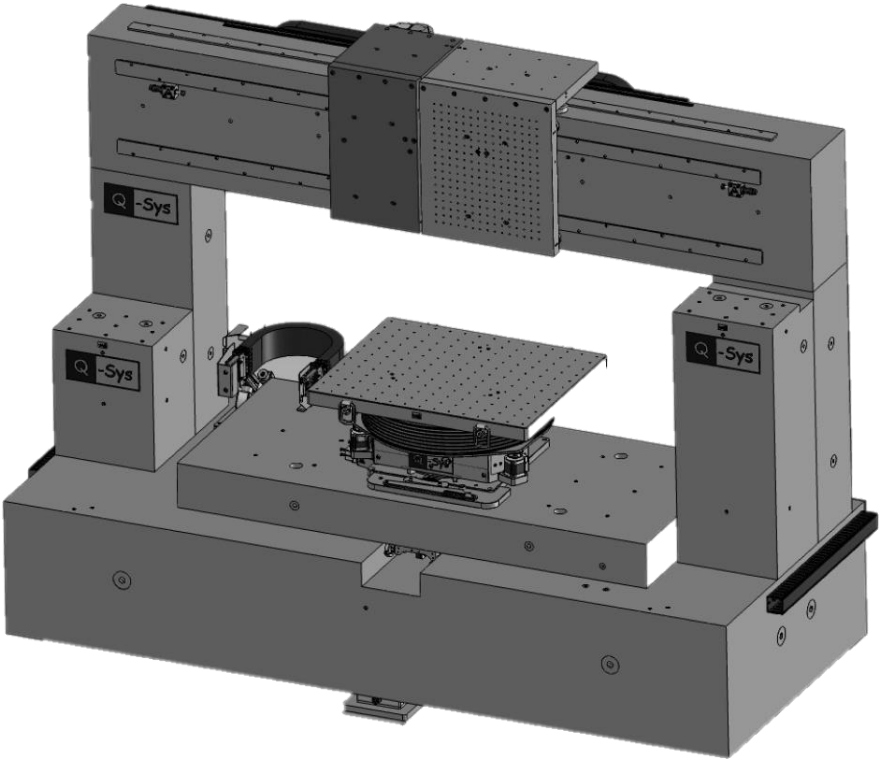
Software integration manual

for

5 Axes Metrology Platform

Model 0390-01



Q -Sys

Document Revision: A

Table of Contents

1 Introduction

1.1 Purpose of this Manual

The purpose of this manual is to provide information to assist application development for the metrology platform. Items covered by this manual are:

- Hardware interface
- Safety integration
- Motion controller documentation
- IDE software
- QsysWare
- Motion commands

The system described in this manual is designed and built to fulfill certain critical motion functions within the context of a machine. The system is not a machine in itself and therefore this manual should not be used and read as an operator or user manual of the machine.

1.2 Copyright

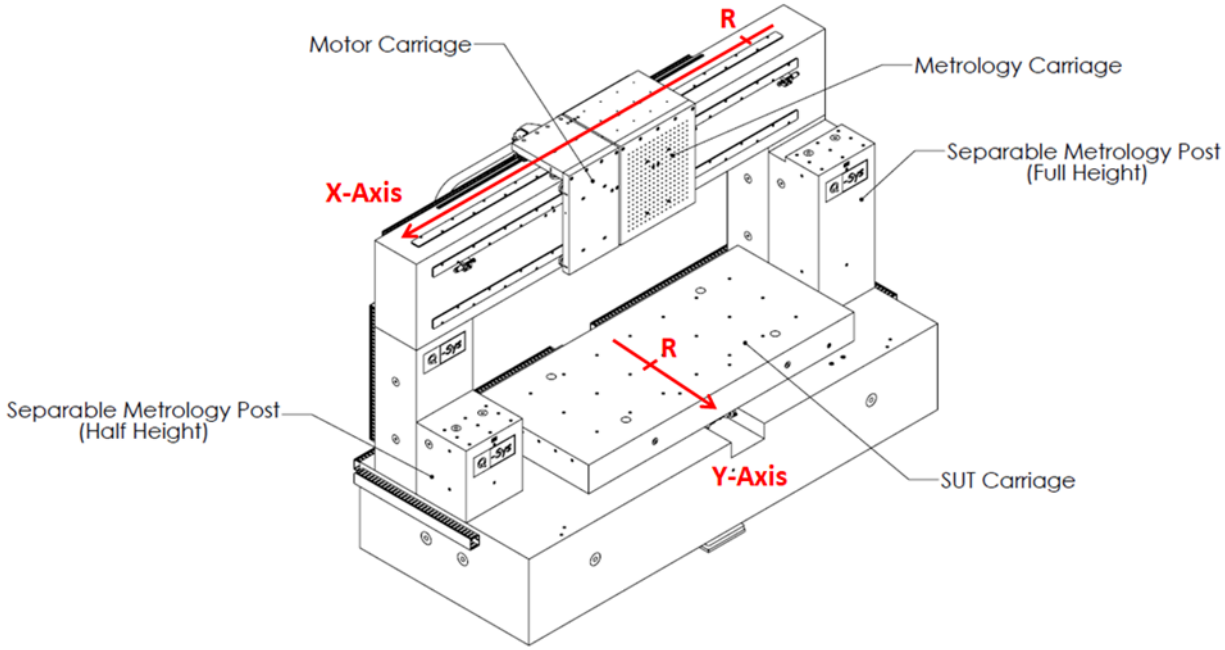
1.3 Contact Information

Should you at any time have a question or require additional information then please do get in touch with us. Below please find our contact details to use should you not have a direct contact already. In case you want to ask your question outside our normal working hours you can always use the email address given below. We will return your call ASAP.

Q-Sys BV
Grasbeemd 15-B
5705 DE Helmond
Netherlands
Phone : +31 492 714434
Fax : +31 492 714435
E-mail : support@Q-Sys.eu
Website: www.Q-Sys.eu

2 General System Description

2.1 Overview



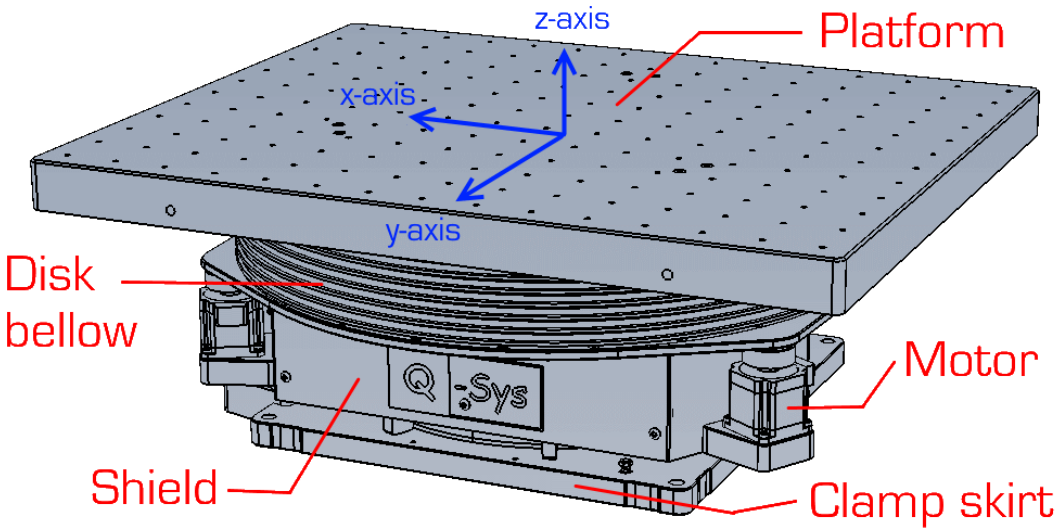
The metrology platform comprises of two linear motion axes, one rotary axis and two tilt axes. The picture above shows the two linear axes. The basis of the system is a granite base on which a granite bridge is mounted. The granite surfaces have been accurately machined and function as a guidance surface for the air bearings.

The top axis is referred to as the X-axis. This axis provides the movement in the longitudinal direction. It consists of two aluminium carriages: a motor carriage and a metrology carriage. The connection between the two carriages is optimized to provide a stiff connection between motor and metrology carriage, and at the same time minimize the heat transfer to the metrology carriage. The motor carriage is driven by an ironless core linear motor. On each carriage a linear encoder read head is mounted to provide position feedback.

The lower axis is the specimen platform and is referred to as the Y-axis. This axis has a moving granite carriage and has its guidance surface on the granite base. This axis is directly driven by a linear motor with an ironless core. The position feedback is obtained through a high accuracy linear encoder.

The granite structure has two post positions to support the stationary metrology equipment. The posts are removable and interchangeable to allow for more flexibility in the experimental setup.

On top of the lower axis the so called rotary-tip-tilt can be placed.



The combined stage is constructed in a stacked configuration. The lower stage is a rotary air-bearing stage with direct drive and optical encoder feedback. The rotary axis has a travel of almost a full circle, which makes it more than suitable for 180 degrees flipping as well as for doing angular yaw alignment. On top of the rotary stage, a tip-tilt stage is constructed. The tip-tilt is created by the combined motion of three linear drive trains. By translating these three actuators, the top platform can be aligned over the roll and pitch angles. As a consequence of this configuration, the three actuators can also be used to translate the platform in the z-axis direction.

The stage is designed to provide exceptional performance, in accuracy as well as in mechanical and thermal stability.

2.2 Axes naming and units

The naming of the axis in the software is determined by the delta tau control platform.

The linear motion axes are referred to as X, Y and Z, with the positive direction as shown in the pictures above. The displacement of the linear axes is expressed in micrometer units.

The rotary motion axes are referred to as the A, B and C-axis. The A-axis is the rotation around the X-axis, with the direction defined by the righthand rule (a clockwise turn when looking in the positive linear direction). Similarly, the B-axis and the C-axis are the rotation around the Y and Z-axis respectively. The displacement of the rotary axes is expressed in degree units.

Note that the A and B rotary axis definition is based on the X and Y-axes for the RTT stage. This coordinate system is locked to the platform, which rotates. When the RTT system C-axis is in the zero (reference) position, the X and Y axes of both coordinate systems coincide.

3 Motion controller

3.1 Downloads

The motion platform is supplied with an advanced motion controller from Omron Delta Tau. The controller type is an 8 axis Power PMAC Brick LV, part number BL8-AA1055-00S00000. The following resources are relevant for the product and are highly recommended for use during the integration:

- Power PMAC Integrated Development Environment (IDE) Software
  - https://automation.omron.com/en/us/products/family/pmac%20ide
  - This installation includes the Power PMAC IDE manual

- Power Brick LV ARM information
  https://automation.omron.com/en/us/products/family/brick%20lv
  - Power Brick LV ARM User Manual
  - Power PMAC Software Reference Manual
  - Power PMAC Users Manual

In this integration document, the machine specific aspects are described. This document is not intended to replace the Delta Tau documentation, it should be used in conjunction with each other.

3.2 Training

Below are some links to on-line resources. If the links are no longer operationl or the topic you are looking for is not covered, do not hesitate to contact us at Q-Sys.

- Youtube training
- Training documentation

3.3 Interface protocol

The power PMAC is a Linux (Debian) machine. Communication is done via a Telnet or SSH connection over TCP/IP. Once the SSH (or telnet) connection has been made, a terminal to the Power pmac must be created by starting the gpascii application. This opens the PMAC terminal which then can be used for interfacing with the motion platform. Note: use gpascii -2 to enable global parameters.

See the Pmac User Manual chapter TALKING TO POWER PMAC for a detailed description on how to set this up.

4 QsysWare

The Q-Sys machine is supplied with a software framework, referred to as QsysWare, which can be used by the customer as a back-end application. This framework implements initialization, housekeeping, and some machine specific functionality. Depending on the application requirements, customization might be required.
The QysWare framework implements:

- Position referencing
- Enable/disable axes
- Releasing air bearing axes for manual motion
- Power off encoders for minimizing thermal disturbance
- Error handling and recovery

The QsysWare framework does not implement any motion commands, other than the reference motion. The motion commands are done using standard PMAC commands as described in the Delta Tau manuals. In this manual, there is a short section which supplies some examples on how to implement motion.

The QsysWare software is written in script routines and can easily be customized. Its usage is is copyright free and can be used in any form as part of your own software project.

4.1 Qsysware Syntax

The QsysWare communication protocol is a very basic protocol, where the host PC is the master. The host master writes command data to specific parameters in the controller and retrieves status data from the controller by requesting the value of other specific parameters. The controller regularly processes the command parameters and updates the status parameter.

The command data consists of two parameters:

- **requestHost**
- **selectAxes**

where **requestHost** defines the action command and **selectAxes** defines the axis which the action command should affect. The tables below show the action commands and a list of the axes.
The parameter **selectAxes** is initialized to the value *selectAll*, which means that each requested action will work on all axes (if they fulfill the appropriate requirements). By modifying the **selectAxes** parameter, any desired combination of axes can be selected.
The order of operation is as follows:

- host sets the **selectAxes** parameter (if required)
- host sets the **requestHost** command
- controller processes action command and resets **requestHost** to requestNone

The possible action commands are shown in below table.

Table 1 Available commands

Action command	Value	Description functionality	Axis specific
requestNone	$0		
requestHome	$1	Execute zero referencing routine, After success go to controlled state	yes
requestReset	$2	abort all motion and re-initialize controller	no
requestAbort	$3	abort all motion, go to idle state	no
requestControl	$4	go to controlled state	yes
requestIdle	$5	go to idle state	yes
requestRelease	$6	release axes for manual motion	yes
requestEncoderPowerOff	$7	Switch off encoder supply for minimal thermal disturbance	yes

4.1.1 requestHome

This command is required after the controller is powered up and before the axis can be used for motion control. It starts the reference routine (homing procedure) which typically consists of motion until the negative end of travel switch is detected, followed by a motion until the encoder reference marker (sometimes referred to as Index) is detected.

Depending on the application requirements, different procedures are possible. Each axis has its own home plc which can be customized as required for the application.

4.1.2 requestReset

The Qsysware has housekeeping routines which monitor certain I/O and status indicators. Certain conditions are considered unwanted and may cause the housekeeping routine to raise and error flag. This will bring the software in an error state, in which no further action is possible until the error is reset. An error code (see status data) reflecting the nature of the error is available as feedback. Reset will clear this data and bring the system to its initial state.

4.1.3 requestAbort

Sometimes an action command or motion command needs to be aborted. This will bring all axes to an immediate controlled halt and into the idle state.

4.1.4 requestControl

This command brings the axis to the controlled motor state. This command will only function if the axis home sequence has been completed since the last reset. The valves to the air bearings are opened and the amplifier is enabled in closed loop control. If this state, motion commands can be issued (see section motion commands).

4.1.5 requestIdle

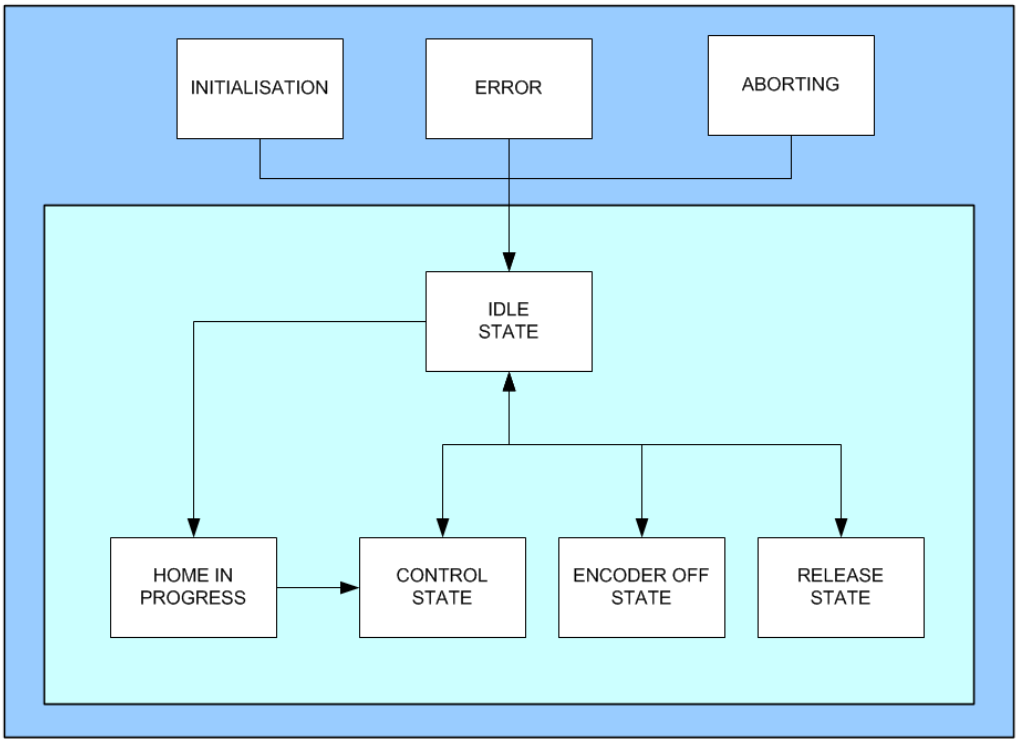
This command returns the stage to the idle state. The air valves to the bearings are closed and the amplifiers are disabled.

4.1.6 requestEncoderPowerOff

This command removes the 5V supply to the encoders of the RTT axes (C axis and ABZ axis). This removes the encoder readhead as a thermal disturbance source. All RTT encoders are switched simultaneously, so this command always affects both the C and the ABZ axis. The encoder power is restored after the requestIdle command is issued.
Switching off the power to the encoders does not result in a loss of the position data for the axis. The encoder position data is stored before switching power off and it is recovered when switching the axis state back to idle.

4.1.7 requestRelease

This command opens the air valve for the specified axis, thus allowing for manual movement of that axis.



The commands to Reset and Abort are not axis specific, which means that they always affect all axes, regardless of the value of **selectAxes.** The reset command aborts and resets the errors and will require rehoming.
The action command can also be viewed as a command to bring an axis in a certain state. Below table lists the possible states.

Axis State	ABZ	C	X	Y
Idle state	initial	initial	initial	Initial
Homing	x	x	x	x
Control	x	x	x	x
Release		x	x	x
EncoderPowerOff	x	x		

There are also two special states, which are axis independent: aborting and error. The abort action is triggered by the user and results in all motion commands being halted and all axes being brought to the idle state.
If an error occurs, an abort action is executed and the system is then brought to the error state. The user can recover the system by issuing a reset, which will bring all axes back in the (unhomed) Idle state.

4.1.8 status

The status of the QsysWare can be obtained by the host by sending a query for the variable QsysStatus. The value of QsysStatus is composed of several bits, where each bit has a particular meaning as described in the table below. Note that multiple status bits can be set.

QsysStatus	HexValue	Description
statusError	$1	An error has triggered, consult QsysError for details.
statusInitialized	$2	System is successfully initialized.
statusHomeComplete	$4	All axes have been homed
statusHomeInProgress	$8	At least one axis is currently homing
statusIdleState	$10	At least one axis is currently idle
statusPowerOffState	$20	At least one axis is currently power off
statusReleaseState	$40	At least one axis is currently released
statusControlState	$80	At least one axis is currently in controlled state

4.1.9 error

If the error bit of QsysStatus is set, the QsysError variable gives detailed information of the exact nature of the error and its possible cause. The value of QsysError is also composed of several bits, where each bit has a particular meaning as described in the table below. Note that multiple status bits can be set.

QsysError	HexValue	Description
noError		
powerOnResetError	$1	Error while initializing, possible hardware problem
SerialEncoderError	$2	Error not applicable for this system
homingErrorX	$4	Error occurred during reference routine X-axis
homingErrorY	$8	Error occurred during reference routine Y-axis
homingErrorC	$10	Error occurred during reference routine C-axis
homingErrorABZ	$20	Error occurred during reference routine ABZ-axis
stoError	$40	Safety input triggered (E-stop input)
pressureErrorC	$80	RTT air pressure detect error
airSupplyError	$100	Global air supply pressure error
statusErrorX	$200	Motor error X-axis
statusErrorY	$400	Motor error Y-axis
statusErrorC	$800	Motor error C-axis
statusErrorABZ	$1000	Motor error ABZ-axis

5 Commanding motion

5.1 Introduction

This section assumes that the axes and coordinate systems are enabled (control state). There are multiple ways to command axes motion with the Delta Tau controller. In this section, the on-line command method is shortly explained. Other methods for axes motion such as using programs and/or rotary buffers are explained in detail in the Delta Tau documentation.

5.2 Definitions

For commanding motion, it is helpful to first have some understanding of the meaning and interrelation of the basic motion objects motor, axis and coordinate system.

5.2.1 Motor

The motor object can be a physical or virtual component. It will in general have a control loop with control parameters (gains). This loop will typically use input from one or more feedback devices and/or the set point generator. It is the most basic component which is needed to create motion.
The Pmac IDE offers information on the motor status, the motor feedback values and offers a motor jog window to create some basic single motor motion.
For many applications, motors in combination with jog commands offer sufficient functionality. However, if multiple motors are used to create multi axes interdependent motion coordinate systems are more suitable.

5.2.2 Coordinate systems, axes and motion programs.

These are defined in the PMAC User manual as follows:

- *A coordinate system in Power PMAC is a grouping of one or more motors for the purpose of synchronizing movements. A coordinate system (even with only one motor) can run a motion program; a motor by itself cannot.*
- *An axis is an element of a coordinate system. It can be thought of as one of the coordinates of the tool, or of the mechanics relative to the tool. An axis in Power PMAC is often similar to a motor, but not the same thing. An axis is referred to by letter. There can be up to 32 independent axes in a coordinate system, selected from the X, Y, Z, A, B, C, U, V, and W single letters, plus the AA through HH, and LL through ZZ double letters.*

Axes, and their relationships to motors, are established either through on-line “axis-definition” commands, which set up a mathematically linear relationship between axis positions and motor positions, or through “kinematic subroutines” which permit these relationships to be defined algorithmically, with extensive math and logic.

Some important aspects:

- Axes can only commanded motion if all axes in the same coordinate system are enable (in control state).

- A motion program is runs can only execute commands for one coordinate system. This is also the case for a rotary buffer.

5.3 Executing motion commands

The pmac distinguishes between on-line commands and program commands. While most commands are possible as on-line commands, certain command structures such as multiline loops and if-then conditions can only be used in programs.
In case the user wants to run a motion program from a remote host PC, he would typically execute the following steps:

- Open/clear program buffer
- Download program in buffer
- Close buffer
- Select Coordinate system and Start program

Alternatively, he could use the on-line method:

- Sent online program commands using the **cpx** instruction.

Note that the commands on the line after the **cpx** command is a single program. A next line with a **cpx** command is a new program, so any setting from the previous line (such as modal commands as *inc* or *rapid*) are no longer active.

5.3.1 Examples

motor[5].jogspeed = 10
&3 cpx rapid inc X10000

This command moves the X-axis with the a speed of 10 mm/s over a distance of 10mm.

The *&3* sets the coordinate system mode to 3, and any program started from this terminal // will run in this coordinate system.

The *rapid* command sets the motion mode to fast point to point motion (in contrast to interpolated motion). By default, the axis speed when moving in this mode is defined by motor parameters *motor[x].jogspeed*.

The *inc* command set the motion mode to incremental. Alternative command is *abs* which sets the motion to absolute (relative to the axis zero position).

&1 cpx rapid abs A1 B-1 Z1000

This command moves the A axis to position +1 degree, the B to -1 degree and the Z-axis to +1mm, all relative to the reference position.

For more info see the Pmac user manual. There are also some program examples found here.

6 Hardware interface

6.1 Safety integration

The motion platform is prepared for integration in an external safety network. The control system has two safety inputs: an abort input and an STO input.
The abort input is intended to signal the controller that the motion controller is to be stopped immediately in a controlled manner and that the motors should be deactivated. If applicable, this is combined with removal of the air pressure from the air bearings and/or applying brakes, to prevent undesired axes movement.
The STO input can be used to remove the power from the controller in a safe manner in accordance with CE Machine safety regulations.
The following sequence is recommended if STO safety is required:

- Switch abort input
- Wait for the axes to stop/disable (0.5 second)
- Switch the STO input

Switching STO without abort, will result in uncontrolled motion and thus possible collision of user equipment. Note that if the delay time between STO and abort is too short, the air pressure is removed during motion. While this will not cause immediate damage, doing so regular will have a negative impact on the system performance.
Important note on the rotary Z:
The rotary Z air bearing is extremely accurate. Air pressure should ideally only be removed when the stage is at standstill. Removing air when moving at higher speeds or moving without proper pressure levels will result in reduced accuracy over time.

6.2 Limits, Flags, and EQU.

The controller connections X13 and X14 are wired via flat cable to break-out boards XB1 and XB2 in the cabinet. So the connection numbering of X13 corresponds with the numbering on XB1, and X14 with XB2.
Detailed information about X13 and X14 can be found in the Power Brick LV ARM User manual, chapter *Limits, Flags, and EQU (X13-X14).* In below tables, it is shown which signals are used in the machine. The signals which are not used can be used as generic inputs.

Table 2 X13 connections

	Channel 1	**Channel 2**	**Channel 3**	**Channel 4**
Sourcing/sinking	sourcing	sourcing	sourcing	sourcing
Negative limit	RX-RY-TZ	RX-RY-TZ	RX-RY-TZ	RZ
Positive limit	RX-RY-TZ	RX-RY-TZ	RX-RY-TZ	RZ
Home flag				
User flag				XS6:4 (relZ)

Table 3 X14 connections

	Channel 1	**Channel 2**	**Channel 3**	**Channel 4**
Sourcing/sinking	sourcing	sourcing	sinking	Free
Negative limit	TX	TY		
Positive limit	TX	TY		
Home flag			STO auxiliary	
User flag	XS6:2 (relX)	XS6:3 (rely)	Press. sensor RZ	

Connector XS6 is a prewired interface connector. Connecting any of these signals to XS6:1 (Common 0V) will toggle the software signals relX, relY or relZ.
The EQU signals are TTL outputs. They are typically intended for high speed triggering (such as cameras and lasers) and can be used with the real time data compare / data capturing functions. Contact Q-Sys if you need assistance with this.

It is also possible to use the EQU signals as generic 5V output.

For full details: consult the electrical schematic diagram.