Bold is required, regular is optional
< > encloses key combinations
variables, program_names and other names italic

Program:

Handling of programs, status, call trees, list, enter/exit programs

Status/call tree:

STATUS	shows status for all programs on the stack
STATUS ***taskno***	shows status for specified task, if task aborted call tree with program names and step numbers are shown

List:

DIRECTORY	lists all programs in memory (RAM)
DIR/M	lists all modified programs in memory (RAM)
DIR/?	lists all faulty programs in memory (RAM)
DIR *ro.**	lists all programs starting with *ro.*
(**LISTP** ***program***)	lists specified programs contents, not so useful

Delete:

DELETEP ***prog1,prog2***	deletes programs ***prog1*** & ***prog2*** in memory (RAM)
DELETE ***a.adv_cal***	deletes program ***a.adv_cal*** incl. all subroutines and all variables belonging to the program ***a.adv_cal***

Abort/restart:

ABORT *taskno*	stops program in task acc. to *taskno*
RETRY *taskno*	starts program – starts program on the step it stopped on
PROCEED *taskno*	starts program – starts program on the step after the one it stopped on (not recommended!)
KILL *taskno*	removes task *taskno* or specified task from the stack

Enter/exit:

SEE ***program****, stepno*	enter program to see or edit, enter step number to enter directly to specified step
SEE	if program stopped program name is not required
DEBUG ***taskno***	enter debugger to see/debug/edit in specified task
DEBUG	if program stopped program name is not required
<F4> (Exit)	exit editor/debugger to monitor

In editor/debugger:

<F3>	enter subroutine – cursor on correct line (CALL ro.sub)
<Shift+F3>	exit subroutine to calling program
<F11>	change from debug to monitor mode
<Shift+F11>	change from monitor to debug mode
<H>	browse among edited programs
<Esc+H>	enter selected/currently listed program
<Esc>	select COMMAND mode
<I>	select INSERT mode (recommended)
<R>	select REPLACE mode, overwrites
<L*nn*>	go to line *nn* – in COMMAND mode
<-*nn* Ctrl+Delete>	regain deleted lines ***nn*** pcs. – in COMMAND mode

Step in debugger:

Place cursor, in debugger, on line to step from, on dividing line current step number is shown;

XSTEP,,*stepnumber*	enter step for start step, NOTE two commas!
<Ctrl+g>	go directly to line where cursor stands, easier than XSTEP
<Ctrl+x>	step
<Ctrl+z>	step without entering subroutines – whole CALL executed
<Ctrl+b>	place breakpoint at current line
<Ctrl+n>	remove breakpoint at current line
BPT	remove all breakpoints in current program

Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Files:

Handling of files, list, copy, save, rename, directories

List:

FDIRECTORY	lists all files and directories in default directory
FDI *drive:\directory\file.ext*	lists files acc. to specification
FDI *drive:\directory*.V2*	lists all files with the extension ***V2***
FDI *drive:\directory*N*.**	lists all files containing ***N***
FDI **.PG*	lists all files with the ext. ***PG*** in default directory

List contents:

FLIST *drive:\didirectory**file.ext***	lists contents of the specified file, seldom used

When copying, saving and renaming, already existing file(s) must be deleted!

Copy:

FCOPY *drive:directory\destfile.ext=drive:sourcefile.ext* copy

FCOPY *c:applic\db-type.st=a:db-type.st* copies file ***db-type.st*** from diskette to directory ***applic*** on hard drive (flash)

Save:

STORE *drive:directory\file.ext*	saves all in RAM to disk
STORE *a:33050123*	saves all in RAM to diskette, extension will be ***.V2***
STOREP *a:33050123*	saves all programs in RAM to diskette, ext. ***.PG***
STOREL *a:33050123*	saves all locations in RAM to diskette, ext. ***.LC***
STOREP *a:ro-main=ro.main*	saves all programs under ***ro.main*** to diskette, ext. ***.PG***
STOREP/*n a:ro-main=ro.main*	saves ***n*** levels under program ***ro.main*** to diskette, if 1 specified only ***ro.main*** is saved – not saving subroutines

Rename:

FRENAME ***destfile.ext=sourcefile.st***	rename file
FREN *db-type.stb=db-type.st*	renames the file ***db-type.st*** to ***db-type.stb*** (BAK-fil)
FREN *applic\main.pg=main.pgb*	renames the file ***main.pgb*** to ***main.pg*** in directory ***applic***

Delete:

FDELETE ***drive:\directory\file.ext***	deletes file
FDEL *c:\applic\main.pg*	deletes file ***main pg*** in directory ***applic*** on hard drive (c:)
FDEL **.pg*	deletes all files with extension ***.pg***

Directories:

FDI/*c drive:\directory*	create directory (***/c*** = create)
FDI/*c c:\course*	create directory ***course*** on hard drive
FDI/*d drive:\directory*	delete directory (***/d*** = delete), directory must be empty
FDI/*d c:\course*	delete directory ***course*** on hard drive
CD *directory*	change default directory (**C**hange **D**irectory)
CD *course*	change default directory to ***course***
DEFAULT ***D=\directory***	change default directory
CD *course*	change default directory to ***course***

Network / TCP/IP / Ethernet / NFS

NFS>	always before file specifications on PC/Network else as above (NFS=Net File Server), long file names works
NFS>	MV controller with net board requires full name
NET	network status, IP address and mounts shown
PING *aaa.aaa.aaa.aaa*	connection check, ***aaa…*** = IP address or node name
FSET tcp /NODE '*pc*' /ADDRESS *aaa aaa aaa aaa*	set node to ***pc*** with IP address ***aaa…***, note spaces not periods and no ' for address
FSET nfs /MOUNT '*xc*' /NODE '*pc*' /PATH '*C:\Adept*'	mount ***xc*** path ***C:\Adept*** on node ***pc*** (older Omni)
FSET nfs /MOUNT '*xc*' /NODE '*pc*' /PATH '*/Adept*'	mount ***xc*** named ***/Adept*** on node ***pc*** (Allegro)

Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Variables:

Handling of variables, list, find, set, local, global, automatic, in/out and jokers

List:

All:

LISTR	reals – numbers
LISTS	strings – text/characters
LISTL	locations – positions

Some:

LISTR/LISTS/LISTL ***var_1****,var_2,var_3…*	list one or several variables (separated by comma)
LISTR/LISTS/LISTL ***var[ ]***	list array
LISTR/LISTS/LISTL ***var[10]***	list array element with index 10
DO *@task* **TYPE** ***variabel***	(not locations), automatic may exist in several tasks

With jokers: (especially when not sure of exact variable name)

LISTR/LISTS/LISTL ***ro?***	lists all variables starting with ***ro***
LISTR/LISTS/LISTL ***ro?[ ]***	lists all one dimensional array variables starting with ***ro***
LISTR/LISTS/LISTL ***ro?[,]***	lists all two dimensional array variables starting with ***ro***
LISTR/LISTS/LISTL ***ro?[,,]***	lists all three dimensional array variables starting with ***ro***

Local/automatic: reservation for automatic – only if program on stack

LISTR/S/L ***@ro.main***	lists all variables in program *ro.main*

In debugger:
Double-click on desired variable, for expressions double-click on parenthesis or equal sign.

Set:

When setting a variable manually the built in security in the program is bypassed, so it is very important that you know what you are doing! DO can only be used in aborted task, use ***@task*** to avoid aborting task 0 (default).

Reals (numbers):

DO *@task* ***variable=123***	set ***variable***=number ***123***
DO *@task* ***variable=variable_2***	set ***variable***=the value of ***variable_2***
DO *@task* ***var[1]=var[2]***	copy the value in ***var[2]*** to ***var[1]***, old value in ***var[1]*** is lost

Strings (text/characters):

DO *@task* ***$variable=”text”***	set ***$variable=text***
DO *@task* ***$string=$name***	set the string ***$string***=contents of the string ***$name***

Locations:

DO *@task* **SET** ***loc*=TRANS*(x,y,z,yaw,pitch,roll)***	with coordinates as variables
DO *@task* **SET** ***loc*=TRANS(300,200,850,0,180,90)**	with coordinates as numbers
DO *@task* **SET** ***loc=loc2***	set ***loc*** equal to ***loc2***
DO *@task* **SET** ***loc=loc2*:TRANS(300)**	set ***loc*** to ***loc2*** offset 300mm in x in *loc2's* directions
HERE ***loc***	set ***loc*** to the selected robots current location
POINT *loc*	set new location or edit existing

In debugger:
Click/select desired variable, press **<Shift-F5>** (**"TEACH"**), enter new value/text med ”” (i.e. ”new text”)/locations with **TRANS*(x,y,z,p,r)*** (change of locations in debugger not recommended! use POINT)

Delete:

DELETER	deletes all REAL variables in RAM
DELETES	deletes all STRINGS in RAM
DELETEL	deletes all LOCATIONS in RAM
DELETER/S/L ***var, var2***	delete specified variables in RAM

Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Variables: (cont.)

There are three types of variable definitions (scopes) global (default), local and automatic.
Global and local variables are undefined from start.

Global:

Global variables should only exist when used by several programs for instance databases (variant data), or for communication between programs. Global variables usually have same prefix as the routines they control like vi.--- for vision routines or ro.--- for robot routines.

Local:

Variables are defined local by the command **LOCAL *var1****, var2…*
When a program been executed the local variables remains in memory and can always be listed/used again.
Local variables are suitable if you want a program to remember for instance what happened last time it was called. (corresponding VB Static)

Automatic:

Variables are defined automatic by the command **AUTO *var1,*** *var2…*
Automatic variables uses temporary memory space and can adopt any value when not assigned in the program and their contents are only valid when program on stack.
The advantage with automatic variables is that a program can be executed in several tasks without the variables interfering with each other. For every task separate memory space are reserved for automatic variables.

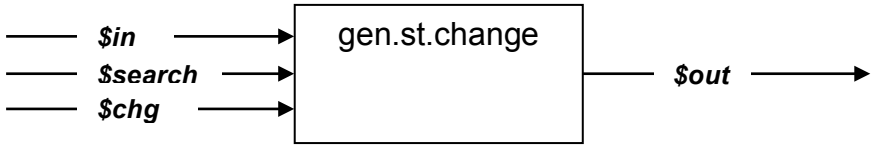
In and out variables:

In program calls in resp. out variables may be used to send variables between caller and subroutines. Subroutines can be seen as a "black box" where data are sent in and a result is sent out.
In/out variables behave as automatic variables and they are by default by reference, expressions and/or parenthesis make them by value.

CALL gen.st.change*($in,$search,$chg, $out)* program to search and replace text in strings.

$in	contains text to change
$search	contains characters/text to search for
$chg	contains characters/text to change to
$out	contains changed text

In the subroutines the variables are described under INPUT PARM.: resp. OUTPUT PARM.:



Signals:

Change state of signals:

SIGNAL ***signal,*** *sig2*	sets signal, several divided by comma
SIGNAL –***q.ro.down***	sets signal ***q.ro.down*** low – minus (**-**) before
SIGNAL ***2****,-2003,f.quit*	can be address or variable

Show state of signals:

LISTR SIG(*i.ro.open*)	shows the state of the signal ***i.ro.open*** (***-1=ON – 0=OFF***)
IO	in- and outputs
IO 1	inputs
IO 2	flags
	end with <Ctrl+c>

To see the state of signals it is easier to use menu functions.

Device-net:

DEVICNET	lists nodes and status of device-net
DN.RESTART	restarts device-net if a node been removed or faulty

Device-net nodes are scanned and mapped in configuration utility (config_c/ACE).

Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Error trapping:

At stop:

Time-out or Error-alarm

Check – rectify

Open Monitor

Check/note any error message:

*** Error message ***

Program task *nn* stopped at *progname* step *nnn date time*

Most common error messages:

*** Undefined value ***	program stopped due to an undefined variable
*** Arithmetic overflow ***	division by zero, or any other unreasonable calculation

If error message has disappeared from monitor, for instance when opened/edited a program use **RET *taskno*** (**RET**RY) to show message again.

or…

Are all task running **STAT** (**STAT**US), check call tree **STAT *taskno***

Check if stopped and in which program and what step.

Enter program **SEE *programe***, *step* specify *step* to put cursor on step line directly.

Various errors in programs:

*** Undefined value *** find out the name of the variable, is it local or global, where and how is it defined/set, write down program step number and variable name and under what circumstances the error arose, find out what value the variable should have – use debugger (local/auto variable) or monitor (global) to set the variable to the correct value.

Variables may be searched for programs by pressing **<F7> (FIND)** and **<F8> (REPEAT)** to find where it is defined. If an array variable (like ***$db.var[i]*** check that pointer/index inside brackets ([*i*]) is defined and has the right value.

Waiting on a variable **(WHILE, UNTIL, IF…GOTO)** program has not stopped, use debugger–step through program and find out why program won't continue, what conditions are not fulfilled – what are the variable names in expressions which values should they have – are variables from another task, why has it not set the variable to the right value maybe the fault is there.

Screen dump

For errors that cannot be rectified in an easy way – that might require controller restart or similar – take a "screen dump" i.e. a picture of the PC screen with any alarms and/or status list. Make sure that all desired parts of windows are shown on screen! Are there any error messages also in the monitor – make sure visible at dump.

Use PScreen program if exists – press icon on task bar – saved in directory "Pictures\PScreen\".

Press <Print Scrn> (Print Screen) – start Paint, paste in and save in suitable directory.

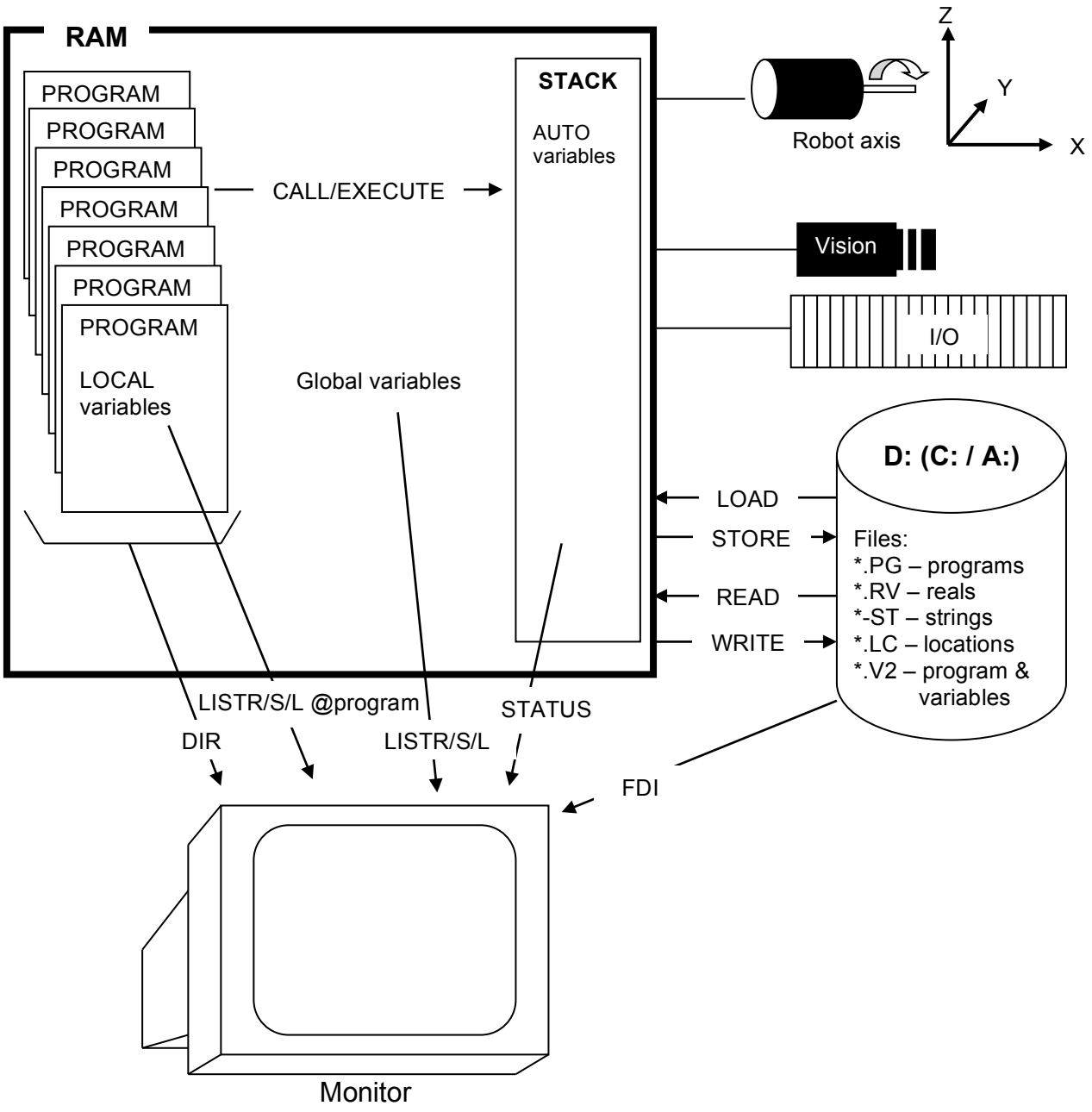
These pictures make it easier for supplier/programmer to find and correct the cause of the error. Screen dumps are easy to e-mail.

<Alt+Print Scrn> saves only selected window on screen.

See also systematization document, program headers and any readme document/program for more information about the applications variables.

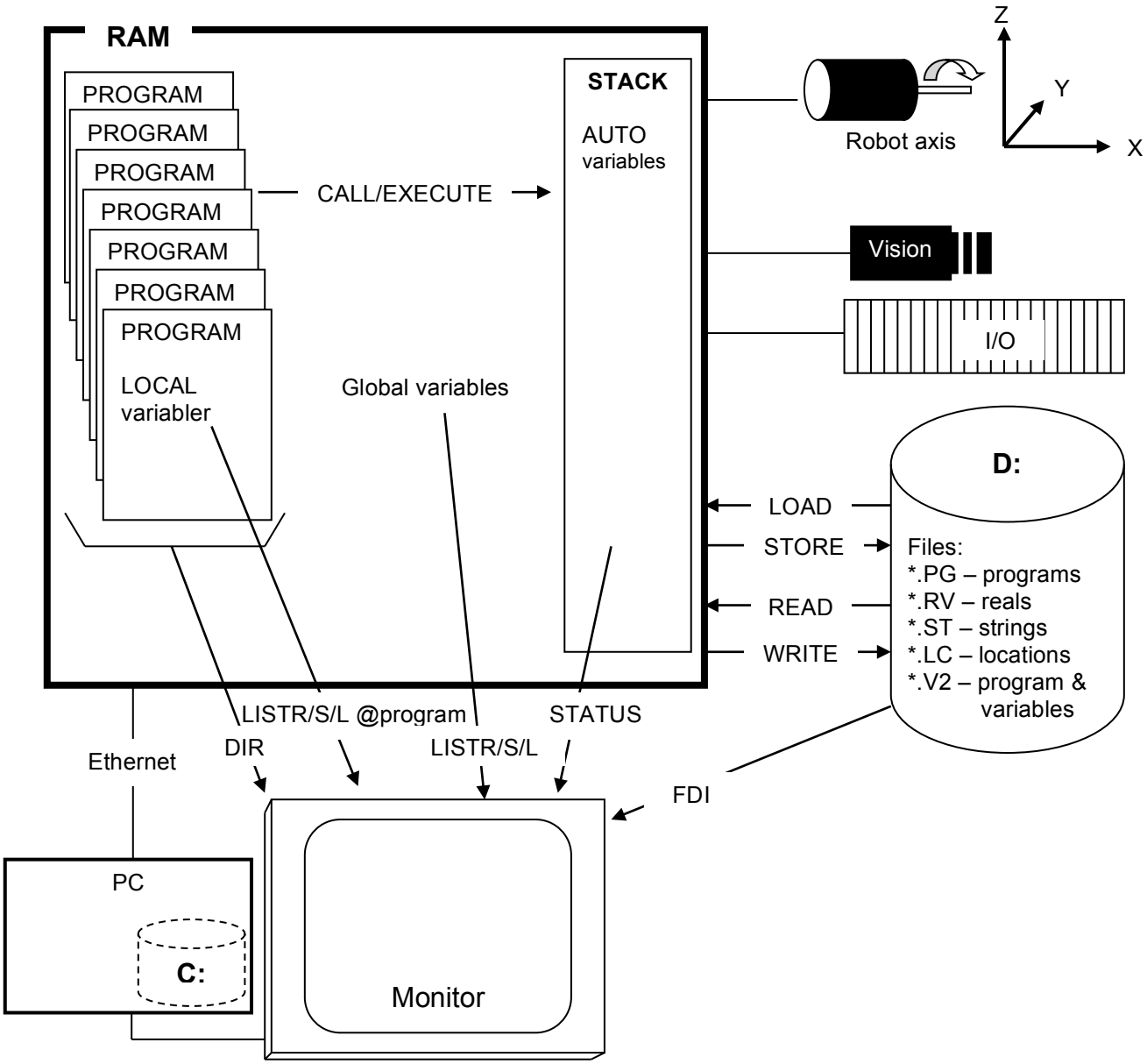
Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Overview MC/MV: variables – programs – files – RAM – stack



Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Overview AWC/CX: variables – programs – files – RAM – stack



Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni

Word list

array a list of variables that can be one, two or three dimensional typed; "variablename[x]", "variablename[x,y]" resp. "variablename[x,y,z]" , x/y/z are element numbers in resp. dimension.

blob an area in a vision image that can be seen as an object.

digitala I/O digital in-/outputs, signals, used for communication between systems and control of valves (outputs), check of sensors (inputs) etc.

directory or catalogue, a subgroup with files, used to organize files on hard drive and diskettes.

edge here for vision, edge/side of an object.

location position, a stored point that the robot can move to or use in calculations of new locations.

pendant programming box.

pixel picture element, the small light sensitive semiconductor elements that a vision camera's sensor surface ("chip") is made of, these converts the light to electric signals that can be treated by the controller. Measures of object in the vision image are often noted in millimetres but sometimes in pixel. At camera calibration the factor between mm/pixel is calculated so the system can present and calculate measures in mm.

real real variable, number variable, contains a numeric value.

real time data shown in real time means that current data is shown and updated directly when changed – direct updating and not at specific periods or events.

string string variable, text variable, contains text/characters, variable names are written with a preceding $-sign ($text).

task in the Adept system several programs can run simultaneously/parallel, so called multi-tasking, the various parallel processes are called task(s). They are numbered 0-27.

variable a register that can have any name and can contain various values or text. (1234, "component type ABC", "Charlie").

Bråmåbovägen 120
792 90 Sollerön
Tele: +46(0)70 54 220 60
mtih@telia.com
TeknikiHarmoni