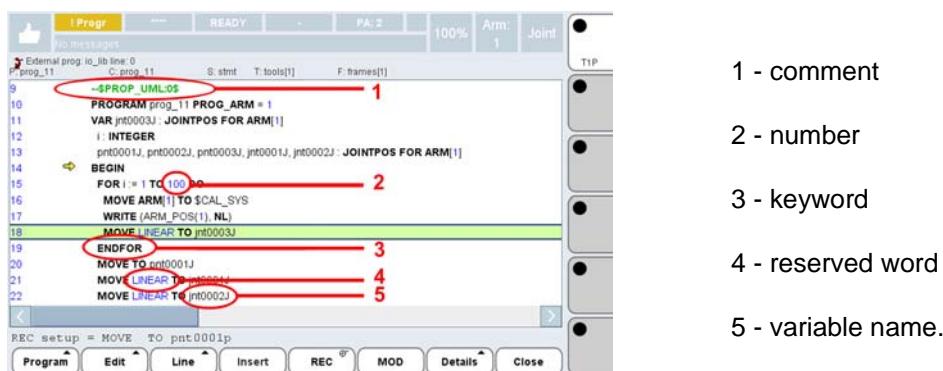
- [Colours](#)
- [Cursors](#).

5.15.2.2.1 Colours

The different components of the program lines are displayed with different colours, as follows:

- **comment** - green, bold
- **key word** - black, bold
- **reserved word** - blue
- **variable name** - black

In the figure below an example is shown about the listed above components.



5.15.2.2.2 Cursors

Three types of cursor could be displayed:

- [EDIT CURSOR](#)

- EXECUTION CURSOR
- MOTION CURSOR.

EDIT CURSOR

It corresponds to the instruction that may be modified by the user. The complete instruction line is highlighted with background:

- **light green** - editable line. Press **ENTER** to enter editing mode on the selected line
- **yellow** - line in editing mode.

During the **editing session**, the user can operate in one or more of the following modalities:

- edit an already existing program line - just move the edit cursor to the involved line and touch twice or press **ENTER**. In this way the editing mode is entered and it is then possible to operate like for the **Insert** command completion.
- insert either a new program line or a complex statement - it is needed to use **Insert** command. It causes the **PDL2 Keyboard** to be opened.
- deleting either an already existing program line or a complex statement - the use of **Delete line** command is needed, which is part of the **Edit** menu.



If executing the current statement is required, press the **START (green)** key and, if it is a motion statement, press the **Enabling Device** too.

Such a statement is then executed according to the current settings (see [par. 5.15.3.1.8 Step on page 137](#)).

EXECUTION CURSOR

It is represented by an arrow and the statement background that is a different colour depending on the current status of the statement:

- if the line background is **dark green**, the pointed statement is the being executed one;
- if the line background is **light green**, the pointed statement is the next statement to be executed;
- if the line background is **yellow**, the system is in AUTO mode and the program is in HOLD state;
- if the system is executed a protected routine, instead of the arrow that points to the statement there is an icon identified by **EXT** text string.

C: prog_29	S: stmt	T:
PROGRAM prog_29 EZ, PROG_ARM		
VAR pnt0001p, pnt0002p, pnt00		
ROUTINE call_prog_29 EXPORTED		
ROUTINE call_prog_29		
BEGIN		
MOVE TO pnt0001p		
MOVE JOINT TO pnt0002p		
MOVE JOINT TO pnt0003p		
MOVE JOINT TO pnt0004p		
END call_prog_29		
BEGIN		
call_prog_29		
END prog_29		

C: prog_29	S: stmt	T:
PROGRAM prog_29 EZ, PROG_ARM		
VAR pnt0001p, pnt0002p, pnt00		
ROUTINE call_prog_29 EXPORTED		
ROUTINE call_prog_29		
BEGIN		
MOVE TO pnt0001p		
MOVE JOINT TO pnt0002p		
MOVE JOINT TO pnt0003p		
MOVE JOINT TO pnt0004p		
END call_prog_29		
BEGIN		
call_prog_29		
END prog_29		

C: prog_29	S: dsbl	T:
PROGRAM prog_29 EZ, PROG_ARM		
VAR pnt0001p, pnt0002p, pnt00		
ROUTINE call_prog_29 EXPORTED		
ROUTINE call_prog_29		
BEGIN		
MOVE TO pnt0001p		
MOVE JOINT TO pnt0002p		
MOVE JOINT TO pnt0003p		
MOVE JOINT TO pnt0004p		
END call_prog_29		
BEGIN		
call_prog_29		
END prog_29		

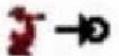
MOTION CURSOR

It indicates the motion statement the Arm is currently executing; if there is no movement in progress, it points to the last executed motion statement. The associated icon is that of a robot, plus integrating symbols that indicate the motion status:

- the robot is on the trajectory but the movement is not completed yet



- the robot has reached the final position



- the robot is out of trajectory



- the current movement has been aborted.



Displaying the currently being executed statement

If the opened program is running, IDE displays the currently being executed statements, by means of the **EXECUTION CURSOR**. The execution may be followed by the user by looking at the cursors moving through the program statements.

At program activation in IDE environment, if the program is not being executed, the **EXECUTION CURSOR** is placed on the first executable statement after the BEGIN statement.

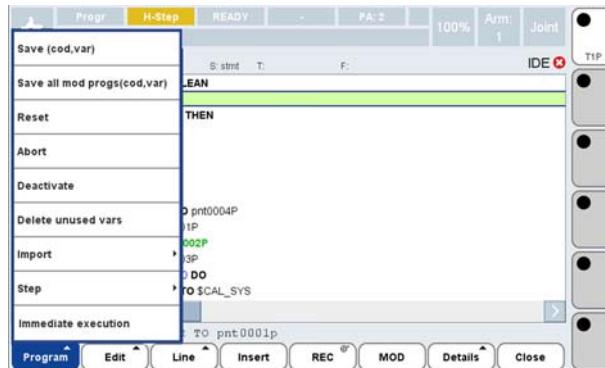
5.15.3 Available functions description

In IDE Page, the **Functional keys menu** available commands are as follows:

- Program
- Edit
- Line
- Insert
- REC
- MOD
- Details
- Close.

5.15.3.1 Program

Allows operating on the currently open program.



As shown in the figure above, the following functions are available:

- [Save \(cod, var\)](#)
- [Save all mod progs \(cod,var\)](#)
- [Reset](#)
- [Abort](#)
- [Deactivate](#)
- [Delete unused vars](#)
- [Import](#)
- [Step](#)
- [Immediate execution.](#)

5.15.3.1.1 [Save \(cod, var\)](#)

Saves both the .COD and the .VAR for the currently open program, in the directory from which it was downloaded; the program remains open.

5.15.3.1.2 [Save all mod progs \(cod,var\)](#)

Saves all the changed programs in IDE environment, both .COD and .VAR.

5.15.3.1.3 [Reset](#)

Deactivates and activates the current program again.

5.15.3.1.4 [Abort](#)

Aborts the current statement execution.

5.15.3.1.5 [Deactivate](#)

Deactivates the current program.

5.15.3.1.6 Delete unused vars

Deletes the variables which are not used by the program, from the declaration section, after the user confirmation.

5.15.3.1.7 Import

It allows to import the declarations from another program that have either the EXPORTED FROM clause or the GOLOBAL attribute, and insert them in the declaration section of the currently open program in IDE environment.



If either a data type, or a variable, or a routine have the GLOBAL attribute, IDE environment inserts only one program line like the following:

```
IMPORT '<name of the program from which importing>'
```

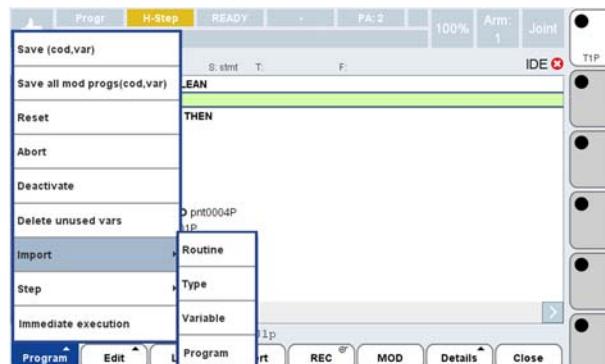
Otherwise, explicit declaration is inserted as it is, like in the program from which it is imported.

Example:

importing *kk* variable from *prog_1* program, **Import** command will insert the following:

```
TYPE gg=record
    i:integer
ENDRECORD

var kk:gg exported from prog_1
```



The classes of importable declarations are:

- Routine
- Type
- Variable
- Program.

When **Program** item is selected, a list is displayed of all programs currently loaded in memory. The User has to choose the wished program. This causes the IMPORT statement to be inserted in the currently open program in IDE environment.

Such a choice allows to import from the chosen program, without declaring any of them, all the objects (types, variables, routines) having the GLOBAL attribute.

Example

If **Program** command is selected and *prog_2* is chosen from within the programs list, IDE inserts the following statement in the currently open program:

```
IMPORT 'prog_2'
```

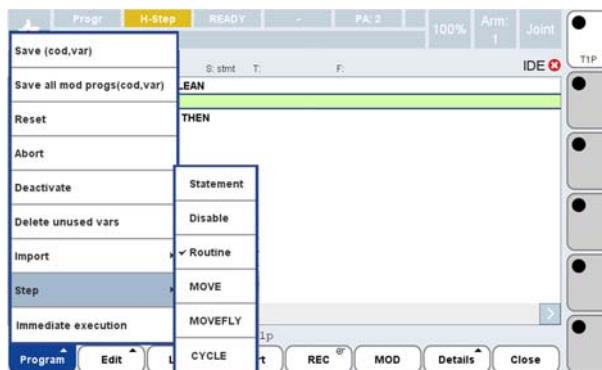


For further information about IMPORT statement and GLOBAL attribute, see [PDL2 Programming Language manual - par. IMPORT Statement](#).

5.15.3.1.8 Step

Sets the program execution step mode.

The current type of step is indicated by a tick (in the below example the current step type is **Routine**).



The available commands are:

- [Statement](#)
- [Disable](#)
- [Routine](#)
- [MOVE](#)
- [MOVEFLY](#)
- [CYCLE](#).

Carefully read the [Operational note](#) about the use of **SHIFT+START** keys, for a special functionality of the program controlled execution.

Statement

The execution stops after each statement. The protected Routines (whose content is not displayed) are executed as a single statement; the execution of non-protected Routines (whose the content is visible) stops at the end of each statement.

Disable

Stops the program execution in steps, causing it to be continuous.

Routine

The execution stops at the end of the routine (the routine is always completely executed: it never stops on any statement inside the routine itself).

MOVE

The execution stops after each single movement. It cannot be executed on non-holdable programs.

MOVEFLY

It causes to execute two or more fly movements before suspending the program execution. It is similar to **MOVE** option, but the execution does not stop after the MOVEFLY statement. It cannot be executed on non-holdable programs.

CYCLE

The chosen step mode is a single program cycle, that has to include the CYCLE or BEGIN CYCLE statement.

Operational note



SHIFT+START

Pressing both the two keys at the same time, causes the program to be executed in **Disable** type, which means continuous execution, irrespectively of the currently set **Step**. The **SHIFT** key can be released as soon as the execution starts: pressing the **START** key is enough.

As soon as the execution is either completed or the **START** key is released, the previously set **Step** mode is resumed.

This special behaviour can be used to modify positions along FLY movements, executing the whole program or just a part of it, without having the needs to change the **Step** setting again and again:

- let's suppose to be in Statement mode
- press **SHIFT+START** to move in continuous mode
- release the **START** key some program lines before reaching the being modified position, causing the robot to stop
- press the **START** key again to resume motion and move towards the being modified position.



WARNING - When the **START** key is released during the execution of FLY movements, the destination at the motion resume time depends on the moment in which the key has been released.

If it happened AFTER starting the FLY segment, the resumed movement will stop on P3 final position.

If it happened BEFORE starting the FLY segment, the resumed movement will stop on P2 intermediate position.

Such two scenarios are shown, respectively, in the following [Fig. 5.31](#) and [Fig. 5.32](#).

Fig. 5.31 - Motion cancelled BEFORE the beginning of FLY segment

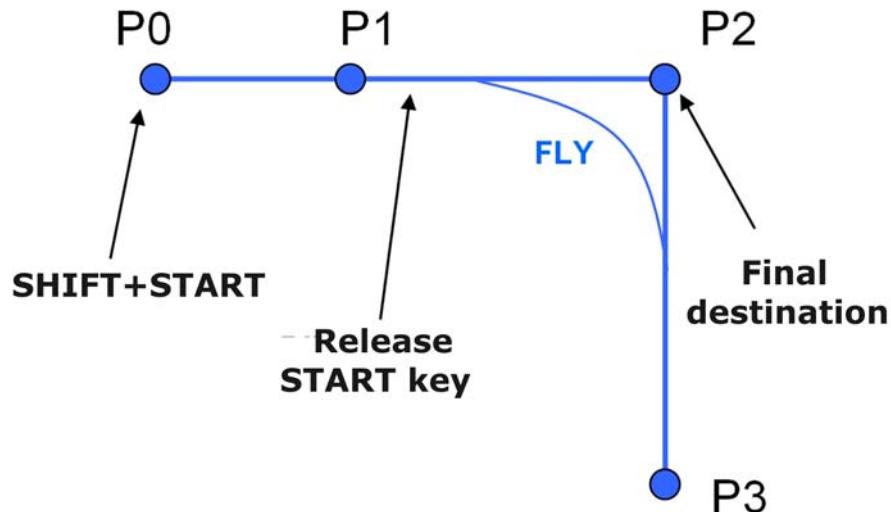
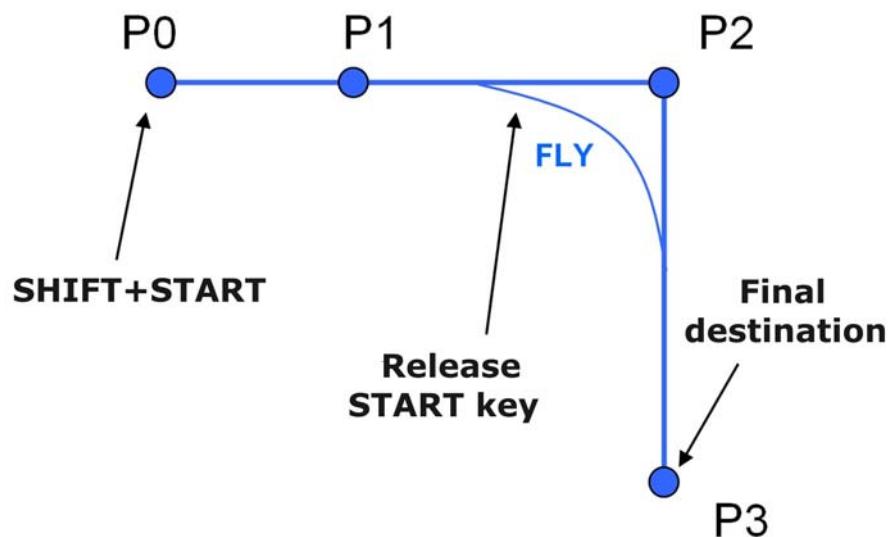
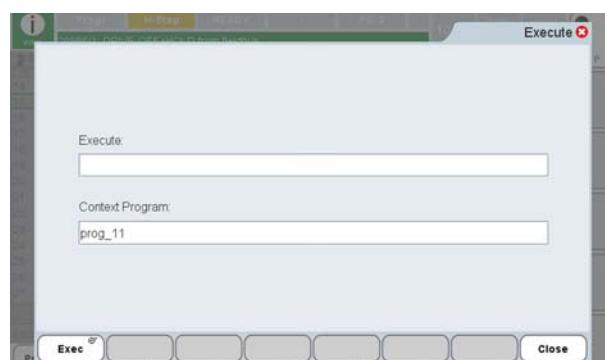


Fig. 5.32 - Motion cancelled AFTER the beginning of FLY segment



5.15.3.1.9 Immediate execution

It allows executing a statement which is not included in the current program body (but anyway taking effect on its execution: e.g. by immediately executing an assignment of a value to a variable belonging to the current program, such a variable content is actually modified).



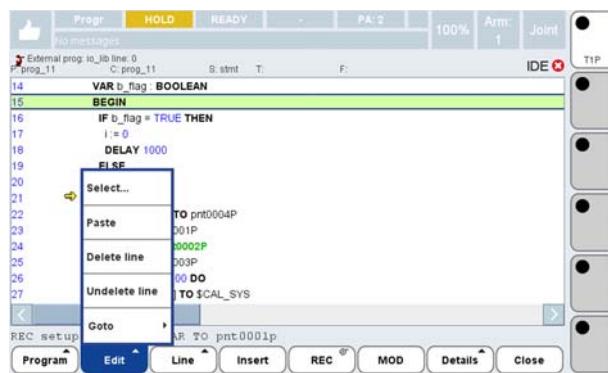
The system opens the shown above page, containing the field in which the being executed single statement must be inserted. Touching such a field, the **PDL2 keyboard** is opened for choosing the wished instruction. Refer to [par. 5.6.3.3 PDL2 keyboard on page 79](#) for a detailed use description.

After selecting the wished instruction, either touch **OK** or press **ENTER**, to confirm it.

Then touch **Exec** key to start the execution.

As already mentioned, IDE allows monitoring, in the **Editor area**, the currently open program execution (see [Displaying the currently being executed statement](#))..

5.15.3.2 Edit



This key allows accessing the following commands:

- [Select](#)
- [Paste](#)
- [Delete line](#)
- [Undelete line](#)
- [Goto](#)

5.15.3.2.1 Select

Activates marking the lines which the **EDIT CURSOR** is moving on. In the **Functional keys menu** the following commands become available:

- **Cut** - cuts the selected lines
- **Copy** - copies the selected lines
- **Deselect** - deactivates marking.



Selecting several program lines is allowed by means of combining [SHIFT+cursor](#).

5.15.3.2.2 Paste

Allows pasting the previously cut or copied lines, inserting them below the program line which is currently pointed by the **EDIT CURSOR**.

5.15.3.2.3 Delete line

Deletes the program line which is currently pointed by the [EDIT CURSOR](#).



If the selected instruction is a structured one (e.g. IF, FOR, etc.), this command deletes the whole structure and NOT only the selected line (from IF to ENDIF, from FOR to ENDFOR, from WHILE to ENDWHILE, etc).

5.15.3.2.4 Undelete line

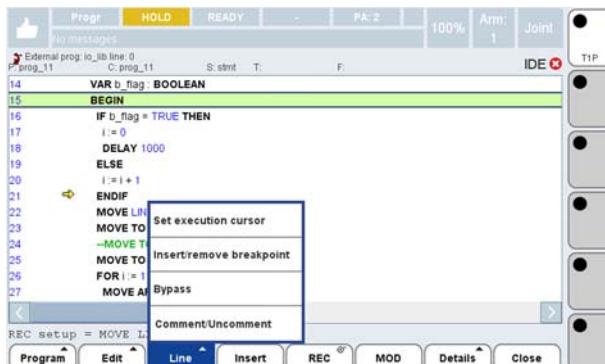
Inserts the last deleted statement again, in the program, at the line pointed by the [EDIT CURSOR](#).

5.15.3.2.5 Goto

Allows the [EDIT CURSOR](#) to be moved to the specified line:

- **Begin** - moves the cursor to the current context BEGIN statement
- **End** - moves the cursor to the current context END statement
- **Line** - moves the cursor to the specified program line (within the current context). The [Numeric keyboard](#) is opened to ask the user to insert the wished line number.

5.15.3.3 Line



This key allows accessing the following commands:

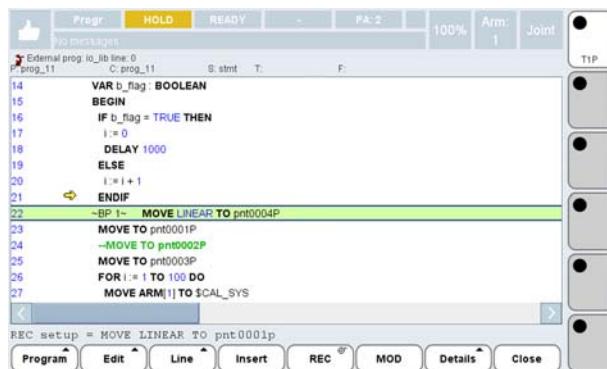
- [Set execution cursor](#)
- [Insert/remove break point](#)
- [Bypass](#)
- [Comment/Uncomment](#).

5.15.3.3.1 Set execution cursor

Allows moving the program execution to a different line from the one actually pointed by the [EXECUTION CURSOR](#). Move the [EDIT CURSOR](#) and touch this command.

5.15.3.3.2 Insert/remove break point

Inserts or removes a break point (execution interruption point) within the program.



```

External prog io.lib line: 0
C: prog_11 S: stmt T: F: IDE
14 VAR b_flag : BOOLEAN
15 BEGIN
16 IF b_flag = TRUE THEN
17   i := 0
18   DELAY 1000
19 ELSE
20   i := i + 1
21 ENDIF
22 ~BP 1~ MOVE LINEAR TO pnt0004P
23 MOVE TO pnt0001P
24 ~MOVE TO pnt0002P
25 MOVE TO pnt0003P
26 FOR i := 1 TO 100 DO
27   MOVE ARM[1] TO $CAL_SYS
28
29
30
31
32
REC setup = MOVE LINEAR TO pnt0001P

```

The break point is positioned where the **EDIT CURSOR** currently is. The program execution will be interrupted before the line identified by such a break point.

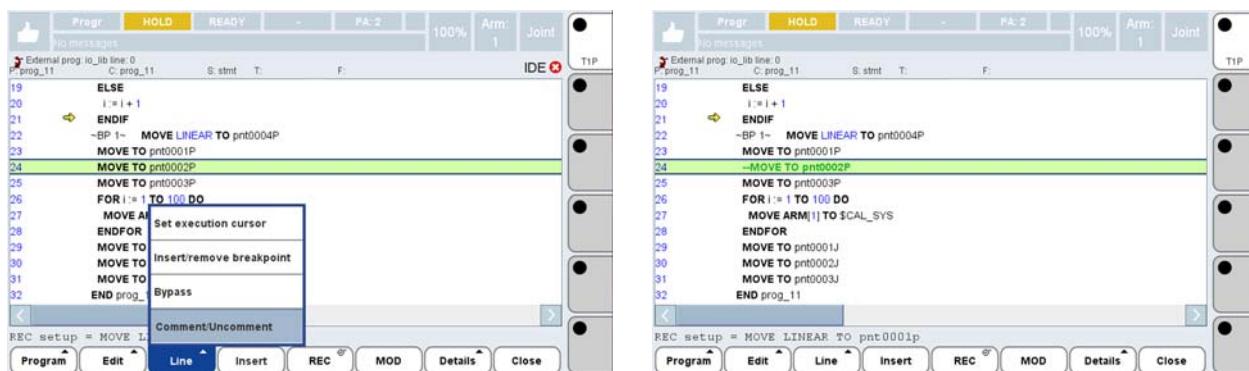
To remove a break point, simply select the line where it has been set, and use this command again.

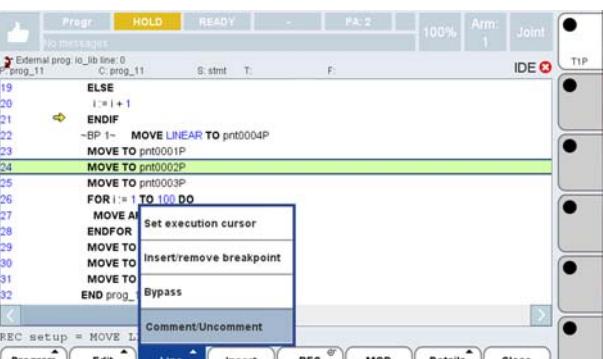
5.15.3.3.3 Bypass

Allows the program execution to continue, in case the program is waiting for the completion of a suspending instruction such as WAIT FOR <condition>, DELAY <time>, etc.

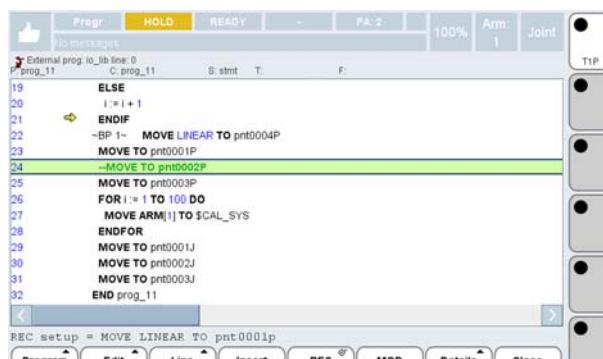
5.15.3.3.4 Comment/Uncomment

Allows the current program line to become a "comment" or to be considered as an executable statement, by putting or removing the comment symbol ('--'). As already mentioned, the commented line is displayed in green (see the below figure on the right).





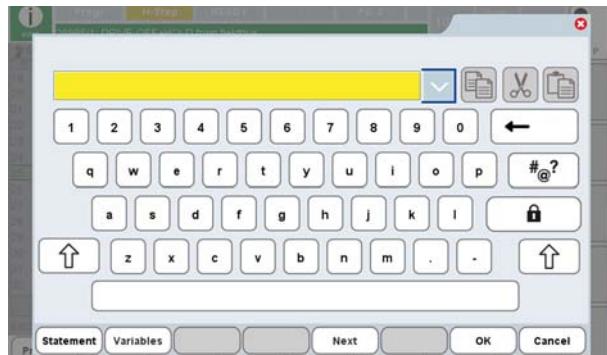
Comment/Uncomment



Comment/Uncomment

5.15.3.4 Insert

Allows inserting a new PDL2 language program line, in the currently open program. Touching this key, causes the system to immediately activate the **PDL2 keyboard**.

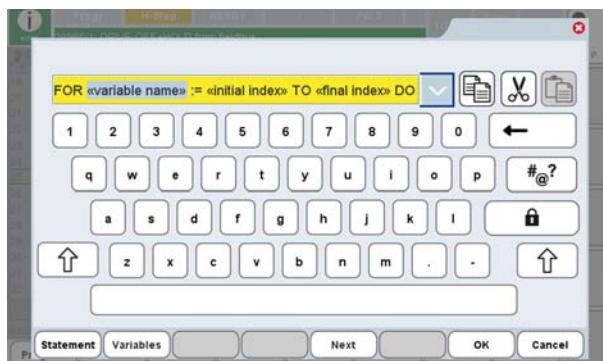


Refer to [par. 5.6.3.3 PDL2 keyboard on page 79](#) for detailed descriptions of such a keyboard as well as **Statement**, **Variables**, **Values** and **History** commands.

Guided insertion of a generic statement

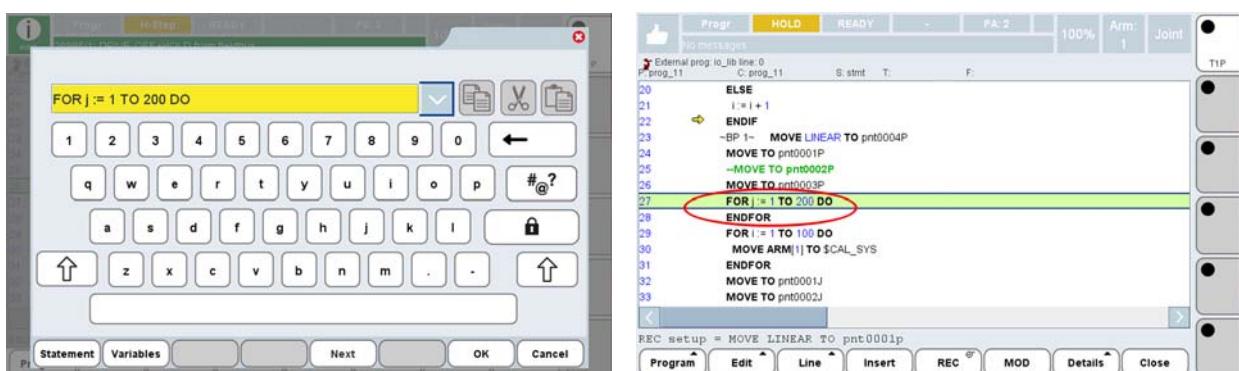
Let's insert a new program line containing the FOR instruction.

After selecting the wished statement type (by means of the **Statement** key), confirm it (either **OK** or **ENTER**) to prepare its insertion.



The system displays the corresponding template of the chosen statement (in our example the FOR instruction - see figure above).

The user is requested to complete all the variable fields (moving among them by means of either using the **Next** key, or touching them), then to confirm the statement insertion by either touching **OK** or pressing **ENTER**.



The system checks that the syntax is correct; if the inserted statement is not correct, an error message is displayed.

After the error has been fixed, the statement has to be confirmed again by either touching **OK** or pressing **ENTER**.

If the instruction is correct and complete, its insertion is terminated.

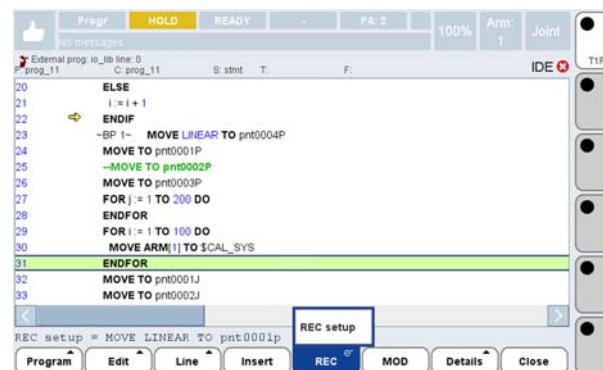
Note that the system has also inserted the **ENDFOR** line (see the previous figure on the right), to complete the structure of the chosen instruction.

5.15.3.5 REC

Short touching this key, the following items are inserted in the program:

- a **motion instruction** and
- the **declaration of the corresponding positional variable** that is assigned the current arm position.

Long touching this key, the REC setup environment is activated (as shown in the figure below).



Its current settings are displayed in the line above the **Functional keys menu** of the IDE Page.

5.15.3.5.1 REC setup

Touching the **REC setup** item, a dedicated environment is opened allowing to setup all the peculiar features of the motion instruction which is inserted each time the **REC** key is touched.



When opening IDE Page, the **REC** key is set to modal joint, if it has not been previously setup.

The user has to define

- the motion instruction type (MOVE or MOVEFLY)
- the trajectory type (JOINT, LINEAR, CIRCULAR)

- the being inserted positional variable type (JOINTPOS, POSITION, XTNDPOS).

Also the following functions can be set for synchronized motion:

- whether the SyncMove clause it to be used or not
- the possibility to write the move instruction on more than one line, i.e. included between the key words MOVE and ENDMOVE. To enable/disable this function, select or unselect the Multi-line checkbox
- the trajectory type (JOINT, LINEAR, CIRCULAR)
- the being inserted positional variable type (JOINTPOS, POSITION, XTNDPOS).



NOTES TO TEACH POSITIONS IN MOVE..SYNCMOVE

- to be able to learn positions related to both Arms, referred by the MOVE ... SYNCMOVE, with the IDE session open on the current program, it is necessary to insert in the program, and run the following statement

ATTACH Arm[n]

for both the involved Arms in the MOVE ... SYNCMOVE statement. The execution of such a statement must take place **BEFORE** teaching any point, otherwise, when the Arm is changed, if the Arm selected by the Teach Pendant is different than the PROG_ARM, the IDE session will be closed by the system.

Example:

If the program uses

MOVE Arm[2] ...SYNCMOVE Arm[1],

at the **BEGINNING** of the program it is necessary to insert the following statement

ATTACH Arm[1],Arm[2].

- To be able to teach the involved Syncmove positions, it could be needed to change the current Arm, in order to be able to refer to the Arm of the wished being taught position. Usually, pressing the **ARM** key once, could (in case of a 3 or 4 Arms system) select an Arm which is different from either the PROG_ARM or the one involved in the ATTACH statement; the currently open program is thus closed ('No programs available in IDE' error occurs) and suitable questions are made to the user about saving code and/or variables. In order to avoid such a situation, it is recommended to **KEEP PRESSED** the **ARM** key until the wished Arm is reached (indicated in the **Status bar**). Then release the **ARM** key.

Finally, touch the **OK** key to confirm the made settings.



Note that the new REC settings are NOT kept upon a system restart. To make them permanent, a Configure Save operation is needed ([Configure](#) in [Setup page](#)).

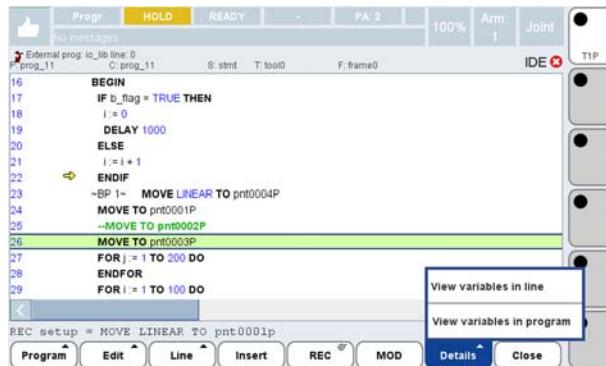
5.15.3.6 MOD

Allows modifying the coordinates value of an existing position. Upon touching this key, the system displays a dialogue window in which the user is prompted to confirm the selected variable name.



If tool and frame which are set for the current MOVE are different than the ones of [REC setup](#), the system will refuse the change.

5.15.3.7 Details



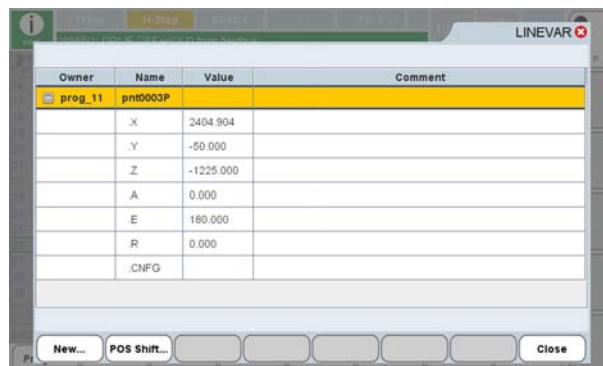
This key allows accessing the following commands:

- [View variables in line](#)
- [View variables in program](#).

5.15.3.7.1 View variables in line

Displays all the variables that are on the program line currently indicated by the **EDIT CURSOR**. The information of their values is refreshed in realtime.

In our example, issuing this command causes the following page to be displayed:



The following commands are available in the [Functional keys menu](#):

- **New..** - allows creating a new variable to be used in the program and adding its declaration
- **POS SHIFT...** - it is just available when the selected variable is of **POSITION** and/or **XTNDPOS**, like in our example, and allows ‘moving’ such positions in the space.

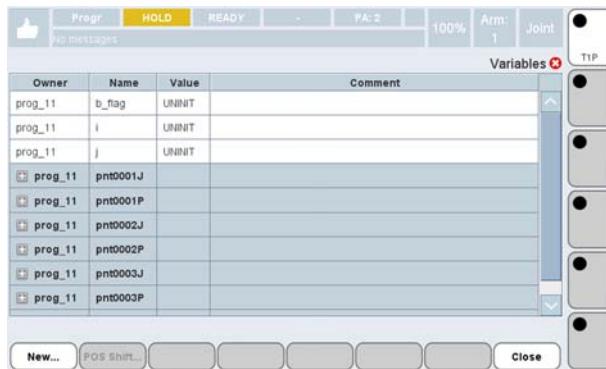


As far as the POS SHIFT key functionality, refer to [par. POS SHIFT on page 127](#), available in [Prog Page](#) chapter.

5.15.3.7.2 View variables in program

Displays all the variables belonging to the currently open program. The information of their values is refreshed in realtime.

In our example, issuing this command causes the following page to be displayed:



The following keys are available in the [Functional keys menu](#), as for previous [View variables in line](#) command:

- **New...** - allows creating a new variable to be used in the program and adding its declaration
- **POS SHIFT...** - it is just available when the selected variable is of **POSITION** and/or **XTNDPOS**, like in our example, and allows 'moving' such positions in the space.



As far as the POS SHIFT key functionality, refer to par. [POS SHIFT on page 127](#), available in [Prog Page](#) chapter.

5.15.3.8 Close

Touch this key to close IDE Page: the program remains in the state it was in before the closure.

If there is a misalignment between the **EDIT CURSOR** and the **EXECUTION CURSOR**, the user is asked if the execution is to be continued from the edit cursor position. It is only possible to exit from the environment if the reply is affirmative. Otherwise it is necessary to realign the cursors by hand before closing.

If the program has been changed, the user is asked if it is to be saved.

5.16 Display Page

This environment DISPLAYS information about the following elements:

- existing I/O points, associated or not to the Devices which are present in the System,
- Program and System Variables.

The user is allowed to combine such [Elements](#) as wished, depending on how many and which of them the user would like to view.

The available [Commands](#) for this environment, allow the user to create several combinations of elements, save them to different files in the **Saved** folder and then use them in the appropriate time.

The main screen allows accessing all the [Elements](#), to select the ones the user wish to display.

Furthermore, it allows opening **Saved** folder, containing the saved files.



5.16.1 Elements

This section describes all subenvironments including all the viewable elements (among which the user can choose) and the available commands to handle them:

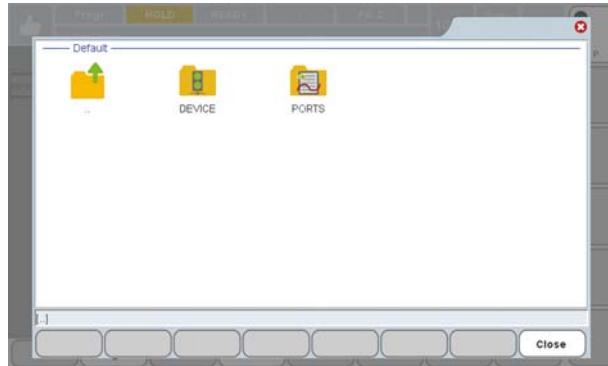
- [IO](#)
- [Vars](#).

Refer to [par. 5.16.2 Commands on page 153](#) to group the wished elements to the user's own liking and to save such groupings in different files.

5.16.1.1 IO

This environment displays the following elements existing in the System:

- the **Devices** ([Device](#)) families,
- the **I/O Ports** ([Ports](#)).



5.16.1.1.1 Device

This environment displays the existing Devices families in the System.

The device (either physical or virtual) **name**, **type** and **state** (connected, not connected, disabled), are indicated in the lowest part of the screen, above the [Functional keys menu](#), highlighted in red in the following figure.



Note that the icons, representing the Devices **State**, are refreshed in realtime. The C5G icon represents the Primary Slave.

Touching once the icon associated to a certain device, the following keys are made available in the [Functional keys menu](#):

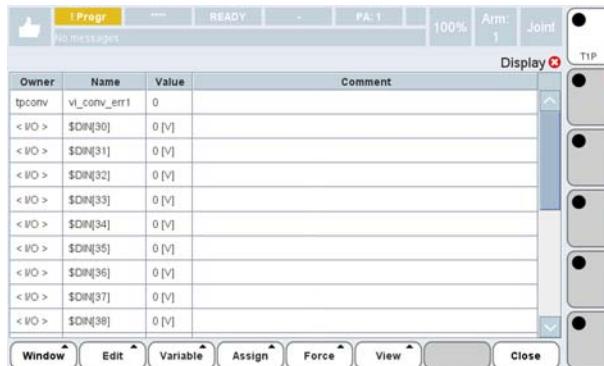
- [Device](#)
- [Remote Inp.](#)
- [Properties.](#)

Touching twice (which means touching again an already selected icon) instead, the icon associated to a certain device, causes to enter the associated I/O ports display page. Refer to [par. 5.16.2.2 Handling I/O points value on page 156](#) for all commands to handle individual I/O ports (assign value, force value, etc).

Device

This functional key is available for **User devices**, i.e. belonging to either a Profibus, DeviceNet, Profinet, Ethernet/IP type Fieldbus, or **Virtual devices**.

Touch **Device** and choose **Disable** item, to disable the selected device.

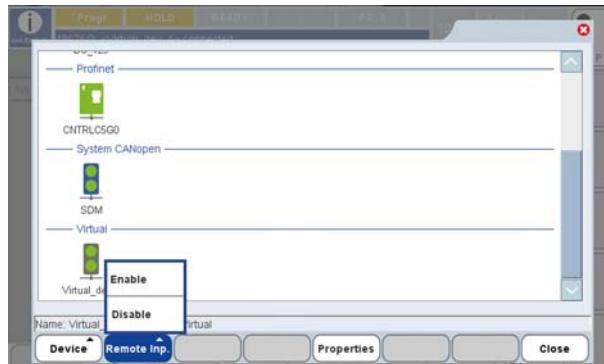


On the contrary, touch **Device** key and choose **Enable** item, to enable a disabled device.

Remote Inp.

This functional key is only available when the selected device is a primary slave; it allows enabling/disabling the use of the following signals from remote: START, HOLD, DRIVE ON, DRIVE OFF.

A menu is opened as soon as such a key is touched, in which the user can choose between either **Enable** or **Disable** command.



If **Disable** command is chosen, the system will not take the listed above signals (such signals only), coming from PLC, into account. Obviously, the system will take them into account again as soon as the **Enable** command is issued.



Note that when the remote signals are disabled, in the first field of the **Status bar**, an exclamation mark '!' is displayed before the current status (either PROGR or LOCAL).



Properties



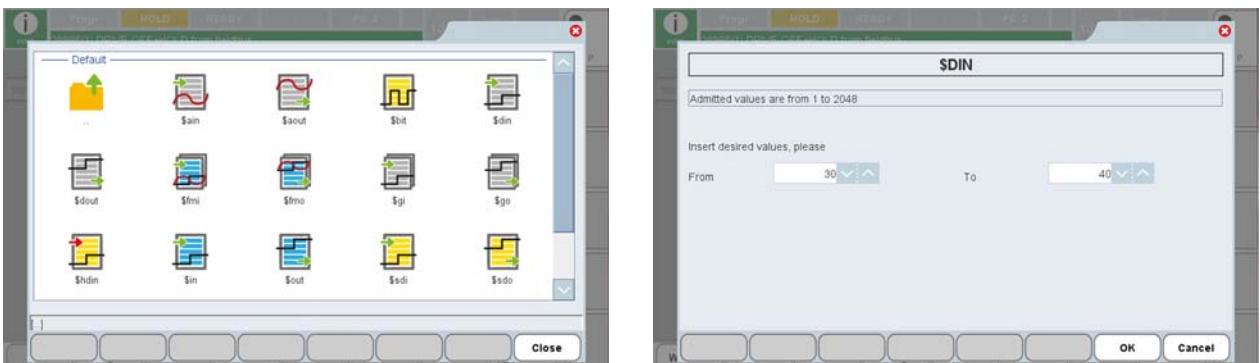
By means of **Properties** key, a sub-page is displayed showing additional information about the currently selected Device (see above figures).

5.16.1.1.2 Ports

This sub-page allows displaying all existing ports in the System (below figure on the left), grouped by port type.

Touching the wished icon, depending on the required port type, the system could display a screen in which the user is asked to specify the range of the being displayed ports (as shown in the following example, figure on the right).

Specify the range by means of the suitable keyboard which is activated by the system.



Touch **OK** key to confirm the choice and to get the required ports list to be displayed, with the following information:

- **Owner** - indicates whether such a row information is associated to an I/O type element or to a program variable element; in that case the owning program name is displayed
- **Name** - element name
- **Value** - element current value
- **Comment** - element description.

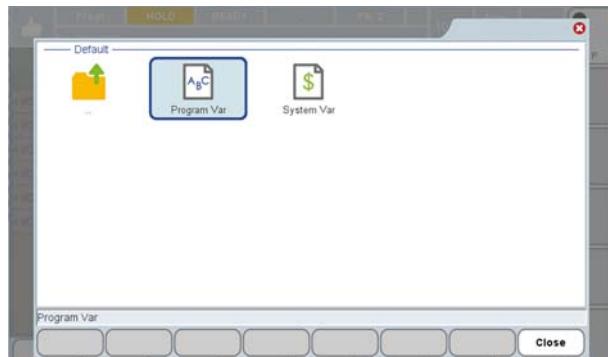
Owner	Name	Value	Comment
< I/O >	\$DIN[30]	0 [V]	
< I/O >	\$DIN[31]	0 [V]	
< I/O >	\$DIN[32]	0 [V]	
< I/O >	\$DIN[33]	0 [V]	
< I/O >	\$DIN[34]	0 [V]	
< I/O >	\$DIN[35]	0 [V]	
< I/O >	\$DIN[36]	0 [V]	
< I/O >	\$DIN[37]	0 [V]	
< I/O >	\$DIN[38]	0 [V]	
< I/O >	\$DIN[39]	0 [V]	

Refer to [par. 5.16.2 Commands on page 153](#) for all commands to handle each single I/O port.

5.16.1.2 Vars

This subenvironment displays

- variables belonging to PDL2 programs currently loaded in memory (**Program Var**),
- a subset of PDL2 language predefined variables (**System Var**).



Different selection pages are opened, in which the user should insert the needed information, depending on the wished variable.

In the below figures, the left one refers to **Program Var** choice, whereas the right one refers to **System Var** choice.

<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> Program: <input type="text"/> Variable: <input type="text"/> Index: <input type="text"/> <input type="text"/> Data type: <input type="text"/> Comment: <input type="text"/> </div>	<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> Variable: <input type="text"/> Index: <input type="text"/> <input type="text"/> Arm: <input type="text"/> Comment: <input type="text"/> </div>
---	--

The system displays a list of available data (**Program** or **Variable**), for each field. The user should choose among them.

5.16.2 Commands

In the current section the available commands are described, corresponding to the currently displayed elements.

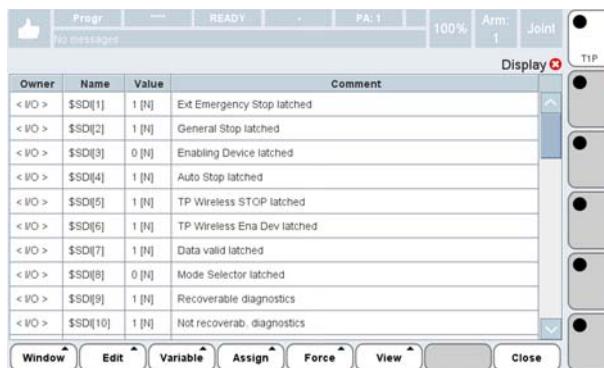
When an I/O points and/or Variables list is displayed, the system makes some commands available to

- create and save the user's own groups of [Elements - Handling customized viewing](#)
- act on the I/O points values - [Handling I/O points value](#).



Note that when one or more elements are displayed, their values are realtime refreshed.

Fig. 5.33 - Commands



Owner	Name	Value	Comment
< I/O >	\$SDI[1]	1 [N]	Ext Emergency Stop latched
< I/O >	\$SDI[2]	1 [N]	General Stop latched
< I/O >	\$SDI[3]	0 [N]	Enabling Device latched
< I/O >	\$SDI[4]	1 [N]	Auto Stop latched
< I/O >	\$SDI[5]	1 [N]	TP Wireless STOP latched
< I/O >	\$SDI[6]	1 [N]	TP Wireless Ena Dev latched
< I/O >	\$SDI[7]	1 [N]	Data valid latched
< I/O >	\$SDI[8]	0 [N]	Mode Selector latched
< I/O >	\$SDI[9]	1 [N]	Recoverable diagnostics
< I/O >	\$SDI[10]	1 [N]	Not recoverab. diagnostics

5.16.2.1 Handling customized viewing

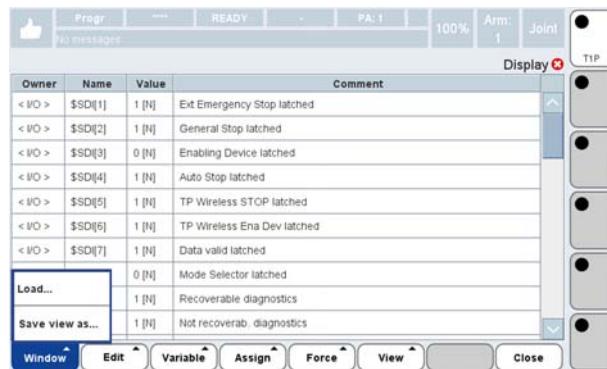
Commands to create, modify, save customized viewing files, as well as adding/removing elements, are as follows (see [Fig. 5.33](#)):

- [Window](#)
- [Edit](#)
- [Variable](#)
- [Assign](#)
- [Force](#)
- [View](#).

5.16.2.1.1 Window

Touch this key to access commands acting on files contained in the **Saved** folder:

- [Load...](#)
- [Save view as...](#)



Load...

Opens the **Saved** folder content.

Touch the wished file icon, to view the elements contained in it.

In the below example (figure on the right) the content of *my_disp* file is displayed, available to the user for viewing it, editing it, saving it again, etc.

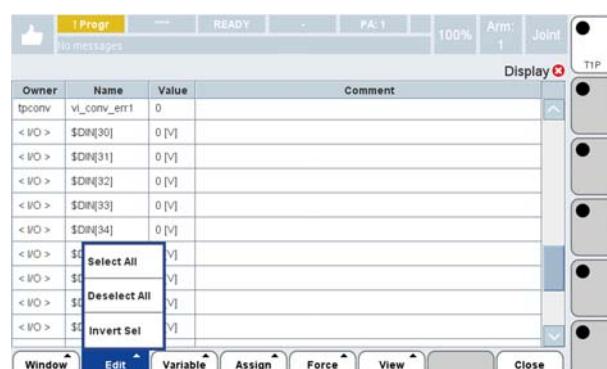


Save view as...

This command allows saving the currently displayed group of elements to the **Saved** folder. The filename must be specified by the user by means of the automatically opened keyboard.

A suitable message informs the user when the file is properly saved.

5.16.2.1.2 Edit



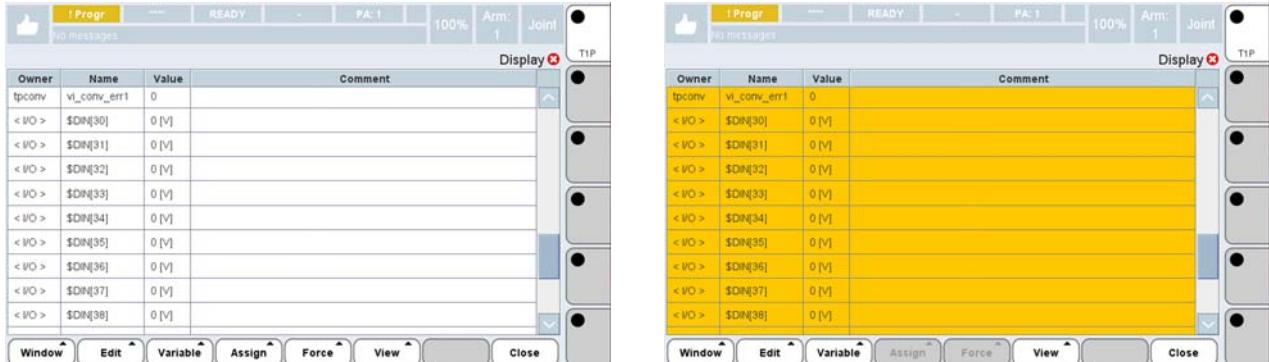
This key opens a menu to handle the being viewed elements selection.

The available commands are as follows:

- [Select All](#)
- [Deselect All](#)
- [Invert Sel.](#)

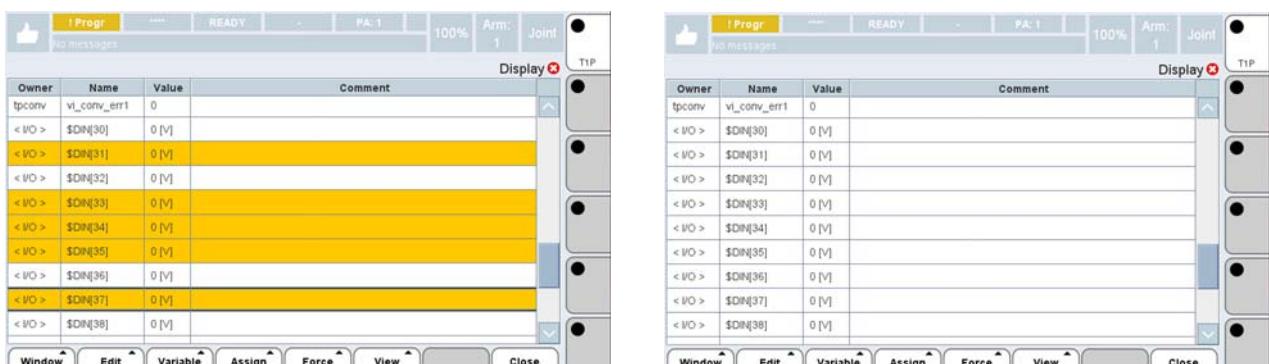
Select All

Choose this command to select all the currently displayed elements.



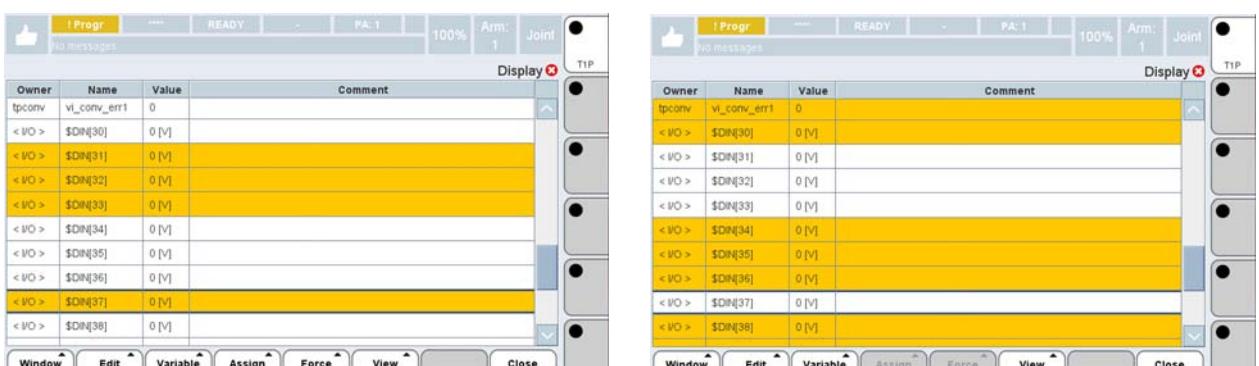
Deselect All

Choose this command to unselect all the currently selected elements.

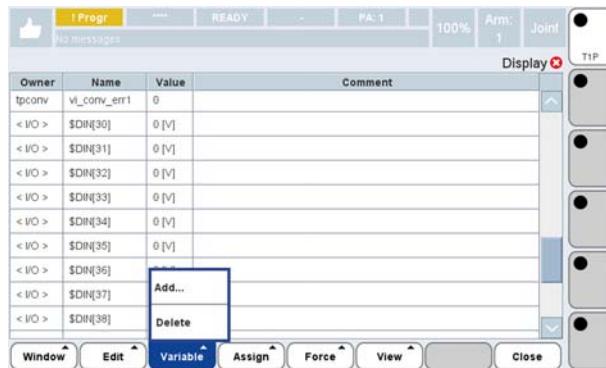


Invert Sel

Choose this command to unselect all the currently selected elements and select the currently not selected ones.



5.16.2.1.3 Variable



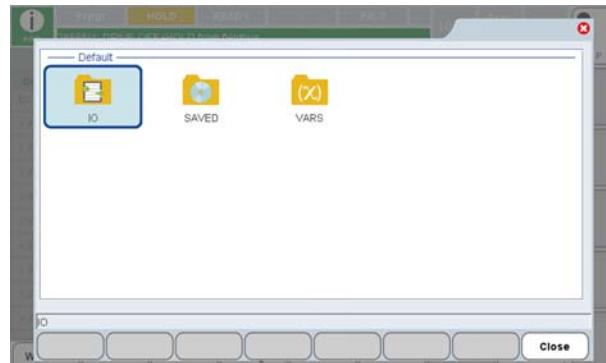
This key opens a menu to handle the elements to be viewed and later saved to a file.

The available commands are as follows:

- [Add...](#)
- [Delete](#).

Add...

When this command is selected, the system opens the Display environment home Page (see figure below).



Thus the user can choose the wished environment in which selecting the element(s) to be inserted in his/her customized group of elements:

- I/O points
- already existing file
- program or system variable.

The new elements choice is then added to the already currently displayed elements.

For further details, refer to [par. 5.16.3.2 Example on page 159](#).

Delete

This command deletes the currently selected elements.

5.16.2.2 Handling I/O points value

If some modifications are needed to I/O points values, the following commands are

available:

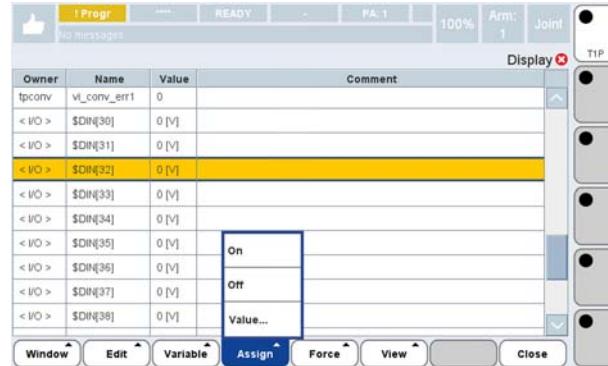
- [Assign](#)
- [Force.](#)

5.16.2.2.1 Assign

This command allows assigning a value to the selected I/O point.



This functionality is not available for \$AIN/\$AOUT ports.



The available choices are:

- [On](#)
- [Off](#)
- [Value...](#)

On

Sets the specified I/O point to TRUE.

Using SYS_CALL, this operation corresponds to

```
SYS_CALL ('E', '<nome_porta>[<indice_porta>] := ON')
```

Off

Sets the specified I/O point to FALSE.

Using SYS_CALL, this operation corresponds to

```
SYS_CALL ('E', '<nome_porta>[<indice_porta>] := OFF')
```

Value...

It is like the two preceding ones, but refers to I/O points composed by more than one bit.

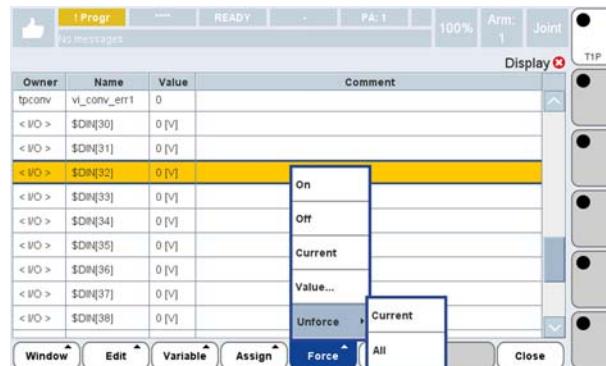
5.16.2.2.2 Force

This command forces the value of a certain I/O. The physical port is no longer considered and the forced value is used.

In order for such an I/O point to be able to assume the current value, the **Unforce** command must be used.



This functionality is not available for \$AIN/\$AOUT, \$GI/\$GO and \$FMI/\$FMO.



The available choices are:

- [On](#)
- [Off](#)
- [Current](#)
- [Value...](#)
- [Unforce.](#)

On

Forces the specified I/O point to TRUE.

Off

Forces the specified I/O point to FALSE.

Current

Forces the specified I/O point to its current value.

Value...

It is just like the preceding ones, but refers to I/O points composed by more than one bit.

Unforce

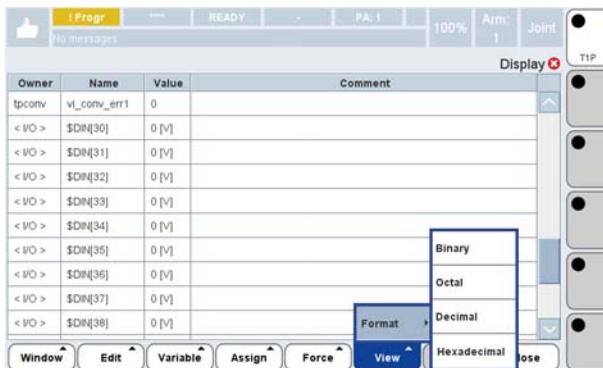
Removes the forced status for the selected I/O point, and restores its original physical configuration; the user must specify one of the following options:

- **Current** - removes the forced status for the selected I/O point, setting the current value
- **All** - removes the forced status of all I/O points of the selected type.

5.16.2.2.3 View

This command allows the user to specify the viewing format for the elements values:

- **Binary**
- **Octal**
- **Decimal**
- **Hexadecimal.**



5.16.3 Procedure and example

With reference to the previous sections, it is now described how to create and save a customized view.

A practical example is also provided.

- [Procedure](#)
- [Example.](#)

5.16.3.1 Procedure

The needed steps to create and save a customized [Elements](#) view, are as follows:

- a. display the first wished elements (either from [Vars](#) or [IO](#) subpages, or from an already existing file)
- b. add some more elements, as liking
- c. if needed, modify the group of elements, deleting one or more of them
- d. save the newly created or modified view to a file (either already existing or new).



Note that the files names are chosen by the user.

5.16.3.2 Example

Starting from the customized view contained in *view_start* file, add the following:

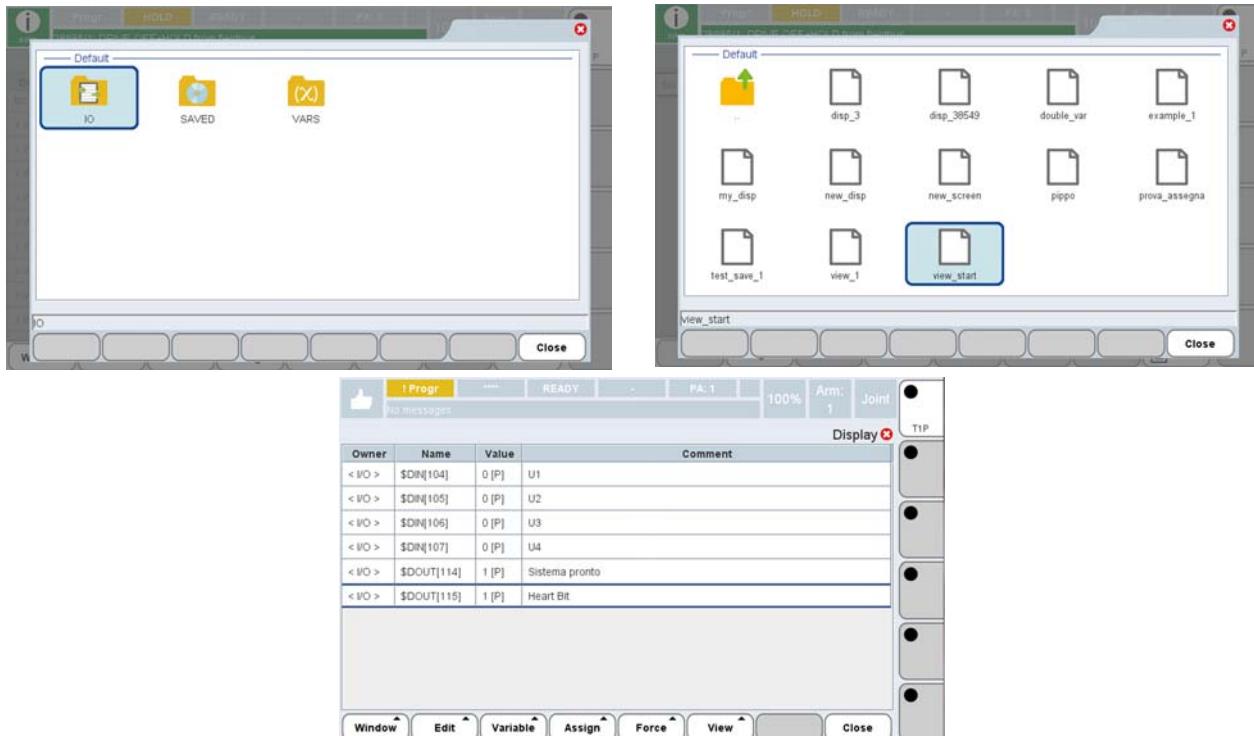
- I/O points from \$BIT[4] to \$BIT[8],

- the content of *view_1* file,
- *vb_save* variable, belonging to *CDetect* program.

Save the new customized view to a file named *new_screen*.

The required steps are the following:

- a. open *view_start* file in order to display the contained customized view

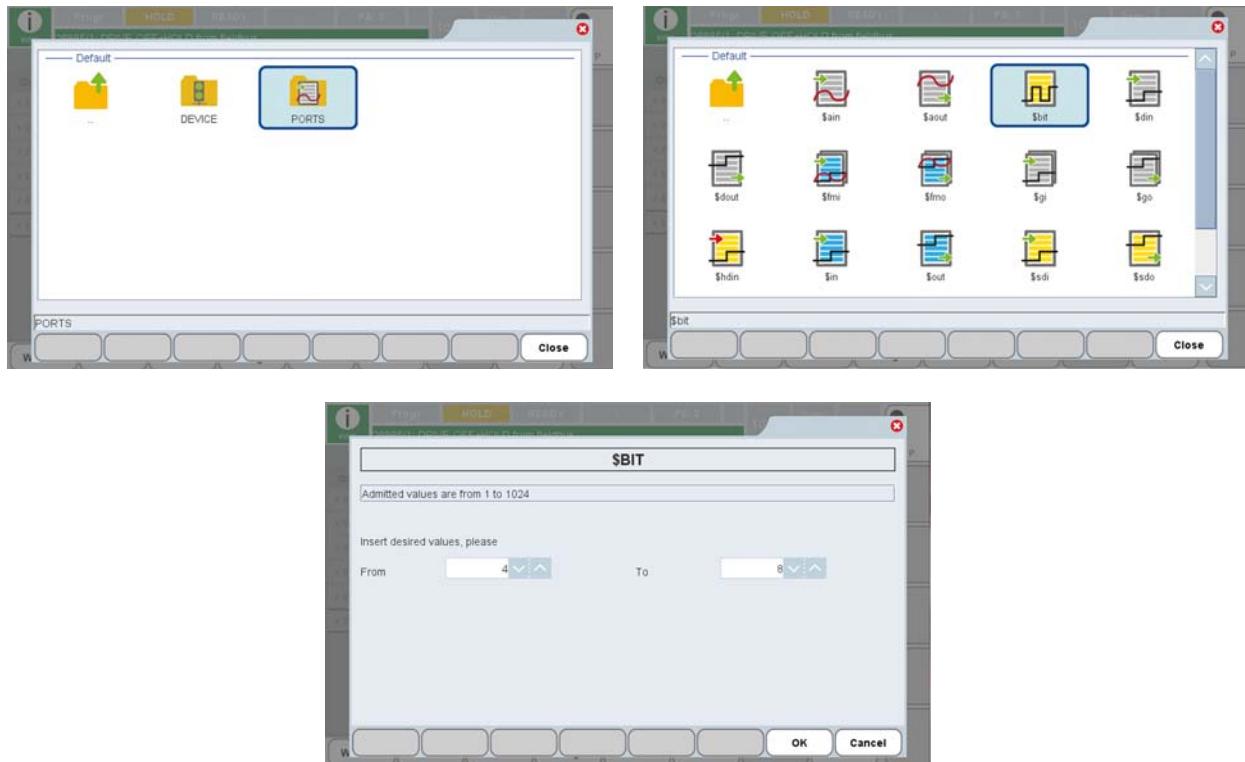


- b. touch **Variable** key and choose **Add...** item

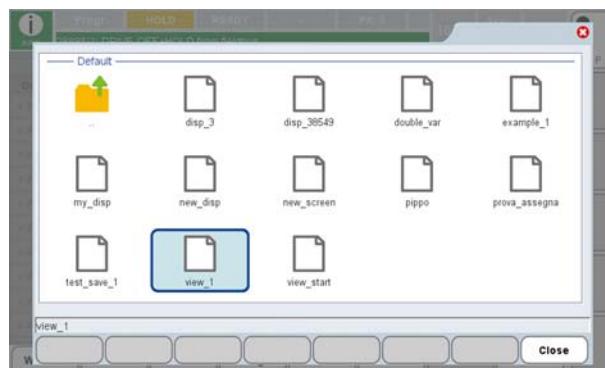
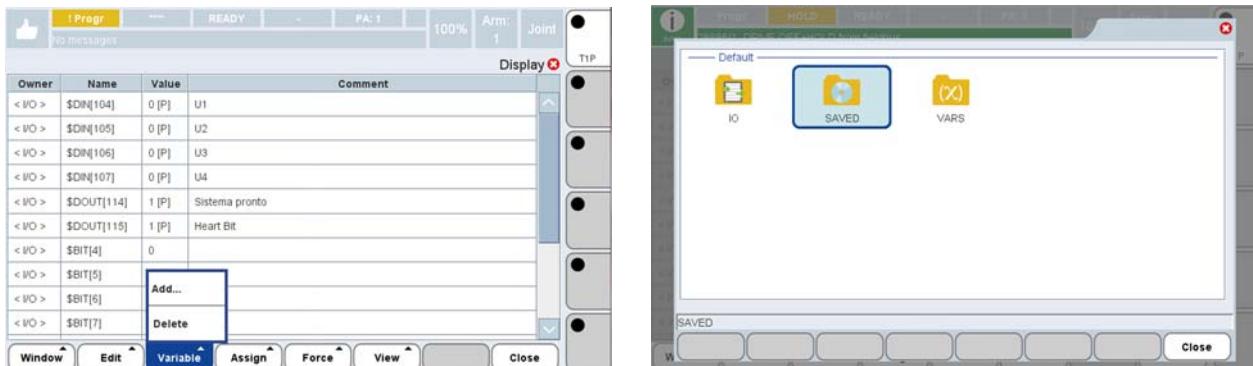


- c. Touch the icon of the **IO** folder.

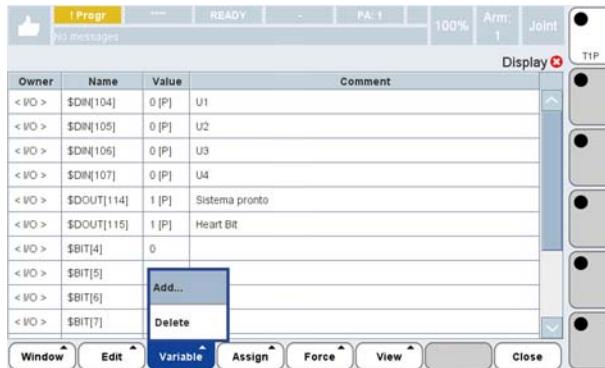
- d. Select **\$bit** ports group and specify a range of **4** to **8**. Touch **OK** key to confirm.



- e. Touch **Variable** key and choose **Add...** item again



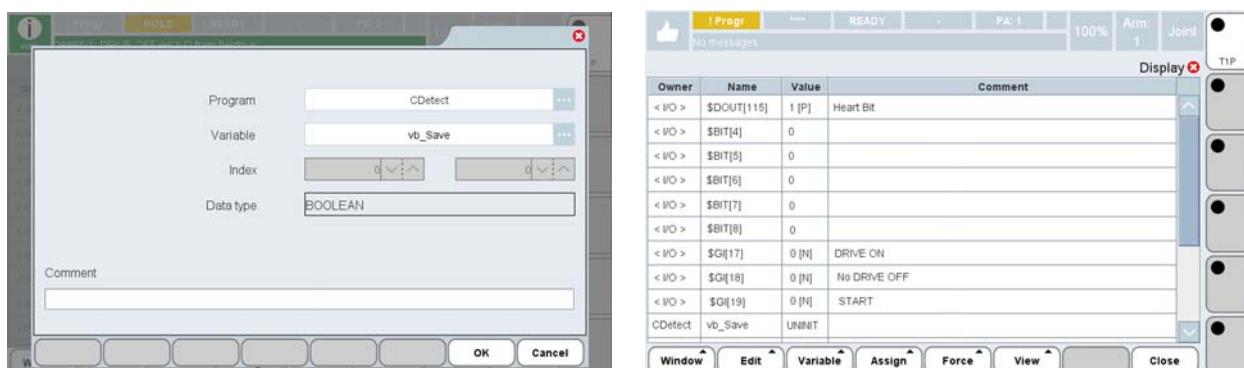
- f. After adding the elements included in **view_1** file, touch **Variable** key and choose **Add...** item again



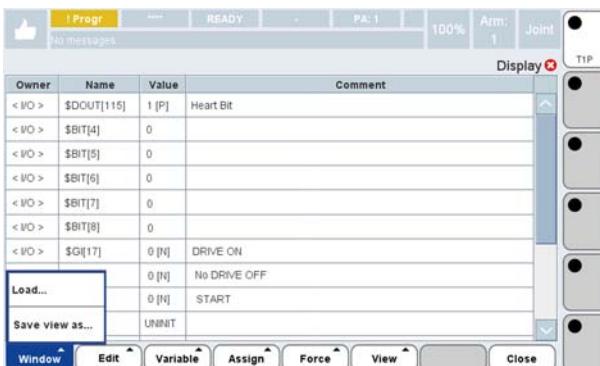
- g. Touch **Vars** icon and then touch **ProgramVar** icon



- h. Insert the required information into the displayed fields (**Program** and **Variable**), then touch **OK** to confirm. The required customized view is now completed.



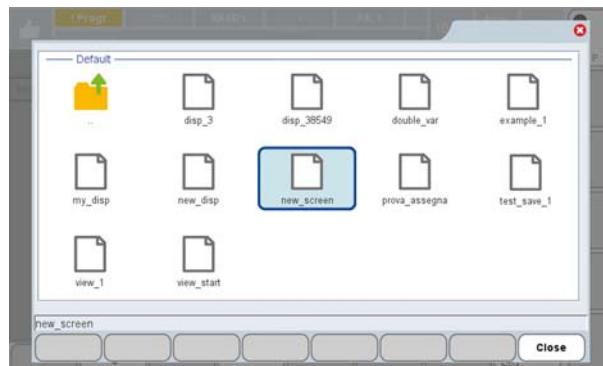
- i. Touch **Window** key and choose **Save view as...** item, to save the newly created customized view to a file (figure below).



- j. Use the automatically displayed keyboard to insert the wished name for the customized view file, then touch **OK** key to confirm.



- k. The file is then saved to **Saved** folder and it is available for future usage.



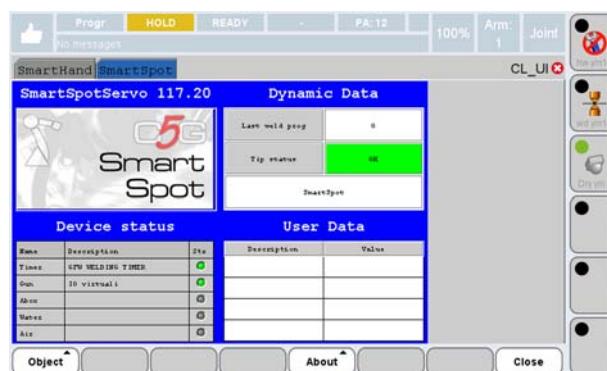
5.17 Appl Page

This page is the User Interface of the application package installed on the Controller Unit.

It is different depending on the application which it is referred to.

The below figure shows the **SmartSpot** application main page as an example of application page.

In such an example, there are two installed application packages: **SmartSpot** and **SmartHand**.

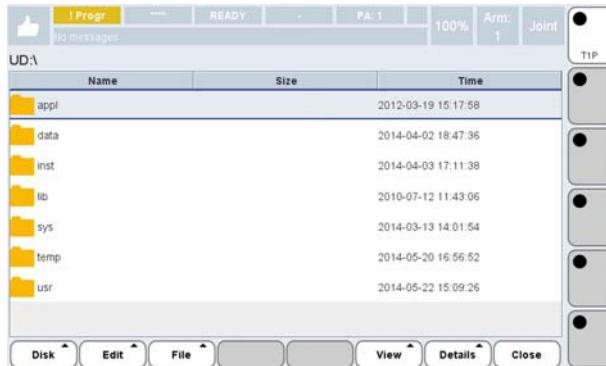


To properly handle the application User Interface, refer to the specific application manual.

5.18 Files Page

The User page of File Manager (Files) allows to carry out all the operations concerning the files: copy, cancel, create and remove files and folders, etc.

Some sort criteria and filters may be set by means appropriate keys.



The available functions are:

- [Disk](#)
- [Edit](#)
- [File](#)
- [View](#)
- [Details](#).

5.18.1 Disk



This menu includes the following utility commands, typical of such an environment:

- [New](#)
- [Backup](#)
- [Backup of Saveset](#)
- [Restore](#)
- [Restore of Saveset](#)
- [Device change](#)
- [Device Properties](#)

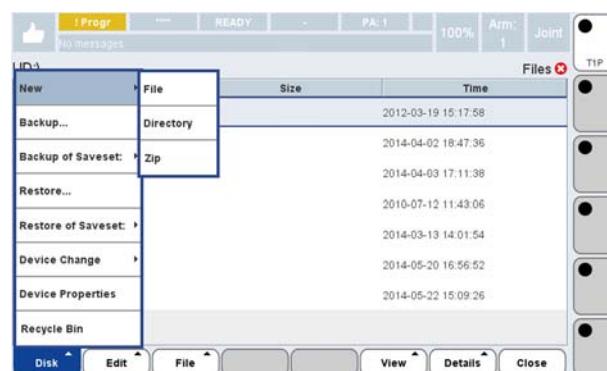
- Recycle bin.

5.18.1.1 New

Create a new **File**, **Directory** or compressed file (**.ZIP**).

Specify the new element name by means of the automatically activated keyboard. If it is a file, indicate the extension too.

Touch **OK** to confirm.



5.18.1.2 Backup

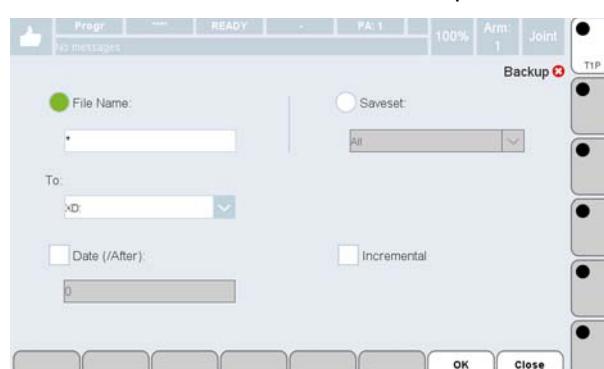
This command copies a range of files (specified by means of either a name or a Saveset) towards the default backup/restore device (\$DFT_DV[3]). The user is also allowed to select some options related to the command.



This is the most complete way to use the Backup command.

To activate this command, operate as follows:

- a. select **Backup** command;
- b. choose whether the **File Name** or the **Saveset** is to be used
- b.1 if **File Name** has been selected, proceed as follows



- b.1.1 Specify the wished **Destination**. If nothing is specified, the default backup/restore device is used (i.e. the current value of \$DFT_DV[3]);

- b.1.2 insert the wished command options, by selecting the corresponding checkbox:
- **Date(/After)**=copy all the files which have been created/modified either before or after a specified date or a certain amount of days,
 - **Incremental(/Incremental)**=copy all the files which have NEVER been saved in previous backup operations
- and any required additional information (e.g. for the **Date(/After)** option it is needed to insert the either date or the amount of day).

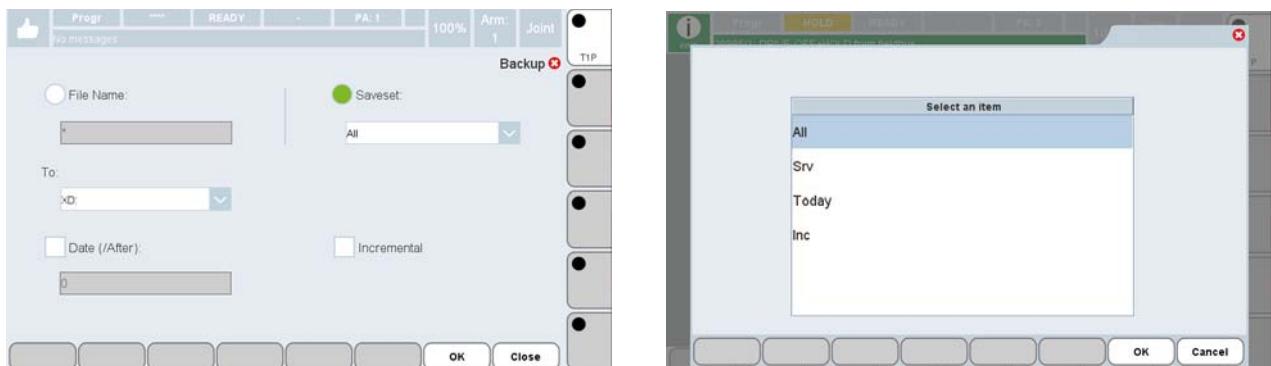
b.2 If, on the contrary, **Saveset** was specified, proceed as follows

b.2.1 touch the field containing the **Savesets** list;

b.2.2 the system opens a subenvironment in which the available **Savesets** are listed; touch the name of the wished **Saveset** and touch **OK** key to confirm;



To check whether the being selected Saveset name, the associated options and the corresponding files group match, it is strongly suggested to open the [Backup](#) sub-page in the [Setup page](#).

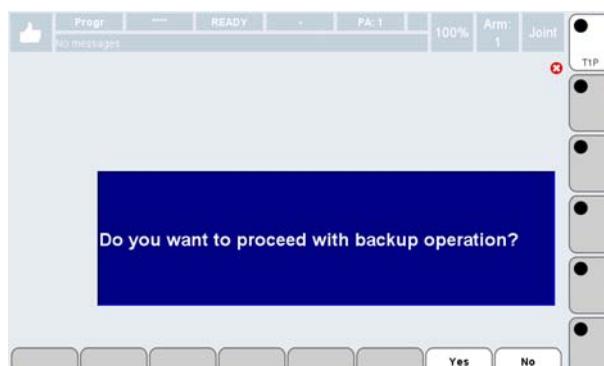


The corresponding options of the selected Saveset have already been inserted from the [Setup page](#), at the Saveset definition time.

It is anyway allowed to specify some more options, provided that they are compatible in respect with the already specified ones.

The allowed options for the Backup Savesets, at this level, are Date(/After) and Incremental (/Incremental).

c. touch **OK** key to confirm;



- d. as shown in the figure above, the system asks for a confirmation to go ahead with the Backup operation. Confirm by touching **Yes** key to start its execution.



HINTS FOR BACKUP AND RESTORE USE

It is recommended to follow the listed below hints, in order to better understand the difference between executing a simple file copy ([FILER COPY \(FC\) command](#)) and performing well organized Backup/Restore operations:

- **always verify the execution memory and the UD: device to be consistent (no unsaved programs and/or data files) before going on with any Backup or Restore operation.**
- **Always save the configuration file ([Save](#) in the [Setup page](#)) before issuing a Backup command; always load the configuration file ([Load](#) in the [Setup page](#)) after a Restore operation, in order to restore the same working conditions, based on .C5G file, after Restart.**
- **Schedule periodic Backup operations, preferably when NO technological operations are running, when motor drives are off (DRIVE-OFF) and when the system is in programming status.**

5.18.1.3 Backup of Saveset

This command copies all files belonging to the specified **Saveset**, towards the default backup/restore device (\$DFT_DV[3]).

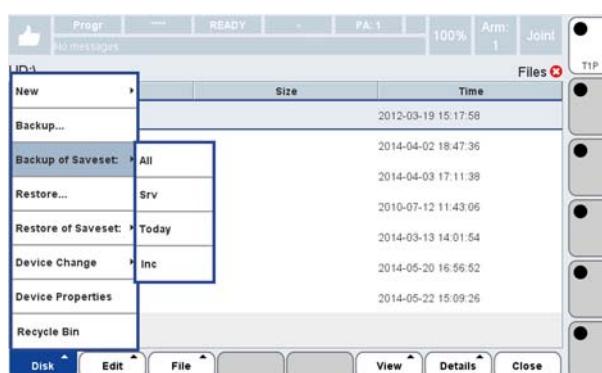
The specified **Saveset** must have already been defined in the [Setup page](#); the user is not allowed to modify in any way the **Saveset**, at this level.



This is the simplest way of using the Backup command.

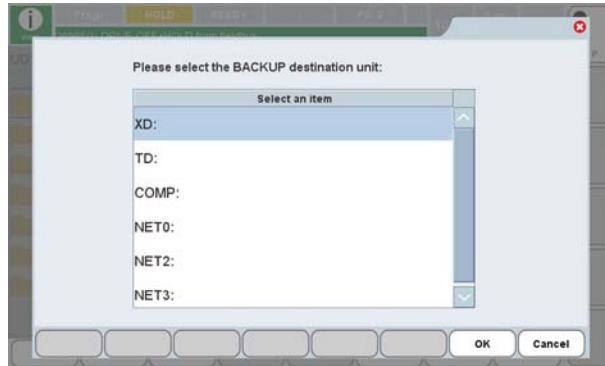
Operate as follows, to activate such a Backup command:

- a. choose the **Backup of Saveset** item selecting the wished **Saveset** ;

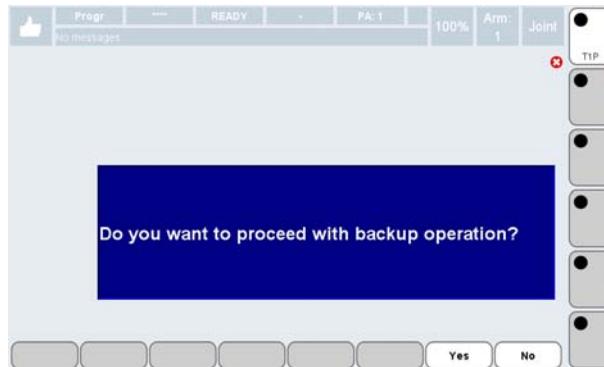


Among the available Savesets, there is a predefined one, called Srv (see above figure), which is to be used to save a specific situation before calling COMAU Assistance. It is **suggested not to** activate such a command during the process: it could slow it down.

- b. the system displays the current Backup destination device; if wished, it is possible to modify it;



- c. touch **OK** key to confirm;
d. the System asks the user again for confirmation, before starting to execute the Backup operation. Touch **Yes** key to confirm.



5.18.1.4 Restore

This command restores all files (specified by means of either a name or a Saveset) from the default backup/restore device (\$DFT_DV[3]), to the device from which they had been previously saved by means of a **Backup** command. The user is also allowed to insert some options associated to the command.



This is the most complete way to use the Restore command.



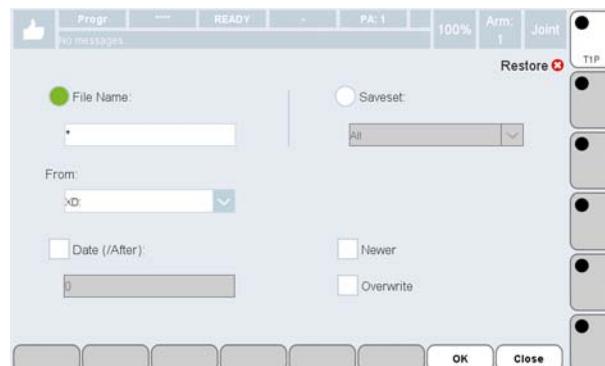
Read carefully the **NOTE** about the total amount of entries in the UD: root directory.



See also [HINTS FOR BACKUP AND RESTORE USE](#).

To activate this command, operate as follows:

- a. select **Restore** command;
- b. chose whether a **File Name** or a **Saveset** is to be used
- b.1 if **File Name** has been specified, proceed as follows

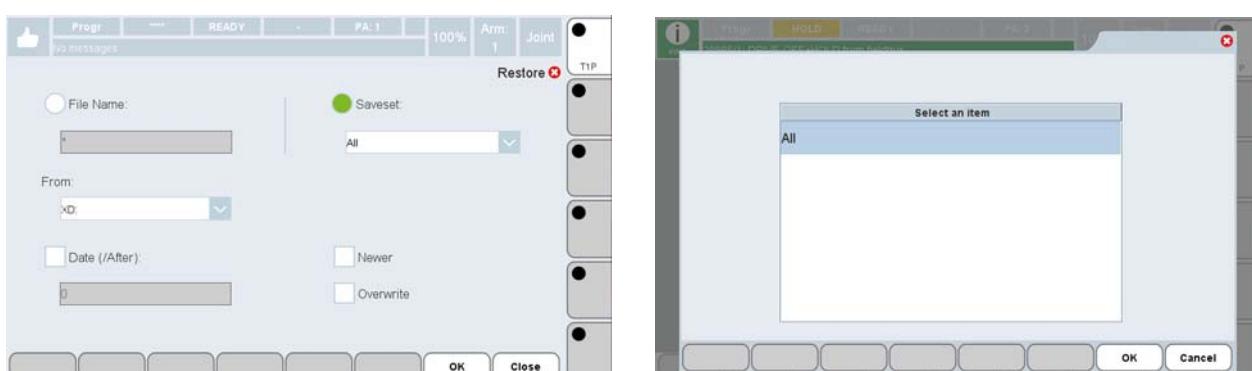


b.1.1 specify the desired **Source**. If nothing is specified, the default backup/restore device (i.e. the current value of \$DFT_DV[3]) is used;

- b.1.2 insert the wished command options, by selecting the corresponding **Checkbox**:
- **Date(/After)**=restores all the files which have been created/modified before or after a specified date or amount of days,
 - **Newer**=restores if more recent only,
 - **Overwrite**=overwrite also read-only files,
- and any required additional information (e.g. for the **Date(/After)** option it is needed to insert either the date or the amount of days).

b.2 If **Saveset** has been specified, proceed as follows

b.2.1 touch the **Saveset** campo, to open the list;



b.2.2 the system opens a subenvironment in which the available **Savesets** are listed; touch the wished **Saveset** name and confirm touching **OK** key;



To check whether the name of the being selected Saveset, the associated options and the corresponding files group match, it is strongly suggested to open the **Restore** sub-page in the [Setup page](#).



Any allowed corresponding options for the selected Saveset have already been inserted from the [Setup page](#), at the Saveset definition time.
It is anyway allowed to specify some more options, provided that they are compatible in respect with the already specified ones.
The allowed options for the Backup Savesets, at this level, are Date(/After), Newer and Overwrite.

- c. press **OK (F5)** to confirm;



- d. as shown in the figure above, the System asks the user again for confirmation, before starting to execute the Restore operation. Touch **Yes** to start the Restore execution.

5.18.1.5 Restore of Saveset

This command copies all files belonging to the specified **Saveset**, to the device from which they had been previously saved by means of a **Backup** command.

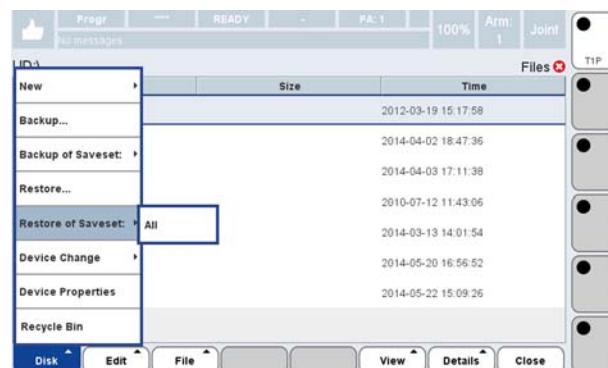
The specified Saveset must have already been defined in the [Setup page](#); the user is not allowed to modify the Saveset setup, at this level.



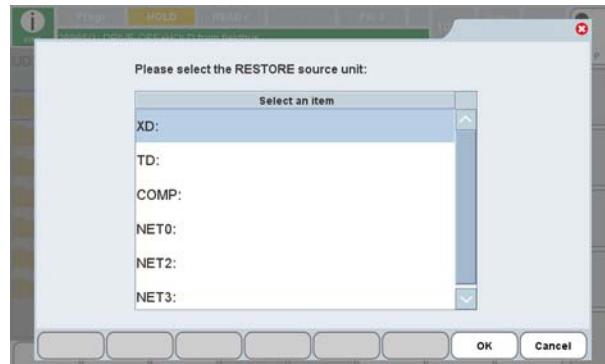
This is the simplest way of using the Restore command.

Operate as follows, to activate such a Restore type command:

- a. activate the **Restore of Saveset** command by selecting the wished **Saveset**;



- b. the system displays the currently active Restore source device; it is editable, if wished;

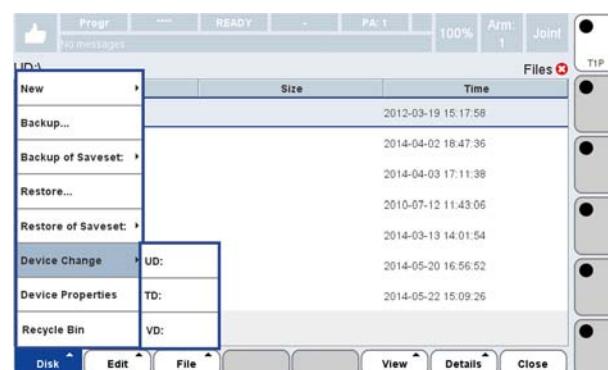


- c. touch **OK** to confirm the choice;
- d. prima di provvedere ad eseguire l'operazione di Restore, il sistema presenta un'ulteriore richiesta di conferma da parte dell'utente. Rispondere affermativamente.the System asks the user again for confirmation, before starting to execute the restore operation. Answer by touching **Yes** to confirm.



5.18.1.6 Device change

This command allows changing the device for displaying the files list. The allowed devices depend on the hardware detected by the system.



Access to a different device may depend on whether or not the Login has been executed.

If a removable device is to be used, insert it in the suitable housing (e.g. a USB flash

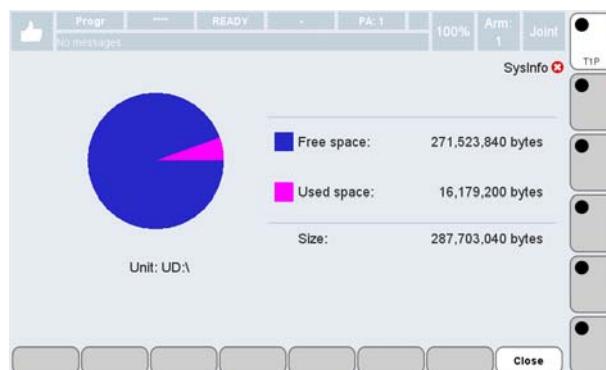
disk in a USB port) before using the required command.

In such a way the wished device will be listed among the selectable devices and the user will be able to choose it.

5.18.1.7 Device Properties

This command displays information about the current storage unit (UD):

- graphic representation of free space and occupied space
- unit name
- available space
- used space
- unit size.



5.18.1.8 Recycle bin

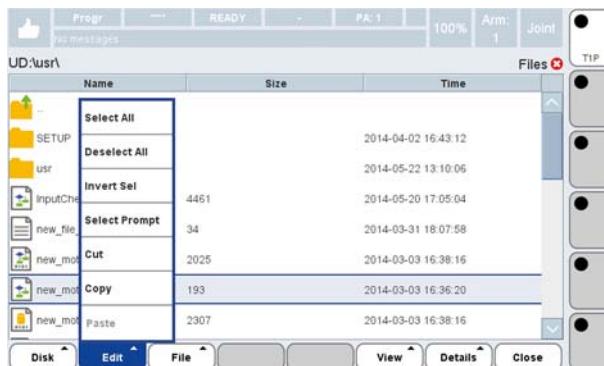


Displays the recycle bin content related to the selected directory. The following functions are available in the [Functional keys menu](#):

- **Empty**
Empties the bin.
- **Erase**
Permanently removes the selected file from the bin.
- **Restore**
Restores the selected file and put it again in the current directory.

5.18.2 Edit

This menu allows executing some operations to handle the currently displayed files.



- Select all
- Deselect all
- Invert sel
- Sel prompt
- Cut
- Copy
- Paste.

5.18.2.1 Select all

Selects all the files included in the current folder.

5.18.2.2 Deselect all

Deselects any already selected files.

5.18.2.3 Invert sel

Inverts the current selection (the selected files are unselected and the non-selected ones are selected).



5.18.2.4 Sel prompt

Sets the criterion to be used to select the files. The user is asked to insert the wished selection criterion; the use of the wildcard is allowed.



In the shown above example, the selection criterion is "select all .cod files."

5.18.2.5 Cut

Cuts the selected files.

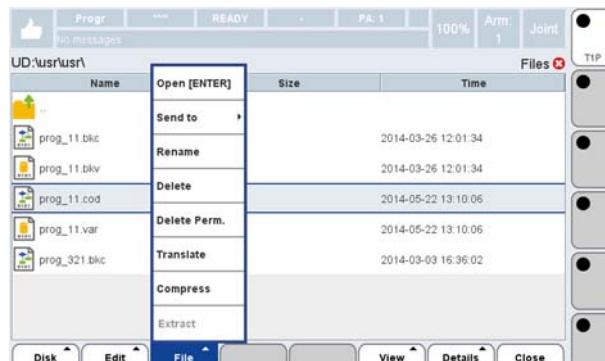
5.18.2.6 Copy

Copies the selected files.

5.18.2.7 Paste

Copies the previously cut or copied files to the current folder.

5.18.3 File



The available choices are:

- Open [ENTER]
- Send to
- Rename
- Delete
- Delete Perm.

- Translate
- Compress
- Extract.

A detailed description is provided for all the listed above functionalities.

5.18.3.1 Open [ENTER]

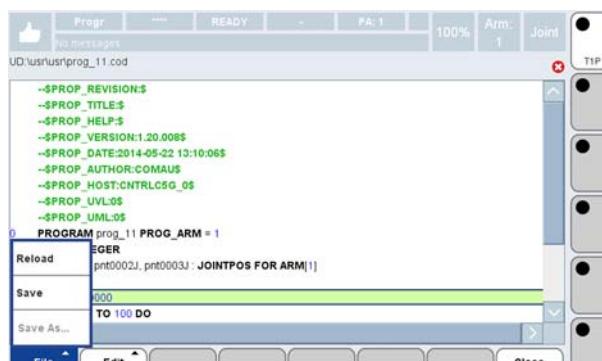
This command opens the file in ASCII editing mode.



In this environment, the available keys in the [Functional keys menu](#) are as follows:

- [File](#)
- [Edit](#).

5.18.3.1.1 File



This menu allows activating the following commands:

- [Reload](#)
- [Save](#)
- [Save as.](#)

Reload

Reloads the last saved version of the currently opened file in edit environment.

Save

Saves the current file with the same name, keeping it open for editing.

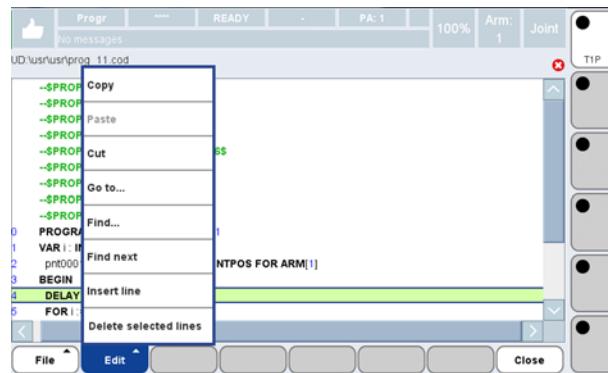
Save as

Saves the current file content with a different name and/or destination, keeping it open for editing.



To save the file, see [NOTE](#) regarding the total amount of entries in the UD: root directory.

5.18.3.1.2 Edit



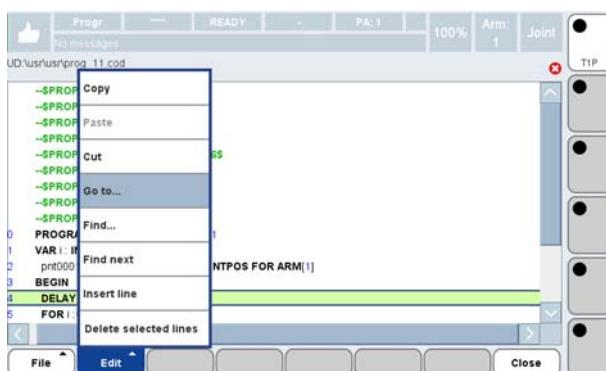
The classic edit commands for a text file are available in this menu:

- Copy
- Paste
- Cut
- Go to..
- Find...
- Find next
- Insert line
- Delete selected lines.

Go to..

This command allows positioning the cursor to the wished line.

Touch to select it. The system automatically opens the [Numeric keyboard](#) to allow the user inserting the wished program line number.



Find...

This command allows finding a text string within an open for editing file.

Touching this command, the system automatically opens the [Alphanumeric keyboard](#) to allow the user to insert the being searched text string.

Insert the wished string and touch **OK** to confirm.

The cursor is then positioned to the line where the requested string is found.

If not found, a suitable informational message is displayed.

Find next

This command allows finding the next occurrence of an already found text string within an open for editing file.

If not found, a suitable informational message is displayed.

Insert line

This command opens the [PDL2 keyboard](#) to allow the user inserting a new program line, with the wished content.

The PDL2 keyboard allows inserting PDL2 statements as well as simply typing text strings.



Delete selected lines

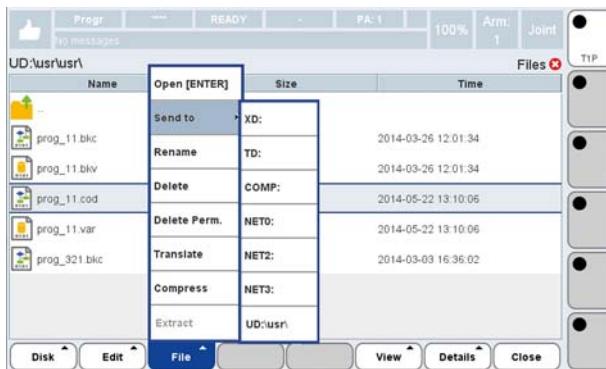
This command acts on the currently open file content.

Select the being deleted line or lines land then activate the command.



5.18.3.2 Send to

Copies the selected file to the specified device.



5.18.3.3 Rename

Allows specifying a different name for the currently selected file.



When the **Rename** command is touched, the system automatically opens the **Alphanumeric keyboard** and displays the current filename, to allows the user to insert a new filename. Insert it and touch **OK** key to confirm.

5.18.3.4 Delete

Deletes the selected file. It can be restored by means of the **Recycle bin** command: select the being restored file and touch the **Restore** key.



The selected file is taken away from the **Recycle bin** and put into the directory from which the **Recycle bin** command was issued.

Note that one **Recycle bin** for each folder is available.

5.18.3.5 Delete Perm.

Permanently deletes the selected file. The user is no longer allowed to restore it.

5.18.3.6 Translate

Converts:

- a PDL2 program in ASCII source code (**.PDL**), to an executable code (**.COD**) file and vice versa;
- a file of variables in ASCII source format (**.LSV**), to a file of variables with a loadable format for the Controller (**.VAR**) and vice versa.

If the destination file (e.g. **.COD**) is already existing, the copy is saved with a different extension (e.g. **.BKC**) before the new file is created.

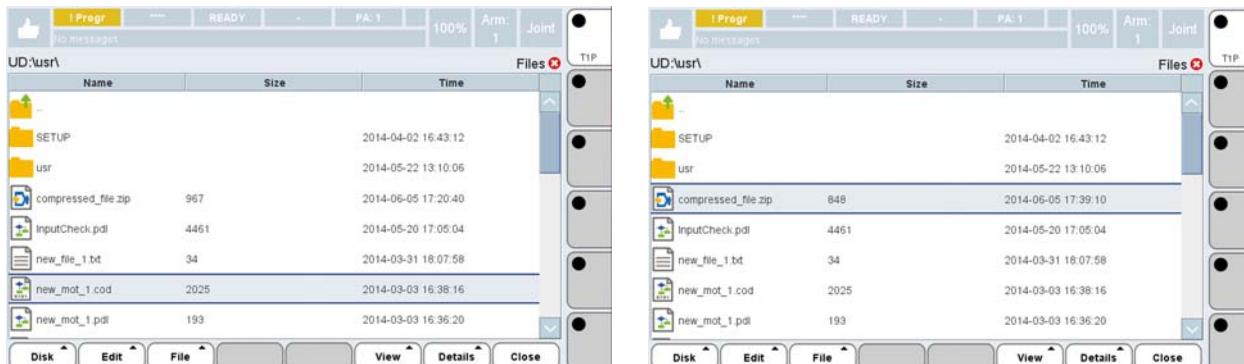
If reports or errors are detected during the conversion, they are displayed in the scrolling window and recorded in **<file_name>.ERR** file.

This command can also be used to convert system binary files (**.C5G**), to the corresponding ASCII format (**.PDL**).

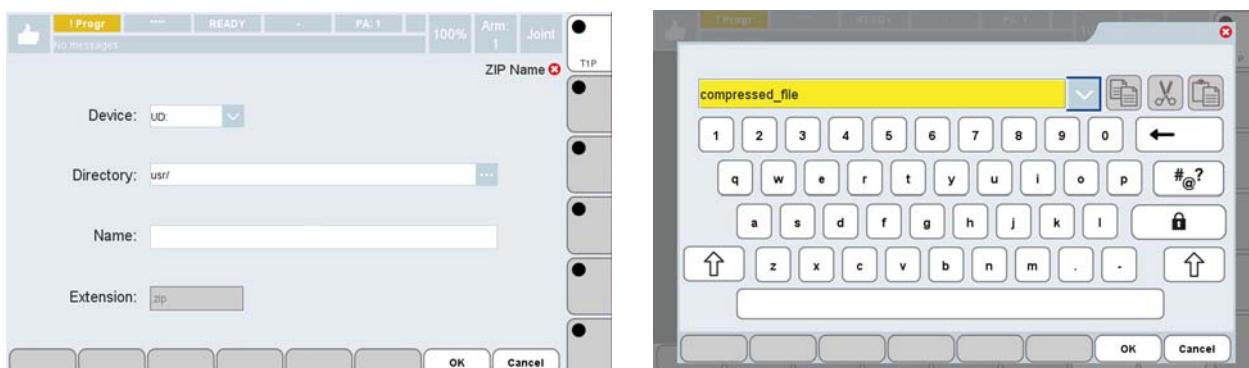
5.18.3.7 Compress

This command allows compressing a file, thus reducing its size. The compressed file has **.ZIP** extension.

In the below example, *new_mot_1.cod* file has been compressed.



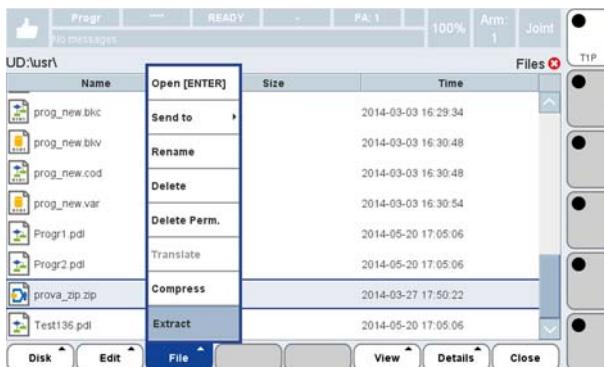
Touching Compress command, the system opens a subpage in which the user should specify the Device, the Directory and the compressed file name. The [Alphanumeric keyboard](#) is automatically opened to type in the file name.



Touch **OK** to start the operation.

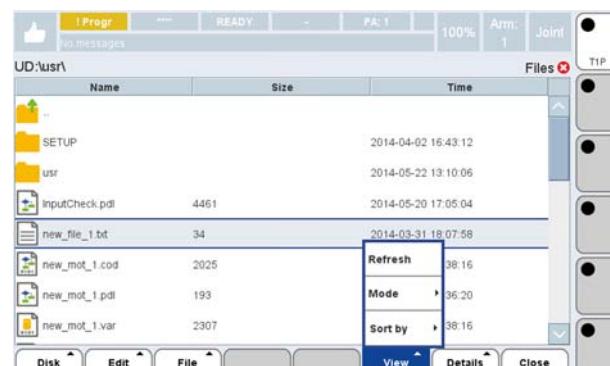
5.18.3.8 Extract

Extracts files from a compressed file (**.ZIP**) and put them to the same folder. Note that if the currently selected file is NOT a compressed one, this command is NOT available.



5.18.4 View

Handles the files information viewing and how the related icons are displayed.



- [Refresh](#)
- [Mode](#)
- [Sort by](#).

5.18.4.1 Refresh

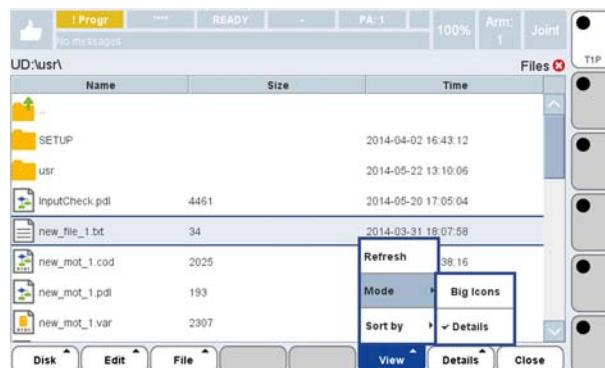
Updates the information included in the files list.



NOTE: to be sure that the displayed information is always consistent, refreshing is to be called; in fact, refreshing is NOT automatically carried out since the displayed information is static.

5.18.4.2 Mode

Specifies how the files list icons are displayed.



- Big icons
 - Details.
-

5.18.4.2.1 Big icons

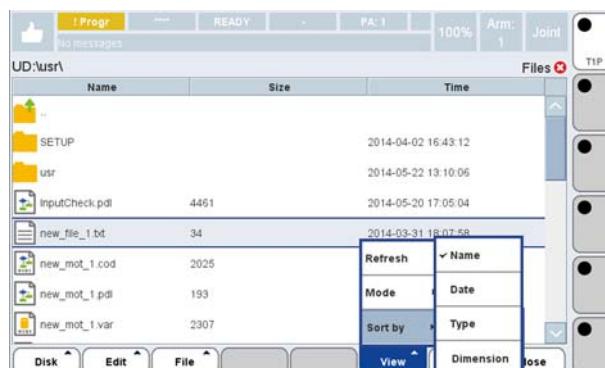
Allows displaying the information with large-sized icon.

5.18.4.2.2 Details

This command allows viewing the files list with the detailed information:

- Name - it is the filename
 - Size - it is the file size in bytes
 - Time - it is the last modification date.
-

5.18.4.3 Sort by



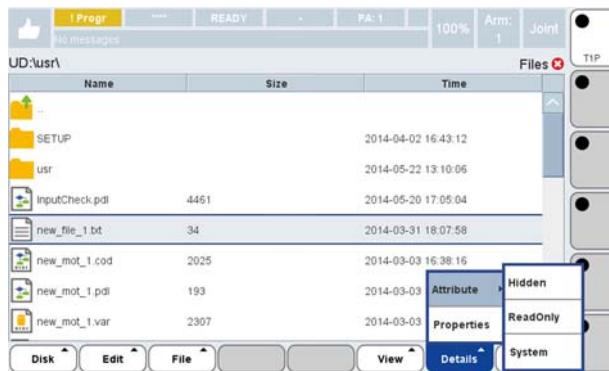
This command allows viewing the list of files according to different criteria:

- Name - alphabetical order
- Date - date of modification (from top downwards, first the oldest files are displayed, followed by the most recent ones)
- Type - type of extension.
- Dimension - file dimension.



Note that the first time a criterium is selected (e.g. "Sort by Name"), the increasing sort is used (e.g. from A to Z); selecting the same criterium again, the decreasing sort is used (e.g. from Z to A).

5.18.5 Details



Allows viewing some detailed information about the currently selected file(s):

- [Attribute](#)
- [Properties](#).

5.18.5.1 Attribute

Allows to assign the selected file or to remove from it one of the following attributes:

- **Hidden** - not viewable files
- **Read-only** - read only files
- **System** - system files.

To remove an attribute, just select it again (toggle type function).

5.18.5.2 Properties

Allows viewing the selected file **Property Notes**.

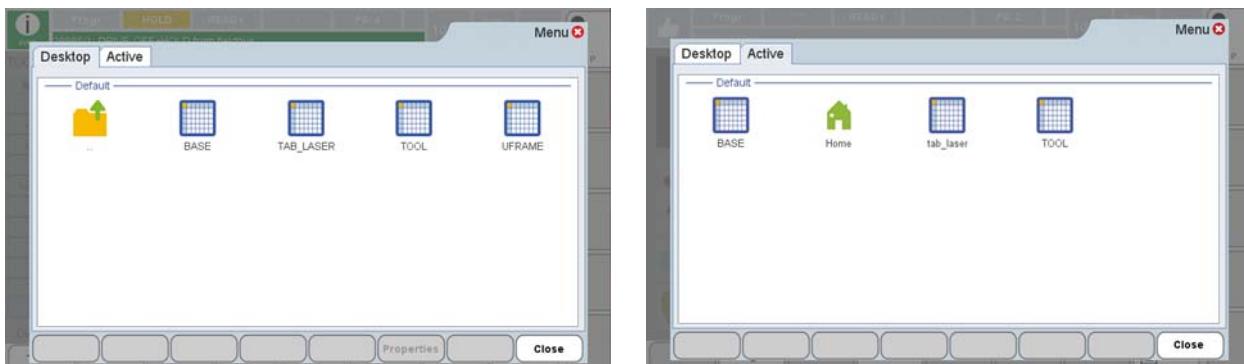


5.19 Data Page

The User DATA page of the Teach Pendant allows to read and modify all the data tables stored in C5G Control Unit .

It is composed by two subpages (panels):

- **Desktop** - includes the icons of all the available tables (left figure below)
- **Active** - includes the icons of all the currently active tables (right figure below).



The way of handling the tables is the same in both the panels, so a common description is provided.

The following tables can be managed in the Data Page:

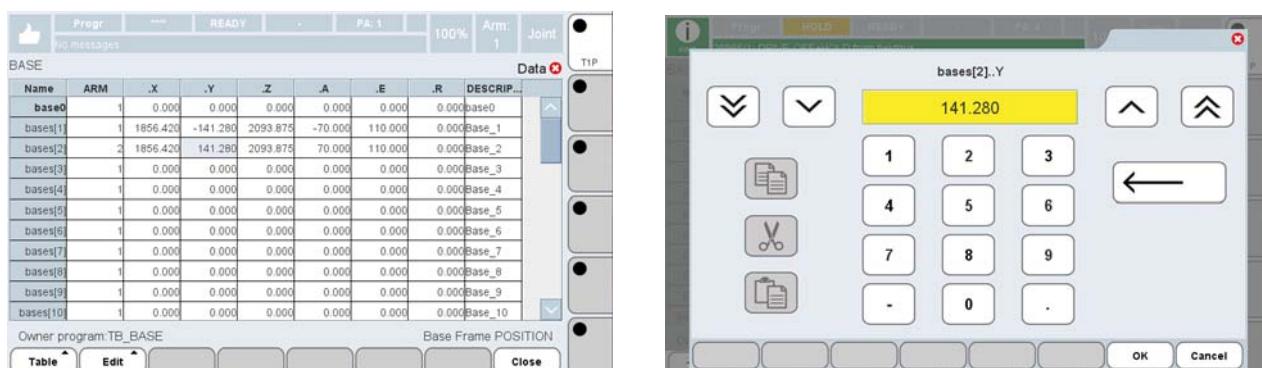
- [System tables](#)
- [Application tables](#)
- [User tables.](#)

5.19.1 General information

When the Data Page is opened, the list is shown of all the existing tables on the Teach Pendant display.

Touch the corresponding icon to open a table.

Touch twice the open table cells to edit them. In such a way, the corresponding keyboard is automatically opened which allows inserting the wished value (see example in the below figures).



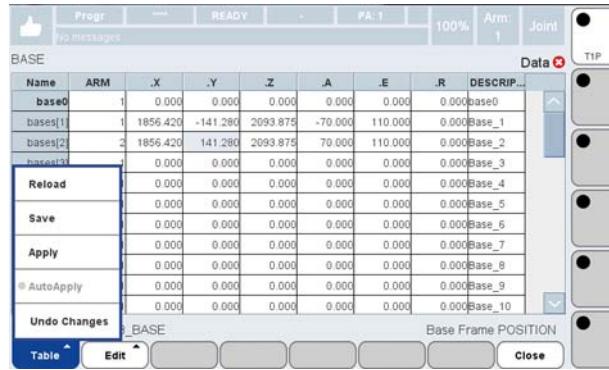
When a table is open, the available commands in the [Functional keys menu](#), sare as follows:

- [Table](#)

- Edit.

5.19.1.1 Table

This menu provides commands to operate on the table and on its corresponding file.



- Reload
- Save
- Apply
- AutoApply
- Undo Changes.

5.19.1.1.1 Reload

Reloads the value of the variables from the file.

5.19.1.1.2 Save

Saves the changes both in the execution memory and in the file. The Table remains open.



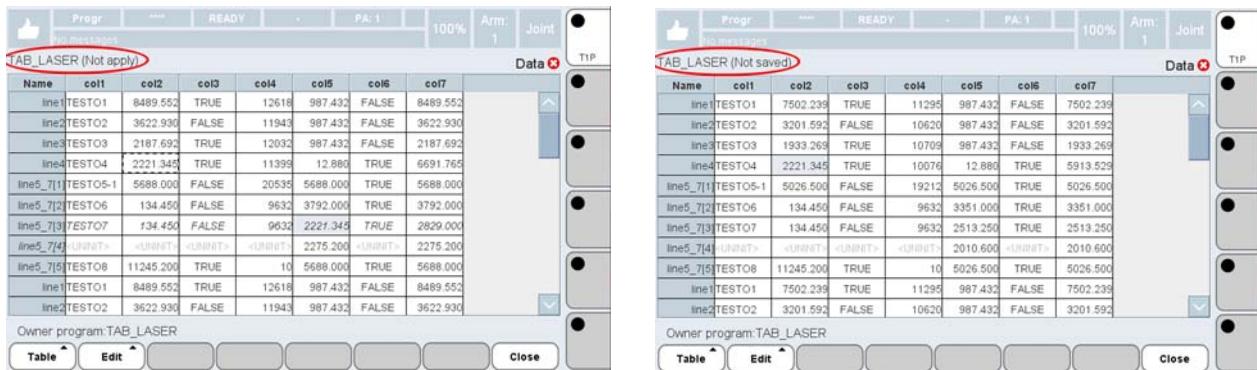
See [NOTE](#) about the total amount of entries in the UD: root directory.

5.19.1.1.3 Apply

Saves the applied changes in the execution memory. The table keeps open, but the changes are NOT stored in the file.

Note that the modifications are written in *italic* until they are not applied to the table (see example in the following figure).

Furthermore, in the topmost field, besides the table name, a “Not applied” text is displayed (see the following figure on the left, highlighted in red).



As soon as the **Apply** command is activated, such a text becomes "Not saved" (see previous figure on the right, highlighted in red).

5.19.1.1.4 AutoApply

Editing in **AutoApply** mode means to make the changes applied to the table cells (i.e. copied in execution memory) get **automatically** operating, with no needs of using **Apply** command.

If a table may be edited in **AutoApply** mode, such a functionality is enabled when the table is opened (a lighted green "led" is displayed on the left of the command).

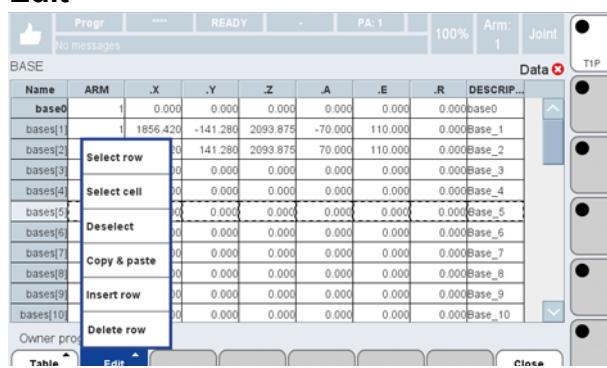
This mode can be disabled/enabled by touching the key again, toggle type functioning.



5.19.1.1.5 Undo Changes

Cancels all modifications in the cells where the **Apply** command has not been used yet.

5.19.1.2 Edit



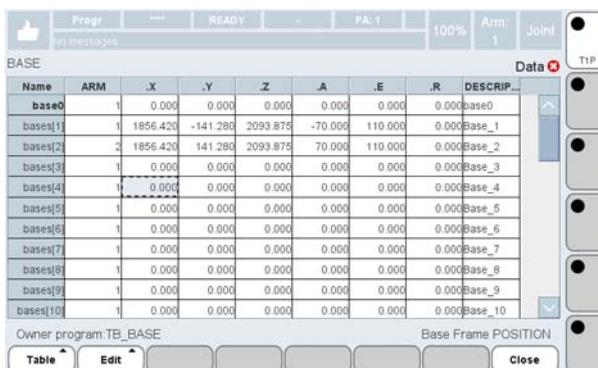
Allows editing the table content.

The following commands are available:

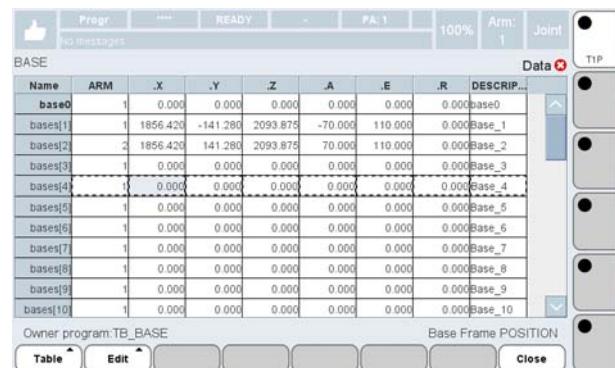
- [Select row](#)
- [Select cell](#)
- [Deselect](#)
- [Copy & paste](#)
- [Insert row](#)
- [Delete row.](#)

5.19.1.2.1 Select row

Touch this command to select (see the figure below on the right) the row where focus is currently placed (see the figure below on the left).



Name	ARM	X	Y	Z	A	E	R	Description
base0	1	0.000	0.000	0.000	0.000	0.000	0.000	base0
bases[1]	1	1856.420	-141.280	2093.875	-70.000	110.000	0.000	base_1
bases[2]	2	1856.420	141.280	2093.875	70.000	110.000	0.000	base_2
bases[3]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_3
bases[4]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_4
bases[5]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_5
bases[6]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_6
bases[7]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_7
bases[8]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_8
bases[9]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_9
bases[10]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_10

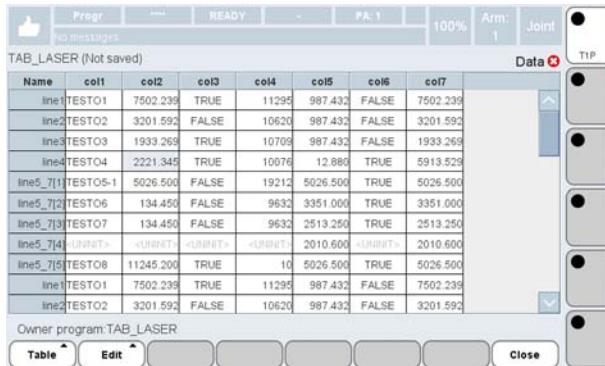


Name	ARM	X	Y	Z	A	E	R	Description
base0	1	0.000	0.000	0.000	0.000	0.000	0.000	base0
bases[1]	1	1856.420	-141.280	2093.875	-70.000	110.000	0.000	base_1
bases[2]	2	1856.420	141.280	2093.875	70.000	110.000	0.000	base_2
bases[3]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_3
bases[4]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_4
bases[5]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_5
bases[6]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_6
bases[7]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_7
bases[8]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_8
bases[9]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_9
bases[10]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_10

If focus is not placed on any cell, this command does not take any effect.

5.19.1.2.2 Select cell

If focus is onto a cell (see left figure below), touching this command causes the cell to be selected (see right figure below).



Name	col1	col2	col3	col4	col5	col6	col7
line1	TEST01	7502.239	TRUE	11295	987.432	FALSE	7502.239
line2	TEST02	3201.592	FALSE	10620	987.432	FALSE	3201.592
line3	TEST03	1933.269	TRUE	10709	987.432	FALSE	1933.269
line4	TEST04	2221.345	TRUE	10076	12.880	TRUE	5913.529
line5_7[1]	TEST05-1	5026.500	FALSE	19212	5026.500	TRUE	5026.500
line5_7[2]	TEST06	134.450	FALSE	9632	3351.000	TRUE	3351.000
line5_7[3]	TEST07	134.450	FALSE	9632	2513.250	TRUE	2513.250
line5_7[4]	TEST08	<UNRT>	<UNRT>	2010.600	<UNRT>	<UNRT>	2010.600
line5_7[5]	TEST09	11245.200	TRUE	10	5026.500	TRUE	5026.500
line1	TEST01	7502.239	TRUE	11295	987.432	FALSE	7502.239
line2	TEST02	3201.592	FALSE	10620	987.432	FALSE	3201.592



Name	col1	col2	col3	col4	col5	col6	col7
line1	TEST01	7955.224	TRUE	11902	987.432	FALSE	7955.224
line2	TEST02	3394.904	FALSE	11227	987.432	FALSE	3394.904
line3	TEST03	2050.000	TRUE	11316	987.432	FALSE	2050.000
line4	TEST04	2221.345	TRUE	10683	12.880	TRUE	6270.586
line5_7[1]	TEST05-1	5330.000	FALSE	19819	5330.000	TRUE	5330.000
line5_7[2]	TEST06	134.450	FALSE	9632	3553.333	TRUE	3553.333
line5_7[3]	TEST07	134.450	FALSE	9632	2665.000	TRUE	2665.000
line5_7[4]	TEST08	<UNRT>	<UNRT>	<UNRT>	<UNRT>	2132.000	2132.000
line5_7[5]	TEST09	11245.200	TRUE	10	5330.000	TRUE	5330.000
line1	TEST01	7955.224	TRUE	11902	987.432	FALSE	7955.224
line2	TEST02	3394.904	FALSE	11227	987.432	FALSE	3394.904

5.19.1.2.3 Deselect

Deselect all currently selected cell/row.

5.19.1.2.4 Copy & paste

This command allows pasting the contents of the currently selected cell or row by copying exactly when the paste operation is performed.

This is very useful when handling tables whose content is changing very fast in real time thus, copying/pasting in two different moments, would mean to paste an already meaningless value, due to the elapsed time.

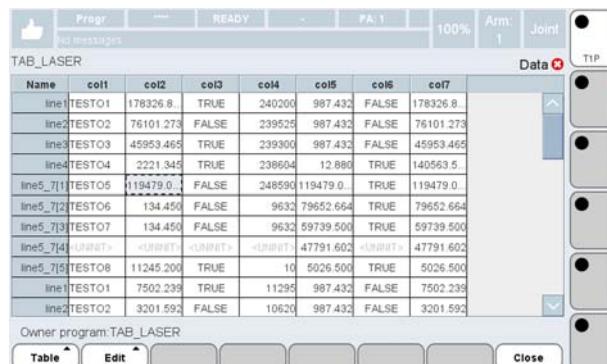
To properly use this command, it is needed to

- touch the being copied cell or one cell of the being copied row (in the shown below example a cell is **copied&pasted**)



Name	col1	col2	col3	col4	col5	col6	col7
line1TESTO1	7502.239	TRUE	11295	987.432	FALSE	7502.239	
line2TESTO2	3201.592	FALSE	10620	987.432	FALSE	3201.592	
line3TESTO3	1933.269	TRUE	10709	987.432	FALSE	1933.269	
line4TESTO4	2221.345	TRUE	10076	12.880	TRUE	5913.529	
line5_7 1TESTO5-1	119133	FALSE	19212	5026.500	TRUE	5026.500	
line5_7 2TESTO6	134.450	FALSE	9632	3351.000	TRUE	3351.000	
line5_7 3TESTO7	134.450	FALSE	9632	2519.250	TRUE	2519.250	
line5_7 4TESTO8	<UNINIT>	<UNINIT>	<UNINIT>	2010.600	<UNINIT>	2010.600	
line5_7 5TESTO9	11245.200	TRUE	10	5026.500	TRUE	5026.500	
line1TESTO1	7502.239	TRUE	11295	987.432	FALSE	7502.239	
line2TESTO2	3201.592	FALSE	10620	987.432	FALSE	3201.592	

- select such a cell or row ([Select cell](#) or [Select row](#) command)



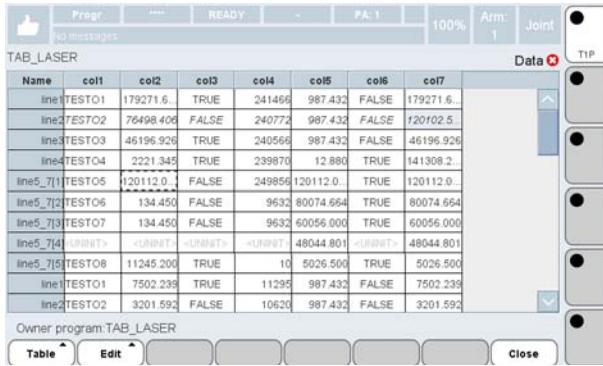
Name	col1	col2	col3	col4	col5	col6	col7
line1TESTO1	178326.8	TRUE	240206	987.432	FALSE	178326.8	
line2TESTO2	76101.273	FALSE	239525	987.432	FALSE	76101.273	
line3TESTO3	45953.465	TRUE	239306	987.432	FALSE	45953.465	
line4TESTO4	2221.345	TRUE	238604	12.880	TRUE	140563.5	
line5_7 1TESTO5	119479.0	FALSE	248591	119479.0	TRUE	119479.0	
line5_7 2TESTO6	134.450	FALSE	9632	79652.664	TRUE	79652.664	
line5_7 3TESTO7	134.450	FALSE	9632	59739.500	TRUE	59739.500	
line5_7 4TESTO8	<UNINIT>	<UNINIT>	<UNINIT>	47791.602	<UNINIT>	47791.602	
line5_7 5TESTO9	11245.200	TRUE	10	5026.500	TRUE	5026.500	
line1TESTO1	7502.239	TRUE	11295	987.432	FALSE	7502.239	
line2TESTO2	3201.592	FALSE	10620	987.432	FALSE	3201.592	

- touch the destination cell or one cell belonging to the destination row



Name	col1	col2	col3	col4	col5	col6	col7
line1TESTO1	178745.5	TRUE	240761	987.432	FALSE	178745.5	
line2TESTO2	76279.938	FALSE	240086	987.432	FALSE	76279.938	
line3TESTO3	46061.348	TRUE	239861	987.432	FALSE	46061.348	
line4TESTO4	2221.345	TRUE	239165	12.880	TRUE	140893.5	
line5_7 1TESTO5	119759.5	FALSE	249151	119759.5	TRUE	119759.5	
line5_7 2TESTO6	134.450	FALSE	9632	79839.664	TRUE	79839.664	
line5_7 3TESTO7	134.450	FALSE	9632	59879.750	TRUE	59879.750	
line5_7 4TESTO8	<UNINIT>	<UNINIT>	<UNINIT>	47903.801	<UNINIT>	47903.801	
line5_7 5TESTO9	11245.200	TRUE	10	5026.500	TRUE	5026.500	
line1TESTO1	7502.239	TRUE	11295	987.432	FALSE	7502.239	
line2TESTO2	3201.592	FALSE	10620	987.432	FALSE	3201.592	

- touch **Copy & paste** command.

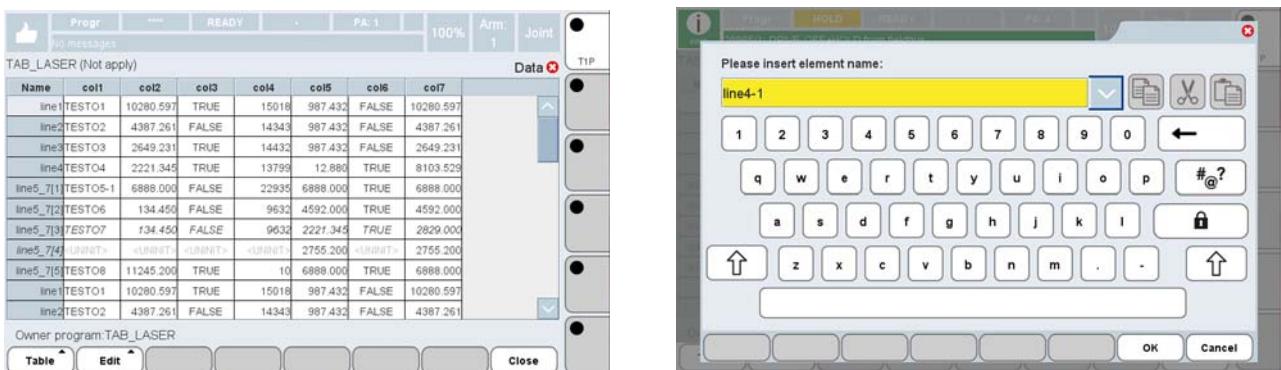


Name	col1	col2	col3	col4	col5	col6	col7
line1 TESTO1	179271.6...	TRUE	241466	987.432	FALSE	179271.6...	
line2 TESTO2	76498.406	FALSE	240772	987.432	FALSE	120102.5...	
line3 TESTO3	46196.926	TRUE	240566	987.432	FALSE	46196.926	
line4 TESTO4	2221.345	TRUE	239870	12.880	TRUE	141308.2	
line5_7 1 TESTO5	120112.0	FALSE	249856	120112.0	TRUE	120112.0	
line5_7 2 TESTO6	134.450	FALSE	9632	80074.664	TRUE	80074.664	
line5_7 3 TESTO7	134.450	FALSE	9632	60056.000	TRUE	60056.000	
line5_7 4 <UNIT>	<UNIT>	<UNIT>	<UNIT>	48044.801	<UNIT>	48044.801	
line5_7 5 TESTO8	11245.200	TRUE	10	5026.500	TRUE	5026.500	
line1 TESTO1	7502.239	TRUE	11295	987.432	FALSE	7502.239	
line2 TESTO2	3201.592	FALSE	10620	987.432	FALSE	3201.592	

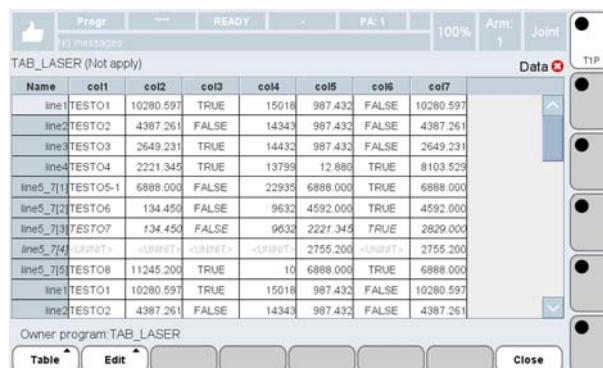
5.19.1.2.5 Insert row

Inserts a new row in the table, in alphabetical order, by **Name**.

Touching this command, the system automatically opens the alphanumeric keyboard to allow the user inserting the new row name (see example in the figures below).



The new row is inserted in the proper position, according to the alphabetical order (see the figure below).



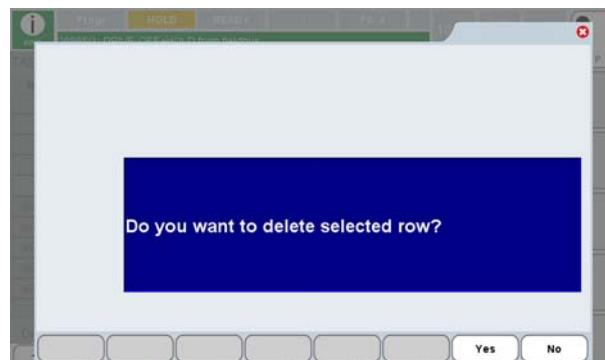
Name	col1	col2	col3	col4	col5	col6	col7
line1 TESTO1	10280.597	TRUE	15018	987.432	FALSE	10280.597	
line2 TESTO2	4387.261	FALSE	14343	987.432	FALSE	4387.261	
line3 TESTO3	2649.231	TRUE	14432	987.432	FALSE	2649.231	
line4 TESTO4	2221.345	TRUE	13799	12.880	TRUE	8103.529	
line5_7 1 TESTO5-1	6888.000	FALSE	22935	6888.000	TRUE	6888.000	
line5_7 2 TESTO6	134.450	FALSE	9632	4592.000	TRUE	4592.000	
line5_7 3 TESTO7	134.450	FALSE	9632	2221.345	TRUE	2829.000	
line5_7 4 <UNIT>	<UNIT>	<UNIT>	<UNIT>	2755.200	<UNIT>	2755.200	
line5_7 5 TESTO8	11245.200	TRUE	10	6888.000	TRUE	6888.000	
line1 TESTO1	10280.597	TRUE	15018	987.432	FALSE	10280.597	
line2 TESTO2	4387.261	FALSE	14343	987.432	FALSE	4387.261	

5.19.1.2.6 Delete row

This command cancels a row complying the following requirements:

- row owning the currently focused cell, or
- row owning the currently selected cell, or
- currently selected row.

Before deleting, the system prompts the user for confirmation.



Answer **Yes** to execute the wished operation.

The remaining rows are moved to the top.

If neither a cell nor a row is selected, nor element is focused , this command does not cause any effect.

5.19.1.3 Close

Closes the Data environment.

If any not saved changes are detected, the system prompts the user with a warning message and waits for an answer



Touch **OK** key to continue with closing operation.

Touch **Cancel** if, instead, saving is wished before closing. [Save](#) the table and then ask for closing again.



PAY ATTENTION

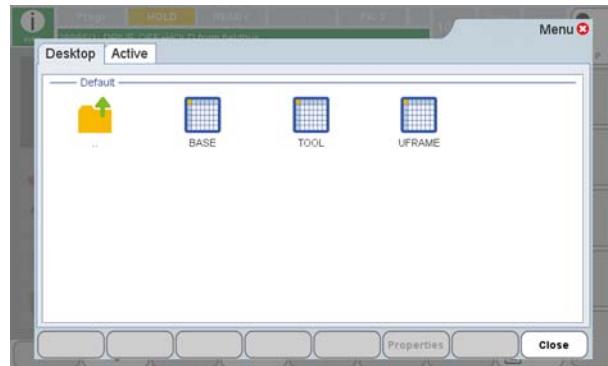
- when the state selector is switched to AUTO, if bit 23 of \$CNTRL_CNFG predefined variable is set to 1, the current values in DATA environment are AUTOMATICALLY SAVED.

5.19.2 System tables

'System Tables' refers to the automatically created tables, by Controller software (see [Fig. 5.34](#)):

- [BASE](#)
- [TOOL](#)
- [FRAME](#).

Fig. 5.34 - System tables



They are directly accessible when entering the Data Page.

It is also allowed accessing them from within a PDL2 program, by means of dedicated routines (see [par. 5.19.2.4 Accessing system tables from within a PDL2 program on page 193](#)).



For further information, please refer to [par. 5.10.1.3 Tool, Frame, Base on page 101](#) in Motion Page.



NOTE that both in AUTO and REMOTE status, tool/base/frame changing is disabled.

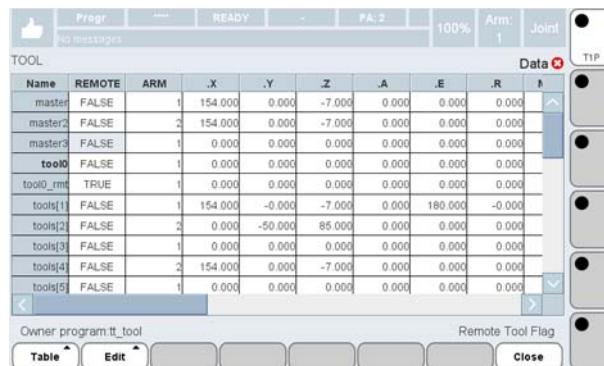
The meaning of the several columns of the system tables is shortly described in the blue line placed above the [Functional keys menu](#). The following [Fig. 5.35](#), [Fig. 5.36](#) and [Fig. 5.37](#) are shown as an [example](#).

Fig. 5.35 - BASE table - 'Position' column



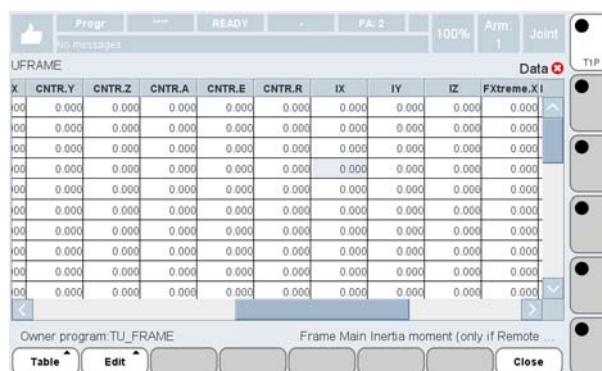
Name	ARM	X	Y	Z	A	E	R	DESCRIPTION
base0	1	0.000	0.000	0.000	0.000	0.000	0.000	base0
bases[1]	1	1856.420	-141.280	2093.875	-70.000	110.000	0.000	base_1
bases[2]	2	1856.420	141.280	2093.875	70.000	110.000	0.000	base_2
bases[3]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_3
bases[4]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_4
bases[5]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_5
bases[6]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_6
bases[7]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_7
bases[8]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_8
bases[9]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_9
bases[10]	1	0.000	0.000	0.000	0.000	0.000	0.000	base_10

Fig. 5.36 - TOOL table - 'REMOTE Flag' column



The screenshot shows a software interface for managing tools. At the top, there are tabs for 'Program', 'READY', 'PA: 2', '100%', 'Arm: 1', and 'Joint'. Below the tabs is a toolbar with icons for 'Data' (with a red exclamation mark), 'TIP', and other tool-related functions. The main area is a table titled 'TOOL' with columns: Name, REMOTE, ARM, X, Y, Z, A, E, R, T. The 'REMOTE' column contains boolean values (e.g., FALSE, TRUE). The table has scroll bars on the right and bottom. At the bottom of the screen, there are buttons for 'Table', 'Edit', and 'Close'.

Fig. 5.37 - FRAME table - 'Main Inertia moment' column



The screenshot shows a software interface for managing frames. At the top, there are tabs for 'Program', 'READY', 'PA: 2', '100%', 'Arm: 1', and 'Joint'. Below the tabs is a toolbar with icons for 'Data' (with a red exclamation mark), 'TIP', and other frame-related functions. The main area is a table titled 'UFRAME' with columns: X, CNTR.Y, CNTR.Z, CNTR.A, CNTR.E, CNTR.R, IX, IY, IZ, Fxtreme.XI. The 'Fxtreme.XI' column contains numerical values. The table has scroll bars on the right and bottom. At the bottom of the screen, there are buttons for 'Table', 'Edit', and 'Close'.

A detailed description follows about the listed above system tables.

5.19.2.1 BASE

The BASE type structure includes, among other elements, the following ones:

- a **Base0**, unique in the system
- a BASE type **34 elements array**, including all the Bases.

Each element is composed by the following information fields:

- Name
- Base Position
- Description.

5.19.2.2 TOOL

The TOOL type structure includes, among other elements, the following ones:

- a **Master Tool** for each Arm
- a **Tool0**, unique in the system
- a **Tool0 Remoto**, unique in the system
- a TOOL type **64 elements array**, including all the Tools, both normal and remote ones.

Each element is composed by the following information fields:

- Name

- Remote Tool Flag
 - Tool Position
 - Mass
 - Tool Center Position
 - Tool Extreme Position
 - Main inertia moments
 - Description.
-

5.19.2.3 FRAME

The FRAME type structure includes, among other elements, the following ones:

- a **Frame0**, unique in the system
- a **Frame0 Remoto**, unique in the system
- a FRAME type **64 elements array**, including all the Frames, both normal and remote ones.

Each element is composed by the following information fields:

- Name
 - Remote Frame Flag
 - Frame Position
 - Mass
 - Frame Center Position (for Remote Frame only)
 - Main inertia moments (for Remote Frame only)
 - Frame Extreme Position (for Remote Frame only)
 - Description.
-

5.19.2.4 Accessing system tables from within a PDL2 program

To access the described above system tables from within a PDL2 program, and process their contents, some dedicated routines are provided:

- [Routine for Tool and Frame setting for a specified Arm](#)
 - [Routine for Base setting for a specified Arm](#)
 - [Routines to get/set/save the Tool, Frame and Base value](#)
 - [Using the routines - examples](#)
 - [Error codes.](#)
-

5.19.2.4.1 Routine for Tool and Frame setting for a specified Arm

- [ToolFrame routine](#)
- [RmtToolFrame routine.](#)

ToolFrame routine

It activates, at the same time, both the **NON REMOTE** Tool index and Frame index to be used for a specified Arm.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE ToolFrame(ai_tool : INTEGER(); ai_frame : INTEGER(); ai_arm : INTEGER())
EXPORTED FROM TT_TOOL GLOBAL
```

Calling sequence: **ToolFrame** (ai_tool, ai_frame, ai_arm)

Parameters	Range
ai_tool : index of the being used Tool	1 .. 64
ai_frame : index of the being used Frame	1 .. 64
ai_arm : index of the Arm for which Tool and Frame are to be set	1..4



The three parameters are optional. In single Arm systems, it is suggested to ALWAYS specify the first two ones.

In multiarm systems, the Arm index is already specified in the table but, for the PDL2 program better readability and for a proper use of the tool0, it is suggested to ALWAYS specify the third parameter too.

RmtToolFrame routine

It activates, at the same time, both the **REMOTE** Tool index and Frame index to be used for a specified Arm.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE RmtToolFrame(ai_tool : INTEGER(); ai_frame : INTEGER(); ai_arm :
INTEGER()) EXPORTED FROM TT_TOOL GLOBAL
```

Calling sequence: **RmtToolFrame** (ai_tool, ai_frame, ai_arm)

Parameters	Range
ai_tool : index of the being used Remote Frame	1 .. 64
ai_frame : index of the being used Remote Frame	1 .. 64
ai_arm : index of the Arm for which Remote Tool and Frame are to be set.	1..4



The three parameters are optional. In single Arm systems, it is suggested to ALWAYS specify the first two ones.

In multiarm systems, the Arm index is already specified in the table but, for the PDL2 program better readability and for a proper use of the tool0, it is suggested to ALWAYS specify the third parameter too.



WARNING - Setting a REMOTE tool or frame is NOT allowed if the other element is not remote!

WARNING - in order to make the **WinC5G** file managing to work properly, it is a good habit to always insert the calling sequence to both **ToolFrame routine** and **RmtToolFrame routine** at the beginning of each routine including motion statements.

5.19.2.4.2 Routine for Base setting for a specified Arm

Base routine

It activates the Base for a specified Arm.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE base(ai_b_num : INTEGER; ai_arm : INTEGER ()) EXPORTED FROM TB_BASE GLOBAL
```

Calling sequence: **Base** (ai_base, ai_arm)

Parameters	Range
ai_base : index of the being used Base	1 .. 34
ai_arm : index of the Arm for which the specified Base is to be set	1..4

5.19.2.4.3 Routines to get/set/save the Tool, Frame and Base value

These routines are used to get/set/save, without using it, either the Tool, the Frame or the Bse value, from within a PDL2 program.

- [**ttri_tooldata_get routine**](#)
- [**turi_framedata_get routine**](#)
- [**tibri_basedata_get routine**](#)
- [**ttri_tooldata_set routine**](#)
- [**turi_framedata_set routine**](#)
- [**tibri_basedata_set routine**](#)
- [**ttri_tooldata_save routine**](#)
- [**turi_framedata_save routine**](#)
- [**tibri_basedata_save routine**](#).

ttri_tooldata_get routine

It allows reading the specified Tool value, from within a program.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE ttri_tooldata_get(ai_t_num : INTEGER; ap_pos : POSITION; ab_remote :  
BOOLEAN) : INTEGER EXPORTED FROM TT_TOOL
```

Calling sequence: **ttri_tooldata_get** (ai_tool, ap_pos, ab_remote)

Parameters	Range
ai_tool : index of the Tool the user wishes to get the value	1 .. 64
ap_pos : Tool position, returned by the routine	
ab_remote : remote Tool flag, returned by the routine	ON = remote OFF = normal

Returned values:

- 0 -> the routine has been properly executed
 - 1 -> wrong parameter(s).
-

turi_framedata_get routine

It allows reading the specified Frame value, from within a program.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE turi_framedata_get(ai_f_num : INTEGER; ap_pos : POSITION; ab_remote :  
BOOLEAN) : INTEGER EXPORTED FROM TU_FRAME
```

Calling sequence: ***turi_framedata_get*** (ai_frame, ap_pos, ab_remote)

Parametri	Range
ai_frame : index of the Frame the user wishes to get the value	1 .. 64
ap_pos : Frame position, returned by the routine	
ab_remote : remote Frame flag, returned by the routine	ON = remote OFF = normal

Returned values:

- 0 -> the routine has been properly executed
 - 1 -> wrong parameter(s).
-

tbri_basedata_get routine

It allows reading the specified Base value, from within a program.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE tbri_basedata_get(ai_f_num : INTEGER; ap_pos : POSITION) : INTEGER  
EXPORTED FROM TB_BASE
```

```
ROUTINE tbri_basedata_save : INTEGER EXPORTED FROM TB_BASE
```

Calling sequence: ***tbri_basedata_get*** (ai_base, ap_pos)

Parameters	Range
ai_base : index of the Base the user wishes to get the value	1 .. 34
ap_pos : Base position, returned by the routine	

Returned values:

- 0 -> the routine has been properly executed

- 1 -> wrong parameter(s).
-

ttri_tooldata_set routine

It is allowed to call this routine from within a program, in order to set a specified Tool.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE ttri_tooldata_set(ai_t_num : INTEGER; ap_pos : POSITION; ab_remote :  
BOOLEAN) : INTEGER EXPORTED FROM TT_TOOL
```

Calling sequence: ***ttri_tooldata_set*** (ai_t_num, ap_pos, ab_remote)

Parameters	Range
ai_tool : index of the being set Tool	1 .. 64
ap_pos : being set Tool position for the specified Tool	
ab_remote : it indicates whether the specified Tool is to be set as Remote or not	ON = remote OFF = normal

Returned values:

- 0 -> the routine has been properly executed
 - 1 -> wrong parameter(s).
-

turi_framedata_set routine

It is allowed to call this routine from within a program, in order to set a specified Frame.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE turi_framedata_set(ai_f_num : INTEGER; ap_pos : POSITION; ab_remote :  
BOOLEAN) : INTEGER EXPORTED FROM TU_FRAME
```

Calling sequence: ***turi_framedata_set*** (ai_frame, ap_pos, ab_remote)

Parameters	Range
ai_frame : index of the being set Frame	1 .. 64
ap_pos : being set Frame position for the specified Frame	
ab_remote : it indicates whether the specified Frame is to be set as Remote or not	ON = remote OFF = normal

Returned values:

- 0 -> the routine has been properly executed
 - 1 -> wrong parameter(s).
-

tbri_basedata_set routine

It is allowed to call this routine from within a program, in order to set a specified Base.

This Routine must be imported into the calling program, in the following way:

```
ROUTINE tbri_basedata_set(ai_f_num : INTEGER; ap_pos : POSITION; ai_ArmOptional :  
INTEGER()) : INTEGER EXPORTED FROM TB_BASE
```

Calling sequence: ***tbri_basedata_set*** (ai_base, ap_pos)

Parameters	Range
ai_base : index of the being set Base	1 .. 34
ap_pos : being set Base position for the specified Base	

Returned values:

- 0 -> the routine has been properly executed
 - 1 -> wrong parameter(s).
-

ttri_tooldata_save routine

It allows saving modifications made by means of the [***ttri_tooldata_set routine***](#).

This Routine must be imported into the calling program, in the following way:

```
ROUTINE ttri_tooldata_save : INTEGER EXPORTED FROM TT_TOOL
```

Calling sequence: ***ttri_tooldata_save***

No input parameters.

Returned values:

- 0 -> the routine has been properly executed
 - -1 -> saving not performed.
-

turi_framedata_save routine

It allows saving modifications made by means of the [***turi_framedata_set routine***](#).

This Routine must be imported into the calling program, in the following way:

```
ROUTINE turi_framedata_save : INTEGER EXPORTED FROM TU_FRAME
```

Calling sequence: ***turi_framedata_save***

No input parameters.

Returned values:

- 0 -> the routine has been properly executed
 - -1 -> saving not performed.
-

tbri_basedata_save routine

It allows saving modifications made by means of the [***tbri_basedata_set routine***](#).

This Routine must be imported into the calling program, in the following way:

```
ROUTINE tbri_basedata_save : INTEGER EXPORTED FROM TB_BASE
```

Calling sequence: ***ttri_basedata_save***

No input parameters.

Returned values:

- 0 -> the routine has been properly executed
- -1 -> saving not performed.

5.19.2.4.4 Using the routines - examples

```

.....
VAR li_id : INTEGER
VAR lr_x, lr_y, lr_z, lr_e1, lr_e2, lr_e3 : REAL
VAR ls_cfg : STRING[15]
VAR li_node : INTEGER
VAR lp_tmp_pos : POSITION
VAR li_tmp : INTEGER
VAR lb_remote : BOOLEAN
VAR li_data : INTEGER
VAR li_test : INTEGER
VAR ls_data : STRING[5]
VAR li_num_arm_base : INTEGER
.....

BEGIN
    li_node := 0
    IF li_tool_type = ki_tool THEN -- Tool on the robot flange
        FOR li_id := 1 TO ki_max_tool_frame DO
        -- If not configured with Nodal Move
            IF bit_test($CNTRL_CNF, 24, OFF) THEN
                li_tmp:=ttri_tooldata_get(li_id, lp_tmp_pos, lb_remote)
        .....

        -- load the calculated Tool values in the Position Table
        -- corresponding to the declared Tool number (li_data)
        li_test:= ttri_tooldata_set(li_data, $ARM_DATA[vi_num_arm].TOOL, TRUE)
        -- load the calculated Remote Tool values in the Position Table
        -- corresponding to the declared Tool number (li_data)
        li_test:= ttri_tooldata_set(li_data, $ARM_DATA[vi_num_arm].TOOL, FALSE)

        .....
        li_test:=turi_framedata_set(li_data,$ARM_DATA[vi_num_arm].UFRAME, TRUE)

        .....

```

```

li_test:=tbri_basedata_set(li_dato,$ARM_DATA[vi_num_arm_base].BASE,
li_num_arm_base)
.....
.....
-- Tool is unchanged and Frame is changed (4th table element)
ToolFrame(, 4)

.....
-- Frame is unchanged and Tool is changed (8th table element)
ToolFrame(8)

.....
-- Frame and Tool are not changed: it doesn't perform any action
ToolFrame(, , 1)

```

5.19.2.4.5 Error codes

Code	Description
43050	not existing Tool
43051	not existing Frame
43053	not existing Base
43054	wrong Arm
43056	uninitialized Variable
43059	wrong Parameters Data Type
43060	uninitialized Tool/Frame
43062	Remote Tool inconsistency
43063	not existing Remote ToolMaster
43064	wrong Parameters

5.19.3 Application tables

Sono tabelle create da chi fornisce l'applicativo Comau.

As already explained in the previous sections, touch the corresponding icon to select the wished table.

In the same way, once the table is open, the user can act inside it.

Note that opening and editing the application tables, are either enabled or not depending on the system state and the login profile required by the specific application.

Depending on the installed application, the loaded tables can be different. Please refer to the specific application manuals for a detailed description about them.

5.19.4 User tables

By means of the PDL2 language, the user can create customized tables to be handled in this User Page.



The procedure to create and add them to the Data Page is described in [par. 5.22.1 User table creation from DATA environment on page 271](#).

Once created and added to the DATA Page, such tables are handled in exactly the same way as the [System tables](#) and [Application tables](#).

This means that, to select the required table, touching the corresponding icon is needed. Furthermore, as already explained in the previous paragraphs, once the table is opened, it is possible to act inside it exactly in the same way.

Note that opening and modifying user tables are enabled/disabled depending both on the system state and the login profile (see [Cap.4. - Access to the Control \(login/logout\)](#) on page 43 - par. 4.2.2 [Summary table of access rights](#) on page 46 of current manual).

5.20 Setup page

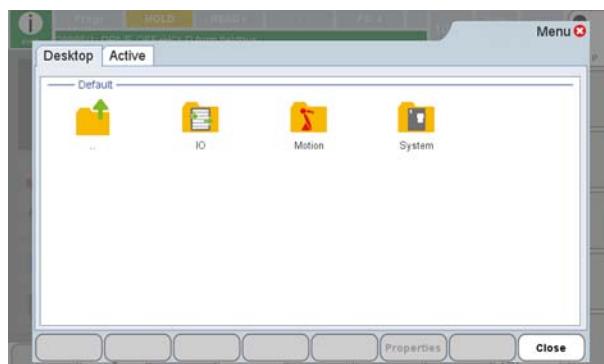
The Setup page is used to read and change the system settings
 To access this page, act as follows

- a. either press the **MENU** button in the **Membrane keyboard**, or touch the **MENU** key available in the **Functional keys menu** of the **Start Page**
- b. touch **Setup** icon (as shown in the below figure).

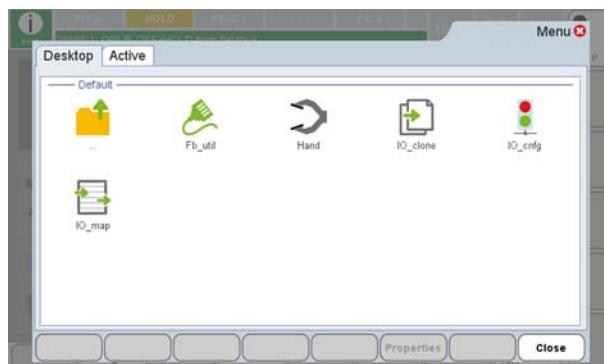


All the subenvironments are classified into the following groups:

- **IO**
- **Motion**
- **System.**



5.20.1 IO



This group includes all programs related to the I/O modules configuration, importing/exporting configurations, Fieldbus modules utilities, I/O ports mapping, Hands handling:

- [FB_UTIL](#)
- [HAND](#)
- [IO_CLONE](#)
- [IO_CNFG](#)
- [IO_MAP.](#)

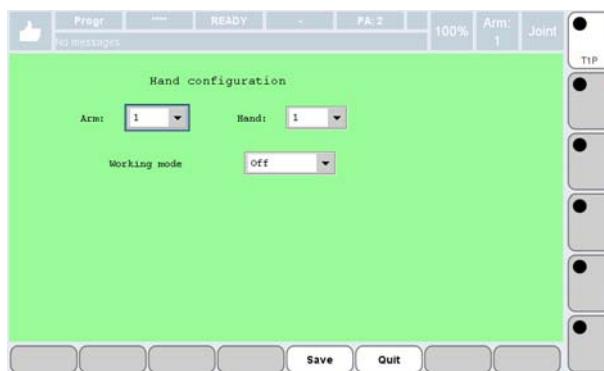
5.20.1.1 [FB_UTIL](#)

This icon allows accessing the [Fieldbus modules configuration](#) program.

For any information about the use of such a program, refer to [Chap. FB_util Program - Fieldbus modules utility on page 442](#).

5.20.1.2 [HAND](#)

This icon allows accessing to [Hands handling](#) for all the existing Arms in the system.



In this environment the user should select **Arm** and **Hand** which it is wished to operate on.

The wished **Working mode** is to be specified too, by touching the corresponding [Combobox](#). The system displays the available modalities list, as shown in the below figure. The user should either touch the wished item to select it and then **OK** to confirm, or touch the wished item twice.



The available working modes are as follows:

- [One](#)

- Dual
- Pulse
- Step.



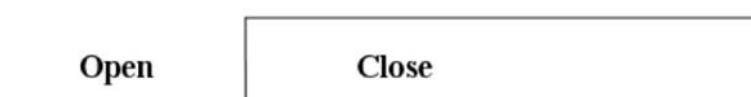
After setting the wished information, touch **Save** to make them permanent. A suitable message states the operation has been successfully completed (as shown in the below figure). Touch **OK** to confirm.

Then touch **Quit** to exit the Hand environment.

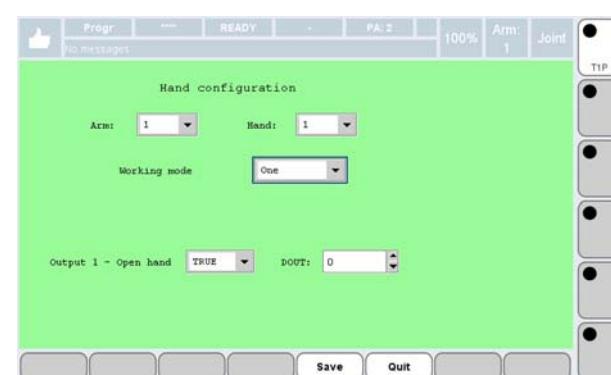


5.20.1.2.1 One

To configure the Hand with a **single output line**.



This choice causes the system to display some more fields shown in the below figure.

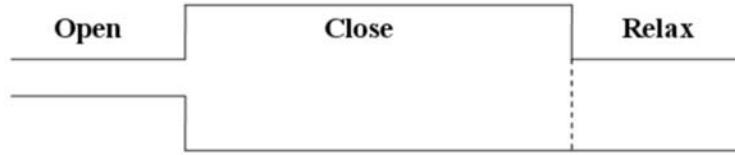


The User has to supply the following information:

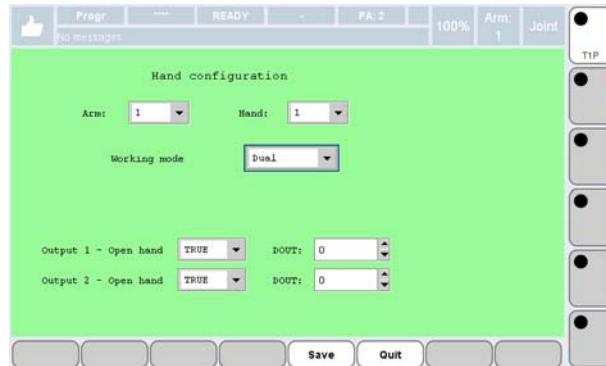
- specify whether the Open Hand signal has to be TRUE (positive logic), or FALSE (negative logic);
- specify \$DOUT port index where the output is to be mapped [1..512].

5.20.1.2.2 Dual

Allows configuring the Hand with **two output lines**.



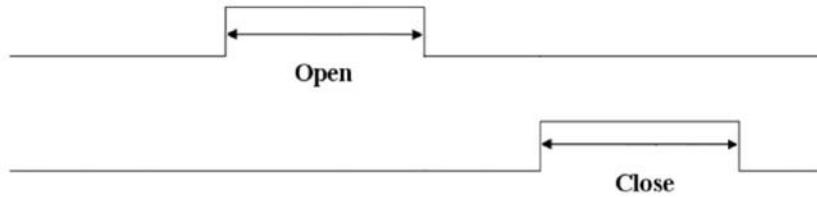
This choice causes the system to display some more fields shown in the below figure.



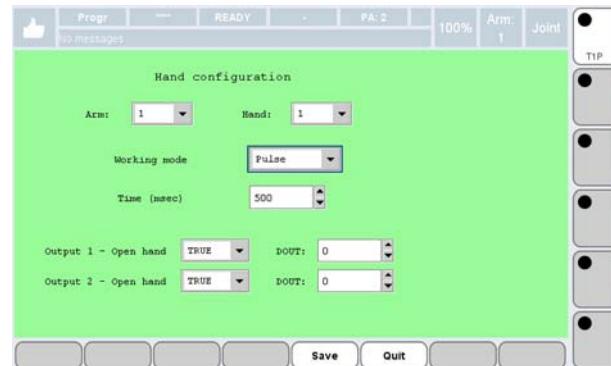
The User has to supply the same information as for [One Working Mode](#), but for both the Output ports.

5.20.1.2.3 Pulse

Allows configuring the Hand in **Pulse** mode, with **two output lines**.



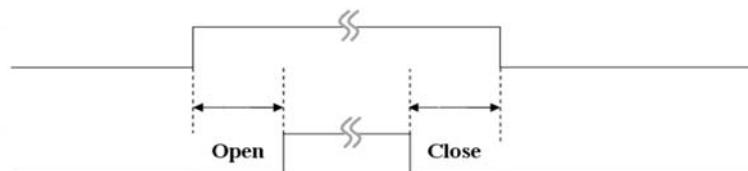
This choice causes the system to display some more fields shown in the below figure.



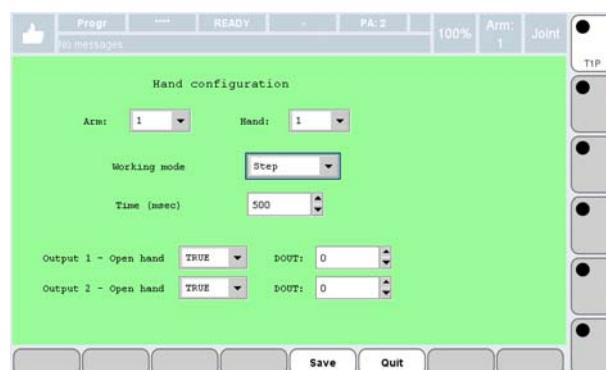
The information to be supplied by the User is the same as for the [Dual](#) operating mode, but it is now necessary to also specify the **pulse duration** (10 ..10000 ms), indicated by **Time (msec)** field.

5.20.1.2.4 Step

Allows configuring the Hand in **Step** mode, with **two output lines**.



This choice causes the system to display some more fields shown in the below figure.



The information to be supplied by the User is the same as for the [Dual](#) operating mode (to which reference is made), but it is now necessary to specify also **the time interval between one step and the next one** (between 10 and 10000 ms), in the **Time (msec)** field.

5.20.1.3 IO_CLONE

This icon allows accessing the program for importing/exporting the Input/Output modules configuration.

For detailed information about such a program, please refer to [Chap. IO_CLONE Program - I/O Configuration Export/Import on page 489](#).

5.20.1.4 IO_CNFG

This icon allows accessing the program to configure the I/O modules in the system.

For detailed information about such a program, please refer to [Chap. IO_CNFG Program - I/O modules configuration on page 420](#).

5.20.1.5 IO_MAP

This icon allows accessing the program to map the I/O points belonging to all the existing Devices in the system.

For detailed information about such a program, please refer to [Chap. IO_MAP Program - I/O ports mapping on page 464](#).

5.20.2 Motion



- [Frames](#)
- [Calib](#)
- [Conveyor](#)
- [IRegions](#)
- [Mounting](#)
- [Portal](#)
- [Positioners](#)
- [Posit_Arm](#)
- [Rail](#)
- [StrokeEnd.](#)

5.20.2.1 Frames

This subenvironment allows calculating the values for positioners BASE, TOOL and UFRAME, in automatic and guided way.

For further information about the use of this program, refer to [Chap. TO_SET Program - Tool handling on page 211](#), in **Motion Programming** manual.

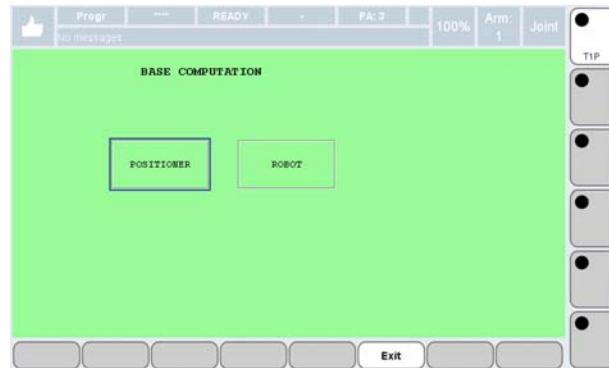
- [Base](#)
- [Tool](#)
- [Uframe](#)

5.20.2.1.1 Base

This subenvironment allows **calculating the positioners BASE**, according to the following requirements:

- [Cooperative Motion \(optional feature\) \(CR17926200\)](#) software option must have been enabled
- the user must be logged in on the TP5 Teach Pendant
- positioners axes must be present in the system configuration

The home page of the Base subenvironment is shown in the following figure.



5.20.2.1.2 Tool

This subenvironment allows **defining the working tool dimensions** when mounted onto the robot flange.

In such a situation the robot is used as a measurement device.

The tool calculation must always be executed the first time a tool is mounted onto the flange, or when, referred to the initial situation, the robot has been moved or calibrated.

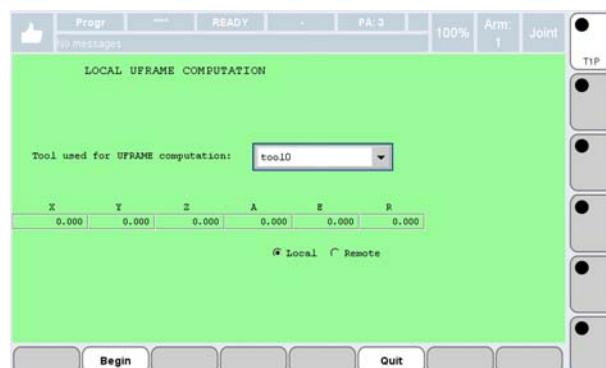
The main page of the Tool subenvironment is shown below.



5.20.2.1.3 Uframe

This subenvironment allows **defining a new reference frame** for teaching the work positions. The calculation accuracy is the one of the robot, used as a measurement device in this situation.

The main page of the Uframe subenvironment is shown below.



5.20.2.2 Calib

This subenvironment provides a set of commands allowing the Arm configuration, according to the user's needs.



Operate from within the [Motion Page](#), [Basic](#) subpage, if changing the current arm is required.



The available keys in the [Functional keys menu](#) are as follows:

- [Turn Set](#)
- [Calib](#)
- [Save](#)
- [Load](#)
- [Edit](#)
- [Close](#).



In these sub-pages, editable data are the ones included in columns with a white background (CAL_USER and CAL_DATA). The other ones are read-only data.

5.20.2.2.1 Turn Set

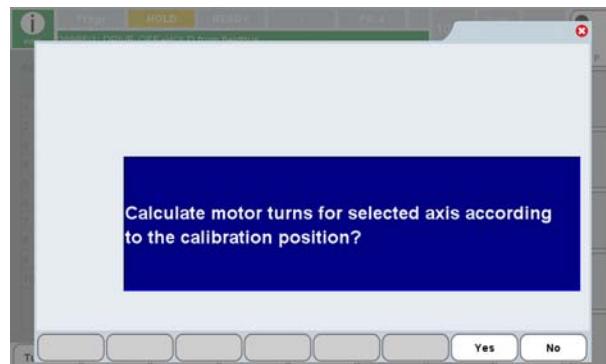
Sets the encoder turns counter at the motion software level. Before issuing the command, the arm has to be moved to the calibration position (within an encoder turn quadrant). This command is useful in case the pre-setting has been lost, e.g. due to a read error of the motors turns.



The system has to be in PROGR status (modal selector switch on T1 position) and the drives must be kept ON during the whole turn-set operation.

To activate the Turn Set procedure, short touch the **Turn Set** key. Before starting execution, the System prompts the User for confirmation:

Fig. 5.38 - Turn Set command confirmation



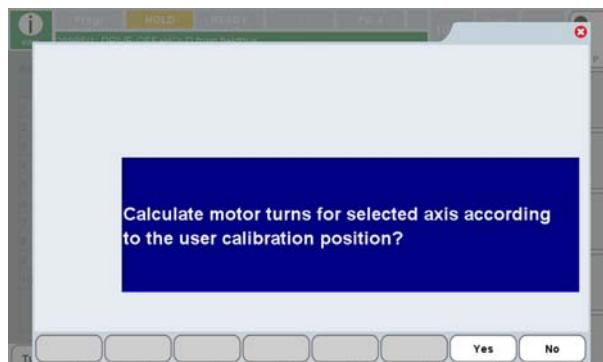
Answer **Yes** to execute the command.

It is also allowed to use the previously taught User Calibration Position (**Turn Set** command - [Fig. 5.39](#)). To activate it, long touch **Turn Set** key, then touch **User** item.

Fig. 5.39 - Turn Set command



The system prompts the user for confirmation before executing the Turn Set operation. Touch **Yes** to confirm.



5.20.2.2.2 Calib

Short touching this key, it is allowed to execute the calibration procedure either for the whole Arm or for the specified axis/axes of such an arm.

Prima di inoltrare il comando, l'arm deve essere posto nella posizione di calibrazione. Per indicare l'asse desiderato, occorre selezionarlo prima di avviare la procedura. Selezionando tutti gli assi (toccare + **SHIFT**) si ottiene la calibrazione sull'intero Arm corrente.

Before issuing the command, the Arm must be moved to the Calibration Position. To

indicate the wished axis, select it before starting the procedure. Selecting all the axes (toccoare + **SHIFT**) means to calibrate the whole current arm.



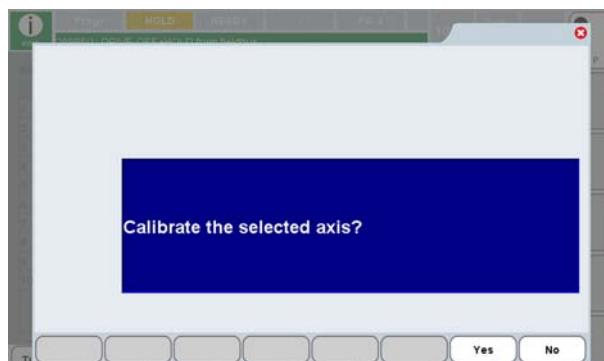
NOTE THAT data are saved when ALL the axis have been calibrated.



The system must be in PROGR state and the drives must be kept ON for the whole calibration operation.

The calibration data are automatically saved in NVRAM memory, in the configuration file (<\$SYS_ID>.C5G) and in the ASCII calibration file <\$SYS_ID>_CAL<num_arm>.PDL.

The system prompts the user for confirmation before executing the Calibration operation.



Touch **Yes** to confirm.

If instead the user wishes to activate the calibration operation using any options, long touch the **Calib** key (see Fig. 5.40).

Fig. 5.40 - Calibration options



The available options are as follows:

The available options are as follows:

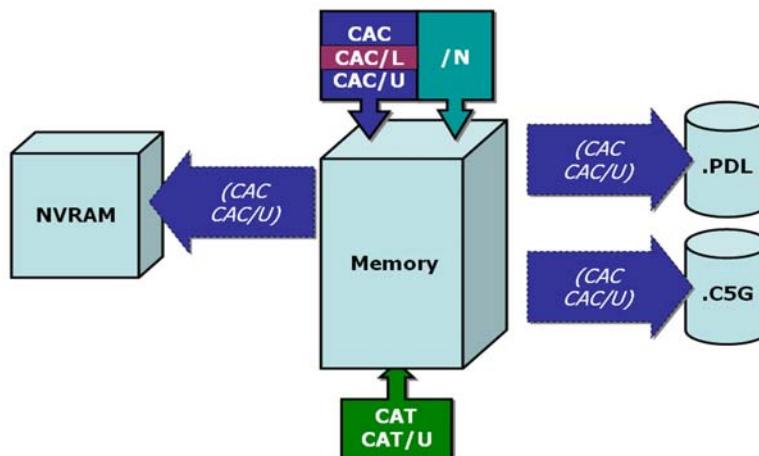
- **Learn (CAC/L)**

Learns the current position as the user calibration position (\$CAL_USER). The learning procedure is executed on all the selected axis. This command DOES NOT perform any saving operation.

- **NoSave (CAC/N)**
NO saving is automatically performed of .C5G file, of <\$SYS_ID>_CAL<num_arm>.PDL ASCII calibration file and of the calibration data in NVRAM memory.
- **User (CAC/U)**
The previously taught or assigned user position (\$CAL_USER) is used as the calibration position.
- **User NoSave (CAC/UN)**
This is a combination of previous **User (CAC/U)** and **NoSave (CAC/N)** options.

Touch the wished option: a prompt message is displayed waiting for the User to confirm it. Answer **Yes** to start the Calibration procedure; **No** to cancel it.

Fig. 5.41 - Data handling in Turn set (CAT) and Calibration (CAC) commands



5.20.2.2.3 Save

Short touching this key causes the axes related data to be saved, from the Controller memory to NVRAM memory, and to automatically update <\$SYS_ID>.C5G configuration file and <\$SYS_ID>_CAL<num_arm>.PDL calibration file.

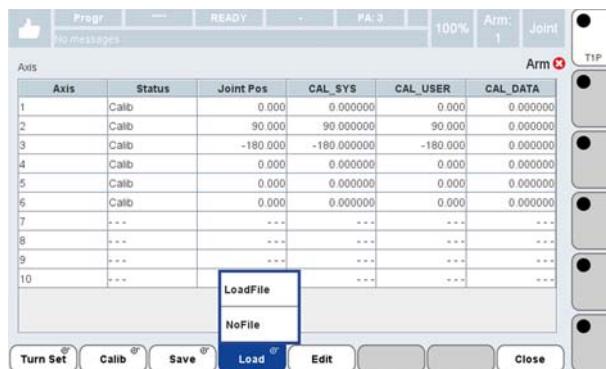
The user can also decide whether disabling automatic updating or not, by long touching **Save** key and choosing **NoFile** option.



5.20.2.2.4 Load

Short pressing this key causes the arm data (calibrated axes mask) and each axis data (axis length, calibration values and calibration constants), to be loaded from NVRAM memory to the Controller. The configuration file (`<$SYS_ID>.C5G`) and the calibration file (`<$SYS_ID>_CAL<num_arm>.PDL`) are automatically updated.

It is also allowed to choose between two options, by long touching **Load** key and touching the wished item.

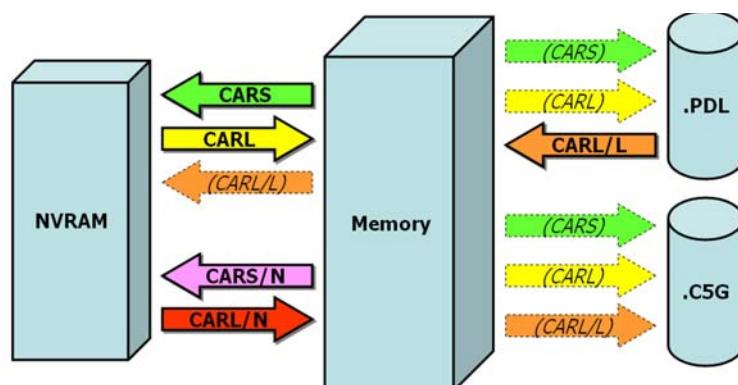


The available options are as follows:

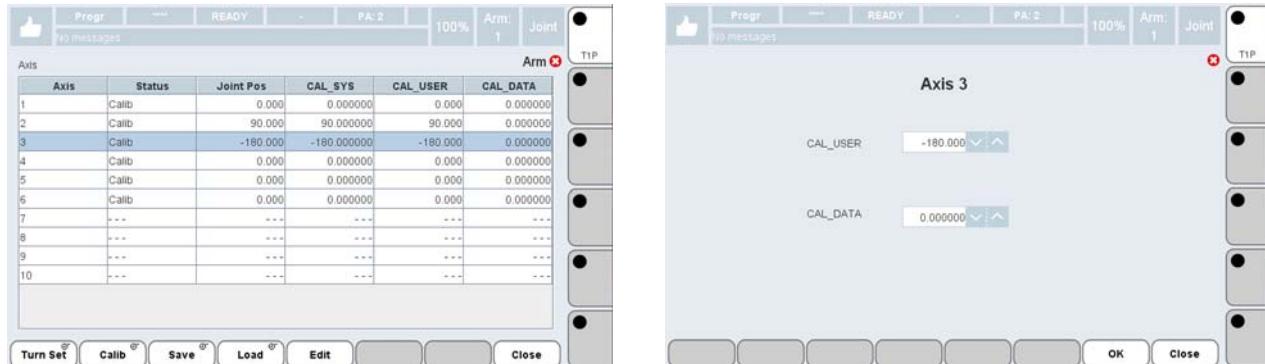
- **LoadFile (CARL/L)**
to enable axes data to be loaded, directly reading them from the calibration file, and both `<$SYS_ID>.C5G` configuration file and NVRAM memory to be saved.
A **.COD** file is automatically created having the same name of the calibration file (`<$SYS_ID>_CAL<num_arm>.PDL`) which is executed.
- **NoFile (CARL/N)**
to disable automatic saving for both `<$SYS_ID>_CAL<num_arm>.PDL` calibration file and `<$SYS_ID>.C5G` configuration file.

A diagram follows to summarize the described above commands (**Save** and **Load**):

Fig. 5.42 - Saving and loading calibration data



5.20.2.2.5 Edit



Touching this key (see above figure on the left) a subpage is opened in which the user is allowed to modify CAL_USER and CAL_DATA values related to the selected axis, as shown in the above figure on the right. Touch **Close** to exit without modifying anything.

5.20.2.2.6 Close

Closes the calibration environment.

5.20.2.3 Conveyor

This environment allows handling setup operations for tracking functionality, by means of sensors, conveyor trucks, etc., integrated into the robotic system.

Fig. 5.43 - Tracking tables summary table



Index	Mask of arms	Application
1	1 - 3 -	Conveyor
2	- 2 - -	Conveyor
3	- - 3 -	Conveyor
4	- - - -	
5	- - - -	
6	- - - -	
7	- - - -	
8	- - - -	
9	- - - -	
10	- - - -	

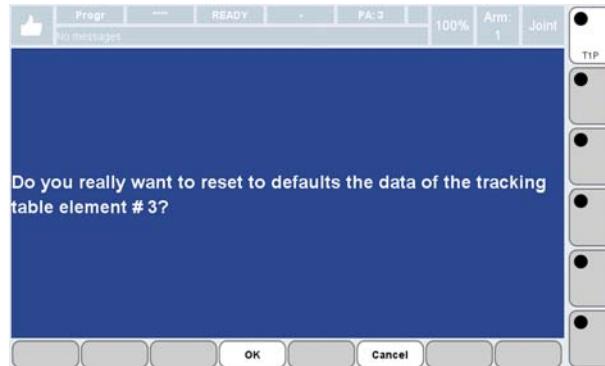
The main page includes a Summary Table of 10 elements corresponding to the currently existing Tracking tables in the system.

The Summary Table columns contain the following information:

- **Index**
is the Tracking Table index
- **Mask of Arms**
is the Arm Mask, associated to such a Tracking Table; only the selected Arms will be enabled for tracking
- **Application**
is the application (e.g. Conveyor) which is associated to such a Tracking Table.

The following commands are available in the [Functional keys menu](#):

- **Edit** - allows editing the selected Tracking Table (see [Tracking Table editing procedure](#))
- **Reset** - all data in the selected table are set back to the default values; the user is asked to confirm



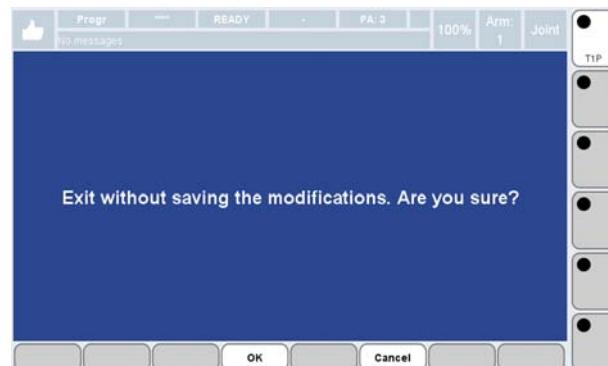
- **Save** - saves all data belonging to the Summary Table to **.C5G** file, WITHOUT leaving the Tracking environment.



Perform a Controller **Cold Restart** (see [par. 5.7.2.2.1 Cold](#) on page 91) operation, in order to make all settings operational.



- **Quit** - quits the Tracking environment without saving; the User is asked to confirm.

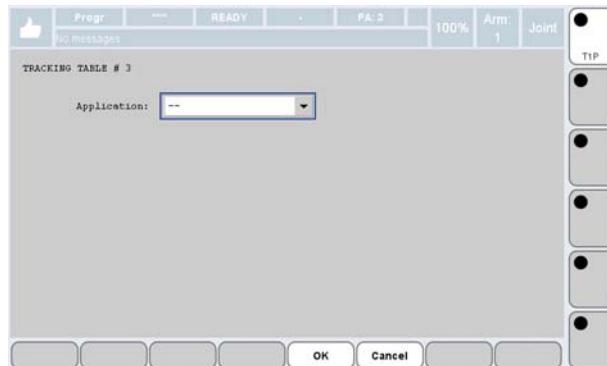


5.20.2.3.1 Tracking Table editing procedure

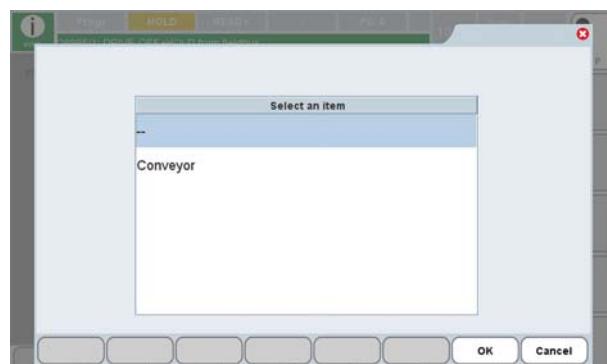
- a. Select the wished Tracking Table touching the corresponding row.

b. Touch the **Edit** key

- b.1 If it is a new Tracking Table, the only displayed field is **Application**. If it is an already existing table, instead, all the fields are displayed, thus directly go to [c. step](#)



- b.2 Touch the **Combobox** to open the Applications list.

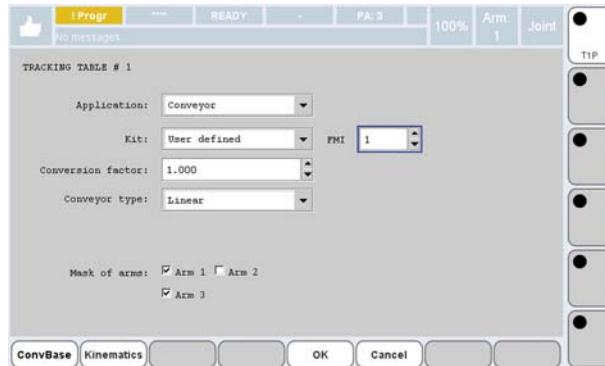


- b.3 Touch the wished application to select it.

- c. The complete page includes the following fields (see [Fig. 5.44](#)):

- **Application** - is the wished tracking application type (e.g. Conveyor)
- **Kit** - is the Tracking device. The available ones are
 - Predefined
 - User defined
- **FMI** - in case of Predefined Kit, \$FMI is automatically set by the system; in case of User defined Kit, the User must specify the wished \$FMI
- **Conversion factor** - is the conversion factor from the input port value unit of measure to the conveyor unit of measure
- **Conveyor type** - is one of the following
 - linear
 - circular
 - bending
- **Conveyor radius** - to be specified (in mm) only for circular Conveyor
- **Mask of Arms** - indicates the enabled Arms for tracking operations to the C5G Controller. Select the wished checkbox(es).

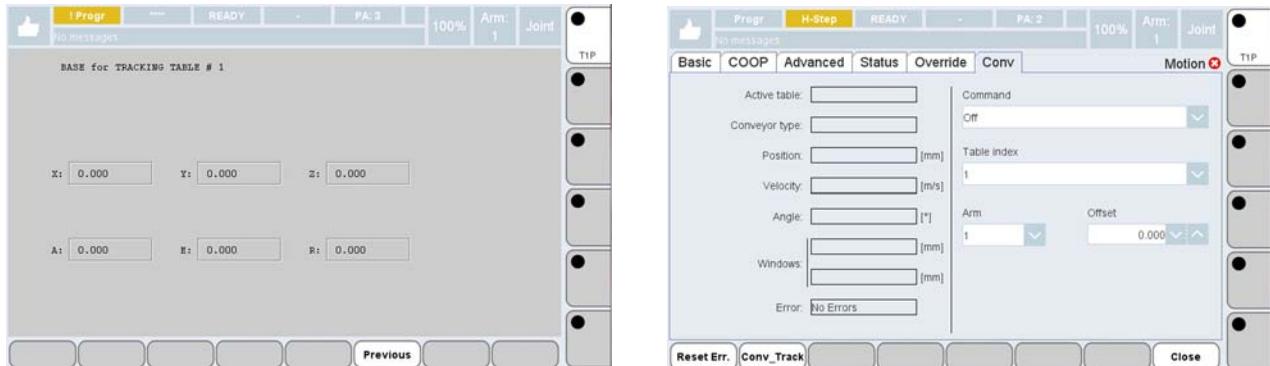
Fig. 5.44 - Definition/Edit Page



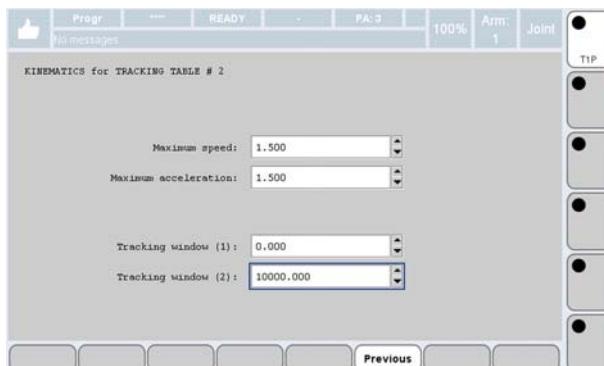
Select the wished values in the listed above fields according to the editing modes of the different **Field types**.

- d. To properly perform tracking operations, a base frame is needed on the conveyor base. Touch **ConvBase** key to view the current Base used by the selected Tracking Table (see image on the left in Fig. 5.45). If the User wishes to modify such values, it is needed to access them from within the [Motion Page - par. 5.10.6 Conveyor Tracking \(optional feature\)](#) on page 108 (see figure below, on the right).

Fig. 5.45 - ConvBase



- e. Touch **Previous** key to go back to the previous page and continue editing the wished Tracking Table (see Fig. 5.44)
- f. Touch **Kinematics** key to define the following values, associated to the selected tracking Table:
 - **Maximum speed** and **Maximum acceleration** - they are limit values for the robot speed (in m/s) and acceleration (in m/s²), to be used while tracking
 - **Tracking window** (1 and 2) - the first value specifies the conveyor quote from which the robot must start with tracking; the second value represents the conveyor quote after which no tracking with the robot is allowed.

Fig. 5.46 - Kinematics

Perform the wished modifications, according to the editing modes of the different **Field types**.

- g. Touch **Previous** key to go back to the previous page ([Fig. 5.44](#)) and complete the definition/edit session
 - h. Touch either **OK** key to confirm all the data in the selected Tracking Table, or **Cancel** to delete all the modifications.
- In any case, the program goes back to the Summary Table ([Fig. 5.43](#)) in the Conveyor main page.

5.20.2.4 IRegions

This environment allows handling Interference Regions setup; they are interference/interchange volumes shared between Robot and Robot, and among Robot and structures.

The home page shows the Interference Regions (**IR**) table, including 32 elements (see [Fig. 5.47](#)).

Fig. 5.47 - Interference Regions (IR) Table

#	Name	Type	Active	Permi	Status	Arm	Shape
1	MY_REG	Forbidden	OFF	ON	Out	1	Cylinder
2	MEA_1234	Forbidden	OFF	OFF	Out	1	Plane
3	JNT-FBD	Forbidden	OFF	OFF	Out	1	Joint
4	MN	Forbidden	OFF	OFF	Out	1	Plane
5	Parall-2	Monitored	OFF	OFF	Out	1	Parallel
6	Hyb-Sphere	Hybrid	OFF	OFF	Out	1	Sphere
7		Fortblden	OFF	OFF	Out	1	Parallel
8		Fortblden	OFF	OFF	Out	1	Parallel
9		Fortblden	OFF	OFF	Out	1	Parallel
10		Fortblden	OFF	OFF	Out	1	Parallel
11		Fortblden	OFF	OFF	Out	1	Parallel

Columns include the following information:

- **#**
is the ordinal index of the selected IR
- **Name**
is the IR name, assigned by the User
- **Type**
is the Region Type, chosen among the following

- Forbidden

the **TCP** (Tool Center Point) will never be allowed to enter such a Region. The System takes care of decelerating the robot in advance, in order to reach zero speed on the Region boundary. It is exactly the opposite of the [Allowed](#) region.

- Allowed

with this type of region, the **TCP** (Tool Center Point) is only allowed to move INSIDE it. The System takes care of decelerating the robot in advance, in order to reach zero speed on the Region boundary. It is exactly the opposite of the [Forbidden](#) region.



To use Allowed Interference Regions, [Advanced Interference Regions](#) option (code [CR17926218](#)) is required.

- Monitored

each time the **TCP** either enters or leaves such Regions, a logical Output port (chosen by the User) is modified to the predefined value.

- Hybrid

the use of such a region type is upon “reservation”, by means of assigning a boolean value to the associated Output port; the access consent is given when the status modification of the associated Input port. One more Output port is used to indicate the Arm presence/absence in the region.



To use Hybrid Interference Regions, [Advanced Interference Regions](#) option (code [CR17926218](#)) is required.

- **Active**

indicates whether the IR is currently active or not.

This could occur after the region parameters have been defined and the Permanent flag is enabled (by this interface)

- **Perm**

indicates whether or not the IR has been activated by this interface. No one else can modify its parameters, just reading them. The Region can be left by deselecting the Permanent flag in the suitable Region page (see [Fig. 5.50](#)).

- **Status**

indicates whether the associated Arm is inside or outside the Region

- **Arm**

is the Arm which is associated to the Region

- **Shape**

is one of the following Region shapes

- [parallelepiped](#),
- [sphere](#),
- [cylinder](#),
- [plane](#),
- [joints](#).

The available keys in the functional keys menu of this Page, are the following:

- **Edit** - allows data editing for the selected region (see [par. 5.20.2.4.1 Interference Regions editing procedure on page 220](#))



WARNING - The regions can be modified by this interface, in PROGR status ONLY.

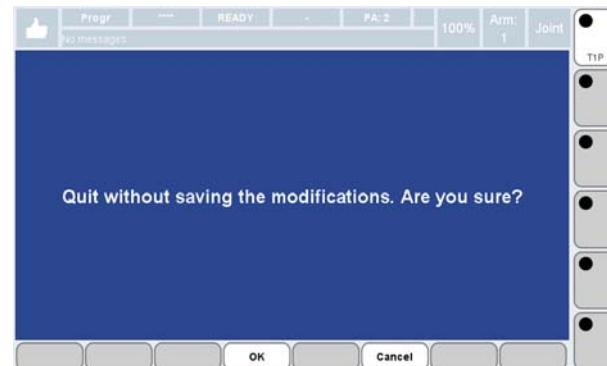
- **Reset** - sets back all the selected region data to the default values
- **Save** - the regions configuration is saved to **.C5G** file (see [Fig. 5.48](#)).

Fig. 5.48 - Save command



- **Quit** - exits from the environment, prompting the User for confirmation (see [Fig. 5.49](#)). Any modification is lost if not previously saved (by means of the described above **Save** command).

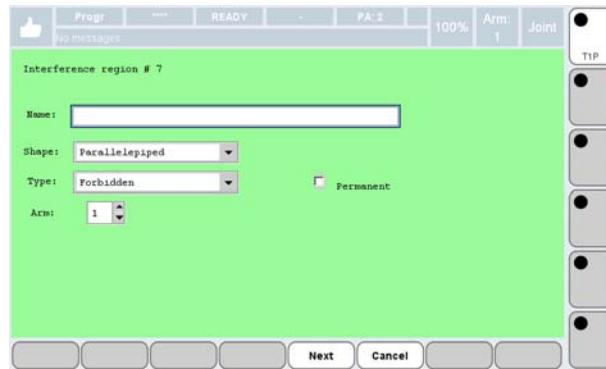
Fig. 5.49 - Quit command



5.20.2.4.1 Interference Regions editing procedure

- a. Select the wished region, in the table,
- b. touch either **Edit** key or press **ENTER** in the [Membrane keyboard](#).

Fig. 5.50 - Edit page - example



- c. First of all choose a **Name** and type it in the suitable field. Press **OK** to confirm;
- d. to specify the region **Shape**, touch **Shape** field to open the corresponding menu;
- e. choose the wished shape and touch **OK** to confirm;
- f. to specify the region **Type**, touch **Type** field to open the corresponding menu;
- g. choose the wished type and touch **OK** to confirm.
- h. If the chosen **Type** needs some logic ports (either **Monitored** or **Hybrid** region - see examples in Fig. 5.51), also indicate
 - **port Type** (\$DOUT, \$OUT, \$BIT, \$WORD, \$FMO, \$DIN, \$IN, \$FMI)
 - **Index**
 - **Logic** to be used (ON / OFF). As soon as the Arm enters such a region, the port is set to the value which is indicated by this field.

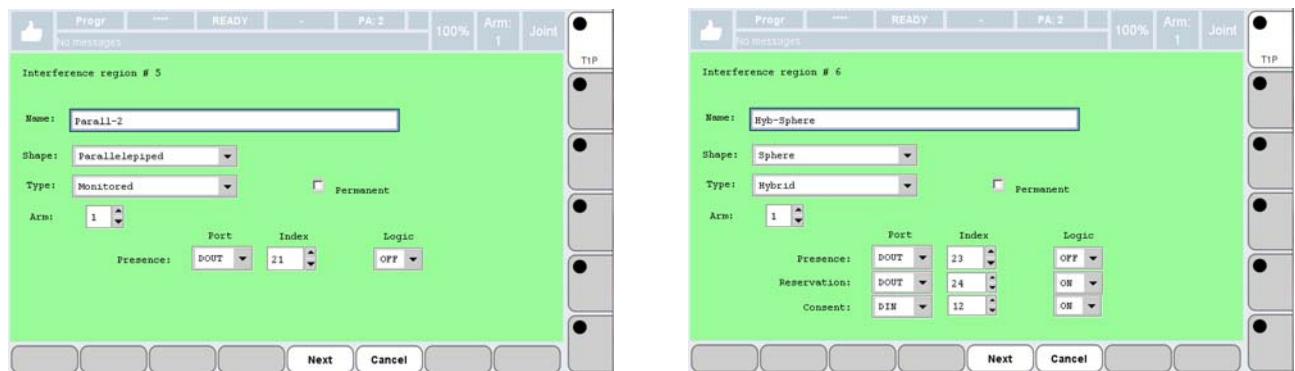
When using analog ports (e.g. \$WORD), also specify the Bit chosen as the Output signal.



WARNING - all the specified logic ports must have already been mapped!

Confirm each data insertion according to the used **Field types**.

Fig. 5.51 - Examples - Monitored and Hybrid Regions



- i. indicate the **Arm** associated to such a region;

- j. touch **Permanent** checkbox if it is wished to activate the region as soon as the region data editing is completed;
- k. touch either **Next** to continue, or **Cancel** to delete the current operation and go back to the [Interference Regions \(IR\) Table](#), starting again from a. step.



WARNING - FORBIDDEN Region

When **Next** key is touched, **Permanent** checkbox is selected and the robot is INSIDE the region, if the user tries to jog the robot arm, the system displays the following message:

62541 - IR yy reached by Arm xx

Act in the following way:

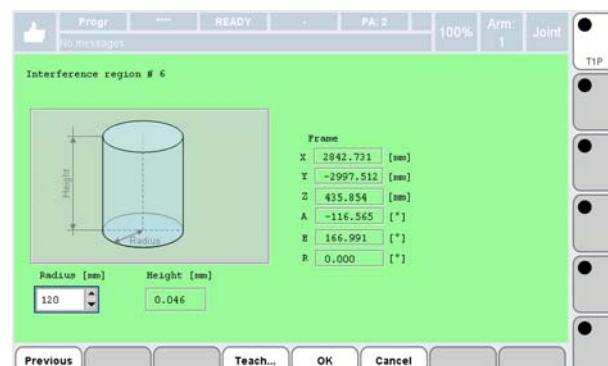
- **deselect Permanent checkbox**
- **jog until the robot is outside the Forbidden region**
- **select Permanent checkbox again.**

- I. A page is displayed, which is different depending on the region shape, allowing to enter any other needed data (in the example shown in [Fig. 5.52](#) the cylinder base radius is required).



For [Joint](#) type regions, refer to [par. 5.20.2.4.2 Tips about special shape regions on page 225](#).

Fig. 5.52 - Inserting Shape specific data

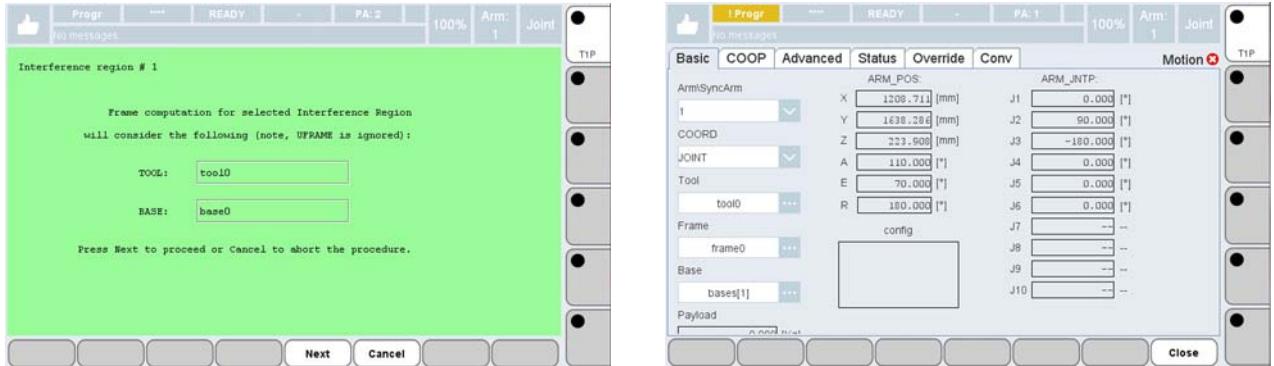


- m. Touch **Teach..** key to continue. Except for Joint regions, it is needed to teach some points to unambiguously define the region.



For [Plane](#) type regions, refer to [par. 5.20.2.4.2 Tips about special shape regions on page 225](#).

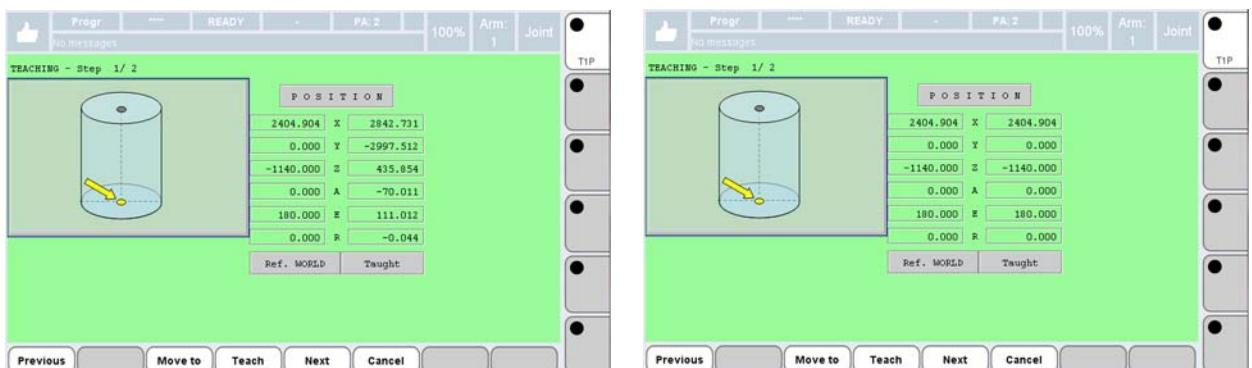
Fig. 5.53 - Arm settings for the TEACH phase



An informational page is displayed about Tool and Base coordinates which are used for the TEACH phase (see Fig. 5.53 on the left): they are the currently selected ones from Motion **General** sub-page (see Fig. 5.53 on the right)

- n. Touch either **Next** key to continue, or **Cancel** key to delete the operation and go back to the previous page (Fig. 5.52), leaving the Teach phase.
- o. One page is displayed for each needed point to define the currently selected region. The total amount of points depends on the chosen shape type.
In the topmost row (see Fig. 5.54 on the left, highlighted in red), the current TEACH phase step is displayed as well as the total amount of required steps for the chosen shape type.
Two columns are also displayed:
 - left column shows the TCP current position coordinates in the WORLD reference frame,
 - right column shows the coordinates of the position pointed out by the yellow arrow (if previously taught).

Fig. 5.54 - TEACHing positions - 1

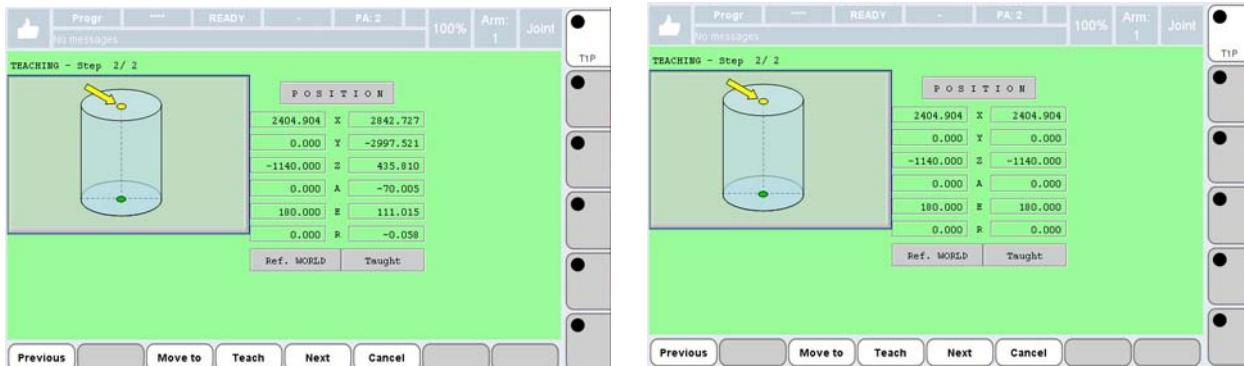


Jog the Arm to the wished position, which will thus be associated to the point indicated by the yellow arrow (see above figure, on the right).

- p. Touch **Teach** key to record the position.
- q. Touch either **Next** key to teach the next point, or **Cancel** to delete the TEACH phase and go back to I. step; the user is prompted for confirmation before deleting the TEACH operation.

- r. A page is displayed to TEACH the next point (see Fig. 5.55, on the left); operate as described for o. step, for each displayed page.

Fig. 5.55 - TEACHing positions - 2



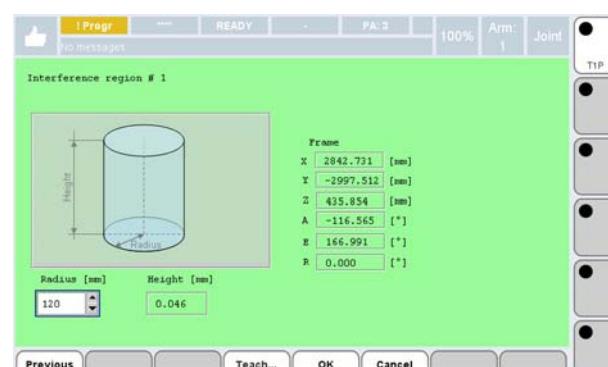
- s. At the end of the TEACH phase for each required point, either touch **Next** key to continue, or **Cancel** key to delete the TEACH phase and go back to l. step.



NOTE THAT if the Arm must be moved to a taught position, the **Move to** key is available, keeping both the **DRIVE ON** status and the **START** button pressed; to interrupt the movement, use the **Abort** key.
Such a movement is of joint type.

- t. A summarizing page is displayed of all the calculated values, also including the **User Frame** associated to such a region (see Fig. 5.56); in our example the cylindrical region height has been calculated.
The data are read-only.

Fig. 5.56 - Calculated Frame



Touch either

- **OK** key to confirm the results and exit from the TEACH phase, affirmatively answering to the related prompt; or
- **Previous** to go back to the previous page (step r.), or
- **Cancel** to delete the TEACH phase and go back to l. step

- u. Go back to the initial table ([Interference Regions \(IR\) Table](#)).



Please, note that the calculated values are NOT saved in the **.C5G** file, until this is explicitly requested by means of Save command, from the main Page of the current functionality (see Fig. 5.47)!

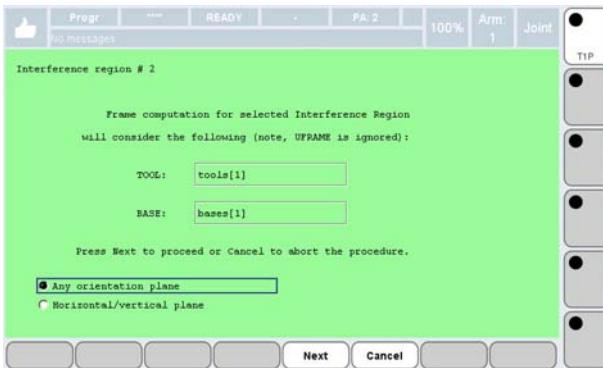
5.20.2.4.2 Tips about special shape regions

The following regions, having special shapes, have to be handled in different ways while editing them.

A detailed description of the required steps is provided, for the regions with the following shape:

- [Plane](#),
- [Joint](#).

Plane



When the region shape is a PLANE, this environment allows choosing between two different modalities, in order to teach the region points (see figure above).

The user must choose between the following TEACH modalities:

- [Any orientation plane](#)
- [Horizontal/vertical plane](#).

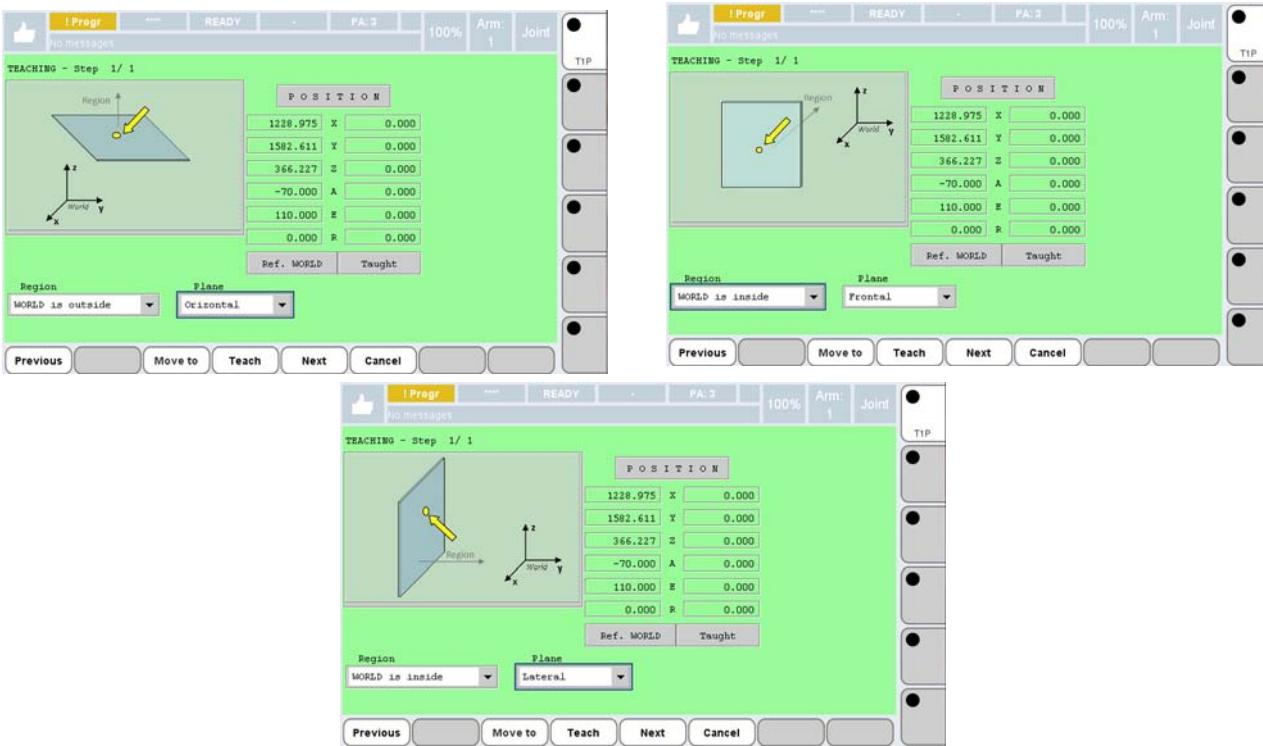
Any orientation plane

In this case **three points** are required to unambiguously define the plane.

The procedure is similar to the previously described one for [Cylinder](#), [Parallelepiped](#) and [Sphere](#) (from n. step to the end of the procedure).



Horizontal/vertical plane



In these cases the system requires to teach **just one position**.

Touching **Next** key, the first TEACH page is displayed.

The region is defined depending on whether the WORLD reference frame is either **inside** or **outside**, referred to it.

The three described cases are shown in the figures above.

The user is requested to specify the characteristics of the wished Plane region

- Region
 - WORLD is inside (left high example)
 - WORLD is outside (high and center right examples)
- Plane
 - Horizontal
 - Frontal
 - Lateral

and perform the TEACH phase for just one point, which is enough to define the Plane, provided that the plane orientation has already been specified. At the end, go back to the [Interference Regions editing procedure execution](#), from **s.** step till the end.

Joint

In the displayed page three columns are shown which include information for each joint of the Arm:

- Negative - negative stroke end for the joint
- Positive - positive stroke end for the joint
- ARM_JNTP - joint current position.

The values belonging to the first two columns (stroke end) are only visible if the

corresponding checkbox is selected (example: see joint 1 and 2 in the figure below).



To define a region, different from the one defined by the stroke end values (without anyway modifying them), touch the checkbox associated to the being modified joint(s), to select it; edit the required value (as shown in the following figure, on the left) and then touch **OK** key to confirm.



At the end of the editing phase, continue the procedure from [S. step](#) till the end.

5.20.2.5 Mounting

When **Mounting** icon is selected, a screen page is displayed showing the current mounting position; the user is allowed to modify it in order to declare to the System how the robot is actually mounted:

- on ceiling,
- on floor.



If the robot type can be mounted on a sloped plane, the following mounting modes are also allowed:

- on a sloped plane referred to the floor,
- on a sloped plane referred to ceiling.



Mounting on a sloped plane is only available for type NS, NM and SIX robots.

To declare the type of robot **Mounting**, proceed as follows (reference is made to the previous figure):

- a. touch the required item to select it; if the **Mounting** is not **SLOWLY TILTED** type, directly go to step b.
- a.1 if the mounting is **SLOWLY TILTED** type, insert the slope **ANGLE**; if the slope is **related to the floor**, go to step b.;
- a.2 if the slope is **related to ceiling**, touch **FROM CEILING** checkbox to select it;
- b. save the configuration by touching **Save** key and exit from **Mounting** screen page (**Close** key).

5.20.2.6 Portal

This environment allows setting needed parameters to configure:

- **Gantry with 2 linear axes** - in this screen page the user has to specify, for each axis
 - the axis type (X, Y and Z),
 - positive and negative stroke-ends
 - calibration position for each axis.
 Furthermore the following information is also to be specified:
 - mounting angle of the robot on the gantry
 - offset.
- **Gantry with 3 linear axes** - like the previous one, but referred to three axes
- **Trans-rotational column** - in this screen page (see an example in the below figure), like the previous ones, the user has to specify positive and negative stroke-ends and the calibration position.

The following parameters have to be specified too:

 - **ROBOT. MOUNT. ANG.** - robot mounting angle on column overhanging beam;
 - **ROBOT. MOUNT. POS.** - robot mounting position on column (low/high);
 - **COLUMN HEIG. [mm]** - column height;
 - **RADIUS [mm]** - radius (from column centre to robot base centre);
 - **ANGLE** - column mounting angle on slide [$^{\circ}$];
 - **OFFSET [mm]** - column mounting offset on slide.

5.20.2.7 Positioners

This environment, for a Positioner seen by the system as an **AXIS**, allows to

- [Creating a new configuration](#) or
- [Modifying an already existing configuration](#),

- Saving the configuration.



A different environment is available to handle Positioners seen by the system as **ARMs**. For further information refer to [par. 5.20.2.8 Posit_Arm on page 232](#).

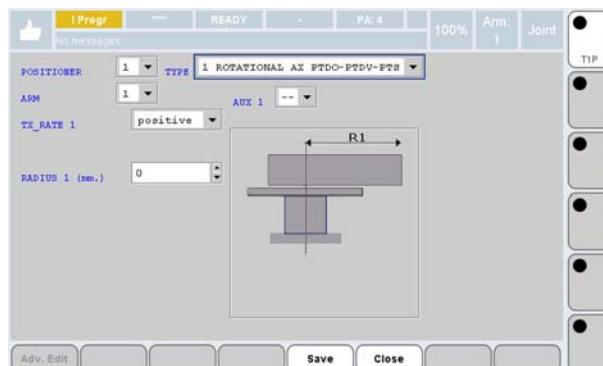
5.20.2.7.1 Creating a new configuration

In the main screen (see figure above) insert the following data:

- **POSITIONER** - the positioner index; C5G system can manage up to 4 positioners
- **TYPE** - the being configured positioner type. The choice is among:
 - ROTATIONAL AXIS
 - PERPENDICULAR AXES
 - NON PERPENDICULAR AXES
 - "L" AXES.
- **ARM** - the arm associated to the being configured positioner.

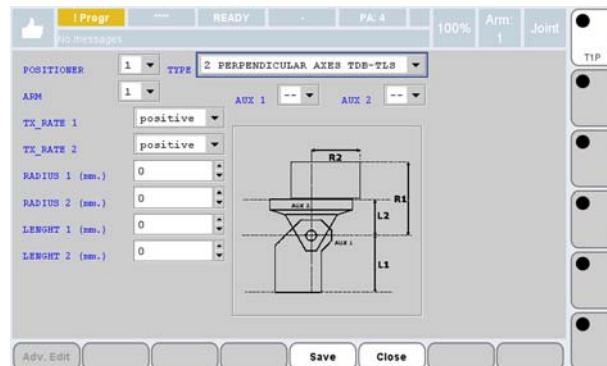
When all the needed data have been inserted and confirmed according to each field type modality, touch **Save** key to save the newly created configuration.

ROTATIONAL AXIS



On this screen page the user has to specify the maximum radius of the part overall dimensions [mm].

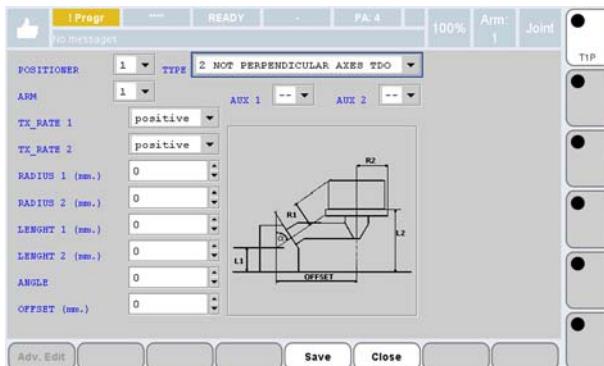
PERPENDICULAR AXES



The user has to enter the following data:

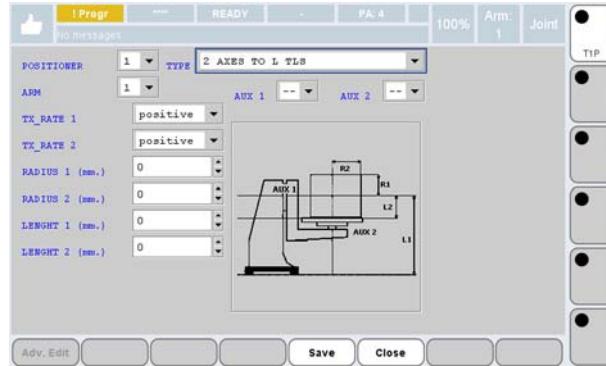
- radius R1 and radius R2 of the maximum overall dimensions of the being machined part [mm]
- 1st axis length (L1)
- 2nd axis offset (L2)

NON PERPENDICULAR AXES



- radius R1 and radius R2 of the maximum overall dimensions of the being machined part [mm]
- 1st axis length (L1) [mm]
- 2nd axis length (L2) [mm]
- angle (alpha) between rotation axis of 1st axis and the vertical referred to the positioner base [°]
- 2nd axis offset (L2) [mm]

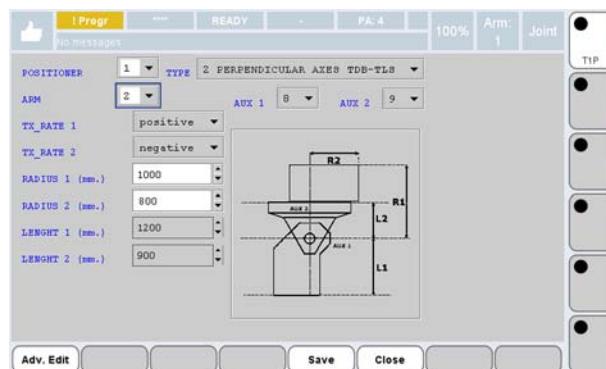
“L” AXES



- radius R1 and radius R2 of the maximum overall dimensions of the being machined part [mm]
- 1st axis length (L1) [mm]
- 2nd axis offset (L2) [mm]
- **ARM** - the arm which the being configured axis (axes) is associated to.

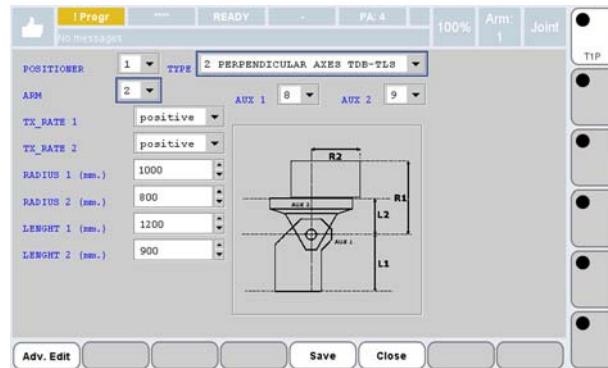
5.20.2.7.2 Modifying an already existing configuration

When the positioner has already been previously configured, the associated data are displayed by the program, giving the User the opportunity of modifying just some data related to the radius values (the other fields are read-only as shown in the below figure): this allows to adjust the maximum overall dimension in order to assure the safety speed while programming.



If the user wishes to modify other already set values instead, **Adv.Edit** is available allowing to modify all the fields of the page. The system prompts the user for confirmation before proceeding.

If the advanced modification is confirmed by the user (**Ack** key), all the fields become editable (e.g. axes lengths, auxiliary axes indices, etc).



5.20.2.7.3 Saving the configuration

At the end, touch **Save** key to save all the inserted configuration data.

A suitable message informs the user that the operation has been successfully completed.

5.20.2.8 Posit_Arm

This environment allows creating/modifying the configuration of a Positioner seen as an **ARM** by the system.

Positioners environment descriptions and configurations are still valid from the user interface stand point. The system itself will take care of handling the inserted data, as belonging to an **ARM**.

5.20.2.9 Rail

To configure the integrated slide, the displayed fields parameters have to be inserted (see figure below).



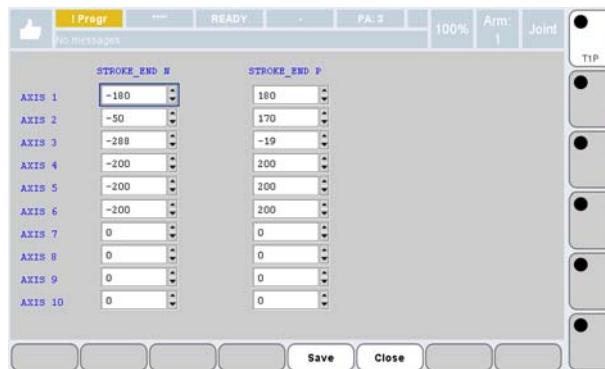
- **Height** - the distance (mm) between the floor and the robot base (installed on the slide)
- **Calib.position** - slide axis calibration position (mm)
- **Stroke-end Pos [mm] / Stroke-end Neg [mm]** - positive stroke-end positivo and negative stroke-end (mm)
- **Angle** - mounting angle of the robot on the slide, measured anticlockwise from the slide axis to the X_{base} robot axis.

- **TX_RATE value** - transmission rate value
- **Integrated** - if selected, it indicates that the slide is handled in integrated mode
- **TX_RATE Sign** - transmission rate sign (Positive / Negative)

The user has to enter the required values and touch **OK** to confirm. To cancel the modifications, touch **Close** or press **ESC**.

5.20.2.10 StrokeEnd

By selecting the **StrokeEnd** icon, the software stroke ends of all axes of the current arm (i.e. the currently set one in the [Motion Page, Basic](#) sub-page) can be changed.



5.20.3 System



This environment allows setting values for the following subenvironments:

- [Network](#)
- [Autologout](#)
- [Backup-Restore](#)
- [Configure](#)
- [Customer](#)
- [Date-Time](#)
- [ExeHelp](#)
- [Install](#)
- [Login](#)
- [Reload-Sw](#)

- [Startup.](#)

5.20.3.1 Network



The following network settings environments are available from within this folder:

- [Ethernet](#)
- [Ethernet Annex](#)
- [UsbToEthernet.](#)

5.20.3.1.1 Ethernet



It handles active network settings on the Control Unit. The following data are displayed:

- network address (**IP address**) of the APC board
- mask for the local network (**Subnet Mask**), referred to the APC board
- **Gateway** on the APC
- **MAC address** of the existing Ethernet ports in the system.

The user is allowed to modify only the three first fields. The other field is read-only.

To make any changes effective, save the configuration file: enter the [Configure](#) environment (in the [SETUP-System](#) page) and touch **Save** key.

5.20.3.1.2 Ethernet Annex

This is an environment in which the user is allowed to configure an Ethernet extra module (Ethernet Annex).



The displayed data are as follows:

- network address (**IP address**) referred to the APC board
- mask for the local network (**Subnet Mask**), referred to the APC board
- **MAC address** of the existing Ethernet ports in the system.

The user is allowed to modify the first two fields only. The third one is read-only.

When all the configuration data have been inserted, touch either **OK** to confirm or **Cancel** to exit without any modifications.

To make the new configuration data operational, save the configuration file: enter the [Configure](#) environment (in the [SETUP-System](#) page) and touch **Save** key.

5.20.3.1.3 UsbToEthernet

This is an environment to configure the USB cable to/from Ethernet.

After selecting **Enable USB to Ethernet** checkbox, the system displays the default values, useful to the user as a help to the being inserted data format:

- **IP address** of the C5G Controller,
- mask for the local network (**Subnet Mask**)
- **IP address of the Remote Host**.

Insert the wished values.

At the end of the configuration data insertion, touch either **OK** to confirm or **Cancel** to finish without any modifications.



To make any changes effective, save the configuration file: enter the [Configure](#) environment and touch **Save** key.

5.20.3.2 Autologout

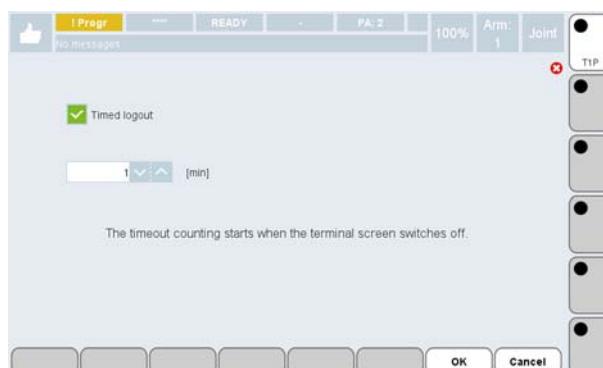
This is the subpage to configure the “Automatic Logout” functionality.



For further information about such a functionality, see [par. 4.3.1 Automatic Timed Logout on page 48](#).

The user is allowed to modify the following fields:

- **Timed Logout** - enable/disable the automatic logout functionality
- **Timeout** - it is the time in minutes which lasts between screen off and Automatic Logout execution.



BE CAREFUL! when a [Startup Login](#) has been defined in the system, this will be the user level the Automatic Logout switches to: this means, in fact, that in case of Automatic Logout the system performs a logout of the current user and immediately logs in the Startup user (see [par. 5.20.3.9.2 Startup on page 251](#)).

If the Automatic Logout is just needed to avoid access to the current user, DO NOT set any Startup user!

5.20.3.3 Backup-Restore



This environment is used to set the SaveSets and the Device for the Backup / Restore operations.

IT DOES NOT EXECUTE THESE OPERATIONS: to do so, after having defined the Saveset, it is necessary to access [Files Page](#), command [Disk](#), [Backup](#), [Backup of Saveset](#) or [Restore](#), [Restore of Saveset](#).

See also [HINTS FOR BACKUP AND RESTORE USE](#).

If the user wishes to create a new SaveSet or to modify an already existing one, the procedure is basically as follows:

- a. touch a row to select it (either associated to an already existing Saveset to modify it, or empty to create a new one)
- b. touch **Edit** key

- c. insert the wished changes or create the new Saveset. The needed tools are automatically made available by the system
- d. touch **OK** key or press **ENTER** key to confirm, or cancel by pressing **ESC** key.

The following setup commands are provided for these functions:

- [Backup](#)
- [Restore](#)
- [Device](#)

5.20.3.3.1 Backup

Displays, allows to create and to modify the Savesets to be used to carry out the Backup operation (as shown in the figure below).



Every Saveset holds information about device, files and sub-directories to be copied towards the default backup/restore device (which is displayed in [Device](#) sub-page), as well as all options to be used for such a Saveset.



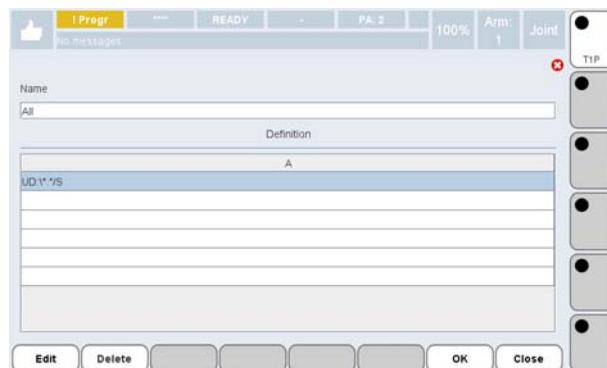
Among the available Savesets, there is a predefined one, called [Srv](#), which is to be used to save a specific situation before calling COMAU Assistance.
It is recommended not to activate the backup operation with such a Saveset, during the process: it could slow it down.

The user can define some new Savesets:

- The displayed value in the first column on the left, is the index of the predefined variable \$BACKUP_SET[] associated to such a Saveset.
- The string displayed in the **NAME** column specifies the name of the Saveset that can be used by the system during a backup operation (carried out from the [Files Page](#), [Disk](#) subpage - [Backup](#) or [Backup of Saveset](#), when the Saveset option is selected).
- **DEFINITION** is the content of this Saveset. Characters following the slash ('/') represent the available options associated to such a Saveset.

To operate on Savesets, act as follows:

- a. touch the wished line, to select it



- b. touch **Edit** to open the next subpage and modify the included settings.
- c. Touch **Name** field if wished to be modified
- d. touch **OK** to confirm.
- e. Select the Saveset to modify its **Definition** if wished to
- f. touch **Edit** to open next page which allows specifying: source device, subdirectories, names of the files to be backed-up and options associated to the Saveset.



- g. Insert the wished modifications into the **File** field (by means of the [Alphanumeric keyboard](#) opened by the system) and then touch **OK** to confirm them.
- h. To add one or more options, touch the corresponding checkbox. If a checkbox is selected, the corresponding option is added to the Saveset definition.
Note that if the chosen option is either **Date** or **Exclude files**, one more field is open allowing the user to insert the corresponding string. The available options are as follows:
 - **Subdirectories (/S)** - the Saveset also includes all subdirectories of the specified directory
 - **Incremental (/I)** - the Saveset only includes files which have never been saved before (without archived attribute, 'a')
 - **Unset archived marker (/U)** - the archived attribute ('a') will not be set for the files belonging to this Saveset
 - **Date (/A)** - the files to be included to this Saveset depends on their date:
 - **-n** = all files of **n** days before today
 - **-gg-mm-aa** = all files before the specified date
 - **gg-mm-aa** = all files after the specified date.

- **Exclude Files (IX)** - the indicated files will not be included in this Saveset. Specify them separated by ‘;’ . Ex. *.aa;*.bkc
- i. To insert some more RULEs to be associated to this Saveset, a new line must be used of the main page, and steps from [a.](#) to [h.](#) must be performed.
- j. Touch **OK** to confirm, when all the modifications are complete.

The Saveset is now ready to be used from within the [Files Page](#) during the execution of a Backup operation.



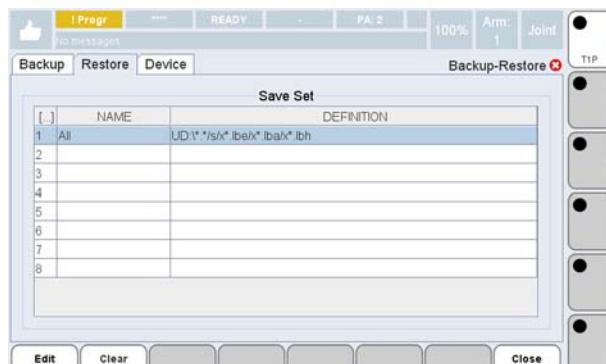
For further information, refer to [par. 6.9.4.8 Backup and Restore commands on page 329](#), in [Chap.6. - System Commands on page 295](#).

5.20.3.3.2 Restore

Displays and allows to modify and create Savesets to be used for Restore operations.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.



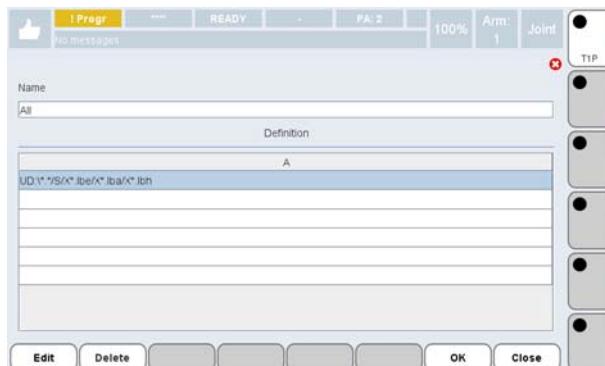
Every Saveset holds information concerning device, files and possible sub-directories to be restored from the default backup/restore device (displayed in the [Device](#) subpage), as well as all options to be used for such a Saveset.

The user can define new Savesets:

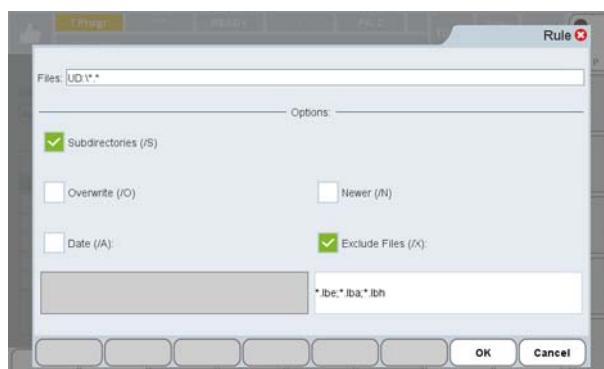
- the value displayed in the first left column is the index of \$RESTORE_SET[] predefined variable, for such a Saveset.
- The displayed string in **NAME** column indicates the Saveset name that can be used by the system during restore operations (performed from within the [Files Page](#), [Disk](#) subpage - [Restore](#) or [Restore of Saveset](#), when the Saveset option is selected).
- **DEFINITION** indicates the content of this Saveset. Characters following the slash ('/') represent the possible options associated to the Saveset.

To operate on Savesets, act as follows:

- a. touch the wished row to select it,



- b. touch **Edit** to access the next page and modify its settings.
- c. Touch **Name** field if it is wished to be modified
- d. touch **OK** to confirm any modification.
- e. If wished, touch the Saveset to select it and then edit the Se lo si desidera, selezionare il Saveset per modificarne la **Definizione**
- f. touch **Edit** to open the next page allowing to specify: source device, subdirectories, names of the files to be restored and options associated to the Saveset.



- g. Type in the modifications in the **File** field and then touch **OK** to confirm them.
- h. To add one or more options, touch the corresponding checkbox. If a checkbox is selected, the corresponding option is added to the Saveset definition.
Note that if the chosen option is either **Date** or **Exclude files**, one more field is open allowing the user to insert the corresponding string. The available options are as follows:
 - **Subdirectories (/S)** - the Saveset also includes all subdirectories of the specified directory
 - **Overwrite (/O)** - in the Restore operation also the read-only files, belonging to the Saveset, will be overwritten
 - **Newer (/N)** - in the Restore operation all files belonging to the Saveset, which are newer than the ones included in the .LST file, will be restored
 - **Date (/A)** - the files to be included to this Saveset depends on their date:
 - **-n** = all files created/modifed **n** days before today
 - **-gg-mm-aa** = all files created/modifed before the specified date
 - **gg-mm-aa** = all files created/modifed after the specified date.
 - **Exclude Files (/X)** - the indicated files will not be included in this Saveset. Specify them separated by ';' . Ex. *.aa;*.bkc

- i. To insert some more **RULEs** to be associated to this Saveset, a new line must be used in the main page, and steps from [a.](#) to [h.](#) must be performed.
- j. Touch **OK** to confirm, when all the modifications are complete.

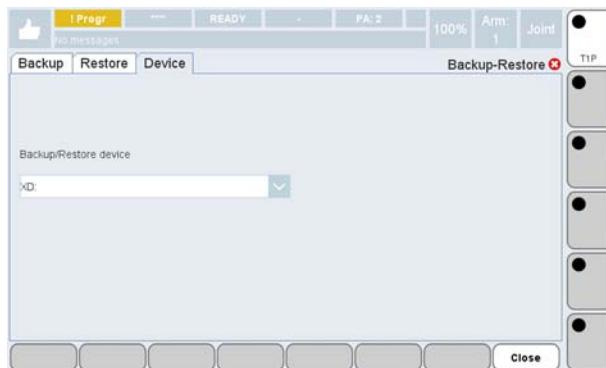
The Saveset is now ready to be used from within the [Files Page](#) during the execution of a Restore operation.



For further information, refer to [par. 6.9.4.8 Backup and Restore commands on page 329](#), in [Chap.6. - System Commands on page 295](#).

5.20.3.3.3 Device

This subpage allows to setup the default device for backup/restore operations. It is the default destination device for the [Backup](#) command and the default source device for the [Restore](#) command.



This device represents the content of \$DFT_DV[3] predefined variable.

To modify the default device, act as follows:

- a. touch **Backup/Restore Device** field; the system opens the available devices list



- b. touch the wished device name, to select it
- c. touch **OK** to confirm.

5.20.3.4 Configure



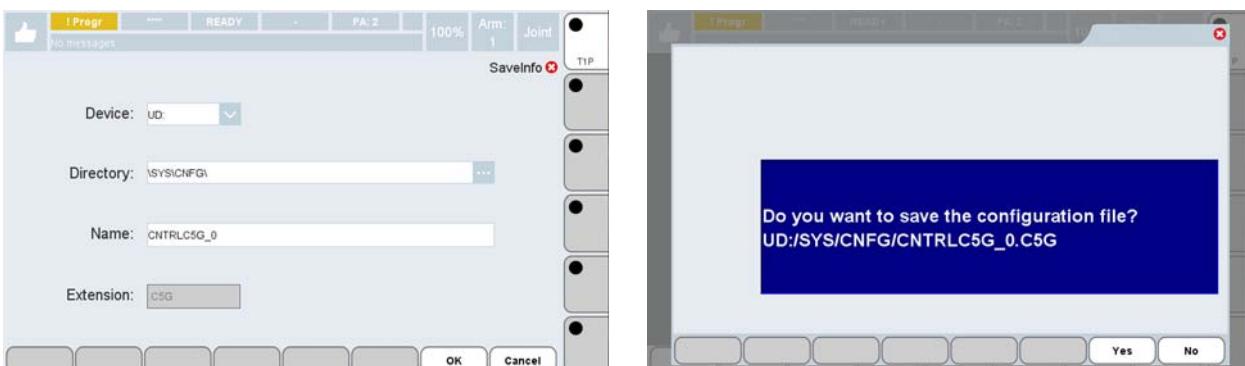
In this subenvironment the user can activate some functions related to the configuration files, by means of the following keys existing in the [Functional keys menu](#) (see figure above):

- **Save**

allows saving modifications, related to the configuration predefined variables values and User data base values; creates a new **<\$SYS_ID>.C5G** configuration file and a new **<\$SYS_ID>.UDB** user data base file.

Note that it is allowed to modify the related data (below figure, on the left), i.e. **Device**, **Directory** and **Name** of the file, before going on with Saving operation.

The system prompts the user for confirmation. Answer **Yes** to start the operation execution (see below figure, on the right).



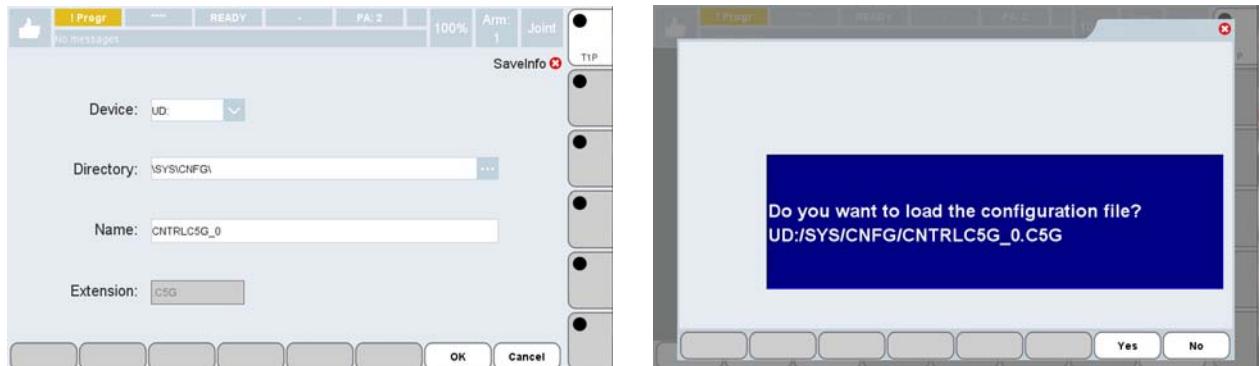
For further information about this functionality, refer to [System Commands](#) chapter, [CONFIGURE SAVE ALL \(CSA\)](#) paragraph.

- **Load**

allows loading all values contained in **<\$SYS_ID>.C5G** and **<\$SYS_ID>.UDB** files to the corresponding predefined variables.

Note that, like for **Save** command, it is allowed to modify the related data (below figure, on the left), i.e. **Device**, **Directory** and **Name** of the file, before going on with Loading operation.

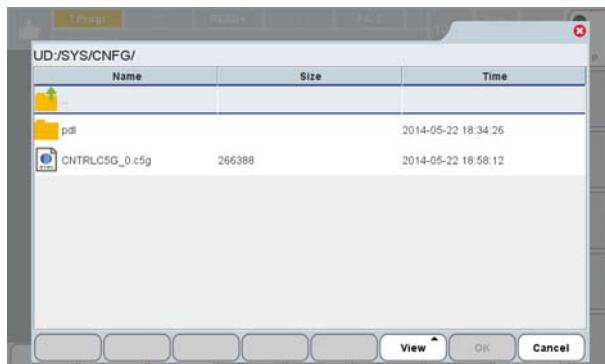
The system prompts the user for confirmation. Answer **Yes** to start the operation execution (see below figure, on the right).



For further information about this functionality, refer to [System Commands](#) chapter, **CONFIGURE LOAD ALL (CLA)** paragraph.

– **Browse..**

allows specifying the wished device and folder, for **<\$SYS_ID>.C5G** and **<\$SYS_ID>.UDB** configuration files.



The **View** key allows choosing the view modality for files and folders (either **Big icons** or **Details**).

At the end (in the shown above example, **CNTRLC5G_0.c5g** file has been selected) touch **OK** key to confirm.

5.20.3.5 Customer



This environment allows the user to initialize/modify some fields of the [Start Page](#):

- **CUSTOM_CNTRL_ID**, is the Controller name assigned by the user (corresponding to the content of \$CUSTOM_CNTRL_ID predefined variable).
Touch the field to open the [Alphanumeric keyboard](#). Insert the wished data and touch **OK** to confirm.
- **CUSTOM_ID**, are the names assigned by the user of the possible 4 Arms (corresponding to the content of the \$CUSTOM_ID predefined variable 4 elements)
Touch each being modified-initialized field, to open the [Alphanumeric keyboard](#). Insert the wished data and touch **OK** to confirm it.

Finally, touch **OK** to exit from Customer sub-page saving modifications. Touch **Cancel** to exit without saving anything.

5.20.3.6 Date-Time

This icon allows the user to access both date and time and daylight saving time setting environment.

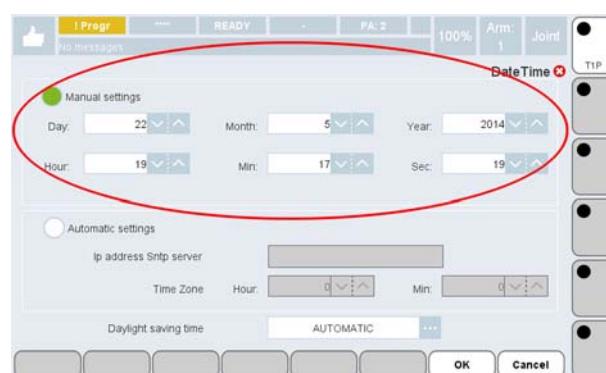
The user can operate on the wished data, by selecting the corresponding field:

- [Manual settings](#)
- [Automatic settings](#)
- [Daylight saving time](#).

Manual settings

The User can edit **Day**, **Month** and **Year** fields for the current **DATE**, and **Hour**, **Min** and **Sec** for the current **TIME** (as shown in the below figure).

Touch the wished fields, use the automatically opened keyboard and touch **OK** key each time, to confirm.

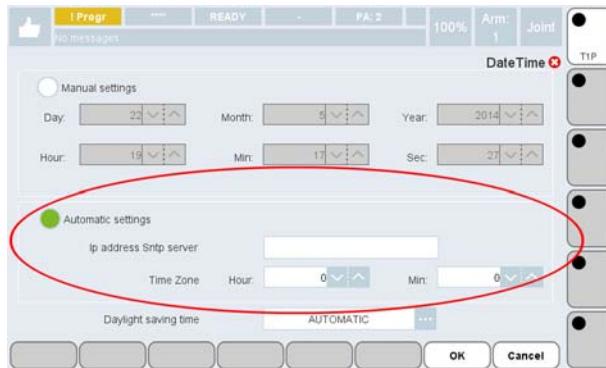


Automatic settings

The user is allowed to synchronize the system clock by means of the **SNTP** (Simple Network Time Protocol) protocol.

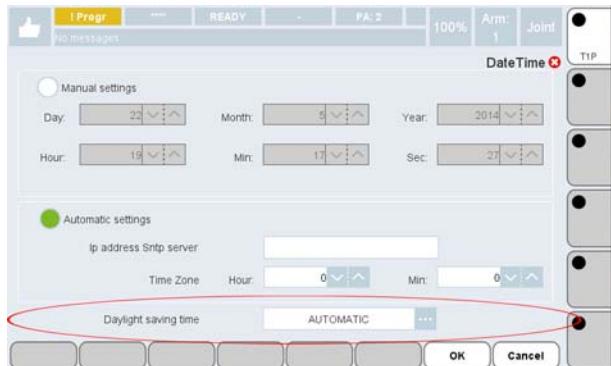
To do so, the following fields must be configured:

- **IP address SNTP server** to synchronize the required setting,
- wished **TIME ZONE**, with **Hour** and **Min** fields. The allowed range for such fields is:
 - **Hour** field -12 to +13
 - **Min** field 0 to 59.



The typical use of such a modality, is timing synchronization of several Controllers on the same network.

Daylight saving time

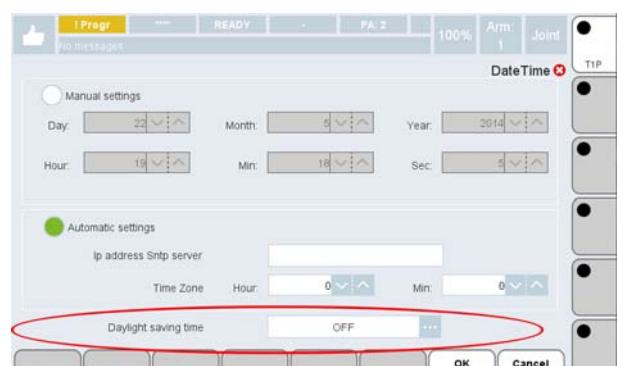
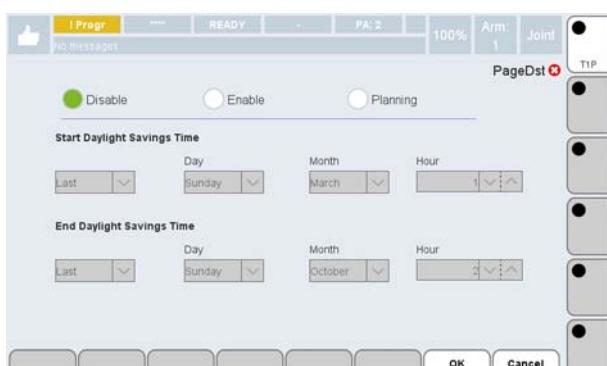


Regardless of previous settings (either manual or automatic), the user can access the Daylight Saving Time settings by pressing the corresponding **Combobutton (Daylight saving time field in the above figure)**.

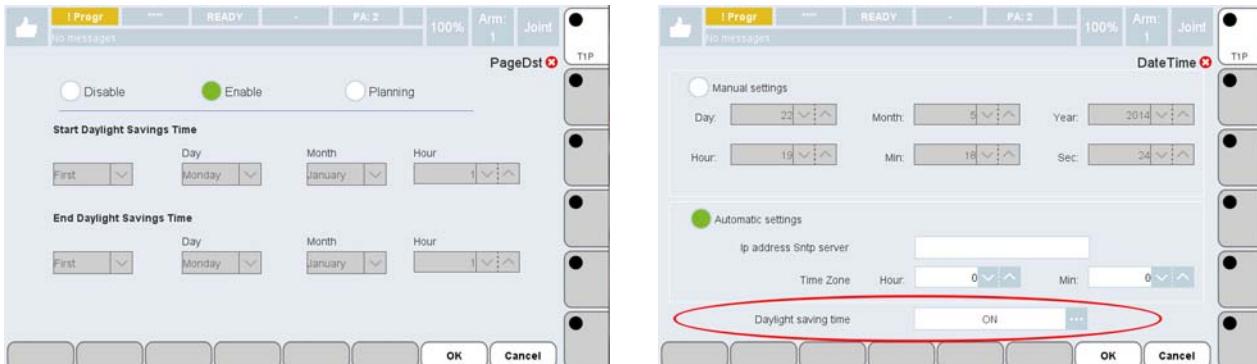
The system displays a page for the daylight saving time settings.

The available functionalities are as follows:

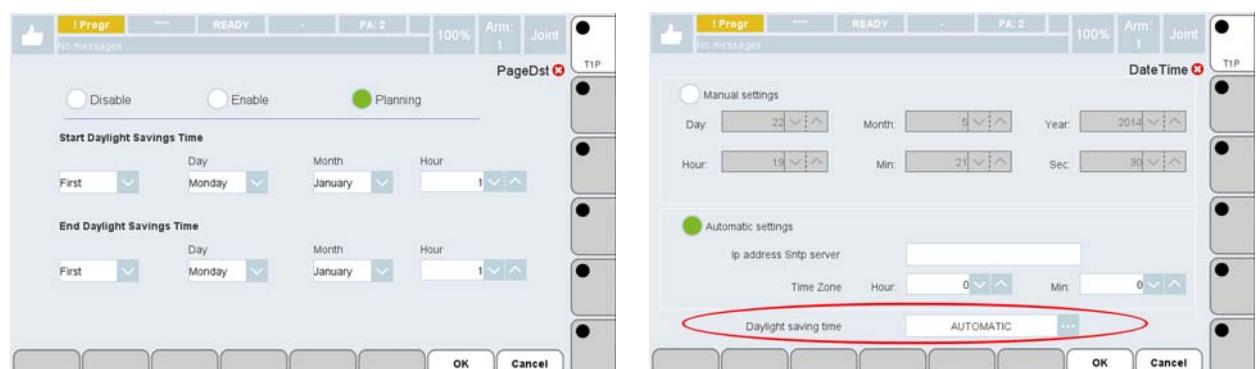
- **Disable** - when selected, the system does not take into account using the Daylight Saving Time. Such a choice causes the Daylight Saving Time field in the Date-Time main page, to be set to **OFF** (next figure, on the right).



- **Enable** - when selected (below figure, on the left), the system switches to the Daylight Saving Time immediately. Such a choice causes the Daylight Saving Time field in the Date-Time main page, to be set to **ON** (below figure, on the right).



- **Planning** - when selected (below figure, on the left), the user is allowed to schedule the **beginning** and the **end** of the Daylight saving time, according to the following criteria:
 - **week of the month** - first..fourth, or simply the last of the month.
 - **day of the week** - sunday..saturday
 - **month of the year** - january..december
 - **time of the day** - 0..23.



For each being inserted value, touch the corresponding field to select it: the system opens either the suitable list or keyboard. After inserting the wished value, touch **OK** to confirm.

Such an information is to be provided both for the Daylight saving time **beginning** and **end**.

Finally, touch either **OK** to confirm or **Cancel** to exit without modifying.

Such a choice causes the Daylight Saving Time field in the Date-Time main page, to be set to **AUTOMATIC** (above figure, on the right).

At the end of the settings, from within the main screen of Date-Time environment, touch either **OK** to confirm or **Cancel** to exit without modifying.

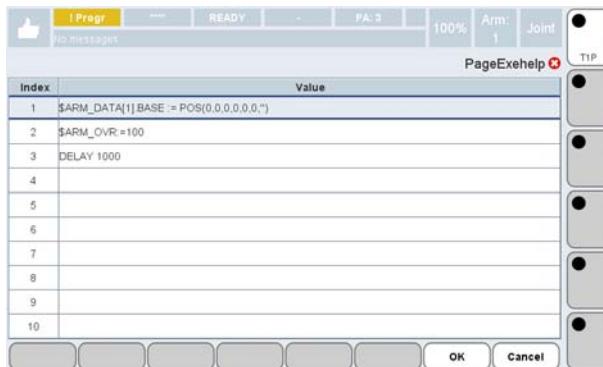


In case of either the SNTP IP and/or the Daylight Saving Time have been modified, the user is prompted to save the configuration file (see the [Save](#) command description)!

5.20.3.7 ExeHelp

This environment allows inserting, in a suitable predefined array (\$EXE_HELP[]), the most used components in the user's PDL2 programs: variable names, constants, routines, statements, etc.

The main page shows the currently inserted elements.



To add a new element, touch twice an empty row: the system automatically opens the [PDL2 keyboard](#).



Insert the wished string according to such a keyboard modalities; touch **OK** to confirm.

Once they are inserted in this environment, all the elements are available in the [History](#) which is the list associated to the [PDL2 keyboard](#) (next figure, on the right).

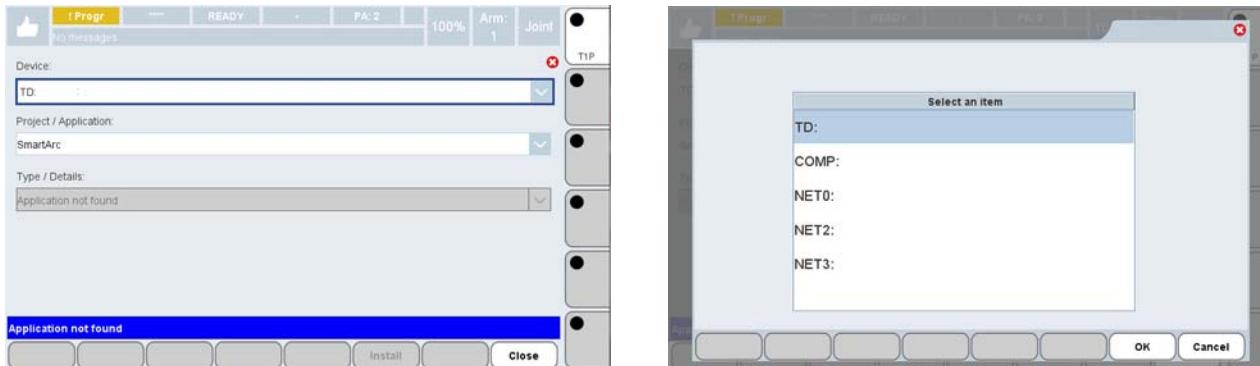
To view it, touch the red highlighted symbol in the following left figure.



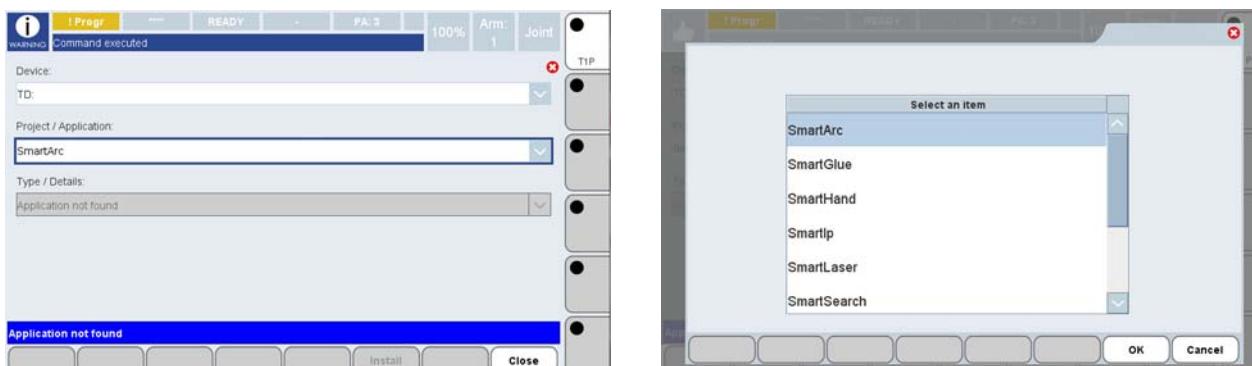
5.20.3.8 Install

Allows to install the specified application. The user has to:

- a. select the device from which the files of the being installed software must be read, i.e. touch **Device** field
- b. touch the wished device name, to select it
- c. touch **OK** key to confirm.



- d. Touch **Project/Application** field (the system automatically opens a list of the available applications)



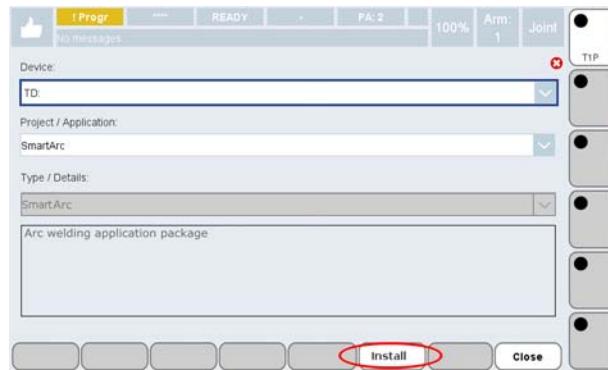
- e. choose the wished **Project/Application**, touching its name in the list.



Note that the available **Projects/Applications**, listed in the current menu, correspond to the existing **.APPL** files in the selected device (step b.) root.

The user/integrator can customize his/her **Projects/Applications** menu (according to what's described in [par. 5.22.5 Adding Projects/Applications, installable from within SETUP Page on page 293](#)).

- f. Touch **OK** key to confirm.
- g. Depending on the chosen **Project/Application**, select the existing either the multi-application or the standard application sections.
- h. If all the page settings are ok, touch **Install** key to install all the requested objects (as shown in the next figure). If they are not ok, on the contrary, such a key is not enabled and a suitable message warns the user about the situation.



5.20.3.9 Login

This environment provides information and allows modifications, as far as concerns the System access (Login).

The displayed information is:

- **Username**
- **Startup** - flag indicating whether this login is used or not upon the system restart.
- **Profile** - associated user profile to this username.



Username	Startup	Profile
ADMIN	<input type="radio"/>	Admin
PU	<input type="radio"/>	Programmer
MU	<input type="radio"/>	Maintenance
GA	<input checked="" type="radio"/>	Admin, Programmer, Maintenance, Service

The following commands are available in the [Functional keys menu](#):

- [Users](#)
- [Startup](#).

5.20.3.9.1 Users

Allows adding/modifying/deleting the configured users. The Administrator profile login is required, to be allowed executing these commands.



The available commands are as follows:

- **Add**

Adds a User (Login).

The operator is asked to specify **Username** and **Password**. The user **Privileges** are to be specified too. In case the user does not give the requested information, the default privileges are used (**Programmer** profile).



While defining the Technology profile, a positive numeric value must be specified, otherwise such a profile will not be taken into consideration.

- **Delete**

Cancels the selected user from the system database.

Select the being deleted user, by touching the corresponding row, then touch **Delete** command.

- **Modify**



Allows modifying the selected user's access rights: user profile and, in case of Technology, profile type (either **1** or **2**). It is enough to edit the wished fields and then confirm by touching **OK**.

5.20.3.9.2 Startup

Enables/disables the selected login to be used upon the system restart.

Touch the wished profile row, to select it.

Touch **Startup** key; a page is displayed in which the user is asked to insert the selected profile password.

Save the system configuration file (by means of **Save** command in **Configure** environment) to make the modifications operational.

Note that, obviously, only one **Startup Login** at a time is allowed.



5.20.3.10 Reload-Sw

The current sub-page allows executing all operations related to load/update the System Software.



PAY ATENTION!

Before selecting the Reload-Software icon, insert, in the Controller USB port, a disk-on-key containing the image of the System Software released by Comau, including the subdirectories. The main directory of the software version has a name like the following:

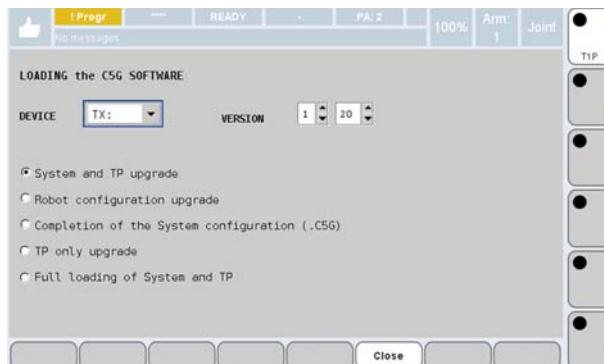
C5G_<V>_<v>

where 'V' and 'v' stand for the full identifier of the being loaded version.

For example:

C5G_1_20

is the main directory where 1.20 Software version is contained.



As soon as the Reload-Software environment is entered, the page shown in the above figure is displayed.

Touch the **DEVICE** field to select the port where the USB Flash Disk containing the being loaded software has been inserted, from the displayed list.

In the two **VERSION** fields the full identifier of the current System Software version is displayed. In the shown above example the being loaded version is 1.20.

Please, note that the USB Flash Disk could include several different Software versions.

Then the user must choose the wished one, among the following procedures:

1. **System and TP upgrade** - it allows to **UPGRADE** both the System Software and the Teach Pendant Software
2. **Robot configuration upgrade** - it allows to **UPGRADE** the System data files without losing the current configuration, set by means of Comau official tools (e.g. stroke-ends, integrated slide data, etc.)
3. **Completion of the System configuration (.C5G)** - it allows to **INTEGRATE** further axes and/or arms to the current configuration
4. **TP only upgrade** - it allows to **UPGRADE**, if needed, the Teach Pendant Software and the User Interface Software
5. **Full loading of System and TP** - it allows to perform **COMPLETE LOADING** of the Controller System Software, the Teach Pendant Software and the robot configuration.

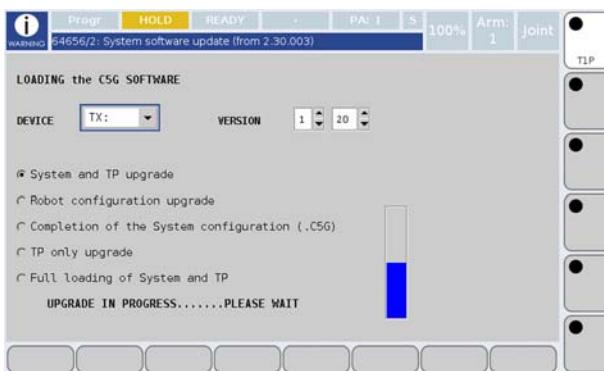
Tab. 5.1 - System configuration recovery (variables and I/Os) according to the software reloading configurations

	OPERATION	CONFIGURATION	DESCRIPTION
1, 3, 4	Upgrading software	System Variables	<u>The current configuration is held.</u> The existing .C5G file is used; it is automatically updated at restart time, according to the new software requirements.
		I/O Configuration	<u>The current configuration is held.</u> The existing .CIO file is used; it is automatically updated at restart time, according to the new software requirements.

Tab. 5.1 - System configuration recovery (variables and I/Os) according to the software reloading configurations (Continued)

	OPERATION	CONFIGURATION	DESCRIPTION
2	Upgrading configuration file	System Variables	<u>The configuration is recovered.</u> A new .C5G file is generated. The .PDL files, currently stored in UD:\SYS\CFG\PDL (generated by means of Comau official PDL2 tools) are automatically run to recover the previous configurations.
		I/O Configuration	<u>The current configuration is held.</u> The content of the existing .CIO file is held.
5	Full loading	System Variables	<u>The configuration can be recovered.</u> A new .C5G file is generated. The .PDL files, currently stored in UD:\SYS\CFG\PDL (generated by means of Comau official PDL2 tools) can be manually used to recover the configurations existing before loading.
5	Full loading	I/O Configuration	<u>The configuration can be recovered.</u> If the .CIO file is already existing, it is automatically used when the same \$SYS_ID containing its name is assigned during the loading procedure; otherwise it is newly generated. The .PDL files corresponding to the existing I/Os, currently stored in UD:\SYS\CFG\PDL (generated by means of Comau official PDL2 tools) can be manually used to recover the configurations existing before loading.

5.20.3.10.1 System and TP upgrade



- After the procedure has been started, a subpage is displayed to inform that upgrading is going on. A vertical progress bar, on the screen right side, shows the working progress for the required operation.
- When needed, the Teach Pendant loads the BSP software on the flash memory: if executed, this operation takes about 10 minutes.



NOTE ABOUT BSP SOFTWARE MANUAL LOADING

Upon BSP loading, coming from versions lower than 2.200.007, an anomalous situation might occur in which the following message is displayed:

"In order to recover this device, all required files must be copied manually".

In such a situation, follow the described below procedure:

- 1 insert USB flash disk into the hub connected to the TP USB port
 - 2 click left on START (left low corner of the screen), by means of the mouse (connected to the hub); then Programs and Windows Explorer
 - 3 Select TX (which is the USB flash disk inserted into the TP)
 - 4 Copy TP.jar and TPSM.exe files; they are included within TP.zip file which is supplied together with the Software version
 - 5 Select \FlashDisk\Comau folder
 - 6 Paste the copied files
 - 7 Restart TP5 by means of Keys to restart the Teach Pendant.
- c. Then the User Interface is loaded onto the flash memory.
- d. Finally, a message is displayed stating that the procedure is accomplished.
- e. A System restart is automatically performed. The procedure is completed.

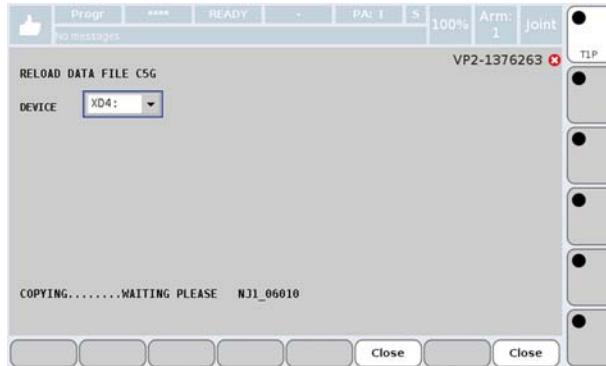
5.20.3.10.2 Robot configuration upgrade

This procedure allows to upgrade the configuration files for each already configured object (robot, axes, etc.), loading them from the USB flash disk which is currently inserted in the specified device.

- a. The procedure runs **SETUP 1** program whith the following start page:



- b. Select the wished language for executing the procedure, by touching the language name.
- c. Touch **DEVICE** field in the next screen page
- d. choose the wished device by touching its name in the list, then touch **OK** key to confirm.
- e. The program then updates the configuration files.

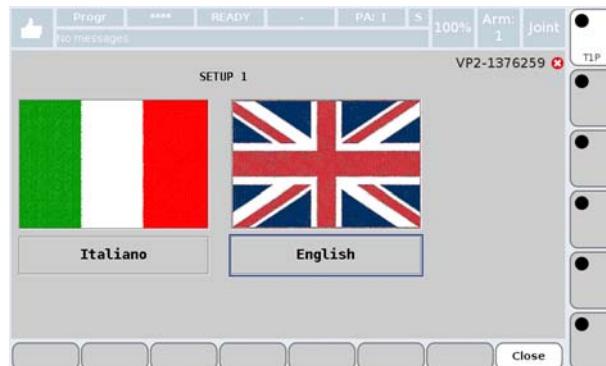


- f. The user will be prompted for touching **OK** to confirm each file.
- g. Touch one of the following keys to complete the procedure:
 - **Save** - to save the inserted data. At the end of the saving operation, touch **Restart** key.
 - **Restart** - to exit from the procedure. The System is restarted. If the inserted data have NOT been saved, e.g. for any wrong operations, an informational message warns the User about that. Provide for saving them and touch **Restart** key again.

5.20.3.10.3 Completion of the System configuration (.C5G)

This procedure allows to complete the already existing configuration, adding further auxiliary axes and/or arms.

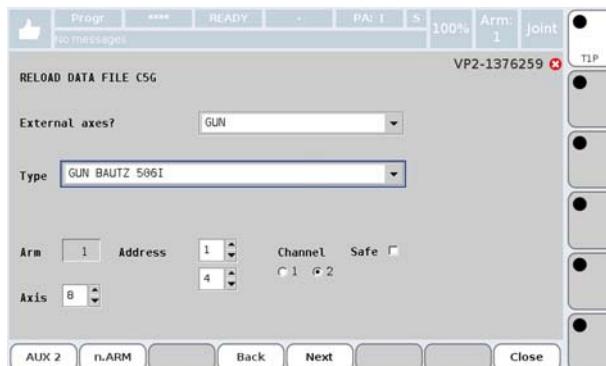
- a. **SETUP 1** program is automatically run



- b. Select the wished language for executing the procedure, by touching the language name.
- c. the program loads the required data and displays a screen page in which the User is requested to select or confirm the device and touch **Next** key to continue.
- d. If an auxiliary axis is to be added, touch the corresponding field
- e. select it by touching its name, from the list opened by the system
- f. touch **OK** key to confirm.
- g. The **Type** field is then displayed in which the user should specify the wished auxiliary axis type.

h. Select it by touching its name, from the list opened by the system

i. touch **OK** key to confirm.



j. Then the program displays some fields related to the chosen auxiliary axis, in which the User must insert the required values:

- **Address** - motor drives address - usually, the number in the upper field is equal to the number of the arm which the axis belongs to; it is the value which is preset on T2 rotary switch (see Fig. 5.58).
The lower field includes a sequential value which is the first available value, starting from 4: 1, 2 and 3 values are reserved to the Robot axes; such a value is the one preset on T1 rotary switch (see Fig. 5.58).
- **Axis** - logical index of the being added auxiliary axis



For further information, please refer to par. [POWERLINK Ethernet](#), in [C5G Controller Unit - Technical Specification](#) manual.

- **Channel** - section of the motor drives module; it is needed to specify such an information if the module is double.



Note that when the module is double and the first section has already been used, the User must confirm the same module and select channel no.2.

- **Safe** - enables/disables using **Safe Motion** functionality for the involved axis.

k. After the above data insertion, add further auxiliary axes and/or arms, by means of **AUX2** and/or **n.Arm** keys.

l. To accomplish the data insertion, touch:

- **Next** key to confirm them
- **Back** key to modify them.

m. The System is automatically set to Minimal Configuration. Going on with the procedure, a screen page is displayed in which the User must specify some settings for the Controller:

- **Serial Number C5G** - it is the Controller serial number together with indication of the referred Country
- **Language** - it is the default language for the Controller
- **Move Programming** - set to MODAL by default

- **Safety stop time** - in case of PDTV/O (/ORB), the user is requested to specify the value, in seconds, for setting the safety stop time.



See also par. [Safety Stop: setting the stop circuit timer](#), in the [C5G Control Unit - Transport and Installation](#), manual, for further information.

- To complete the procedure, touch one of the following keys:
 - **Save** - to save the inserted data. At the saving end, touch **Restart** key.
 - **Restart** - to exit from the procedure. The System is restarted. If the inserted data have NOT been saved, e.g. for any wrong operations, an informational message warns the User about that. Provide for saving them and touch **Restart** key again.

5.20.3.10.4 TP only upgrade

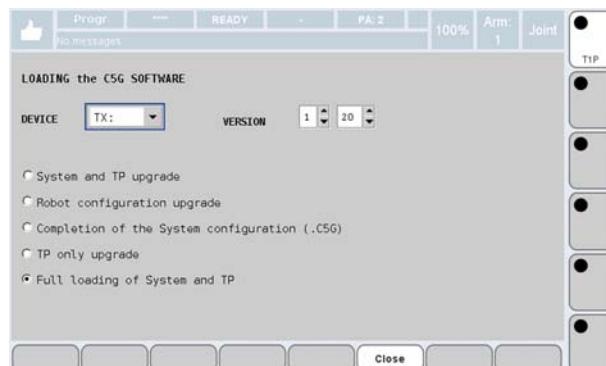
This procedure allows to upgrade the Teach Pendant Software only, without modifying in any case the Controller Software.

The procedure steps are the same of the [System and TP upgrade](#) procedure, as far as concerns the TeachPendant section: from step **b.** to the procedure end.

5.20.3.10.5 Full loading of System and TP

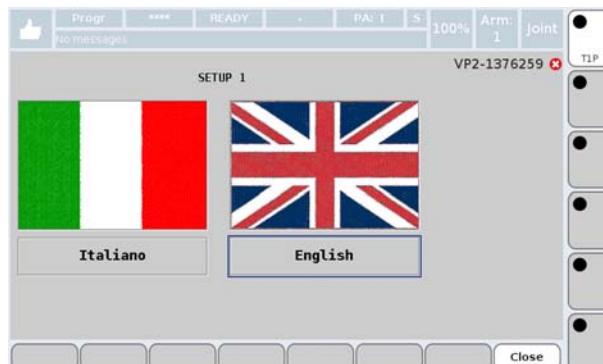
This procedure allows to load all the Controller System Software, the Teach Pendant BSP Software, the TeachPendant User Interface Software and the robot configuration.

- After starting the procedure, the steps are the same of [System and TP upgrade](#) procedure.

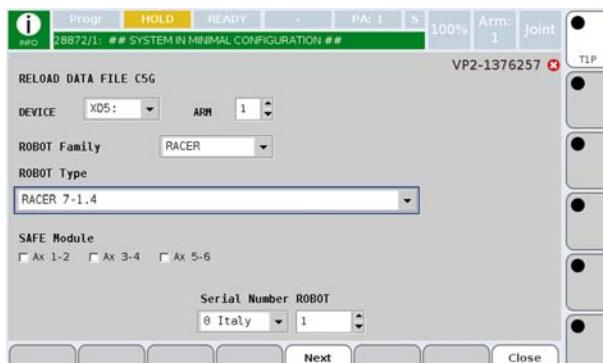


- The System is switched to minimal configuration. At the end of data upgrading, a restart cold operation is automatically performed ([par. 5.7.2.2.1 Cold on page 91](#)).
- The procedure runs **SETUP 1** program:

Fig. 5.57 - SETUP 1 - Language selection



- d. Select the wished language, by touching its name
- e. the program loads the needed data and displays a screen page in which the user should either select or confirm the device and then touch **Next** to continue.
- f. Some screen pages are displayed in which the User is requested to insert information such as:
 - arm number (**ARM** field)
 - **ROBOT Family**
 - **ROBOT Type**
 - **Serial Number ROBOT**



- g. After the data insertion is completed, touch **Next** key to continue.
- h. The User is asked to confirm the current Arm data. Touch **OK** key to confirm; **SETUP 1** program is ready for further auxiliary axes adding, if required.
- i. If wished, add an auxiliary axis, selecting it from the menu which is opened as soon as the **External Axes** field is touched
- j. touch **OK** key to confirm.
- k. Select the axis **Type**, from the field which is then displayed
- l. touch **OK** key to confirm.
- m. The program displays some fields related to the chosen auxiliary axis, in which the User should insert the required values:



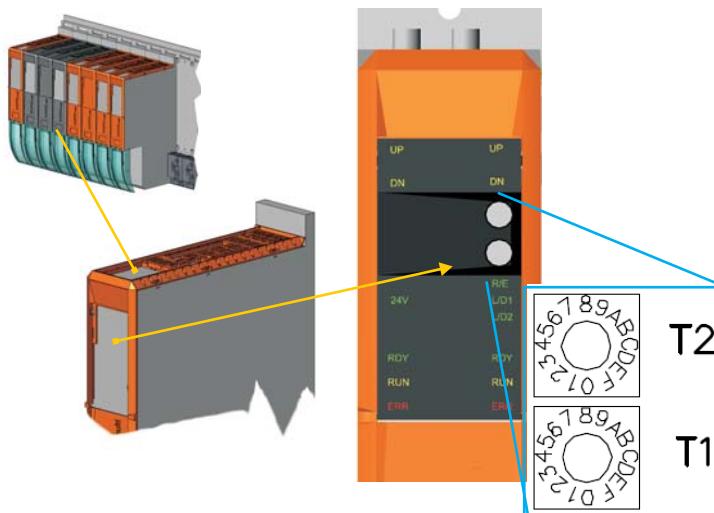
- **Axis** - logical index of the being added auxiliary axis
- **Address** - motor drives address - usually, the number in the upper field is equal to the number of the arm which the axis belongs to; it is the value which is preset on T2 rotary switch (see Fig. 5.58).

The lower field includes a sequential value which is the first available value, starting from 4: 1, 2 and 3 values are reserved to the Robot axes; such a value is the one preset on T1 rotary switch (see Fig. 5.58).



For further information, please refer to par. [Ethernet POWERLINK address](#), chap. [Details about the C5G Control Units](#), in [C5G Controller Unit - Technical Specification](#) manual.

Fig. 5.58 - Rotary switch for motor drives address setting



- **Channel** - section of the motor drives module; it is needed to specify such an information if the module is double.



Note that when the module is double and the first section has already been used, the User must confirm the same module and select channel no.2.

- **Safe** - enables/disables using the **Safe Motion** functionality for the involved axis.

- n. After the above data insertion, add further auxiliary axes and/or arms, by means of **AUX2** and/or **n.Arm** keys.
- o. After the above data insertion, add further auxiliary axes and/or arms, by means of **AUX2** key. If a new Arm is wished, touch **n.Arm** key. No matter of which is the choice (either adding axis, or adding arm, or going on with the procedure), the program displays the inserted values and waits for the User to confirm them.
- p. Touch **OK** key to confirm.
- q. Let's suppose the User has touched **n.Arm** key to add a new Arm. Select the wished device or confirm the current one by touching **OK**.
- r. Like described above for adding a new axis, some screen pages are displayed in which the User is requested to sequentially insert the needed information for the new Arm:
 - **ROBOT Family**
 - **ROBOT Type**
 - **Serial Number ROBOT**
 - **Axis** - logical index for the axis of the being added Arm
 - **Address** - motor drives address - usually, the number in the upper field is equal to the number of the arm which the axis belongs to; it is the value which is preset on T2 rotary switch (see [Fig. 5.58](#)).
The lower field includes a sequential value which is the first available value, starting from 4: 1, 2 and 3 values are reserved to the Robot axes; such a value is the one preset on T1 rotary switch (see [Fig. 5.58](#)).



For further information, please refer to par. [Ethernet POWERLINK address](#), chap. [Details about the C5G Control Units](#), in [C5G Controller Unit - Technical Specification](#) manual.

- **Channel** - section of the motor drives module; it is needed to specify such an information if the module is double.
- s. To accomplish the data insertion, touch:
 - **Next** key to confirm them
 - **Back** key to modify them.
- t. The User is asked to confirm the data both for Arm 1 and Arm 2.
- u. Touch **OK** key to confirm.
- v. Touch **Next** key again in the next screen page.
- w. A screen page is displayed in which the User must specify some settings for the Controller:
 - **Serial Number C5G** - it is the Controller serial number together with indication of the referred Country
 - **Language** - default language for the Controller
 - **Move Programming** - set to MODAL by default

- **Safety Stop Time** - in case of PDTV/O (/ORB), the user is requested to specify the value, in seconds, for setting the safety stop time.



See also par. [Safety Stop: setting the stop circuit timer](#), in the [C5G Control Unit - Transport and Installation manual](#), for further information.



- To complete the procedure, the User can choose one of the following keys (as shown in the above figure):
 - **Save** - to save the inserted data. Touch **Restart** at the end of saving operation.
 - **Restart** - to exit from the procedure. The System is restarted. If the data have NOT been saved, e.g. for any wrong operationy, an informational message warns the User about that. Provide saving data and then touch **Restart** key again.

5.20.3.11 Startup

In this sub-page it is allowed specifying the name of the startup program which is to be run each time the system restarts (**STARTUP** field in the below figure).

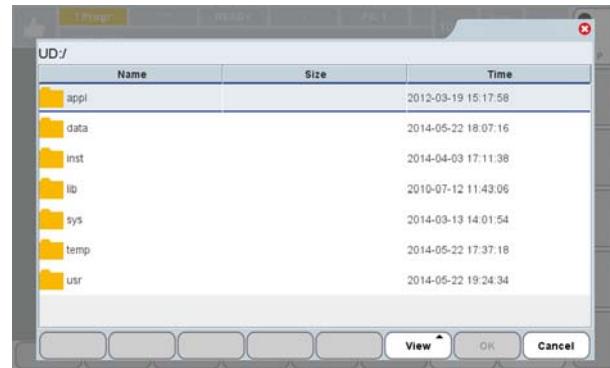
Such a program may be holdable or non holdable; running a holdable program only starts when the **START pushbutton is pressed**.

The name of the startup program is stored in \$STARTUP predefined variable.
To specify it, touch **STARTUP** field and choose it in the displayed folder and subfolders, if existing.

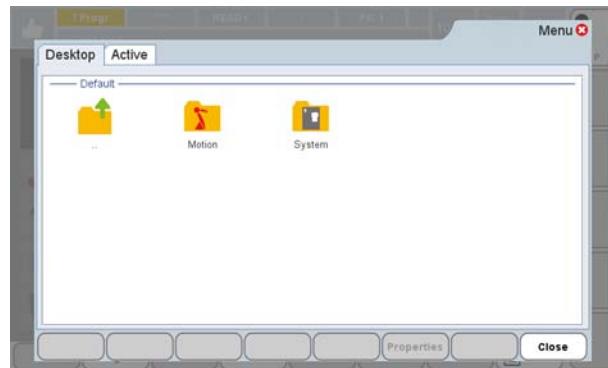
Select the wished file and touch **OK** to confirm.



If no program name is selected with this command, the contents of variable \$STARTUP are reset, therefore, if the setting is saved, no startup program will be run at the Controller restart time.



5.21 Service Page



This User Page is the starting point for the service operations.

It is composed by two subenvironments, selectable by touching the corresponding icons:

- [Motion](#)
- [System](#).

Select the icon related to the wished subpage to activate it.



When it is wished to switch the active subpage, touch **Close** key from the current subpage, go back to the main page and select the wished subpage.

5.21.1 Motion



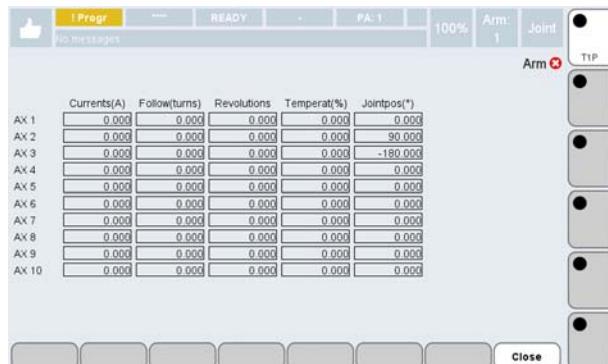
- [Arm](#)
- [MultiArm](#).

**PAY ATTENTION!**

The use of **AuxAxis** and **SetTemp** environments is reserved to Comau After Sales Service.

5.21.1.1 Arm

Fig. 5.59 - Arm subpage



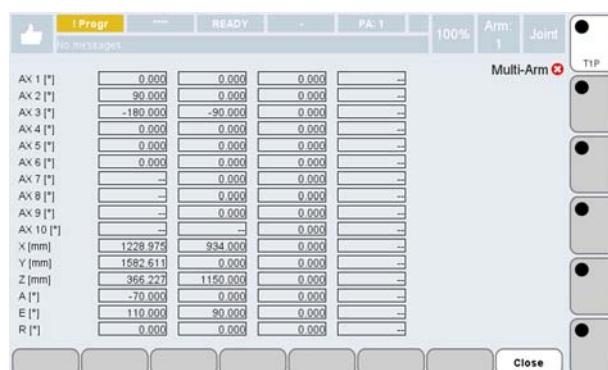
This screenshot shows the 'Arm' subpage of the service interface. The top bar includes tabs for 'Program', 'READY', 'PA-1', '100%', 'Arm: 1', 'Joint', and 'TIP'. A message indicator shows 'No messages'. On the right, there's a status bar with 'Arm 1' and a red 'TIP' icon. Below the header is a table with columns: Currents(A), Follow(turns), Revolutions, Temperat(%), and Jointpos(*). The table contains data for 10 axes (AX 1 to AX 10), all showing 0.000 values. At the bottom are several buttons and a 'Close' button.

Currents(A)	Follow(turns)	Revolutions	Temperat(%)	Jointpos(*)
AX 1	0.000	0.000	0.000	0.000
AX 2	0.000	0.000	0.000	90.000
AX 3	0.000	0.000	0.000	-180.000
AX 4	0.000	0.000	0.000	0.000
AX 5	0.000	0.000	0.000	0.000
AX 6	0.000	0.000	0.000	0.000
AX 7	0.000	0.000	0.000	0.000
AX 8	0.000	0.000	0.000	0.000
AX 9	0.000	0.000	0.000	0.000
AX 10	0.000	0.000	0.000	0.000

This page includes more specific information than the one in [Basic](#) subpage ([Motion Page](#)), related to the axes existing in the system (see [Fig. 5.59](#)). These are read-only data:

- **Currents** - indicates the current of the axis motor (ampere)
- **Follow** - indicates the error between the current position of the axis and the wanted one, expressed in motor turns
- **Revolutions** - indicates the current position of the axis, expressed in motor turns
- **Temperat.** - indicates the temperature of the motor relating to that axis, as a percentage of the ratio between the mean value absorbed and the motor characteristics. More precisely, the calculation of this value is based upon a formula using the square of the average current taken by the motor and the square of such a motor characteristic.
- **Jointpos** - indicates the current joint position of each axis of the current arm

5.21.1.2 MultiArm



This screenshot shows the 'Multi-Arm' subpage of the service interface. The top bar includes tabs for 'Program', 'READY', 'PA-1', '100%', 'Arm: 1', 'Joint', and 'TIP'. A message indicator shows 'No messages'. On the right, there's a status bar with 'Multi-Arm 1' and a red 'TIP' icon. Below the header is a table with various parameters. The table includes rows for axes (AX 1 to AX 10) with their respective values in parentheses, and global coordinates X, Y, Z in mm, and angles A, E, R in degrees. At the bottom are several buttons and a 'Close' button.

AX 1 ["]	0.000	0.000	0.000	—
AX 2 ["]	90.000	0.000	0.000	—
AX 3 ["]	-180.000	-90.000	0.000	—
AX 4 ["]	0.000	0.000	0.000	—
AX 5 ["]	0.000	0.000	0.000	—
AX 6 ["]	0.000	0.000	0.000	—
AX 7 ["]	—	0.000	0.000	—
AX 8 ["]	—	0.000	0.000	—
AX 9 ["]	—	0.000	0.000	—
AX 10 ["]	—	—	0.000	—
X [mm]	1228.975	934.000	0.000	—
Y [mm]	1582.511	0.000	0.000	—
Z [mm]	366.227	1150.000	0.000	—
A ["]	-70.000	0.000	0.000	—
E ["]	110.000	90.000	0.000	—
R ["]	0.000	0.000	0.000	—

Displays the current position (in Joint and in Cartesian mode) of all the axes of all the existing Arms in the System.

The maximum number of displayed axes is 40.

5.21.2 System



- Execute
- Info
- Moni
- Trace.



PAY ATTENTION!

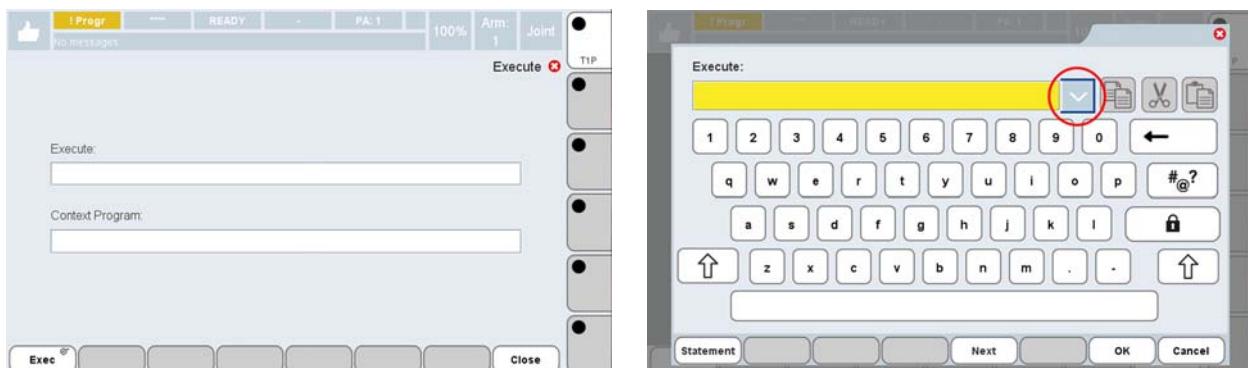
The use of **R.o.o.t.** environment is reserved to Comau After Sales Service.

5.21.2.1 Execute

This subpage is used to execute an instruction without the need to have it in the body of a PDL2 program.

The editable fields (as shown in the following left figure) are:

- **Execute**
is a field containing the being executed instruction. Usually such an instruction should have a short execution time, however it is possible inserting more than one instruction in this field, separating them by ';' (semi colon).
- **Context Program**
is an option that can be activated by means of **Exec** key (long touch).



It is to be specified only when the being executed instruction contains references to predefined variables (preceded by \$, e.g. \$CYCLE) belonging to "Program Stack" category (see [PDL2 Programming Language](#) manual, chapter [PREDEFINED VARIABLES LIST](#)).

If a Context Program is specified (name of a PDL2 program), the predefined variable will belong to it. If such a program is not active, an error occurs.

If instead no Context Program is specified, the predefined variables of program stack type will belong to the virtual EXECUTE program, existing during the execution only.

To use the **Execute** command, proceed as follows:

- a. touch **Execute** field to open the [PDL2 keyboard](#)
- b. a list of predefined choices is displayed when touching the red highlighted field (see previous right figure), to help inserting the instructions. Note that the being executed instruction can also be either directly typed in or chosen by touching **Statement** key.
- c. When the wished instruction has been inserted (e.g. *DELAY 10000*), touch **OK** key to confirm it.
- d. To start the execution, short touch **Exec** key in the [Functional keys menu](#).



During the execution, the **Abort** key is available in the [Functional keys menu](#) to interrupt the required operation, if needed (see the shown example in the above figure).

5.21.2.2 Info



Displays the following information groups in a [TreeGrid](#) type structure:

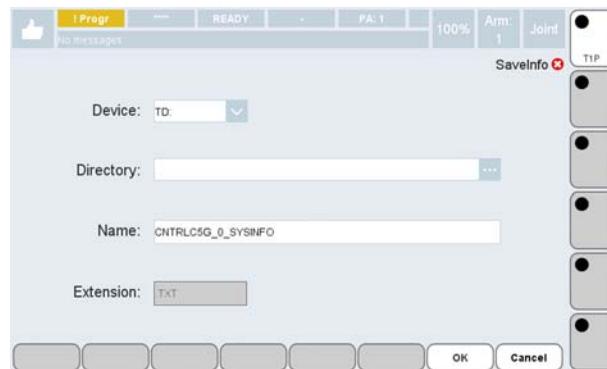
- **CONFIGURATION**
- **SOFTWARE VERSIONS**
- **SOFTWARE OPTIONS**
- **HARDWARE**

– COMMUNICATION.

Save to command, in the [Functional keys menu](#), allows to save System information into a text file according to the following rules:

- the default **Name** is **<machine_name>_SYSINFO.txt** (where **<machine_name>** stands for the machine SYS_ID - in the shown example in the below figure, it is **CNTRLC5G_0**)
- the default **Device**, when NO USB flash disks are inserted, is **TD**: otherwise, when a USB flash disk is inserted in a USB port, the default device is the port in which the USB flash disk is currently inserted.

When the insertion is completed, touch **OK** key to confirm saving.



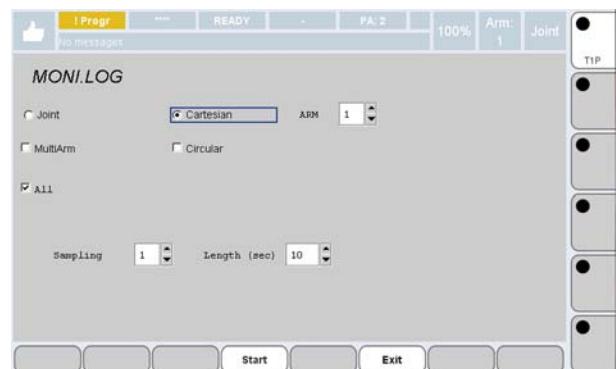
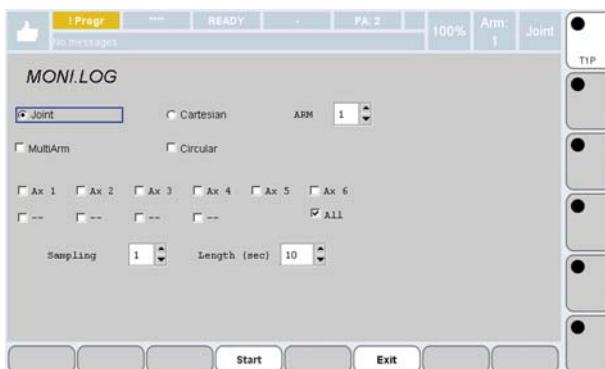
5.21.2.3 Moni

Moni is a debug tool, so the user should use it just in case of probable problems for the Arm (or for the Arms), in order to collect data about it.

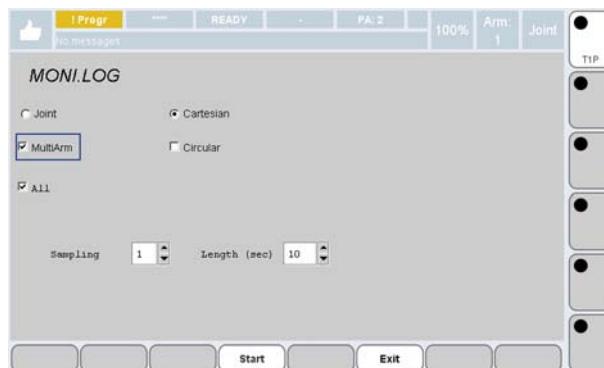
Such data are saved into binary files which should be sent to Comau in order to analyze the situation.

First of all, the User must select the motion characteristics to be monitored:

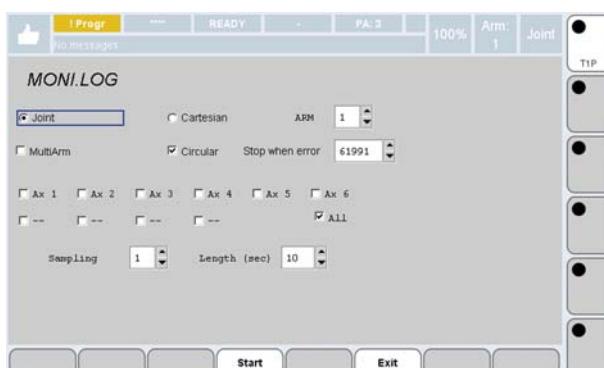
- **Joint/Cartesian** - indicatez the being monitored motion type, thus the acquired data type. If Cartesian is selected, **Ax1..Ax6** checkgroup is not available; the only available is **All** checkbox.
- **ARM** - indicates the involved Arm for the data acquisition



- **Multiarm** - if checked, it causes the **ARM** choice to be automatically uninfluential (because the Multiarm moni is performed on all the Controller Arms).

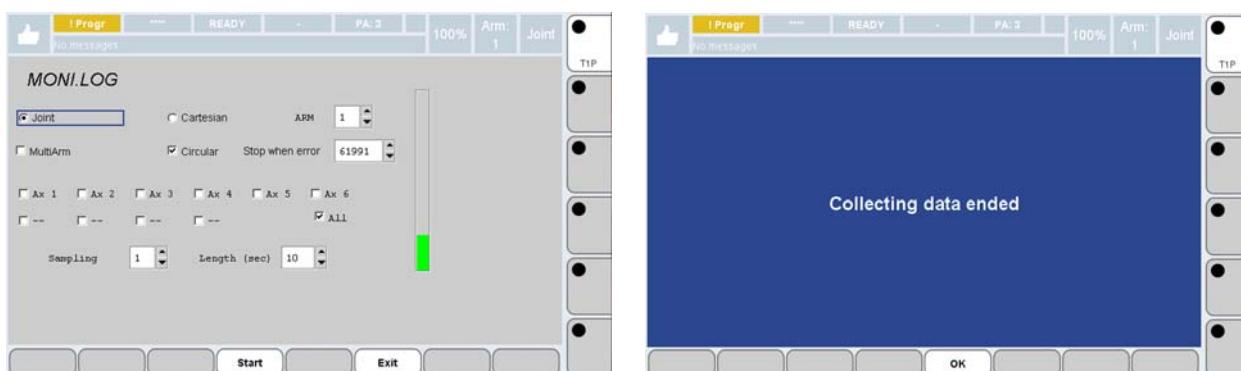


- **Circular** - indicates whether the data file will be cyclically overwritten, until the specified error in field **Stop when error** is triggered
- **Stop when error** - this field is displayed just when **Circular** field is selected. It indicates the error code whose occurrence is to be monitored. As soon as such an error occurs, the program closes the data file and ends with data acquisition.



- **Ax1..Ax6** and subsequent ones / **All** - indicate the axes whose data will be collected
- **Sampling** - indicates the number of interpolator ticks (\$IPERIOD) between an acquisition and the following one
- **Length (sec)** - indicates the data file dimension (the longer the acquisition time, the bigger the file).

After setting the wished features, touch **Start** key to begin data acquisition. A progress bar on the right, informs the User about the status of the operation. Touch **OK** when completed.

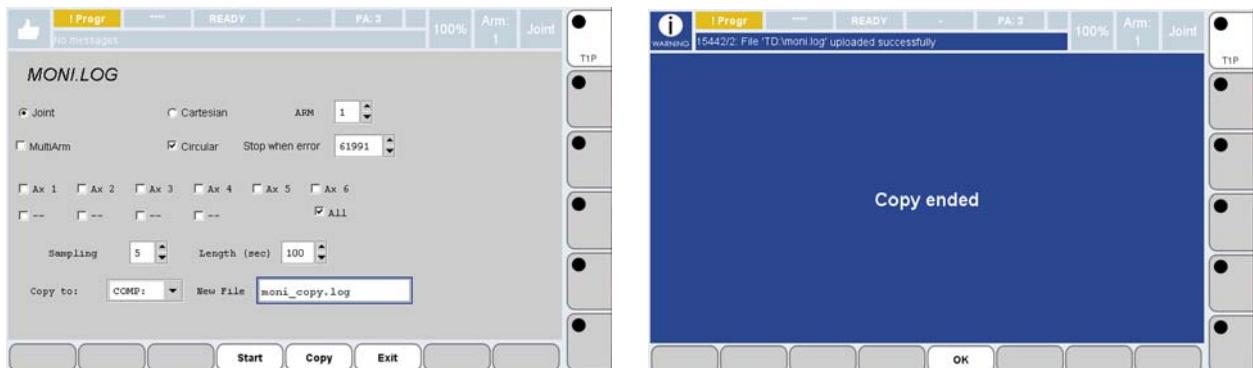


The program creates a binary file called '**moni.log**'.



WARNING - Each new data acquisition will overwrite the previous file and the possible controller shutdown will cause the file to be deleted. It is then strongly recommended to save it with a meaningful name, in order not to lose the acquired information.

It is allowed to create a copy of '**moni.log**' file of the collected data, with a name chosen by the user, on the wished device (as shown in the below left figure).



To do so, act in the following way:

- select the device in **Copy to** field
- insert the wished name in **New File** field
- Touch **Copy** key; please note that to copy to **COMP:** device, it must have previously been set in the suitable **WinC5G** window ('File Transfer Folder' field, [Properties Window](#)).
- Finally, touch **OK** key.
- If no further acquisition is requested, touch **Exit** key to finish.

The data file should be sent to Comau to analyze the situation.



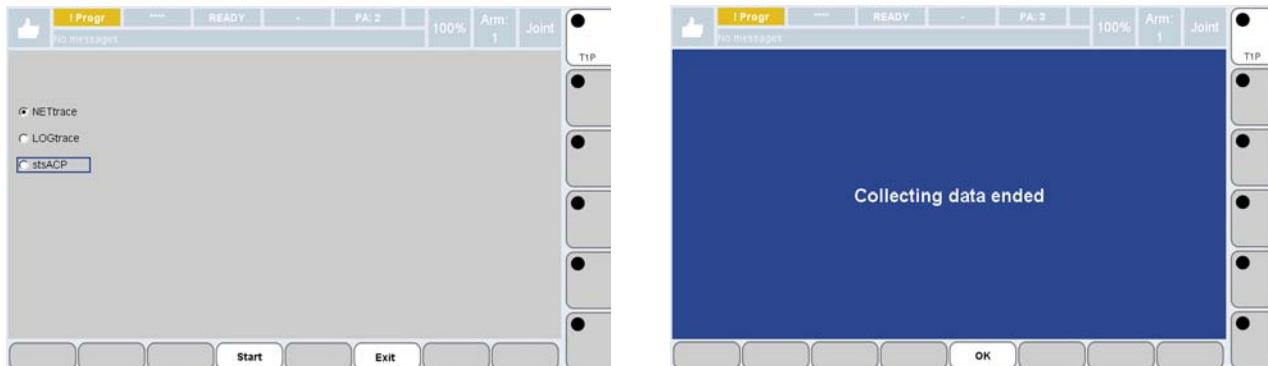
Before using Moni environment, please contact Comau After Sales Service for information about optimal acquisition conditions and problem analysis.

5.21.2.4 Trace

This subpage allows collecting data about drives communication problems on the POWERLINK network.

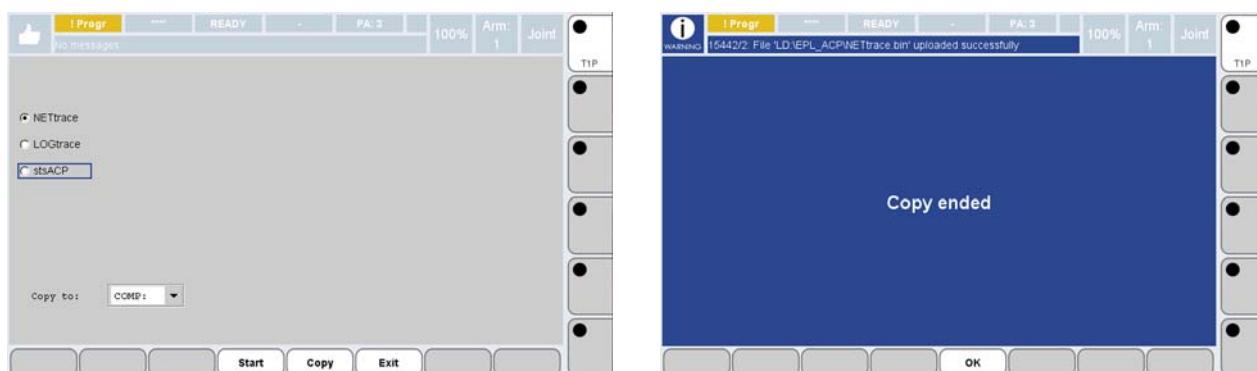
Such data are saved into files which should be sent to Comau in order to analyze the situation.

After selecting the required data type, touch **Start** key to start collecting them.



Touch **OK** key when completed.

Now the program allows copying the collected data to a binary file: select the wished device onto which the file is to be created, by means of **Copy to** field and touch **Copy** key.



At the operation completion, touch **OK** key.

If no further acquisition is requested, touch **Exit** key to finish.

The data file should be sent to Comau to analyze the situation.



Before using Trace environment, please contact Comau After Sales Service for information about optimal acquisition conditions and problem analysis.

5.22 Customizations on the TP

Current section contains information about the following topics:

- [User table creation from DATA environment](#)
- [Handling TP right menu](#)
- [Customizing the PDL2 Statements insertion, in IDE environment.](#)
- [Creating user pages in SETUP and SERVICE environment](#)
- [Adding Projects/Applications, installable from within SETUP Page.](#)

5.22.1 User table creation from DATA environment



Please, refer to par. [5.19 Data Page on page 184](#), for further information.

The PDL2 language allows the user to create a table and add it to the TP interface in the [Data Page](#).

The operation to be done are as follows:

- a. Table type definition: a RECORD must be defined which fields are the columns of the table.
- b. Table elements definition: the rows of the table are variables declared of the above type, as in a. step.



Variables of the above type that do not need to be displayed in the DATA table, must be declared with the NODATA attribute (please refer to [par. VAR Statement in PDL2 Programming Language manual](#)).

- c. Table viewing: the ***TABLE_ADD built-in procedure*** must be called.

Here follows a sample program.

```

PROGRAM exampletable NOHOLD
TYPE usertable = RECORD
    col1 : INTEGER
    col2 : BOOLEAN
    col3 : REAL
ENDRECORD

VAR line1, line2 : usertable
    line3_6 : ARRAY[4] OF usertable
BEGIN
    line1.col1:=10
    line1.col2:=TRUE
    line1.col3 :=4.5

    TABLE_ADD('usertable', $PROG_NAME, $PROG_NAME)

```

```
END exempletable
```

- The table contents are stored in a .VAR file specified in the **TABLE_ADD Built-In Procedure**.
- The table is removed from the DATA Page issuing the **TABLE_DEL Built-In Procedure**.

In the following sections detailed information is given about:

- [Properties](#)
 - [Variable <type>_signal](#)
 - [Variable <typename>validate_name_err](#)
 - [Example program for a table creation.](#)
-

5.22.1.1 Properties

It is possible to associate some properties [to the whole table](#) or [to fields](#) of it. Detailed information is provided in the following paragraphs.

For doing it, some variables have to be defined in the program header, according to the following syntax:

<type_name>_<operation> : when the operation is applied to the table.

<type_name>_<field_name>_<operation> : when the operation is applied to a field of the table.

<type_name>_<field_name>_<field_of_a_position>_<operation> : when the operation is applied to a field of a positional variable belonging to the table.

<type_name> is the table;

<field_name> is the field of the table;

<field_of_a_position> is a field of a positional variable belonging to the table;

<operation> is one of the predefined functions listed here below.

They are grouped as follows:

- [Table \(global\) Properties](#)
 - [Field Properties](#)
 - [Field of a POSITION Properties.](#)
-

5.22.1.1.1 Table (global) Properties

The listed below properties apply to the whole table (i.e. they are “global” to the table).



NOTE that the Table Property Name must always be preceded by **<type_name>** (i.e. the table **typename**).

Example:

if the table **typename** is “mytable”, the “[_get_active_element](#)” property must be written as

```
mytable_get_active_element
```

- [_get_active_element](#)
- [_get_auto_editable](#)

- [_get_can_auto_apply](#)
- [_get_editable](#)
- [_get_max_entries](#)
- [_get_msg_on_save](#)
- [_get_refresh_event](#)
- [_get_title](#)
- [_get_var_path](#)
- [_integer_get_max](#)
- [_integer_get_min](#)
- [_integer_get_step](#)
- [_real_get_max](#)
- [_real_get_min](#)
- [_real_get_step](#)
- [_get_default_name](#)
- [_get_all_variables.](#)

_get_active_element

<i>Name</i>	<code>_get_active_element</code>
<i>Description</i>	Get the active element in the table
<i>Data type</i>	STRING
<i>Read upon</i>	modifying the value of the variable (it is used for monitoring a variable)
<i>Default value</i>	none

_get_auto_editable

<i>Name</i>	<code>_get_auto_editable</code>
<i>Description</i>	Enabling flag for the table modification when the state of the controller is LOCAL. It is checked before enabling the modification of a field.
<i>Data type</i>	BOOLEAN
<i>Read upon</i>	table modification (of a field)
<i>Default value</i>	modifying is not enabled in LOCAL state

_get_can_auto_apply

<i>Name</i>	<code>_get_can_auto_apply</code>
<i>Description</i>	Enabling flag for the automatic apply function, which consists in the copy of the data viewed, in the table, directly in variables loaded in execution memory.
<i>Data type</i>	BOOLEAN
<i>Read upon</i>	table opening
<i>Default value</i>	Auto apply is enabled by default

_get_editable

<i>Name</i>	<code>_get_editable</code>
-------------	----------------------------

<i>Description</i>	Indicates whether that table can be edited or not
<i>Data type</i>	BOOLEAN
<i>Read upon</i>	table opening
<i>Default value</i>	TRUE (by default, modification is always permitted)

_get_max_entries

<i>Name</i>	_get_max_entries
<i>Description</i>	Maximum number of rows in the table
<i>Data type</i>	INTEGER
<i>Read upon</i>	adding a new row to the table
<i>Default value</i>	none

_get_msg_on_save

<i>Name</i>	_get_msg_on_save
<i>Description</i>	Generation of a message upon the event of table saving
<i>Data type</i>	STRING
<i>Read upon</i>	table saving
<i>Default value</i>	none

_get_refresh_event

<i>Name</i>	_get_refresh_event
<i>Description</i>	Indicates the PDL2 event used to inform the DATA Page that a global property has changed.
<i>Data type</i>	INTEGER
<i>Read upon</i>	table opening
<i>Default value</i>	0 (by default, no event is provided)

_get_title

<i>Name</i>	_get_title
<i>Description</i>	Table name
<i>Data type</i>	STRING
<i>Read upon</i>	table opening
<i>Default value</i>	table name

_get_var_path

<i>Name</i>	_get_var_path
<i>Description</i>	Path (directory and file name) in which saving the .VAR file containing the table data. The '\\' character used in the directory path specification must be replaced by '\\'
<i>Data type</i>	STRING
<i>Read upon</i>	table saving
<i>Default value</i>	the same path used upon table load

_integer_get_max

Name	_integer_get_max
Description	Maximum value for INTEGER fields
Data type	INTEGER
Read upon	table opening
Default value	none

_integer_get_min

Name	_integer_get_min
Description	Minimum value for INTEGER fields
Data type	INTEGER
Read upon	table opening
Default value	none

_integer_get_step

Name	_integer_get_step
Description	Incremental and decremental step for INTEGER fields
Data type	INTEGER
Read upon	table opening
Default value	1

_real_get_max

Name	_real_get_max
Description	Maximum value for REAL fields
Data type	REAL
Read upon	table opening
Default value	none

_real_get_min

Name	_real_get_min
Description	Minimum value for REAL fields
Data type	REAL
Read upon	table opening
Default value	none

_real_get_step

Name	_real_get_step
Description	Incremental and decremental step for REAL fields
Data type	REAL
Read upon	table opening
Default value	0.1

_get_default_name

Name	_get_default_name
Description	Predefined prefix for new elements name
Data type	STRING
Read upon	table opening
Default value	none (the new variables haven't got a predefined prefix)

_get_all_variables

Name	_get_all_variables
Description	Enable reading variables from all loaded programs into memory
Data type	BOOLEAN
Read upon	table opening
Default value	FALSE (by default, variables of the owning program only are read)

5.22.1.1.2 Field Properties

The listed below properties apply to the fields of the table.



NOTE that the Field Property Name must always be preceded by both <**typename**> and <**field_name**> (i.e. the table **typename** and name of the field which it applies to), separated by '_' character.

Example:

if the table **typename** is "tab", and the field name is "curr", the "[_get_editable](#)" field property must be written as

```
tab_curr_get_editable
```

- [_get_editable](#)
- [_get_item_list](#)
- [_get_item_values](#)
- [_get_max](#)
- [_get_min](#)
- [_get_msg_on_change](#)
- [_get_step](#)
- [_get_tip](#)
- [_get_title](#)
- [_get_visible](#)
- [_get_format.](#)

_get_editable

Name	_get_editable
Description	Read-only flag for the field

<i>Data type</i>	BOOLEAN
<i>Read upon</i>	enabling the field modification
<i>Default value</i>	FALSE, which means that the field is enabled to the modification

_get_item_list

<i>Name</i>	_get_item_list
<i>Description</i>	List of possible items associated to a field of the table
<i>Data type</i>	ARRAY OF STRING
<i>Read upon</i>	table opening
<i>Default value</i>	none

_get_item_values

<i>Name</i>	_get_item_values
<i>Description</i>	List of values related to each item listed with the _get_item_list property
<i>Data type</i>	ARRAY OF INTEGER
<i>Read upon</i>	table opening
<i>Default value</i>	0 .. [max_items-1]

_get_max

<i>Name</i>	_get_max
<i>Description</i>	Maximum value for the field
<i>Data type</i>	INTEGER or REAL
<i>Read upon</i>	selecting the field for being modified
<i>Default value</i>	maximum limit of the PDL2 data type

_get_min

<i>Name</i>	_get_min
<i>Description</i>	Minimum value for the field
<i>Data type</i>	INTEGER or REAL
<i>Read upon</i>	selecting the field for being modified
<i>Default value</i>	minimum limit of the PDL2 data type

_get_msg_on_change

<i>Name</i>	_get_msg_on_change
<i>Description</i>	Prompt a message after a field modification
<i>Data type</i>	STRING
<i>Read upon</i>	Apply command is issued on the field
<i>Default value</i>	none

_get_step

<i>Name</i>	_get_step
-------------	-----------

<i>Description</i>	Incremental/decremental step for the field
<i>Data type</i>	INTEGER or REAL
<i>Read upon</i>	selecting the field for being modified
<i>Default value</i>	1 for INTEGERs and 0.1 for REALs

_get_tip

<i>Name</i>	_get_tip
<i>Description</i>	Help string assignement
<i>Data type</i>	STRING
<i>Read upon</i>	table opening
<i>Default value</i>	no Help string

_get_title

<i>Name</i>	_get_title
<i>Description</i>	Field name
<i>Data type</i>	STRING
<i>Read upon</i>	table opening
<i>Default value</i>	field name

_get_visible

<i>Name</i>	_get_visible
<i>Description</i>	Flag for enabling the viewing of a field of the table
<i>Data type</i>	BOOLEAN
<i>Read upon</i>	table opening
<i>Default value</i>	TRUE, which means that the field is always viewed

_get_format

<i>Name</i>	_get_format
<i>Description</i>	Formatting floating point values in a field of the table. Used syntax for rounding type is ...0.01, 0.1 , 1 ,10, 100...
<i>Data type</i>	REAL
<i>Read upon</i>	table opening
<i>Default value</i>	no predefined format

5.22.1.1.3 Field of a POSITION Properties

The listed below properties apply to the Position type fields of a table.



NOTE that the Positional Fields Property Name must always be preceded by <type_name>, <field_name> and <field_of_a_position> (i.e. the table **typename**, the name of the field and the name of the positional field, which it applies to), separated by ‘_’ character.

Example:

if the table **typename** is “tab”, the field name is “pos1” and the field of the position is “x”, the “get_max” field property must be written as

tab_pos1_x_get_max

- get_max
- get_step
- get_min.

get_max

Name	<u>get_max</u>
Description	Maximum value for a field of a POSITION
Data type	INTEGER or REAL
Read upon	selecting a field for being modified
Default value	max limit of the PDL2 data type

get_step

Name	<u>get_step</u>
Description	Increment of a POSITION field
Data type	INTEGER or REAL
Read upon	selecting a field for being modified
Default value	1 for INTEGERs and 0.1 for REALs

get_min

Name	<u>get_min</u>
Description	Minimum value for a field of a POSITION
Data type	INTEGER or REAL
Read upon	selecting a field for being modified
Default value	minimum limit of the PDL2 data type

5.22.1.2 Variable <type>_signal

Such a variable is used by the user interface to signal an event to the PDL2 program which handles the tables. The PDL2 program has to check the variable value, in order to know whether the event occurred or not. The predefined value (0) indicates that none happened. In case of special events (for example table saving), for which a notification to the PDL2 program is needed, the user interface modifies its value as follows:

Event type	Variable value	When is it modified?
Saving a table	1	At the end of saving operations, if successfully completed
Reloading a table	2	At the end of reloading operations, if successfully completed

It is up to the PDL2 program to reset the variable (to the default value of 0) at the end of the event service routine, otherwise a further event of the same type could be not detected (since the value of the variable wouldn't change!).

If such a variable is not in the table, the user interface doesn't handle it at all.

5.22.1.3 Variable <typename>validate_name_err

This variable is used to validate the name of new elements in tables belonging to DATA environment.

- In the PDL2 program which handles the table, a routine should be defined having the following name:
`<typename>_validate_name(name: STRING)`
- When a new row is inserted in the table, the TP asks for the new element name and calls such a routine. The passed parameter contains the name typed in by the user:
 - if the routine does not exist, no errors occur: the TP goes on inserting the new element into the table (this is for compatibility with the current PDL2 programs which don't have such a routine).
 - On the contrary, if it is defined, the PDL2 routine performs all the required checks and sets a STRING type variable:
`VAR <typename>_validate_name_err: STRING [nn]`
- The TP reads such a variable in the following way:
 - if `<typename>_validate_name_err` variable either does not exist, or is UNINIT, or is empty, the name chosen by the user is considered as a valid name and a new row is created in the table.
 - If `<typename>_validate_name_err` variable includes a text, it is displayed to the user and the new row creation does not continue.

A PDL2 sample program follows, with guide-lines for creating a complete program.

5.22.1.3.1 Sample program for a variable name validation

```

PROGRAM test_var NOHOLD

TYPE usertable = RECORD
    col1 : REAL
    col2 : REAL
    col3 : REAL
    col4 : REAL
    col5 : REAL
ENDRECORD

VAR line1, line2 : usertable
    line3_4 : ARRAY[2] OF usertable
    usertable_validate_name_err : STRING[50]

-- Routine called by the TP
ROUTINE usertable_validate_name(name : STRING) EXPORTED FROM test_var
ROUTINE usertable_validate_name(name : STRING)

BEGIN
    -- Perform the needed check on the variable name
    -- If usertable_validate_name_err variable either does not exist, or is UNINIT,
    -- or is empty, the name chosen by the user is assumed as valid and a new row
    -- is created in the table. Otherwise the user is warned by means of the message
    -- included in the variable itself
    usertable_validate_name_err := 'the variable has got a wrong name'
END usertable_validate_name

BEGIN
    usertable_validate_name_err := ''
    .....
    TABLE_ADD('usertable', $PROG_NAME, $PROG_NAME)
END test_var

```

5.22.1.4 Example program for a table creation

```

PROGRAM test_tab NOHOLD
TYPE tab_def = RECORD
  curr : REAL
  curr1 : REAL
  volt1 : REAL
  volt2 : REAL
  pressure : REAL
  spd : INTEGER
  acc : INTEGER
  jerk : INTEGER
  dec : INTEGER
  mass : REAL
  enabled : BOOLEAN
  stokeend : BOOLEAN
  opened : BOOLEAN
  pos : POSITION
ENDRECORD
VAR
  tab_def_get_title : STRING[13] ('Test Table') NOSAVE
  tab_def_max_entries : INTEGER (5) NOSAVE
  tab_def_get_msg_on_save : STRING[44] ('Sample save message') NOSAVE
  tab_def_curr_get_title : STRING[7] ('Current') NOSAVE
  tab_def_curr_get_tip : STRING[13] ('Total current') NOSAVE
  tab_def_curr_get_max_value : REAL (3.0) NOSAVE
  tab_def_curr_get_min_value : REAL (-3.0) NOSAVE
  tab_def_curr1_get_visible : BOOLEAN (FALSE) NOSAVE
  tab_def_get_auto_editable : BOOLEAN (FALSE) NOSAVE
  tab_def_spd_get_item_list : ARRAY[3] OF STRING[10]
  tab_def_mass_get_editable : BOOLEAN (TRUE) NOSAVE
  tab_def_mass_get_msg_on_change : STRING[30] ('Sample edit message') NOSAVE
  tab_def_get_editable : BOOLEAN (TRUE) NOSAVE
  tab_def_get_refresh_event : INTEGER (50100) NOSAVE

  tab1 : tab_def
  tab2 : tab_def
  tab3 : tab_def
  tab4 : tab_def
  tab5 : tab_def
  tab6 : tab_def
  tab7 : ARRAY[3] OF tab_def

BEGIN
  -- Creates the table
  TABLE_ADD('tab_def', $PROG_NAME)

  tab_def_spd_get_item_list[1] := 'low'
  tab_def_spd_get_item_list[2] := 'medium'
  tab_def_spd_get_item_list[3] := 'high'

CYCLE

  -- Makes the table editable
  tab_def_get_editable := TRUE

```

```
-- Changes other properties
-- . . .

-- Informs the TP
SIGNAL EVENT 50100      -- uses the tab_def_get_refresh_event value

-- Wait 30 seconds
DELAY 30000

-- The table is not editable anymore
tab_def_get_editable := FALSE

-- Changes other properties
-- . . .

-- Informs the TP
SIGNAL EVENT 50100      -- uses the tab_def_get_refresh_event value

-- Wait 30 seconds
DELAY 30000

END test_tab
```

5.22.2 Handling TP right menu



Please, refer to par. 5.5.1 Display areas on page 65, for further information.

There are 6 keys on the TP that can be programmed by the user basing on the application to be developed.

Each page of the application could need more than 6 keys; pressing MORE key (on the right of the right menu) the remaining softkeys of the current page are displayed.

The changing from one page to another is circularly handled via the MORE key and the swapping occurs when the last softkey of the page has been displayed.

It is possible to disable a softkey but it is not allowed to disable a page of softkeys.

Each softkey can be in the following status:

- Enabled or disabled.
- The pressure of a disabled softkey does not have any effect. Please note that the key aspect remains unchanged in respect to when it is enabled: the PDL2 softkey handler program has to modify the key in terms of color, icon, text so to underline the disabled key status.
- Pressed or Released
- When the softkey is in the pressed state, the softkey is represented like if it were embedded.

A configuration file in XML format has to be written containing the definition of the softkeys; this file must be stored in the following folder:

UD : \SYS\TP\CONFIG\RIGHTMENU

Upon the Controller restart, all the files contained in the above directory are automatically loaded, in alphabetical order, on the TP.

Each page corresponds to the definition of a group of softkeys, which can be more than 6. The syntax is as follows:

```
<?xml version="1.0"?>

<NewMenu>
<group>
  <progName>GroupName</progName>
  <numItems>Number</numItems>
  <arName>Names</arName>
  <arIconName>Icons</arIconName>
  <arStatus>Colours</arStatus>
  <arEnabled>Enabled</arEnabled>
  <evPressed>NumberOfEvent</evPressed>
  <evLongPressed> NumberOfEvent </evLongPressed>
  <evReleased> NumberOfEvent </evReleased>
</group>

</NewMenu>
```

The words included between `< >` should always be present. The words specified in bold have to be replaced by a true parameter.

Example: 2 XML files are defined. In the first file, one group with 8 softkeys is defined; in the second one, 2 groups with 3 softkeys each are defined. TP will create the following pages:

- One page with the first 6 softkeys of the group of the first file
- One page with the remaining 2 softkeys of the group of the first file
- One page with the 3 softkeys of the first group of the second file
- One page with the 3 softkeys of the second group of the second file.

A detailed description follows about the following topics:

- [XML Tag Parameters](#)
- [Softkey Pressure and Release events](#)
- [Example of XML configuration file](#)
- [Example of right menu configuration.](#)

5.22.2.1 XML Tag Parameters

Here follows the description of the XML tag parameters to be defined in the group specification.

```
<progName> ... </progName> Softkeys handler
```

The information required for handling each key belonging to a group are stored in variables which should be loaded in execution memory (Load command). The definition and the initialisation of these variables should be included in this program. The program should be loaded by the controller startup program.

```
< numItems >...</numItems > Number of items to be considered
```

This XML parameter indicates the number of elements which should be considered inside the array variables which contain the properties for each softkey.

The arrays could have a greater dimension but the number of elements taken into account is the one indicated by this parameter.

<arName>...</arName> Labels for the softkeys

This is the name of a PDL2 variable, defined in the <progName> program, which contains the labels associated to the softkeys of the group. This variable should be defined as an ARRAY OF STRING. The maximum number of characters of each label depends on the dimension of the key and by the font used for the label. The strings which overflow the key dimension are truncated.

The TP continuously monitors the content of the array, and if it changes the effect on the softkey is immediate.

<arIconName>...</ arIconName > Icons for the softkeys

This is the name of a PDL2 variable, defined in the <progName> program, which contains the icons associated to the softkeys of the group. This variable should be defined as an ARRAY OF STRING. The content of each element is the name of the file (.GIF) containing the icon. This file should be stored in the same folder of the .XML file. For example, if the element 5 of the PDL2 variable <arIconName> defined in <progName> program has the value 'TEST', the TP shows the icon present in TEST.GIF.

The TP continuously monitors the content of the array, and if it changes, the effect on the softkey is immediate.

< arEnabled >...</ arEnabled > Softkey Enabling / disabling flag

This is the name of a PDL2 variable, defined in the <progName> program, which contains the enabling/disabling flag for the softkeys of the group. This variable should be defined as an ARRAY OF BOOLEAN. The value FALSE disallowes the pressure of the corresponding softkey. Please note that the look and feel of the softkey is not changed at all by the TP; the softkey handler program <progName> should eventually handle a change of aspect of the key.

The TP continuously monitors the content of the array, allowing the content to be changed anytime and the effect on the softkey is immediate.

< arStatus>...</ arStatus > Softkey colour

This is the name of a PDL2 variable, defined in the <progName> program, which contains the colour for the softkeys of the group. This variable should be defined as an ARRAY OF INTEGER. The possible values are the predefined constants related to the colours : WIN_BLUE, WIN_RED, WIN_... (please refer to **Chap.12. - Predefined Variables List - PDL2 Programming Language** manual).

The TP continuously monitors the content of the array, and if it changes the effect on the softkey is immediate.

5.22.2.2 Softkey Pressure and Release events

There are three kind of events related to the pressure of a softkey:

- Softkey Pressure Event (< evPressed >...</ evPressed >)
- Softkey Release Event (< evReleased >...</ evReleased>)
- Softkey 'long lasting' pressure Event (< evLongPressed>...</ evLongPressed >)

They are described here below.

```
< evPressed >...</ evPressed >
```

An event of key pressure can be associated to each softkey.

A user event CONDITION (WHEN EVENT ...) statement should monitor the triggering of these kind of events. The class of user events is identified in the range of errors included between 49152 and 50175. This XML tag should contain the event number associated to the first softkey of the group. The TP will automatically assign the events for the other softkeys of the group in a sequential order. For example, defining `<evPressed>50100</evPressed>` in a group of 6 softkeys, the event 50100 will be generated upon the key pressure of the first softkey of the group, the event 50105 will be generated upon the key pressure of the 6th softkey of this group.

```
< evReleased >...</ evReleased >
```

An event of key release can be associated to each softkey.

The way for using it is the same as for the key pressure described above. The only difference is in the XML tag definition.

```
< evLongPressed>...</ evLongPressed >
```

This event triggers when the softkey of the TP is maintained pressed for more than 2 seconds.

The way for using it is the same as for the key pressure described above. The only difference is in the XML tag definition.

```
< evShow>...</ evShow > Force the viewing of a softkey group
```

It is possibile to force the viewing of a group of softkeys. For doing this a user event must be defined (WHEN EVENT..) which will trigger upon a SIGNAL EVENT of the same number. The TP shows the group when the event triggers.

5.22.2.3 Example of XML configuration file

This example file defines two groups of softkeys. The first group has 4 keys with the names defined in 'nomi', the icons are defined in 'icone', the enabling is stored in 'en' and the colours are defined in 'colori'. The handler program is 'group1'.

The second group defines 2 softkeys and the attributes are stored in the same variables. The handler program is 'group2'.

```
<?xml version="1.0"?>

<NewMenu>
<group>
  <progName>group1</progName>
  <numItems>4</numItems>
  <arName>nomi</arName>
  <arIconName>icone</arIconName>
  <arStatus>colori</arStatus>
  <arEnabled>en</arEnabled>
  <evPressed>50100</evPressed>
  <evLongPressed>50120</evLongPressed>
  <evReleased>50130</evReleased>
</group>

<group>
  <progName>group2</progName>
```

```

<numItems>2</numItems>
<arName>nomi</arName>
<arIconName>icone</arIconName>
<arStatus>colori</arStatus>
<arEnabled>en</arEnabled>
<evPressed>50150</evPressed>
<evLongPressed>50160</evLongPressed>
<evReleased>50170</evReleased>
</group>

</NewMenu>

```

This example program, assigns a colour to the key which has been pressed and disables the other softkey.

```

PROGRAM group1 NOHOLD
VAR
    nomi : ARRAY[5] OF STRING[10]
    icone: ARRAY[5] OF STRING[10]
    en : ARRAY[5] OF BOOLEAN
    colori : ARRAY[5] OF INTEGER
BEGIN
    CONDITION[1] NODISABLE :
        WHEN EVENT 50100 DO -- press. of the 1st softkey of the group
            colori[1] := WIN_RED -- make RED the first softkey
            en[2] := FALSE -- disable the second softkey
        WHEN EVENT 50101 DO -- press. of the 2nd softkey of the group
            colori[2] := WIN_GREEN -- make GREEN the first softkey
            en[1] := FALSE -- disable the first softkey
    ENDCONDITION
    ENABLE CONDITION[1]
CYCLE
---
DELAY 1000
---
END group1

```

5.22.2.4 Example of right menu configuration

This XML file defines a group with 5 softkeys. The handler program is called ‘rightmenu’. The names are stored in ‘nomi’, the icons in ‘icone’, the enabling in ‘en’ and the colours in ‘colori’. The event for the pressure of the first softkey is 50120. The event for the long pressure of the first softkey is 50120. The first event for releasing of the first softkey is 50000.

```

<?xml version="1.0"?>

<RightMenu>

<group>
<progName>rightmenu</progName>
<numItems>5</numItems>

```

```

<arName>nomi</arName>
<arIconName>icone</arIconName>
<arStatus>colori</arStatus>
<arEnabled>en</arEnabled>
<evPressed>50100</evPressed>
<evLongPressed>50120</evLongPressed>
<evReleased>50130</evReleased>
<evShow>50000</evShow>
</group>

</RightMenu>
```

This program initializes the first four softkeys of the right menu.

```

PROGRAM rightmenu NOHOLD
VAR
  nomi : ARRAY[5] OF STRING[10]
  icone : ARRAY[5] OF STRING[10]
  en :      ARRAY[5] OF BOOLEAN
  colori : ARRAY[5] OF INTEGER
BEGIN
  -- first softkey initialization
  nomi[1] := 'S1'
  icone[1] := 'i1'
  en[1] := TRUE
  colori[1] := WIN_BLUE

  -- second softkey initialization
  nomi[2] := 'S2'
  icone[2] := 'i1'
  en[2] := TRUE
  colori[2] := WIN_RED

  nomi[3] := 'S3'
  icone[3] := 'i1'
  en[3] := TRUE
  colori[3] := WIN_YELLOW

  nomi[4] := 'S4'
  icone[4] := 'i1'
  en[4] := TRUE
  colori[4] := WIN_GREEN

  CONDITION[1] NODISABLE :
    WHEN EVENT 50100 DO
      colori[1] := WIN_RED
      colori[2] := WIN_GREEN
    WHEN EVENT 50101 DO
      colori[1] := WIN_PINK
      colori[2] := WIN_RED
  ENDCONDITION
  ENABLE CONDITION[1]
```

```

CYCLE
  DELAY 1000
  icone[2] := 'i2'
DELAY 1000
  icone[2] := 'i1'
END rightmenu

```

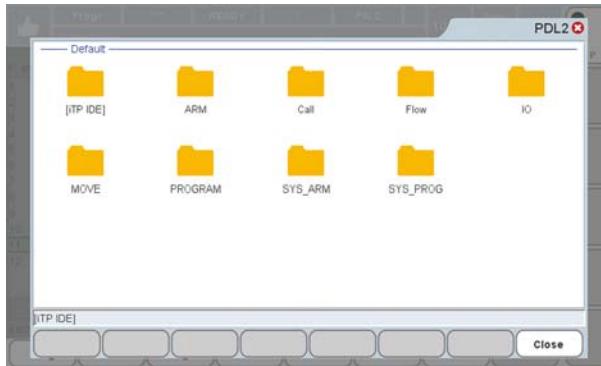
5.22.3 Customizing the PDL2 Statements insertion, in IDE environment

In IDE environment (on the Teach Pendant), the user is allowed to customize the PDL2 menu, adding new instructions to it.

5.22.3.1 Adding User Instructions to the PDL2 menu

To do that, the user has to write an XML file to fully describe the structure of the being added instruction.

The XML file must be included in the suitable directory of the Controller, so that it is listed together with all the already available instructions.



Three examples follow, of XML files, for

- [Adding a Statement](#),
- [Adding a Routine](#),
- [Adding a Built-in Routine](#).

5.22.3.1.1 Adding a Statement

In the following example, the DELAY instruction template is described by means of an XML file, to obtain the template shown in [Fig. 5.60](#).

Fig. 5.60 - DELAY statement template

```

DELAY <time>
<keyword>
<text>DELAY % </text>
<param>
<text>time</text>

```

```
</param>
</keyword>
```

- **<text>DELAY % </text>** states that a string will be displayed containing the instruction name (**DELAY**), with **one parameter**.
 - The section included between **<param>** and **</param>** describes the structure of parameter:
 - the field name displayed in the template (**<text>time</text>**).
-

5.22.3.1.2 Adding a Routine

In the following example, a Routine Calling template is described by means of an **xml** file, to obtain the template shown in [Fig. 5.61](#).

Fig. 5.61 - Routine Calling template

```
A_TOOL(<index>)
<keyword>
  <import>
    <progName>tt_tool</progName>
    <type>
      <typeName>routines</typeName>
      <name>a_tool</name>
    </type>
  </import>

  <text>A_TOOL (%) </text>
  <param>
    <text>index</text>
  </param>
</keyword>
```

- the section between **<import>** and **</import>**, describes any information which is needed to EXPORT the routine FROM
 - **<progName>tt_tool</progName>** is the name of the owning program (tt_tool) the routine is to be EXPORTED FROM
 - **<typeName>routines</typeName>** indicates the type of the object to be EXPORTED FROM
 - **<name>a_tool</name>** is the routine name
 - **<text>A_TOOL (%) </text>** states that a string will be displayed in the template, containing the routine name (**A_TOOL**), with **one parameter**
 - The section included between **<param>** and **</param>** describes the structure of parameter:
 - the field name displayed in the template (**<text>index</text>**).
-

5.22.3.1.3 Adding a Built-in Routine

In the following example, the AUX_COOP Built-in Routine calling template is described by means of an **xml** file, to obtain the template shown in [Fig. 5.62](#). Such a file must be

saved in the following directory:

UD : \SYS\TP5\IDE\KEYWORDS\

Fig. 5.62 - Built-in calling template

```
AUX_COOP(<flag>, <aux_joint>, <arm_num>)

<keyword>
  <text>AUX_COOP (% % %) </text>

  <param>
    <text>flag</text>

    <context>ON</context>
    <context>OFF</context>
  </param>

  <param>
    <text>aux_joint</text>
  </param>

  <param>
    <text>arm_num</text>
  </param>

</keyword>
```

- **<text>AUX_COOP (% % %) </text>** states that a string will be displayed in the template, containing the routine name (**AUX_COOP**), with **three parameters**
- The three sections included between **<param>** and **</param>** describe the structure of the parameters:
 - the field names displayed in the template (**<text>flag</text>**, **<text>aux_joint</text>**, **<text>arm_num</text>**)
 - **<context>ON</context>** and **<context>OFF</context>** indicate the strings to be displayed in the menu which becomes available when the 'flag' field is selected.

When the **AUX_COOP.xml** file is created in the suitable directory, the AUX_COOP built-in routine is immediately available.

5.22.4 Creating user pages in SETUP and SERVICE environment

This functionality allows accessing to new user pages, written in VP2 language, from within SETUP or SERVICE environments.

To do so, for each new user page it is needed to create an **XML** file including the name of the being activated user program, together with the path to reach the .COD file; the **XML** file must be stored in the following folder

UD : \SYS\TP5\PAGES

together with the corresponding icon image (if existing), having the following characteristics:

- format - **gif**,
- dimensions - **32x32 pixel**
- image filename - **<name>.xml.gif**, where **<name>** is the same name of the **XML** file describing the new user page.



Please, NOTE THAT both the XML file and the associated icon gif file, MUST be stored in the same folder!

A full example of a VP2 program creating a new user page (**PDL** file) and an **XML** file to access it, is shown below:

- VP2 program to create the **SpinDial** user page (**SpinDial.PDL** file):

```

PROGRAM SpinDial NOHOLD
IMPORT 'VP2_LIB.COD'

VAR ve_screen : vp2screen
  vi_scrn_id : INTEGER
  ve_spin : vp2spindial

BEGIN
-- The page is available in different ways, depending on the following value:
-- 0: The page is shown as a subpage in the SETUP page
--   (in this case, an XML file will be required)
-- 1: The page is available from the LEFT MENU
ve_screen.ordinal := 0

ve_screen.x := 0
ve_screen.y := 0
ve_screen.w := -1
ve_screen.h := -1
ve_screen.name := $PROG_NAME

ve_spin.x := 50
ve_spin.y := 50
ve_spin.w := 200
ve_spin.h := 32
ve_spin.value := 42

vi_scrn_id := VP2_SCRN_CREATE(ve_screen)
VP2_ADD(ve_screen, ve_spin)

-- Switch according to the ordinal selected above
IF ve_screen.ordinal > 0 THEN
-- Add to left menu
  VP2_SCRN_ADD(PDV_TP, vi_scrn_id)
ELSE
-- Make it available for the Setup page
  VP2_SCRN_AVAIL(vi_scrn_id, IP_TO_STR($SYS_PARAMS[61]), '255.255.255.0')
ENDIF

PAUSE
END SpinDial

```

- **XML** file to access the **SpinDial** user page (***SpinDial.XML*** file)

```
<?xml version="1.0" ?>
<config>
  <pageID>
    <class-name>VP2</class-name>
    <prog-name>UD:\Usr\SpinDial</prog-name>
  </pageID>
</config>
```

The words included between `<>` should always be present.

The content of this file, indicates that ***SpinDial.cod*** file, including the Program, is stored in **UD:\Usr\SpinDial** folder.

- The icon image file name must be ***SpinDial.XML.gif*** and it must be store in the same folder of ***SpinDial.XML*** file.

5.22.5 Adding Projects/Applications, installable from within SETUP Page

This functionality allows creating a customized installation program for an application to be listed in the **SETUP\INSTALL (FUI)** Page of the Teach Pendant.



To do so, it is needed to create a **.APPL** file for each application to be displayed in the Project/Application list. Each file can include more than one “Type”, and such an organization is useful, e.g. to handle different configurations of the same application. The following skeleton is provided as an example:

The required information, for each “type”, is listed below:

- name and path of the **.ZIP** file including the being installed files
- name of the being executed **.COD** program within the **.ZIP** file. After extracting all files from **.ZIP** file, the TP runs such a program which will copy/install the files in the suitable directories
- Synchronization variables with the TP. Such variables must be declared and handled by the **.COD** program in order to indicate both the operation status in the User Interface (e.g. operation in progress, operation completed, etc.) and any error message.

The following skeleton is provided as an example:

```
<?xml version="V1.0"?>
<proj>
  <type>
    <name>"TYPE NAME"</name>
```

```

<appl>
  <file> "NAME AND PATH OF THE .ZIP SOURCE FILE"</file>
  <help>
    <it>"ITALIAN DESCRIPTION or HELP"</it>
    <en>"ENGLISH DESCRIPTION or HELP"</en>
  </help>
  <progr-name>"NAME OF THE BEING RUN PROGRAM(CONTAINED IN.ZIP)"</progr-name>
  <var-result>"NAME OF THE RETURNED RESULT VARIABLE"</var-result>
  <var-message>"NAME OF THE MESSAGE VARIABLE FOR THE USER"</var-message>
  <new-zip>"NAME OF THE BEING USED DESTINATION .ZIP FILE"</new-zip>
</appl>
</type>

<type>
  .... further types
</type>

</proj>

```

The words included between '< >' should always be present.

Both the **.APPL** files and the corresponding **.ZIP** ones, must be created in the root directory of the device that will be specified in the [Install](#) SETUP subpage, during the installation operation. In this way, all related Projects/Applications are listed in the menu and the user is able to ask for them to be installed.

6. SYSTEM COMMANDS

6.1 Introduction

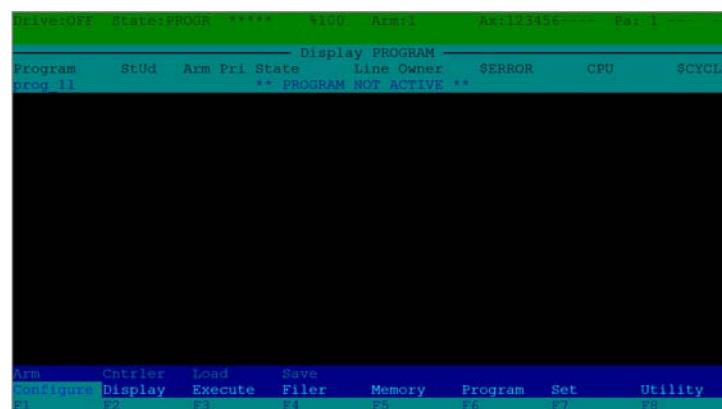
This chapter describes the system commands available for the C5G Control Unit. The commands described can be entered from the following interfaces:

- PDL2 program - using the pre-defined SYS_CALL routine
- From PC - **WinC5G** program, Terminal window menu

The items of the commands are shown here in the most extended form, as they are seen on the **WinC5G** screen page.

[Fig. 6.1](#), shows the **WinC5G** Terminal windows.

Fig. 6.1 – Commands window on PC (WinC5G)



(2) - Terminal window on WinC5G

In particular, the following are dealt with:

- Commands sent from WinC5G (Terminal window)
- Program commands sent via SYS_CALL
- Wildcard use
- Command options
- Viewing and NOPAGE option
- Access to the control (login)
- Directories
- Description of commands
- Types of files available in the System.

6.2 Commands sent from WinC5G (Terminal window)

The following subjects are dealt with in this paragraph:

- Command selection
- Typing-in the characters
- Call-up of recently used parameters (History)

6.2.1 Command selection

The commands can be entered using one of these methods:

- **using the cursor keys.** Enter it using the **ENTER** key.
After the selection, another menu of commands or a brief description of the command is displayed over the selected item.
- The selected command is only executed after the **ENTER** key has been pressed.
- On the PC screen and on the teach pendant, disabled or protected menu commands are shown with the first letter in low case, whereas all the commands that are available have the first letter in high case.
- **Press the function keys (F1 - F8)** on the PC keyboard or on the teach pendant to execute the corresponding command. For example, if **F2** is pressed, **DISPLAY** from the main menu will be executed.
- **Press the key that corresponds to the first letter of the command**, only for WinC5G Terminal. For example, if D is pressed, DISPLAY from the main menu will be executed.

In some cases, after a command has been sent, prompts, called parameters, are shown that require additional information. For example, PROGRAM GO displays a prompt that requires a program name. To enter the parameters in answer to a prompt, use the alphanumeric keys of the PC keyboard or the characters menu of the teach pendant. Furthermore, for many commands it is possible to select the parameter (file, program, I/O point, etc.) from a list supplied by the system after the HELP key near the prompt is pressed; to scroll the list use the cursor keys and the ENTER key. Besides which, for some commands, if the HELP key is pressed following the introduction of one or more characters after the prompt, only the parameters with the initials of the characters introduced will be displayed.

To exit from the HELP window without selecting any parameters, press **ESC**.

To return to the command prompt, press **SHIFT ESC** to return to the main menu.

For files, there is also the possibility to select the name of the device to be specified together with the file name by pressing **ENTER** at <device> that is displayed as the first item on the list of the possible files.

The answer to a parameter prompt is to be typed before pressing **ENTER**. To move the cursor on the answer line, the arrow keys can be used to the left or to the right. The delete key removes the character situated immediately before the cursor, whereas the arrow keys to the left and to the right if pressed together with **SHIFT** move the cursor to the start and to the end of the answer.

6.2.2 Typing-in the characters

On the PC, use the keyboard.

On the Teach Pendant, there is an alphanumeric keypad that works in the same way as one of the most common standard mobile telephone keypads. In detail:

- The bottom right key (to the right of '0') serves to set the mode to use the keypad. Each time this key is pressed, the mode is changed, in sequence, between
 - alphabetical, high case (ABC)
 - alphabetical, low case (abc)
 - numerical (123)
 The current mode is shown on the status bar of the TP display (last field on the right).
- Special characters are activated by pressing '1'; the only special characters that can be typed directly are the '-' symbol (bottom left key) and the '.' symbol (bottom right key).

6.2.3 Call-up of recently used parameters (History)

The upward arrow key is used to call the last ten parameters entered for the command category they belong to. This avoids rewriting the same parameter for several commands.

6.3 Program commands sent via SYS_CALL

The predefined PDL2 SYS_CALL routine can be used to execute most of the commands described in this chapter from the program, except those which require direct interaction with the user (for example, Program Edit). For the same reason the requests for confirmation are not displayed and the default answer for that command is assumed. The codes that are shown in brackets, next to the title of each command, are those to be specified as first parameter for the SYS_CALL.

The parameters required by each command have to be specified as STRINGS (for example: SYS_CALL ('ML' , 'pippo')).



For further details regarding the syntax of this instruction, see the PDL2 Programming Language manual.

6.3.1 Option /4 to view in the 40 columns

The **/4** option serves to format the displays so that the information shown does not exceed a width of 40 columns. This option is only available from SYS_CALL and only for those commands that, if not forwarded by teach pendant, display data that covers over 40 columns.

6.4 Wildcard use

The wildcard (*) can be used in a parameter to substitute the beginning, the end of the entire parameter. It permits one command to act on several items.

For example, **MEMORY LOAD (ML)** requires a program name parameter that can be substituted with a wildcard as shown below:

ML *	loads all the programs
ML arc*	loads all the programs that start with arc

For each parameter only one wildcard can be used (*arc* generates error). This character cannot be inserted inside the parameter (a*rc generates error).

6.5 Command options

For some commands there are options available. If a command has options, the code '(OPT)' is shown after the prompt that requires the last parameter. The slash (/) separates the option from the command. For example:

FC/C -- FILER COPY with option CONFIRM

Pressing / key, the command associated options are displayed.

To navigate among the various options, use the cursor keys, then select with the **ENTER** key.

If an option that is already displayed is requested, the option is removed. For further information about the options, see the descriptions of the individual commands.

6.6 Viewing and NOPAGE option

The commands to view information direct the output by default to the scrolling window of the system screen on the device used; therefore:

- for the **WinC5G** Terminal, to the window identified by LUN_CRT
- for SYS_CALL, to the window identified by LUN_TP (unless the predefined \$SYS_CALL_OUT variable is changed, a copy of which exists for every PDL2 program)

When the information called by a certain command occupies more than one page, the display is held at the end of the page until the user presses a key (no matter which) to show the next page of information. From WinC5G the lines can be scrolled one at a time by pressing the space bar of the PC keyboard.

Commands that show more than one page of information have the **/NOPAGE** option used to scroll all the information without pauses at the end of the pages.

6.7 Access to the control (login)

To obtain access to the C5G Control Unit from a certain device (Teach Pendant or WinC5G program from PC), a Login is necessary, otherwise the only commands available are those for displays.

A detailed description of the operations needed is contained in [Cap. Access to the Control \(login/logout\) on page 43](#).

6.8 Directories

The directories (up to 8 depth levels) are managed on the C5G Controller. The specification of the directory and the associated device follow the MS-DOS syntax:

<name of device>:\<directory> For example: UD:\directory\filename

The directory and the default device used by the commands are set in a field of the \$DFT_DV variable.

The default directory can be changed using the command Filer Utility Directory Change that only has effect on the device that the command is sent from (teach pendant or Personal Computer with WinC5G active).

Therefore it could happen that on the teach pendant a certain default directory is set, and on the PC it is another, and so the Help shows two different directory contents (on the two different devices).

6.9 Description of commands

The menus are collected in the following branches:

- [CONFIGURE branch](#)
- [DISPLAY branch](#)
- [EXECUTE command \(E\)](#)
- [FILER branch](#)
- [MEMORY branch](#)
- [PROGRAM branch](#)
- [SET branch](#)
- [UTILITY branch](#)

The following table lists all the commands available, in alphabetical order. The string of characters that, in most cases, is shown in brackets, is the code of the first parameter to be used if the corresponding command is to be sent from SYS_CALL.
 If the code is not displayed, the command cannot be called from SYS_CALL.

Command and code for the first SYS_CALL parameter
CONFIGURE ARM CALIBRATE (CAC)
CONFIGURE ARM RETENTIVE MEMORY LOAD (CARL)
CONFIGURE ARM RETENTIVE MEMORY SAVE (CARS)
CONFIGURE ARM TURN_SET (CAT)
CONFIGURE ARM VIEW_CAL (CAV)
CONFIGURE CONTROLLER LOGIN ADD (CCLA)
CONFIGURE CONTROLLER LOGIN DELETE (CCLD)
CONFIGURE CONTROLLER LOGIN GROUP (CCLG)
CONFIGURE CONTROLLER LOGIN MODIFY (CCLM)
CONFIGURE CONTROLLER LOGIN STARTUP (CCLS)
CONFIGURE CONTROLLER LOGIN VIEW (CCLV)
CONFIGURE CONTROLLER RESTART COLD (CCRC)
CONFIGURE CONTROLLER RESTART UPGRADE (CCRU)
CONFIGURE CONTROLLER RESTART SHUTDOWN (CCRS)
CONFIGURE CONTROLLER STARTUP (CCS)
CONFIGURE CONTROLLER TIME (CCT)
CONFIGURE CONTROLLER VIEW (CCV)
CONFIGURE LOAD ALL (CLA)
CONFIGURE LOAD CATEGORY
CONFIGURE LOAD CATEGORY ARM (CLCA)
CONFIGURE LOAD CATEGORY CONTROLLER (CLCC)
CONFIGURE LOAD CATEGORY DSA (CLCD)
CONFIGURE LOAD CATEGORY IO (CLCI)
CONFIGURE LOAD CATEGORY RETENTIVE (CLCR)
CONFIGURE SAVE ALL (CSA)
CONFIGURE SAVE CATEGORY
CONFIGURE SAVE CATEGORY ARM (CSCA)
CONFIGURE SAVE CATEGORY CONTROLLER (CSCC)
CONFIGURE SAVE CATEGORY DSA (CSCD)
CONFIGURE SAVE CATEGORY IO (CSCI)
CONFIGURE SAVE CATEGORY RETENTIVE (CSCR)
DISPLAY ARM CURRENTS (DAC)
DISPLAY ARM DATA (DAD)

Command and code for the first SYS_CALL parameter
DISPLAY ARM FOLLOWING (DAF)
DISPLAY ARM JOINT (DAJ)
DISPLAY ARM POSITION (DAP)
DISPLAY ARM REVOLUTIONS (DAR)
DISPLAY ARM STATUS (DAS)
DISPLAY ARM TEMPERATURE (DAT)
DISPLAY CLOSE ARM (DCA)
DISPLAY CLOSE INPUT (DCI)
DISPLAY CLOSE OUTPUT (DCO)
DISPLAY CLOSE PROGRAM (DCP)
DISPLAY CLOSE RESPLC (DCR)
DISPLAY CLOSE SELECT (DCS)
DISPLAY CLOSE TOTAL (DCT)
DISPLAY CLOSE VARS (DCV)
DISPLAY INPUT AIN (DIA)
DISPLAY INPUT DIN (DID)
DISPLAY INPUT FMI (DIF)
DISPLAY INPUT GI (DIG)
DISPLAY INPUT IN (DII)
DISPLAY INPUT SYSTEM (DIS)
DISPLAY OUTPUT AOUT (DOA)
DISPLAY OUTPUT DOUT (DOD)
DISPLAY OUTPUT FMO (DOF)
DISPLAY OUTPUT GO (DOG)
DISPLAY OUTPUT OUT (DOO)
DISPLAY OUTPUT SYSTEM (DOS)
DISPLAY PROGRAM (DP)
DISPLAY RESPLC (DR)
DISPLAY VARS BIT (DVB)
DISPLAY VARS WORD (DVW)
EXECUTE command (E)
FILER COPY (FC)
FILER DELETE (FD)
FILER EDIT
FILER PRINT (FP)
FILER RENAME (FR)

Command and code for the first SYS_CALL parameter
FILER TRANSLATE (FT)
FILER UTILITY ATTRIBUTE HIDDEN (FUAH)
FILER UTILITY ATTRIBUTE READONLY (FUAR)
FILER UTILITY ATTRIBUTE SYSTEM (FUAS)
FILER UTILITY BACKUP (FUB)
FILER UTILITY RESTORE (FUR)
FILER UTILITY COMPRESS DELETE (FUCD)
FILER UTILITY COMPRESS EXTRACT (FUCE)
FILER UTILITY COMPRESS MAKE (FUCM)
FILER UTILITY COMPRESS VIEW (FUCV)
FILER UTILITY DIRECTORY CHANGE (FUDC)
FILER UTILITY DIRECTORY DELETE (FUDD)
FILER UTILITY DIRECTORY MAKE (FUDM)
FILER UTILITY INSTALL (FUI)
FILER UTILITY PROTECT (FUP)
FILER UTILITY SEARCH (FUS)
FILER VIEW (FV)
MEMORY DEBUG
MEMORY ERASE ALL (MEA)
MEMORY ERASE PROGRAM (MEP)
MEMORY ERASE VARIABLE(MEV)
MEMORY LOAD (ML)
MEMORY SAVE(MS)
MEMORY TEACH
MEMORY VIEW PROGRAM (MVP)
MEMORY VIEW TYPE (MVT)
MEMORY VIEW VARIABLE (MVV)
PROGRAM ACTIVATE (PA)
PROGRAM DEACTIVATE (PD)

Command and code for the first SYS_CALL parameter
PROGRAM EDIT
PROGRAM GO (PG)
PROGRAM ResPLC ACTIVATE (PRA)
PROGRAM ResPLC DEACTIVATE (PRD)
PROGRAM ResPLC RESTORE (PRR)
PROGRAM ResPLC UTILITY PROJDIR (PRUP)
PROGRAM ResPLC UTILITY SAVE (PRUS)
PROGRAM ResPLC UTILITY UNLOAD (PRUU)
PROGRAM ResPLC UTILITY VIEW (PRUV)
PROGRAM STATE BYPASS (PSB)
PROGRAM STATE PAUSE (PSP)
PROGRAM STATE UNPAUSE (PSU)
PROGRAM TEST BREAK INSERT (PTBI)
PROGRAM TEST BREAK PURGE (PTBP)
PROGRAM TEST BREAK VIEW (PTBV)
PROGRAM TEST PROFILE ENABLE (PTPE)
PROGRAM TEST PROFILE DISABLE (PTPD)
PROGRAM TEST PROFILE RESET (PTPR)
PROGRAM TEST PROFILE VIEW (PTPV)
PROGRAM TEST STEP CYCLE (PTSC)
PROGRAM TEST STEP DISABLE (PTSD)
PROGRAM TEST STEP FLY (PTSF)
PROGRAM TEST STEP MOVE (PTSM)
PROGRAM TEST STEP ROUTINE (PTSR)
PROGRAM TEST STEP STATEMENT (PTSS)
PROGRAM TEST STEP VIEW (PTSV)
PROGRAM VIEW (PV)
SET ARM DISABLE (SAD)
SET ARM ENABLE (SAE)
SET ARM GEN_OVR (SAG)
SET ARM NOSTROKE(SAN)
SET ARM SIMULATE (SAS)
SET ARM TP_MAIN (SAT)
SET ARM UNSIMULATE

Command and code for the first SYS_CALL parameter
SET CNTRL KEY_LOCK (SCK)
SET CNTRL LANGUAGE (SCL)
SET CONTROLLER VIEW (SCV)
SET CONTROLLER WIN_CLEAR (SCW)
SET INPUT FORCE AIN (SIFA)
SET INPUT FORCE DIN (SIFD)
SET INPUT FORCE FMI (SIFF)
SET INPUT FORCE IN (SIFI)
SET INPUT REMOTE DISABLE (SIRD)
SET INPUT REMOTE ENABLE (SIRE)
SET INPUT UNFORCE AIN (SIUA)
SET INPUT UNFORCE DIN (SIUD)
SET INPUT UNFORCE FMI (SIUF)
SET INPUT UNFORCE IN (SIUI)
SET INPUT UNFORCE TOTAL (SIUT)
SET INPUT VIEW AIN (SIVA)
SET INPUT VIEW DIN (SIVD)
SET INPUT VIEW FMI (SIVF)
SET INPUT VIEW GIN (SIVG)
SET INPUT VIEW IN (SIVI)
SET INPUT VIEW SYSTEM (SIVS)
SET LOGIN (SL)
SET OUTPUT FORCE AOUT (SOFA)
SET OUTPUT FORCE DOUT (SOFD)
SET OUTPUT FORCE FMO (SOFF)
SET OUTPUT FORCE OUT (SOFO)
SET OUTPUT UNFORCE AOUT (SOUA)
SET OUTPUT UNFORCE DOUT (SOUD)
SET OUTPUT UNFORCE FMO (SOUF)
SET OUTPUT UNFORCE OUT (SOOU)
SET OUTPUT UNFORCE TOTAL (SOUT)
SET OUTPUT VIEW AOUT (SOVA)
SET OUTPUT VIEW DOUT (SOVD)

Command and code for the first SYS_CALL parameter
SET OUTPUT VIEW FMO (SOVF)
SET OUTPUT VIEW GOUT (SOVG)
SET OUTPUT VIEW OUT (SOVO)
SET OUTPUT VIEW SYSTEM (SOVS)
UTILITY APPLICATION (UA)
UTILITY COMMUNICN DISMOUNT (UCD)
UTILITY COMMUNICN MOUNT C5G_Int (UCMC)
UTILITY COMMUNICN MOUNT Modem (UCMM)
UTILITY COMMUNICN MOUNT 3964R (UCM3)
UTILITY COMMUNICN PORT_CHAR (UCP)
UTILITY COMMUNICATION SET_DEF (UCS)
UTILITY COMMUNICATION VIEW (UCV)
UTILITY LOG ACTION (ULA)
UTILITY LOG ERROR (ULE)
UTILITY LOG LATCH ACKNOWLEDGE (ULLA)
UTILITY LOG LATCH VIEW (ULLV)

6.9.1 CONFIGURE branch

A description of these subjects follows:

- [Arm menu](#)
 - [Cntrler Login menu](#)
 - [Cntrler Restart menu](#)
 - [Load menu](#)
 - [Save menu](#)
-

6.9.1.1 Arm menu

The ARM menu displays a menu of commands to configure the arm.

6.9.1.1.1 CONFIGURE ARM CALIBRATE (CAC)

Runs a calibration procedure on the whole arm or on specific axes of the arm. Before the command is sent, the arm must be placed in calibration position. If the arm number is not specified, the default arm indicated on the prompt is used, whereas the number of the axis must always be specified. To indicate a complete arm, use the wildcard (*).

This command is only allowed in PROGR state by Service user profile.

The system has to be in PROG status and the drives must be on. The calibration data is automatically saved in NVRAM, in the configuration file (<\$BOARD_DATA[1].SYS_ID>.C5G) and in the calibration file <\$BOARD_DATA[1].SYS_ID>_CAL<arm_num>.PDL.

Options: **/Learn** stores the current robot position as the user calibration position (\$CAL_USER).

The axis number is asked for, so as to allow the learning of the position (\$CAL_USER) axis by axis.

/Nosave disables the automatic saving of the configuration file, the calibration file and the calibration data in the NVRAM memory;

/User for the user calibration position the system uses that learnt or assigned previously (\$CAL_USER).

Syntax: **CAC <arm_num><axis_num>**
CAC/L <arm_num><axis_num>
CAC/N <arm_num><axis_num>
CAC/U <arm_num><axis_num>

6.9.1.1.2 CONFIGURE ARM RETENTIVE MEMORY LOAD (CARL)

Loads the data into memory that relates to the arm (calibrated axes mask) and to the separate axes (length of axes, calibration values and calibration constants) contained in the NVRAM retentive memory of the controller and automatically updates the configuration file (<\$BOARD_DATA[1].SYS_ID>.C5G) and the calibration file (<\$BOARD_DATA[1].SYS_ID>_CAL<arm_num>.PDL).

This command is only allowed in PROGR or AUTO-T state by Maintenance or Service user profile.

Options:	/Loadfile - data related to the axes are loaded from the calibration file and automatically saved in the configuration file and in the NVRAM memory. The command creates a .COD file with the same name as the calibration file (<\$BOARD_DATA[1].SYS_ID>_CAL<arm_num>.PDL) and executes it. /Nofile disables the automatic saving of the calibration and configuration files
Syntax:	CARL <arm_num> CARL/L <arm_num> CARL/N <arm_num>

6.9.1.1.3 CONFIGURE ARM RETENTIVE MEMORY SAVE (CARS)

SAVE command stores the data regarding the axes from the Controller memory to the NVRAM memory and automatically updates the configuration file <\$BOARD_DATA[1].SYS_ID>.C5G and the ASCII calibration file <\$BOARD_DATA[1].SYS_ID>_CAL<arm_num>.PDL.

This command is allowed by Maintenance or Service user profiles.

Options:	/Nofile disables the automatic saving of the data in the calibration and configuration files.
Syntax:	CARS <arm_num> CARS/N <arm_num>

6.9.1.1.4 CONFIGURE ARM TURN_SET (CAT)

Sets the counting of the encoder turns in the movement software. Before entering the command, the arm must be placed in calibration position (within an encoder dial turn). This command is useful if the pre-set is lost, for example following a motor revolutions reading error.

If the arm number is not specified, the default arm is used, whereas the number of the axis must always be specified. To indicate all the arm axes, use the wildcard (*).

To run this command the system must be in PROG status and power to the drives must be on.

Options:	/User uses the user calibration position previously learnt or assigned.
Syntax:	CAT <arm_num> <axis_num> CAT/A <arm_num> <axis_num> CAT/U <arm_num> <axis_num>

6.9.1.1.5 CONFIGURE ARM VIEW_CAL (CAV)

Displays the calibration constants.

Options:	/4 (see Option /4 to view in the 40 columns)
Syntax:	CAV <arm_num> CAV/4 <arm_num>

6.9.1.2 Cntrler Login menu

This menu allows operation on the controller database relating to the login profiles that can be used on the system.

6.9.1.2.1 CONFIGURE CONTROLLER LOGIN ADD (CCLA)

Adds a Login to the system database. A user identification, the Username and Password have to be specified. It is advised to always specify, using the options associated to the command, the required user profile. Otherwise a default profile will be assigned with a restricted number of operations enabled.

This command is only allowed to a user who has the Admin profile.

Options:

- /Administ** assigns the Administrator level to the login definition
- /Maintain** assigns the Maintenance level to the login definition
- /Programmr** assigns the Programmer level to the login definition
- /Service** assigns the Service level to the login definition
- /Technology** allows access to some functions of the application installed, as a function to the application itself. See the application manual for further details.



While defining a Technology profile, a positive numeric value, otherwise the profile will not be taken into consideration.

Syntax: **CCLA <UserID><Username><Password><PasswordAgain>**
CCLA/Administ <UserID><Username><Password><PasswordAgain>
CCLA/Maintain <UserID><Username><Password><PasswordAgain>
CCLA/Programmr <UserID><Username><Password><PasswordAgain>
CCLA/Service <UserID><Username><Password><PasswordAgain>

6.9.1.2.2 CONFIGURE CONTROLLER LOGIN DELETE (CCLD)

Deletes the user specified as parameter from the Controller database.

This command is only allowed to a user who has the Admin profile.

Syntax: **CCLD <Username>**

6.9.1.2.3 CONFIGURE CONTROLLER LOGIN GROUP (CCLG)

Adds a Login of Group type in the System data base. The user is requested to specify the Group Identifier. It is recommended to always specify, by means of the corresponding options, the wished user profile. If not, a default profile will be assigned, having a restricted number of enabled operations.

This command is only allowed to a user who has the Admin profile.

Options:

- /Administ** assigns the Administrator level to the being defined login
- /Maintain** assigns the Maintenance level to the being defined login
- /Programmr** assigns the Programmer level to the being defined login
- /Service** assigns the Service level to the being defined login

/Technology assigns the Technology level to the being defined login; this profile allows accessing to some functionalities belonging to the installed application, depending on the application itself. Please, refer to the specific application manual for further information.

Syntax:

CCLG <GroupName>
CCLG/A <GroupName>
CCLG/M <GroupName>
CCLG/P <GroupName>
CCLG/S <GroupName>
CCLG/T <GroupName>

6.9.1.2.4 CONFIGURE CONTROLLER LOGIN MODIFY (CCLM)

This command allows to redefine access rights for a certain User or Group. The newly defined settings override the existing ones.

This command also allows to associate a specified User to an already existing Group.

This command is only allowed to a User who has the Admin profile.

If a User belongs to a Group, it is only allowed to redefine the Group rights and not the User rights.

If no options are specified, the assigned rights to the specified User/Group are the ones of the Default profile.

Options:

/Administ assigns the Administrator rights to the specified User/Group
/Group defines a member of the specified Group
/Maintain assigns the Maintenance rights to the specified User/Group
/Programmr assigns the Programmer rights to the specified User/Group
/Service assigns the Service rights to the specified User/Group
/Technology assigns the Technology rights to the specified User/Group

Syntax:

CCLM <IdUtente>
CCLM/A <IdUtente>
CCLM/G <IdUtente><GroupName>
CCLM/P <IdUtente>
CCLM/S <IdUtente>
CCLM/T <IdUtente>

6.9.1.2.5 CONFIGURE CONTROLLER LOGIN STARTUP (CCLS)

Is to be used to have access to the control after a restart (Restart).

This command is only allowed to Programmer, Maintenance and Service users.

Options:

/Reset allows the deletion of the login to be used at the start of the Control. To make this deletion effective, use ConfigureSave (All or Category Controller)

Syntax:

CCLS <UserID> <Password>
CCLS/R <UserID> <Password>

6.9.1.2.6 CONFIGURE CONTROLLER LOGIN VIEW (CCLV)

Displays the Logins that are currently defined in the system. The information given is: Username, the number of active Logins with that Username, and the associated access rights.

Syntax: **CCLV**

6.9.1.3 Cntrler Restart menu

This menu allows the Controller to be restarted in different modes without cutting out the power supply.

A program can be run before the Restart command is completed.

To do this, enter the name of the program to be run in the predefined \$RESTART variable. The \$RESTART_MODE variable contains instead a number that indicates the restart mode in which this program will be considered : 0 for Cold, 1 for shutdown, 2 for reload; if the variable is equal to -1, the program will not be run.

The system waits until the program remains deactivated for a maximum time indicated in the variable \$TUNE[4].

6.9.1.3.1 CONFIGURE CONTROLLER RESTART COLD (CCRC)

Restarts the system software and clears the memory. It is very useful if the system has been altered by the user. the activation of the command is preceded by the display of a confirm prompt. After the command has been executed the system I/Os (\$SDI[n] / \$SDO[n]) are updated to the current situation. For the user I/Os (\$DIN [n] / \$DOUT [n]), the output status is reset and the inputs are updated to the current system situation. The PDL2 holdable/noholdable programs are deactivated and deleted from the execution memory.

Syntax: **CCRC**



Note that, at the end of the Restart operation, the current directory is
UD:\USER

6.9.1.3.2 CONFIGURE CONTROLLER RESTART UPGRADE (CCRU)

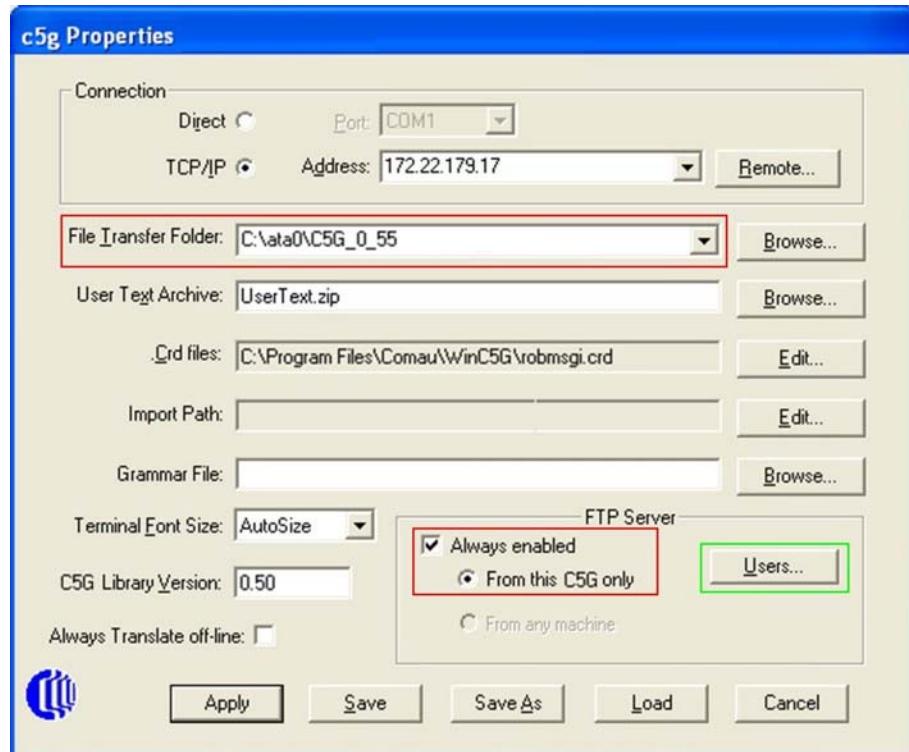
Updates the system software, copying and running the SETUP0.PDL program.



For full details about the software reload complete procedure, please refer to [Cap. Setup page on page 202 - par. 5.20.3.10 Reload-Sw on page 251](#).

Before issuing this command, be sure that all needed settings have been performed in WinC5G [Properties Window](#), shown in the following picture (highlighted in red):

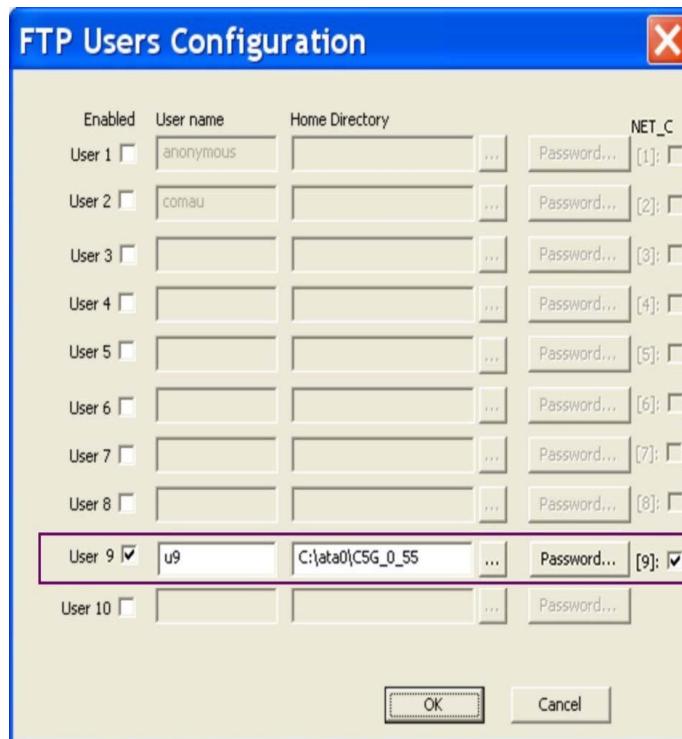
Fig. 6.2 - Properties Window set-up



They are:

- **File Transfer Folder** (it is the directory including the Software version)
- **FTP Server**
 - always enabled
 - from this C5G only

Furthermore, an FTP User must have been configured (**Users...** key in the FTP section - see Fig. 6.2 highlighted in green; needed information: User name and Home Directory, i.e. the **File Transfer Folder** specified in the Properties Window in Fig. 6.2).

Fig. 6.3 - FTP User configuration

The User will be asked for the **User name** (in our example it is 'u9' - see Fig. 6.3, highlighted in purple) during CCRU command execution.

Issuing this command, SETUP0 program (which is in NET0 directory) is activated.

Syntax: **CCRU**

6.9.1.3.3 CONFIGURE CONTROLLER RESTART SHUTDOWN (CCRS)

Before cutting out the main switch of the Controller this command can be entered for the explicit saving of data that is otherwise automatically saved by the system if there is a voltage drop. This avoids using the battery to power the system while saving the data, thus lengthening the battery life. The command is useful for those who usually shut down the control at set times (for example at the end of the day).

Syntax: **CCRS**

6.9.1.4 CONFIGURE CONTROLLER STARTUP (CCS)

Sets the name of the program that is run automatically at every controller restart. This program, referred to as "startup", may be either holdable or non holdable type; the execution of a holdable program only starts when the START button is pressed. Startup program name is stored in the \$STARTUP variable. If this command is sent without specifying the name of the program, the \$STARTUP variable is assigned a null string.

Syntax: **CCS <program_name>**

6.9.1.5 CONFIGURE CONTROLLER TIME (CCT)

Sets the time (HH:MM:SS) and the date (DD-MM-YY) according to the specified parameters. The month is indicated with a numeric value (1-12). This command can be

used by a user with maintenance and service profile.

Syntax: **CCT <hour> <date>**

Example: **CCT 13:00:00 01-01-97** —13hours (1 p.m.) of January 1st

6.9.1.6 CONFIGURE CONTROLLER VIEW (CCV)

Displays:

- the system software version both for the Controller and the Teach Pendant
- the Controller ID
- type of machine, name and number of axes for each arm
- date and time.

The screen page is the same that appears when the control is switched on.

Options: **/4** (see [Option /4 to view in the 40 columns](#))

Syntax: **CCV**

6.9.1.7 Load menu

6.9.1.7.1 CONFIGURE LOAD ALL (CLA)

Assigns to the predefined variables in memory the values stored in both the Configuration file and User Data Base file that may be the default ones, <\$SYS_ID>.C5G and <\$SYS_ID>.UDB, or files specified in the associated parameter. If no parameter is specified, the values will be read from the <\$SYS_ID>.C5G and <\$SYS_ID>.UDB files and assigned to the predefined variables in memory.

The system is to be in PROGR state with the drives off. The user profile can only be Maintenance and Service.

From SYS_CALL the parameter 'CL', that previously corresponded to ConfigureLoad, is interpreted as 'CLA' (to maintain the compatibility with the old PDL2 programs written on the C3G controller).

Options: **/Rules - \$C5G_RULES**, manual **PDL2 Programming Language**, chap. **Predefined Variables List** -

Syntax: **CLA <file_name>**

CLA/R <file_name> <rules>

6.9.1.7.2 CONFIGURE LOAD CATEGORY

Assigns the values stored in the configuration file (<\$BOARD_DATA[1].SYS_ID>.C5G) or in the file specified as parameter to this command and previously created through a [CONFIGURE SAVE CATEGORY](#) to the predefined variable belonging to the category indicated. The file, if specified, is searched for in the currently selected directory. Otherwise the search is made in the system directory (UD:\SYS). The variables that do not belong to the indicated category are ignored.

Syntax: **CLCx <filename>** --where x is the Load Category ,

**MINOR CATEGORY**

Besides the Load Category, available as COMMAND, there is a Minor Category, available as OPTION for each of the predefined variables.

In the paragraphs that follow, for each ConfigureLoadCategory and ConfigureSaveCategory command, the Minor Categories provided are listed.

For further information, and above all, to know whether a certain variable will be loaded or saved sending a ConfigureLoadCategory or ConfigureSaveCategory command, see the [PDL2 Programming Language](#) manual, chapter [Predefined Variables List](#), item [Minor Category](#) (minor category of [Load Category](#)) in the table regarding the variable involved.

For example:

\$SENSOR_ENBL belongs to the Arm Load Category) and to the Sensor Minor Category. Therefore reference will be made to the ConfigureSaveCategoryArm and ConfigureSaveCategoryArm/Sensor commands (or ConfigureLoadCategoryArm and ConfigureLoadCategoryArm/Sensor).

CONFIGURE LOAD CATEGORY ARM (CLCA)

Load the variables that have Arm as Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

- /Auxiliary
- /Calibrate
- /Collision
- /Configure
- /Conveyor
- /Chain
- /Rules - \$C5G_RULES, manual [PDL2 Programming Language](#), chap. [Predefined Variables List](#) -
- /Sensor

Syntax: **CLCA <file_name>**

CONFIGURE LOAD CATEGORY CONTROLLER (CLCC)

Loads the variables that have Cntrl as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

- /Conveyor
- /Environ
- /Flow
- /Parameter
- /Rules - \$C5G_RULES, manual [PDL2 Programming Language](#), chap. [Predefined Variables List](#) -
- /Shared
- /Unique
- /Vars

Syntax: **CLCC <file_name>**

CONFIGURE LOAD CATEGORY DSA (CLCD)

Loads the variables that have Dsa as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. [Predefined Variables List](#) -

Syntax:

CLCD <file_name>

CLCD/R <file_name> <rules>

CONFIGURE LOAD CATEGORY IO (CLCI)

Loads the variables that have Input/output as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Applicatn
/Configure
/Fieldbus
/Hand
/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. [Predefined Variables List](#) -
/Swim

Syntax:

CLCI <file_name>

CONFIGURE LOAD CATEGORY RETENTIVE (CLCR)

Loads the variables that have Retentive as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. [Predefined Variables List](#) -

Syntax:

CLCR <file_name>

CLCR/R <file_name> <rules>

6.9.1.8 Save menu

6.9.1.8.1 CONFIGURE SAVE ALL (CSA)

Creates both a new Configuration file with the predefined variable values according to the data currently in the system memory, and a new UDB file including the System User Data Base.

The file names and extensions are, by default, <\$SYS_ID>.C5G and <\$SYS_ID>.UDB and the files are saved in the UD:\SYS system directory .If a different filename is specified, this is stored by default in the current directory.

From SYS_CALL the 'CS' parameter, that in the past corresponded to ConfigureSave, is interpreted as 'CSA' (to maintain compatibility with old PDL2 programs written on C3G).

Syntax: **CSA <file_name>**

6.9.1.8.2 CONFIGURE SAVE CATEGORY

To save in a file the stored values of predefined variables belonging to the indicated category.

The user may need to:

- Create a file containing ONLY the values of variables of the indicated category. This file will be used for partial loading of the values in the memory ([CONFIGURE LOAD CATEGORY](#)) or it will be copied on several controls to unify certain initializations. The name of the file is to be specified in the command and it is to be different from the name of the <\$SYS_ID>.C5G Configuration file. The directory, if not specified, will be the one currently selected on the device the command is sent from and the file extension will be .C5G.
- Save data of the category indicated in the global Configuration file (<\$SYS_ID>.C5G). In this case only the section containing the variables of the pre-selected category will be updated; the values of the other variables in the file will remain unaltered. To do this, the command must be sent without specifying the name of the file.

Syntax: **CSCx <filename>** --where x is the Load Category



See [MINOR CATEGORY](#) note.



The ConfigureSaveCategory command, if executed on a .C5G file created with a software version prior to that loaded, is allowed, providing the variables of the category involved have not changed in either number or size.

CONFIGURE SAVE CATEGORY ARM (CSCA)

Saves the values of the variables that have Arm as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

- /Auxiliary
- /Calibrate
- /Collision
- /Configure
- /Conveyor
- /Chain
- /Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. Predefined Variables List -
- /Sensor

Syntax: **CSCA <file_name>**

CONFIGURE SAVE CATEGORY CONTROLLER (CSCC)

Saves the values of the variables that have Cntrl as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Conveyor
/Environ
/Flow
/Parameter
/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. Predefined Variables List -
/Shared
/Unique
/Vars

Syntax: **CSCC <file_name>**

CONFIGURE SAVE CATEGORY DSA (CSCD)

Saves the values of the variables that have Dsa as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. Predefined Variables List -

Syntax: **CSCD <file_name>**

CSCD/R <file_name> <rules>

CONFIGURE SAVE CATEGORY IO (CSCI)

Saves the values of the variables that have Input/output as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Options:

/Applicatn
/Configure
/Fieldbus
/Hand
/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. Predefined Variables List -
/Swim

Syntax: **CSCI <file_name>**

CONFIGURE SAVE CATEGORY RETENTIVE (CSCR)

Saves the values of the variables that have Retentive as the Load Category. See previous note regarding [MINOR CATEGORY](#).

Opzioni:

/Rules - \$C5G_RULES, manual PDL2 Programming Language, chap. Predefined Variables List -

Syntax: **CSCR <file_name>**

CSCR/R <file_name> <rules>

6.9.2 DISPLAY branch

The DISPLAY branch displays a menu of commands that open viewing windows with information that is continually updated until the DisplayClose command is sent.

Several windows can be opened at the same time.

In the case of SYS_CALL the DISPLAY commands direct the display by default to the Terminal window shown by the **WinC5G** program (if connected).

A description of these subjects follows:

- [Arm menu](#)
 - [Close menu](#)
 - [Input menu](#)
 - [Output menu](#)
 - [Vars menu](#)
-

6.9.2.1 **Arm menu**

The ARM menu contains a menu of commands for the display of information regarding the specified arm.

6.9.2.1.1 **DISPLAY ARM CURRENTS (DAC)**

Displays the currents of the axis motors in amperes.

Syntax: **DAC <arm_num>**

6.9.2.1.2 **DISPLAY ARM DATA (DAD)**

Displays the data regarding the arm interpolator.

Syntax: **DAD <arm_num>**

6.9.2.1.3 **DISPLAY ARM FOLLOWING (DAF)**

Displays the axis following errors in bit units.

Syntax: **DAF <arm_num>**

6.9.2.1.4 **DISPLAY ARM JOINT (DAJ)**

Displays the current position target of each axis, in degrees.

Syntax: **DAJ <arm_num>**

6.9.2.1.5 **DISPLAY ARM POSITION (DAP)**

Displays the current Cartesian position target, indicating the x,y,z location, the Eulero angles and the configuration string; the values of the joints (in mm, for translating axes, or in degrees for rotating axes) of the auxiliary component in a system with auxiliary axes.

Syntax: **DAP <arm_num>**

6.9.2.1.6 **DISPLAY ARM REVOLUTIONS (DAR)**

Displays the number of motor revolutions.

Syntax: **DAR <arm_num>**

6.9.2.1.7 DISPLAY ARM STATUS (DAS)

Displays the current status of the arm (calibrated, in lock, etc.), the number of joints that form the arm, the co-operative movement and the state of the auxiliary axes (connected, enabled, disabled). It is also indicated whether the robot is along the programmed path (On Trajectory).

Syntax: **DAS <arm_num>**

6.9.2.1.8 DISPLAY ARM TEMPERATURE (DAT)

Displays the temperatures of the axis motors as percentage of the maximum temperature allowed for the motor.

Syntax: **DAT <arm_num>**

6.9.2.2 Close menu

The CLOSE menu removes the DISPLAY windows from the screen.

6.9.2.2.1 DISPLAY CLOSE ARM (DCA)

Closes all the windows opened with the DISPLAY ARM command.

Syntax: **DCA**

6.9.2.2.2 DISPLAY CLOSE INPUT (DCI)

Closes all the windows relating to the inputs.

Syntax: **DCI**

6.9.2.2.3 DISPLAY CLOSE OUTPUT (DCO)

Closes all the windows relating to the outputs.

Syntax: **DCO**

6.9.2.2.4 DISPLAY CLOSE PROGRAM (DCP)

Closes the window related to the active PDL2 programs.

Syntax: **DCP**

6.9.2.2.5 DISPLAY CLOSE RESPLC (DCR)

Closes the DISPLAY window related to the PLC environment.

Syntax: **DCR**

6.9.2.2.6 DISPLAY CLOSE SELECT (DCS)

Allows the operator to select the display window to be closed.

Syntax: **DCS**

6.9.2.2.7 **DISPLAY CLOSE TOTAL (DCT)**

Closes all the windows opened by DISPLAY commands.

Syntax: **DCT**

6.9.2.2.8 **DISPLAY CLOSE VARS (DCV)**

Closes all the windows opened with the DISPLAY VAR commands.

Syntax: **DCV**

6.9.2.3 **Input menu**

With the INPUT menu the state of the inputs can be viewed, both those defined by the user and those of the system .

A text string, set by the user, can be associated to the displayed Inputs. To assign such descriptions to the Input points, use the [IO_MAP Program - I/O ports mapping, par. 11.4.2.1.1 Map or Modify - I/O points on page 469](#). The System Inputs HELP strings are always present.

6.9.2.3.1 **DISPLAY INPUT AIN (DIA)**

Displays the values of the analog inputs defined by the user (\$AIN), in decimal format. If the value is forced, it is shown in red instead of blue on the WinC5G screen.

Several indexes can be specified in the parameter for the command: to indicate different indexes, use dashes (-) to separate them; to indicate an interval between indexes, use two dots (...).

Syntax: **DIA <string_of_numbers>**

Example: **DIA 1**
DIA 1- 2 - 4..7

6.9.2.3.2 **DISPLAY INPUT DIN (DID)**

Displays the values of the digital inputs defined by the user (\$DIN), where 1 = TRUE and 0 = FALSE. If one of these values is forced, an "F" (forced to false) or a "T" (forced to true) is shown instead of 0 or 1.

If a number of input is typed, a series of digital inputs is displayed.

Syntax: **DID**

Example: **DID 1**

6.9.2.3.3 **DISPLAY INPUT FMI (DIF)**

Displays the values of flexible multiple inputs (\$FMI) defined by the user, in hexadecimal format.

Several indexes can be specified in the parameter for the command: to indicate different indexes, use dashes (-) to separate them; to indicate an interval between indexes, use two dots (...).

Syntax: **DIF <indexes>**

Example: **DIF 1- 2 - 4..7**

6.9.2.3.4 DISPLAY INPUT GI (DIG)

Displays the values of general system inputs (\$GI). To move with the cursor along the bits use the **LH** and **RH** arrow keys, to change module use the **UP** and **DOWN** arrow keys.

Syntax: **DIG**

6.9.2.3.5 DISPLAY INPUT IN (DII)

Displays the values of the privileged digital inputs defined by the user (\$IN), where 1 = TRUE and 0 = FALSE. If one of these values is forced, an "F" (forced to false) or a "T" (forced to true) is shown instead of 0 or 1.

If a number of input is typed, a series of digital inputs is displayed.

Syntax: **DII**

Example: **DII 1**

6.9.2.3.6 DISPLAY INPUT SYSTEM (DIS)

Displays the digital inputs defined in the system (\$SDI). **SEL** key works exactly like **DISPLAY INPUT GI (DIG)** command.

Syntax: **DIS**

6.9.2.4 Output menu

The OUTPUT menu displays the state of the outputs defined by the user and of the system.

A text string, set by the user, can be associated to the displayed Outputs. To assign such descriptions to the Output points, use the [IO_MAP Program - I/O ports mapping, par. 11.4.2.1.1 Map or Modify - I/O points on page 469](#). The System Outputs HELP strings are always present.

6.9.2.4.1 DISPLAY OUTPUT AOUT (DOA)

Displays the values of the analog outputs (\$AOUT), in decimal format. Forced values, are shown in red on the WinC5G screen.

Several indexes can be specified in the parameter for the command: to indicate different indexes, use dashes (-) to separate them; to indicate an interval between indexes, use two dots (...).

Syntax: **DOA <string_of_numbers>**

Example: **DOA 17**
DOA 17-18-25..27

6.9.2.4.2 DISPLAY OUTPUT DOUT (DOD)

Displays the values of the digital outputs defined by the user (\$DOUT), where 1 = TRUE and 0 = FALSE. If one of these values is forced, an "F" (forced to false) or a "T" (forced to true) is displayed instead of 0 or 1.

When a number of output is typed, a set of digital outputs is displayed.

Syntax: **DOD**
 Example: **DOD 17**

6.9.2.4.3 DISPLAY OUTPUT FMO (DOF)

Displays the values of flexible multiple outputs (\$FMO) defined by the user, in hexadecimal format.

Several indexes can be specified in the parameter for the command: to indicate different indexes, use dashes (-) to separate them; to indicate an interval between indexes, use two dots (..).

Syntax: **DOF <indexes>**
 Example: **DOF 17-18-25..27**

6.9.2.4.4 DISPLAY OUTPUT GO (DOG)

Displays the values of general system outputs (\$GO). **SEL** key works exactly like [DISPLAY INPUT GI \(DIG\)](#) command.

Syntax: **DOG**

6.9.2.4.5 DISPLAY OUTPUT OUT (DOO)

Displays the values of the privileged digital outputs defined by the user (\$OUT), where 1 = TRUE and 0 = FALSE. If one of these values is forced, an "F" (forced to false) or a "T" (forced to true) is shown, instead of 0 or 1. If a number of output is typed a series of digital outputs is displayed.

Syntax: **DOO**
 Example: **DOO 17**

6.9.2.4.6 DISPLAY OUTPUT SYSTEM (DOS)

Displays the digital outputs defined by the system (\$SSDO). **SEL** key works exactly like [DISPLAY INPUT GI \(DIG\)](#) command.

Syntax: **DOS**

6.9.2.5 DISPLAY PROGRAM (DP)

PROGRAM displays information about the active PDL2 programs.

A particular program name can be specified. It is also possible to insert a wildcard in the program name. If no parameter is given, the information is displayed regarding all the activated programs.

The information shown is the same as for the [PROGRAM VIEW \(PV\)](#) command.

Options: **/Full** will display also all the programs with protected code, that are not usually displayed.

Syntax: **DP <program_name>**

6.9.2.6 DISPLAY RESPLC (DR)

The information related to PLC resource are shown:

- Resource name
- Resource identifier
- Cycle time (ms)
- Number of performed cycles
- Time for executing the latest cycle
- Maximum time for executing a scanning
- State of the resource (Running, Stop, etc.)

Syntax: **DR**

6.9.2.7 Vars menu

The VARS menu contains the commands to display current values of the memory arrays commonly used by programs that use the \$BIT and \$WORD variables.

6.9.2.7.1 DISPLAY VARS BIT (DVB)

Displays \$BIT, where 1 = TRUE and 0 = FALSE, according to 2 different displays:

Several indexes can be specified in the parameter for the command:

- To indicate different indexes, separate them with dashes (-);
The maximum number of \$BIT that can be displayed at the same time is 4 (for the Teach Pendant) and 11 (for WINC5G).
For example: "DISPLAY VAR BIT" "1-35-100" displays the value of \$BIT[1], \$BIT[35] and \$BIT[100].
- To indicate an interval between indexes, use two dots (..).A maximum of 16 bits (on Teach Pendant) and 32 bits (for WINC5G) can be displayed at the same time.
"DISPLAY VAR BIT" "1..15" displays all the \$BIT having an index from 1 to 15.

Syntax: **DVB**

6.9.2.7.2 DISPLAY VARS WORD (DVW)

Displays the value of a \$WORD in decimal format.

Several indexes can be specified in the parameter for the command: to indicate different indexes, use dashes (-) to separate them; to indicate an interval between indexes, use two dots (..).

Options: **/Hex** displays the data in hexadecimal format
/Binary displays the data in binary format

Syntax: **DVW <string_of_numbers>**

6.9.3 EXECUTE command (E)

With the EXECUTE command a PDL2 instruction can be executed from commands level. This type of execution is also called immediate execution, because the instructions have to be short and not locking (for example to assign a value to a

predefined variable).

The first parameter is the instruction to be executed. If the instruction refers to program variables, the name of the program they belong to has to be specified in the second parameter.

<i>Options:</i>	/Context is available in the prompt of the second parameter. It is specified ONLY when the instruction contains references to predefined variables (preceded by \$, such as \$CYCLE) that belong to the "Program Stack" category (see PDL2 Programming Language Manual , chapter PREDEFINED VARIABLES LIST); the /Context option indicates that the predefined variable will be that of the program indicated as second parameter. If this option is not used, the program stack type predefined variables will be those of the virtual EXECUTE program, that only have life for the duration of the command. When this option is used the program specified as parameter must be active, otherwise an error will be generated.
-----------------	---

<i>Syntax:</i>	E instruction <program_name>
<i>Example:</i>	<p>E 'I:=10' 'pioppo' - Assigns the value 10 to variable i of program pippop. The program pippop must be loaded in memory.</p> <p>E 'WRITE(\$ERROR,NL)' 'pioppo' — Value 0 will be shown</p> <p>E/C 'WRITE(\$ERROR, NL)' 'pioppo' — The value of pippop \$ERROR will be shown</p>

The following instructions cannot be used with the EXECUTE command:

- WAIT
- CONDITION
- ENABLE CONDITION
- DISABLE CONDITION
- PURGE CONDITION

The program variables cannot be declared and created with this command.

The execution can be deactivated by pressing ^C.

To execute instructions that require movements, the following rules are valid:

- the Robot Controller Cabinet status selector switch is to be set for programming
- all movements are executed at safe reduced speed
- the enabling device is to be kept pressed during the movement execution
- **START** is to be kept pressed to maintain the movement
- press **HOLD** to interrupt the movement (this movement can be resumed by pressing **START**)

6.9.4 FILER branch

The FILER branch displays a menu of file management commands.

For most FILER commands it is necessary to indicate as parameters the file names, devices, name and extension:

dv:file_name.ext

The name of the device can be formed by one to four characters followed by the colon (:) symbol. Some commands are only operative with certain devices. For further details, see the individual commands.

The filename can be from 1 to 256 characters, that may be letters, digits and subscripts (_). However, if the file is to contain a program name, the maximum size for the name is 32 characters.

The extension starts with a dot (.) and consists of three characters that can be letters, digits and subscripts (_).



For a complete list of foreseen system software extensions, see [par. 6.10 Types of files available in the System on page 363](#).

Some commands are only operative on certain devices. For details, see the individual commands.



NOTE

In the UD root directory : there is a limitation to the maximum number of entries (512). A file name longer than 8 characters occupies more than one entry in the FAT (File Allocation Table).

The problem does NOT exist in subdirectories, where the maximum entry limit is higher (more than 65000).

If the root directory is too much occupied, it is advised to use file names with a maximum of 8 characters, or to work in subdirectory.

Furthermore, it is advised to periodically delete the .BK* files in a permanent manner.

Besides the individual commands, the following are also described:

- [UTILITY ATTRIBUTE menu](#)
- [Backup and Restore commands](#)
- [Compressed files management commands](#)
- [Directory management commands](#)

6.9.4.1 FILER COPY (FC)

COPY command creates copies of files. No change is made to the original file.

Specify the source file and the destination file. For the extensions and the file names a wildcard can be used.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

It is advised to use FILER UTILITY BACKUP and FILER UTILITY RESTORE (instead of FILER COPY) if all the files from/to the default backup device (\$DFT_DV[3], usually XD: or COMP:) are to be copied.

Options: **/Confirm** requires confirmation before copying each file.
/Overwrite rewrites the existing files without displaying the confirmation prompt.

Syntax: **FC <source_file> <dest_file>**

6.9.4.2 FILER DELETE (FD)

Removes the files from the user device.

It is asked to specify the files. For the extensions and the filenames the wildcard can be used. If no device is specified, the default device is used.

The user is asked for confirmation before each file is deleted.

If the command is sent without the /Permanent or /Cleanup option, the files are not permanently deleted and can therefore be recovered with the /Undelete option.

Options:

- /Cleanup empties the bin containing the previously deleted files
- /Noconfirm does not ask the user for confirmation before deletion.
- /Permanent deletes the files permanently.
- /Undelete allows the recovery of a file that has been deleted (providing the /Permanent option has not been used when deleting the file).

Syntax: **FD <file_name>**

6.9.4.3 FILER EDIT

The EDIT command calls the ASCII editor of the files. This can be used to create or to edit ASCII files on the user disk (UD:).



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

A parameter is required for the filename. The wildcard cannot be used. Only the user disk UD: can be used as device. If no extension is specified, the extension .PDL is used. The command cannot be used if any other type of Editor is active. This command is not allowed from SYS_CALL.

Syntax: **FE <file_name>**

6.9.4.4 FILER PRINT (FP)

To display the contents of a file on the screen (WinC5G Terminal) or on another file.

The destination device may consist of the Teach Pendant monitor or the PC, a file, communication port or a window defined by the user. If no destination device is specified, the device from which the command has been sent will be used.

The name of the start file is required as parameter. For the extension and file name the asterisk can be used. If no device is specified, the default device is used (\$DFT_DV[2]).

The Filer Print command is used to view the contents of binary files: .COD or .BCK program files; .VAR or .BKV variable files; files in the system directories (.C5G configuration files, .LBE error log files, .LBA action log files). It is assumed that all the other files are in ASCII format.

The command is also a means to convert binary files to the corresponding ASCII format.

In the case of a Filer Print command for a .VAR file to an .LSV file or for a .COD to a .PDL, a back-up file is created with extension .BKL and .BKP respectively for that .LSV file. If no destination device is specified, the file is printed on the video.

If a .COD file contains checksum errors or other damaged data and it is not possible to recover a backup copy of the same file, with FILER PRINT the program can be converted to an ASCII file to save in the user disk (UD:).



See **NOTE** regarding the total amount of entries in the UD: root directory.

The ASCII file can then be edited to correct the instructions and corrupted identifiers that are distinguished by three question-marks ("??"). At the end, use FILER TRANSLATE to convert the file again to a file with .COD extension.

As from system software version 3.20, further information can be printed, using the option /Property. Standard attributes (Property) associated to the files are the following:

- drawn up by
- file creation date
- version
- host where file was created.

Options: **/Full** prints a variables file without shortening the names of the identifiers contained (variables, programs, routines and data classes).

/List generates a listing file (.LIS) from a .COD or .BKC file. The listing file gives the name of the program and the line number at the side of each PDL2 instruction.

/Overwrite Overwrite rewrites the old copy of the corresponding file without asking for confirmation of the user.

/Nopage

/Range to print the lines of a file between the serial and final line specified as additional parameters.

/Property to print the file properties (drawn up by, date of creation, version, device where it was created).

Syntax:

```
FP <file_name> <dest_device>
FP/L <file_name> <dest_device>
FP/N <file_name> <dest_device>
FP/R <file_name> <dest_device> <start_line><end_line>
FP/P <file_name> <dest_device>
```

6.9.4.5 FILER RENAME (FR)

RENAME changes the name and/or extension of an existing file.

It is necessary to specify as parameters the original file (source) and the new file (destination). A wildcard can be used for the extension. If no device is specified for the source file, the default device is used. It is not possible to specify different devices for the source file and the destination file.

If a new file already exists, before the original file is renamed, a prompt is displayed asking for confirmation.

Renaming of a program file (.COD or .BKC) is not allowed if the new name is present as identifier inside the file. This type of error is only detected when the program is loaded.

Options: **/Confirm** displays a confirm prompt before each file is renamed.
/Overwrite rewrites the existing files without displaying any confirm prompt.

Syntax: **FR <source_file> <dest_file>**

6.9.4.6 FILER TRANSLATE (FT)

TRANSLATE command converts:

- a PDL2 program, from ASCII source code (.PDL) to an executable code (.COD) and vice versa
- a variables file in source ASCII format (.LSV), to a variables file in format to be loaded on the Controller (.VAR) and vice versa.
- system binary files (.C5G, .LBE, .LBA) in the corresponding ASCII format (.PDL, .LOG, .ACT).

The name of the source code file is required. In the name of the start file the asterisk can be used. As device it is possible to specify only the user disk UD:.



See **NOTE** regarding the total amount of entries in the UD: root directory.

If the extension is not given, the .PDL extension is used.

If the destination file (for example .COD) already exists, the back-up copy is saved changing the extension (for example .BKC) before the creation of the new file.

If indications or errors are found during the conversion, these are displayed in the scrolling window and recorded in the <file_name>.ERR file.

Options: **/Back** converts a .COD file to a .PDL file. It is the same as the Filer Print command of a .COD file to a .PDL.
/Confirm asks the user for confirmation before the conversion of each file.
/List generates a listing file (.LIS) and can only be used with the /BACK option.
/Variables converts a variables .LSV ASCII file in a .VAR file. When used with the /BACK option, it converts a .VAR variables file to the corresponding .LSV with ASCII format.
/Output redirects the file on another directory.

Syntax: **FT <file_name>**

6.9.4.7 UTILITY ATTRIBUTE menu

Contains the commands to change the file attributes. Access is allowed to "Maintenance" or "Service" user profile.

6.9.4.7.1 FILER UTILITY ATTRIBUTE ARCHIVE (FUAA)

To set/remove the attribute *archived ('a')* to a file.

Options: **/Reset** deletes attribute *archived ('a')* to the specific file.
Syntax: **FUAA <file_name>**
 FUAA/R <file_name>

6.9.4.7.2 FILER UTILITY ATTRIBUTE HIDDEN (FUAH)

Sets the “hidden” attribute to a file that therefore will not be displayed on the user disk UD:. With this attribute set, the file, if a .COD, can however be loaded in the execution memory (Memory Load).

Options: **/Reset** deletes the hidden attribute.
Syntax: **FUAH <file_name>**
 FUAH/R <file_name>

6.9.4.7.3 FILER UTILITY ATTRIBUTE READONLY (FUAR)

Sets the “read only” attribute to a file. When this attribute is set, the file cannot be edited or deleted.

Options: **/Reset** deletes the read only attribute.
Syntax: **FUAR <file_name>**
 FUAR/R <file_name>

6.9.4.7.4 FILER UTILITY ATTRIBUTE SYSTEM (FUAS)

Sets the system attribute for a file. When set, this attribute prevents file deletion.

Options: **/Reset** deletes the system attribute.
Syntax: **FUAS <file_name>**
 FUAS/R <file_name>

6.9.4.8 Backup and Restore commands

For better understanding the difference between a simple file copy ([FILER COPY \(FC\)](#)) and a backup/restore operation properly organized, follow the listed below tips:

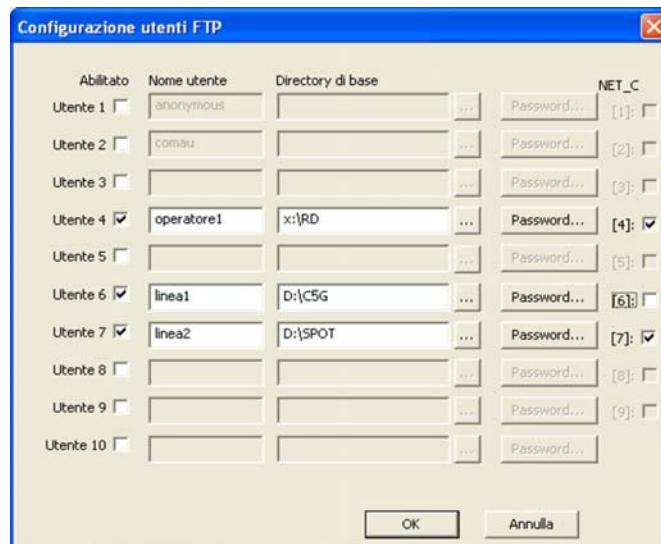
- always make sure that execution memory and UD: device are aligned (there shouldn't be any not saved programs or data), before starting any Backup or Restore operation;
- always perform a ConfigureSaveAll command ([Save in the Setup page](#)) before any Backup operation and a ConfigureLoadAll command ([Load in the Setup page](#)) after a Restore operation to restore the same work conditions, based on .C5G file, after a Restart;
- plan periodic Backup operations, better if during the night;
- possibly use the .LST file created during the Backup operations, when it is useful to compare several backups with each other to see possible differences;
- compress the Backup in a.ZIP file. Execute the Restore decompressing the.ZIP file.

If WinC5G acts as server for the Controller, the configuration of the backup and restore operations requires that the NETx: devices are set correctly. This can be done by entering sub-page [FTP Users Configuration](#). The system allows definition of a

Username and a Password for the user requesting the operation, for each NETx: device, and the base Directory for backup and restore operations. When these operations are required, the specified user has to be already defined and logged on the Controller.

The NET_C checkbox indicates that the parameters associated to it will be used to automatically configure the different \$NET_C system variables (see [PDL2 Programming Language](#) manual - chap.[Predefined Variables List](#)). Therefore it is not necessary that the user sets them.

Fig. 6.4 - FTP Users Configuration





NOTES FOR BACKUP / RESTORE

As from system software version 3.20, it is possible to obtain the backup and restore of files contained in a certain device (UD:, LD:, VD:, TD:), keeping the original structure of the directories and selecting them according to:

- either a date or a number of days,
- whether they have already been saved in a backup or not,
- whether they are newer,
- whether they belong to a certain directory or Saveset.

Note that the backup operation should preferably be executed when NO technological processes are running, when motor drives are off (DRIVE-OFF) and when the system is in programming status.

The backup can be sent to a USB flash disk (XD: or TX:), including subdirectories, COMP: (device WinC5G), NETx: (device client FTP), TD: .

At each backup (e.g. FUB/T * NET8), a <\$SYS_ID>.LST file is created on the specified device, in the specified base directory, containing the information related to the backup operation.

Information regarding several backups is inserted at the bottom of the file. At each backup operation, these files are updated.

Example:

```
#####
# Backup created 05-MAY-08 16:17:00
# Backup appended 05-MAY-08 16:44:40
# Backup appended 05-MAY-08 16:48:57
# Backup appended 05-MAY-08 16:49:55
UD:\box.pdl@02-04-07 11:24:08
UD:\brac_7ax1.cod@14-03-07 14:32:48
UD:\brac_7ax1.pdl@14-03-07 14:32:44
UD:\brev_002_r2.cod@13-11-07 10:18:16
UD:\brev_002_r2.lsv@19-11-07 13:34:44
UD:\brev_002_r2.pdl@12-11-07 18:54:44
UD:\brev_002_r2.var@16-11-07 11:30:20
```

When a file is copied in a backup, it is "marked" by attribute 'a'.

To be able to use these functions, there are some options described in the following sections (see [par. 6.9.4.8.1 FILER UTILITY BACKUP \(FUB\) on page 331](#) and [par. 6.9.4.8.2 FILER UTILITY RESTORE \(FUR\) on page 333](#)):

- For Backup operation - /Saveset, /After, /Increment, /To
- For Restore operation - /Saveset, /After, /From, /Newer, /Overwrite.

6.9.4.8.1 FILER UTILITY BACKUP (FUB)

Copies the files from the directory selected when executing the command to the default backup device (\$DFT_DV[3]). If no file is specified, all the files of the current directory are copied.

Options:

/Saveset is used to specify a predefined saveset that indicates where the files are to be copied from. The saveset must already been created.
If, for example, the use of a Saveset referred to as '**set1**'

UD:\dir1*.*\s is required, the Backup command must be issued by specifying 'set1' as parameter. In such a way, all the files will be copied that are contained in **UD:\dir1** and the associated subdirectories.

/After copies files that are more recent than a certain date.

To specify date or count of days a sign can be indicated that has the meaning of "before" ('-') or "after" the indicated date or count of days. INTEGER to indicate the total amount of days BEFORE today; DATE format with sign to indicate BEFORE or AFTER the specified date

Examples:

-1 means all files before 00 of today;

-18-06-08 means files before 18/06/08;

18-06-08 means files EQUAL and AFTER that date.

/Incremental copies all files that have NOT been saved in previous Backups (the system uses the 'a' attribute that indicates "file already archived").

/Incremental all files which ave not been saved in previous backup operations are copied (the system uses 'a' attribute which specifies whether or not the file has already been previously archived).

/To copies on a device that is not the default (\$DFT_DV[3]). The required directory can be specified on this device.

Syntax:

FUB <file_name>
FUB/S <saveset_name>
FUB/A <date_or_days>
FUB/I
FUB/T <device_name>

Examples:

FUB * XD:r026 copies in subdirectory r026 of device XD:

FUB/A0 executes backup of today's files

FUB/A-1 executes backup of files from yesterday

FUB/I * executes backup of files not yet saved in a previous backup

FUB/T * NET8: executes backup on another device

Using Savesets in the Backup command



As already said describing the /Saveset option, it is possible to specify a Saveset name instead of a filename (in such a case, when issuing the command, the /Saveset option must be inserted).

The specified Saveset must have already been created in the corresponding **Backup** subpage of [Setup page](#).

In this way, using a single command it is possible to execute the backup of files from different devices or directories, files with a certain name, include the subdirectories, exclude some files, etc.

The following switches are provided to properly setup the Saveset:

/A (=After) to execute the backup of created/modified files BEFORE or AFTER a specified date - INTEGER to indicate the total amount of days BEFORE today; DATE format with sign: the

negative sign means BEFORE the specified date, the positive sign means AFTER the specified date.

/U (=Unset) do not to set attribute 'a' for a certain file

/I (=Incremental) execute the backup of files that have not been previously saved only (without 'a' attribute)

/S (=Subdirectories) include subdirectories

/X (=eXclude) exclude files from the backup

At present the following predefined Savesets are available:

```
$BACKUP_SET[1] : "All|\"TTS_USER_DEV"\*\.* /s"
$BACKUP_SET[7] : "Today|\"TTS_USER_DEV"\*\.* /s/a0"
$BACKUP_SET[8] : "Inc|\"TTS_USER_DEV"\*\.* /s/i"
```

6.9.4.8.2 FILER UTILITY RESTORE (FUR)

The RESTORE command allows to copy files back from default backup/restore device (\$DFT_DV[3]) to the device from which such files had been saved by the Backup operation.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.



See [NOTES FOR BACKUP / RESTORE](#).

Options:

/Configure allows to recover the old configuration file that, after being copied with .C4O extension, is loaded on the Controller. This is followed by ConfigureSaveAll operation.

/Saveset to specify a predefined Saveset that indicates where the files are to be taken from to be copied.

The Saveset must have already been created.

For example, if it is required to use a Saveset referred to as '**set1|UD:\dir1*.***', parameter '**set1**' must be specified when issuing the Restore command. In that way, any files included in **UD:\dir1** and corresponding subdirectories, will be restored.

/After copies files that are more recent than a certain date. It is useful to restore files from a certain date, for example, today.

To specify date or count of days a sign can be indicated that has the meaning of "before" ('-') or "after" the indicated date or count of days.

Examples:

0 means files as from today

-1 means all files before 00 of today;

-18-06-08 means files before 18/06/08;

18-06-08 means files OF THAT DATE and AFTER IT.

/From copies files that are not on the default device (\$DFT_DV[3]). It is possible to specify the source directory on this device.

/Newer executes the restore of files that are more recent than the ones already existing in the destination.

/Overwrite overwrites also read-only files.

Syntax:	FUR <file_name> FUR/S <saveset_name>
Examples:	FUR executes restore of all files, from backup, and leaves the archive bit not set FUR *.cod executes restore of all files with extension .cod FUR/A-1 executes restore of new files, up to yesterday FUR/F executes restore from a different device FUR/N executes restore of new files ('new' is referred to the previous backup) FUR/S executes restore, including subdirectories FUR/S all executes restore of ALL Savesets



Please note that after a Restore operation, the files attributes are no longer read-only. Even if no files are restored, no error messages are displayed.

Using Savesets in the Restore command



As already said describing the /Saveset option, it is possible to specify a Saveset name instead of a filename (in such a case, when issuing the command, the /Saveset option must be inserted).

The specified Saveset must have already been created in the corresponding [Restore](#) subpage of [Setup page](#).

In this way, using a single command it is possible to restore files with a certain name to different devices and/or directories, include the subdirectories, exclude some files, etc.

The following switches are provided to properly setup the Saveset:

- /A** (=After) to execute the restore of created/modified files AFTER a specified date.
- /N** (=Newer) to execute the restore of files which are more recent than the ones included in the .LST file
- /O** (=Overwrite) to overwrite also read-only files
- /S** (=Subdirectories) to include subdirectories
- /X** (=eXclude) to exclude some files from the restore operation

At present the following predefined Saveset is available:

```
$RESTORE_SET[1] : "All|\"TTS_USER_DEV"\\"*.*\s/x*.lbe/x*.lba"
```

6.9.4.9 Compressed files management commands

6.9.4.9.1 FILER UTILITY COMPRESS DELETE (FUCD)

Deletes a file from a compressed archive (.ZIP)

Syntax: **FUCD <archive_name> <file_name>**

6.9.4.9.2 FILER UTILITY COMPRESS EXTRACT (FUCE)

Extracts the files contained in a compressed archive (.ZIP). The files extracted from the

archive will have the same attributes as the source file used to create the archive.

Syntax: **FUCE <archive_name> <file_name>**

6.9.4.9.3 FILER UTILITY COMPRESS MAKE (FUCM)

Creates a compressed file archive (.ZIP).

Options:

- /after only files with a date later than that specified are included in the archive
- /before only files with a date prior to that specified are included in the archive.
- /nodirpath The specification of the directories folder is not included in the archive.
- /refresh The archive is updated with the files that have been changed after the creation of the said archive .
- /subdir Files contained in subdirectories are also included.

Syntax: **FUCM <archive_name> <file_name>**

6.9.4.9.4 FILER UTILITY COMPRESS VIEW (FUCV)

Lists the files contained in a compressed archive (.ZIP).

As from system software version 3.20, further information can be viewed, using /Property option. Standard attributes (Property) associated to compressed files are the following:

- drawn up by
- file creation date
- version
- host where file was created.

Options:

- /Bare displays only the file name and not the other detailed information.
- /Property displays file properties (drawn up by, date of creation, version, device where it was created)

Syntax:
FUCV <archive_name> <file_name>
FUCV/B <archive_name> <file_name>
FUCV/P <archive_name> <file_name>

6.9.4.10 Directory management commands

The DIRECTORY branch displays a menu of commands to operate on the structure of the directories on the user disk UD:..

6.9.4.10.1 FILER UTILITY DIRECTORY CHANGE (FUDC)

To position on the desired directory specifying the name. For example, to have access to the directory UD:\pippo, the following command has to be sent

FUDC pippo

starting from the root of the user disk UD:..

To return to the upper directory level, the parameter has to be specified as '..'. To return to the root the parameter has to be specified as '\'.

This command is not allowed from SYS_CALL. To change the directory from PDL2

program, use the predefined routine DIR_SET.

Syntax: **FUDC <directory_name>**

6.9.4.10.2 FILER UTILITY DIRECTORY DELETE (FUDD)

To delete the specified directory.

Options: **/Noconfirm** eliminates the confirm prompt

Syntax: **FUDD <directory_name>**
FUDD/N <directory_name>

6.9.4.10.3 FILER UTILITY DIRECTORY MAKE (FUDM)

To create a new directory starting from the current position in the directories structure.

Syntax: **FUDM <directory_name>**

6.9.4.11 FILER UTILITY INSTALL (FUI)

It allows to install the specified application program, by automatically executing all the required operations: backup device identification from which files are to be copied (\$DFT_DV[6] equivalent to a default 'COMP:'), copy of the application installation file on the user disk UD: and its activation.

The requested parameter is the name of the application installation file (.COD file name) \$FUI_DIRS predefined variable contains the directory path where the being installed application is searched for.

Syntax: **FUI <installation_file>**

If the Ethernet network is to be used to install the required application, all \$NET_ prefix predefined variables containing references about the HOST where the desired application files are being installed, must be configured first.

For further details about Ethernet network configuration, see **PDL2 Programming Language Manual**, predefined \$NET_xx. variables description.

6.9.4.12 FILER UTILITY PROTECT (FUP)

Allows protecting .COD files on the Controller. The user is requested to specify the filename and the wished password. The use of wildcards is allowed. On the Controller the password is case insensitive.

Options: **/Noconfirm** eliminates the confirm prompt
/Unprotect clears the file protection

Syntax: **FUP <file_name> <password> <password_again>**
FUP/N <file_name> <password> <password_again>
FUP/U <file_name> <password>

6.9.4.13 FILER UTILITY SEARCH (FUS)

Can be used to find a string inside a file on the user disk UD:.

To run this operation the file has to be specified and the string to be searched for. For the file a wildcard can be used.

Syntax: **FUS <file_name> <string_searched_for>**
 Example: **FUS *.COD' '\$DOUT'**

6.9.4.14 FILER VIEW (FV)

The VIEW command displays the contents of the specified disk (UD: by default) in the directory currently selected on the device from which the command is sent. For each file the date of creation, the size in bytes, the attributes, the name and the extension are displayed.

If the file has a Readonly or System attribute this is indicated by the letter 'r' or 's' accordingly.

To know the attributes of a file from the PDL2 program, use the predefined FL_STATE routine.

One or more files can be specified as parameter. If no parameter is specified, all the files are listed.

At the bottom of the list of files the total number of files contained in the selected directory is indicated, with the number of bytes occupied. Also the size, in bytes, of the disk UD: is indicated, with the number of bytes occupied and the number that are free.

Options: **/After** - after a date: 1=yesterday; 0=today. This parameter can have two different formats: INTEGER or DATE-TIME. For the both of them the meaning is as follows: POSITIVE=AFTER, NEGATIVE=BEFORE.

/Bare shows the list of the file names, hiding the information regarding the date and the occupation

/Deleted Displays only the deleted files (in a non-permanent mode)

/Length displays the list of files from the smallest to the largest

/Modified shows the list of files starting from that with the oldest changes to the most recent

/Nopage

/Subdir displays also the list of files included in the subdirectories

/Unsort does not sort

Syntax: **FV <file_name>**
FV/D <file_name>
FV/L <file_name>
FV/M <file_name>
FV/N <file_name>
FV/S <file_name>
FV/U <file_name>

Examples: examples of using **/After** option:
FV/A * -1 displays all files before yesterday
FV/A * 1 displays all files after yesterday
FV/A * 0 displays all files which have been modified today
FV/A * -0 displays all files which have been modified before today

If today is 29-10-2009, then:

FV/A * 28-10-2009 : all files which have been modified since yesterday

FV/A * -28-10-2009 : all files which have been modified before yesterday

6.9.5 MEMORY branch

The MEMORY branch displays the commands to manage the programs and variables in the execution memory of the programs.

Besides the individual commands, also the following are described:

- ERASE menu
 - VIEW menu
-

6.9.5.1 MEMORY DEBUG

The DEBUG command calls the editor in view mode and debugs the program, that has to be already loaded in the execution memory.

This function allows the programmer to carry out these operations:

- change the current instruction being run
- change the position variables
- display the program running
- display interaction between programs
- change parameters that control the execution, such as break-point step, insertion of points
- change a program text (only if loaded with /FULL option).



If the program has been loaded with the /FULL option, the text can also be changed. Programs loaded with the /PERMANENT option, only available from SYS_CALL, cannot be used in MEMORY DEBUG.

The command requires the parameter with the name of the program.

To use this command the control must not be in FATAL state. DEBUG cannot be used if the commands FILER EDIT, PROGRAM EDIT or MEMORY TEACH are active, or if the IDE environment is active.

This command is not allowed from SYS_CALL.

Syntax: **MD <program_name>**

6.9.5.2 ERASE menu

A menu of commands to delete data and programs from the memory. A wildcard can be used for the names of the programs and the variables. Before starting to delete, all the programs display a confirm prompt.

6.9.5.2.1 MEMORY ERASE ALL (MEA)

Deletes the code and the variable of the program specified.

Options: **/Noconfirm** eliminates the confirm prompt.

Syntax: **MEA <program_name>**

6.9.5.2.2 MEMORY ERASE PROGRAM (MEP)

Deletes the part of the code referring to the specified program. If PROGRAM VIEW is already active, MEP cannot be used.

Options: **/Noconfirm** eliminates the confirm prompt.

Syntax: **MEP <program_name>**

6.9.5.2.3 MEMORY ERASE VARIABLE(MEV)

Deletes one or more variables of the specified program and not referred by a loaded program. The name of the variable has to be specified as a parameter.

Options: **/Noconfirm** eliminates the confirm prompt.

Syntax: **MEV <program_name> <variable_name>**

6.9.5.3 MEMORY LOAD (ML)

The LOAD command is used to load the program codes (.COD) and the variable files (.VAR) in the execution memory from the default storage device (UD),

It is asked to specify as parameter the name of the program to be loaded; the asterisk can be used. If a program does not have a data file (.VAR), the variables are created in memory during the loading.

A program cannot be loaded when it is in program ready, paused or running condition.



If it is not possible to load the program due to damage to the data contained in the .COD file, recover a backup copy of the same file (.BKC) or translate the program into ASCII format using the FILER TRANSLATE command. Next, edit the ASCII file obtained and correct the damaged instructions and identifiers, that are distinguished by three question marks ("??"). Once the corrections have been made, reconvert the file to a .COD file using the FILER TRANSLATE command.

Options: **/As** to load programs with a different data file (.VAR), specified as a parameter. Asterisks cannot be used with this option.

/Convert converts, where possible, conflicts between data file and the program

/Depend automatically loads the code and the variables of the programs referred in the program code specified as parameter to MEMORY LOAD. If the referred files are in different directories the predefined variable \$DEPEND_DIRS has to be set, that contains the path that the Load command will use in the search.

/Full loads the entire program with the related comments, wrong statements and empty lines. This option is useful for no motion programs. All other kind of programs are already entirely loaded.

/Nosavevars disables the saving of the variables (or VAR) when MEMORY SAVE is executed on this program. This option is only available if used together with the internal SYS_CALL routine.

/Variables loads only the data file of the specified program.

/Permanent loads the program code in the memory in permanent mode, not permitting deletion. The code uses less memory because of the optimizing of the loading process. To recover the memory, the control has to be restarted. This option is only

available if used together with the internal SYS_CALL routine.

Syntax:

```
ML <program_name>
ML/A <data-file program_name>
ML/F <program_name>
```

6.9.5.4 MEMORY SAVE(MS)

The SAVE command saves the file containing the code or the variables of a program that is loaded in memory.

If no option is specified, the command saves by default only the program variables, creating a new .VAR file with the name of the program.

The directory where the file of variables is saved is (in the order indicated here below):

- the one from where the .VAR was originally loaded;
- if no .VAR file was loaded, the directory from where the .COD file was loaded;
- if no .COD was loaded, the directory which is currently selected on the device from where the save command is issued (programming unit, Terminal of WinC5G, currently selected directory in case of SYS_CALL).



The MemoryViewProgram command is useful to check which file and directory have been used to load the code and the variables of a program.

Issue the /Code option to save the code.

If there is already another data or program file with the same name, this will be renamed and it will be assigned a .BKV or .BKC extension .

If data have not been loaded with /Nosavevar option, variables of this program will not be saved.

Options:

- /As** allows the saving of the program code or data in a file with the name indicated in the second parameter. The asterisk cannot be used with this option.
- /Code** saves the code of a program (instead of the variables), loaded with the /FULL option and if it has been modified since when it was loaded.
- /Dirorig** saves the file of variables with the same name and directory used while loading.
- /Overwrite** if the file already exists in the user disk, it is replaced without the display of the confirm prompt.
- /Total** (only from SYS_CALL).

Syntax:

```
MS <program_name>
MS/A <program_name> <file_for_saving>
MS/C <program_name>
MS/D <program_name>
MS/O <program_name>
MS/T <program_name>
```

6.9.5.5 MEMORY TEACH

The TEACH command is used to change the position variable values in the execution memory and to act on the nodes of a PATH. It is only allowed from Teach Pendant.

This environment allows the operator to carry out these operations:

- assign a physical position of the robot to a position variable
- change position variable values
- create, delete and change the data of a PATH.

It is necessary to specify as a parameter the name of the program where the variables are to be learnt. The program must be loaded in the memory, but not active.

TEACH cannot be used at the same time with the IDE environment, and the PROGRAM EDIT (DATA mode) and MEMORY DEBUG commands; nor if the system is in FATAL state. It is not allowed from SYS_CALL.

Syntax: **MT <program_name>**

6.9.5.6 VIEW menu

VIEW menu is used to have access to a commands menu to display the contents of the memory.

All MEMORY VIEW commands support the /NOPAGE option.

6.9.5.6.1 MEMORY VIEW DATABASE (MVD)

Displays a Database currently present in memory. The use of the wildcard '*' is allowed. If the wildcard is used as first parameter, all the currently present in memory Databases are displayed; if it is specified as second parameter, the command displays all the records belonging to the Database which is selected by the first parameter.

Options: **/Nopage**
/Output

Syntax: **MVD <database_name> <record_key>**

6.9.5.6.2 MEMORY VIEW PROGRAM (MVP)

Displays the information regarding the programs loaded and the data regarding the occupation of the memory as described here below:

- program name
- program attributes (H=hold, N=nohold, A=attach, D=detach)
- name of data file (.VAR) used for loading (- indicates not loaded from a file)
- occupation of the program code (.COD) (0 indicates not loaded) and number of lines
- size of data (0 indicates not loaded) and number of variables
- date and time of creation of the .COD file and the .VAR file loaded in memory for this program. To obtain this information the /Full option is needed.
- a symbol indicating whether the instructions or the variable loaded have been changed in the memory, but not saved on the user disk. Entering this command by WINC5G or by SYS_CALL, the symbol is displayed next to the size of the code and the size of the variables. If instead the command is sent from the teach pendant, also the /Full option has to be entered. The following symbols are used:
 - “S” indicates that the program code or variables have been saved

- “**” indicates that the program code or the variables have been changed, but not saved
- “-” indicates that the program code or the variables have not been loaded into the memory
- “P” indicates that the program code has been loaded by SYS_CALL with the /Nosavevars option
- “?” indicates that the file is no longer present, so it is not possible to compare with the current situation in memory.

Further information can be viewed, using /Property option. Standard attributes (Property) associated to the files are the following:

- author
- file creation date
- version
- host where file was created.

Options: **/Full** displays all the information referring to the program.

/Nopage

/4 (see [Option /4 to view in the 40 columns](#))

/Property displays program properties (drawn up by, date of creation, version, host where it was created).

Syntax:

MVP <program_name>

MVP/F <program_name>

MVP/N <program_name>

MVP/4 <program_name>

MVP/P <program_name>

6.9.5.6.3 MEMORY VIEW SYSTEM (MVS)

Displays the specified system variable.

Options: **/Nopage**
/Output
/Rules

Syntax:

MVS <system_variable_name>

MVS/N

MVS/R

6.9.5.6.4 MEMORY VIEW TREE (MVR)

Displays all the tree structures currently present in memory. The use of the wildcard “*” is allowed.

Options: **/Full** provides full information, not only the hierarchy
/Nopage
/Output

Syntax:

MVR <tree_structure_name>

MVR/F <tree_structure_name>

6.9.5.6.5 MEMORY VIEW TYPE (MVT)

Displays the type of data (TYPE declarations in the programs loaded in memory) defined by the user, as follows:

- name of type of data (TYPE)
- size in bytes
- counter of references from part of programs or variables to this type of data
- number of fields
- type of structure (record or node)
- description of the fields (information on the name and type of data of each field).

The type of data defined by the user are global in the system. However they remain in the memory as long as there is a program loaded or referring variables of a program.

If no type of parameter is specified, the information is displayed regarding all the types loaded. The asterisk can be used.

Options: **/Nopage**

Syntax: **MVT <type_name>**

6.9.5.6.6 MEMORY VIEW VARIABLE (MVV)

Displays the information regarding the variables loaded, as follows:

- name of variable
- type of variable
- references counter (number of programs loaded that refer to that variable)
- status (private, exported, local routine)
- value.

If the name of the program or of the variable are not specified as parameters, the information is displayed regarding all the loaded variables. A wildcard can be used instead of the name of the variable or the program.

The data class variables type STRING or ARRAY, where an asterisk was used in the declaration and for which the real size has not been declared, are displayed with size 0.

Options: **/Nopage**

/Full displays the complete names of the variables, without shortening because of their length.

Syntax: **MVV <program_name> <variable>**

6.9.5.6.7 MEMORY VIEW VP2 (MVV)

Displays the VP2 components of the specified program.

Options: **/Full** displays the components names in full mode, i.e. without any truncation due to the length.

/Nopage

/Output

Syntax: **MVV <program_name>**

MVV/N

MVV/F

6.9.6 PROGRAM branch

PROGRAM branch displays a menu of commands used to develop and run the

programs.

Besides the individual commands, the following are also described:

- [ResPLC menu](#)
 - [UTILITY menu](#)
 - [STATE menu](#)
 - [TEST branch](#)
-

6.9.6.1 PROGRAM ACTIVATE (PA)

The effect of the ACTIVATE command depends on the holdable/non-holdable attribute of the program specified as parameter:

Activating a holdable program, it is put in ready state. To run it, press the **START** button.

Activation of a non-holdable program sets it directly in running state.

If a holdable program is already active on an arm and one of the programs has the DETACH attribute specified, a prompt will be displayed to confirm whether two or more programs are to remain active on the same arm. If a holdable program is still active and the DETACH attribute is specified, a message will be displayed indicating that the arm is already “DETACHED”.

Before it can be activated, the program has to be loaded into the memory. The asterisk cannot be used. The control must not be in FATAL state.

The following options, except the ARGUMENT option, can be used to activate the program with a step other than the default step. See the indications regarding PROGRAM TEST STEP to have the exact significance of each step.

The ARGUMENT option is used to specify a parameter to pass to the program, when it is activated. This parameter is automatically saved in the predefined variable \$PROG_ARG, and can be used at will within the program.

Options:	/Argument - to pass a parameter to a program when it is activated /Cycle /Disable /Fly /Move /Routine /Statement
-----------------	---

Syntax: **PA <program_name>**

6.9.6.2 PROGRAM DEACTIVATE (PD)

DEACTIV command deactivates the programs that are in running, ready, paused or unpause status. After they have been deactivated the programs remain in the memory, but the execution cannot continue. To do so, PROGRAM ACTIVATE has to be used.

The program to be deactivated is specified as a parameter. The asterisk can be used inside the parameter to deactivate more than one program. The control must not be in FATAL state.

Options:	/Confirm displays the confirm prompt before the program deactivation.
-----------------	--

Syntax: **PD <program_name>**

All active programs can be deactivated by PD * or CTRL Y (^Y). CTRL Y can be emitted regardless of the level of menu that is active and is only valid if the mode that is active is PROG and there is no protection active.

Protected programs cannot be deactivated by this command, to deactivate them it is necessary to use the instruction or the action of PDL2 DEACTIVATE.

6.9.6.3 PROGRAM EDIT

The EDIT command is an integrated programming environment that allows the programmer to edit, learn points and check PDL2 programs by executing them.

The program editor uses the files containing the codes of the PDL2 programs (.COD) stored in the user disk (UD:). The name of the program is used as parameter and it is not possible to use a wildcard.

When it is called, the editor tries to enter by default in DATA mode. The programs can be edited in DATA mode only when the status selector switch is set to T1 and the program is not active.

If it is not possible to enter in DATA mode, the program is opened in CODE mode, that allows the programmer to change the program instructions, but not to execute the verification or the learning of variables. The CODE mode has to be used to make changes in the declarations section of a program. In CODE mode any program can be edited, whether activated or loaded.



If it is not possible to load the program due to damage to the data contained in the file, recover a backup copy of the same file or convert it to an ASCII file using the FILER PRINT command.

This command is not allowed from SYS_CALL.



The EDIT command cannot be used when the control is in FATAL state; the same applies in cases where the FILER EDIT, MEMORY DEBUG or PROGRAM EDIT (DATA or CODE mode) commands are active or when the IDE environment is enabled. The use of the PROGRAM EDIT command with the MEMORY TEACH command active is possible in CODE mode, but not in DATA mode.

Options: **/Code** calls up the editing in CODE mode, allowing the programmer to edit the program instructions, but it is not possible to execute learning or program verification. CODE is to be used to make changes in the declaration section of a program. While in CODE mode, it is possible to edit any program, even if it is activated or loaded.

Syntax: **PE <program_name>**

6.9.6.4 PROGRAM GO (PG)

GO command loads and activates the specified program. It is a combination of the MEMORY LOAD and PROGRAM ACTIVATE commands.

If a holdable program is already active on an arm and one of the programs has the DETACH attribute, a prompt will be displayed to confirm whether two or more programs

are to remain active on the same arm. If a holdable program is still active and the DETACH attribute is not specified, a message will be displayed indicating that the arm is already "ATTACHED".

If the program is already active the programmer is asked if it is to be deactivated.

If the program is already loaded and the .COD file belongs to a more recent version than that which is loaded, the operator will be asked whether the new version is to be loaded. The asterisk cannot be used.

The command cannot be used when the control is in FATAL state.

For a holdable program, the first program move is executed at reduced speed; this also happens for the first move after the CTRL C (^C) key has been pressed or if the MEMORY DEBUG cursor has been moved.

This safety function that reduces the speed can be disabled by setting the opportune bits in the predefined variable \$PROG_CNFG. To establish whether a move is executed at reduced speed, see the predefined variable \$SAFE_ENBL.



If it is not possible to load the program due to damage to the data contained in the file, recover a backup copy of the same file or convert it to an ASCII file using the FILER PRINT command. For further information regarding this procedure see the section relating to the FILER PRINT command.

The following options, except for the Update and Argument, can be used to activate the program with a step that is not the default step. See the description of the PROGRAM TEST STEP command.

UPDATE option disables the question to the user, if the program to be activated is already loaded and/or active in memory and in the user disk there is a more recent version.

The ARGUMENT option is used to specify a parameter to pass to the program, when it is activated. This parameter is automatically saved in the predefined variable \$PROG_ARG, and can be used at will within the program.

Options: **/Argument** - to pass a parameter to a program when it is activated
/Cycle
/Disable
/Fly
/Move
/Routine
/Statement
/Update: replaces the program in the memory, even if active, loading the new copy from the user disk and activating it.

Syntax: **PG <program_name>**

6.9.6.5 ResPLC menu

The ResPLC branch contains the PLC interfacing commands.

6.9.6.5.1 PROGRAM ResPLC ACTIVATE (PRA)

Starts the PLC resource execution.

Options: **/Restart** restarts the execution of the resource

/Single_scan executes a single resource scan

Syntax: PRA <resource_name>

6.9.6.5.2 PROGRAM ResPLC DEACTIVATE (PRD)

Disables the execution of the PLC resource.

Syntax: PRD <resource_name>

6.9.6.5.3 PROGRAM ResPLC RESTORE (PRR)

Restores the PLC resource.

Syntax: PRR <resource_name>

6.9.6.6 UTILITY menu

6.9.6.6.1 PROGRAM ResPLC UTILITY PROJDIR (PRUP)

Selects the default directory for the PLC resource.

Syntax: PRUP <directory_name>

6.9.6.6.2 PROGRAM ResPLC UTILITY SAVE (PRUS)

Saves the PLC resource.

Syntax: PRUS <resource_name>

6.9.6.6.3 PROGRAM ResPLC UTILITY UNLOAD (PRUU)

Deletes the resource of memory.

Syntax: PRUU <resource_name>

6.9.6.6.4 PROGRAM ResPLC UTILITY VIEW (PRUV)

Views information about the PLC environment:

- resource name
- name of the project present on the controller (for example UD:\RESPLC)
- project directory
- resource number
- dimension in bytes of the resource file
- resource execution modality
- cycle time.

Syntax: PRUV <resource_name>

6.9.6.7 STATE menu

The STATE menu displays the commands to change the execution state of an active program.

6.9.6.7.1 PROGRAM STATE BYPASS (PSB)

Allows the programmer to skip the current instruction when checking the program. This is useful if the program is suspended, for example waiting for a certain condition to be verified. BYPASS cannot be used when the control is in FATAL state.

Syntax: **PSB <program_name>**

6.9.6.7.2 PROGRAM STATE PAUSE (PSP)

Changes the program state from running to paused. Both holdable and non holdable programs are set in paused state.

The asterisk can be used in the program name. The command cannot be used when the control is in FATAL state.

Options: **/Confirm** displays the confirm prompt before setting the programs in paused state.

Syntax: **PSP <program_name>**

6.9.6.7.3 PROGRAM STATE UNPAUSE (PSU)

Continues the execution of the paused program. Holdable programs are set in ready state and resume execution when START is pressed. Non holdable programs are set to running state.

The asterisk can be used for the program name. The command cannot be used when the control is in FATAL state.

Syntax: **PSU <program_name>**

6.9.6.8 TEST branch

The TEST branch displays the menu with the commands to Debug a program without being inside an editor environment. TEST cannot be used when the control is in FATAL state.

A description of these subjects follows:

- [Break menu](#)
- [PROFILE menu](#)
- [STEP menu](#)

6.9.6.8.1 Break menu

BREAK menu displays a commands used to set and use the break points in a program.

A break point is a point inside the program where the execution stops, setting the program in ready. To resume the program, press **RUN**. For holdable programs a **START** is also necessary.

The BREAK commands can be used for any loaded program, including programs currently active. The break points are numbered automatically by the system.

PROGRAM TEST BREAK INSERT (PTBI)

Inserts a break point. The break points inserted have immediate effect.

The required parameters are the name of a program and the line number; that must correspond to an executable code. The break takes place on the specified line number, before the instruction is executed. If no line number is specified, an error message is displayed.

Options: **/Label** to specify a label instead of the line number.
/Routine to specify a routine name instead of the line number.



Both the **/LABEL** and **/ROUTINE** options are to be used to specify a label inside a routine.

Syntax: **PTBI <program_name> <line_number>**
PTBI/L <program_name> <label>
PTBI/R <program_name> <routine_name>
PTBI/LR <program_name> <routine_name> <label>
PTBI/RL <program_name> <routine_name> <label>

PROGRAM TEST BREAK PURGE (PTBP)

Removes the break points from inside a program. The break point has to be specified, or a wildcard can be used to remove all the break points.

Syntax: **PTBP <brk_pt_num> <program_name>**

PROGRAM TEST BREAK VIEW (PTBV)

Displays a list of break points. A wildcard can be used to specify the name of a program.

Options: **/Nopage**

Syntax: **/PTBV <program_name>**

6.9.6.8.2 PROFILE menu

PROFILE menu displays a menu of commands to control the performance regarding the time the CPU is occupied by active programs. This information is useful when several programs are executed simultaneously.

PROGRAM TEST PROFILE DISABLE (PTPD)

Disables the performance calculation. If PROFILE is already disabled, a warning is displayed.

Syntax: **PTPD**

PROGRAM TEST PROFILE ENABLE (PTPE)

Enables the performance calculation. The CPU time calculation starts as soon as PROFILE is enabled.

Syntax: **PTPE**

PROGRAM TEST PROFILE RESET (PTPR)

Resets the information related to the service calculations.

Syntax: **PTPR**

PROGRAM TEST PROFILE VIEW (PTPV)

Displays the information on the performance:

- PROFILE starting time
- PROFILE duration (time lapsed since its enabling)
- PROFILE state (enabled or disabled)
- following information for each active program: program name, CPU time used, percentage of CPU time used.

If during the PROFILE enabling an EXECUTE command has been sent, also the calculations on the service related to its execution will be displayed, identified by the program name "E".

The performance calculations listed near the program named "Free" indicate the CPU time not used by the programs or by EXECUTE.

Options: **/4** (see [Option /4 to view in the 40 columns](#))

Syntax: **PTPV**

6.9.6.8.3 STEP menu

The STEP menu displays the commands that allow the execution step of a certain program to be changed. A step defines how many instructions are to be executed at a time. As each step is completed, the program is set in READY state. To resume execution a START is needed for holdable programs, or RUN for the non holdables.

To set the execution step, the program name has to be specified (asterisk is allowed) and it must be already loaded in memory.

When a program step is changed while running, the new step will come into effect immediately after the execution of the current instruction.

PROGRAM TEST STEP CYCLE (PTSC)

Defines the step as a single cycle inside the program. The program must contain the CYCLE or BEGIN CYCLE instruction.

Syntax: **PTSC <program_name>**

PROGRAM TEST STEP DISABLE (PTSD)

Restores the program continuous execution mode and disables the previously set execution step. The asterisk can be used to specify a program.

Syntax: **PTSD <program_name>**

PROGRAM TEST STEP FLY (PTSF)

Executes two movements in fly before suspending the running of the program. Defines the step as a single movement: it is similar to PTSM, but the execution does not stop after MOVEFLY. It cannot be used on non-holdable programs.

Syntax: **PTSF <program_name>**

PROGRAM TEST STEP MOVE (PTSM)

Defines a step as a single movement. It cannot be used on non-holdable programs.

Syntax: **PTSM <program_name>**

PROGRAM TEST STEP ROUTINE (PTSR)

Defines the step as one instruction at a time, except that the routines are executed as if they were a single instruction.

Syntax: **PTSR <program_name>**

PROGRAM TEST STEP STATEMENT (PTSS)

Defines the step as a single instruction.

Syntax: **PTSS <program_name>**

PROGRAM TEST STEP VIEW (PTSV)

Displays the step associated to a program. The program name can be specified with a wildcard, otherwise all the steps regarding all the programs will be displayed.

Options: **/Nopage**

Syntax: **PTSV <program_name>**

6.9.6.9 PROGRAM VIEW (PV)

VIEW command is used to displays the following information regarding active programs together with these values:

- program name;
- program attributes;
- arm number;
- priority;
- primary state (running, in hold, suspended);
- number of the line that is executed and program the line belongs to;
- error number;
- CPU time - is the time, in seconds, during which the program has been actually interpreted by the PDL2 programs interpreter.
- number of cycles.

A wildcard can be used for the program name. The command cannot be used when the control is in FATAL state or when MEMORY ERASE PROGRAM is active.

Options: **/Chain** displays the nesting of the calls to routines

- routine name
- program the routine belongs to
- line number
- execution context (main program, program routine, routine exported by another program, call to routine from CONDITION, private)

- state
- /Full** displays further information, such as:
- program attribute
 - arm number
 - priority
 - stack dimension (StSz)
 - stack currently used (StUd);
 - maximum percentage of stacks used so far by the program (StMax)
 - program current state (running, held, suspended)
 - reason why the execution has been suspended
 - number of line being executed and program it belongs to
 - date and time when the program was activated
 - breakpoints in the program code
 - step execution mode
 - number of condition managers (CONDITION) defined and list of them (those enabled, locally or globally, are marked with a wildcard)
 - number of errors on which the predefined ERR_TRAP_ON routine is active
 - predefined variable values in the Program Stack category and for which a copy exists for each program
 - nesting of the routine calls
 - CPU time, in seconds, actually occupied by the PDL2 programs interpreter to interpret the current program.

/Nopage

/4 (see [Option /4 to view in the 40 columns](#))

/Wide (only from SYS_CALL) Displays some information in full length (for example owner program name of the line), in relation to the default mode.

Syntax: **PV <program_name>**

6.9.7 SET branch

A description of these subjects follows:

- [SET ARM menu](#)
 - [SET CNTRLR menu](#)
 - [SET INPUT menu](#)
 - [SET OUTPUT menu](#)
-

6.9.7.1 SET ARM menu

ARM menu displays a menu of commands used to set a certain state of the arm.

6.9.7.1.1 SET ARM DISABLE (SAD)

Disables the DRIVE ON function for an arm, the errors generated when the encoder cables are not connected to the robot, the indications of the drive failure; it is only allowed when the system is in PROGR or ALARM state and when the power supply to the drives is off.

The arm remains disabled until it receives a SetArmEnable command, regardless of any repowering that may be executed.

The SetArmDisable command is usually used when it is necessary to perform repair operations on a certain arm. After the operations have been carried out, the system may still report the presence of some errors because a hardware component has remained blocked. To remove these alarms it is necessary to restart the Controller.

For systems that have more than one arm, it is necessary to specify the arm number as a parameter. To indicate all the arms an asterisk can be used.

Syntax: **SAD <arm_num>**

6.9.7.1.2 SET ARM ENABLE (SAE)

Enables the DRIVE ON function for an arm. This command can only be used when the system is in PROGR state, with the drives power supply disabled.

For systems with several arms, the number of the arm is to be specified as a parameter. To indicate all the arms a wildcard can be used.

Syntax: **SAE <arm_num>**

6.9.7.1.3 SET ARM GEN_OVR (SAG)

Assigns the value specified in the parameter to the \$GEN_OVR variable. This command can be used to increase or decrease the value of \$GEN_OVR. If the command is entered from WinC5G or from SYS_CALL, and the system is in PROGR mode, the value can only be decreased.

Options: **/Increment** sets the increase/decrease default unit used to update the current value of \$GEN_OVR when the %+ and %- keys of the teach pendant are used to change the value. The default value is 5.

Syntax: **SAG <value>**
SAG/I <value> <unit_incr/decr>

6.9.7.1.4 SET ARM NOSTROKE(SAN)

For the temporary disabling of the limit switches set for a certain arm. With the limit switches disabled the robot can only be moved manually and therefore the system has to be in PROGR state. The limit switches are automatically reset when the system is in AUTO or HOLD, when the drives power supply is disabled or when the arm referred by the Teach Pendant is changed.

For systems with several arms the number of the arm must be indicated as a parameter. The asterisk can be used to indicate all the arms.

Syntax: **SAN <arm_num>**

6.9.7.1.5 SET ARM SIMULATE (SAS)

To run programs simulating the movements without actually moving the arm. All the movement calculations are executed just the same. This command can only be sent with the power to the drives off (DRIVE OFF).

After the arm has been set in the simulated state, the drives can be powered, after setting the status selector switch on PROGRAMMING and moving the robot manually. The command requires an arm as a parameter and can be used to:

- test programs without moving the arm;
- establish approximate cycle times;
- check the program logic.

Syntax: **SAS <arm_num>**

6.9.7.1.6 SET ARM TP_MAIN (SAT)

Sets the default arm (usually set at 1) for movement by teach pendant. It requires the arm as a parameter. Requires the arm as a parameter. To use this command, in PROG, the power supply to the drives must be off. This command is not possible if it is introduced by WinC5G or by SYS_CALL. The command is only enabled in LOCAL and REMOTE states, even with DRIVE ON.

The command is not allowed if it is inserted from WInC5G or SYS_CALL with the Teach Pendant connected (or disconnected while the system is in PROGR state).

Syntax: **SAT <arm_num>**

6.9.7.1.7 SET ARM UNSIMULATE

Deactivates simulation on the arm. This command requires the arm as a parameter and it is to be sent with the drives power supply disabled (DRIVE OFF).

Syntax: **SAU <arm_num>**

6.9.7.2 SET CNTRLR menu

6.9.7.2.1 SET CNTRL KEY_LOCK (SCK)

Prevents unauthorised use of the menu in the Terminal window on **WinC5G**. To use this command a password is necessary. The screen remains locked and asks for the password to remove this condition and resume access to the commands menu.

Syntax: **SCK password**

6.9.7.2.2 SET CNTRL LANGUAGE (SCL)

To set the wished language on the Controller. The User has to specify a parameter which indicates the language. The command has immediate effect as soon as it is sent.

Syntax: **SCL <chosen_language>**

The parameter specifies the wished language (the languages list is automatically opened) and must be chosen among the following:

- **cs** - Czech language
- **de** - German language
- **es** - Spanish language
- **en** - English language
- **fr** - French language
- **it** - Italian language
- **pl** - Polish language
- **pt** - Portuguese language (for Portugal)

- **tr** - Turkish language
- **zh** - Chinese language
- **en.us** - English language (for U.S.A.)
- **pt.br** - Portuguese language (for Brazil).

6.9.7.2.3 SET CONTROLLER VIEW (SCV)

Displays the current values about the following information:

- System configuration (System identifier, date and time, Controller name, configuration file name, modal/nodal MOVE, existing Arms)
- versions (System software and compiling details, TP software, user interface, etc.) and software options (synchronized motion, motion with weaving, interference regions, etc.)
- hardware (model, revision and serial number, related to the CPU, the TP and all other modules, hourmeter, existing axes, etc.)
- communication (IP address, logged users both from **WinC5G** and from TP, etc.)

A prompt message asks for the output destination.

Options: **/4** (see [Option /4 to view in the 40 columns](#))
/Output redirects the output to a file



NOTE - Compatibility in the file format is not guaranteed, between different versions of the system software.

Syntax: **SCV**

6.9.7.2.4 SET CONTROLLER WIN_CLEAR (SCW)

Deletes the information shown in a window.

Syntax: **SCW <window_name>**

6.9.7.3 SET INPUT menu

SET INPUT menu contains commands for handling the inputs.

The FORCE, SIMULATE and UNFORCE commands are not allowed if they are not issued from the teach pendant and the state is T1 or T2.

A description of these subjects follows:

- [SET INPUT FORCE branch](#)
- [SET INPUT REMOTE branch](#)
- [SET INPUT UNFORCE branch](#)
- [SET INPUT VIEW branch](#)

6.9.7.3.1 SET INPUT FORCE branch

FORCE command forces the value of a certain input. The physical input is no longer considered and the forced value is used. The operator can choose through a sub-menu whether to force analog (AIN), digital (DIN), privileged digital (IN), flexible multiples (FMI)

or group (GIN) inputs.

To indicate the input to be forced it is necessary to specify the index.

The asterisk cannot be used. As value to be forced it is possible to indicate:

- T: TRUE (ON)
- F: FALSE (OFF)
- C: the input will be forced to its current physical state.

After selecting the appropriate value, press **ENTER**.

A digital input that is part of a group of inputs has to be forced as a group and not as a separate digital input.

This command is not allowed if sent by WinC5G or by SYS_CALL with the teach pendant not on the RCC (Robot Controller Cabinet).

SET INPUT FORCE AIN (SIFA)

Syntax: **SIFA <index_num> <forced_value>**

SET INPUT FORCE DIN (SIFD)

Syntax: **SIFD <index_num> <forced_value>**

SET INPUT FORCE FMI (SIFF)

Syntax: **SIFF <index_num> <forced_value>**

SET INPUT FORCE IN (SIFI)

Syntax: **SIFI <index_num> <forced_value>**

6.9.7.3.2 SET INPUT REMOTE branch

Enables/disables the use of some input signals from remote: START, HOLD, DRIVE ON, DRIVE OFF.

The following commands are associated to the User Interface **Remote Inp.** key on the Teach Pendant.

SET INPUT REMOTE DISABLE (SIRD)

Disables the use of the listed above input signals, coming from remote. When this command is issued, the System does not take them into consideration, regardless of their value from PLC.

Syntax: **SIRD**

SET INPUT REMOTE ENABLE (SIRE)

Enables the use of the listed above input signals, coming from remote. When this command is issued, such signals status is driven by the PLC.

Syntax: **SIRE**

6.9.7.3.3 SET INPUT UNFORCE branch

Returns a forced logic input to the original physical configuration. The command acts on analog inputs (AIN), digital inputs (DIN), privileged digital inputs (IN), flexible multiple inputs (FMI) or group inputs (GIN).

The asterisk cannot be used.

SET INPUT UNFORCE AIN (SIUA)

Syntax: SIUA <index_num>

SET INPUT UNFORCE DIN (SIUD)

Syntax: SIUD <index_num>

SET INPUT UNFORCE FMI (SIUF)

Syntax: SIUF <index_num>

SET INPUT UNFORCE IN (SIUI)

Syntax: SIUI <index_num>

SET INPUT UNFORCE TOTAL (SIUT)

Syntax: SIUT

6.9.7.3.4 SET INPUT VIEW branch

Displays the forced inputs. The operator can choose through the sub-menu, whether to display the forced analog (AIN), digital (DIN), privileged digital (IN), flexible multiples (FMI) or group (GIN) inputs.

If a group of inputs is forced, all the corresponding digital inputs will be displayed as forced from the moment the SIVD command is used.

SET INPUT VIEW AIN (SIVA)

Syntax: SIVA

SET INPUT VIEW DIN (SIVD)

Syntax: SIVD

SET INPUT VIEW FMI (SIVF)

Syntax: SIVUF

SET INPUT VIEW GIN (SIVG)

Syntax: SIVG

SET INPUT VIEW IN (SIVI)Syntax: **SIVI**

SET INPUT VIEW SYSTEM (SIVS)Syntax: **SIVS**

6.9.7.4 SET LOGIN (SL)

To login on the Controller. The user identification (Userid) and the Password have to be entered. The user has to declare beforehand the user profile to be used to access the Controller, through the ConfigureControllerLoginAdd command sent by a user with Administrator profile.

This command is not allowed from SYS_CALL.

Options: **/Logout** allows the user currently using the Controller to logout from the device where the command is entered.



As far as the Logout, a function is also available called "Automatic Logout". For further details see [par. 4.3.1 Automatic Timed Logout on page 48](#).

Syntax: **SL <username><password>**

6.9.7.5 SET OUTPUT menu

OUTPUT menu contains commands for handling the outputs.

The FORCE, SIMULATE and UNFORCE commands are not allowed if they are not issued from the teach pendant and the state is T1 or T2.

6.9.7.5.1 SET OUTPUT FORCE branch

FORCE command simulates the outputs. The forced value is used by the program and assigned to the physical output. Any assignment to the forced outputs will not change the physical outputs or the program execution. It is not possible to change the value of the forced outputs until the forced state is removed by UNFORCE. The operator can choose whether to force the analog (AOUT), digital (DOUT), privileged digital (OUT), flexible multiples (FMO) or group (GOUT) outputs.

To indicate the output to be forced, the index has to be specified.

The asterisk cannot be used.

As value to be forced it is possible to indicate:

- T: TRUE (ON)
- F: FALSE (OFF)
- C: the output will be forced to its current physical state.

Select the appropriate value and press **ENTER**.

A digital output that is part of a group has to be forced as group and not as a single digital output.

SET OUTPUT FORCE AOUT (SOFA)

Syntax: **SOFA <index_num> <forced_value>**

SET OUTPUT FORCE DOUT (SOFD)

Syntax: **SOFD <index_num> <forced_value>**

SET OUTPUT FORCE FMO (SOFF)

Syntax: **SOFF <index_num> <forced_value>**

SET OUTPUT FORCE OUT (SOFO)

Syntax: **SOFO <index_num> <forced_value>**

6.9.7.5.2 SET OUTPUT UNFORCE branch

Removes the forced value from an output. The physical state of the output is not changed.

The operator can choose whether to reconfigure the analog (AOUT), digital (DOUT), privileged digital (OUT), flexible multiples (FMO) or group (GOUT) outputs.

To indicate the output to be forced, the index has to be specified. The asterisk cannot be used.

SET OUTPUT UNFORCE AOUT (SOUA)

Syntax: **SOUA <index_num>**

SET OUTPUT UNFORCE DOUT (SOUD)

Syntax: **SOUD <index_num>**

SET OUTPUT UNFORCE FMO (SOUF)

Syntax: **SOUF <index_num>**

SET OUTPUT UNFORCE OUT (SOOU)

Syntax: **SOOU <index_num>**

SET OUTPUT UNFORCE TOTAL (SOUT)

Syntax: **SOUT**

6.9.7.5.3 SET OUTPUT VIEW branch

Displays the forced outputs. The operator can choose whether to display analog (AOUT), digital (DOUT), privileged digital (OUT), flexible multiples (FMO) or group (GOUT) outputs.

If a group of outputs is forced, all the corresponding digital outputs are displayed as

forced from the moment the SOVD command is used.

SET OUTPUT VIEW AOUT (SOVA)

Syntax: **SOVA**

SET OUTPUT VIEW DOUT (SOVD)

Syntax: **SOVD**

SET OUTPUT VIEW FMO (SOVF)

Syntax: **SOVUF**

SET OUTPUT VIEW GOUT (SOVG)

Syntax: **SOVG**

SET OUTPUT VIEW OUT (SOVO)

Syntax: **SOVO**

SET OUTPUT VIEW SYSTEM (SOVS)

Syntax: **SOVS**

6.9.8 UTILITY branch

A description of these subjects follows:

- [UTILITY APPLICATION \(UA\)](#)
- [UTILITY COMMUNICN MOUNT menu](#)

6.9.8.1 UTILITY APPLICATION (UA)

Reserved.

Syntax: **UA**

6.9.8.2 UTILITY COMMUNICN menu

Includes the commands to manage the communication devices (serial ports and network devices) to the Controller.

The description of these commands/menus follows:

- [UTILITY COMMUNICN DISMOUNT \(UCD\)](#)
- [UTILITY COMMUNICN MOUNT menu](#)
- [UTILITY COMMUNICN PORT_CHAR \(UCP\)](#)
- [UTILITY COMMUNICATION SET_DEF \(UCS\)](#)
- [UTILITY COMMUNICATION VIEW \(UCV\)](#)

6.9.8.2.1 UTILITY COMMUNICN DISMOUNT (UCD)

Disconnects the currently activated protocol on a communication device.

Syntax: **UCD <comm_port>**

6.9.8.2.2 UTILITY COMMUNICN MOUNT menu

Contains commands for the installation of different communication protocols to be activated on a certain communication device. If this is not specified, the default command (\$DFT_DV[8]) will be used.

UTILITY COMMUNICN MOUNT C5G_Int (UCMC)

Activates the communication protocol to the WinC5G program that can be activated on PC. Before typing this command on the control, make sure that the program has been started on the PC and that the connection with the C5G Controller is set correctly. To terminate this type of communication with the PC send the **UTILITY COMMUNICN DISMOUNT (UCD)** command.

Syntax: **UCMC <port_name>**

UTILITY COMMUNICN MOUNT Modem (UCMM)

This "mount" command reserves the use of the specified serial communication port for the modem.

For the modem configuration, the predefined variables \$MDM_INT and \$MDM_STR are available.

To release the port use the **UTILITY COMMUNICN DISMOUNT (UCD)** command.

Syntax: **UCMM <port_name>**

UTILITY COMMUNICN MOUNT 3964R (UCM3)

Installs the 3964 R protocol on the port specified as parameter.

Options: **/APPL** reserved to start the protocol for a specific application.
 /lowpri lowers the priority of the server that manages the communication on the C5G side

Syntax: **UCM3 <port_name>**

6.9.8.2.3 UTILITY COMMUNICN PORT_CHAR (UCP)

Used to set the characteristics of a communication device. For example for the serial port the following characteristics can be set:

- Baud rate: 110, 300, 1200, 2400, 4800, 9600, 19200, 38400 baud
- Bit/characters: 7, 8 bit
- Stop bit: 1, 1.5 or 2 bit
- Parity: none, odd, even
- Readahead buffer presence
- Ahd Size: size of readahead buffer) 0 –4096 with steps of 128 (e.g: 0-128-256-...).
Only present when the buffer is present.

- Protocol: none, XON/XOFF
- Character management: none, ASCII, passall, ASCII + passall

The command displays a window containing the current settings. The right-hand and left-hand arrow keys can be used to select the different characteristics. To pass to the next characteristic, press the **ENTER** key, or the right, left arrow keys. With the **upward** and **downward** arrow keys values can be selected for the selected characteristic. After setting the required values for the characteristics, press **PREV/TOP**.

The CTRL C (^C) key interrupts the UtilityCommunPort_char command. If changes have been made, the user is asked if the parameters are to be saved.

This command is not allowed from SYS_CALL.

Syntax: **UCP <port_name>**

6.9.8.2.4 UTILITY COMMUNICATION SET_DEF (UCS)

Assigns the port specified as parameter to the predefined variable \$DFT_DV [5]. It will be used as default communication port .

To use this command, neither memory nor system protections are to be active.

Syntax: **UCS <port_name>**

6.9.8.2.5 UTILITY COMMUNICATION VIEW (UCV)

Displays the characteristics of a communication device.

Options: **/Nopage**
/4 (see [Option /4 to view in the 40 columns](#))

Syntax: **UCV <port_name>**

6.9.8.3 UTILITY LOG branch

LOG command is used to have access to a commands menu that displays the significant events recorded by the system (errors and actions taken by the user).

6.9.8.3.1 UTILITY LOG ACTION (ULA)

Shows the last actions carried out on the Controller. The requested parameter indicates the number of actions to show. If not specified, 20 actions will be shown.

Options: **/Nopage**
 Syntax: **ULA <num_actions>**

6.9.8.3.2 UTILITY LOG ERROR (ULE)

Shows the last errors that have occurred on the Controller. The required parameter indicates the number of errors to be shown. If not specified, 20 errors will be shown.

Options: **/Nopage**
 Syntax: **ULE <num_err>**

6.9.8.3.3 UTILITY LOG LATCH ACKNOWLEDGE (ULLA)

Acknowledges the latched alarms.

This command has to be used to confirm an alarm that is defined as "latched" (that requires confirmation from the user before it can move the robot). The alarms of this category are described in the **PDL2 Programming Language Manual** under the predefined variable \$LATCH_CNFG.

Syntax: **ULLA <alarm>**

6.9.8.3.4 UTILITY LOG LATCH VIEW (ULLV)

Displays the "latched" alarms, i.e. those which have not yet received confirmation from the user before the robot can be moved.

Syntax: **ULLV <alarm>**

6.10 Types of files available in the System

Extension	Description	Backup extension (where applicable)
.ACT	Single file containing list of actions executed on Controller (ASCII format of .LBA files)	.BKA
.LBA	Binary file containing list of actions executed on Controller	.BKB
.COD	PDL2 program code, in binary format. It can be edited in IDE environments, Program Edit, Memory Debug	.BKC
.LBE	Binary file containing list of errors detected on Controller	.BKE
.LOG	FileLOG file, for example ACTION.LOG and ERROR.LOG	.BKG
.LSV	ASCII version of .VAR file	.BKL
.XML	eXtensible Markup Language type file, used to define additional syntaxes on Controller	.BKM
.PDL	ASCII version of a PDL2 program	.BKP
.UDB	User Data Base type file that contains all permitted User Profiles	.BKT
.VAR	Binary file containing variables of a program	.BKV
.VPS	Source file for VP2.Builder	.BKW
.LVP	ASCII version of VP2 file	.BKX
.ZIP	Compressed file	not expected
.C5G	Binary file containing system configuration. It can be loaded or saved in the system memory using Configure , from Setup page (and the corresponding ConfigureLoad and ConfigureSave commands from WinC5G ; for further information, see CONFIGURE branch , Load menu and Save menu).	.BK5
.CIO	Binary file including the I/Os tree	.BKO
.LIO	ASCII of .CIO file	.BKI

7. WINC5G PROGRAM - INTERFACE TO C5G ON PERSONAL COMPUTER

7.1 Foreword

This chapter contains the information concerning the **WinC5G** program that represents the interface on Personal Computer, to Robot Control Unit C5G.

In detail, it describes:

- [WinC5G Activation](#)
 - [Connection to the Robot Control Unit](#)
 - [User interface](#)
 - [WinC5G operating parameters set-up](#)
 - [Commands Menu](#)
 - [Most common problems](#)
-

7.2 Overview

The **WinC5G** program is a Windows style interface on PC to C5G Control Unit. It contains several functions:

- display of files installed on the robot;
- possibility to edit and translate them in executable form (.COD);
- errors search and display;
- possibility to open a Terminal window to directly forward commands onto the Control Unit;
- conversion of already existing programs as a function to new reference points.

WinC5G is contained in the system software CD-ROM. It can run on Microsoft® Windows® operating systems.

It is to be installed on the PC invoking setup.exe program that is in the directory of such a program.

There are two types of licences for the **WinC5G** program:

- basic licence
- full licence:
also enables the handling operations of the file (mirror and linear/rotational shift functions, verification of axes, tool and frame updating), that can be selected from the Handling menu, access from Remote by Proxy.

To pass from a basic licence to a complete licence, follow the steps contained in the paragraph [How to obtain a new licence for WinC5G](#).

7.3 WinC5G Activation

To activate it, just select WinC5G.exe by clicking on the file or its corresponding link.

WinC5G can also be started from the commands prompt. Choose the directory that contains WinC5G program and enter "WinC5G" characters. The program will start the application using the default Properties file (.WCI). To start the application using a specific .wci file, add the access path (associated or complete) to the .wci file (for example "WinC5G c:\line\model1")

7.4 Connection to the Robot Control Unit

The PC connection to the Robot Control Unit is based on the TCP/IP protocol.

The connection modes are the following:

- Ethernet connection to the local network
- Ethernet point to point connection
- Remote connection via Internet
- FTP Server mode connection
- Remote connection by Proxy
- Connection by serial line.

After the connection has been made, start **WinC5G** program ([par. 7.3 WinC5G Activation on page 365](#)) and open the Properties Window ([Fig. 7.39 - Properties Window on page 402](#)). In the Connections section, select the required connection between the following ones:

- TCPIP, for Ethernet and enter the IP address of the APC board installed on the Control Unit to which connection is to be made;
- direct, for connection with serial line, specifying the PC communication port.

It is also necessary to correctly set the other parameters in the Properties Window ([Fig. 7.39 - Properties Window on page 402](#)), described in [par. 7.6 WinC5G operating parameters set-up on page 401](#).

To activate the connection, select File/Connect.

You will be asked to enter the Username and the Password that are to be already defined on the Control Unit by means of the ConfigureControllerLoginAdd command.

If the connection is successful, the user can access the Robot Control Unit opening the Terminal window, display the errors list, etc.

On the C5G Control the communication protocol to WinC5G is automatically installed on the Ethernet communication channel

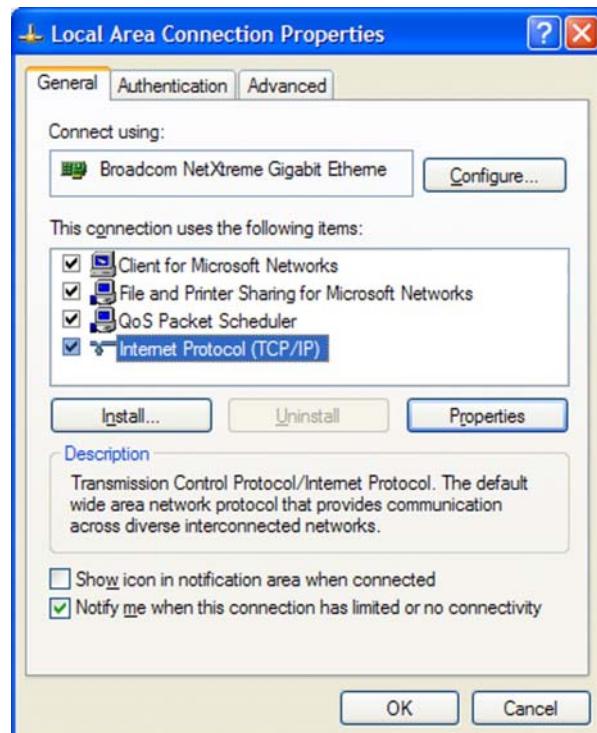
To disconnect from the Control Unit, enter File/Disconnect that will close the connection to the robot and the Terminal window (if open).

7.4.1 Ethernet point to point connection

The Ethernet cable is to be crossed class 5 and is to be connected (with no HUB or Switch network connection device) onto the PC Ethernet port and on Ethernet port of the APC board. The connection can also be executed immediately.

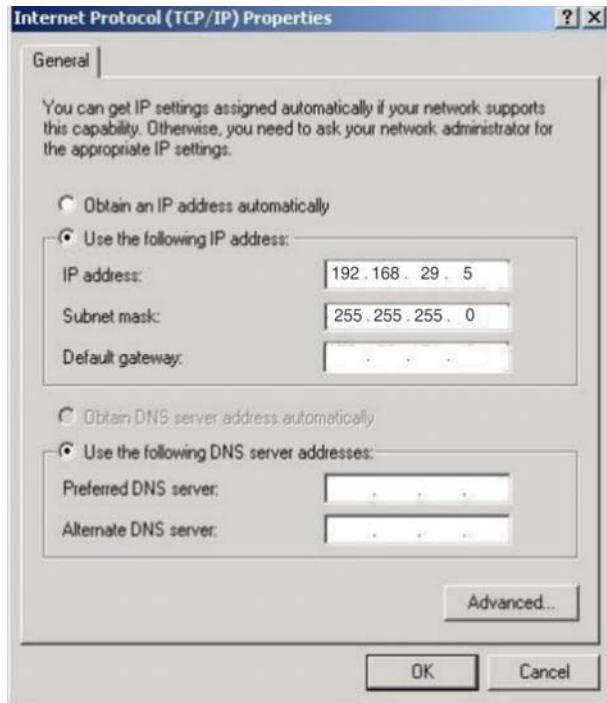
A local network between PC and Control Unit is to be created. From PC select the Properties field of the Internet protocol entering the PC network settings. To do so, select the Start menu, choose Settings, Network and network connections, Connection to LAN local network; select the Properties and Internet protocol key (TCP/IP) (see Fig. 7.1).

Fig. 7.1 - Connection to LAN local network



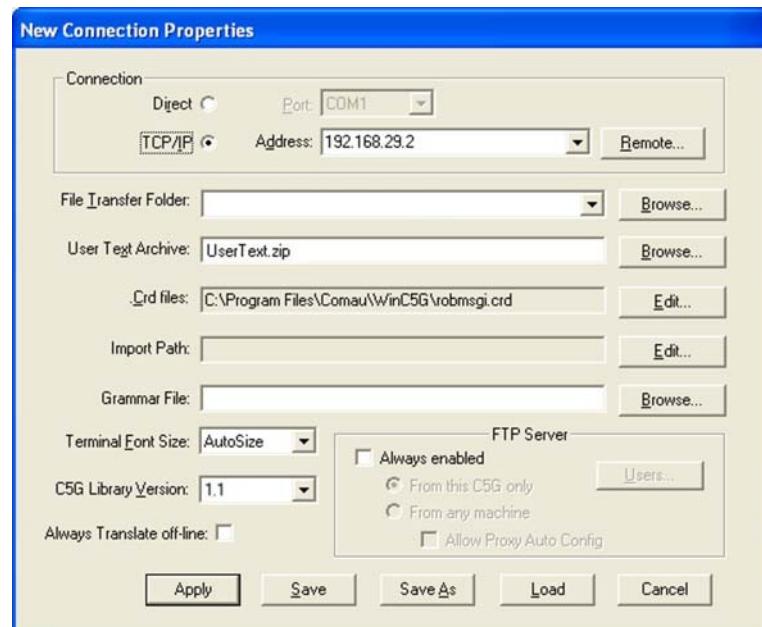
In the Internet Protocol properties:

- the IP address has to have an address on the same Control Subnet.
Example - if the Control has the address 192.168.29.2, in the IP address field 192.168.29.5 can be specified.
- the Subnet Mask must be set to the same value as on the Control. If in doubt, contact the net administrator.
- the predefined Gateway is to remain empty.



Start the **WinC5G** program ([par. 7.3 WinC5G Activation on page 365](#)) and open the Properties window to set the field related to the type of connection (see [Fig. 7.2](#)):

Fig. 7.2 - Enter connection type



- a. select TCP/IP as connection type
- b. in the address field specify that of the Control Unit for the connection. If this address is not known, read the IP address from the [Start Page](#) of the Teach Pendant.

7.4.2 Ethernet connection to the local network

If the Control Unit is already connected to an Ethernet network, it will only be necessary:

- connect also the PC to the network
- from **WinC5G**, access the Properties window and enter the address of the Control as the IP address
- connect to the Control, selecting File Connect command.

7.4.3 Remote connection via Internet

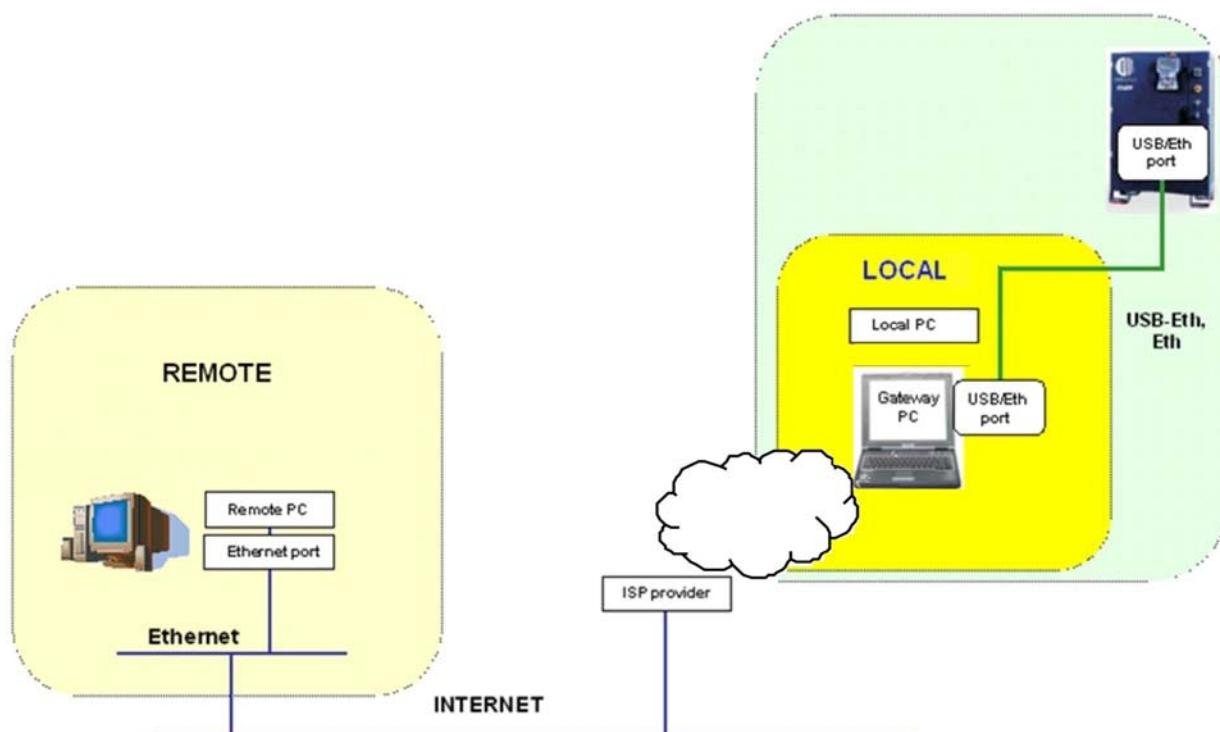
To connect the Controller using Internet, the following equipment is necessary:

- PC with Windows 2000 operating system, or later (NT and 95/98 may not function)
- cable to connect PC to Controller: USB-Ethernet cable or crossed Ethernet cable if Controller Ethernet connection is available
- device to connect to PC.

From Remote side, another PC is necessary to connect to remote Controller. This PC must be visible by Internet.

The layout of the connections described above is shown in [Fig. 7.3](#):

Fig. 7.3 - Remote connection via Internet



The following topics are now described in detail:

- Network configuration
- VPN configuration in Windows XP environment
- Connection procedure.

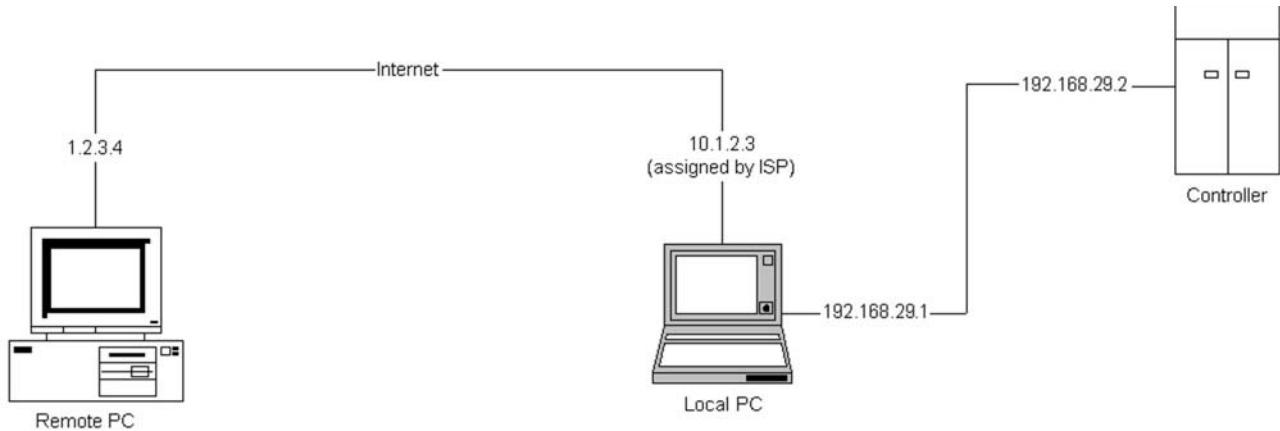
7.4.3.1 Network configuration

To reach the Controller, remote PC, local PC and C5G have to be "informed" on how to send/receive packages on the network.

For example (see Fig. 7.4):

- the Remote PC is connected to Internet with address 1.2.3.4;
- the Local PC is connected to Internet with address 10.1.2.3 (dynamically assigned by ISP, in most cases a private address);
- the Local PC is connected to the Controller with address 192.168.29.1;
- the Controller is connected to the Local PC with address 192.168.29.2 .

Fig. 7.4 - Example of network configuration



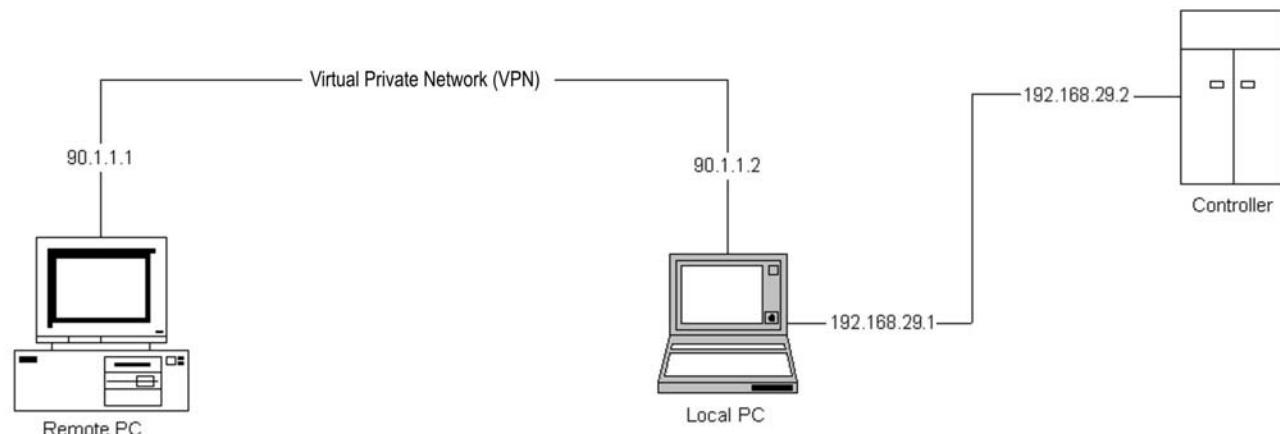
Note that in the configuration of Fig. 7.4, the Internet infrastructure is not aware of the sub-network existing between Local PC and Controller (192.168.29.2), therefore the Remote PC is not able to send/receive packages on address 192.168.29.1.

For this reason a direct connection is necessary between Remote PC and Local PC. This direct connection can be obtained connecting the two PCs via Virtual Private Network (VPN=Virtual Private Network).

With VPN, the Remote PC and Local PC are connected as if on the same network (hence the term VPN).

The above description is illustrated in Fig. 7.5:

Fig. 7.5 - Connection via Virtual Private Network



Note that now Local PC and Remote PC have addresses on the same sub-network.

To complete the installation, the Remote PC has to "know" how to reach the Controller and vice-versa. This is feasible by just executing a direct "route add" on the two machines.

On Remote PC: "routeAdd 192.168.29.2 90.1.1.2"

On Controller: DV_CNTRL #5 equivalent to "routeAdd 90.1.1.1 192.168.29.1

7.4.3.2 VPN configuration in Windows XP environment

- Remote PC configuration (server VPN)
- Local PC configuration (client VPN).



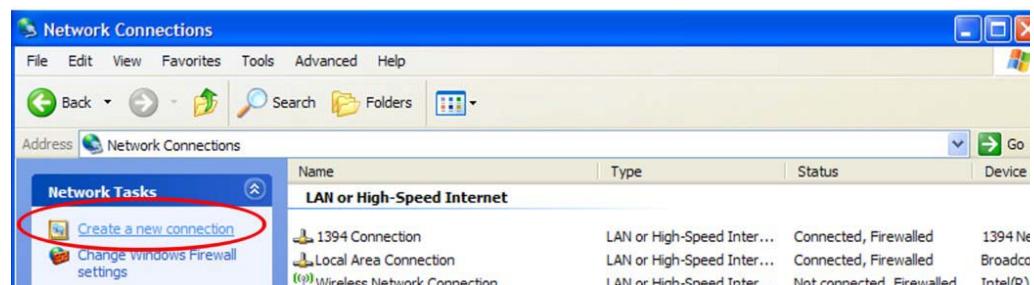
Note: during the description of the procedures that follow, it is assumed that the PC has Windows XP operating system installed and that the user has Administrator privileges.

7.4.3.2.1 Remote PC configuration (server VPN)

The Remote PC is to be configured as VPN server. The procedure is as follows:

- a. open the Control Panel from Start menu,
- b. open "Network Connections",
- c. click on "Create a new connection" (highlighted in red in Fig. 7.6).

Fig. 7.6 - Create a new connection



- d. click "Next" in the first dialogue window that appears.
- e. In the next dialogue window, select "Set up an advanced connection" and press "Next" (see Fig. 7.7)

Fig. 7.7 - Advanced connection installation



- f. In the new dialogue window select “Accept incoming connections” and press “Next” (see Fig. 7.8).

Fig. 7.8 - Accept incoming connections



- g. In the next window, deselect all items present on the screen page and press “Next”.
- h. In the next window select item “Allow virtual private connections” and press “Next” (see Fig. 7.9).

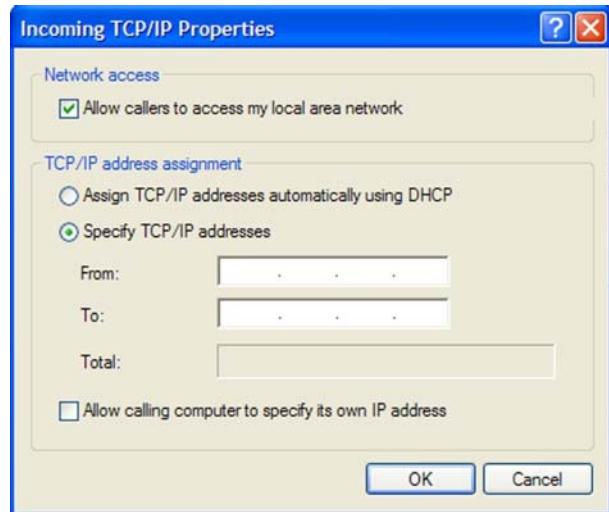
Fig. 7.9 - Allow virtual private connections

- i. In the next window, select (and/or if necessary add) Windows users who have permission to connect to the machine using VPN. This login information must also be known to the Local PC users, when wishing to connect. Press "Next". (see [Fig. 7.10](#)).

Fig. 7.10 - User Authorisations

- j. In the next dialogue window, select "Internet Protocol (TCP/IP)" and click on "Properties";
- k. on the screen page of [Fig. 7.11](#), select items "Allow callers to access my local area network" and "Specify TCP/IP addresses"

Fig. 7.11 - Properties



- I. Deselect item “Allow callers to access my local area network” and insert a range of TCP/IP addresses that are not in conflict with its own IP address. Press “OK” then press “Next”.
- m. On the next screen page (see [Fig. 7.12](#)) press “Finish” to complete the configuration.

Fig. 7.12 - Completion of configuration VPN server



Note: only one server instance at a time can be active on the machine.

7.4.3.2.2 Local PC configuration (client VPN)

The Local PC is to be configured as client VPN.

It is also necessary to enable the IP forwarding: this requires manual modification of the system register and machine restart. To do this, read the note below very carefully.



IP forwarding enable

WARNING! wrong use of the system register editor can cause serious problems that could require reinstallation of the operating system. The use of the system register editor is at the risk and peril of the user.

In Windows XP, IP forwarding is deactivated by default.

To activate it, the procedure is as follows:

- a. Start the system register editor (Regedit.exe).
- b. In the system register editor, find the following key:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters
- c. Set the following register value:
 - **Value name: IPEnableRouter**
 - **Type of value: REG_DWORD**
 - **Value data: 1**

A value equal to 1 permits IP forwarding for all network connections installed and used by the computer.

- d. Close the system register editor.

The Local PC configuration procedure is as follows:

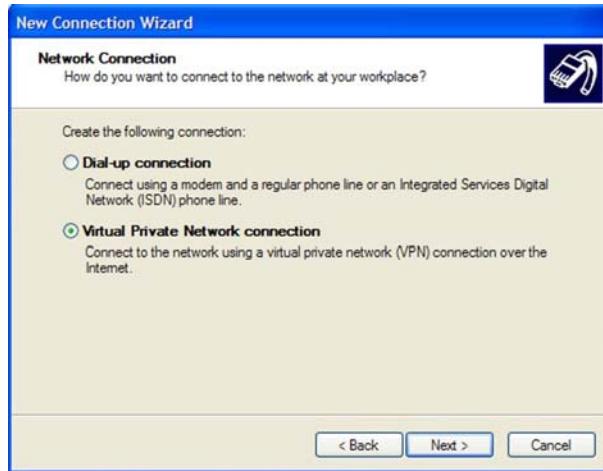
- a. open the Control Panel from Start menu,
- b. open “Network Connections”,
- c. click on “Create a new connection” (highlighted in red in Fig. 7.6),
- d. click “Next” in the first dialogue window that appears.
- e. In the next dialogue window, select “Connect to the network at my workplace” and press “Next” (see Fig. 7.13)

Fig. 7.13 - Connect to the network at my workplace



- f. In the new dialogue window select “Virtual Private Network connection” and press “Next” (see Fig. 7.14).

Fig. 7.14 - VPN connection

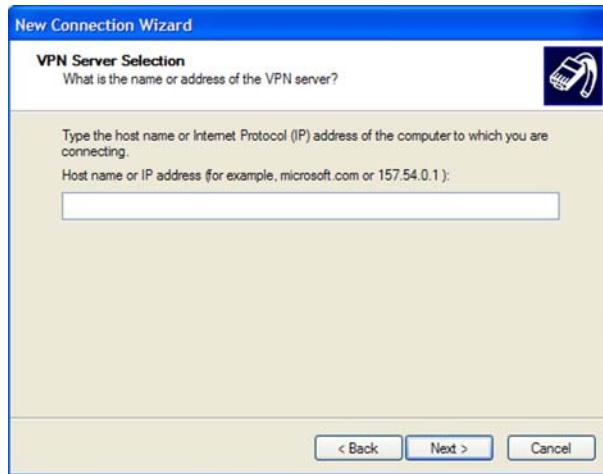


- g. The next window asks for the Connection Name: type in a name at will and press "Next".

Fig. 7.15 - Connection Name



- h. The next window asks for the internet or intranet address of the VPN server for the connection. Insert this information and press "Next" (see Fig. 7.16).

Fig. 7.16 - VPN server selection

- i. In the next screen page select the required item, according to whether the connection may be used by all or has been created for user personal use only. Press "Next".
- j. On the next screen page (see [Fig. 7.17](#)) press "Finish" to complete the configuration.

Fig. 7.17 - Completion of configuration VPN client

The connection is ready to be activated. When connected (see [par. 7.4.3.3 Connection procedure on page 377](#)), the system displays the following screen page (Fig. 7.18):

Fig. 7.18 - Connection



Type Username and Password then press "Connect".

7.4.3.3 Connection procedure

- [Remote PC side](#)
- [Local PC side.](#)

7.4.3.3.1 Remote PC side

From Remote PC side, the system manager shall carry out these operations:

- a. connect to Internet, if not already connected
- b. Launch VPN server
- c. When the Local PC is connected, add a route to the Controller, via VPN server, with the command DOS "route add 192.168.29.2 90.1.1.2".
- d. Insert the Controller IP address (in this example 192.168.29.2), Username and Password.

Connection should be active. **WinC5G**, Ping, FTP can be used for local connection.

7.4.3.3.2 Local PC side

From Local PC side, the system manager shall carry out these operations:

- a. enable IP forwarding (as described in note [IP forwarding enable](#))
- b. connect Local PC to Controller. In our example we suppose that the Controller IP address is 192.168.29.2 and the Local PC IP address is 192.168.29.1.
- c. A route to the Remote PC has to be added on the Controller. In our example the command will be: DV_CNTRL(5, '90.1.1.1', '192.168.29.1').
- d. Connect to Internet
- e. ask Remote PC manager for IP address of the PC address on Internet and launch client VPN

- f. give the Remote PC manager the Controller IP address (in our example 192.168.29.2), Username, and Password.

After connection, delete the route previously added using command DV_CNTRL(6, '90.1.1.1', '192.168.29.1').

7.4.4 FTP Server mode connection

WinC5G can interface to a C5G controller in FTP server mode. This function can be useful in some software installation situations and when interfacing with the C5G controller in FTP client mode.

The configuration of this function is through the [Properties Window](#) (activated from the [Files Menu](#)). Through this window the user can:

- enable/disable the server. By default is it DISABLED. To enable the function, select the "Always enabled" checkbox. Note that before using this command at least one FTP server has to be specified.
- choose whether the FTP server will accept connections only from the C5G Controller set in the Connection Address box. The default is ONLY FROM THIS (C5G).
If the option "From any machine" is selected, the FTP Server will accept connections from other machines or Controllers.
- to configure the Users table (activated by the "Users" softkey in the [Properties Window](#) see [Fig. 7.19](#)) that defines the users to whom access is allowed. A maximum of 10 users is possible

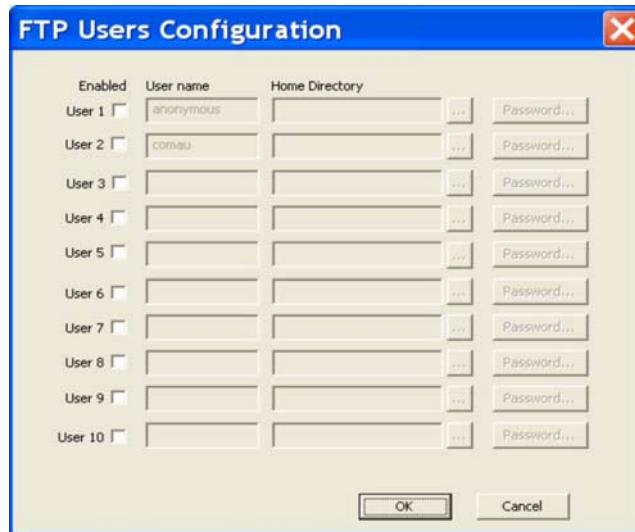
For each user it is necessary to specify:

- whether enabled or disabled
- user's name
- basic directory
this is the home directory when the customer connects to the server
- password
- NET_C - For backup and restore functionalities, with **WinC5G** that acts as server for the Controller, the NETx: devices (with 'x' from 1 to 9) have to be correctly configured. This means that for each NETx: device which is supposed to be used, it is necessary to define a [user's name](#) and a [password](#) (associated to the FTP server that requires the operation) and the [basic directory](#) to be used for the backup/restore operation. Obviously, when the backup/restore operation is requested, the specified user has to be already defined and has already accessed the Controller (login). If NET_C checkbox is selected for a certain user, such an FTP server on the PC is authorized to communicate with the C5G.

The first two users are predefined and for them only the home directory can be set.

The FTP server enables/disables when **Apply** is pressed in the [Properties Window](#).

Fig. 7.19 - FTP Users Configuration

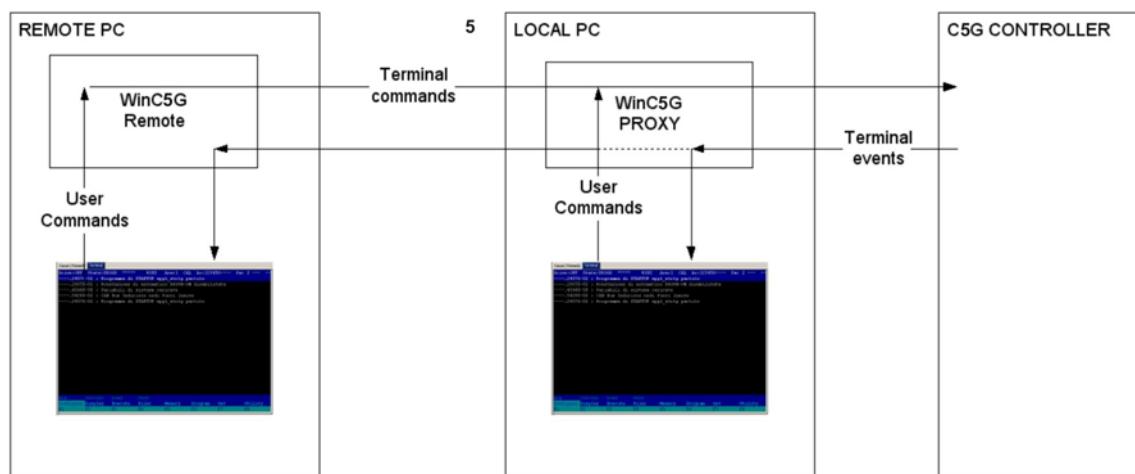


7.4.5 Remote connection by Proxy

To facilitate the work of the Service Engineers, **WinC5G** has two different operating modes:

- Proxy **WinC5G** : the local **WinC5G** acts as "mediator" to the Control Unit
- Remote **WinC5G** : the Remote **WinC5G** connects to the Proxy **WinC5G** to access the Control Unit services.

In this way the two users (local **WinC5G** and remote **WinC5G**) can use the terminal of the same Control Unit simultaneously; the output is repeated on both the Terminal Windows. In this way, the local user can see the operations performed by the remote user and vice-versa (as can be seen in the next figure).



7.4.5.1 Chat Service

The Remote **WinC5G** and the Proxy **WinC5G** are connected by a chat type of system that enables real time communication between local and remote users. Each user has to specify his/her nickname for the chat.

7.4.5.2 WinC5G configuration in Proxy mode



To be able to use this function it is necessary to be in possession of the [full licence](#):

WinC5G Proxy mode can be selected providing no other terminal connection is active. It is configured by the "Remote Access" command from the Tools menu (or through the corresponding key on the Toolbar). The following dialogue window is shown:



Enable: Enables the **WinC5G** Proxy mode

Allow screen disable: authorises the blacking out of the local terminal by the remote user.

Chat user name: name of user for the chat system.

Remote WinC5G address: Remote PC IP address. If this field is set, the Controller configuration is run automatically by Local PC. As a consequence of connection with Controller, the Route (Route Add) is established on the Controller to Remote PC and IP redirection (forwarding) is enabled.

The Proxy service will be active at the next terminal connection to the Control Unit.

7.4.5.3 WinC5G configuration in Remote mode

On the other PC, the remote mode for **WinC5G** can be selected with the 'Remote' key in the Properties window of the Files menu. When this key is pressed, the following dialogue window is displayed:



Enable: enables **WinC5G** Remote mode

Auto Configure: enables automatic configuration of Remote PC with C5G Controller,

executing the following operations:

- Addition of Route, on Remote PC, to arrive to Controller through Local PC
- Addition of Route, on Controller, to communicate with Remote PC, through Local PC
- Enabling of IP redirection (forwarding) on Local PC.



NOTE

Local PC is to be connected and communicating with Controller, when connection with Remote PC is requested.

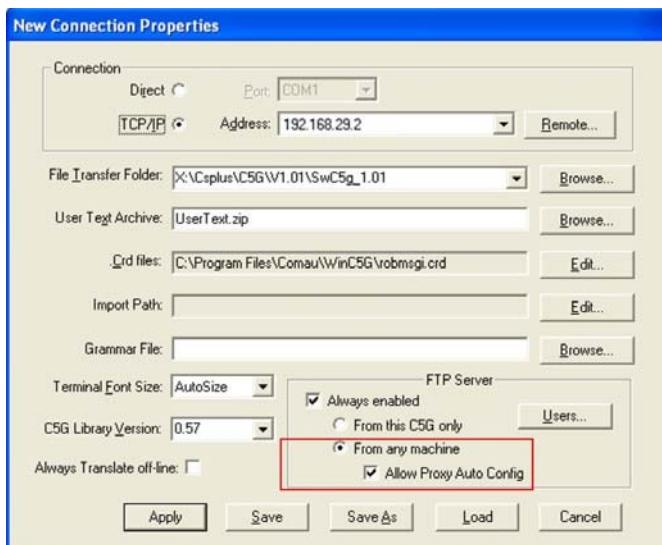
Upon termination of connection, the route add, on Remote PC and on Controller, is removed. Moreover, on Local PC, the FTP server is to be active, with checkbox “Allow Proxy Auto Config” selected (see Fig. 7.20).

Interface: Selection of available network interfaces

Remote WinC5G address: address of Remote **WinC5G** that is acting as Proxy. Remote PC IP address. If this field is set, the Controller configuration is run automatically by Local PC. Upon connection, the RouteAdd on Controller to Remote PC is automatically executed and IP forwarding is enabled.

Chat user name: name of user for the chat system.

Fig. 7.20 - Proxy self-configuration



When the terminal connection is activated, the Remote **WinC5G** is connected to the Proxy **WinC5G**.

7.4.5.3.1 Auto Config

The Remote PC, Local PC and Controller configuration, can be executed in three different modes:

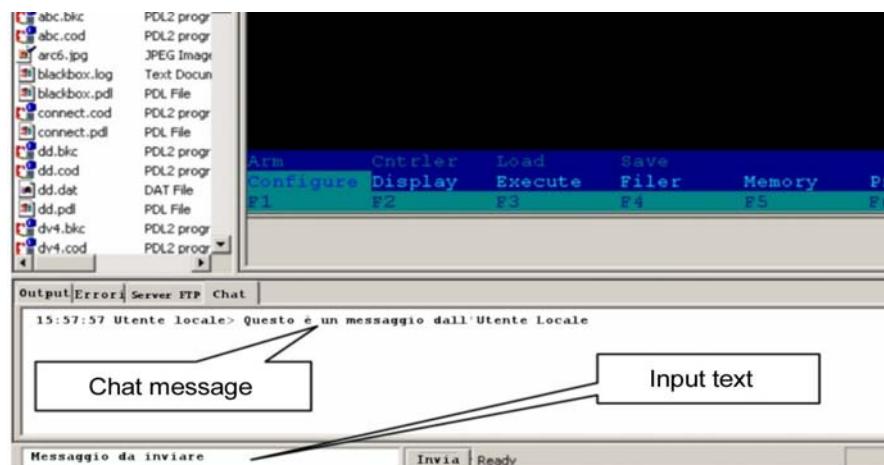
- Manual - the Manual configuration is that described above and requires that the user inserts the route add, IP re-addressing, etc. by hand.

- Semi-Manual : Specified Proxy - Semi-manual means that the configuration on Local PC and on Controller is run automatically by the Local PC. However, the user has to supply the Remote PC IP address manually.
- Completely automatic - means that the configuration of all three devices is executed by WinC5G by a command from Remote PC. To select the completely automatic configuration mode, the following have to be enabled:
 - Local PC - Enable Remote Access
 - Local PC - Enable FTP server and option "Consenti Auto Config Proxy" ([Fig. 7.20](#))
 - Remote PC - Remote mode enabled with option "Auto Config" and correct interface selected
 - Note that the Local PC has to be connected to the Controller and Teach Pendant has to be active, before the Remote PC can perform the necessary configuration.

Obviously, Completely automatic mode is the most convenient, but it is based on the need that the Local PC is active on its FPT server and permits "any machine" to connect, which may not be possible or opportune.

7.4.5.4 Using the Chat Window

When a terminal connection is made, a Chat Window is opened on the [Output Panel](#) on both the Proxy WinC5G and the Remote WinC5G.



The "Chat message" area displays the messages sent by the two users. Each line indicates the time and the sender of the messages.

The "Input text" area is used to type the message to be sent.

The "Invia" key is used to send the message written in the "Input text" area. The message can also be sent by pressing ENTER on the keyboard.

7.4.5.5 Disabling the Terminal

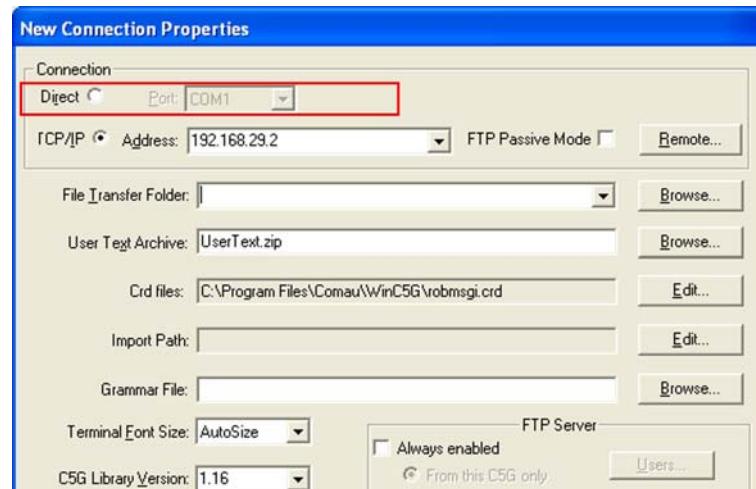
If this has been allowed by the Proxy WinC5G during configuration of the Proxy mode ('Authorise disable'), Remote WinC5G can black out the Terminal window on Proxy WinC5G, using the "Disable Remote" function of the Tools menu (or with the corresponding key on the Toolbar). The system will indicate that disabling has taken place by means of a message in the Chat window.

7.4.6 Connection by serial line

From the Controller side, after dismounting of any other existing protocols, ([UTILITY COMMUNICN DISMOUNT \(UCD\)](#)), issue an [UTILITY COMMUNICN MOUNT C5G_Int \(UCMC\)](#) command and insert, as the communication port, the second serial port on the APC board (called **COM1:** , placed on the left).

Furthermore, the user should have set the **direct** connection in the Property Window , inoltre, aver impostato il tipo di connessione **diretto** nella Finestra delle Proprietà (see [Fig. 7.21](#), highlighted in red).

Fig. 7.21 - Direct connection



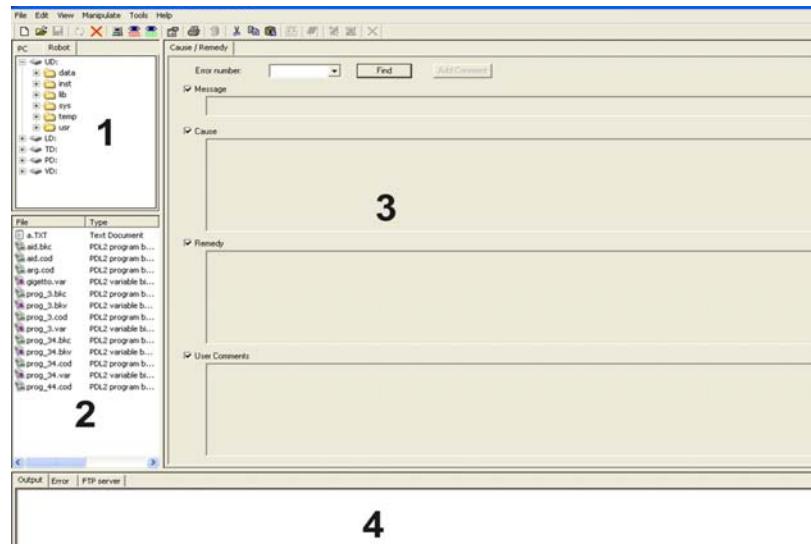
In order to keep the connection after a Controller restart, **\$DFT_DV[8]** variable must be set to the communication logical port which is connected to the serial line.

For further information, please refer to **PDL2 Programming Language** manual - chap.[Predefined Variables List](#).

7.5 User interface

The WinC5G interface consists of four main panels each of which can contain one or more windows with information for the user (see [Fig. 7.22 - User interface panels on page 384](#)).

To move between the panels, the (CTRL + Tab) keys are used; to move among the windows of a certain panel, the (CTRL+Alt+Tab) keys are used.

Fig. 7.22 - User interface panels

1. Directories Panel
2. Files Panel
3. Tools panel
4. Output Panel

7.5.1 Tools panel

The Tools panel is the main panel of the application. It is in this area that most of the information is displayed.

Several application functions may be present on this panel. The possible windows are:

- **The Terminal** displays the information that is on the Control Unit video
- **File Translation** window where the user can display and translate the programs
- **Cause/Remedy** window where the user can read the cause of a certain error and find a possible solution to the problem
- **Errors Display and Actions** window where the user can highlight and put in order the errors and actions generated by the Control Unit
- **Files Manipulation** window that groups together operations for verification and/or variable values conversion (.VAR)
- Window **Viewing and editing of .UDB file (optional feature)** to read and modify user profiles defined in UDB file (optional service).

To close a window on the Tools Panel, send the Close command from the File menu.

7.5.1.1 The Terminal

This is the direct interface with the Robot Control Unit.

To send arm data display commands, to activate/deactivate file management, editing programs (see [Description of commands](#)). When transferring files from PC to Control Unit, it is important to set the directory to be used on PC side. To do so, operate on the [Properties Window](#).

It is also possible to set, still through the [Properties Window](#), the directory that the

program will use to transfer the files from or to the selected device on the Control Unit.

The size of the font can be set through the [Properties Window](#); selecting Automatic mode, the font used by the program will be as extensive as possible to adapt to the screen.

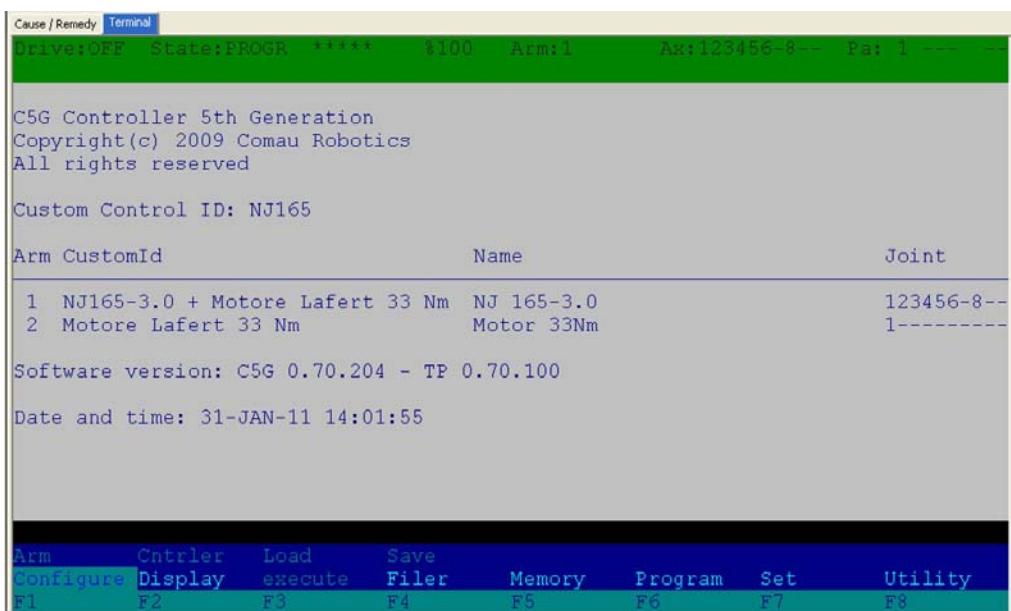
To open the Terminal Window ([Fig. 7.23 - TheTerminal on page 385](#)), it is necessary to first connect to the Control Unit, then open the Terminal (with the Open Terminal command from the View menu or from the corresponding key in the Toolbar).

If another PC is already connected to the same Control Unit, it will not be possible to open the Terminal and a message informing the user will be displayed on the screen.

Connection with WinC5G can be to several Robot Control Units of the same PC, activating the WinC5G program several times; but, as already stated, only one terminal can be active on a single Control Unit.

The Guide/Significance of Terminal keys menu displays the correspondence of the keys between the PC keyboard and some editing functions foreseen by the Control Unit, in Program Edit and Memory Debug environments and on the Terminal pages.

Fig. 7.23 - TheTerminal



7.5.1.2 File Translation

The Translation Window ([Fig. 7.26 - Translation Window on page 386](#)) is used to display, to translate and edit the most part of the files installed on the robot, that is, the files: .PDL, .COD, .LSV, .VAR, .C5G, .XML and their back-up copies (.BKP, .BKC, .BKL, .BKV, .BK5, .BKM).

To open a binary file, from the Files panel, select the name of the file required and send the Open command from the File menu. Otherwise it is possible to click twice on the name of the file to open it directly.

Further information regarding binary files can be viewed. Standard attributes (called Property - see also predefined variables \$PROP_xxx in **PDL2 Programming Language** manual, chapter **Predefined Variables**) associated to the files are as follows:

- filename
- author
- file creation date
- version
- revision
- host where file was created
- title
- help.

When the file is opened, also the Properties are displayed.

Another mode to view Properties of a file is to click with the mouse right-hand key on the name of the files involved, and select “Property Notes” (see Fig. 7.24).

Fig. 7.24 - Property Notes

Cause / Remedy Terminal Property Notes							
Filename	Author	Date	Version	Revision	Host	Title	Help
UD:\c.cod	COMAU	06-Feb-08 18:19:52	3.20.999		NJ_05001		

This viewing also functions on a multiple selection of files, as shown in Fig. 7.25.

Fig. 7.25 - Multiple selection

Cause / Remedy Terminal Property Notes						
Filename	Author	Date	Version	Revision	Host	
UD:\APPL\CL\cl_activate.cod	cmu2910	13-Feb-07 16:15:02	3.10.999		ITGRUTW03758	
UD:\APPL\CL\cl_alarm.cod	cmu2910	07-May-07 15:38:34	3.10.999		ITGRUTW03758	
UD:\APPL\CL\cl_alarm.var		26-Jul-07 14:48:01	3.11.999		NH3_2633	
UD:\APPL\CL\cl_appsetup.cod	cmu2910	04-Sep-07 18:36:42	3.10.999		ITGRUTW03758	
UD:\APPL\CL\clio_cnfg.cod	cmu2910	21-May-07 09:41:44	3.10.999		ITGRUTW03758	

If a binary file is opened (.COD, .VAR or .C5G), this is converted to ASCII, a format that can be read by the user. If errors are found during the translation, these are displayed to the user and the operation is interrupted. In such cases it is best to check that the file is not corrupt, and that it has been generated using a translator software version that is compatible with that used by the application (i.e. lower or same version).

The .C5G files can be displayed, edited but not reconverted to binary format (.C5G).

Fig. 7.26 - Translation Window

```
Cause / Remedy passaman.cod
BEGIN
$TOOL_MASS := 80
$TOOL_CNTR := POS(850, 0, 100, 0, 0, 0, '')
$TERM_TYPE := NOSETTLE

CYCLE
MOVE TO (-46.599998, 53.900002, -54.599998, -108.6,
SIGNAL semmoni
DELAY 550
-- presa per passamano
WRITE LUN_CRT ('Tciclo:', $TIMER[1], NL)
$TIMER[1] := 0
MOVEFLY TO (41.700001, -7.6999998, -136.7, -8.600000
MOVEFLY TO (70.900002, 12.7, -115.5, -3, 59.200001,
MOVEFLY TO (76.800003, 56.400002, -58.700001, 0, 64.
MOVE TO (76.800003, 58.299999, -62.099998, 0, 59.599
DELAY 550
--deposito
MOVEFLY TO (76.800003, 56.400002, -58.700001, 0, 64.
MOVEFLY TO (70.900002, 12.7, -115.5, -3, 59.200001,
```

The files can be translated from Binary to ASCII, selecting the required .COD file and entering the Translate command from the Edit menu. In this way a .PDL file is created that the user can open in the Translations Window. With the same command (Translate from the Edit menu) it is possible to ask for the translation from ASCII to binary: in this way the system creates a new .COD file and saves the previous version in a .BKC file.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

Possible translation errors will be displayed in the Output Window.

The only exception is the .C5G file that cannot be translated and saved in ASCII, but can only be displayed.

Since the PDL2 language is continually evolving, the translator is subject to version updating. When necessary the translator version number to be used has to be set again in the [Properties Window](#) in the 'C5G Library Version' box. The number to be specified is to correspond to the C5G<v_xx> file that is part of the WinC5G installation. The application must be closed and opened again to load the new library of functions.

7.5.1.3 Cause/Remedy

The Cause/Remedy Window ([Fig. 7.27 - Cause/Remedy Window on page 388](#)) is used to find information on the cause of a certain error and how to solve the problem.

The user has to specify the number of the error and enter Find.

The application looks for the error in file with the extension .CRD indicated in the [Properties Window](#) at the Select .CRD key. The files containing all the errors and their causes and solutions are divided by language and are called:

- allrobmsgd.crd - German
- allrobmsge.crd - English
- allrobmsgf.crd - French
- allrobmsgi.crd - Italian
- allrobmsgp.crd - Portuguese
- allrobmsgs.crd - Spanish

The ones for applications can be added to them, if and when they are distributed.

- aw_ms.crd - arc welding
- cl_ms.crd - common libraries
- gl_ms.crd - glue
- hn_ms.crd - handling
- st_ms.crd - stud
- sw_ms.crd - spot welding
- tc_ms.crd - tool change.

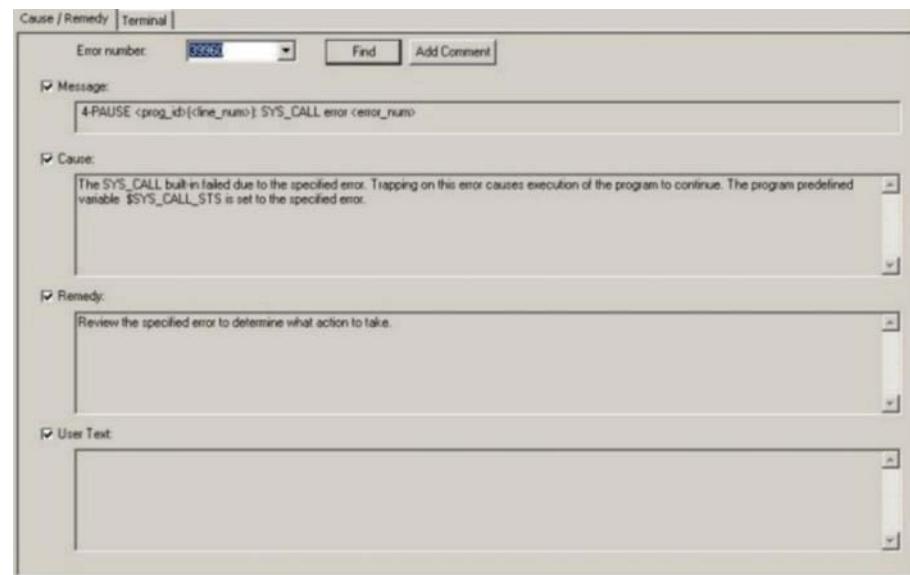
To these, those of the applications can be added if they are distributed. Through this dialogue box a .CRD file can be added or removed (for example one that is connected to a specific application) to the error list. The maximum number of CRD files that can be selected is 10.

The user can customize the comments associated to a certain error through the Add Comment key on the Cause/Remedy page. A dialogue box is opened to enter a text that will be displayed with that error.

This text will be saved in the specified .ZIP file that has been entered in the [Properties Window](#), in the User Texts Archive box.

The display of the User Text can be enabled or disabled through the checkbox.

Fig. 7.27 - Cause/Remedy Window



From WinC5G all the C5G errors contained in the selected ROB*.CRD file can be saved in a file or printed as a list. This operation requires a few minutes.

The related commands are 'File Print All' and 'File Save All as'.

It is indispensable that these commands be sent when the Cause/Remedy page in the Properties Window [Properties Window](#), is active.



Please, note that, in case of errors identified by SYSTEM ERROR attribute, neither causes nor remedies are displayed, since they are internal errors.

In such situation, please contact COMAU Technical Service.

7.5.1.4 Errors Display and Actions

The contents of the binary files (.LBE and .LBA) with the list of the errors that have occurred on the Control Unit and the actions taken by the user and recorded on the Control Unit can be viewed.

To do so, send the appropriate command from the View menu : 'View Errors from Robot' and 'View Actions from Robot'; the contents of all the error and action files in the LOG system directory can be viewed as a list.

The errors list is displayed in the Errors and Actions Window, shown in [Fig. 7.28 - Errors and Actions Window on page 389](#).

This list can be in the order of the dates, error numbers, seriousness of the error, message order number. It is possible to open simultaneously both the errors and actions files and, in this case, the contents are shown in a manner to correlate the sequence of the actions with the occurrence of the errors.

There is an error screening mechanism that allows the user to narrow down the range

of data display on the basis of an interval of time, the error number, seriousness of the error, etc. The filter setting can be activated from the Edit Menu or the right-hand key of the PC Mouse.

Fig. 7.28 - Errors and Actions Window

Cause / Remedy	CNTRLCS5G_400	C5G	CNTRLCS5G_35	CNTRLCS5G_403	CNTRLCS5G
Timestamp	Count	Error / ...	Severity	Message	
14/04/2011 08:52:17	9237e	43012	4	Sample error	
14/04/2011 08:52:18	9241e	43016	4	Sample error	
14/04/2011 08:52:18	9240e	43015	4	Sample error	
14/04/2011 08:52:18	9239e	43014	4	Sample error	
14/04/2011 09:18:50	9242e	43017	4	Sample error	
14/04/2011 10:03:04	9243e	37070	4	alarm2(19): e	
14/04/2011 10:04:25	9244e	37070	4	alarm2(19): e	
14/04/2011 10:05:09	9245e	39977	4	alarm2(32): e	
14/04/2011 10:09:26	9246e	39977	4	alarm2(33): e	
14/04/2011 10:15:04	2033e	8193	10	Mandatory m	

7.5.1.5 Files Manipulation



If it is wished to be able to have all files handling functions, It is necessary to have software option WinC5G Full.

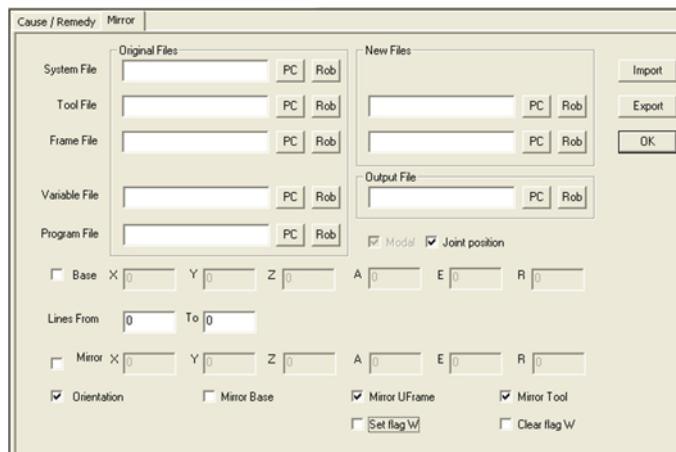
For the Comau codes of the options, see [Cap.12. - Appendix - Software Options on page 494](#).

The Manipulation menu contains features that check and convert the values stored in a .VAR file.

The functions are:

- check that the positional values stored in a .VAR are included in the range of the stroke end limits ([Axis check](#)).
- modification to positional values stored in a .VAR, based on new values of \$TOOL and \$UFRAME predefined variables ([Tool and Frame update](#))
- linear shift of positional values stored in a .VAR. ([Linear Shift](#)).
- linear shift and shift in the orientation of positional values stored in a .VAR ([Linear Rotation](#)).
- mirror referred to a plane ([Mirror](#)).
- modification to positional values stored in a .VAR basing on changes to calibration data ([Calibration](#)).
- modification to positional values stored in a .VAR after modifications to the robot kinematics ([Sysdata adjust](#)).
- modification of positional values stored in a .VAR after a wrong Turn Set operation ([Turn difference](#)).

In the Manipulation menu the ‘Settings’ page allows to define the properties which are common to the pages shown in the Manipulation menu (see [Fig. 7.35 - File Manipulation settings on page 397](#)).

Fig. 7.29 - File Manipulation Window

The parameters which the user must insert before executing one of the listed above functions are grouped in different areas (as an example, see [Fig. 7.29 - File Manipulation Window on page 390](#)):

- Original files area: the user must select (via the PC or Rob text box), either on the PC or on UD: controller device, the files that are needed as input to the execution of the selected function. The files must be binary and not ASCII because the manipulation functions do not work on ASCII format. The files of interest are: the configuration file (.C5G), the (.VAR) file containing the values of the Tools and the Frames, the user program files (.COD and .VAR).
- Output files area: the user must select the name of the .VAR file to create as result of the operation.
- New files area: the user must specify if needed, the new files to use in the operation as an alternative to the files specified in the Original files area. This window is only active for the 'Tool and Frame update', 'Calibration', 'Sysdata adjust' pages.
- an instruction of the following type
`MOVE TO p1`
 preceded by
`ToolFrame(1,1)`
 is only handled if the "Modal" flag is selected.
- Data related to the Base: the user must specify in these text boxes the values which define the cartesian coordinates of the position to be used as \$BASE if the conversion must consider a \$BASE that is different from the one defined in the .C5G configuration file.

It is possible to perform the conversion operation considering the whole .COD file or a section of it; in this case the starting (Lines from:) and ending (to:) lines must be specified. Otherwise the whole file will be computed.



WARNING! The application does not carry out the operation until all the required parameters are specified.

Please note that:

- the application will not perform the operation until all requested parameters are specified
- the setting for a specific operation can be saved in a file and reused in successive

commands. This avoids the need to enter the parameters each time the operation is repeated.

Use the **Import** box to choose a .WCF file already present; the path will be asked for to take the file.

Use the **Export** box to save a file with the required settings in a certain directory. A file will be created with the .WCF extension.

For all the functions of the Manipulation menu, input parameters to be ALWAYS specified are:

- System file: name of the .C5G file.
- Tool file: name of the .VAR file containing the tool values used by the program (for example UD:\DATA\ATT_TOOL.VAR).
- Frame file: name of the .VAR file containing the frame values used by the program (for example UD:\DATA\TU_FRAME.VAR).
- Variable file: name of the .VAR file of the user program to be converted.
- Program file: name of the .COD file of the user program to be converted.

Other parameters could be needed depending on the operation to be performed.

Here below the features belonging to the Manipulation menu are described in further details:

- Axis check
- Tool and Frame update
- Linear Shift
- Linear Rotation
- Mirror
- Calibration
- Sysdata adjust
- Turn difference
- Dialogue box for File Manipulation settings.

7.5.1.5.1 Axis check

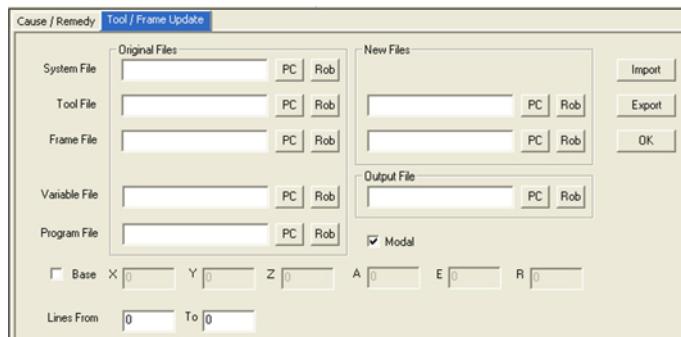
This operation allows to check that a particular axis does not overcomes the foreseen limits during the operations performed by a program on a specific robot.

- Axis: it is the axis on which the check must be done.
- Range from ... to: contains the real values (degrees/axis) defining the limits, for the specified axis, which should not be overcome in the positional variables of the program. For example from 0.0 to 30.0 for axis 1.

In case some limits are overcome, informational messages are displayed in the output window of WinC5G.

7.5.1.5.2 Tool and Frame update

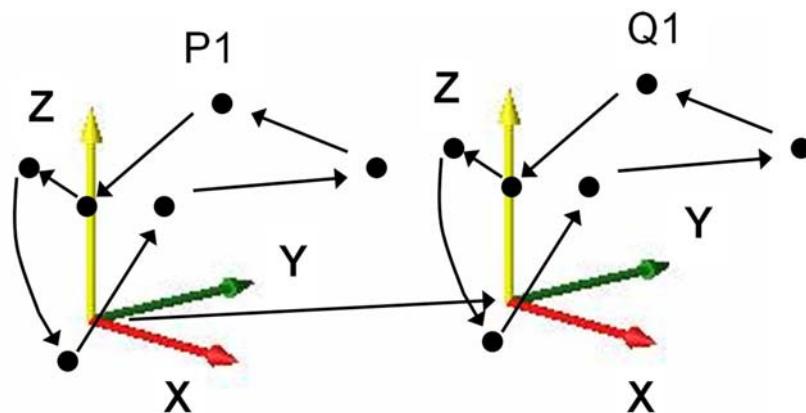
This operation modifies the positional values of an existing program basing on new values of the tool and/or frame.

Fig. 7.30 - Tool and Frame update

- **New files:** name of the files containing the new values of the tool and the frame to be used in the conversion of the user program. Both parameters must be specified; if the conversion only concerns the tool, the frame file specification must be the same inputted in the Original Files area (and viceversa).
- **Output file:** the user must define the name of the .VAR file to create, containing the positional data modified basing on new tool and frame values.

7.5.1.5.3 Linear Shift

This operation performs a linear translation of the positions of an existing program.



P1 - starting position

Q1 - end position

- **Start and end positions:** the user must specify the X,Y,Z coordinates of the two points which define the distance to use in the translation.



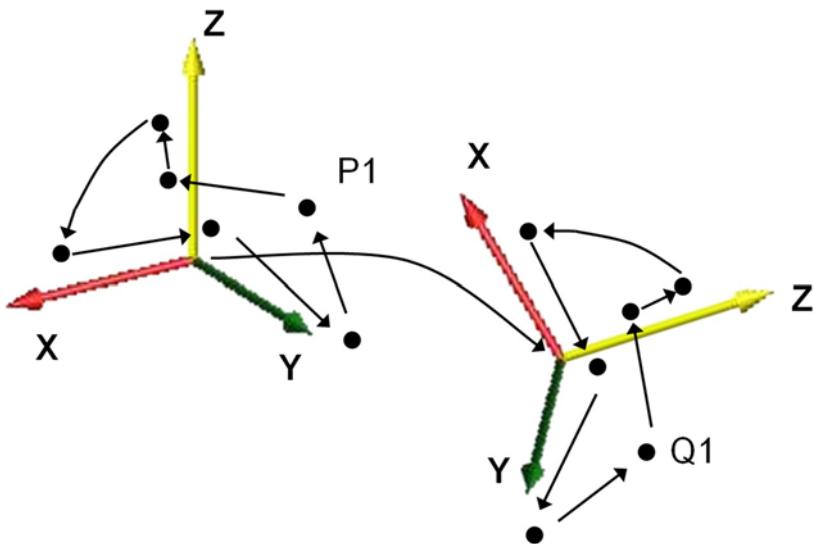
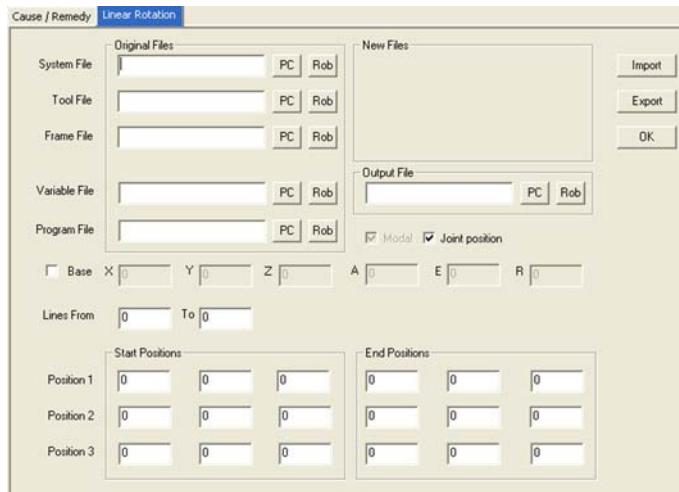
P1 and Q1 may NOT belong to the trajectory.

- **Output file:** the user must define the name of the .VAR file to create after the conversion applied by the translation.

7.5.1.5.4 Linear Rotation

This operation performs a roto-translation of positions inside an existing program.

Fig. 7.31 - Linear Rotation



P1 - starting position
Q1 - end position

- **Start and end positions:** the user must provide three starting positions and three ending positions which define 2 planes. A linear translation from the first final position is performed first; then the orientation relative to the final plane is considered.

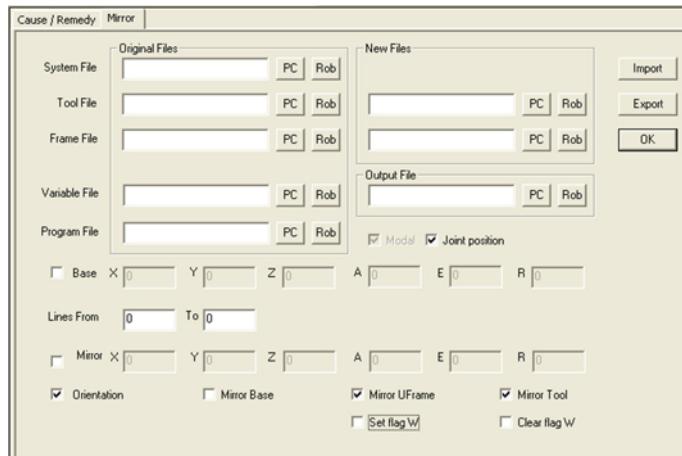


P1 and Q1 may NOT belong to the trajectory.

- **Output file:** the user must define the name of the .VAR file to create, containing the positional data modified accordingly to the roto-translation defined by the shift operation.

7.5.1.5.5 Mirror

Fig. 7.32 - Mirror

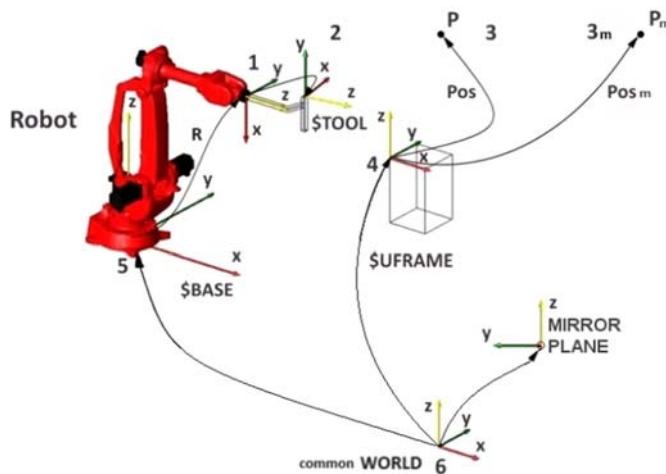


Mirror functions are similar to the Shift operations, but include a symmetric transformation related to a plane. The X-Z plane of the robot is the default plane.

It is possible to obtain the mirrored trajectory, for either a robot working on two different sides of an object (see [Fig. 7.33 - Case A - Single Robot working on one part on page 394](#)), or a second robot working on the other side of an object which is common to both the robots (see [Fig. 7.34 - Case B - Two mirrored Robots working on the same part on page 395](#)).

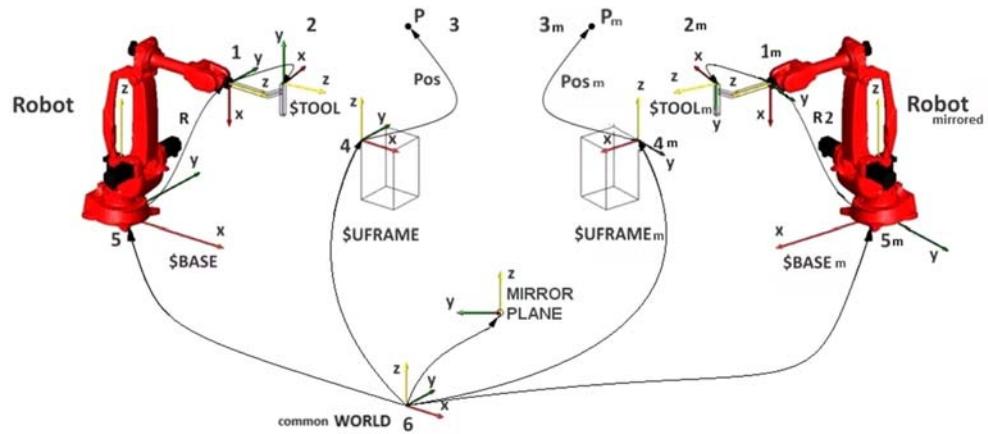
In some specific situations it is also allowed to individually select the reference frames (see [Fig. 4.1](#) in **Motion Programming** manual) which are involved in the mirroring operation.

Fig. 7.33 - Case A - Single Robot working on one part



In such a case, the mirrored trajectory refers to the same \$BASE, \$UFRAME and \$TOOL.

Fig. 7.34 - Case B - Two mirrored Robots working on the same part



In the shown above situation, \$BASE and \$UFRAME are mirrored with respect to the specified mirror plane; whereas \$TOOL is mirrored with respect to the Robot flange XZ plane.

The editable fields, in the Files Manipulation Window (see Fig. 7.29), are as follows:

- **Orientation:** this box must be checked when the orientation has to be considered during the symmetric transformation. Otherwise the transformation will only be related to X, Y, Z values.
- **Mirror Plane:** the robot XZ plane is defined by default. For using another plane, please define the coordinates in these edit boxes.
- **Mirror Base:** when selected, the positions are mirrored so that they can be reached by a robot with the Base mirrored with respect to the specified mirror plane (see the NOTE about [Reachability of the mirrored positions](#)). In Fig. 7.34, the \$BASE reference frame is mirrored in \$BASEm. \$TOOL and \$UFRAME change according to the other fields. The newly calculated Base is provided as an output in the New Base fields
- **Mirror Frame:** when selected, the positions are mirrored so that they can be reached by a robot with the User Frame mirrored with respect to the specified mirror plane (see the NOTE about [Reachability of the mirrored positions](#)). In Fig. 7.34, the \$UFRAME reference frame is mirrored in \$UFRAMEm. \$TOOL and \$BASE change according to the other fields. A new TU_FRAME.COD file is provided as an output, including the new User Frames table.
- **Mirror Tool:** when selected, the positions are mirrored so that they can be reached by a robot with the Tool mirrored with respect to the specified symmetry plane of the robot flange (see the NOTE about [Reachability of the mirrored positions](#)). In Fig. 7.34, the \$TOOL reference frame is mirrored in \$TOOLm. \$UFRAME and \$BASE change according to the other fields. A new TT_TOOL.COD file is provided as an output, including the new Tools table.
- **Joint Position:** mirror is applied by default to positional variables of POSITION and XTNDPOS data type. If also JOINTPOS should be involved, this checkbox should be selected. In case of JOINTPOS, Mirror function is only applied to the values of axis 4 and 6 of the robot.
- **Output file:** the user must define the name of the being created .VAR file, containing the positional data modified accordingly to the required symmetric transformation.

- Set Flag W and Clear Flag W: if selected (they are mutually exclusive), they respectively, either set or clear the W configuration flag of the mirrored position; if not selected, the configuration flags of the mirrored position are the same of the original position ones

As an example, the picture in [Fig. 7.33](#) was obtained with Mirror Base, Mirror Tool and Mirror Frame fields NOT selected.

The one in [Fig. 7.34](#), instead, was obtained with all such fields selected.



Note that multiturn flags, if any, are automatically inverted by the Mirror function.
E.g.: 'WS T2:-2' becomes 'WS T2:+2'.



Reachability of the mirrored positions

The reachability of the mirrored positions is not guaranteed: it depends upon both whether it is placed inside the working area (if not, a warning message is issued), and the approaching MOVE type to the position itself and the configuration flags. As far as such two last conditions, no warning messages can be issued, during mirroring phase. They can anyway be issued at program execution time.

7.5.1.5.6 Calibration

This operation is used for correcting a program in case the variable which stores the calibration constant changes.

- New files: the user must specify the name of the .C5G configuration file, which contains the calibration values (\$CAL_DATA[axis]) to use in the conversion of the .VAR.
- Output file: the user should define the name of the .VAR file to create after the conversion of the positional data basing on the new calibration constant.

7.5.1.5.7 Sysdata adjust

This operation must be used for correcting a file of variables in respect to a new robot kinematics.

- New files: specify the new .C5G configuration file to use during the conversion.
- Output file: the user has to define the name of the .VAR file to create, containing the positional data modified accordingly to the new kinematics.

7.5.1.5.8 Turn difference

This operation must be done, after a wrong Turn Set operation, for correcting the number of turns of a specific axis and for generating a new file of variables.

The user must specify the axis that is subject to the modification and the turns difference to apply to the positional data of the program.

In order to do that the robot must be moved to the calibration position (MOVE TO \$CAL_SYS); the number of turns must be read on the programming unit (Revolution information in the Service page); the axis must be moved in correspondence of the reference heel; the motor turns information must be read. Execute the difference of the integer side only. Insert the computed value in the 'Difference' field.

If the difference value is negative, the 'Negative difference' checkbox must be selected.

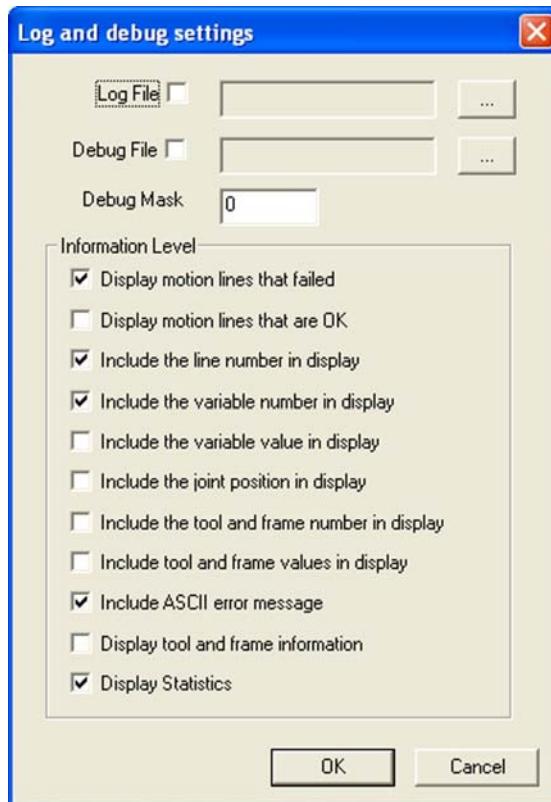
- **Output file:** the user must define the name of the .VAR file to create, containing the positional data modified accordingly to the requested correction of the number of turns.

7.5.1.5.9 Dialogue box for File Manipulation settings

The Settings dialogue box ([Fig. 7.35 - File Manipulation settings on page 397](#)) is opened by the Settings option from the Manipulation Menu; it is used to set the level of information that is generated during manipulation operations on the files and to define where the files recording the actions are to be stored (Log File).

If the boxes of the Log File and the Debug File are labelled, it will be possible to specify where these files are to be created.

Fig. 7.35 - File Manipulation settings



The following information available for the user to choose from to see in the log file:

- Wrong movement lines.
Displays in the log file the movement lines on which the data conversion generates an error.
- Correct movement lines.
Displays in the log file the movement lines that have been properly converted.
- Line number display.
Allows the application to add the line number next to each movement instruction written in the log file.
- Variable Name.
The application adds the variable name to the log message to inform the user which variable has been manipulated.

- Variable Value.
The application adds the value of the manipulated variable to the log message.
- Joints position.
Adds the position of the joints in the message generated for the manipulation of a movement instruction.
- Tool and Frame Value.
The Tool and Frame values can be viewed in the log file where an instruction has been manipulated.
- Error message.
Enables displaying of error messages that have been generated during the manipulation of an instruction.
- Tool and Frame value.
The data referring to the Tool and the Frame is added in the log file.
- Statistics.
The summarized data on the executed operation (for example, the number of changed instructions) is added to the log file.

7.5.1.6 Viewing and editing of .UDB file (optional feature)

With this function, clicking on the .UDB file (or.BKU) displayed on WinC5G file listing (on PC or on Control), it is possible to read the user profiles defined in it and if necessary modify them. However, it is not possible to read the UserId and the Password.

If WinC5G Full is installed, it is also possible to create a new user profile to be stored in the UDB, sending the **New** command from the right-hand key.

Fig. 7.36 - Addition of a new user

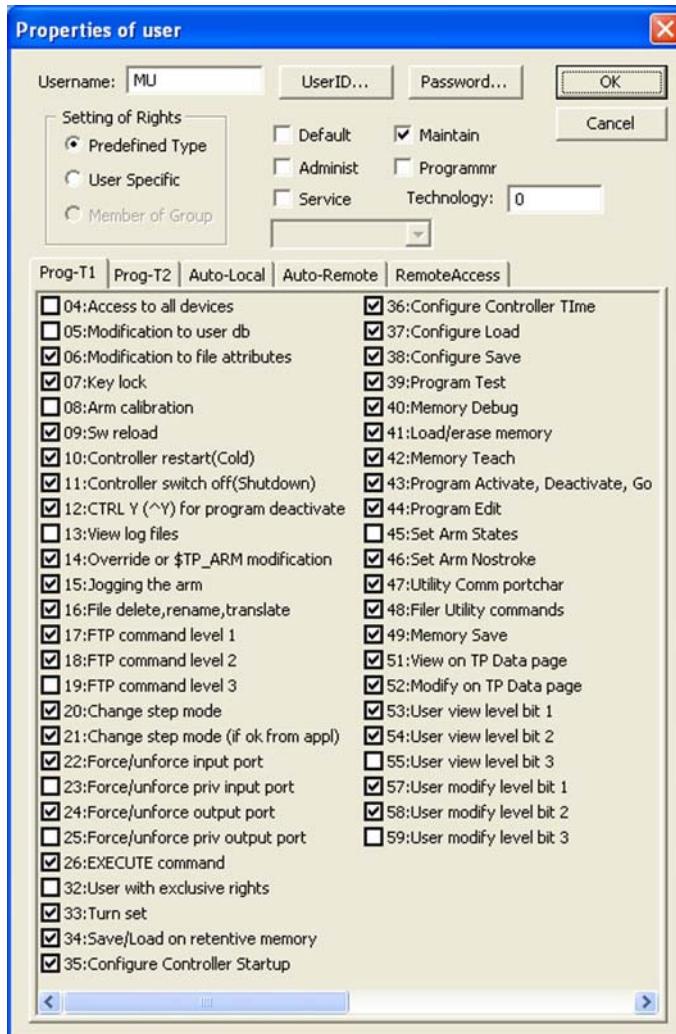
Causa / Rimedio Terminale		NH4_06001.udb
Nome Ut...	Tipo di utente	Diritti Technology
MU	Maintain	0
PU	Programmr	0
ADMIN	Administ	0
<input type="button" value="Edit"/> <input type="button" value="Nuovo"/> <input type="button" value="Cancella"/>		

First a window is shown that, as first screen page, displays the default profile with the operations enabled for the profile, corresponding to the different states of the system (see Fig. 7.36).

If it is wished to modify the characteristics of the new user profile to be created, in relation to the default, this can be done by selecting one of the standard profiles provided by the system (Administ, Service, Maintain, Programmr, Technology) or one which has been completely customised.

It is possible to act on the checkboxes shown in Fig. 7.36, enabling/disabling the various operations available.

Fig. 7.37 - User profile properties window



The User view level and User modify level bits indicate, respectively, the viewing and modifying level. They are used to allow either viewing or modifying a Program in [IDE Page](#) on the TP, comparing them to the corresponding file property.

7.5.2 Directories Panel

The Directories panel shows the structure of the PC and the Control Unit directories (for the latter, if the connection is active)

(see [Fig. 7.22 - User interface panels on page 384](#)).

Only the directories on the devices are displayed, NOT the files they contain.

The Directories panel has one or two windows according to whether the connection to the Robot Control Unit is active or not.

If the connection on the Control is not active, only one window is visible, that indicates the disks present on the PC with the corresponding directories. If instead the connection is active, the structure of the directories on the UD: disk of the Control Unit is shown.

The display can be updated using the "Display" menu, "Update list of files" command.



The list of the devices is read once only, during connection. If devices are added or removed during the connection (for example Disk-on-Key on the Control Unit or Windows network drives on PC), the Directories Panel will not be updated with these devices. To run the list of devices refresh, close the current session and reconnect.

7.5.3 Files Panel

The Files panel shows the list of the files contained in the device or folder that has been selected on the Directories panel (see Fig. 7.22 - User interface panels on page 384).

For each file the information is shown about name, type, size and revision date. The items can be sort according to one of these categories.

Further information can be viewed (Properties) about the files, right clicking the mouse key and selecting "Property Notes" (see also predefined variables \$PROP_xxx in **PDL2 Programming Language** manual, chapter **Predefined Variables**).

Standard attributes associated to the files are the following:

- filename
- author
- file creation date
- version
- revision
- host where file was created
- title
- help.

The files can be opened by clicking on the name of the relevant file (in the open file also the Properties are displayed); and they can be closed through the Close command in the Files menu (or by the related key in the Toolbar).

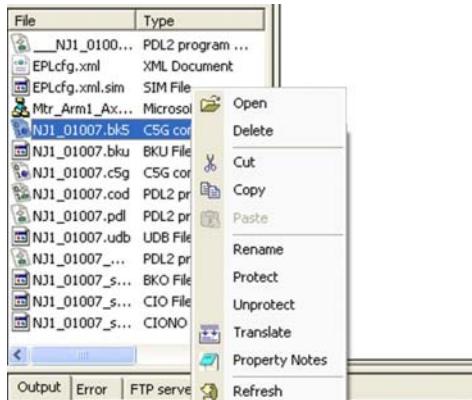
To open several files, they can be selected by highlighting them and subsequently sending the Open File command from the menu, or from the tool bar, or clicking with the mouse right-hand key. If one of the selected files is not acknowledged, it will be ignored.

There is also the **drag and drop** function that the user can use to:

- "drag" a file from Windows Explorer to the file edit/view window. The file will be opened or transferred to the selected directory. The transfer operation to the Controller, in some cases, could require a few seconds.
- "drag" a file or a directory from the files panel or the **Directories Panel** to Windows Explorer. In this case, the file/directory will be transferred from the selected directory to the local file system.

It is possible to protect a file, currently selected from the files panel, by right clicking on the filename using the mouse key and then choosing 'Protect' command.

Fig. 7.38 - Further commands related to Files



A password is required to be used when removing the protection ("Unprotect").



The same operation can also be carried out from the Tools menu ("Protect" and "Unprotect").

7.5.4 Output Panel

The Output panel displays the messages that an application generates to the user. The main use is to show the result of the program files translation, such as .PDL and .LSV, including the translation errors and messages coming from the file manipulation operations

(see [Fig. 7.22 - User interface panels on page 384](#)).

To open this Panel, use "Show Output Window" in the View menu; to close it, use "Close Output Window" from the same menu.

The output panel contains four windows.

The first displays messages generated during the translation, with indication of the translated files and the translation results.

The second window shows the errors found in the last translation performed.

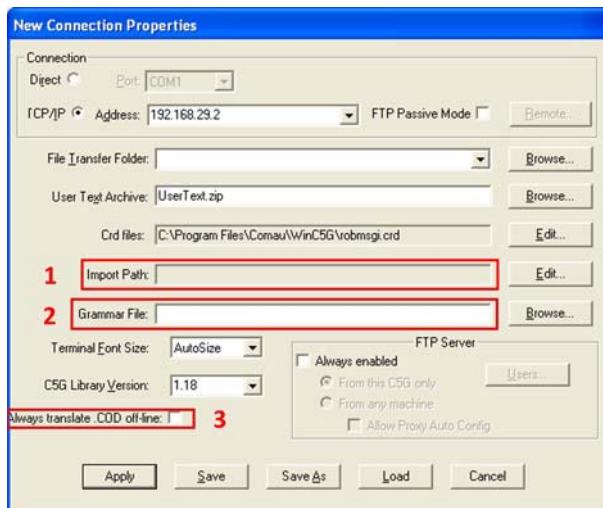
Clicking twice on an error, the file is open on the line where the error has been found.

The third window displays messages generated by the FTP server.

The fourth window is used for chatting, if WinC5G is connected as Remote or as Proxy (see [Remote connection by Proxy](#)).

7.6 WinC5G operating parameters set-up

The Properties Window ([Fig. 7.39 - Properties Window on page 402](#)) that is used to set the operating parameters of the WinC5G application is of particular importance. It is activated from the [Files Menu](#), selecting [Properties](#).

Fig. 7.39 - Properties Window

- **File Transfer Directory**

It is used by the Terminal window during the file list transfer/display operations from and to the connected PC (FilerCopy, FilerView, etc.).

- **User Texts Archive**

It is a compressed file that stores the comments added by the user regarding the single error messages in the Cause/Remedy window. It is a .ZIP file.

- **.Crd file**

It allows the addition and the removal of a .crd file to/from the list of files where the search is to be made for the cause and remedy of the errors from the Controller. The files may be in any device on the network, not necessarily in the PC where the application is running; there could be an application slow-down if the files are on a server.

- **Import Path (1 - Fig. 7.39)**

To translate PDL programs that contain the **IMPORT** clause, read the declarations regarding variables, types and imported routines, in the files where they are contained.

If checkbox "Always translate .COD off-line" (3 - Fig. 7.39) is selected from the Properties window, library **c5gx_xx.dll** will be used that is part of **WinC5G** program installation (even if ON-line).

In case of **off-line** translation, all .COD files containing the declarations, must be loaded on the PC in the location specified in the Properties window in **Import Path** text field. If this path is not set, **WinC5G** program will check **C5G_IMPORT** variable, that must have been previously set on the PC. For example, define: if the Controller is connected, by default the files opened by **WinC5G** are **on-line translated**; if on the PC, sent to the Controller via FTP server, translated then copied again onto the destination device. This has the advantage of not having to worry too much about the machine configuration, since the robot is already operational and well configured.

```
C5G_IMPORT=%CRC_HOME%\user\include
```

The advantage of off-line translation is that programs can be edited and translated without requiring the robot; however, all the necessary programs are to be available on PC.

- **Grammar File (2 - Fig. 7.39)**

Checkbox Grammar File is only used when the system is CUSTOMIZED NODAL

type (see [GLOSSARY](#) in par. [IDE Page](#)). It contains the indication of the directory, on PC, where the file containing the syntax is to be read. If this checkbox is not set, the WinC5G program will go to search for the definition of the *C5G_GRAMMAR* environment variable.

- **Dimens Font of the Terminal**
It determines the size of the character Font in the Terminal windows. The changes take place when the Apply button is pressed. By setting the Automatic value, the application sets the largest font as to the size of the screen. The font cannot be less than 8pt.
- **C5G library version**
It indicates the version of the C5G translator library that will be used to translate the binary files (.COD, .VAR, .C5G). If the specified version is not among those provided, the WinC5G program will search for a previous version library. The selected version is possibly to be the same or a higher version than the software on the control where it is to be connected. Otherwise, the translation operation may not function correctly since the data in the binary files may not be acknowledged.
- **Save and Load**
The user can save and load the properties of the WinC5G program. In this way the properties linked to a specific Robot Control Unit connection can be set just once and subsequently referenced through Load.
- **FTP server**
To enable the FTP server for use (see [FTP Server mode connection](#)). This is useful in certain software installation situations
After the information regarding the selected library has been changed, the WinC5G program has to be started again so that the change becomes effective.

7.7 Commands Menu

This section describes the controls and the effects they have. Information is given regarding these Menus:

- [Files Menu](#)
- [Edit Menu](#)
- [View Menu](#)
- [Manipulation Menu](#)
- [Help Menu](#)
- [Tools Menu](#)

7.7.1 Files Menu

- **New**
Opens a new page to write a file in ASCII. This file can be saved and if required, translated.
- **Open**
Opens a file that already exists. If the file is binary, it is translated into ASCII for the editing.
- **Close**

Closes the active window in the TOOL Panel. If a .COD or .VAR file is currently open, this command automatically starts the translation process before saving. The Cause/Remedy Window cannot be closed.

- **Close All**
Closes all the open windows in the Tool Panel except the Cause/Remedy page.
- **Save**
Saves the contents of the active window in the Tool Panel in the folder where it was opened. If this window has been opened by New, the Save with Name Window will automatically be opened.
- **Save on PC**
Saves the contents of the active window in the Tool Panel in a folder that can be selected on the PC.
- **Save on Robot**
Saves the contents of the active window in the Tool Panel in a folder that can be selected by the user on the Robot Control Unit.
- **Save All with name**
If the active page is the Cause and Remedy, the list of all the messages and related Cause/Remedy can be saved in a messages.txt file. The description of the messages will be in the language selected relating to the .CRD file (see [.Crd file](#)).
- **Connect**
Allows the user to connect to the desired robot Control, accessing the Directories panel and activating the Terminal Window. Upon connection, Login is requested. The reference database to find the UserName and the Password is that on the Robot Control Unit to which the connection is made.
- **Disconnect**
Closes the connection with the Robot Control Unit. Also the Terminal Window is closed (if open) and the user cannot execute any action that requires connection to the Control Unit.
- **Delete**
To delete the file currently selected in the File Panel.
- **Print**
Prints the contents of the window in the foreground on the Tool Panel.
- **Print all**
Prints all that is displayed in the active window.
- **Page set-up**
To set the printing modalities to be applied for the page to be printed.
- **Properties**
Opens Properties.
- **Quit**
Closes the application.

7.7.2 Edit Menu

- **Undo**
Cancels the last operation executed in the window in the foreground of the Tool Panel.
- **Cut**

Copies the selected text in the clipboard and deletes it from the open window of the Tool Panel.

– **Copy**

Copies the selected text, to the clipboard in the active window of the Tool Panel.

– **Paste**

Copies the clipboard content to the current position of the active window cursor, in the Tool Panel.

– **Select all**

Selects the whole text of the file opened and active in the Tool Panel.

– **Search**

To search for and/or replace an identifier in the text of the open file of the [Tools panel](#). The following search functions are provided:

- Find - find the first occurrence of a certain string.
- Find next - find the next occurrence of the previously inserted string .
- Replace - replaces the string that was previously inserted with another specified string.
- Go to line - to position on a certain line in the text.

– **Translate**

Translates the file that is in the active window in the [Tools panel](#), into the corresponding binary format. If there are errors, these are displayed in the Output Panel in the errors window.

– **Property Notes**

Allows to edit the selected file Property Notes.

– **Filter**

Activates a filter in the Cause/Remedy Window, to limit the number of errors shown. The filter is linked to a timeout, the severity of the error, the code, the counter or the text.

– **Hide selection**

To temporarily hide the display of information previously selected in the errors and actions window.

– **Show All**

To resume the complete display of the information hidden in the errors and actions window, after the Hide command was sent.

– **Modify Log File heading**

To modify the heading of an actions or errors records file.

– **Add/Remove Log File**

To add or remove a file in the errors and actions display ([View Menu](#))

– **Add Comment**

To add a comment to an error displayed in the cause and remedy window.

7.7.3 View Menu

– **Tool Bar**

Activates and deactivates the Tool Bar.

– **Status Bar**

Activates and deactivates the Status Bar

– **Open Terminal**

Opens the Terminal Window if the user is connected to the Control Unit.

- **View robot errors**
The files are opened that contain the error records (.LBE).
- **View robot actions**
The files are opened that contain the action records (.LBA).
- **Next Error**
Displays the next translation error.
- **Update files list**
Updates (refresh) the list of files in the File Panel.
- **Clear Output Window**
Removes the current information from the Output Window in the Output Panel.
- **Show Output Window**
Displays the Output Window.
- **Close Output Window**
Hides the Output Window.

7.7.4 Manipulation Menu



To have access to these functions it is necessary to own the [full licence](#).

The options of this menu are described in detail in par. 7.5.1.5 Files Manipulation on [page 389](#).

- **Axes verification**
- **Tool / Frame Updating**
- **Linear shift**
- **Linear rotation**
- **Mirror**
- **Calibration**
- **Adjustment to system data**
- **Difference in turns**
- **Settings**

7.7.5 Help Menu

- **Meaning of terminal keys**
Opens the dialogue box that describes the keys correspondence between the PC keyboard and some editing functions of the Control Unit, in Program Edit and Memory Debug environments and in the Terminal page.
- **Registration**
To be used to obtain a new WinC5G licence.
- **Information about WinC5G**
Opens the window that gives a general overview of the application.

7.7.6 Tools Menu

- **Protect:** to protect the file selected in the Files panel. A Password has to be specified.
- **Unprotect:** to unprotect (remove the protection) of the file selected in the Files Panel. The user has to enter the Password that was specified when the request was made to protect the file
- **Accept remote:** allows the [WinC5G configuration in Proxy mode](#)
- **Disable Remote:** disables the [WinC5G configuration in Proxy mode](#).

7.8 How to obtain a new licence for WinC5G

To obtain a new licence for WinC5G contact COMAU technical service. Before doing so, make a note of the MAC address of the PC from which the WinC5G is to be run. Proceed as follows:

- a. from WinC5G, select the [Help Menu](#) and 'Insert the log-in code'. Read the number beside the 'Identifier' field.
- b. contact COMAU Technical Service supplying this address.
- c. COMAU will send an ASCII file containing the access word to the function. Open the file with any Editor,
- d. select the whole file content,
- e. copy it,
- f. insert it in the window where it is indicated to insert the licence password;
- g. confirm by pressing Record and check that the files Handling menu is enabled.



After recording, it is important that the PC hardware configuration (network boards, Ethernet interface) does not undergo variations.

7.9 Most common problems

This section describes the most common problems that the user may come up against when operating with WinC5G and what to do to solve the problems.

The time displayed in the error list or in the files list is different to that read on the C5G Control.

WinC5G manages the 'time in WinC5G' information coming from the connected C5G as Greenwich Mean Time. Therefore, asking for the errors list or reading the files list from the Files Panel, the information regarding the time may be different to that read directly on the control through the Teach Pendant.

If, for example, a PC with an Italian Windows version connects, through WinC5G, to a C5G control with Italian time, the time read is displayed with the difference of

an hour; an error that takes place on the control at 20.05 will be shown in the error list with the time of 21.05.

It is therefore necessary to inform your PC that the time read has a certain offset as to Greenwich. For an Italian PC an hour must be added.

In the case of an Italian PC connected to a control in Detroit, it is necessary to subtract 6 hours.

The environment variable to be set on your PC (Start / Settings / system / advanced / environment variables) is TZ.

For an Italian PC connected to an Italian control, TZ must be set to GMT+1:00 (GMT is Greenwich Mean Time).

The application is closed immediately showing a window that states that the application has not been able to load a specific DLL.

This happens when the DLLs (wcres<language>.dll) containing information on the menus have not been found by the application. The DLLs must be in the same directory as the .exe file. Check that the DLL is in the correct position.

The application starts but a window informs the user that a C5G<version>.DLL cannot be loaded. This message is repeated until there is a generic message that informs that the translator DLL cannot be loaded. The application starts but most of the operations are disabled.

The problem arises because the DLL specified in the [Fig. 7.39 - Properties Window on page 402](#) is not in the same directory as the WinC5G.exe file. To solve the problem, open the Properties Window, in the C5G box, library version, and specify a translator library version that is in the same directory as WinC5G.EXE or add the required DLL to the directory containing WinC5G.EXE.

The application replies stating that the selected message cannot be found or that some messages have not been found.

This may happen when the user enters a wrong number on the Troubleshooting page or when opening log files. Both these actions search for the error texts in the .CRD files indicated by the user in the Properties Window with the Select .CRD file button. Several files with the extension .CRD can be added/removed to/from the list used for the search.

The Terminal window remains blank (black)

The application is unable to open the Robot Control Unit window. This happens when someone else is already connected to the Robot Control Unit with a Terminal window opened by another WinC5G. In fact, several connections are allowed to the robot but only one Terminal can be active. To open the Terminal it is therefore necessary that the user who is already connected closes that Terminal session.

When attempting to connect to the Control Unit the application replies with the message "Impossible to connect with the server"

Should it be impossible to connect to the server, there could be a conflict problem between DHCP and proxy Server. To check this, try connecting to the Control Unit with the ftp command from the Controls Prompt and from Internet Explorer. If the first command functions and the second one fails, it is necessary to disable the use of a server proxy in the settings of the local network (LAN).

Furthermore before starting WinC5G, check that the settings for the Server Proxy are correct for the network it is connected to.

Upon connection the error "Authorisations not available for the password"

This happens when the user has made a Login with a Password that is not compatible with the users database defined in the system. That is to say, the combination of Login and Password is not foreseen by the Controller that the user is connecting to.

Upon connection or the Set Login command sent from the terminal, the error "Exclusive Login" is returned

This error occurs when a user is working on the Controller with a right to exclusive access, i.e. that does not permit the use of the system by other devices. It is necessary to Logout from other devices that are not the one required for connection and, if necessary, to also delete the Startup Login.

Upon connection, or when a file is selected from the [Files Panel](#), a message is displayed that indicates "Conflict between Controller and Translator versions"

To solve this problem, it is necessary that the translator DLL, indicated in the C5G box Library version in the [Properties Window](#) is greater than or equal to the Controller version.

Attempting to connect to the Control Unit, the application replies with the message "The computer is not connected to the network"

To solve this problem, open the Internet Explorer, go to the Files menu and deselect "Not in line".

8. FILE MANAGEMENT BETWEEN PC AND CONTROL UNIT

This chapter describes the procedures to delete, view and transfer files through the PC. To operate it is necessary to use the **WinC5G** program, that has to be installed on the PC.

After installing the program, activate the terminal connection to **WinC5G** and from the system/keyboard menu type the commands indicated in the procedures.

The following procedures are described here:

- [View files on PC](#)
- [Delete files on PC](#)
- [File transfer from PC to Controller](#)
- [File transfer from Controller to PC](#)
- [Automatic file transfer from Controller to PC](#)
- [Automatic file transfer from PC to Controller](#)



Any file transfers are logged in the ACTION.LOG file.

8.1 View files on PC

- a. By opening the Properties window from **WinC5G**, the selected directory for files transfer can be viewed/changed.
- b. Type **FV** (Filer, View): <**COMP:(FILE_NAME)**> and press the **ENTER** key on the PC. The selected file will be displayed.

8.2 Delete files on PC

- a. By opening the Properties window from **WinC5G**, the selected directory for files transfer can be viewed/changed.
- b. Type **FD** (Filer, Delete): <**COMP:(FILE_NAME)**> and press the **ENTER** key on the PC.

8.3 File transfer from PC to Controller

- a. By opening the Properties window from **WinC5G**, the selected directory for files transfer can be viewed/changed.

- b. Type **FC** (Filer, Copy): <**COMP:(FILE_NAME)**> and press the **ENTER** key on the PC; this message will be displayed:

Name of destination file

- c. Enter the name of the destination file, <**UD:(FILE_NAME)**> and press the **ENTER** key on the PC; after the operation this text will be displayed:

Done

The contents between round brackets can be omitted.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

8.4 File transfer from Controller to PC

- a. By opening the Properties window from **WinC5G**, the selected directory for files transfer can be viewed/changed.

- b. Type **FC** (Filer, Copy): <**UD:(FILE_NAME)**> and press the **ENTER** key on the PC; this message will be displayed:

Name of destination file

- c. Enter the name of the destination file, <**COMP:(FILE_NAME)**> and press the **ENTER** key on the PC; after the operation this text will be displayed:

Done

The contents between round brackets can be omitted.

8.5 Automatic file transfer from Controller to PC

From within [Files Page - Disk](#), select either [Backup](#) command or [Backup of Saveset](#) command. Such commands copy a list of files (specified either by a name or by a Saveset), from the controller towards the default backup device (XD: or COMP:).

To copy from different directories it is necessary to use [Backup of Saveset](#) command.



For further details, see [par. 5.18.1.2 Backup on page 166](#), [par. 5.18.1.3 Backup of Saveset on page 168](#) and [par. 6.9.4.8.1 FILER UTILITY BACKUP \(FUB\) on page 331](#).

8.6 Automatic file transfer from PC to Controller

From within [Files Page - Disk](#), select either [Restore](#) command or [Restore of Saveset](#) command. Such commands recover the specified files (indicated either by a name or by a Saveset), from the default restore device to the one from which they had been previously saved by means of a Backup operation.



See [NOTE](#) regarding the total amount of entries in the UD: root directory.

The directories and their contents are not copied.



For further details, see [NOTE](#), [par. 5.18.1.4 Restore on page 169](#) and [par. 6.9.4.8.2 FILER UTILITY RESTORE \(FUR\) on page 333](#).

Issue **FUR** (FilerUtilityRestore) command: it copies to the directories, if already existing.

9. SYSTEM OPERATING MODES AND STATES

9.1 Foreword

This chapter describes the following:

- System operating modes
- System states
- Stand-by function

9.2 System operating modes

The C5G Robot Control Unit can operate in three different modes that can be selected through the modal selector switch on the Teach Pendant:

- programming (T1),



- local automatic (AUTO) and



- remote automatic (REMOTE).



Local automatic mode (AUTO) is used to execute production programs; as they contain instructions for the robot movement, to be able to start it is necessary to press the **START** key on the Teach Pendant. The status selector switch must be set on AUTO. Active TOOL, BASE and FRAME cannot be changed when working in AUTO.

The **Automatic remote mode (REMOTE)** is the same as **Automatic local mode (AUTO)**, but the commands (for example the start) are sent from a remote device (for example a PLC).

The state selector switch must be set to the REMOTE position. Active TOOL, BASE and FRAME cannot be changed when working in REMOTE.

The **Programming mode (T1)** is used to create and verify programs. The robot moves, for safety reasons, are run at a lower speed than in automatic mode (maximum robot speed allowed in programming is 250 mm/s on the flange centre).

When the status selector switch is set on position T1, the programs can be developed using editor environment and the spots can be taken from the Teach Pendant moving the robot manually with the motion keys; the programs can be set up using the debug tools of the system. In programming mode, the execution of a move instruction requires that the operator presses the **START** key and the enable device on the Teach Pendant.

When the status selector switch has been set on T1, the system is under the control of the operator. When the selector is set on REMOTE, the system is under remote control (for example from PLC).

Active TOOL, BASE and FRAME cannot be changed when working in REMOTE.

Before any operation can be executed that requires movement, the drives must be powered:

- if the state selector switch is in T1 position, press in the intermediate position the Teach Pendant Enabling Device, to power ON the drives; to switch them OFF and activate brakes on all axes controlled by the Control Unit, just release the Teach Pendant Enabling Device,
- if the state selector switch is in AUTO position, press the R5 softkey (Teach Pendant right menu - it means DRIVE ON when in AUTO state), to power ON the drives; to switch them OFF and activate brakes on all axes controlled by the Control Unit, press the R5 softkey again (Teach Pendant right menu - now it means DRIVE OFF).
Active TOOL, BASE and FRAME cannot be changed when working in AUTO.
- if the state selector switch is in REMOTE position, DRIVEs ON and OFF are remote controlled.

A detailed description follows of all the possible system states.

9.3 System states

Mainly, the system status depends on:

- the status selector switch
- the **DRIVE ON**, **DRIVE OFF** and **HOLD** keys on the Teach Pendant
- system alarm

Transition from one state of the system to another is also influenced by the enable device on the Teach Pendant.

The Control Unit may be in one of these conditions:

- **HOLD status**: the robot is gradually decelerated until the stopping point is reached; movement is suspended and also the execution of the movement program (holdable). When there are all the necessary conditions to exit from the **HOLD** status, the system returns to the previous state (programming or automatic), but to continue to execute the movement program it is necessary to press **START**.
- **AUTO status**: this is usually used to execute production programs that control the robot movements (status selector switch positioned on AUTO or REMOTE or T2). Active TOOL, BASE and FRAME cannot be changed when working in AUTO or REMOTE.
- **PROGR status**: the robot can be moved manually using the jog keys or executing program instructions (from editor environment or by **EXECUTE**). In the latter case,

in order that the movement be executed, the **START** key and the enabling button have to be kept pressed.

If the controlled stop function class 1 (EN 60204-1) is active, the power cut-out (opening of the power contactor) may take place with a delay that ranges from a minimum of 1 second to a maximum of 2 seconds.

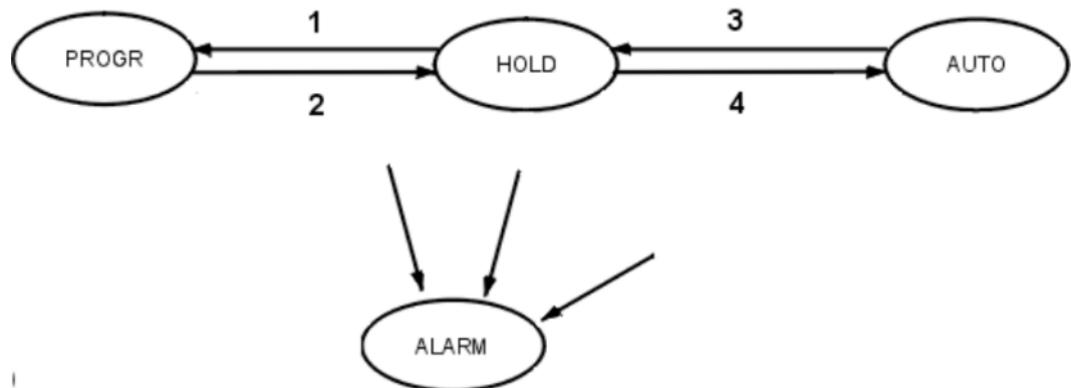
With the status selector switch positioned on T1, the power cut-out is immediate (EN 60204-1, class 0 stop).

- **ALARM status:** this status is entered when there is a system alarm. According to how serious the error is, the system takes different actions, such as suspending the program execution, deactivation of the drives, etc. A situation may occur where the alarm cannot be reset, therefore the drives cannot be switched on.

The current system status is displayed on the first status line of the Teach Pendant (or in the Terminal window of tool **WinC5G** on PC).

The figure shows a simplified diagram of the actions that determine the system change-over from one state to another.

Fig. 9.1 - Simplified diagram of the system states



1. Status selector switch on T1 + HOLD released
2. HOLD or DRIVES OFF or selector switch change
3. HOLD or DRIVES OFF or selector switch change
4. Status selector switch on AUTO or REMOTE + HOLD released

Note: To perform transient 4 also the enabling device key has to be pressed

9.3.1 HOLD status

The safety rules to be complied to when operating with the Control Unit have been studied so that the system enters the HOLD status every time a change is made in the operating mode, passing for instance from LOCAL to PROGR mode.

To exit from the HOLD status to enable a certain operating mode, there must be all the required safety conditions. A typical example is when the operator brings the status selector switch to PROGR to work near the robot, holding the Teach Pendant to carry out learning operations for the points.

In PROGR, exiting from HOLD can be obtained by pressing **START**, this is controlled by the system and therefore is active when an instruction or a movement program is executed. When the **START** key is released again the system returns to HOLD status.

When entering the HOLD status, the corresponding **HOLD key** on the Teach Pendant is considered as pressed. Further pressure on the key causes the system to exit from HOLD status.

If the HOLD status has been caused by pressing the **DRIVE OFF** key on the Teach Pendant (either Enabling Device released or R5 softkey pressed meaning **DRIVE OFF**), the **DRIVE OFF** and **HOLD** keys must be pressed again to exit from HOLD status, and then re-power the drives (either intermediate pressure of the Enabling Device or press R5 softkey meaning **DRIVE ON**).

9.3.2 AUTO status

To have the system in AUTO status, the status selector switch on the Robot Control Cabinet must be set on AUTO or REMOTE. Active TOOL, BASE and FRAME cannot be changed when working in AUTO or REMOTE.

In AUTO status, to start programs ready for execution, press the **START** key on the Teach Pendant or activate the START input from remote device.

Conditions that change the system status from AUTO to HOLD are:

- status selector switch changed to another position;
- **DRIVE OFF or HOLD pressed**;
- system alarm.

To return to AUTO, bring the selector switch back to the required position, and press again the previous buttons (**DRIVE OFF** and/or **HOLD**). To continue the movement program execution, press **START** after making sure that the drives are powered (**DRIVE ON**).

9.3.3 PROGR status

PROGR status is active when:

- the status selector switch is set to T1.

In this state the robot can be moved manually, using the jog keys on the Teach Pendant. It is also possible to run programs from IDE environment (see **IDE Page** in **C5G Control Unit Use** manual) to check that they are correct and if necessary make changes. Movements are at slow speed.

9.3.4 ALARM status

The system enters **ALARM** status when an alarm is generated. An error message is displayed on the second status line of the system screen and the associated LED, next to the **ALARM** key on the Teach Pendant, lights up.

There are different conditions that can generate an alarm and the action to be taken to exit from **ALARM** status and bring the system back to the previous state vary according to how serious the error is.

9.4 Stand-by function

The purpose of the Stand-by function is to cut down the current consumption when the robot is stationary.

The function is automatically activated when the Control Unit is in local automatic or remote automatic mode and after the robot has remained stationary for a time defined by variable \$TUNE [27]; this function activates the motor brakes to keep the static position of the robot. The value of variable \$TUNE [27], set by COMAU, is 120 seconds; if this variable is set to 0 the function is deactivated.

The Stand-by function is automatically deactivated at the first request to start movement again (START, RESUME) from the system.

The system Stand-by status is displayed in the status bar of the Teach Pendant. To display the state of a single arm, read this status on the **Status** sub-page, the **Motion** page on the Teach Pendant.



The safety precautions are to be scrupulously observed regarding this operating condition of the Controller.

10. START AND STOP CYCLE

Automatic execution of the production cycle means executing the production program which handles the robot(s) motion..

To **start the working cycle** in automatic state, the following conditions must be satisfied at the same time:

- the **Modal selector switch** must be put to either the AUTO (LOCAL) or REMOTE position; it is placed onto the Teach Pendant for the **iTP** (wired) model and onto the docking station for the **WiTP** (wireless) model. For further details, please refer to [par. 5.2 Modal selector switch on page 51](#);
- **no active alarms** with severity greater or equal to 4. For further details about handling alarms, please refer to [par. 5.13 Alarm Page on page 113](#);
- the motion **program** must have been **loaded and activated** (which means to be in READY state).

To do that, using the Teach Pendant ([Prog Page](#)), use ‘Load...’ and ‘Activate’, or ‘Activate’ commands;

from within **WinC5G**, use commands either **MEMORY LOAD (ML)** and **PROGRAM ACTIVATE (PA)** or **PROGRAM GO (PG)** ;

- the **Stop pushbutton** must be released. For further details about it, please refer to [par. 5.3.2 Pushbuttons and LEDs on page 62](#);
- the **HOLD** button must be released. See in detail the **HOLD (yellow)** button description;
- drives must be on (**DRIVE ON**). Further information about the corresponding buttons are provided in [par. 5.5.1.2 Right Menu on page 69](#);
- the **START** button must be pressed (see [START \(green\)](#)).

In AUTO (LOCAL) mode, the **Teach Pendant** is used to handle Cycle Start/Stop; in REMOTE mode, **I/O segnali** are used.

The **cycle stop** occurs whenever one of the above listed conditions is not verified.



Please refer to [C5G Control Unit - Technical Specification](#) manual, chap. [Start and Stop Controls](#), and [PDL2 Programming Language](#) manual chap.[Ports](#) for any information about, respectively, hardware and programming topics.

11. I/O CONFIGURATION PROGRAMS

The following programs are available, written in VP2 language, to handle respectively:

- configuring I/O modules - [IO_CNFG Program - I/O modules configuration](#)
- configuring Fieldbus networks devices - [FB_util Program - Fieldbus modules utility](#)



To configure I/O modules on the Fieldbus Master network, it is needed to use both SYCON.net and **WinC5G** programs, on the PC, to create the file which is then processed by the [IO_CNFG Program - I/O modules configuration](#) and [FB_util Program - Fieldbus modules utility](#).

A detailed description is provided in [par. 11.3 Project of a Master Fieldbus Network on page 449](#).

- mapping I/O Points - [IO_MAP Program - I/O ports mapping](#)
- copying the I/O configuration from a Controller to another one - [IO_CLONE Program - I/O Configuration Export/Import](#).

11.1 IO_CNFG Program - I/O modules configuration

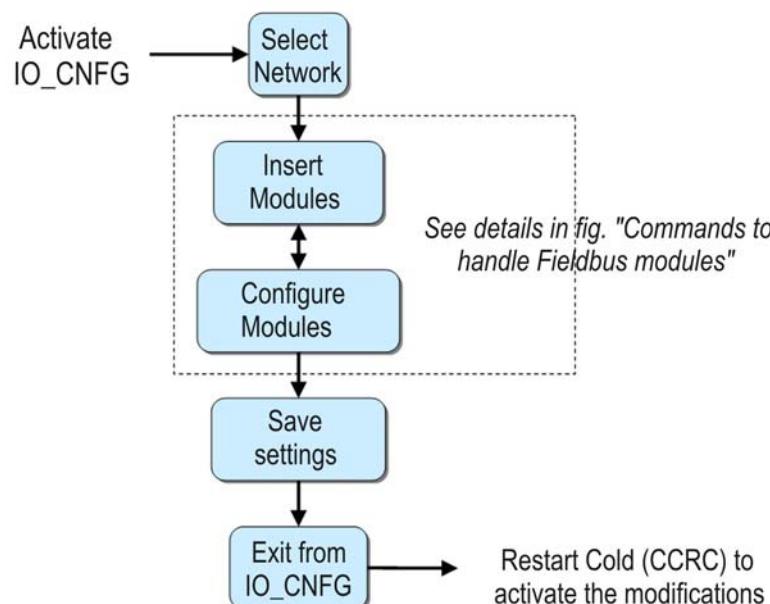
11.1.1 Activation of IO_CNFG program

This program, **implemented in Visual PDL2 language**, is used to declare the presence and to configure the I/O modules on the C5G Controller Unit.

- Slave Fieldbus Module
- Master Fieldbus Module
- IEAK (Integrated Electronics Arm Kit)
- IESK (Integrated Electronics Safety Kit)
- Conveyor Tracking Kit
- Seam Tracking Kit
- COM0: serial Port.

Each module is given a **NAME** which can be chosen by the User. Such a name will be used to easily recognize the module during some operations (e.g. when viewing Devices in the [Display Page](#), or in the Devices list of [IO_MAP Program - I/O ports mapping](#), etc.).

Fig. 11.1 - Main flow of the configuration operations



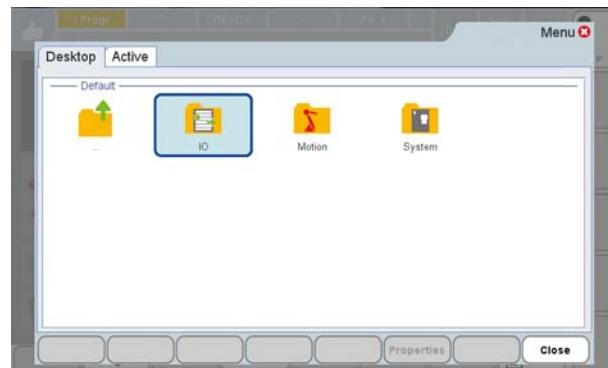
Note that before mapping^(*) I/O points (see [IO_MAP Program - I/O ports mapping](#)), it is needed to have already configured their corresponding modules, by means of this program.

(*) 'Mapping' means to put in relation the Devices I/O Points with the PDL2 logical Ports (es. \$DIN, \$DOUT, etc.).

The **IO_CNFG** program is always present and available on the Teach Pendant.

To use it, act as shown in [Fig. 11.1](#). Such a procedure steps, fully described, are as follows:

- a. access the [Setup page](#), on the Teach Pendant
- b. touch **IO** icon



- c. touch **IO_CNFG** icon



- d. The system runs **IO_CNFG** program



In the main page of **IO_CNFG** program, the following functionalities are available:

- [Network](#)
- [Serial](#)
- [Acopos](#)
- [Annex](#)

- [Save](#)
 - [Exit.](#)
-

11.1.2 Network

Allows selecting the Network on which to insert and configure a new I/O module.

Fig. 11.2 - Scelta della rete



By touching **Network** key, the program opens a menu for the User to choose the wished network. The following choices are available:

- [Powerlink](#) - for Fieldbus and X20 I/O modules
- [Virtual](#) - for creating not physically existing devices.

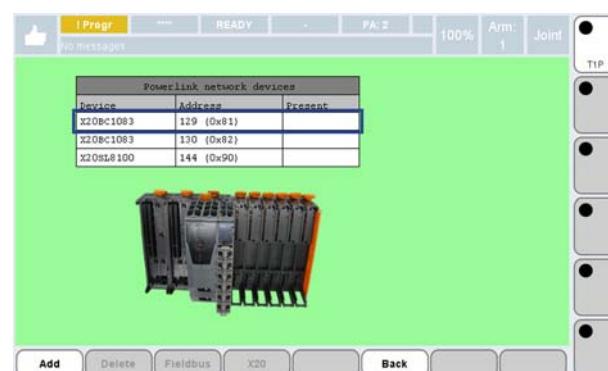


Back

Note that in all screen pages of all the existing levels, **Back** key is always present to allow quitting such a step of the procedure, and going back to the previous one.

11.1.2.1 Powerlink

Touching **Powerlink** item, the following table is displayed.



The screenshot shows the 'Powerlink' screen. At the top, there's a toolbar with icons for 'Program', 'READY', 'PA: 2', '100%', 'Arm: 1', and 'Joint'. On the right side, there's a vertical stack of circular status indicators labeled 'TIP'. The central area displays a table titled 'Powerlink network devices' with three rows of data. The columns are 'Device', 'Address', and 'Present'. The data is as follows:

Device	Address	Present
X20BC1083	129 (0x81)	
X20BC1083	130 (0x82)	
X20SL8100	144 (0x90)	

Below the table, there's a 3D model of an industrial I/O module. At the bottom, there's a navigation bar with buttons for 'Add', 'Delete', 'Fieldbus', 'X20', and 'Back'.

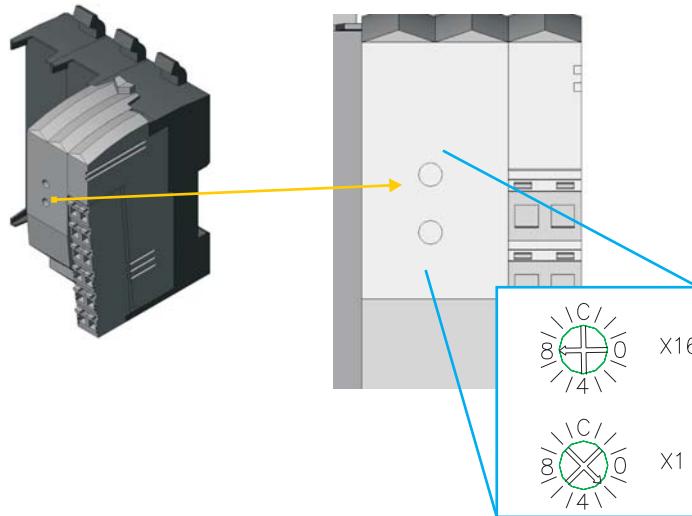
In such a table, there are some rows, for the two allowed Bus Coupler devices to be inserted into the Powerlink Network.

The table columns are:

- **Device** - is the device code

- **Address** - is the fixed address, set by means of the rotary switches, placed onto the module.
For the Bus Couplers addresses, they are fixed to 129 (0x81) and 130 (0x82).

Fig. 11.3 - Rotary switches for address setting



Tab. 11.1 - Powerlink Network address settings for the Bus Coupler module

Bus Coupler module function	Rotary switches settings (Address)		Notes about the addresses configuration
	X16	X1	
First installed module	8	1	Address of the first installed module
Second installed module	8	2	Address of the second installed module.



For further information, please refer to par. [Ethernet POWERLINK address](#), chap. [Details about C5G Control Unit in C5G Controller Unit - Technical Specifications](#) manual.

- **Present** - indicates whether the device is currently present (**YES**) or not (**NO**) onto the Powerlink net.

This subpage [Functional keys menu](#) provides the following commands:

- [Add](#)
- [Remove](#)
- [Fieldbus](#)
- [X20](#)
- [Back](#).

11.1.2.1.1 Add

Permette di aggiungere il dispositivo alla rete Powerlink, per poter procedere alla configurazione dei moduli ad esso relativi.

Toccando questo tasto, il programma inserisce il dispositivo e la dicitura **SI** nella corrispondente colonna **Presente**.



It allows adding the Device to the Powerlink network, to be able to configure its corresponding modules.

As soon as this key is touched, the program inserts the word **YES** in **Present** column.

11.1.2.1.2 Remove

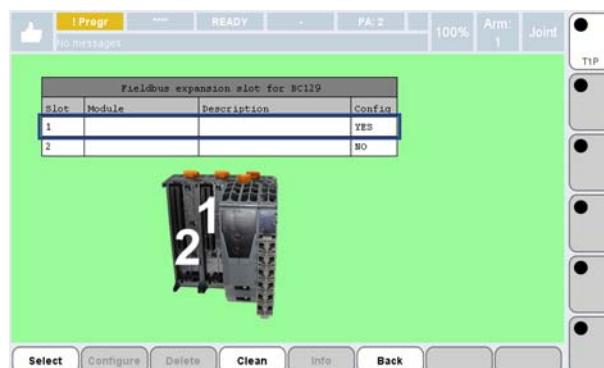
It allows removing the selected device, from current network. This key is available just if the device is present.

As soon as this key is touched, the program removes the whole row content.

11.1.2.1.3 Fieldbus

It allows the configuration of the Fieldbus modules inserted in the Bus Coupler.

By touching this key, the program displays a table corresponding to the two slots of the being configured Fieldbus.



Columns include the following information:

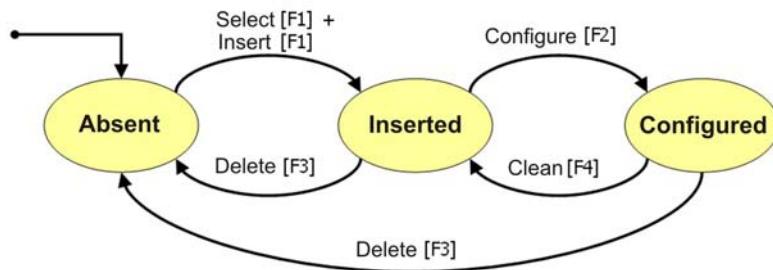
- **Slot** - is the slot number in which the selected module is physically mounted. Note that the closest to the Bus Coupler CPU is number **1**.
- **Module** - is the Builder Code of the module inserted in the slot.
- **Description** - is the description of the module inserted in the slot.

- **Config** - indicates whether the module is configured or not.

The above figure show the situations in which both slot 1 and 2 are empty.

The following diagram indicates the actions corresponding to the available commands in the Bottom Menu.

Depending on the Fieldbus module handling status, some commands are enabled or not. **Back** command is always enabled.



When **Select** key is touched, the wished slot is selected and the User is allowed to insert a new module in such a slot



NOTE - In the C5G Control Unit, 3 Fieldbus modules are allowed to be present, distributed between the 2 Bus Couplers.

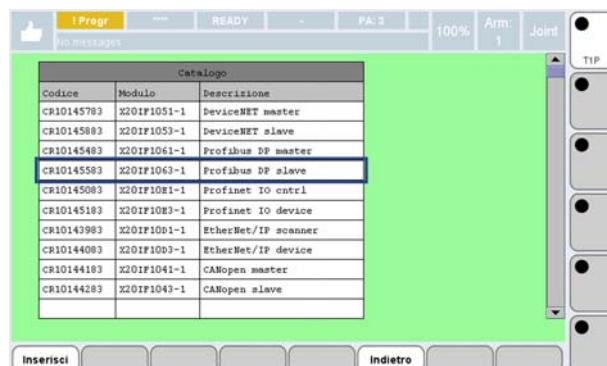
They can be:

- **2 Slaves and 1 Master, or**
- **2 Masters (in 2 different Bus Couplers) and 1 Slave.**

- [Slave Fieldbus Module](#)
- [Master Fieldbus Module.](#)

Slave Fieldbus Module

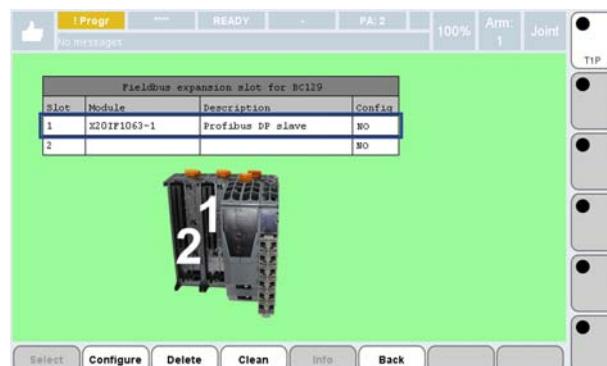
A table is displayed in which the User is allowed to choose the Slave module type to be inserted in the selected slot (note that the list also includes Master modules).



Columns contain the following information:

- **Code** - is the Comau Code
- **Module** - is the Builder Code
- **Description** - indicates the module type.

To continue, choose the wished module and touch **Insert** key to insert it into the slots table.



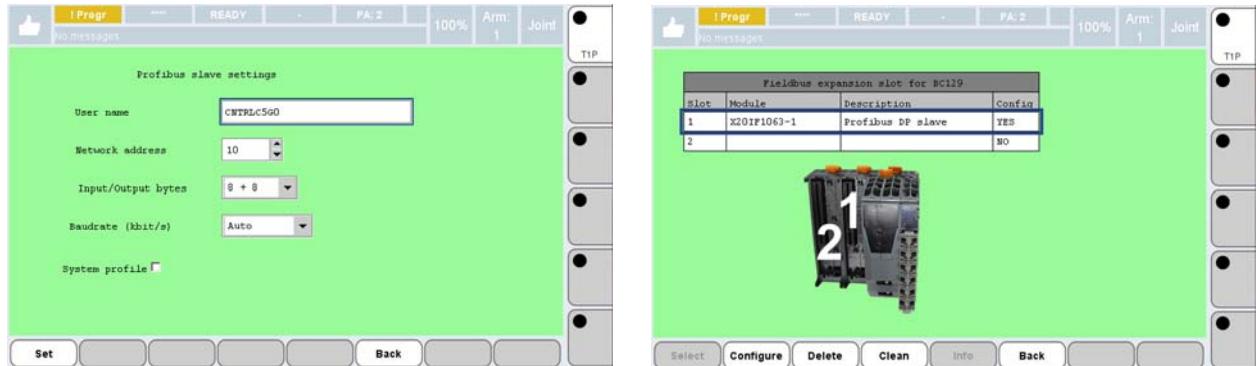
The program displays a screen page in which it is possible either to continue configuring the newly inserted module (**Configure** key), or clearing it (**Delete** key) and selecting a new one.

When touching **Configure** key, the program displays the following screen page (an example for **Profibus** network is shown), in which the User has to fill up the existing fields:

- **User name** - is the name chosen by the User, to identify the being inserted module. This name follows the PDL2 variables standard naming, a length of 8 alphanumeric characters is suggested. Furthermore, in case of Profinet network, DO NOT use the underscore (_) character)
- **Network address** - is the node address on the network which the Slave is connected to
- **Input/output bytes** - is the total number of exchanged input/output bytes
- **Baud rate** - is the operational speed of the module, kbit /s

- **System profile** - the User must select it to be able to associate the System Profile to it, by means of [Profile](#) function of [IO_MAP Program - I/O ports mapping](#).

Fig. 11.4 - Profibus Slave module Configuration



Touch **Set** key to make the inserted data operational.

The word **YES** is displayed in **Config** column at the end of the configuration operation.

Other Fieldbus types

In the previous example, the configuration of a Profibus Slave module has been described.

Depending on the Fieldbus type, the configuration page can be different compared to the one in such an example:

- **Profinet** - **Network address** and **Baudrate** are not specified



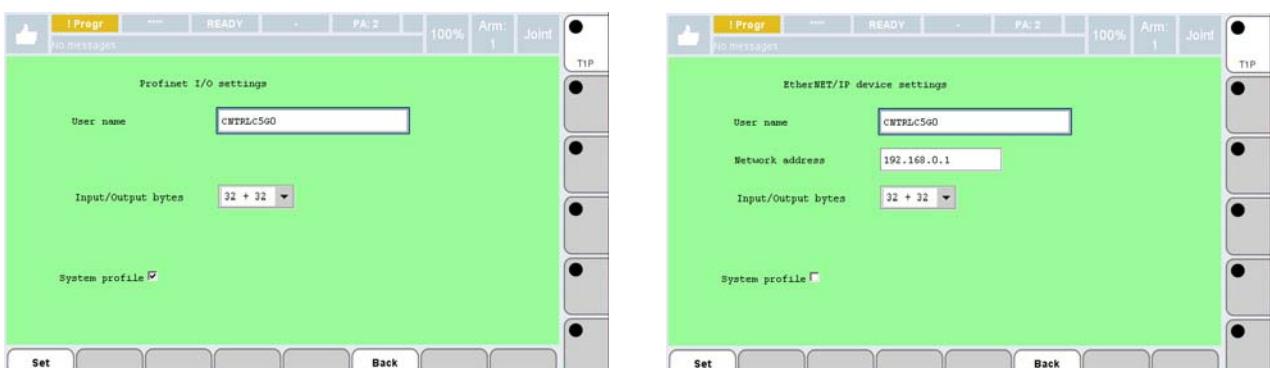
NOTE

Comau Profinet Slave DOES NOT implement LLDP protocol, which is used for Profinet network topological handling.

Thus, mainly in Siemens development environments, the network topology MUST NOT be used.

- **Ethernet/IP** - the **Network address** is the IP address (when the Controller is connected to the same Ethernet network, this address MUST NOT be equal to the Controller's one); the **Baudrate** is not specified.

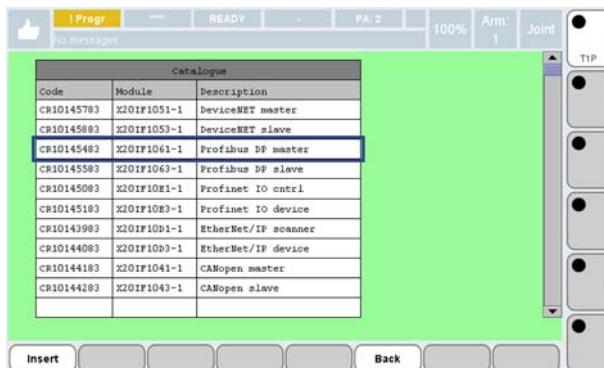
Fig. 11.5 - Other Fieldbus types - configuration page



For any other field, please refer to the **Profibus** Slave configuration description.

Master Fieldbus Module

A table is displayed (see the figure below) to choose the Master module to be inserted in the selected slot (note that the list also includes the Slave modules).



The information contained in the table columns are as follows:

- **Code** - is the module Comau Code
- **Module** - is the Builder Code
- **Description** - indicates the module type and whether Master or Slave

Choose the wished module and touch **Insert** key to insert it in the slots table.



In the shown above figure, the User can either continue configuring the newly inserted module (**Configure** key), or clearing it (**Delete** key) and selecting a new one.

Configuring a module on the Master network means to define all the connected devices. It is performed **offline** (on Personal Computer) by means of **SYCON.net** and **WinC5G** programs (please, refer to [par. 11.3 Project of a Master Fieldbus Network on page 449](#)).



WARNING - while configuring a device again on the Master network (e.g. to add one more I/O module), any device of the new configuration which was already present in the old one, is NOT modified, so it still keeps the previous I/O mapping, if already existing.

As soon as the **.spb** file has properly been transferred to **UD:\SYS\CNFG** directory on the Controller, touching **Configure** key causes the selected module to be configured on the Master network.

In directory

LD:\EPL

the following files are created (the ones corresponding to the being configured network only, obviously):

- 42764b1.fw.gz** - for DeviceNet Master
- 42774b1.fw.gz** - for Profibus Master
- 42781_1.fw.gz** - for PROFINET Master
- 42781a1.fw.gz** - for PROFINET Master.

Furthermore, file **epicfg.xml** which is in the same directory, is modified.

A copy of the listed above files is also stored in

UD:\SYS\CNFG

to restore the Controller original configuration.

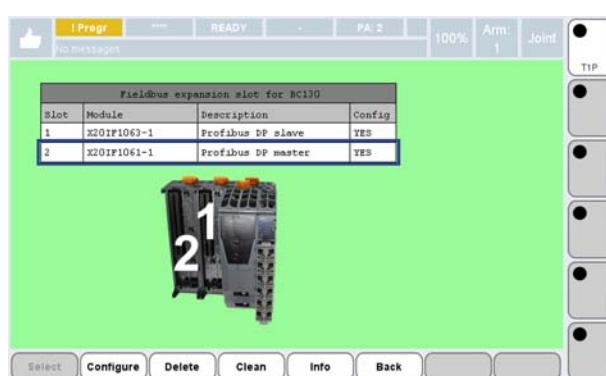
IO_CNFG program displays a table, on the Teach Pendant, including all the devices connected to the configured module.

The table contains the following information:

- **Addr.** - is the address of the being configured device, within the chosen Master network
- **Name** - is the device name
- **Inp byte / Out byte** - is the total amount of exchanged Input and Output BYTES
- **Enabled** - if this field value is **YES**, the corresponding device must always be connected, otherwise an error occurs. If the value is **NO**, no errors occur even if not connected; this functionality is useful for devices which could temporarily be disconnected (e.g. after an automatic tool change).

This field can be modified by **Enable** and **Disable** keys, available in the Functional keys menu.

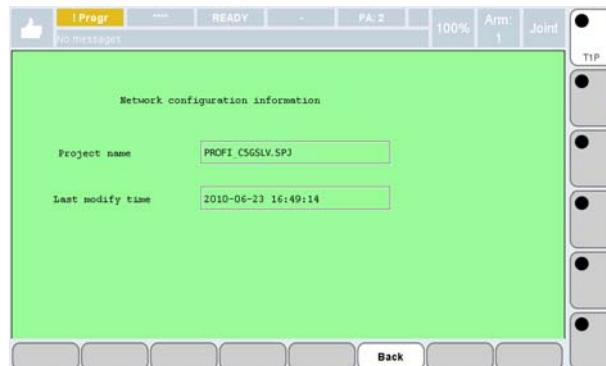
At the end of the configuration operation, a screen page is displayed stating the configuration has been performed.



Command **Info** is now available in the Functional keys Menu, to display a screen page including information about the Master network configuration (as shown in the following figure).

The displayed data are:

- Project file name (**.spj**) corresponding to the selected Master network
- Project file last modification date and time.

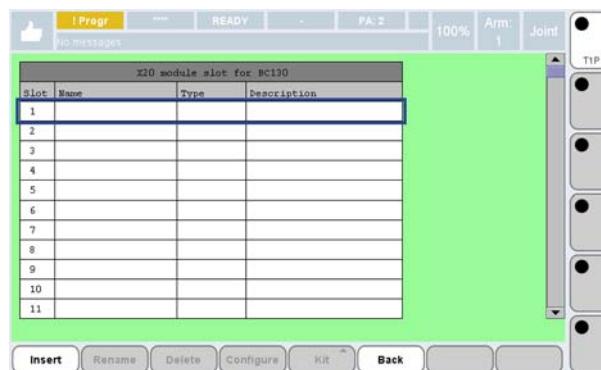


11.1.2.1.4 X20

It allows configuring the X20 modules for digital and analog I/Os currently inserted in the Bus Coupler.

Touching this key, the program displays a table containing all the available slots (maximum 10).

The X20 modules are to be inserted sequentially, starting from slot n.1, exactly like they have been physically mounted into the Bus Coupler.



Touch **Insert** key to insert a new module. The program displays a table containing the list of all the available modules.



The information included in the table columns, are as follows:

- **Code** - is the module Comau Code
- **Module** - is the module Builder Code
- **Description** - indicates the module type

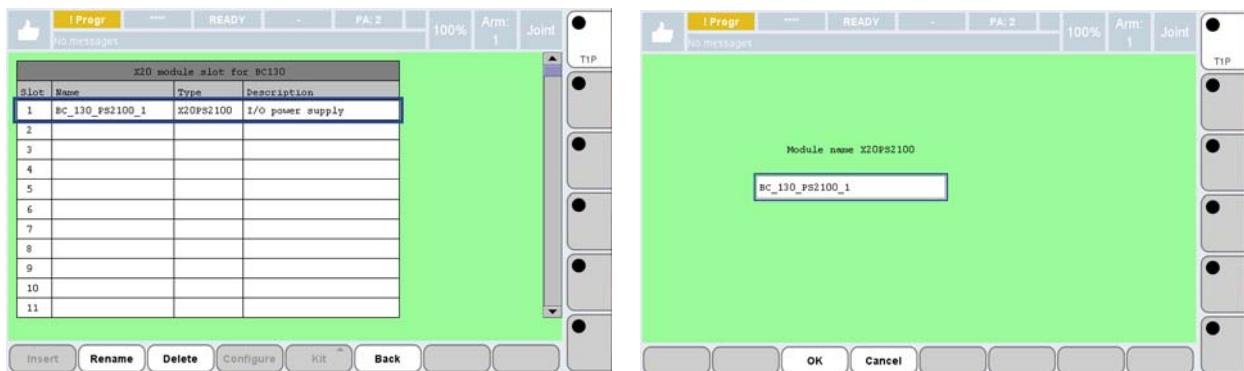
Select the wished module and press **Insert** key to confirm.



Note that there are some X20 modules allowing to configure some parameters. If so, Configure command is available in the Functional keys Menu. Touch such a key and insert the required information.

The program automatically generates a module name. The User is allowed to modify it, if wished, taking care of choosing a unique name in the system.

Fig. 11.6 - Rename



To do so, touch **Rename** key: in the displayed screen page, insert the wished name and touch **OK** to confirm.

This name follows the PDL2 variables naming, a length of 8 alphanumeric characters is suggested.

X20 modules configuration

IO_CNFG program is used to inform the System about the physical presence of objects attending the following functions:

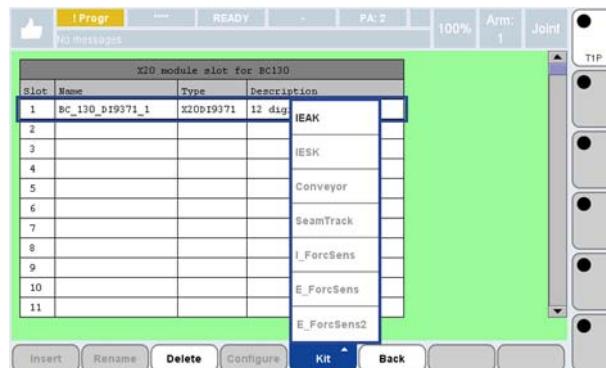
- [IEAK \(Integrated Electronics Arm Kit\)](#)
- [IESK \(Integrated Electronics Safety Kit\)](#)
- [Conveyor Tracking Kit](#)
- [Seam Tracking Kit](#)

and to provide everything needed to handle them from within PDL2 programs.

IEAK (Integrated Electronics Arm Kit)

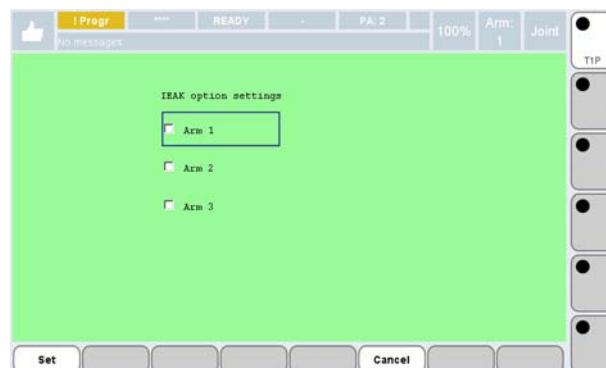
It is a safety device allowing the interaction between the Operator and the robot, even if in Automatic state.

This kit includes **Input** points which have to be physically mounted onto the X20 module; to declare their presence in the configuration, it is needed to insert them as shown in the following figures. Such an operation causes the **Kit** menu to be available in which the User has to select item **IEAK (1)**.



Each IEAK module can handle until 4 Arms: in the displayed page, the User has to select the Arm which the being configured IEAK module is associated to.

As soon as the IEAK module has been installed and configured, MOTOR ON and MOTOR OFF commands are made available for each selected Arm.



Signals made available by the System (MOTOR ON and MOTOR OFF) MUST be mapped by the User to the wished logical ports.

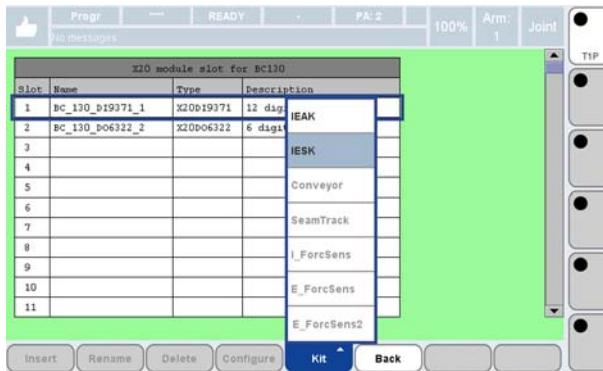
At the end of the required fields completion, touch **Set** key to activate the wished settings.

IESK (Integrated Electronics Safety Kit)

It is a safety device allowing the interaction between the Operator and the robot, even if in Automatic state.

This kit includes **Input** and **Output** points which have to be physically mounted onto the X20 module; to declare their presence in the configuration, it is needed to insert them. Selecting the inserted Input points the **Kit** menu is made available, in which the User has to select item **IESK**.

Fig. 11.7 - IESK - Configuration

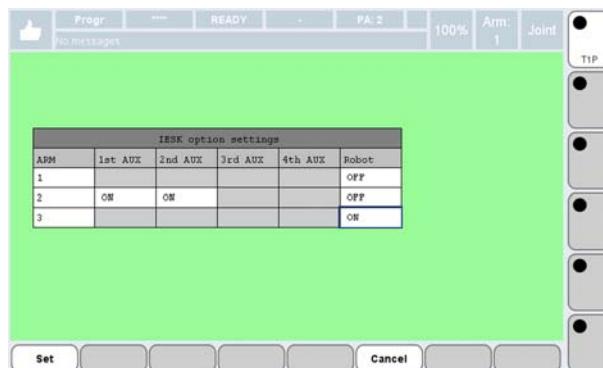


Touch it to confirm. The System displays the page shown in the following figure, including a table which allows to ACTIVATE/DEACTIVATE IESK devices: this means to tell the System the option (i.e. the kit) is PRESENT.

In the table, columns are available for:

- existing **Arms**
- existing **Auxiliary Axes** - note that the index of the real auxiliary axis does not match with the one indicated in the table, which is just a sequential ordinal number
- **Robot** - it is the whole robot seen as it was a unique axis, thus being able to have got its own IESK.

If a IESK is **ACTIVE (ON)**, when connecting the associated Axis (or Robot) is wished, the System checks whether or not such a IESK is enabled.



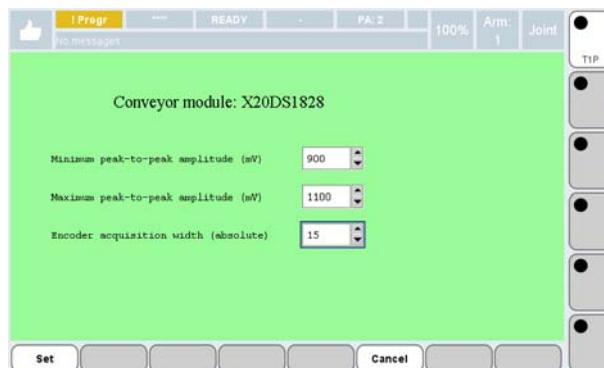
To modify the activation settings, touch the cell of the wished Arm and Axis, or Robot. Choose the new setting and touch **OK** to confirm.

At the end of the settings modification, touch **Set** key to confirm the whole table content.

Conveyor Tracking Kit

This option includes an interface module allowing the communication with a device (Encoder, Resolver, etc.) able to read the current position of a moving device (conveyor belt, press, etc.).

The interface must be physically mounted and declared as being present in the configuration, taking care of inserting it.



When selecting the inserted interface module, a list is made available (**Kit** menu), in which **Conveyor** item must be chosen.

Depending on the module type, the displayed page is different, because of the appropriate parameters to be set.

When all the wished values have been inserted, touch **Set** key to confirm.

The above described configuration operation makes some ports available to handle the Conveyor device (reading its position); they are of \$FMI type, with 129, 130, 131 and 132 indexe. They are associated to the 4 allowed Conveyors, starting from 129 which is associated to the first Conveyor.

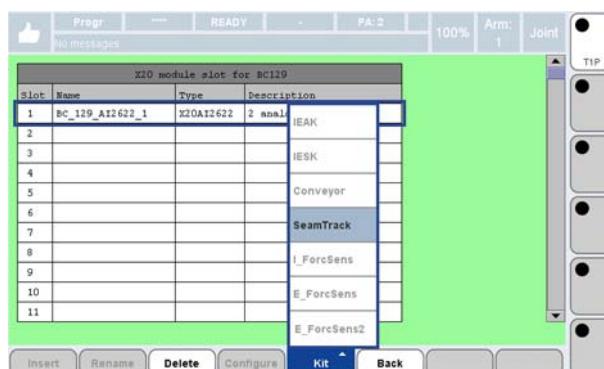


After saving all settings, in order to make the modifications operational, a Controller cold **restart** command must be issued (see [par. 5.7.2.2.1 Cold on page 91](#)).

Seam Tracking Kit

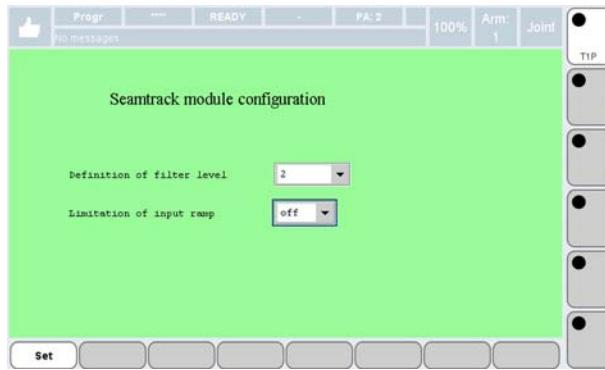
This option includes an interface allowing the communication with a seam tracking device, during the arc welding process.

Such an interface must be physically mounted and declared as present within the configuration, taking care of inserting it.

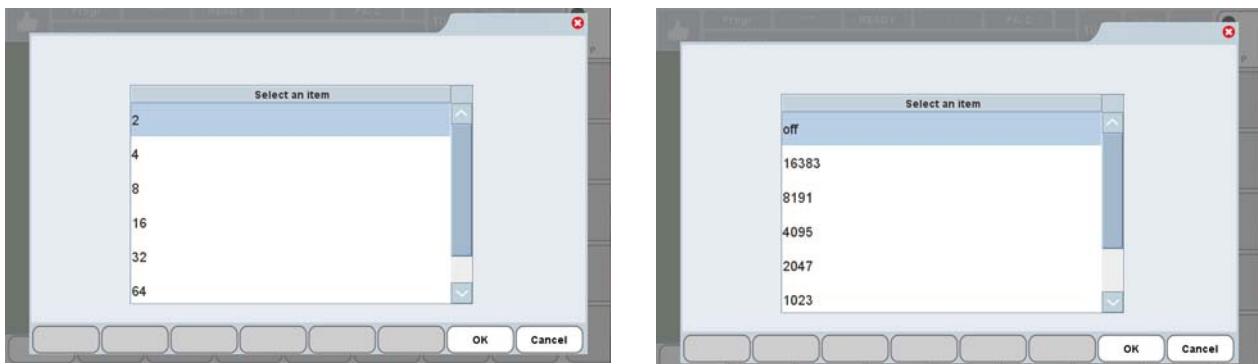


Selecting the newly added interface, the **Kit** menu is made available, in which the user must choose **SeamTrack** item.

A page is displayed for the user to enter the wished values for the selected module.

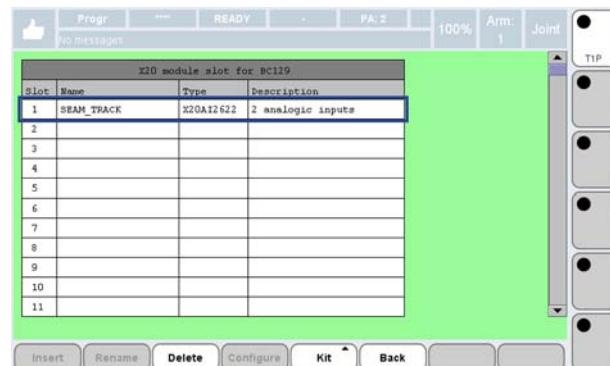


Definition of filter level field indicates the level of the filter which is applied to the input signal. The higher the specified level, the stronger the filter action which must limitate such a signal variations.



Limitation of input ramp field allows to specify how to limitate spikes on the input signal: its value indicates the maximum acceptable variation for such a signal.

After entering the wished data, touch **Set** key to confirm them. For filter level values, see the left figure above; for input ramp limitation values see the right figure above. The X20 module name, so entered and configured, becomes **SEAM_TRACK**.



11.1.2.2 Virtual

When the physical devices are not available yet and it is wished to start developing PDL2 programs which use them, the User is allowed to define some general Virtual Devices, specifying their names and total amount of bytes.

Such points will be later mapped, by means of [IO_MAP Program - I/O ports mapping](#), to the PDL2 logical ports the User wishes to use in the being developed program.

Touch the **Virtual network** choice to confirm it.

The table in the figure below is displayed, including the lists of the currently present Virtual Devices.



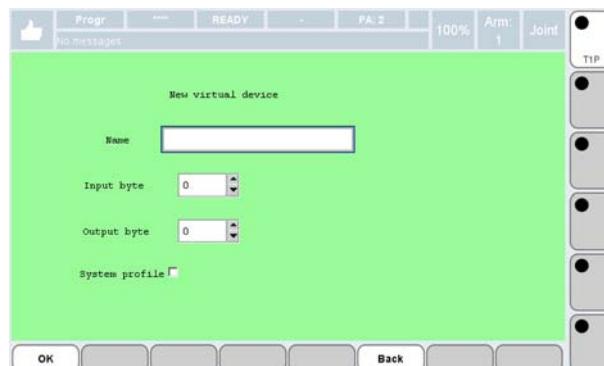
This sub-page Functional keys menu includes the following commands:

- [Add](#)
- [Edit](#)
- [Delete.](#)

11.1.2.2.1 Add

Adds a new Virtual Device.

In the shown above table select an empty line and touch **Add**.



The program displays the screen page above, in which the User has to insert the following information:

- **Name** - is the Virtual Device name (chosen by the User, taking care of assigning a unique name in the system). It must comply PDL2 variables naming rules; a length of 8 alphanumeric characters is suggested. - to edit this field, touch it and type in the wished name by means of the automatically opened [Alphanumeric keyboard](#). Touch **OK** key to confirm.
- **Input byte** and **Output byte** - are, respectively, the total amount of Input and Output bytes associated to the Virtual Device. Touch the being edited field and insert the value by means of the automatically activated [Numeric keyboard](#). Move focus to the field to be edited and press **ENTER**. Either type in or use up/down keys to reach the wished value; then press **ENTER** to confirm.

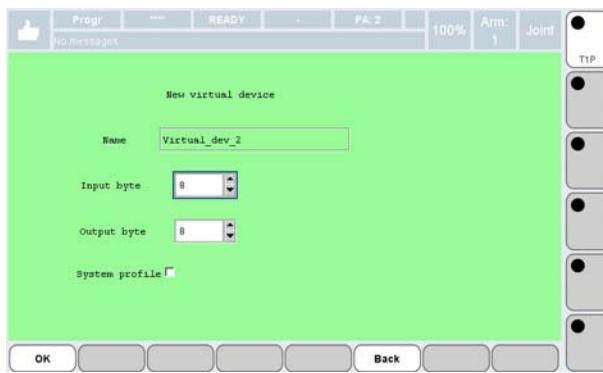
- **System profile** - this checkbox must be selected to associate the Virtual Device to the [Profile](#) functionality of [IO_MAP Program - I/O ports mapping](#). When selected, such a Device becomes part of the System Profile.

When data insertion is completed, touch either **OK** key to confirm or **Back** key to cancel the operation and go back to the previous screen page.

11.1.2.2.2 Edit

It allows modifying the total amount of either Input or Output bytes, for an already existing Virtual Device. The program displays the screen page shown in the figure below.

As shown in the figure below, it IS NOT allowed to modify **Name** field. The use of this screen page is similar to the one associated to [Add](#) command.



11.1.2.2.3 Delete

It deletes the currently selected Virtual Device. In the example of the figure below, touching **Delete**, causes **Virtual_dev_2** Virtual Device to be removed from the table.



11.1.3 COM0: serial Port

11.1.3.1 Serial

This functionality has been designed to handle situations in which an application program must use a second serial port.

Touching **Serial** key, it is possible to enable **COM0:** port.

With reference to the left figure below, touch the displayed field to open a list (right figure below) and choose the wished item. Touch **OK** to confirm.

Then touch **Set** key to make the choice operational.



Since that moment, the port is available to be used in PDL2 programs.

Operate in the same way, to disable it.

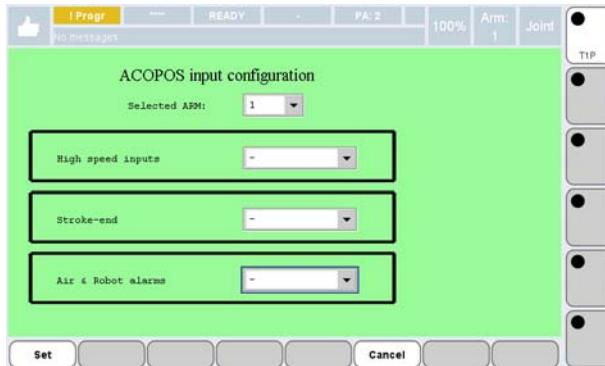
11.1.4 Acopos

Some special purpose Inputs exist, which are dedicated to specific signals, available to handle the robot axes:

- High speed Input
- Electrical stroke ends
- Air and robot alarms.

They can be associated to the **ACOPOS** modules (each one of such modules has got a maximum of 2 available Inputs for as much axes).

The default configuration is NOT ASSIGNED (represented by the hyphen character, as shown in following figure).

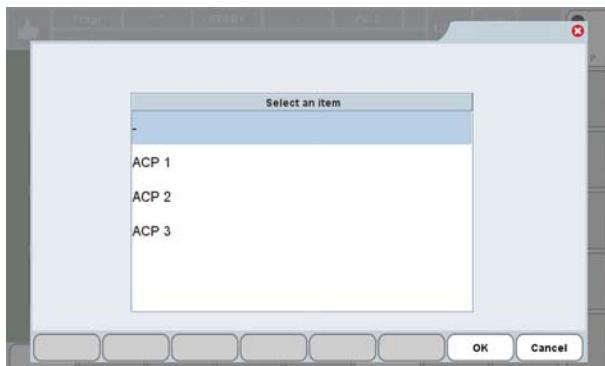


L'utente può variarla nel caso in cui sorgano delle esigenze specifiche. A tale scopo, nella pagina **Configurazione Input ACOPOS** mostrata qui sopra, l'utente può modificare i seguenti dati, semplicemente toccandoli ed aprendo l'elenco delle scelte predefinite. I campi modificabili sono:

- **ARM selezionato** - è l'ARM al quale appartengono gli assi interessati
- **High speed inputs, Stroke-end, Allarmi aria e robot** - sono degli elenchi di possibili moduli ACOPOS, tra i quali scegliere quello da associare all'Input specifico.

The user is allowed to modify it, for specific needs. To do that, in the shown above **ACOPOS input configuration** subpage, the user can edit the the following data by simply touching them to list the available predefined choices. The editable fields are:

- **Selected ARM** - is the ARM which the involved axes belong to
- **High speed inputs, Stroke-end, Air & Robot alarms** - are lists of existing ACOPOS modules, among which selecting the one to be associated to the specific Input.



At the end of the wished operations, touch **Set** key to confirm all modifications.

11.1.5 Annex

This command is provided to configure new additional network modules (Powerlink Master, Powerlink Slave, C5GOpen interface, secondary Ethernet interface).

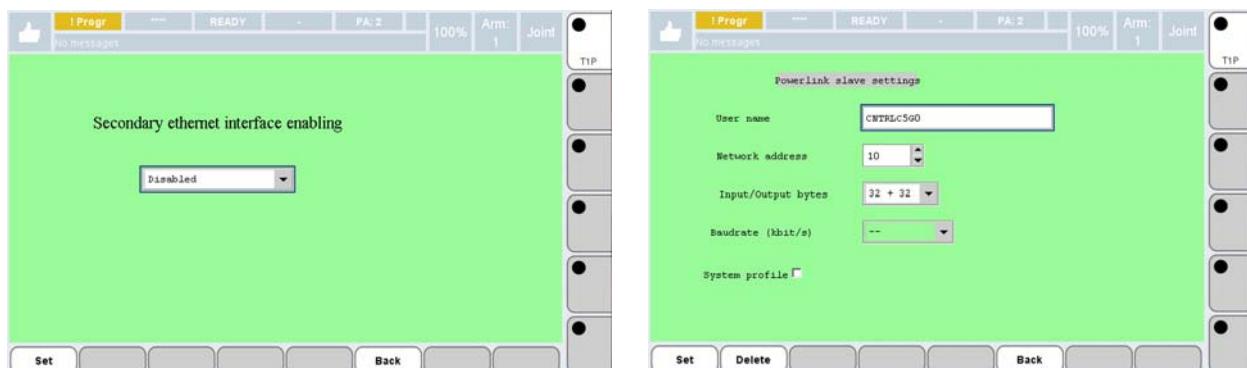
Touch **Annex** key and select the wished item:

- Powerlink network - master, slave or C5GOpen
- Ethernet network.



According to the choice made by the user, the corresponding page is opened to enter data for either setting or enabling/disabling the specified module. In the figures below, two examples are shown:

- enabling the secondary Ethernet interface (left figure),
- setting a Slave Powerlink annex (right figure).

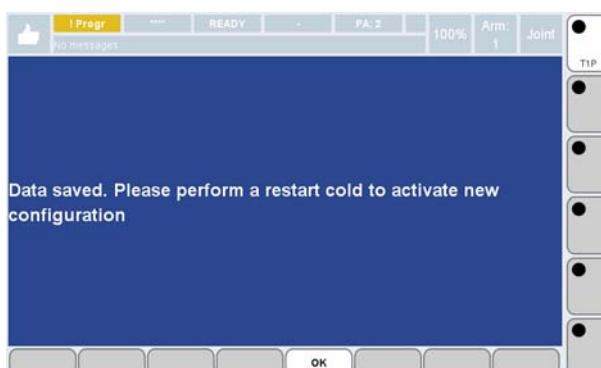


In order to operate about the wished annex modules, touch the corresponding fields, select the wished choice and touch **OK** key to confirm it.

Touch **Set** key to finally confirm all the setting.

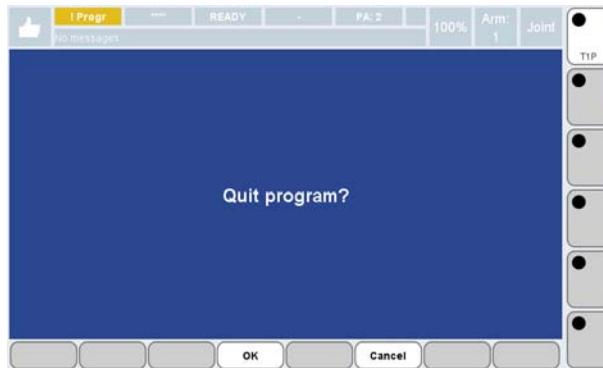
11.1.6 Save

This command saves all the performed modifications, without asking for any further confirmation. An informational message tells the User that data saving has been executed and that a **Restart Cold** operation is needed to make the configuration operational.



11.1.7 Exit

This command allows quitting **IO_CNFG** program. A suitable message prompts the user for confirmation. Touch **OK** key to exit.



The performed modifications, if not previously saved by means of **Save** command, are lost.

11.2 FB_util Program - Fieldbus modules utility

11.2.1 Activation of FB_util

This program, **implemented in Visual PDL2 language**, acts on the FIELDBUS networks whose I/O Controller (Master) is the C5G Control Unit, and allows detecting the presence of Fieldbus modules on the network.

Furthermore, for Ethernet based Fieldbuses it handles the modules replacement and specifying and/or inserting some related data (e.g. the name of the Profinet network devices).

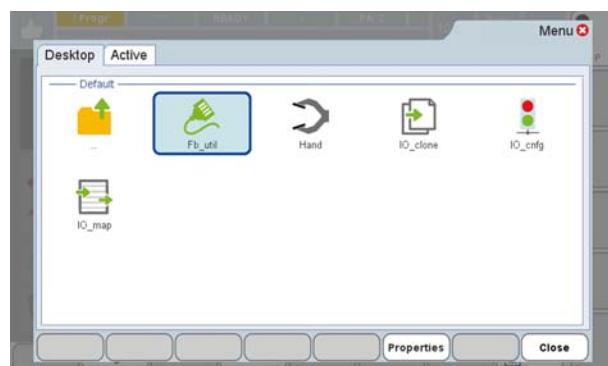
The ***FB_util*** program is always present and available on the Teach Pendant.

To use it

- open the [Setup page](#) on the Teach Pendant
- touch **IO** icon



- touch **Fb_util** icon.



- The system runs ***Fb_util*** program (see the following figure).

In the home page of ***Fb_util*** program, the following functionalities are available:

- [Profinet](#)
- [Profibus](#)
- [DeviceNET](#)
- [Quit.](#)

**Back**

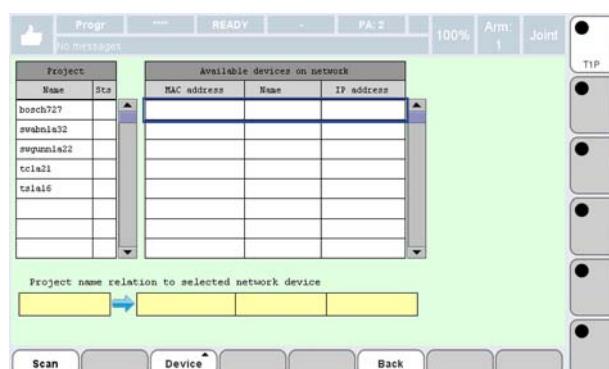
Note that in the screen pages of the further levels, **Back** key is always present to allow quitting such a step of the procedure, and going back to the previous one.

11.2.2 Profinet

This command allows to operate on the PROFINET network devices, whose I/O Controller (Master) is the C5G Control Unit, viewing them and associating them to the names already included in the network project implemented on **SYCON.net**.

Such an activity is needed at the network first installation time as well as when a new module is added. It is also needed when a module must be replaced. For further details, see the following [par. 11.2.2.1 Operational procedures on page 446](#).

Fig. 11.8 - Profinet - Tables



The tables included in the page (see [Fig. 11.8](#)) are as follows:

– Project

lists all devices inserted at project level (by means of **SYCON.net** program - see [PROFINET Network in Cap. Project of a Master Fieldbus Network on page 449](#)).

This table has got two columns:

- **Name** - is the name of the device previously inserted at project level
- **Sts** - indicates whether such a device is connected or not.

– Available devices on network

lists all the currently present devices onto the Profinet network. This table has got three columns:

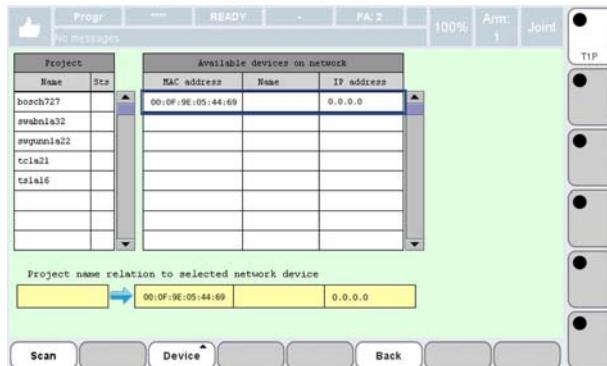
- **MAC address** - is the device MAC address

- **Name** - is the wished name to refer to it within all the system environments.
- **IP address** - is the device IP address.

Mediante il comando **Scan**, il programma esegue una scansione per verificare la presenza di dispositivi attualmente sulla rete e inserisce le corrispondenti informazioni nella tabella **Available devices on network** (vd.[Fig. 11.9](#)).

By means of **Scan** command, the program performs a scanning operation to check the presence of any devices currently on the netwok and inserts the corresponding information into **Available devices on network** table (see [Fig. 11.9](#)).

Fig. 11.9 - Scan - Available devices on network



Given a MAC address, it is not so easy to understand which is the corresponding physical device.

In order to physically identify it, the following command is available

Device -> Signal -> ON

Such a command activates the identification functionality on the device at the specified MAC address (e.g. switching on a led or some other similar functions, depending on the device type).



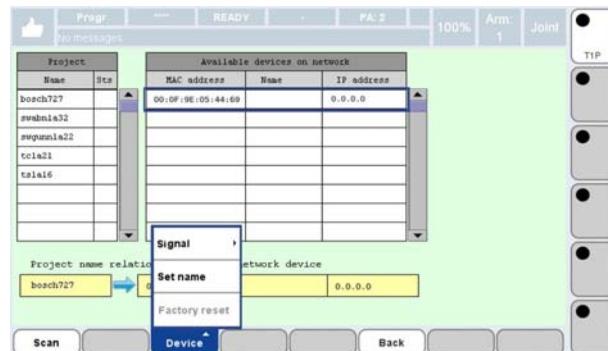
As soon as the device has been identified, switch off the identification functionality, by means of the following command

Device -> Signal -> OFF.

If such a device is the wished one, select its name in the **Project** table and go ahead with device-name association, by means of the following command (see the figure below)

Device -> Set name

Fig. 11.10 - Name-device association



The selected name is then associated to the specified device and, if it is the right one, such a device is declared as 'connected' (by means of the word **OK** in the corresponding **Sts** column of [Project](#) table - see below figure)



If the model of the selected module is wrong, the name-device association is anyway performed, but such a device IS NOT declared as ‘connected’.

By pressing **Scan** key again, the program reads the corresponding **IP address** for the connected device, and inserts it into the corresponding column of [Available devices on network](#) table (see [Fig. 11.11](#)).

Fig. 11.11- Scan - IP address



If going back to the initial situation is required, for a single device, select it in the [Available devices on network](#) table and issue the following command (see Fig. 11.12)

Device -> Factory reset

Fig. 11-12- Device - Factory reset



When the wished operations have been accomplished, press **Back** key to go back to the **FB_util** program start page.

11.2.2.1 Operational procedures

Useful procedures related to PROFINET network:

- [First installation procedure](#)
 - [Device replacement procedure](#).
-

11.2.2.1.1 First installation procedure

In case of first installation in the PROFINET network, after having

- designed the network by means of the programs for [Creating a new project](#),
 - physically connected all the devices onto the network and
 - configured the network by means of [IO_CNFG Program - I/O modules configuration](#),
- all the devices will result as NOT connected.

It is needed to associate the devices names which had been defined at design time, to the MAC addresses of the devices currently connected onto the physical networkE'.

Then the User has to use **FB_util** program, according to the following steps:

- a. choose **Profinet** - the program displays the page shown in [Fig. 11.8](#): the devices included in the Profinet network design are listed in the left table (called the [Project](#) table); nothing is listed in the right table (called the [Available devices on network](#) table).
- b. Touch **Scan** key to ask for the list of the currently present devices onto the network (see [Fig. 11.9](#)).
- c. Select the device whished to be visually detected, from the [Available devices on network](#) table.
- d. If it is not possible to read the MAC address which is printed onto the object, touch **Device** key; otherwise directly go to j. step;
- e. choose item **Signal**
- f. and then select **ON** item - such a command activates a signalling functionality onto the selected device (e.g. blinking a LED or similar functionalities, depending on the device type).
- g. As soon as the device has properly been detected, touch **Device** key
- h. choose **Signal** item
- i. and then select **OFF** item to deactivate the functionality.
- j. Select, from [Project](#) table, the being connected device name, assigned at the design time;
- k. Select, from the [Available devices on network](#) table, the being connected device MAC address;

- i. touch **Device** key,
 - m. choose **Set name** - **FB_util** program connects the selected device. This is notified by **FB_util** program, both by means of a suitable message in the message line and by the word **OK** in the **Sts** column ([Project](#) table), for the corresponding connected device.
 - n. Repeat the procedure for any other device to be connected onto Profinet network, starting again from c. step.
 - o. At the end touch **Scan** again, to update the tables: the IP address is then displayed too for each connected device (see [Fig. 11.11](#)).
-

11.2.2.1.2 Device replacement procedure

Whenever replacing a device is needed, execute the following steps:

- a. physically dismount the device
- b. replace it with an identical one
- c. Touch **Scan** key to ask for the list of the currently present devices onto the network (see [Fig. 11.9](#)).
- d. Select the MAC address of the just inserted device, from the [Available devices on network](#) table.
- e. If it is not possible to detect it for sure, execute steps from d. to i. of previous [First installation procedure](#);
- f. select the name of the just inserted device, from the [Project](#) table,
- g. choose **Signal** item
- h. and then select **ON** item - such a command activates a signalling functionality onto the selected device (e.g. blinking a LED or similar functionalities, depending on the device type).
- i. As soon as the device has properly been detected, touch **Device** key
- j. choose **Signal** item
- k. and then select **OFF** item to deactivate the functionality.
- l. Select the being connected device MAC address, from the [Available devices on network](#) table,
- m. touch **Device** key,
- n. choose **Set name** item - **FB_util** program connects the selected device. This is notified by **FB_util** program, both by means of a suitable message in the message line and by the word **OK** in the **Sts** column ([Project](#) table), for the corresponding connected device.
- o. At the end touch **Scan** key again, to update the tables: the IP address is then displayed too for each connected device (see [Fig. 11.11](#)).

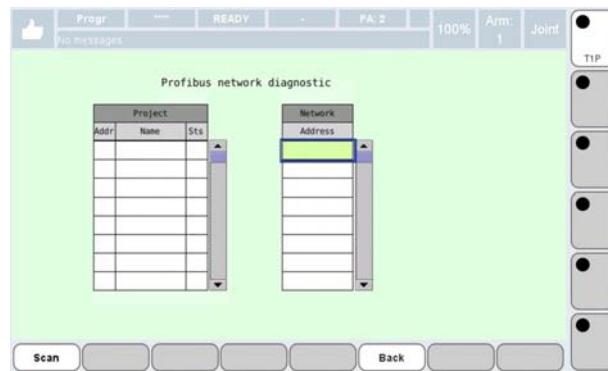
11.2.3 Profibus

It is just a diagnostic functionality: it allows to scan the network and detect the currently present devices.

When this modality is selected, the page shown in Fig. 11.13 is displayed.

By means of **Scan** command, the system is asked to detect and list all the currently present devices onto the **Profibus** network.

Fig. 11.13- Profibus



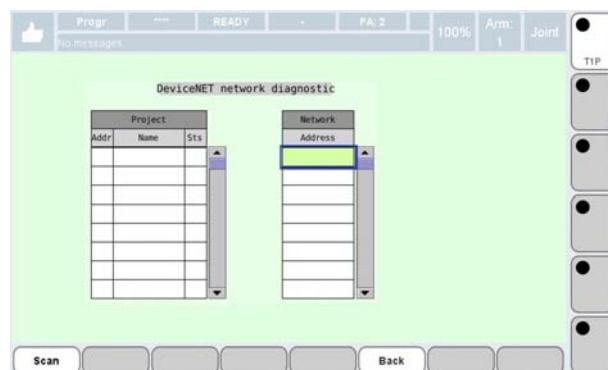
11.2.4 DeviceNET

It is just a diagnostic functionality: it allows to scan the network and detect the currently present devices.

When this modality is selected, the page shown in Fig. 11.14 is displayed.

By means of **Scan** command, the system is asked to detect and list all the currently present devices onto the **DeviceNET** network.

Fig. 11.14- DeviceNET



11.2.5 Quit

This command causes to quit **FB_util** program.

11.3 Project of a Master Fieldbus Network

The project of a Master Fieldbus network, is performed by means of two programs, **SYCON.net** and **WinC5G**, available in the System Software CD.

During **SYCON.net** installation phase, the User is asked whether such a program has to be installed for any PC User or for the current User only. The suggestion is to make the first choice (for any User).

- Creating a new project
- Modifying a project already saved onto the Controller.

11.3.1 Creating a new project

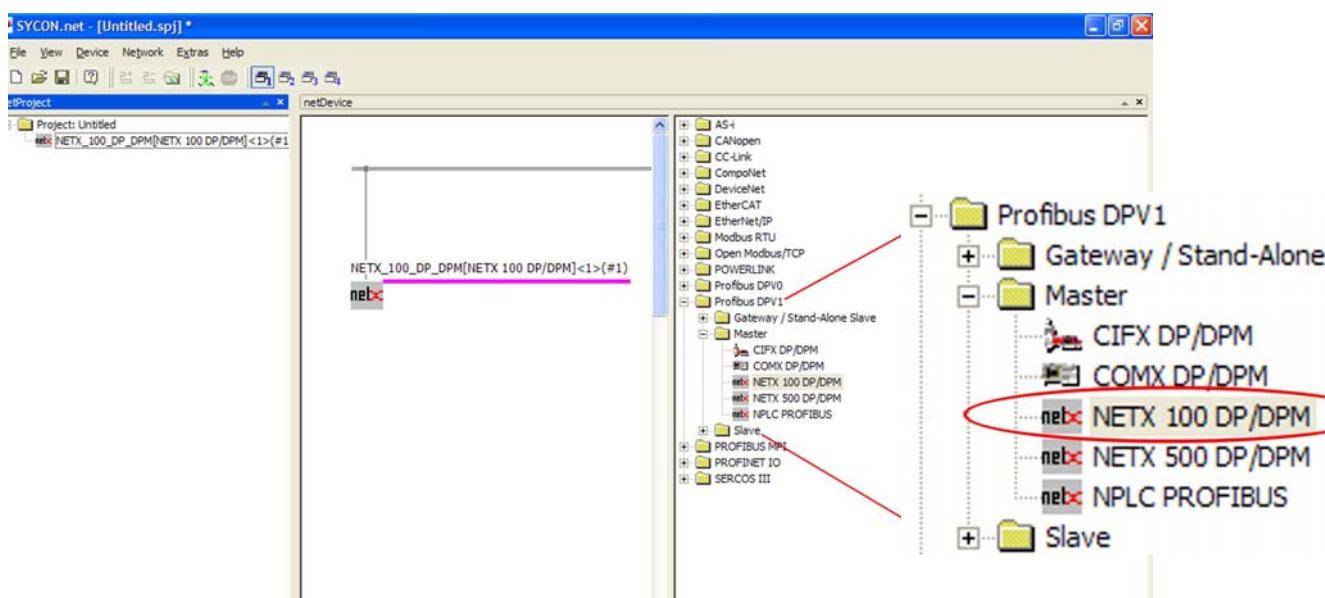
Creating a Fieldbus Master network project is made into two phases which use two different programs:

- Phase one - SYCON.net program
- Phase two - WinC5G program.

11.3.1.1 Phase one - SYCON.net program

- a. run **SYCON.net** program
- b. during **SYCON.net** installation, it is needed to insert a password which will be asked and checked later, every time the program is run.
- c. As the Master for Profibus, DeviceNet and PROFINET networks, respectively, choose the option with **NETX 100** (e.g. the module highlighted in red in Fig. 11.15, for the Profibus network). A description follows for creating a new project for a Master Profibus network. Then, a section follows containing the **Differences in creating a Master DeviceNet network**.

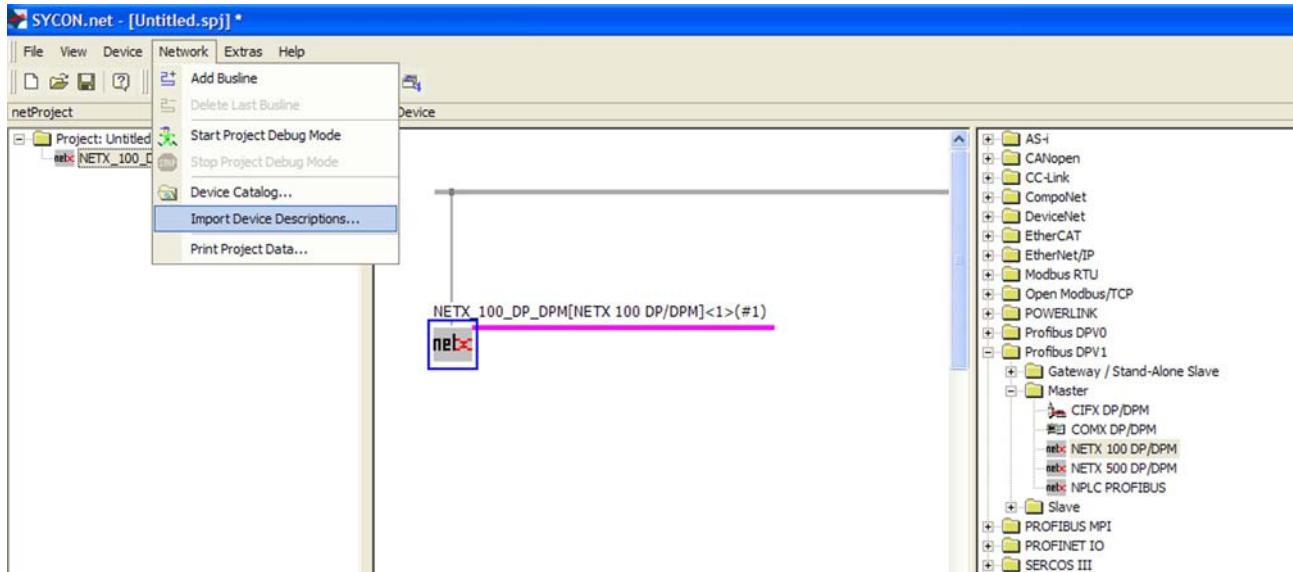
Fig. 11.15- Choosing the Master module - Profibus



d. When the Slave is not present in the catalog yet, execute the following steps:

d.1 Open the **Network** menu

Fig. 11.16- Slave not present in the catalog

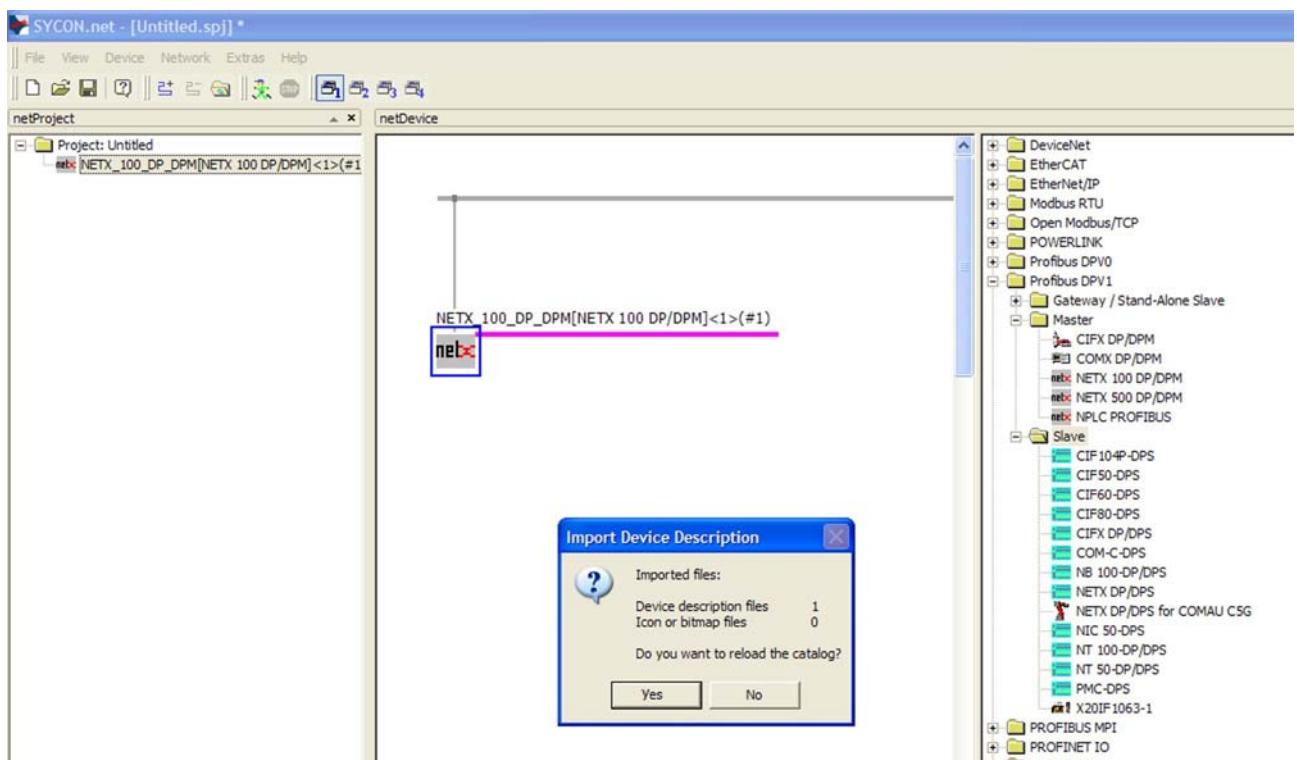


d.2 Select item **Import Device Descriptions** (see Fig. 11.16)

d.3 Select the descriptor file for the wished configuration (the file extension for Profibus network is **.gsd**)

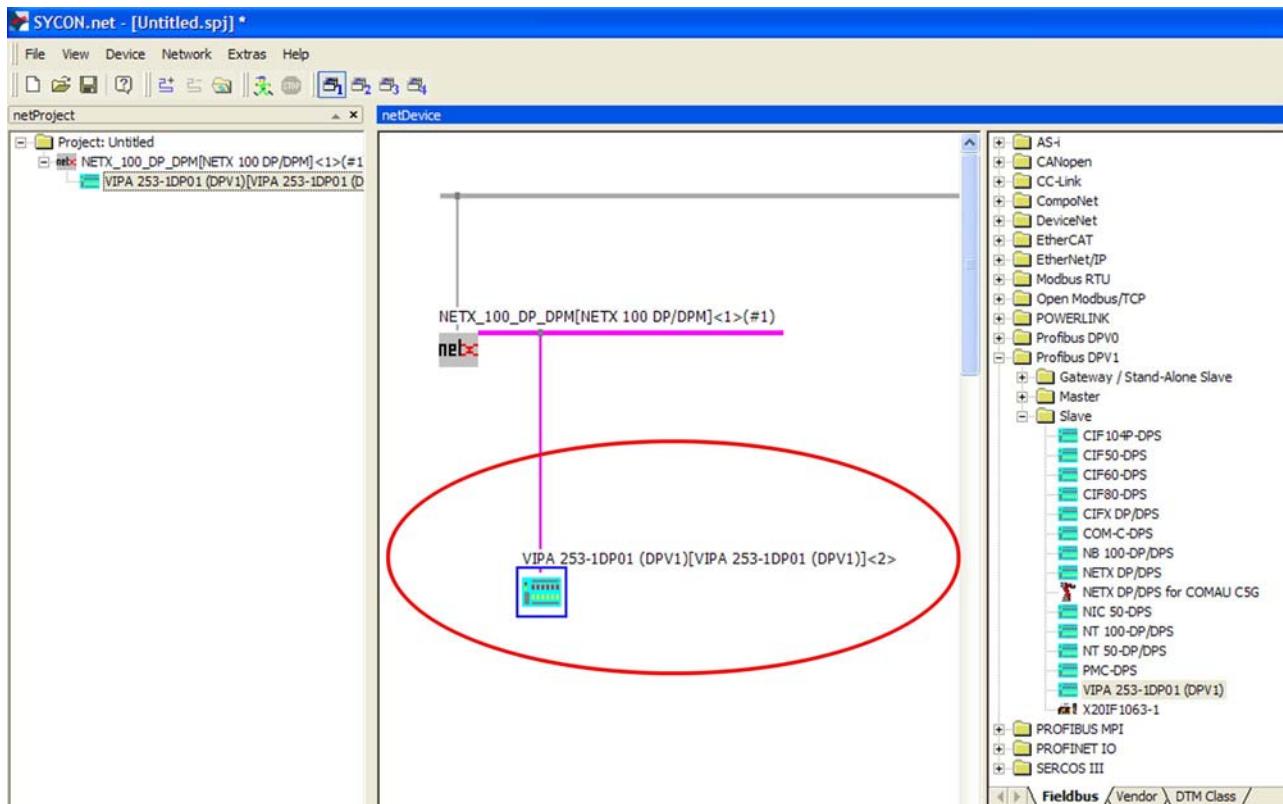
d.4 Answer **Yes** to Reload the Catalog (see Fig. 11.17)

Fig. 11.17- Reload the configuration descriptors catalog



- e. The wished Slaves are inserted onto the Master network, by dragging & dropping them (see Fig. 11.18, highlighted in red) from the **Slave** directory.

Fig. 11.18- Inserting a Slave



- f. Assign a symbolic name to the Slaves (see Fig. 11.20), i.e.:
- move the cursor to the wished Slave
 - click the mouse right key
 - choose **Symbolic Name** item.
- Please note that in Fig. 11.18 and Fig. 11.20 screens, the symbolic name contains NOT allowed characters, since it has still to be modified by the user.

Starting from Fig. 11.21 screen on, the chosen new symbolic name is displayed, according to the proper syntax. In the described example, the chosen name is **VIPA253_10**.



NOTE ABOUT THE SYMBOLIC NAME SYNTAX

The chosen name must be unique in the system, according to PDL2 variables naming rules. For example, characters such as -, +, *, (,), /, \, :, blank, etc. are NOT allowed. To choose a syntactically correct name, it is suggested to only use alphanumeric characters and underscore (_) character; please, remember that underscore character is NOT allowed for PROFINET network. Furthermore, it is NOT allowed to use PDL2 language reserved words!

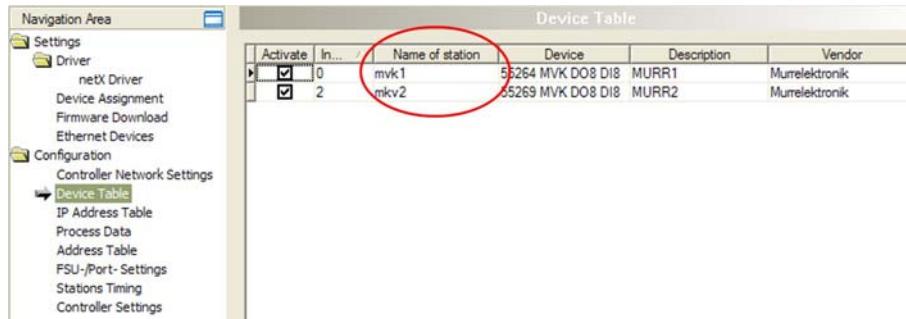
It is also suggested not to use too long names (e.g. not more than 8 characters).



PROFINET Network

WARNING! In case of PROFINET network, choosing Symbolic Name to specify the Slave Name does not take any effect. In such a situation the User must choose the Master Configuration: in the Master Navigation Area, select “Configuration”, “Device Table” and specify the Slave name in “Name of Station” column (see Fig. 11.19 highlighted in red).

Fig. 11.19- Assigning a name to the Slaves - PROFINET case



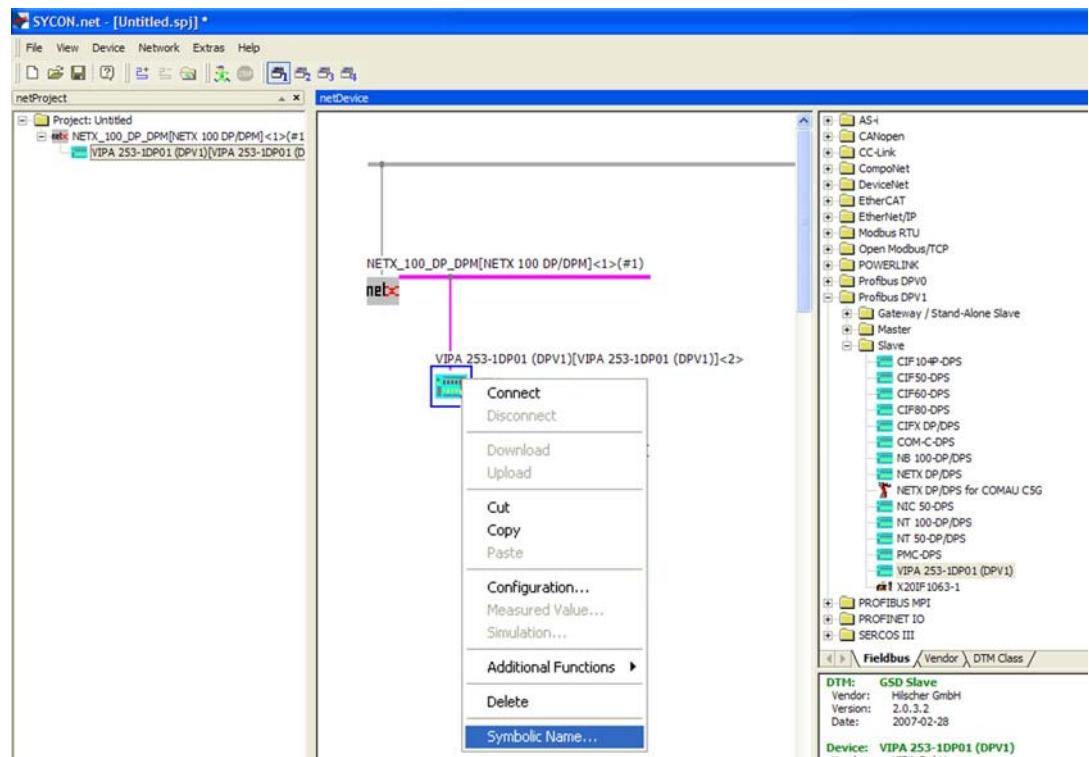
Activate	In...	Name of station	Device	Description	Vendor
<input checked="" type="checkbox"/>	0	mvk1	55264 MVK D08 DI8	MURR1	Murelektronik
<input checked="" type="checkbox"/>	2	mvk2	55269 MVK D08 DI8	MURR2	Murelektronik



NOTE - Insert a name which is unique in the system and comply the PDL2 variables naming rules.

To do so, it is suggested to just use alphanumeric characters and the underscore (_) character; remember that the underscore character is NOT allowed for PROFINET network. It is NOT allowed to use reserved words belonging to PDL2 language!! Furthermore, it is suggested not to use too long names (e.g. maximum length of 8 characters).

Fig. 11.20- Assigning a name to the Slaves - general case



g. Go on with the Slave configuration, i.e.:

- g.1 move the cursor to the wished Slave
- g.2 click the mouse right key
- g.3 choose **Configure** item



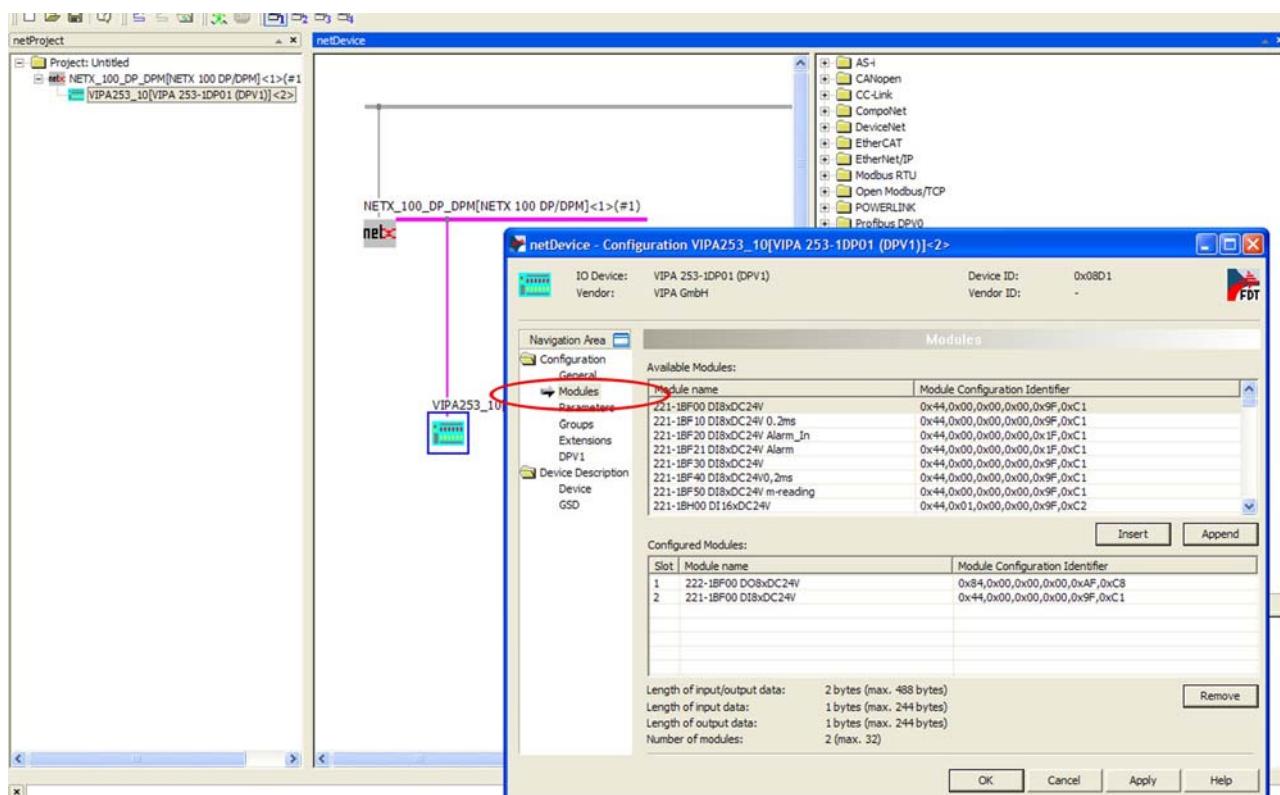
If the Slave is C5G Controller, when the being used bytes are selected (in the Modules window), the only allowed options are as follows:

- **8 Input bytes and 8 Output bytes,**
- **16 Input bytes and 16 Output bytes,**
- **32 Input bytes and 32 Output bytes,**
- **64 Input bytes and 64 Output bytes.**

These options are mutually exclusive: one and just one option must be selected.

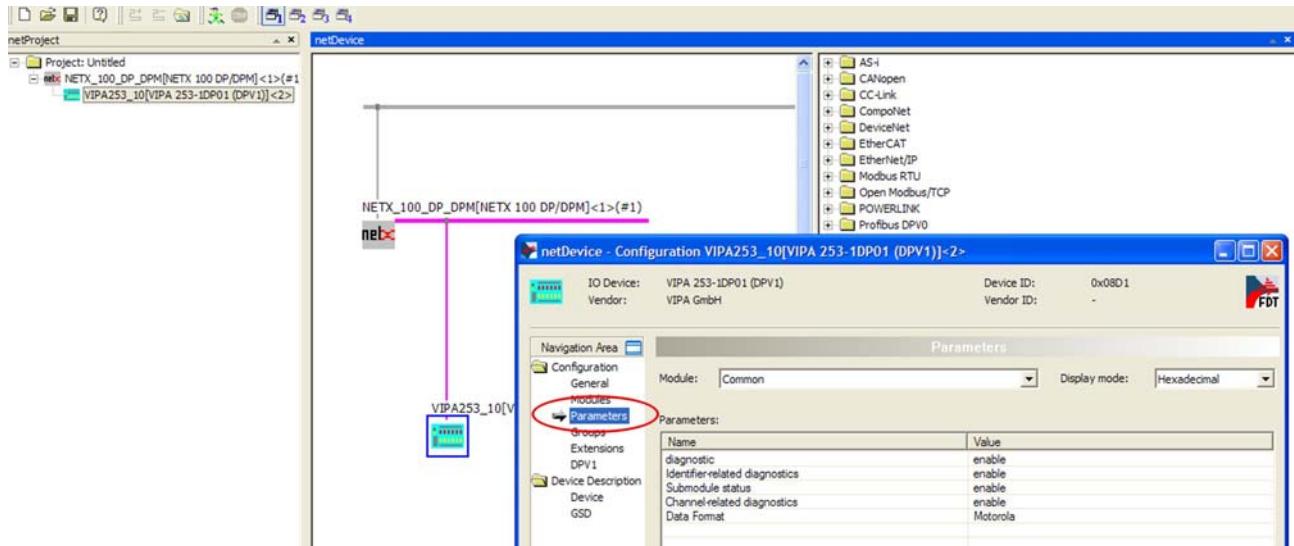
- g.4 select **Modules** item in the Navigation Area (see Fig. 11.21 highlighted in red)
- g.5 choose the modules which are present in the User's Hardware and insert them in the proper sequence. For a detailed description of such operations, please refer to the specific **SYCON.net** online manual
- g.6 Press **Apply**

Fig. 11.21 - Configuring the inserted Slaves



- g.7 Each device has got its own parameters to be configured. Set them by selecting **Parameters** item in the Navigation Area (see Fig. 11.22 highlighted in red)

Fig. 11.22- Selecting the parameters



g.8 Press **Apply** to confirm the modifications.

g.9 Depending on the Slave supporting or not the DPV1 protocol, enable/disable such a functionality (select **DPV1** item to access the enable/disable checkbox).

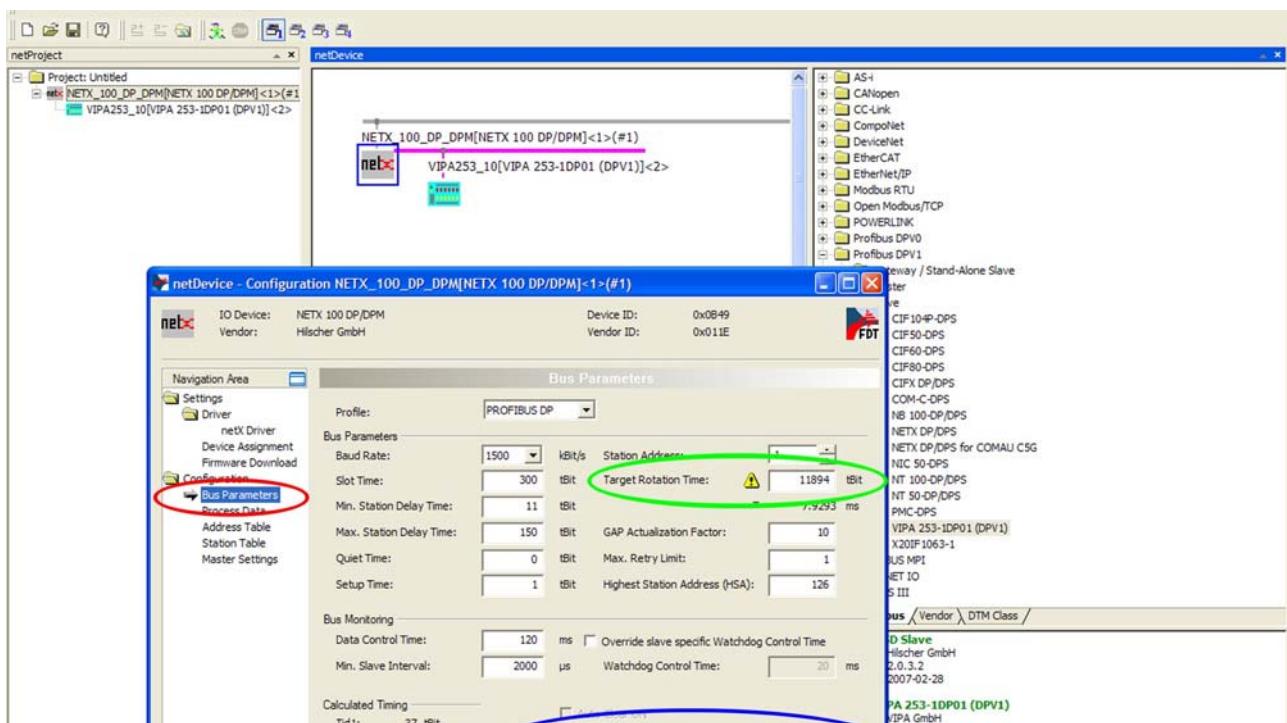
g.10 When the Slaves configuration is completed, press **OK** to confirm all the inserted data.

h. Move the cursor to the Master module

i. click the mouse right key and choose **Configuration**

j. select **Bus Parameters** item (see Fig. 11.23, highlighted in red)

Fig. 11.23- Configuring the Master module - Bus Parameters



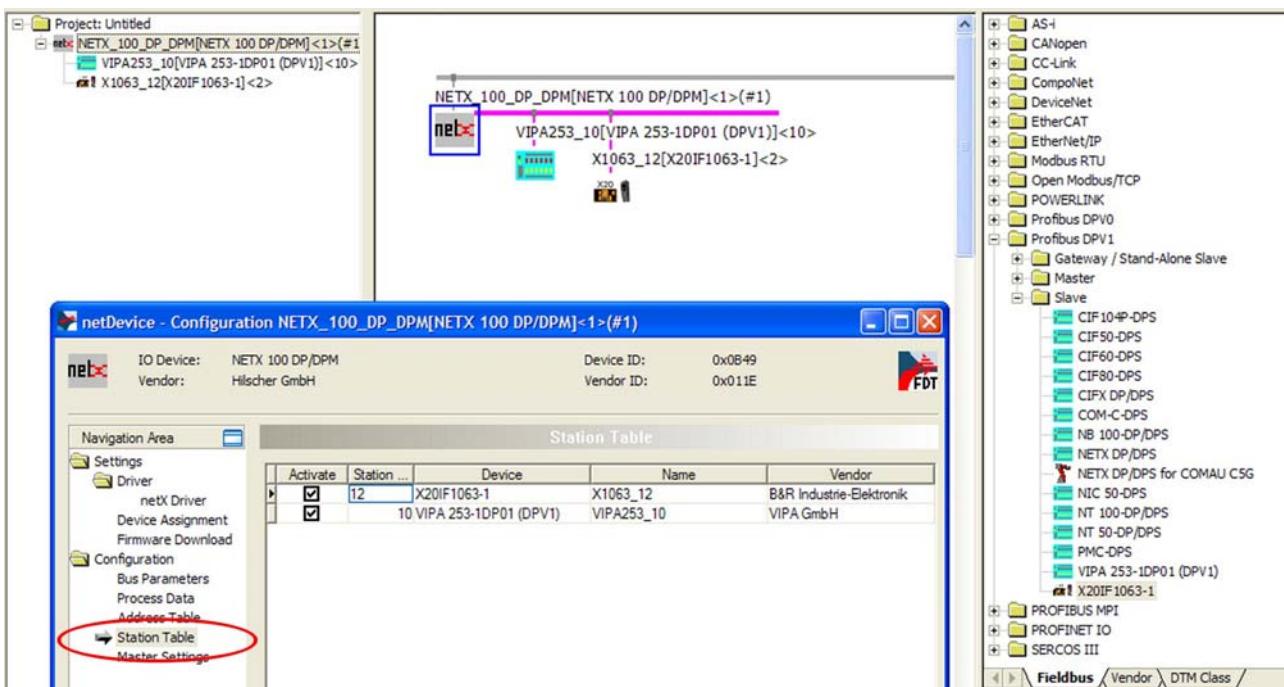
- k. insert the suitable values, e.g. the Baud rate
- l. fix any value marked by an exclamation mark (see Fig. 11.23, field highlighted in green), by pressing the **Adjust** key (highlighted in blue in Fig. 11.23).



Note that such an operation must be executed when ALL the Slaves have already been inserted!

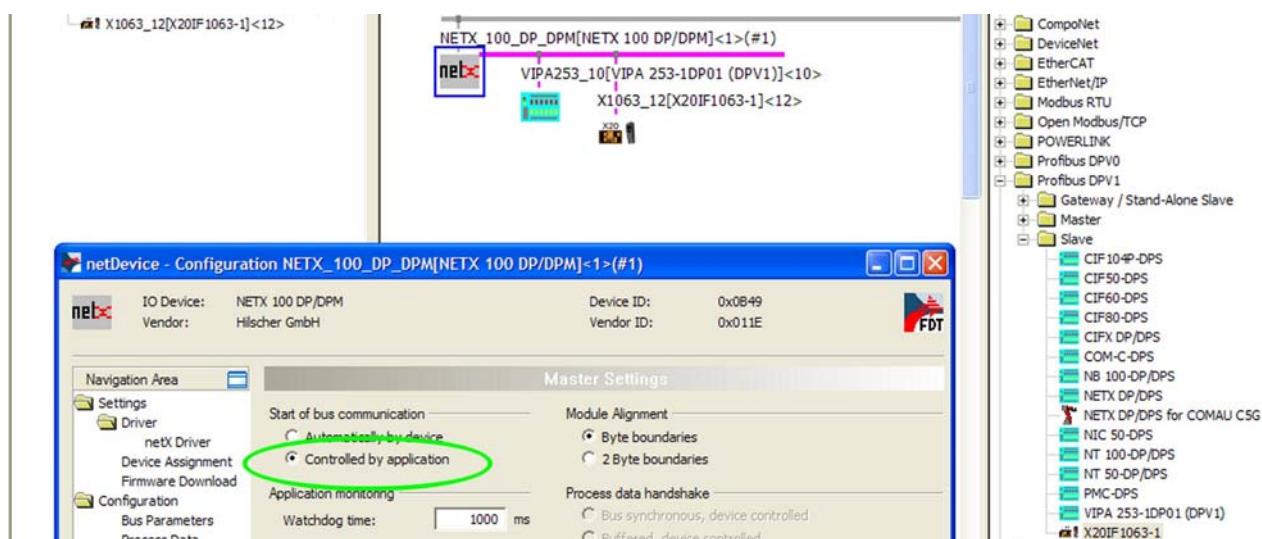
- m. Select **Station Table** item (see Fig. 11.24 highlighted in red). This screen page allows setting the Slaves addresses.

Fig. 11.24- Configuring the Master module - Station Table



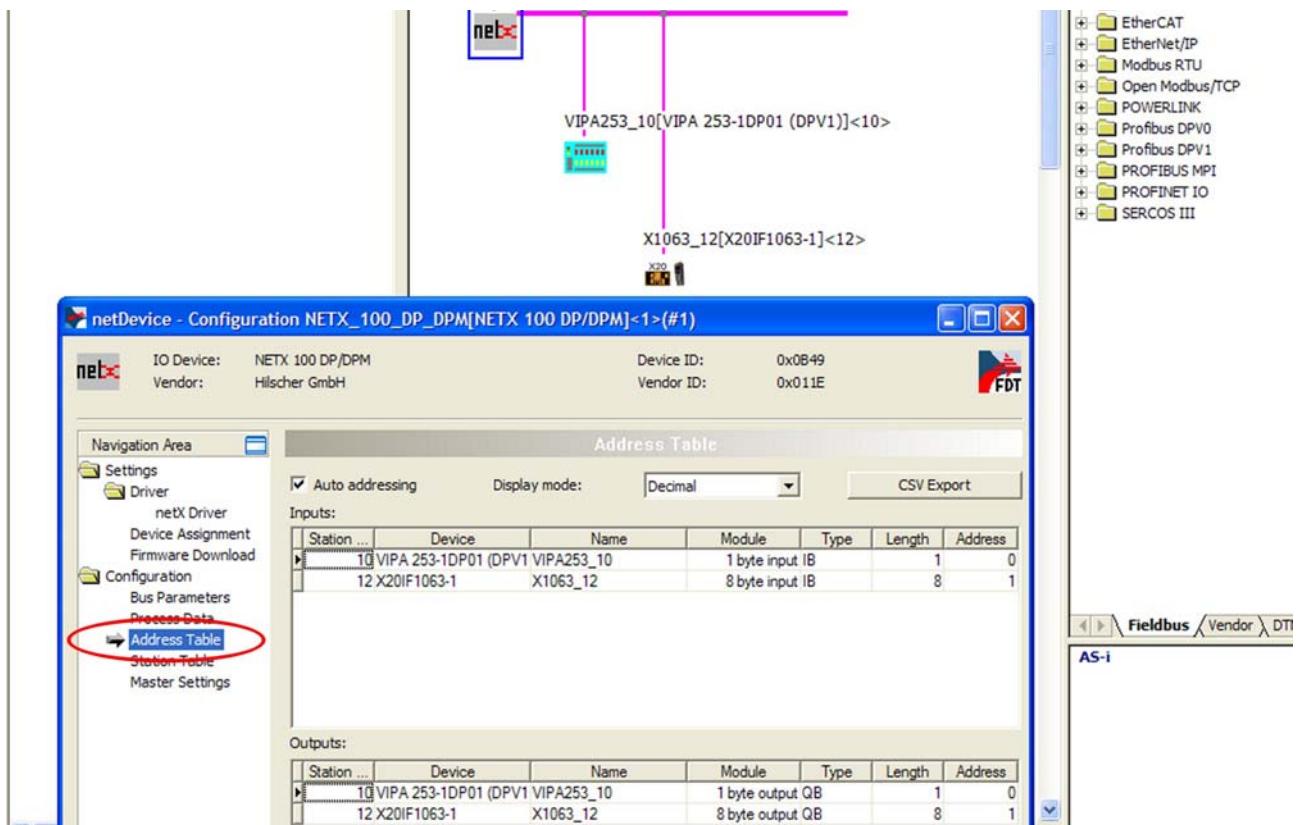
- n. Select **Master Settings** item (see Fig. 11.25 highlighted in red)

Fig. 11.25- Configuring the Master module - Master Settings



- o. select **Controlled by application** radio-button (see Fig. 11.25, highlighted in green).
- p. Press **OK**.
- q. Select **Address Table** item (see Fig. 11.26 highlighted in red). This summary page contains the same information that will be included in the **.csv** file, in ascending order of address, one table for Inputs and one for Outputs.

Fig. 11.26- Configuring the Master module - Address Table



- r. At the end of the project activity, go on with saving operation; **SYCON.net** program generates a file, with extension **.spj** and a directory with the same namee una.



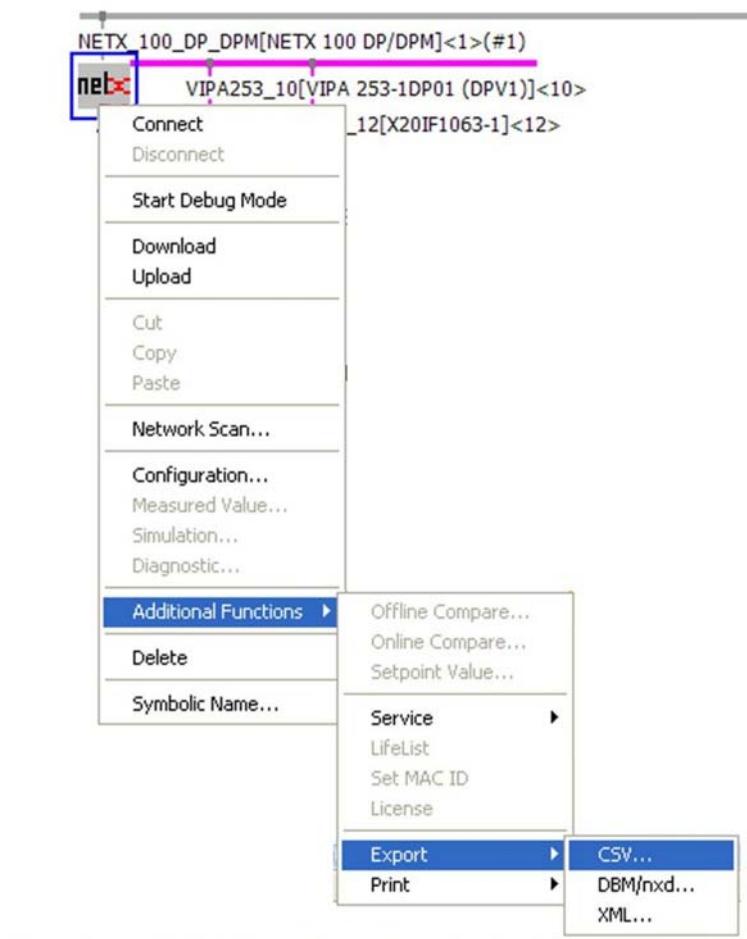
The project name, used both for **.spj** file and the homonymous directory, must only include alphanumeric characters, with neither special characters nor blanks. It is NOT allowed to use reserved word belonging to PDL2 language!
The underscore (' _ ') character is allowed.

- s. Generate the needed files for exporting the project towards the Robot Controller (see Fig. 11.27). The below described procedure must be executed both for **.csv** files (**CSV** item in the menu) and for **.nxd** files (**DBM/nxd** item in the menu). The file names to be used for exporting, must be the same of the **.spj** file name (e.g. if the project name is **PROJ25.spj**, the being exported files are called **PROJ25.csv** and **PROJ25.nxd**).

- s.1 Move the cursor to the **Master**
- s.2 click the mouse right key

- s.3 choose **Additional Functions** item
- s.4 choose **Export** item
- s.5 choose the wished file type (sequentially, **CSV** and then **DBM/nxd**). As the destination directory, choose the same where **.spj** file is stored.

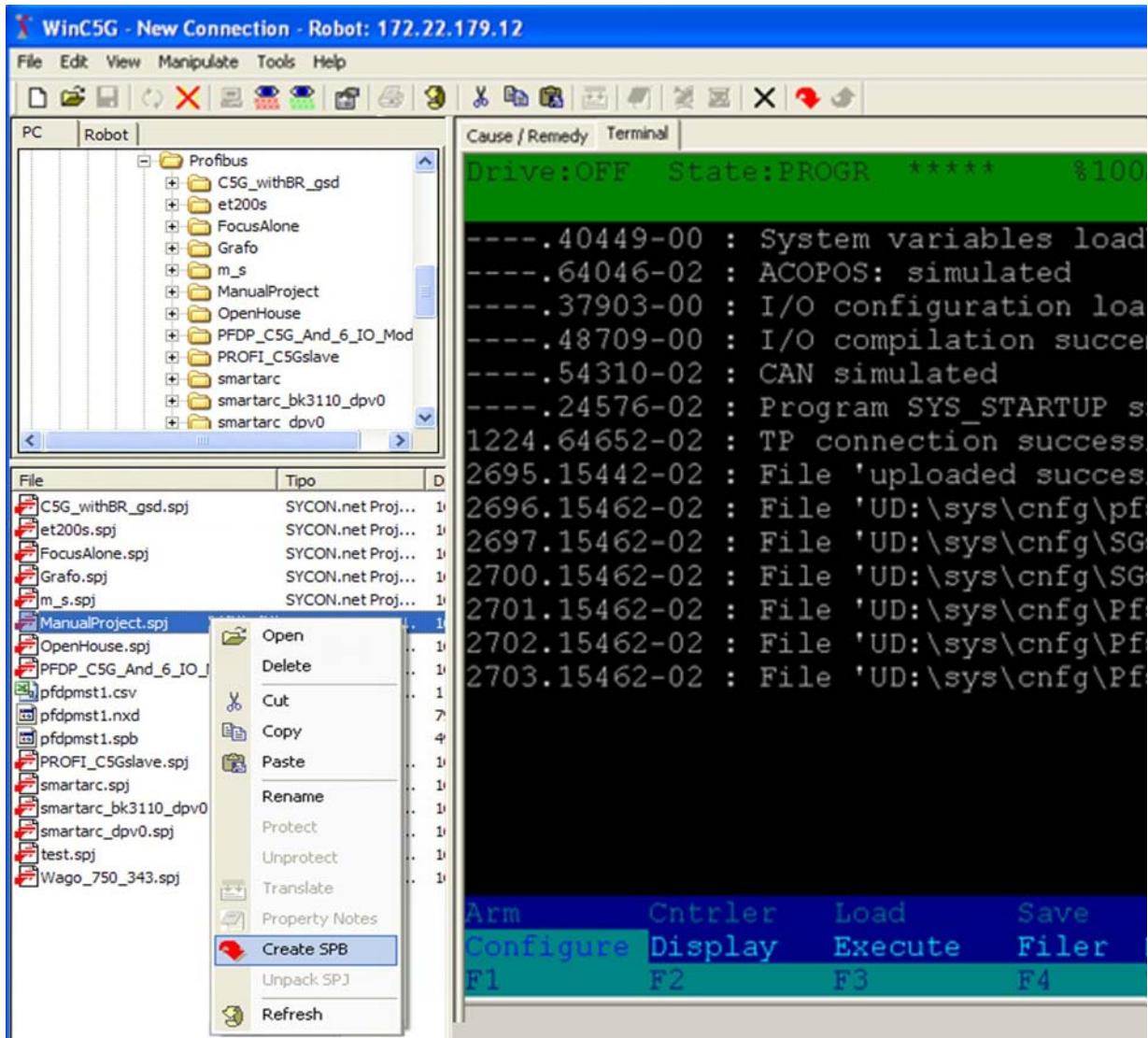
Fig. 11.27 - Creating .csv and .nxd files



11.3.1.2 Phase two - WinC5G program

- t. Run **WinC5G** program
- u. select **.spj** project file. In the same directory, the corresponding **.csv** and **.nxd** previously exported files, must be present.

Fig. 11.28- Creating .spb file



- v. Click the mouse right key
- w. choose **Create SPB** command (see Fig. 11.28).
- x. Select the fieldbus which the project refers to (see Fig. 11.28)
- y. A file is created, with **.spb** extension, containing **.spj** file, the directory with the same name, the exported files (**.csv** and **.nxd**) and the Slave modules descriptors used in the Fieldbus network (**.gsd** or **.gsdml** or **.eds**, depending on the chosen network; for Profibus the extension is **.gsd**).
- z. Transfer **.spb** file to the Controller, in the following directory:

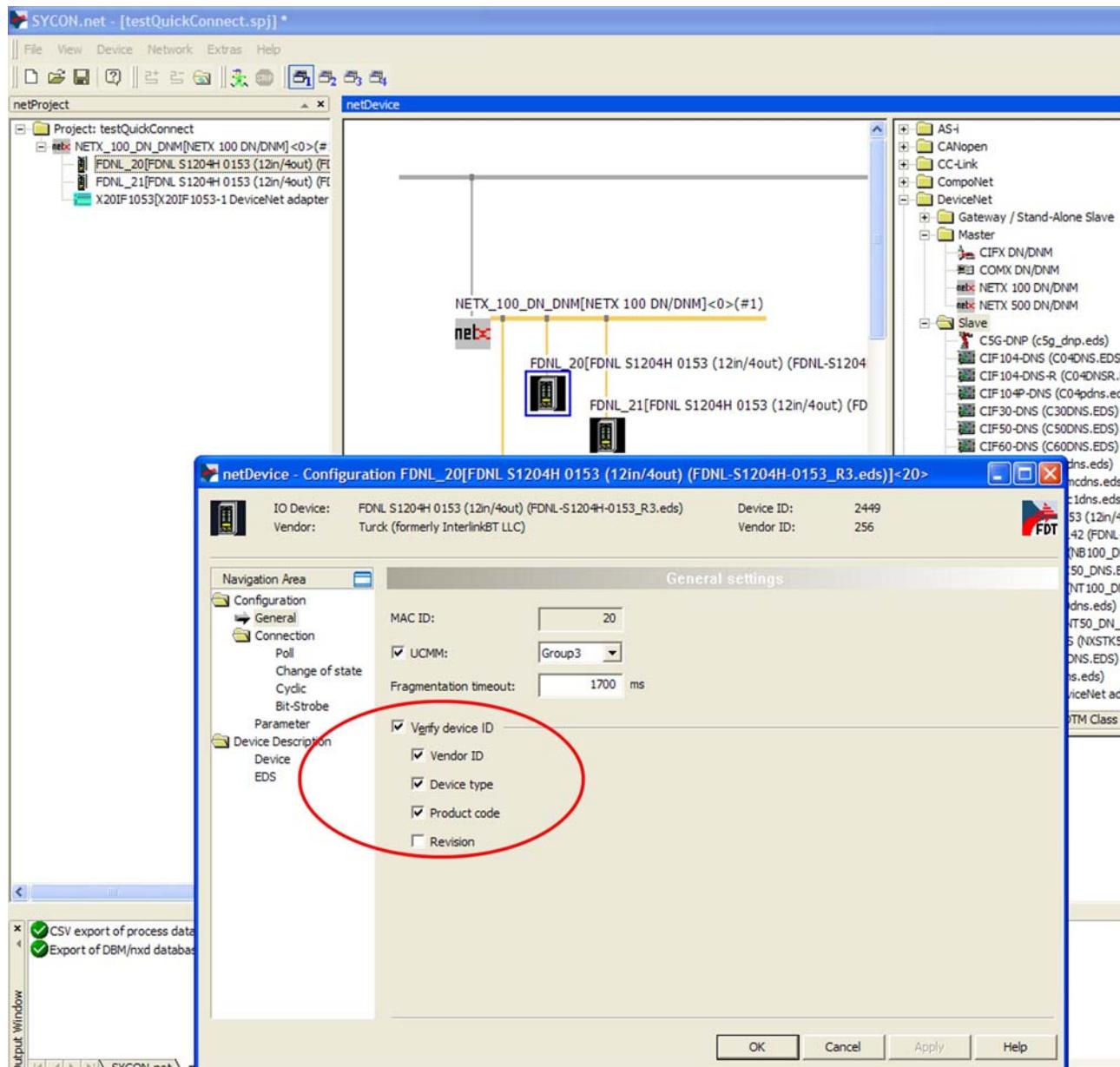
UD:\SYS\CFG

11.3.1.3 Differences in creating a Master DeviceNet network

Creating a project for Master DeviceNet network is similar to what already explained for the Master Profibus one. The only differences are as follows:

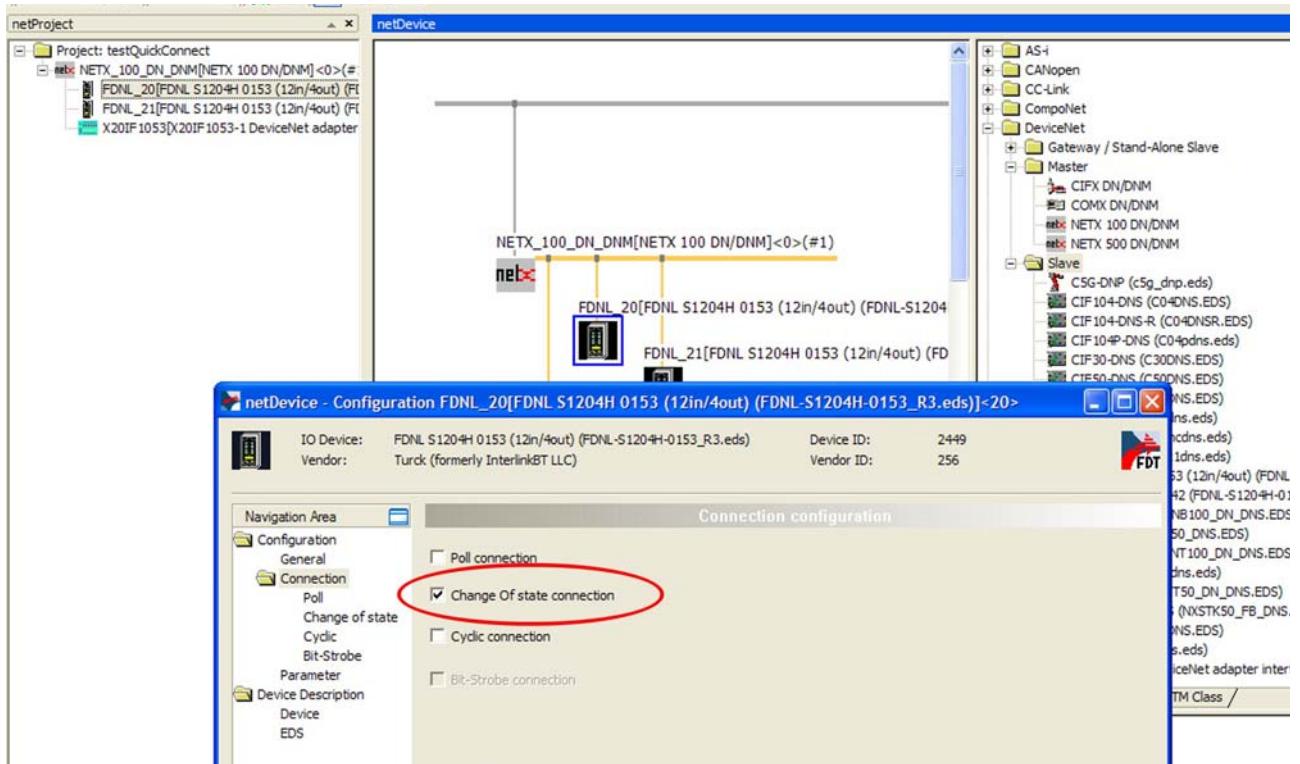
- Slaves configuration

Fig. 11.29 - DeviceNet - Slaves configuration - General



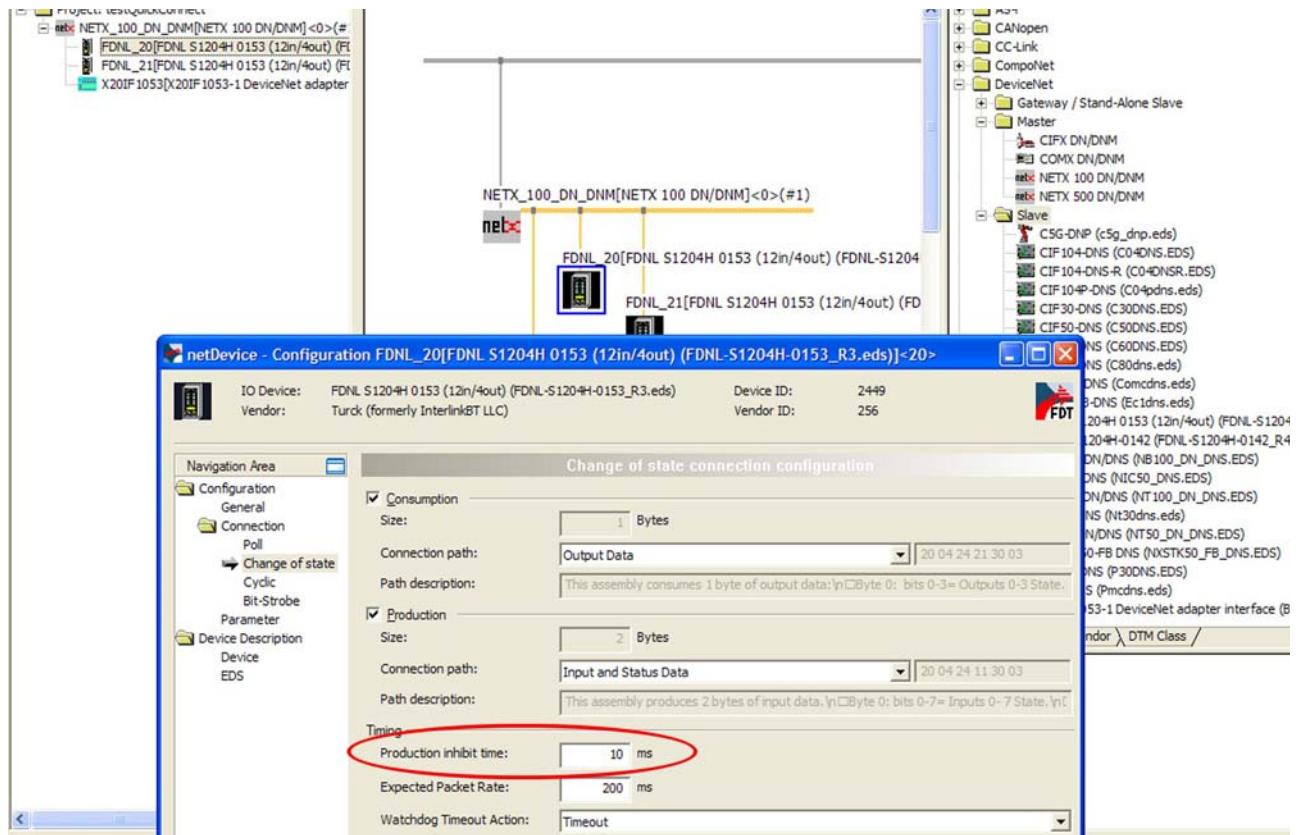
- choosing **General** area, select all the check-boxes related to **Verify device ID**, except **Revision** (as shown in Fig. 11.29, highlighted in red)
- choosing **Connection** area, if the module allows it, select **Change OF state connection** check-box (as shown in Fig. 11.30, highlighted in red).

Fig. 11.30 - DeviceNet - Slaves configuration - Connection



- choosing **Connection** area, if the module allows it, and **Change of state**, sub-area, set **Production Inhibit Time**; suggested values **10** or **20** or **30** ms (as shown in Fig. 11.31, highlighted in red).

Fig. 11.31 - DeviceNet - Slaves configuration - Change of state



- If, on the contrary, **Change OF state** is disabled, select **Poll** area and check that the **Production inhibit time** is NOT set to zero! Suggested values: **10** or **20** or **30** ms.
- Master configuration
 - In **Configuration** area, instead of **Station table** there is **MAC ID table** and, as **Bus parameters**, the only item to be modified is the network **Baud rate**. For the other items there are no differences.

11.3.2 Modifying a project already saved onto the Controller

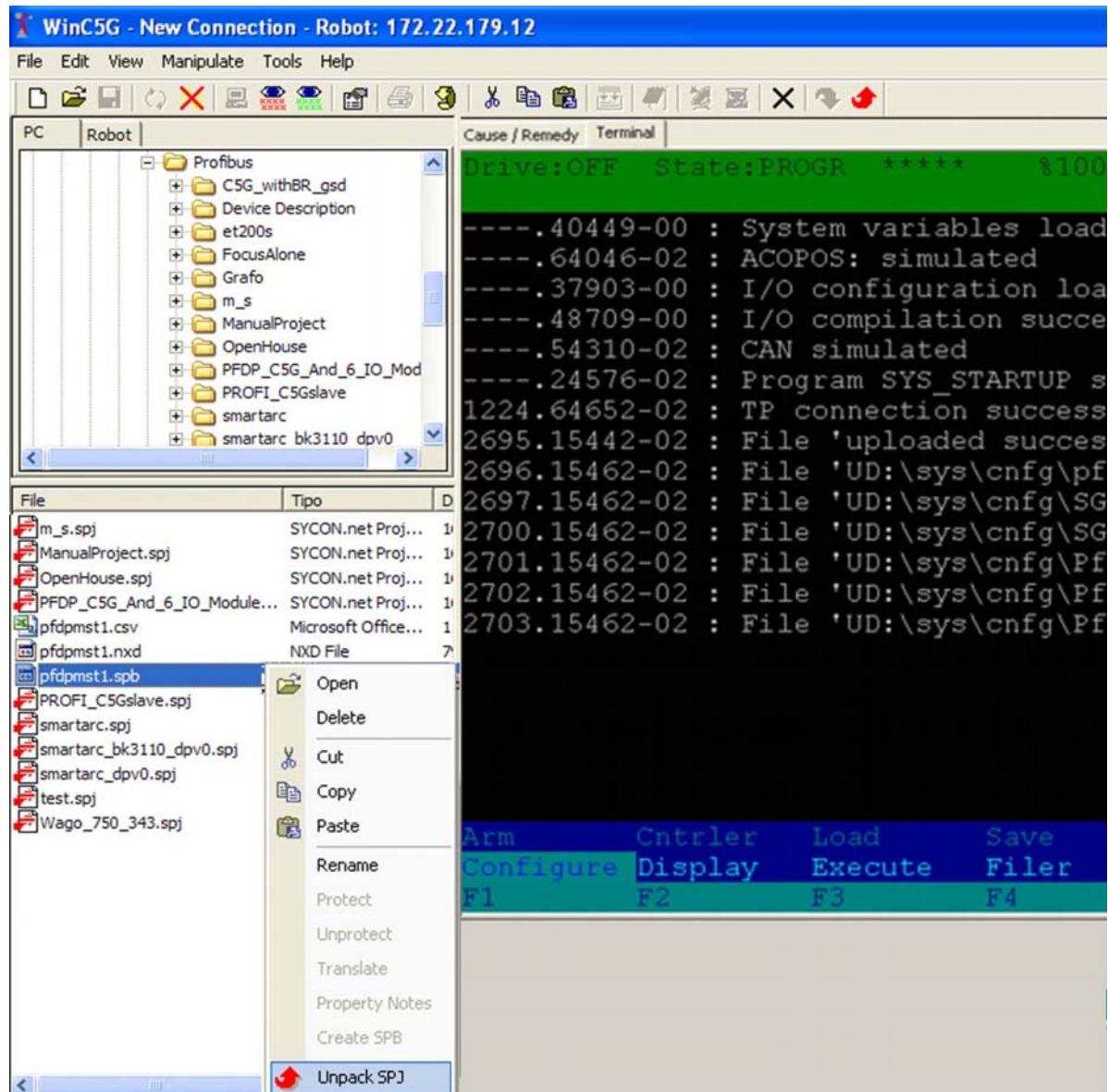
To modify a Master Fieldbus network project, two phases are performed using two different programs:

- Phase one - WinC5G
- Phase two - SYCON.net

11.3.2.1 Phase one - WinC5G

- a. Run **WinC5G** program
- b. Transfer **.spj** file from **UD:\SYS\CFG** directory to the work directory on the PC.
- c. Click the mouse right key

Fig. 11.32- Unpacking .spj file



- d. choose **Unpack SPJ** command - (see Fig. 11.32) **.spj** file, the homonymous directory and the Slave modules descriptors (which are saved to **Device Description** sub-directory) are extracted.
-

11.3.2.2 Phase two - SYCON.net

- e. Run **SYCON.net** program,
- f. open **.spj** file,
- g. if some descriptor files are not present (in such a situation a red **X** replaces their names), the User must import them into the catalog taking them from the **Device Description** directory;
- h. modify the project like what described for the procedure for creating a new project (see d., f. and g. steps of par. 11.3.1 Creating a new project on page 449).
- i. At the end of the project modification activity, go on with saving operation (see step r. of par. 11.3.1 Creating a new project on page 449).
- j. Execute steps s. through z. (of par. 11.3.1 Creating a new project on page 449), to update the project onto the Controller.

11.4 IO_MAP Program - I/O ports mapping

11.4.1 Activation of IO_MAP program

This program, **implemented in Visual PDL2 language** is used to map I/O points on C5G Controller Unit.

To map, using the **IO_MAP** Program, means to set a relationship among Input/output points belonging to any Devices, and **digital, analog, flexible** logical ports available in PDL2 language (ad es. \$DIN, \$DOUT, etc.).



Note that before going on with mapping I/O points, it is needed to have already configured their corresponding modules (see [IO_CNFG Program - I/O modules configuration](#)).

The **IO_MAP** program is always present and available on the Teach Pendant.

To use it, act as follows:

- open the [Setup page](#), on the Teach Pendant
- touch **IO** icon



- touch **IO_MAP** icon



- the system runs **IO_MAP** program



As shown in the figure above, in the home page of **IO_MAP** program the following functionalities are available:

- [Devices](#)
- [Port list](#)
- [Profile](#)
- [Link](#)
- [Save](#)
- [Exit.](#)



Back

Note that in all screen pages of all the existing levels, **Back** key is always present to allow quitting such a step of the procedure, and going back to the previous one.

11.4.2 Devices

This command allows to display the [list of all Devices currently existing in the System](#) and handle mapping their associated I/O points.



There's also the possibility of handling virtual devices defined as [USER Devices](#), useful for the installed applications in the System.

They apply to additional modules, application specific, such as welding timers, welding guns, etc.

Please refer to the specific description in the corresponding [par. 11.4.2.5 USER Devices on page 472.](#)



Available devices			
Device	Inp byte	Out byte	Mapped I/O
Virtual_dev_1	10	10	10 in/10 out
SDM	1	1	0 in/0 out
bosch727	8	8	0 in/0 out
mmchnla32	2	1	0 in/0 out
tclal21	6	2	0 in/0 out
tclal16	6	2	0 in/0 out
sequnlal22	6	2	0 in/0 out
CNTDL-C500	8	8	0 in/0 out

PAG 1/2

The table displayed in this environment contains the following information:

- **Device** - is the Device Name
- **Inp byte** - total amount of Input bytes currently associated to the selected Device
- **Out byte** - total amount of Output bytes currently associated to the selected Device
- **Mapped I/O** - indicates how many Input and Output points, associated to the selected Device, are already mapped.

Once the wished Device has been selected, the following commands are available in the Functional keys Menu:

- [Modify - Device](#)
- [Clean - Device](#)
- [Move - Device.](#)

In case in which the table includes more than one page, there are two more enabled keys, allowing to visit all the pages:

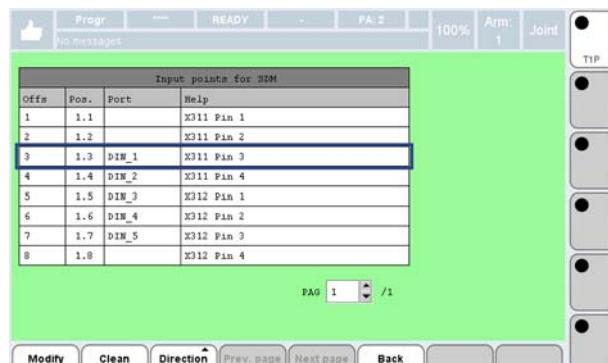
- [Prev.page](#), [Next page](#)

11.4.2.1 Modify - Device

In this sub-page a table is displayed containing the [list of all I/O points belonging to the selected Device](#) and the User is allowed to map them.



If the total amount of I/O points is too much high (e.g. 128 pages), it is useful to be able to directly go to the wished page, without being forced to pass all the previous ones. It is then available a [Numeric field](#) to directly select the wished page.



Input points for 33M			
Offs	Pos.	Port	Help
1	1.1		X311 Pin 1
2	1.2		X311 Pin 2
3	1.3	DIM_1	X311 Pin 3
4	1.4	DIM_2	X311 Pin 4
5	1.5	DIM_3	X312 Pin 1
6	1.6	DIM_4	X312 Pin 2
7	1.7	DIM_5	X312 Pin 3
8	1.8		X312 Pin 4

PAG 1 /1

Buttons at the bottom: Modify, Clean, Direction, Prev. page, Next page, Back.

This table columns have the following meaning:

- **Offs** - sequential index of the Device I/O points
- **Pos** - byte and bit of the Device
- **Port** - indicates the PDL2 port associated to the I/O point
- **Help** - a Help string describing the I/O point.

Each Device has its own distribution of both the status bits and I/O images. It is then mandatory to know such a distribution, in order to properly map the I/O points.



Note that Input and Output points are in two different arrays, so the offsets of these and those are referred to their own array.

Here follow the status bits layout and the I/O images offsets, for the following Devices:

- [X20PS2100 - Power supply 24V](#)
- [X20DI9371 - 12 Digital Input module](#)
- [X20DO6322 - 6 Digital Output module](#)
- [X20AI2622 - 2 Analog Input module](#)
- [X20AO2622 - 2 Analog Output module](#)
- [X20CM1941 - Resolver module](#)
- [X20DC1198 - Interface module SSI](#)
- [X20DS1828 - Hyperface module](#)



In the next sections, reference is made to the I/O points Tables.

- **X20PS2100 - Power supply 24V**

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1 [^] byte	IN		1
StatusInput01	1 [^] Input bit	2 [^] byte	IN	2.1	2
StatusInput02	2 [^] Input bit		IN	2.2	2

- **X20DI9371 - 12 Digital Input module**

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1 [^] byte	IN		1
DigitalInput01	1 [^] Input bit	2 [^] byte	IN	2.1	2
...	...		IN	...	2
DigitalInput08	8 [^] Input bit		IN	2.8	2
DigitalInput09	9 [^] Input bit	3 [^] byte	IN	3.1	3
...	...		IN	...	3
DigitalInput12	12 [^] Input bit		IN	3.4	3

- **X20DO6322 - 6 Digital Output module**

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1 [^] byte <i>(Input)</i>	IN		1 (<i>Input</i>)
DigitalOutput01	1 [^] Output bit	1 [^] byte <i>(Output)</i>	OUT	1.1	1 (<i>Output</i>)
...	...		OUT	...	1 (<i>Output</i>)
DigitalOutput06	6 [^] Output bit		OUT	1.6	1 (<i>Output</i>)

StatusDigitalOutput01	1^ Input bit	2^ byte (<i>Input</i>)	IN	2.1	2 (<i>Input</i>)
...	...		IN	...	2 (<i>Input</i>)
StatusDigitalOutput06	6^ Input bit		IN	2.6	2 (<i>Input</i>)

- X20AI2622 - 2 Analog Input module

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1^ byte	IN		1
AnalogInput01	1^ Analog Input	2^ & 3^ bytes	IN	2.1	2
AnalogInput02	2^ Analog Input	4^ & 5^ bytes	IN	4.1	4
StatusInput01	1^ Input bit	6^ byte	IN	6.1	6

- X20AO2622 - 2 Analog Output module

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1^ byte (<i>Input</i>)	IN		1 (<i>Input</i>)
AnalogInput01	1^ Analog Input	1^ & 2^ bytes (<i>Output</i>)	OUT	1.1	1 (<i>Output</i>)
AnalogInput02	2^ Analog Input	3^ & 4^ bytes (<i>Output</i>)	OUT	3.1	3 (<i>Output</i>)
StatusInput01	1^ Input bit	2^ byte (<i>Input</i>)	IN	2.2	2 (<i>Input</i>)

- X20CM1941 - Resolver module

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1^ byte	IN		1
Position		2^, 3^, 4^ e 5^ byte	IN	2.1	2
StatusInput		6^ byte	IN	6.1	6

- X20DC1198 - Interface module SSI

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1^ byte	IN		1
PowerSupply01		2^ byte	IN	2.1	2
PowerSupply02		2^ byte	IN	2.2	2
DigitalInput01	1^ Input bit	3^ byte	IN	3.1	3
DigitalInput02	2^ Input bit		IN	3.2	3
Encoder01		4^, 5^, 6^ & 7^ bytes	IN	4.1	4

- X20DS1828 - Hyperface module

Name	Description	Byte	Direction	Pos.	Page to start
ModuleOK	Status byte (do not map)	1 [^] byte	IN		1
PosTime		2 [^] , 3 [^] , 4 [^] e 5 [^] byte	IN	2.1	2
PositionHLW		6 [^] , 7 [^] , 8 [^] e 9 [^] byte	IN	6.1	6
PositionLW		10 [^] , 11 [^] , 12 [^] e 13 [^] byte	IN	10.1	10

The involved subpage is provided with a Functional keys menu including the following commands:

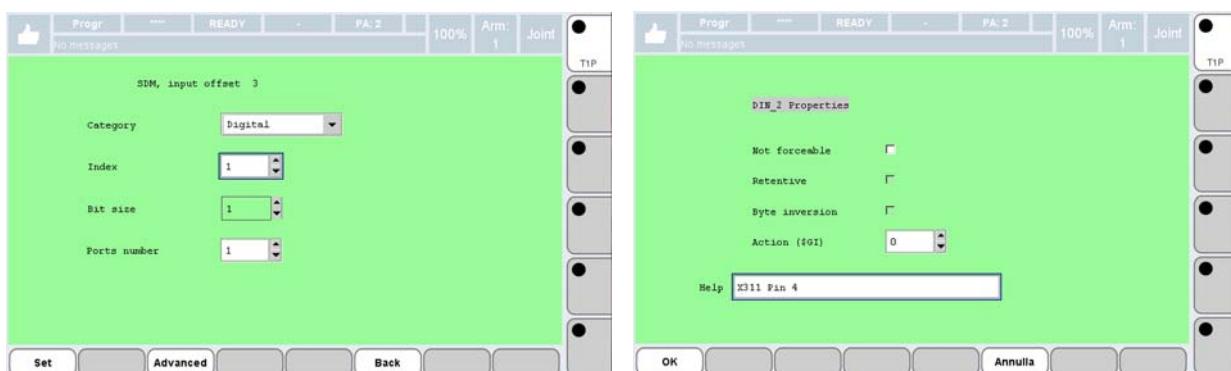
- Map or Modify - I/O points
- Clear - I/O points
- Direction - I/O points

In the case in which the table includes more than one page, there are two more enabled softkeys, allowing to visit all the pages:

- Prev.page and Next page - I/O points.

11.4.2.1.1 Map or Modify - I/O points

To operate on a certain I/O point, (mapping a not yet mapped point, modifying an already mapped one), the User must select the table row containing it, touch **Map/Modify** key and perform the wished actions, in the suitable screen page. The example below are referred to the previous table of SDM device.



a - Input point to be mapped

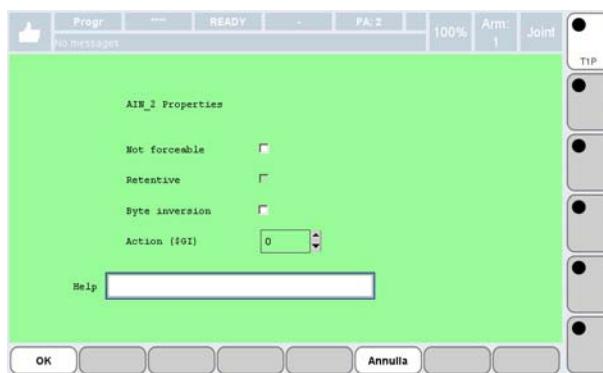
b - already mapped Input point

If the selected point is to be mapped, all fields, except **Bit Size** field, are editable. On the contrary, if the selected point is already mapped, the only editable fields are **Unforceable** flag (both for Input and Output points) and **Retentive** flag (for Output points only). All the other fields are read-only.

The displayed fields are as follows:

- **Category** - 1 Digital, 2 Analog, 3 Flex
- **Index** - index of the PDL2 port (e.g. if the port is \$DIN[3], the index is 3)

- **Bit Size** - dimension of the involved PDL2 port (total amount of bits). Available for Flex ports only.
- **Ports number** - total amount of ports, of the specified type, involved in the operation
- **Unforceable** - it indicates whether the involved I/O points are NOT subject to the **Force** command
- **Retentive** - if the being mapped ports are Output ports, the User can specify that, after a powerup, they keep the status they had before the Controller shut-down. To do that, this flag must be selected.
- **Swap byte** - in case of analog I/O, a checkbox is available stating whether or not exchanging the bytes is required, when the device data format does not comply the one expected by the Controller.



- **Help** - help string which should “help” the User to better understand the meaning of the associated I/O points.

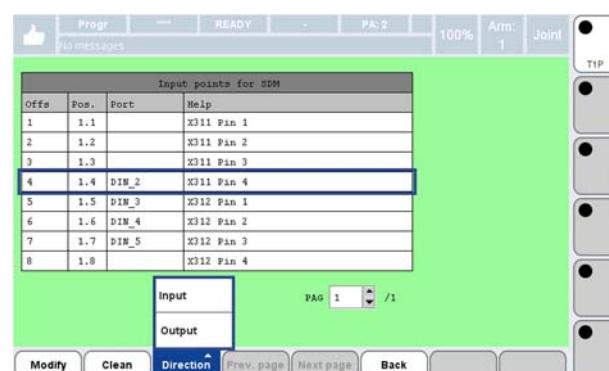
When all the information has been inserted, touch either **OK** key to confirm or **Cancel** key to quit without saving.

11.4.2.1.2 Clear - I/O points

If the currently selected I/O point is already mapped, it allows to clear its mapping.

11.4.2.1.3 Direction - I/O points

It indicates whether to operate on either Input points or Output points: pressing this key, a menu is opened to choose the wished item.

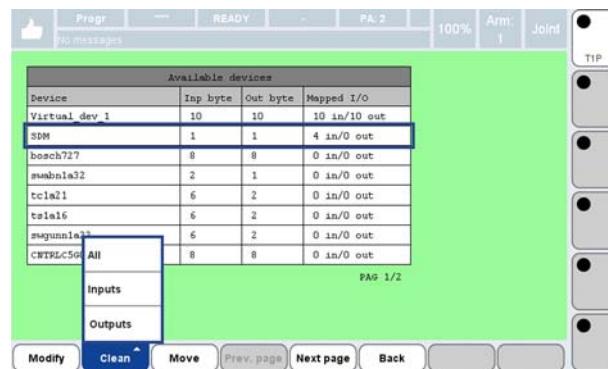


11.4.2.1.4 Prev.page and Next page - I/O points

In the case in which the table includes more than one page, there are two more enabled keys, allowing to visit all the pages. Please refer to [par. 11.4.2.4 Prev.page, Next page on page 472](#) for a detailed description.

11.4.2.2 Clean - Device

Fig. 11.33- Clean - Device

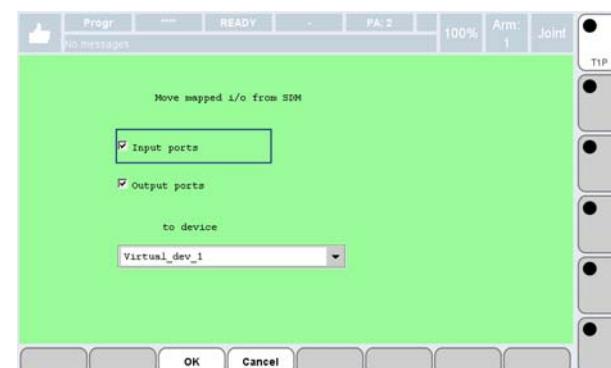


It allows to clear the mapping of the I/O points belonging to the currently selected Device. A menu is displayed when this functionality is activated, allowing to choose whether to clear

- **All** - all the I/O points belonging to the Device
- **Inputs** - the Input points only
- **Outputs** - the Output points only.

11.4.2.3 Move - Device

This functionality allows moving the mapped I/O points, from one Device to another. It is useful when the target Device is not exactly known yet: in such a situation it is possible mapping the wished I/O points to a Virtual Device (see [par. 11.1.2.2 Virtual on page 435](#)) and then moving them later to the target Device.



Touching **Move** key, the shown above page is displayed in which the User is requested to provide the following information:

- **Input Ports / Output Ports** - specifies whether the Input Ports and/or the Output Ports are moved to the target Device

- **to device** - is the name of the Device which the currently selected Device is to be moved to (in the previous figure example, the being moved Device is **SDM**). To select the target Device, touch the **Combobox**, choose the wished Device and press **OK** to confirm.



Once the choice has been made, confirm moving by touching **OK** key.

11.4.2.4 Prev.page, Next page

It allows visiting all the table pages. A short pressure causes to go to the **Prev./Next** page in the table; a long pressure causes to go, respectively, to the **first** page (**Prev.**) and to the **last** page (**Next**) in the table.

11.4.2.5 USER Devices

It allows accessing virtual devices (if existing) prearranged for the installed application. In the displayed page, select the wished virtual device.



Available devices			
Device	Inp byte	Out byte	Mapped I/O
Virtual dev_1	10	10	10 in/10 out
USER_APP_DEV	8	8	8 in/8 out
SDM	1	1	0 in/0 out
bosch727	8	8	0 in/0 out
swabn1a32	2	1	0 in/0 out
tc1a21	6	2	0 in/0 out
ts1a16	6	2	0 in/0 out
swgunn1a22	6	2	0 in/0 out

PAG 1/2

Buttons at the bottom: Modify, Clean, Move, Prev. page, Next page, Back.



Note that the corresponding real user device MUST be already mapped!

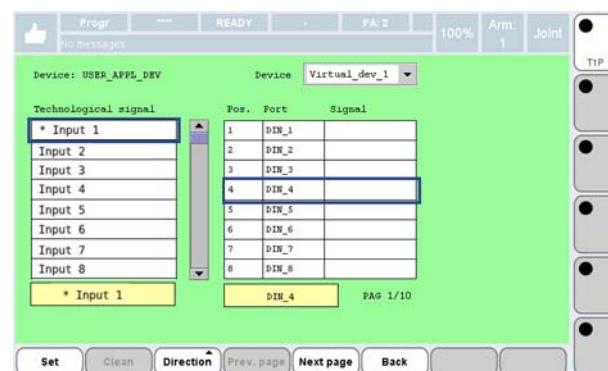
Touching **Modify** key, the program displays the following page in which the user should associate technological signals to ports.

The technological signals displayed with a '*' symbol, are MANDATORY, so it is NEEDED for each one of them to be associated to a port.

Select a technological signal in the left column. In the shown example, the technological signal "Input 1" is selected.



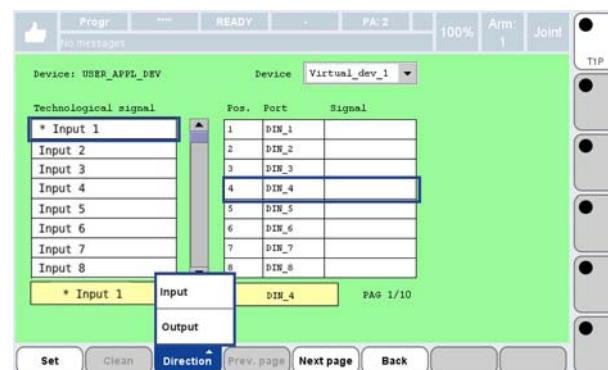
Select the wished port to be associated to the technological signal, in the right column e.g. Digital Input 4.



Touch **Set** key, to associate the selected technological signal to the selected port. Example: "Input 1" technological signal associated to Digital Input 4 port.

In case in which the user wishes to modify an already set association, press **Clean** key.

Use **Direction** key to choose between Input and Output environment.



When associating the wished technological signals and the corresponding ports is completed, touch **Back** key to exit from the subpage.

At the end, touch **Back** key again to exit from this subpage.

If even just one mandatory technological signal has NOT been associated to any ports, a warning message is issued.

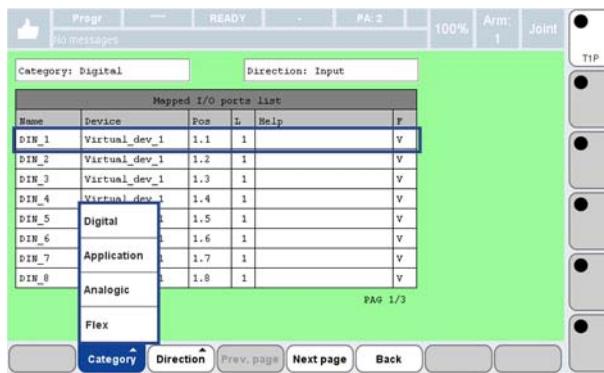
If, anyway, the user wishes to quit the subpage, touch **OK** key. Otherwise, choose **Cancel** and set the required association. Touch **Back** key to exit.

11.4.3 Port list

This command allows displaying the [list of all the existing ports in the System](#).



Note that this is merely a view functionality. As already told before, if the User wishes to modify one or more points belonging to one or more Devices, it is needed to use the [Devices](#) command.



The screenshot shows a software interface with a title bar "Program" and "READY". Below the title bar, there are status indicators: "PA:2", "100%", "Arm: 1", and "Joint". On the right side, there are several circular buttons labeled "TIP". The main area displays a table titled "Mapped I/O ports list". The table has columns: "Name", "Device", "Pos", "L", "Help", and "F". The rows show port entries such as DIN_1 through DIN_8, each associated with a device like "Virtual_dev_1" and a position like "1.1". The table also includes sections for "Digital", "Application", "Analog", and "Flex". At the bottom of the table, it says "PAG 1/3". Below the table, there is a menu bar with buttons: "Category", "Direction", "Prev. page", "Next page", and "Back".

As shown in the above figure, a table is displayed containing the following information:

- **Name** - is the port name
- **Device** - is the Device the port is associated to
- **Pos** - is the position of the I/O point of such a Device
- **L** - port length (in bits)
- **Help** - is the help string, describing the port meaning
- **F** - it is a flag indicating whether the port is Retentive (R) and/or Unforceable (N) or Virtual (V)

The available commands in the bottom menu are as follows:

- [Category](#) - Port list
- [Direction](#) - Port list
- [Prev.page](#) and [Next page](#) - Port list.

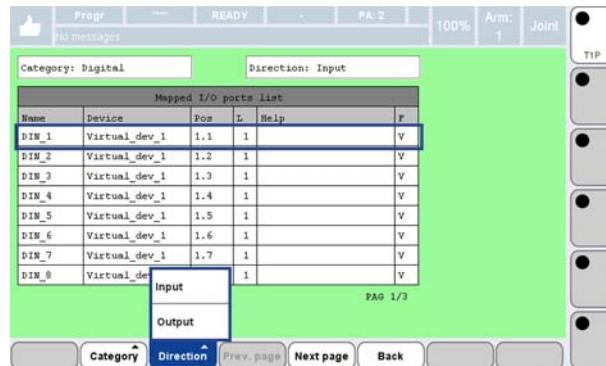
11.4.3.1 Category - Port list

Touching this key a menu is opened (see previous figure) to choose the port type:

- 1 Digital - \$DIN, \$DOUT
- 2 Application - \$IN, \$OUT
- 3 Analog - \$AIN, \$AOUT
- 4 Flex - \$FMI, \$FMO.

11.4.3.2 Direction - Port list

Touching this key a menu is opened to choose either Input or Output.



11.4.3.3 Prev.page and Next page - Port list

This command allows visiting all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.4 Profile

This command allows to associate a System/Application action or event to a **digital, analog, flexible** PDL2 language logical port (e.g. \$DIN, \$DOUT, etc.)

To enter this sub-environment, press **Profile** key: a menu is displayed to choose whether the action/event to be associated to an I/O point, is related to the **System** or to an **Application**.

The **Application** profile is only available if at least one COMAU **Smart***(es. **SmartSpot**, **SmartHand**, etc.) application is currently installed on the Controller Unit.



System and Application actions/events can be associated, as global Profile, to the Slave Fieldbus module which has been declared as System Device.

In the case of System Profile, while configuring such a module, the System Profile field must have been selected - see [Fig. 11.4 - Profibus Slave module Configuration on page 427](#).

System actions/events can be associated, as a Single Port, to any I/O point.



Select the wished item touching it.

- [System Profile](#)
- [Application Profile](#).

11.4.4.1 System Profile

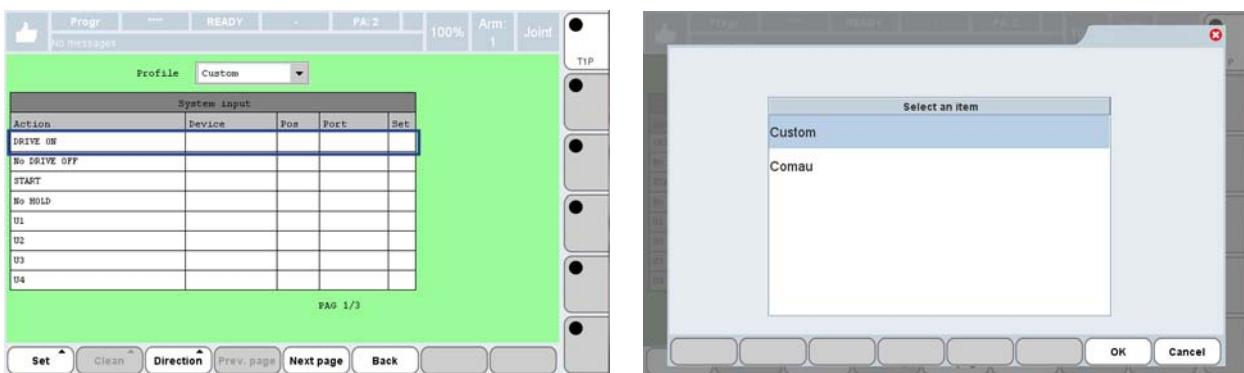
The displayed table in this environment, lists the following items:

- **actions/events** to be associated,
- **device** name,
- **position** (byte.bit) and name of the action/event associated port,
- indication of whether associated or not.



Each indicated Action/Event in the System Profile, has got the corresponding image into some \$GI e \$GO type ports: \$GI17..\$GI26 and \$GO17..\$GO32.

For any further information about them, mainly for Actions/Events meaning, refer to Ports to check the start and stop statuses table, chap.8 - Start and stop controls in C5G Control Unit - Technical Specification manual.



In the **System Profile** page, a combobox (**Profile**) is provided in order to select the wished profile, choosing it from a list which is automatically opened; the available ones are:

- [System Profile - Comau](#) - predefined profile with suggested default ports
- [System Profile - Custom](#) - profile which the User is allowed to customize.

11.4.4.1.1 System Profile - Comau

After selecting **Comau** item, touch **OK** key to confirm. The already existing associations are displayed.



For Comau predefined profile, the default values suggested by the program are viewed in the table. This DOES NOT mean that such values have already been automatically

associated: it is just a suggestion and the association must explicitly be required by the user, touching **Set** key.

Please note that the suggested PDL2 ports are the previously mapped ones (see [par. 11.4.2.1 Modify - Device on page 466](#)) on the Device indicated in the table.

The following commands are available in the Functional keys menu:

- [Set - System Profile - Comau](#)
- [Clean - System Profile - Comau](#)
- [Direction - System Profile - Comau](#)
- [Prev.page and Next page - System Profile - Comau.](#)

Set - System Profile - Comau

This command associates actions/events to the specified ports. The User cannot modify them, except clearing one or more of them ([Clean - System Profile - Comau](#)).



Choosing this command, a menu is open which allows selecting the following items:

- **Profile** - to associate all the chosen profile ports (Comau, in our example) to all the actions/events.



In **Set** column of each row, the word **Yes** appears to indicate the performed association.

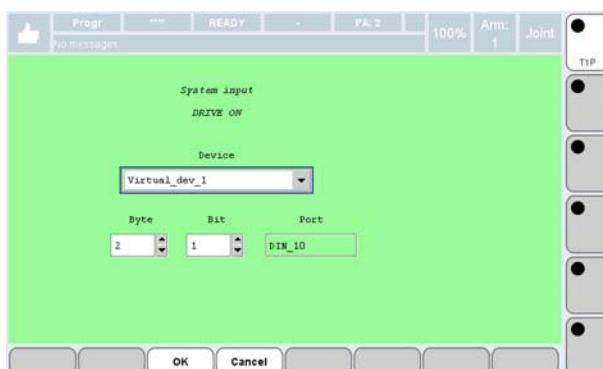
If one or more points of the device are not mapped, a message is displayed. The User must take care of mapping them before issuing command **Set** again.

At the end of the Set operation, an informational message appears to state the command has been executed, and the required associations are displayed.

- **Single port** - to associate the selected action/event to a single port.



The program asks the User to specify which is the **Device-Byte-Bit** the selected action/event is to be associated to. Depending on such 3 parameters, the corresponding PDL2 logical port is displayed, if mapped. Touch **OK** key to confirm.



If in the required position there is not a mapped port, the **OK** key is not available, so the User must Cancel the current operation and map it, before issuing the **Set** command again.

At the end of the operation, the word **Yes** is displayed in the table to indicate the action/event has been associated to the required port.



Note that, setting a single port, the current Profile is automatically switched to [System Profile - Custom](#).

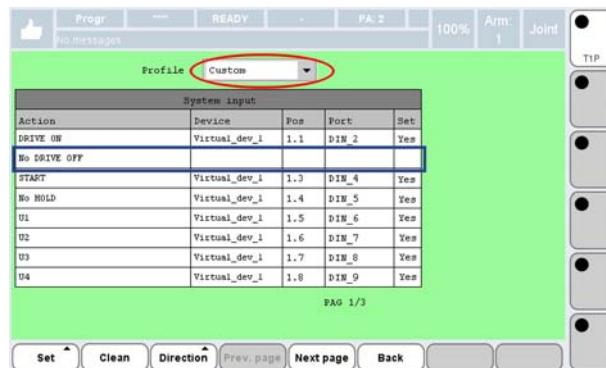
Clean - System Profile - Comau

This command allows to delete a previously set association. This means the currently selected association is cleared.

In the below example, the association between No DRIVE OFF action and its corresponding port has been deleted.



Note that, moreover, when modifying the Comau System Profile, the current Profile is automatically switched to [System Profile - Custom](#).



Direction - System Profile - Comau

This command allows to choose to operate on either Input or Output. Touch the wished item to select it.



Prev.page and Next page - System Profile - Comau

This command allows to visit all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.4.1.2 System Profile - Custom

After selecting **Custom** item, press **ENTER** to confirm. Already existing associations are displayed.



The following commands are available in the Functional keys menu:

- Set - System Profile - Custom
- Clean - System Profile - Custom
- Direction - System Profile - Custom
- Prev.page and Next page - System Profile - Custom.

Set - System Profile - Custom

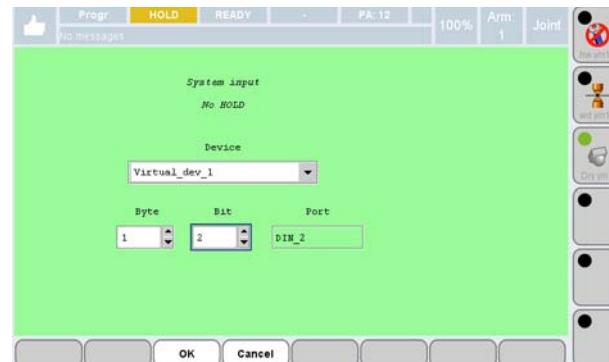


Note that, if the choice is Custom, the only possibility for associating actions/events to I/O points, is Single port.

A menu is opened to associate the selected action/event to a **Single port**.



The program asks the User to specify which is the **Device-Byte-Bit** the selected action/event is to be associated to. Depending on such 3 parameters, the corresponding PDL2 logical port is displayed, if mapped. Touch **OK** key to confirm.



If in the required position there is not a mapped port, the **OK** key is not available, so the User must Cancel the current operation and map it, before issuing the **Set** command again.

At the end of the operation, the word **Yes** is displayed in the table, **Set** column, to indicate the action/event has been associated to the required port.



Clean - System Profile - Custom

This command allows deleting a previously set association. To use it, please refer to the description of [par. Clean - System Profile - Comau on page 478](#).

Direction - System Profile - Custom

This command allows to choose operating on either Input or Output points. Select the wished item and press **ENTER** to confirm. To use this command, please refer to the description of [par. Direction - System Profile - Comau on page 479](#).

Prev.page and Next page - System Profile - Custom

This command allows to visit all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.4.2 Application Profile



Please remember that Application actions/events must only be associated to Fieldbus Slave points.



The table displayed in this sub-environment, lists the following information:

- **actions/events** to be associated,
- **device name**,

- **position** (byte.bit) and name of the port corresponding to the action/event,
- **port length (L)** in bit (Application programs can also use \$FMI and \$FMO ports, whose length is not predefined)
- **Set** - indication of whether associated or not.

In the **Application** Profile page (see figure above), a combobox (**Profile**) is provided in order to select the wished profile; the available ones are:

- **Application Profile - Comau** - predefined profile with suggested default ports
- **Application Profile - Custom** - profile which the User is allowed to customize.



11.4.4.2.1 Application Profile - Comau

After selecting **Comau** item, press **ENTER** to confirm. The already existing associations are displayed.



Action	Device	Pos	L	Port	Cfg
Application fault reset	PROFISLAVE	1.1	1	DIM_40	
Dry cycle command	PROFISLAVE	1.2	1	Not mapped	
Application exclusion command	PROFISLAVE	1.6	1	Not mapped	
Process YES/NO command	PROFISLAVE	1.4	1	DIM_43	

The default values for the **Comau** predefined profile, suggested by the program, are displayed in the table. This DOES NOT mean such values have already been automatically associated to the actions/events: they are just a suggestion and, if wished, the User must intentionally touch **Set** key to associate them.

The following functionalities are available in the bottom menu:

- **Set - Application Profile - Comau**
- **Clean - Application Profile - Comau**
- **Direction - Application Profile - Comau**
- **Prev.page . and Next page - Application Profile - Comau.**

Set - Application Profile - Comau

This command associates actions/events to the specified ports. The User cannot modify

them, except clearing one or more of them ([Clean - Application Profile - Comau](#)).



Choosing this command causes a menu to be opened to select the following items:

- **Profile** - to associate all the chosen profile ports (Comau, in our example) to all the actions/events. In column **Cfg** of each row, the word **Yes** appears to indicate the performed association.



If one or more points of the device are not mapped, a message is displayed (see figure below). The User must take care of mapping them before issuing command **Set** again.

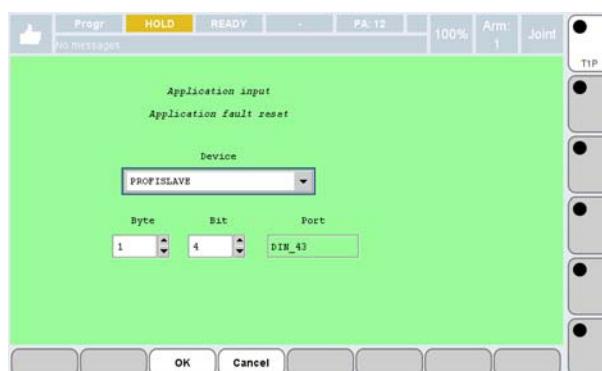


At the end of the **Set** operation, an informational message occurs to state the command has been executed, and the required associations are displayed.

- **Single port** - to associate the selected action/event to a single port.



The program asks the User to specify which is the **Device-Byte-Bit** the selected action/event is to be associated to. Depending on such 3 parameters, the corresponding PDL2 logical port is displayed, if mapped. Touch **OK** key to confirm.



If in the required position there is not a mapped port, the **OK** key is not available, the User must Cancel the current operation and map it, before issuing **Set** command again.

At the end of the operation, the word **Yes** is displayed in the table (**Cfg** column) to indicate the action/event has been associated to the required port.



Note that, setting a single port, the current Profile is automatically switched to [Application Profile - Custom](#).

Clean - Application Profile - Comau

This command allows deleting a previously set association. This means the currently selected association is cleared.

In the example shown in the following figure, the association between **Application fault reset** action and its corresponding port has been deleted.



Note that, moreover, when modifying the Comau Application Profile, the current Profile is automatically switched to [Application Profile - Custom](#).



Direction - Application Profile - Comau

This command allows to choose to operate on either Input or Output. Touch the wished item to select it.

Fig. 11.34- Direction



Prev.page . and Next page - Application Profile - Comau

This command allows to visit all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.4.2.2 Application Profile - Custom

After selecting **Custom** item, press **ENTER** to confirm. Already existing associations are displayed.



The following commands are available in the Functional keys menu :

- Set - Application Profile - Custom
- Clean - Application Profile - Custom
- Direction - Application Profile - Custom
- Prev.page and Next page - Application Profile - Custom.

Set - Application Profile - Custom



Note that, if the choice is Custom, the only possibility for associating actions/events to I/O points, is Single port.

A menu is opened to associate the selected action/event to a **Single port**.



The program asks the User to specify which is the **Device-Byte-Bit** the selected action/event is to be associated to. Depending on such 3 parameters, the corresponding PDL2 logical port is displayed, if mapped. Touch **OK** key to confirm.



If in the required position there is not a mapped port, the **OK** key is not available, the User must Cancel the current operation and map it, before issuing **Set** command again.

At the end of the operation, the word **Yes** is displayed in the table (**Cfg** column) to indicate the action/event has been associated to the required port.

Clean - Application Profile - Custom

This command allows deleting a previously set association. To use it, please refer to the description of par. **Clean - System Profile - Comau** on page 478.

Direction - Application Profile - Custom

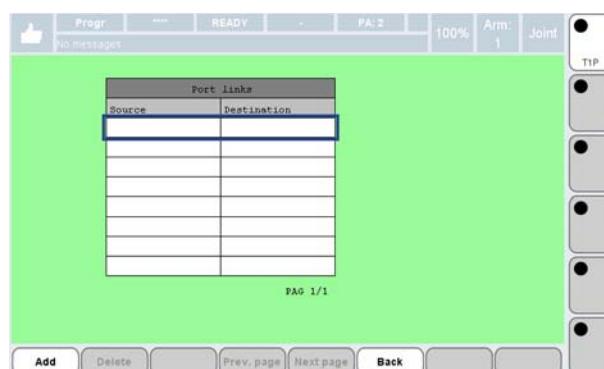
This command allows to choose operating on either Input or Output points. Select the wished item and press **ENTER** to confirm. To use this command, please refer to the description of par. [Direction - System Profile - Comau on page 479](#).

Prev.page and Next page - Application Profile - Custom

This command allows to visit all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.5 Link

This command allows to create an association (link) between two ports: the second Port (called *destination*) takes the same value of the first one (called *source*). It is, therefore, a logical copy.



The shown above table, includes the following information:

- **Source** - it is the source Port; it can be either an Input or an Output Port
- **Destination** - it is the destination Port; it must be an Output Port, except the situation of Ports which are mapped onto a [Virtual](#) network: in such a case it can be either an Input or an Output Port.

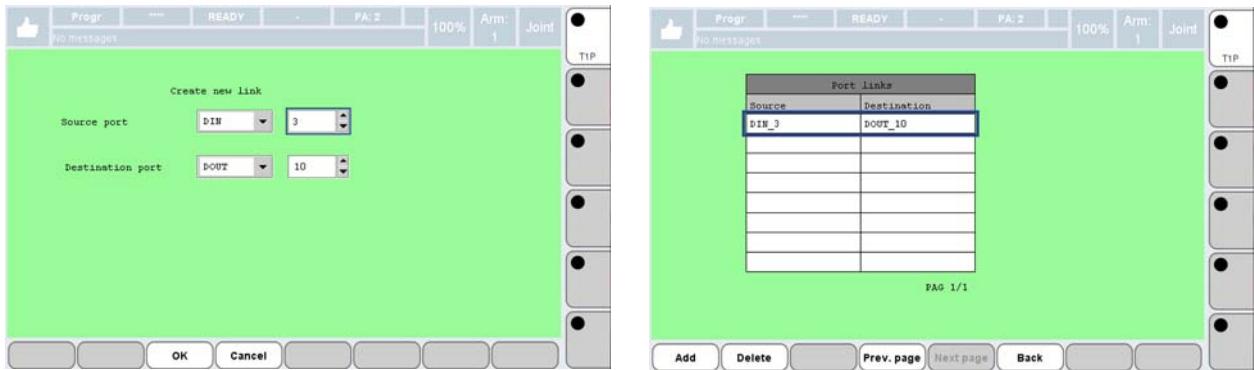
The available commands in the Functional keys menu allow to operate on the links, to add some more or to remove already existing ones. They are:

- [Add - Link](#)
- [Delete - Link](#)
- [Prev.page and Next page - Link](#).

11.4.5.1 Add - Link

This command creates a new association (link) between two ports

In the following example (see the two figures below), **Digital Output 10** is associated to **Digital Input 3**. When all the required data have been inserted in the corresponding fields, touching **OK** key in the left screen page below, causes the association between the two specified Ports to take effect. This is shown in the right figure below.



If it is wished to exit from **Add** functionality, without operating any modifications, just touch **Cancel** key.

11.4.5.2 Delete - Link

This command deletes the selected link between two Ports.

11.4.5.3 Prev.page and Next page - Link

This command allows to visit all the table pages. For a full description, see [par. 11.4.2.4 Prev.page, Next page on page 472](#).

11.4.6 Save

This command saves all the performed modifications, without asking for any further confirmation. An informational message tells the User that data saving has been executed.

NO further commands are needed to make the modifications operational.

11.4.7 Exit

This command allows quitting **IO_MAP** program.



The performed modifications, if not previously saved by means of **Save** command, are lost.

11.5 IO_CLONE Program - I/O Configuration Export/Import

11.5.1 Activation of IO_CLONE

This program, **implemented in Visual PDL2 language**, is used to “copy” the I/Os configuration and mapping, from a C5G Controller to another one.

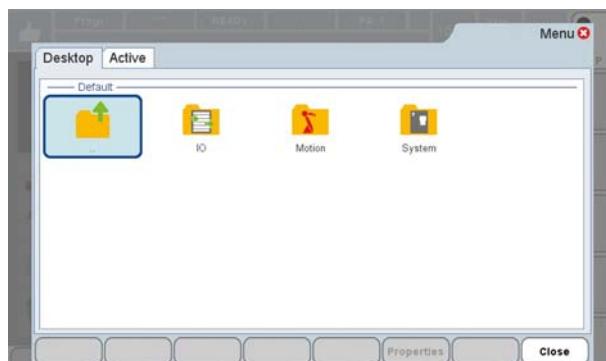
It is extremely useful for robotized production lines, because the User can configure and map the I/O points on a C5G Controller only and then copy such a mapping to all the other C5G Controllers belonging to the robotized line.

To carry out the above described goal, it is needed to:

- a. perform I/Os **configuration** and **mapping** on just ONE C5G Controller, by means of [IO_CNFQ Program - I/O modules configuration](#) and [IO_MAP Program - I/O ports mapping](#);
- b. **export** I/Os mapping and Slave configuration settings, by means of **IO_CLONE** program; a **.CIO** file is created including all the required information;
- c. **import** all data included in such a file, to all the C5G Controllers belonging to the robotized line;
- d. **save** the imported data.

The **IO_CLONE** program is always present and available on the Teach Pendant.

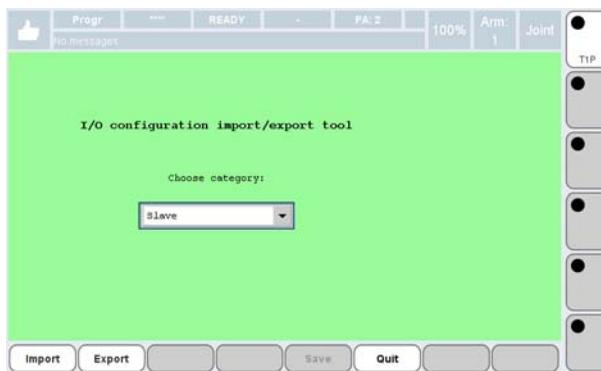
- a. access the [Setup page](#) on the Teach Pendant
- b. touch **IO** icon



- c. touch **IO_CLONE** icon



d. The system runs **IO_CLONE** program



In the **IO_CLONE** program, the following functionalities are available to allow the above described functions:

- [Export](#)
- [Import](#)
- [Save](#)
- [Quit.](#)



[Back](#)

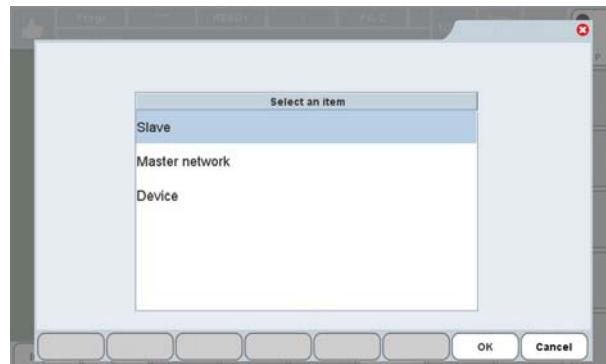
Note that in the screen pages of the other levels, [Back](#) key is always present to allow quitting such a step of the procedure, and going back to the previous one.

Depending on the **category** for which “cloning” is wished, the listed above operations can be activated for (see [Fig. 11.35](#)) for:

- **Slave**
- **Master network**
- (single) **Device**.



NOTE - Before issuing any command, it is needed to choose the wished category, by selecting it in the list activated when touching the **Choose category** combobox.

Fig. 11.35- Choosing the Category

After choosing the **category**, go ahead with the wished operations, as described in the following sections.

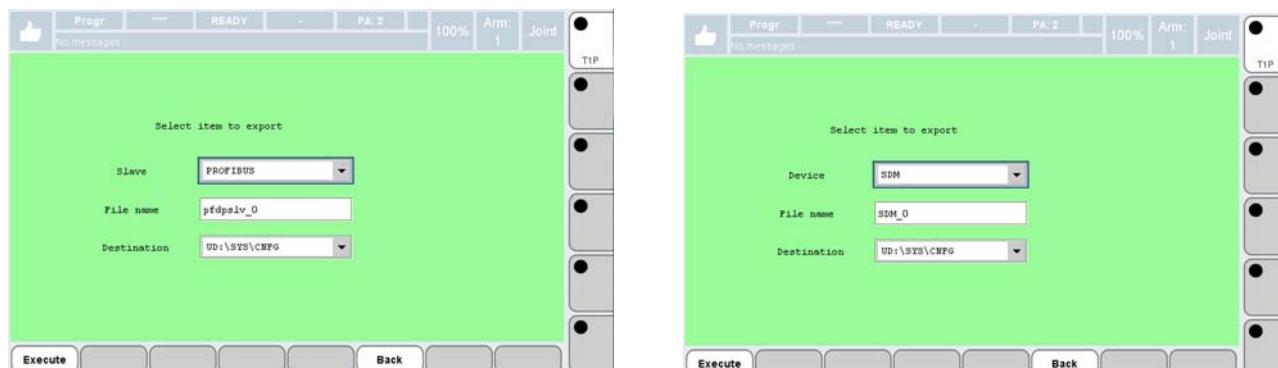
11.5.2 Export

This command allows creating a **.CIO** file containing the I/O points mapping together with some configuration information, for the currently selected category.

Such a file will then be importable, to be copied onto other C5G Controllers.

When this command is issued, the program displays a page in which the User must specify the following information:

- **Slave / Master / Device** - each one of these three categories is associated to a different page; open the corresponding combobox, operate the wished choice and touch **OK** to confirm. In the figures below, Slave and Device pages are shown.



NOTE - for MASTER NETWORK and DEVICE categories, I/O points mapping only is exported. No configuration information. The Export operation for DEVICE category is similar to the MASTER NETWORK one, but referred to a single device.

- **File name** - name of the being created **.CIO** file; it will include the I/O points mapping and some configuration information.
- **Destination** - directory path for **.CIO** file: device and directory. It is suggested by **IO_CLONE** program; when a USB flash disk is inserted, **XD:** and **TX:** type devices are also suggested. **TX:** device is suggested just if and when the USB flash disk is inserted into the Teach Pendant USB port.



When the chosen category is SLAVE, exporting will be successful only if the expected Slave type is present on the destination.

As soon as the required data insertion is completed, press **Execute** to start exporting.

At the end of the operation, the program displays a suitable message to inform about the Export operation completion.



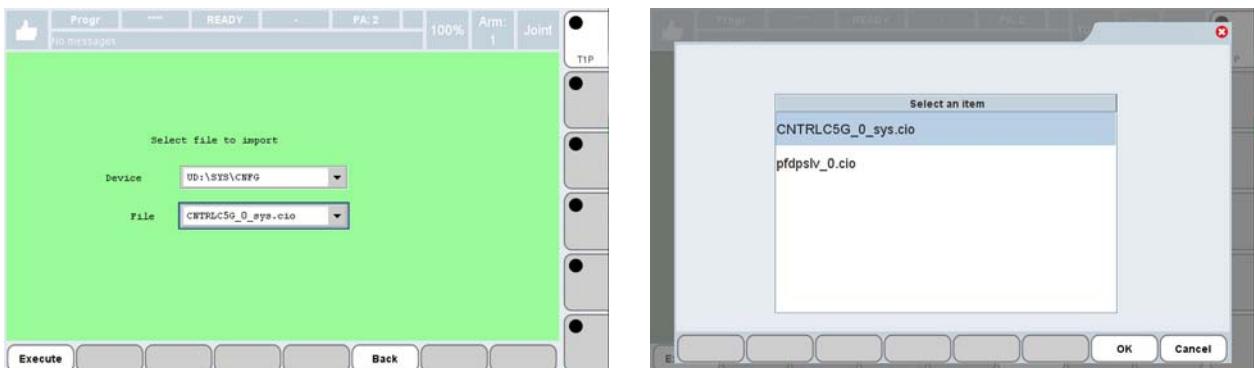
Touch **OK** to go back to the main page.

11.5.3 Import

This command allows “copying”, onto the specified device, a previously exported I/O points mapping (see [par. 11.5.2 Export on page 491](#)).

The program displays a page in which the User must specify the following information:

- **Device** - directory path for **.CIO** file: device and directory. It is suggested by **IO_CLONE** program; when a USB flash disk is inserted, **XD:** and **TX:** type devices are also suggested. **TX:** device is suggested just if and when the USB flash disk is inserted into the Teach Pendant USB port
- **File** - name of the previously created **.CIO** file, including some configuration information and mapping for all the I/O points. When more than one file of such a type are available, the User must choose the wished one opening the corresponding combobox, selecting it and touching **OK** to confirm.



As soon as the required data have been inserted, press **Execute** to perform Import operation.



WARNING - if the chosen category is 'Slave', a page is displayed in which the User must specify the address of the destination Slave, by means of the displayed Numeric field. When modified, touch Set to confirm it.

At the end of the operation, the program displays a suitable message to inform about the Import operation completion.

Touch **OK** to go back to **IO_CLONE** home page.

On the contrary, if any error occurs, a suitable error message is issued to notify the User.



WARNING - importing can be successful ONLY IF the destination device has ALREADY been CONFIGURED!

At the end of the Import operation, the User must save the new configuration ([Save](#) key) and perform a **restart cold** operation (see [par. 5.7.2.2.1 Cold on page 91](#)).

11.5.4 Save

This command must be used to save all modifications made by an Import operation (see [par. 11.5.3 Import on page 492](#)).

11.5.5 Quit

This command allows quitting **IO_CLONE** program.



Any imported information will be lost if not previously saved by means of [Save](#) command.

12. APPENDIX - SOFTWARE OPTIONS

A full list of the available Software Options, both for C5G Control Unit and for Personal Computer, is provided in the following tables:

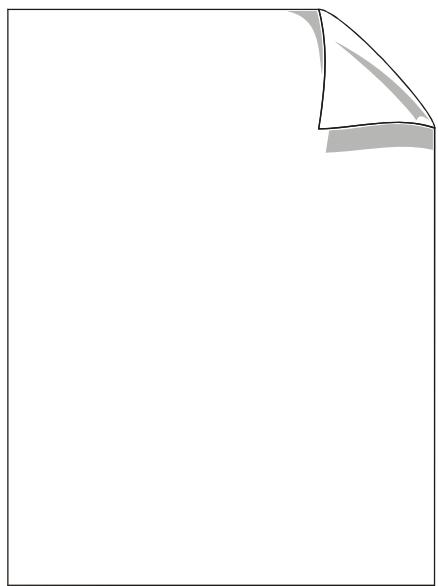
- [Tab. 12.1 - C5G Software Options](#) on page 494
- [Tab. 12.2 - Personal Computer Software Options](#) on page 495

Tab. 12.1 - C5G Software Options

Part Code	Description	SW release Version
MOTION		
CR17926200	Cooperative Motion	0.70.200
CR17926201	Collision Detection	0.6x
CR17926202	Automatic Payload Identification	0.6x
CR17926203	Joint Soft Servo	0.6x
CR17926204	Synchronized Arms	0.70.200
CR17926205	Sensor Tracking	0.70.200
CR17926206	Conveyor Tracking	t.b.d.
CR17926207	Weaving Motion	0.70.200
CR17926208	Robot Absolute Accuracy	0.6x
CR17926209	Speed Control for Arm	0.6x
CR17926210	SmartMove	0.6x
CR17926211	Path Governor	0.6x
CR17926212	Multipass	0.6x
CR17926213	Manual Guidance	t.b.d.
CR17926214	Palletizing Motion	0.6x
CR17926215	Ceiling mounted robot	0.6x
CR17926216	Interference Regions	0.6x
CR17926218	Advanced Interference Regions	1.15
CR17926217	Axes Pursuit	0.70.200
CR17926219	Low Resolution Euler Angles	1.15
COMMUNICATION		
CR17926300	PDL2 Read/Write on TCP/IP	0.6x
USER INTERFACE		
CR17926400	VP2.Builder	0.6x
CR17926401	VP2.Frames	0.6x

Tab. 12.2 - Personal Computer Software Options

Part Code	Description	SW release Version
CR17926100	WinC5G Full Edition	0.70.200





Comau in the World

**COMAU S.p.A.
Headquarters**
Via Rivalta, 30
10095 Grugliasco - TO (Italy)
Tel. +39-011-0049111

Powertrain Machining & Assembly
Via Rivalta, 30-49
10095 Grugliasco - TO (Italy)
Tel. +39-011-0049111
Telefax +39-011-0049688

Body Welding & Assembly
Strada Borgarett, 22
10092 Borgarett di Beinasco - TO (Italy)
Tel. +39-011-0049111
Telefax +39-011-0048672

Robotics & Service
Via Rivalta, 30
10095 Grugliasco - TO (Italy)
Tel. +39-011-0049111
Telefax +39-011-0049866

Engineering, Injection Moulds & Dies
Via Bistagno, 10
10136 Torino (Italy)
Tel. +39-011-0051711
Telefax +39-011-0051882

Comau France S.A.
5-7, rue Albert Einstein
78197 Trappes Cedex (France)
Tel. +33-1-30166100
Telefax +33-1-30166209

Comau Estil
10, Midland Road
Luton, Bedfordshire LU2 0HR (UK)
Tel. +44-1582-817600
Telefax +44-1582-817700

Comau Deutschland GmbH
Monzastrasse 4D
D-63225 Langen (Germany)
Tel. +49-6103-31035-0
Telefax +49-6103-31035-29

German Intec GmbH & Co. KG
Im Riedgrund 1
74078 Heilbronn (Germany)
Tel. +49-7131 28 22-0
Telefax +49-7131 28 22-400

Mecaner S.A.
Calle Aita Gotzon 37
48610 Urduliz - Vizcaya (Spain)
Tel. +34-94-6769100
Telefax +34-94-6769132

Comau Poland Sp. ,Z.O.O.
Ul. Turyńska 100
43-100 Tychy (Poland)
Tel. +48-32-2179404
Telefax +48-32-2179440

Comau Romania S.R.L.
Oradea, 3700 Bihor
Str. Berzei nr.5 Suite E (Romania)
Tel. +40-59-414759
Telefax +40-59-479840

Comau Russia S.R.L.
Ul. Bolshaya Dmitrovka 32/4
107031 Moscow (Russian Federation)
Tel. +7-495-7885265
Telefax +7-495-7885266

Comau SPA Turkiye Bursa Isyeri
Panayır Mah. Buttimis İş Merkezi
C Block Kat 5 no.1494
16250 Osmangazi-Bursa (Turkey)
Tel. +90-0224-2112873
Telefax +90-0224-2112834

Comau Inc.
21000 Telegraph Road
Southfield, MI 48034 (USA)
Tel. +1-248-3538888
Telefax +1-248-3682531

Comau Pico Mexico S. de R.L. de C.V.
Av. Acceso Lotes 12 y 13
Col. Fracc. Ind. El Trébol 2° Secc.
C.P. 54610, Tepotzotlán (Mexico)
Tel. +52-5 8760644
Telefax +52-5 8761837

Comau Canada Inc.
4325 Division Road Unit # 15
Ontario N9A 6J3 (Canada)
Tel. +1-519-9727535
Telefax +1-519-9720809

Comau do Brasil Ind. e Com. Ltda.
Rua Do Paraíso, 148 - 4º Andar
Paraíso - Cep. 04103-000
São Paulo - SP (Brazil)
Tel. +55-11-21262424
Telefax +55-11-32668799

Comau Argentina S.A.
Ruta 9, Km 695
5020 - Ferreyra
Córdoba (Argentina)
Tel. +54-351-4503996
Telefax +54-351-4503909

Comau SA Body Systems (Pty)
Hendrik van Eck Drive
Riverside Industrial Area
Uitenhage 6229 (South Africa)
Tel. +27-41-9953600
Telefax +27-41-9229652

Comau (Shanghai) Automotive Equipment Co., Ltd.
Pudong, Kang Qiao Dong Road Nr. 1300
Block 2 - Kang Qiao
201319 Shanghai (P.R.China)
Tel. +86-21-68139900
Telefax +86-21-68139622

Comau India Pvt. Ltd.
33Km Milestone Pune-Nagar Road
Shikrapur, Pune - 412208 (India)
Tel. +91.2137.678100
Telefax +91.2137.678110