PUMA Control

User's Manual

for PUMA Control Console and PUMA Control Software

Document created: 07.04.2021,   Last edited: 31.07.2021

PUMA: The Open Source (GPL v3.0) 3D printed microscope system designed for direct eye observations and ultra-portability with advanced options for digital imaging, measurements and computer control.

Contents in Brief

Contents in Detail

[This space is intentionally left blank]

Legal Information

License

Limitations of Use

The PUMA microscope and its associated systems do not have any certifications or regulatory approvals in any country for use in clinical diagnostics or treatment (human or veterinary).

The PUMA microscope and its associated systems are released to be used for research and educational purposes only.

Disclaimer

All PUMA project information, including without limitation any CAD file or STL file and all documentation, advice and instruction (whether provided in video form, audible form, written form or otherwise) is provided 'as is' in good faith and is intended to be helpful but comes with no warranty whatsoever.

Anyone attempting to build or use a PUMA microscope or other PUMA-related material, accessory, module or derivative is hereby advised that there will be risk involved in 3D printing, post-print processing, assembly and usage of the resulting structures. This risk includes, without limitation, the risk of personal damage and loss of resources.

Dr Paul J. Tadrous, TadPath and OptArc cannot accept any liability for any such loss or damages that may occur. All those who attempt to build or use any aspect of the PUMA project or derivatives thereof do so at their own risk.

Safety Information

Throughout this manual please take heed of **warnings given in bold text and highlighted yellow** to avoid possible damage to equipment and/or harm to people.

PUMA microscopes and associated systems are not toys. They contain small parts which may come loose such as tiny metal screws and washers and glass components that may splinter or break or otherwise present a choking or sharp object hazard or chemical hazard (for batteries). **Please do not let babies or young children play with or use any aspect of a PUMA system without close appropriate adult supervision. Likewise keep PUMA systems away from pets**.

Known Bugs / Limitations

The interaction of the power regulation board with the LED lamp and the power regulator built into the Arduino Nano microcontroller, together with the relatively simple LED current ammeter circuit means that there are some issues with the system that you should be aware of.

1. The power regulator can sometimes loose current regulation. This is known as 'tripping the power regulator' which can cause excessive and damaging current flow to the LED lamp. This is described in more detail on page 15.

2. Spurious ammeter readings. When using batteries, as these wear down and loose voltage and current capabilities the reading on the ammeter can spuriously increase to the point where this causes an over-current alarm even when there is still enough current draw capability to avoid tripping the power regulator. This issue is further discussed on page 15. For this reason the ammeter reading while on batteries should not be regarded as accurate. With v1.1 of the software you now have the option to disable the ammeter (and any alarms associated with it) to avoid annoying false alarms but this must be used with caution because you also loose the ability to detect a true power regulator trip event.

Abbreviations

Some common abbreviations used in this manual are listed below for convenience.

AR	Augmented reality (see page 16)
CM	Control mode (see page 24)
FOV	field of view
GUI	Graphical user interface
H&E	Haematoxylin and Eosin
HUD	Heads-up display (see page 17)
IDE	Integrated development environment (software)
PC	Personal computer
PCC	The PUMA Control console (see page 10)
SLM	Spatial light modulator (see page 19)
TFT	Thin film transistor (liquid crystal display)

Introduction

PUMA is an open source 3D printed high quality microscope system. The name PUMA stands for ***P***ortable, ***U***pgradeable, ***M***odular and ***A***ffordable – key features of this system.

Due to its modular design, a PUMA microscope can come in various configurations from a simple mirror-based microscope that uses no electronics or software at all, to an advanced research microscope with motorised Z-stage and TFT screens for optical image overlay and Fourier optics digital light processing. These electronic peripherals are controlled by an ATMega microprocessor running software to control them.

PUMA comes with it's own standard software known as "PUMA Control". This manual explains how to use this standard software. Because PUMA is an open source project you may customise this software or write your own interface from scratch. You can even run the PUMA peripherals via a desktop PC or smartphone with your own software interface. This manual does not discuss such alternative control strategies.

PUMA Control was designed to run on an Arduino Nano v.3 microprocessor board programmed using the Arduino IDE v.1.8.15. It uses 30682 bytes of program storage space.

The peripherals that PUMA Control is designed to run are those that are contained in or connect to the PUMA Control Console (PCC) – an Arduino-based control box device that is part of the PUMA open source standard. Instructions on how to build the PCC are provided on the PUMA GitHub pages. It is possible to build a prototype of the PCC on a standard breadboard following the circuit diagram provided on the PUMA GitHub site if you want to try the software but don't want to go all the way to making the finished portable control console.

PUMA Control provides the following input functionality:

- Receives user input from a 5-way joystick keypad.
- Monitors the current flowing to the microscope LED lamp.
- Monitors the state of the focus mechanism Z limit switch.
- Monitors the position of the Z-stage by an open loop system.

PUMA Control provides the following output functionality:

- Control of a stepper motor (for the focussing mechanism of the microscope).
- Displays the current flowing to the microscope LED lamp on a TFT screen.
- Generation and display of a graphical user interface (GUI) on a TFT screen.
- Generation and display of various aperture patterns on a TFT screen.
- Generates audible feedback signals to the user via a piezo buzzer.

This manual describes how to interpret and control these features so you can use the advanced features of the PUMA microscope.

This manual pertains to PUMA Control v1.1

Hardware

PUMA Control is all about controlling certain hardware so you can't understand how to use it without knowing something about that hardware. This chapter familiarises you with the input and output devices that you will be using with the PUMA Control software. The terminology and abbreviations introduced here will be required to fully understand the rest of the manual.

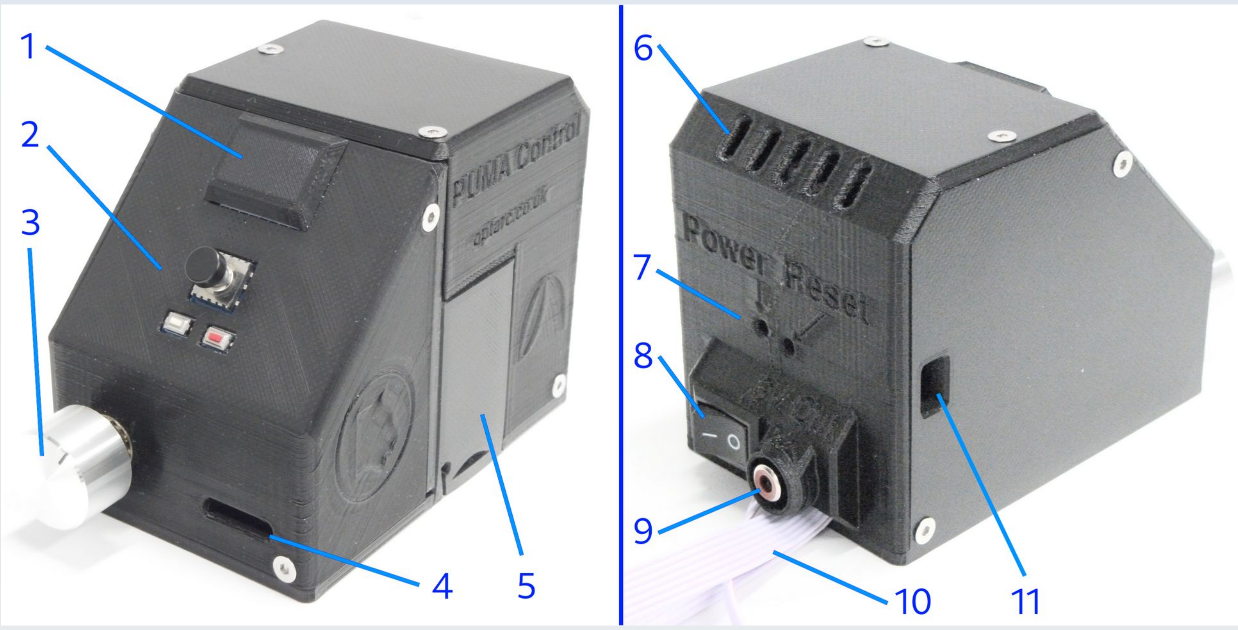


Figure 1. The PUMA Control Console (PCC). Left: Front and right aspect. Right: Back and left aspect. Labels: 1. Mouse upgrade port (this is a window designed to allow an external mouse to be used instead of the integrated joystick keypad. The mouse device has to be a special PUMA mouse designed for Arduino – not a standard USB or PS2 mouse). 2. Joystick keypad. 3. Lamp brightness potentiometer (Bourns 10-turn linear type). 4. Window to allow viewing of the Z-motor control board status indicator LEDs., 5. Battery compartment slide-on cover. 6. Passive cooling vents. 7. Window to allow viewing of the Arduino Nano power indicator LED and to allow access to the reset button. 8. Power On/Off switch. 9. Power output to the microscope LED lamp. 10. Cables to connect to external peripherals (Z-motor, Z limit switch and TFT screen of either the SLM or the AR projector). 11. Access to the USB port of the Arduino Nano.

The PUMA Control Console (PCC)

The PCC is the battery-operated portable control box that houses the Arduino Nano microprocessor which runs the PUMA Control software. It has the following features (see figure 1):

- It runs from two 9v batteries (type 6LR61) or, alternatively from an external power source with the following characteristics:
  - it must be able to provide at least 1 Amp of current,

- it must have a Voltage anywhere in the range of 7 to 12 Volts,
- it must be able to connect via the standard 6LR61-type connector.

Such an external power source may be an external battery pack (e.g. holding 8x AA batteries) or a regulated mains power adapter.

- It has an internal piezo buzzer to produce audible feedback signals for the user.
- It contains a ULN2003-based stepper motor controller board with 4 LED indicators visible from a window in the front of the casing.
- It has a 10-turn Bourns potentiometer on the front connected to an internal constant current power regulator board to provide some manual user-control over the current output to the microscope lamp. It is advisable to keep this potentiometer turned down (to its low brightness setting) or at least to its mid-way position whenever the unit is switched off (and prior to switching it on) because otherwise the PCC may draw a large inrush current when switched on and this could result in a transient voltage drop at the input of the internal power regulator that could 'trip' requiring immediate corrective action (see 'Tripping the power regulator – Immediate Action Required') below.
- It has a 5-way joystick plus 2-button keypad on the front panel as a manual user interface. More detail on the ways you can provide input from this keypad will be given below.
- It has a phono socket on the back which is a 5 V power output port for the microscope LED lamp. **Only connect the microscope LED lamp to this phono socket – it is NOT a video or audio port. Attempting to connect any video or audio equipment to this phono socket could permanently damage the audio or video equipment.**
- It has three ribbon cables coming out of the back with JST XH connectors on them. These are for connecting to the Z-motor (the 5-pin connector), the Z-limit switch (the 4-pin connector) and an ST7789 miniature TFT screen (the 7-pin connector).

  ST7789 TFT screens come in two forms: one that works from a 5 Volt supply in both logic and power supply and another form that can only work from 3 Volts. **Only the 5 Volt versions of these mini TFT screens can be used with the PCC. If you attempt to use a 3 Volt TFT display you will instantly and permanently destroy the display.** The PUMA open hardware standard specifies that only 5 volt versions of the ST7789 TFT screen may be used when constructing the augmented reality (AR) projector and the spatial light modulator (SLM).

Note that all text and numbers displayed via the PCC on a TFT screen will be in mirror-writing. This is because the optics of the scope optically ‘flip’ the display image before it is presented to the user. A single PCC can only connect to one TFT device at a time so either the AR projector or the SLM can be used – not both simultaneously. You would need two PUMA Control consoles to run both these modules simultaneously. If you require a GUI but don’t want to build (or otherwise don’t have) either an SLM module or AR projector then you can connect a bare 5 Volt compatible ST7789 screen to the PCC but you will see the GUI in mirror-flipped form. You can use a mirror to correct this. If you do this take care to ensure the connections are the correct way round because ST7789 modules usually do not come with a JST connector attached. For bare pins connections the ‘Gnd’ (ground) pin of the TFT module connects to pin 1 of the JST connector coming from the PCC (pin 1 is the pin adjacent to the ‘V’-shaped notch in the short side lip of the casing of the JST XH female connector and this notch can be seen in figure 2). For connecting to the AR projector unit, the correct polarity is shown in figure 2.

Note that these TFT screens are NOT ‘hot swappable’ i.e. you must have the screen device of your choice connected prior to switching on the PCC so the screen electronics can ‘boot up / sync’ correctly. If you need to change the TFT screen you must switch off the PCC first then disconnect the current screen, connect your new screen and then switch on. Attempting to change screens while the PCC is powered up will likely result in the newly attached screen not working (although this can usually be remedied by switching the PCC off then on again). You may find during normal use that the GUI on your connected TFT screen freezes and becomes unresponsive. This happens if the connection between the screen and PCC is disrupted e.g. by moving a cable that has a tenuous connection. The only way to fix this is to switch the PCC off then on again – and ensure the connection is robust (possibly strengthened by taping the connector ends together) to avoid repeat disruption.

- A USB port to allow connection of the PCC to a computer for the purposes of programming the PCC’s internal Arduino Nano, downloading data from the Nano to an external device or interfacing the unit to an external computer. See the section on ‘Programming the PCC’ below for more details on how to use this port including important safety information.

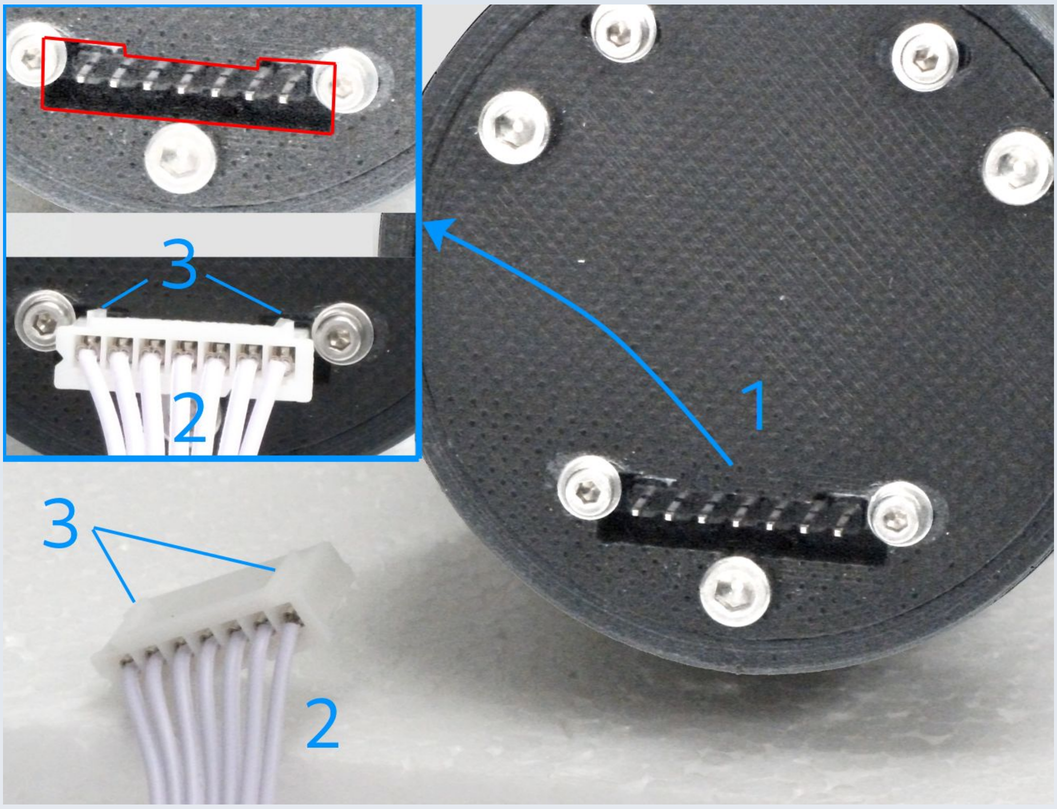


Figure 2. Polarity of connection to AR Projector. Main figure: Back of AR Projector (see figure 4 for the whole AR Projector). Inset shows the connector with its outline highlighted in red (top) and the connector in place (bottom). The JST connector has two raised lugs on one surface and these go into the recesses provided in the cut-out for the connecting pins at the back of the AR projector as shown. 1. Cut-out and pins at the back of the AR projector. 2. 7-pin JST XH connector coming from the PCC. 3. Lugs on the 7-pin JST XH connector.

Using an external power source

To use an external power source with the PCC (either an external battery pack or a mains adaptor) you must do the following (figure 3):

1. Switch off the PCC and open the battery compartment
2. Remove the two 9V batteries from inside (completely disconnect both of them)
3. Put the upper battery connector lead back inside the battery compartment but externalise the lower battery connector lead.
4. Ensure that the externalised battery connector lead cable is in the ‘V’-shaped recess in the case then slide the battery compartment cover in almost all the way (but not fully).
5. Thread the externalised battery connector cable into the slot in the battery compartment cover then close the cover all the way.

You now have an externalised 6LR61-type connector which you can use to connect an external power source, including a regulated mains adapter. Ensure that the connected source provides power in the range of 7 to 12 Volts (any higher will damage the Arduino Nano, any lower and some components will not work properly or at all). The external power supply must be able to supply at least 1 Amp of current or the PCC will not function correctly and may give unpredictable / erratic behaviour. Also, if current is insufficient, it can result in a voltage drop at the input of the internal power regulator of the PCC and this can cause it to 'trip' requiring immediate corrective action (see 'Tripping the power regulator – Immediate Action Required', below).

Important: Always remove both internal batteries as described above when using an external power source (whether battery pack or mains adaptor). **Never leave an internal battery connected when using an external power source or the internal battery may leak corrosive toxic chemicals or explode**. This is because both battery connectors are directly connected to each other and you will effectively be 're-charging' the connected internal battery using the external battery or power source and this re-charging will be uncontrolled.

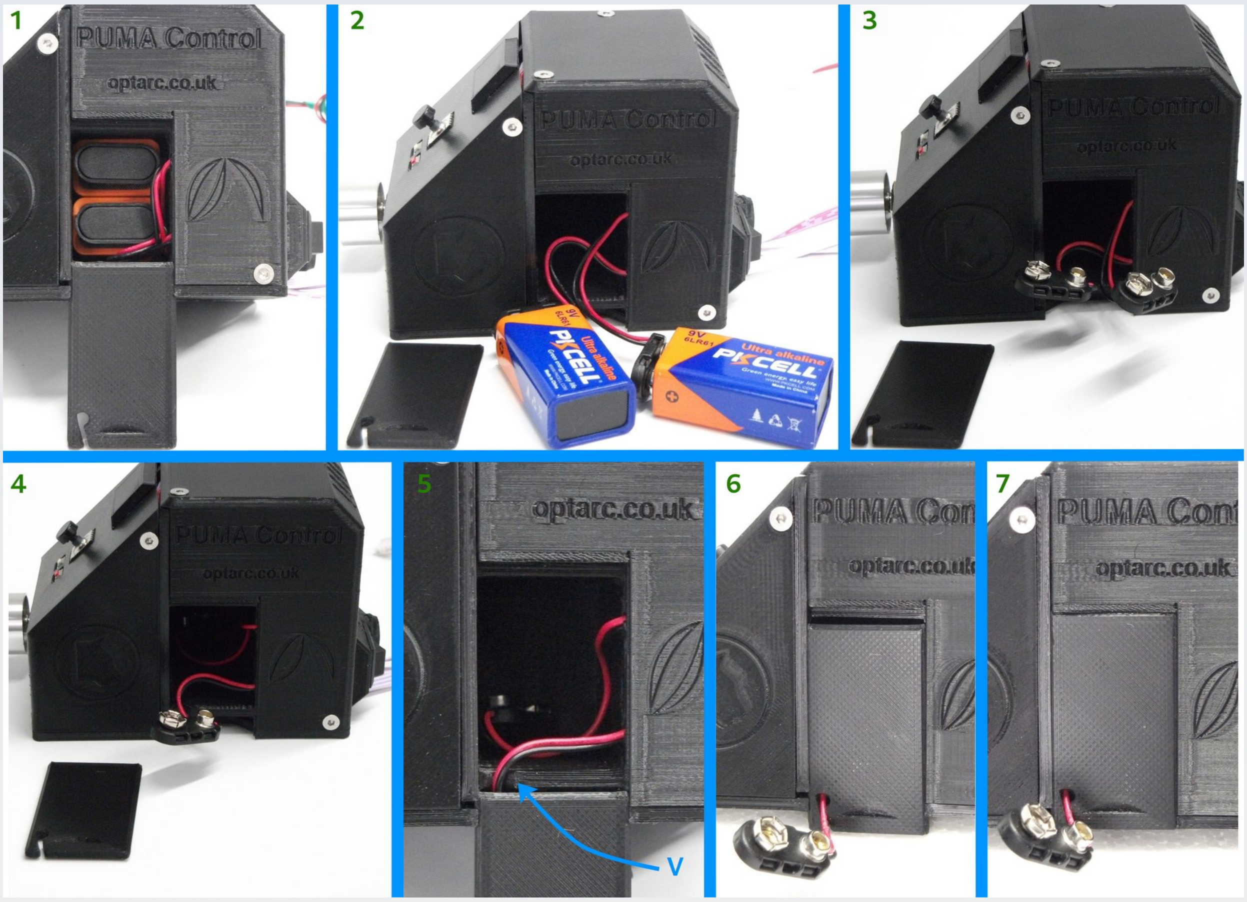


Figure 3. Externalising the power connector on the PCC. 1. Open the battery compartment. 2. Remove both batteries. 3. Disconnect both batteries. 4. Put the top battery connector back in the battery compartment. 5. Leave the lower connector out of the case and begin to slide the battery compartment cover back on making sure the externalised lead is in the 'V'-shaped groove in the casing (marked 'V' here). 6. Continue to slide the cover on and insert the externalised connector lead into the slot in the battery compartment cover. 7. Completely insert the battery compartment cover.

Tripping the power regulator – Immediate Action Required

Any cause of low voltage at the input to the internal power regulator of the PCC can cause the regulator to 'trip'. This means the regulator no longer regulates the output current properly which, in turn, could overload the LED bulb. This will usually be accompanied by an audible warning alarm of the piezo buzzer and a visual warning in the TFT display of the augmented reality projector (the current value display will turn red and flash). It is possible that such warnings may fail to occur and in that case you will notice the fault by noticing that the LED bulb appears much brighter than it normally does.

If this happens you should remedy the situation quickly by simply switching off the PCC, turning down the brightness potentiometer (turn it clockwise to do this) and then switch on again. **Failure to act quickly in such a situation could result in drastically reducing your LED lamp life or cause it to blow altogether.**

Situations that can result in tripping the power regulator are any which result in a low voltage at the input of the regulator. This is usually due to a low current power supply situation. This can occur either if your batteries are running low or you use an external mains supply that cannot provide over 1 Amp of current or you attempt to use the unit on only one 9V battery instead of 2. When using a power supply that is failing or has a low current rating, the 'trip' may not occur immediately when you switch on. Instead it may only occur when you try to move the Z motor or when you turn up the brightness with the potentiometer – then you may notice a trip as a sudden 'jump' in the brightness of the microscope LED lamp (if the above mentioned warning signals don't engage).

It is also possible that a false alarm high current signal may be generated as batteries wear low. This is due to the simple nature of the ammeter circuit and is a known limitation of the current software / hardware PUMA Control system. If this becomes an irritation you can disable the ammeter from the 'Defaults' menu but then you would loose this audible visual warning for when a real over-current situation occurs. If you use the PCC with the brightness potentiometer turned down this should not be a practical issue but it can be a problem if you use the PCC with the brightness potentiometer set at or near its maximum brightness setting. Use this ammeter disable option with caution.

Programming the PCC

The PCC has a USB port to enable the internal Arduino Nano to be interfaced to another device. This 'other device' will usually be a PC which is used to update or upgrade the software on the Nano or to read data from it.

IMPORTANT: Before connecting any device to the USB port of the PCC, ensure that the PCC is switched OFF from its main power switch and physically remove the LED lamp connector from the 5V output phono socket. Do not attempt to use the stage motor or plug in the LED lamp while the PCC is connected to an external device via its USB port. The only exception to this rule is if you have set up your own custom interface system with sufficient power distribution to handle the attached devices. Failure to heed this advice could permanently damage your attached PC (or whatever device the USB cable is attached to) because the devices in the PCC may attempt to draw too much current from the USB-connected device.

Once the PCC is switched off and microscope lamp disconnected from the phono socket at the back of the PCC, connect the PCC to the device used for programming (such as a PC running the Arduino IDE) with a standard IDE cable. You can then upload the new software as per the instructions for using the Arduino IDE. Once complete, remove the USB cable from the PCC, reconnect your lamp and switch on the PCC to use the new software.

The augmented reality (AR) projector and heads-up display (HUD)

PUMA provides for a digital overlay – called a 'heads-up display' (HUD) – of interactive information that is superimposed onto the live optical image of the specimen via a projector attachment module known as the augmented reality (AR) projector (figure 4).

Figure 4. The AR Projector module and the HUD. Main image shows the AR Projector (highlighted in the blue box) attached to a PUMA microscope. The inset lower right shows an example of the HUD as seen through the ocular lens with graphics (a grid and some numbers and symbols) superimposed on the live optical microscope image. In the rest of this manual, the HUD will be illustrated without an underlying microscope image for clarity of illustration except where demonstration images are needed to highlight certain features.

The larger end of the AR projector contains an ST7789 colour TFT display screen which connects to and is driven by the PCC. The rest of the AR projector consists in lenses that project the image of this TFT display onto the back focal plane of the ocular lens – the same plane that the objective lens of the microscope projects the image of the specimen on the stage. In this way the two images are superimposed to the viewer looking down the ocular lens(es) of the microscope.

The panels of the HUD

The details of what information is displayed will be discussed throughout this manual when discussing the various modes of operation of the PCC (known as 'control modes') but, in general, the HUD divides the round field of view (FOV) of the ocular lens into 5 regions, a central square region and 4 peripheral segment display panels outside the central square (figure 5). The peripheral green ring indicates the limits of the field of view of the HUD and is displayed on the boot up screen and in menu mode (only). It is intended to be used as a guide to the user when they set up the AR Projector so as to help centre the HUD in the field of view and ensure optimal magnification is achieved when adjusting the AR projector. Optimal magnification is achieved when the innermost part of the peripheral green ring is just visible all around the field of view when looking down the ocular lens.

Figure 5. The HUD. From left to right: The boot-up screen, an example menu screen, the 5 regions of the HUD, an example of user-customised colours for different information displays.

For many of these display areas the user may select the colour of the text and graphics via a setting in the GUI menu as will be described in more detail later – the default is white. If the colour 'Black' is selected then this will effectively make that part of the display invisible. In fact, that is the reason why 'black' is offered as an option – to give the user the ability to selectively 'switch off' specific aspects of the HUD.

The 5 main display regions are:

1. A central square area whose corners reach the limits of a 20 mm FOV
2. A left panel outside the central square area
3. A right panel outside the central square area
4. A top panel outside the central square area
5. A bottom panel outside the central square area

The following gives a summary of what information is shown in each region. Not all the information items for a given region will be shown all the time. The amount of information shown depends on user settings and on what control mode the user is currently in. Various symbols are briefly mentioned but not fully explained. Such detail will be given elsewhere in the chapters of this manual when the various control modes are described.

The central square area

This can show:

- Any grid, cross-hairs or scale bar patterns
- The interactive cursor / pointer*
- Any marker symbols laid down by the user*
- Lines drawn for the purpose of measurement*
- The menu options when in menu mode

* Items marked with an asterisk can actually be drawn / displayed anywhere on the TFT screen, including in any of the peripheral segment panels and even outside the round FOV of the ocular.

The left panel

This can show:

- The menu item selection arrowhead when in menu mode.

The right panel

This can show (from top to bottom):

- An integer representing the number of HUD pixels between divisions of any grid or cross-hair pattern or the length (in HUD pixels) of any scale bar. This number only shows when in Graticule control mode.

- The symbol for Z-motor control mode ( ) or Timer control mode ( ) when either of those modes are the currently active mode.
- The hours on the timer
- The minutes on the timer
- The seconds on the timer

The top panel

This has two lines of information.

The upper line can show (from left to right):

- An icon representing the Z-motor coil energisation state (Ø, ▲ or ▼).
- The coarse focus speed symbol (+).
- The ‘stage is homed’ symbol (H).
- The ‘sweep mode’ symbol (Σ).

The lower line can show (from left to right):

- The current position of the Z stage as the characters “Z: “ followed by an integer.
- A triplet of integer digits representing the lamp current.

The top panel also shows the menu level heading when in menu mode.

The bottom panel

This has two lines of information.

The upper line can show:

- A number (integer or floating point) representing the result of a measurement or the calibrated distance between divisions of a grid or cross-hair pattern or the calibrated length of a scale bar.

The lower line can show:

- The units of a measurement.

The Spatial Light Modulator (SLM)

In traditional microscopes the condenser’s aperture, and the aperture of the field stop in Köhler systems, are regulated by a mechanical iris that presents an essentially round stop the diameter of which can be varied by turning a dial. This very limited system of aperture control suffices for most routine observations. Some advanced scopes have the ability to block the central part of the condenser's aperture with a round spot of various diameters to allow for dark ground microscopy.

PUMA takes a different and more flexible approach. The PUMA systems use a filter-like tray containing an aperture pattern which is inserted into a slot at the base of the condenser (to control the condenser illumination) or a slot just beyond the lower collector to control the illuminated field of light.

This may at first seem like a retrograde step – a physical filter can only have a fixed pattern and can't be adjusted with a rotary control like an iris diaphragm. However, this is not so. The filter used with a PUMA scope can contain very complex patterns – even arbitrary shapes, so is not limited to a simple circular hole. Also, the PUMA aperture filter can have shades of grey or colour transparency so it not limited to all-or-nothing transmission like an iris diaphragm. While these complex pattern can be printed onto a fixed glass filter, a much more flexible approach is to use a transparent colour TFT computer screen as the filter to allow dynamic computational control over the aperture function. Such a dynamic light modulation system is known as a 'spatial light modulator' (SLM) and figure 6 shows an example of a manual filter and of the TFT-based SLM.

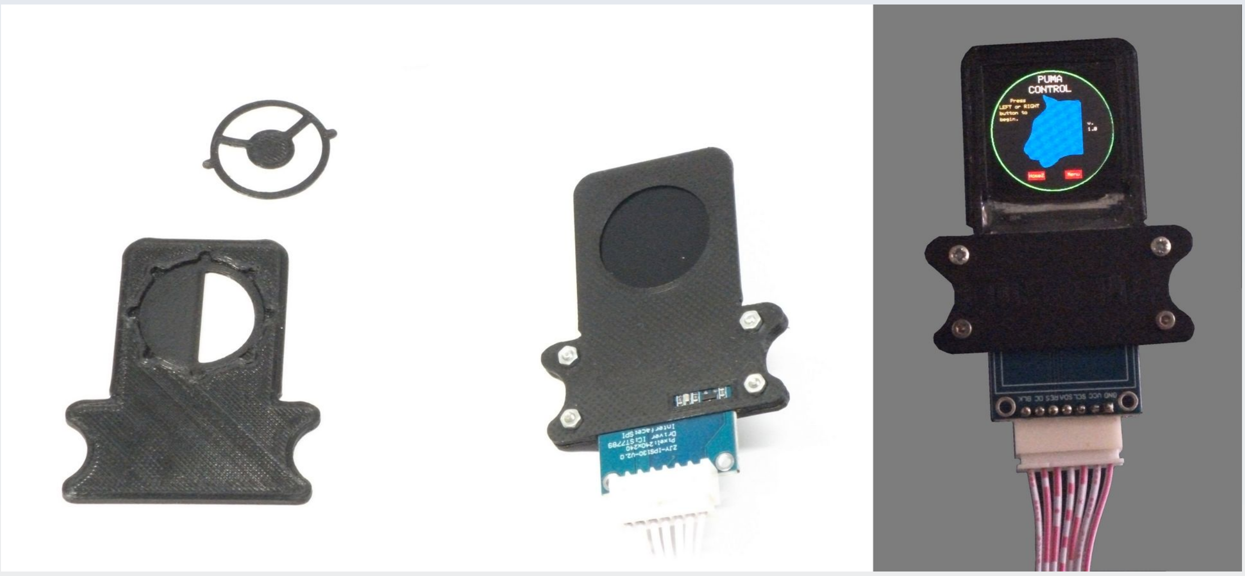


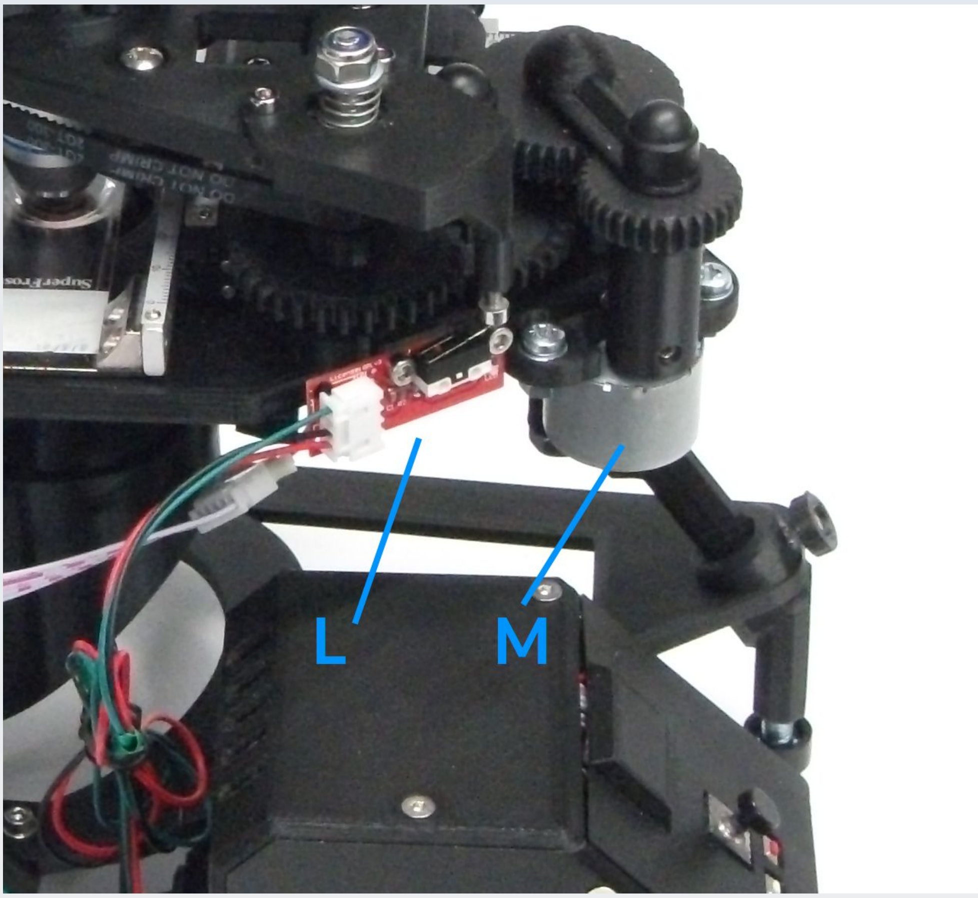
Figure 6. A manual aperture filter (left) and the PUMA SLM filter (middle – seen from the bottom aspect and switched off, right – seen from the top aspect and switched on showing the boot up screen). The Manual filter takes up to two filter discs simultaneously to built up complex patterns. It can also accept a round coverslip glass disc on which may be printed even more complex patterns with colour and semi-transparency but it will be a fixed aperture either way. The SLM uses a TFT screen so it can generate active and changing complex apertures. Both fit into the same filter slot so they are interchangeable. The picture on the far right shows the SLM in front of a bright light which is how the image is made visible – there is no intrinsic backlight or reflective panel in the SLM.

The apertures presented on the SLM can be controlled by the PCC but the amount of control allowed in the default PUMA Control software is very limited compared to the full potential of what can be achieved with this device. The SLM is an advanced feature of the PUMA microscope and it is anticipated that those wishing to use it will write their own interface software to achieve the type of transmission functions they require for their specific purposes.

The Z-motor and Z-limit switch

The stage of a PUMA microscope can be fitted with a stepper motor to enable motorised focus control. The type of motor used in the current PUMA standard is a 12 Volt 28BYJ48 motor. As a safety feature and also to allow open loop monitoring of the Z position of the stage at any time, an end stop limit switch is also part of the PUMA stage motorisation kit. These apparatus can be seen in figure 7.

Figure 7. The PUMA stage fitted with a stepper motor (M) and limit switch (L) to allow motorised control over focus. The PCC to which they are connected is shown at the bottom of the figure.



The PCC is designed to connect to and drive the stage motor and to monitor the limit switch state. Details will be provided in the dedicated section of this manual.

User input via the joystick keypad

The joystick keypad on the front panel of the PCC was introduced above and illustrated in figure 1. This section gives detailed information on the types of input actions (sometimes called 'key actions') a user can make with this keypad and how these input actions are referred to in the rest of this user manual.

The joystick

The joystick has 5 primary activation actions. The user can also use simultaneous combinations of these 5 actions to effect additional signals to the PCC.

The joystick can be moved in any of the 4 primary directions (up, down, left, right) and can also be clicked directly inwards (hence '5-way'). It can also be moved in the 4 diagonal directions e.g. you can move it up and left diagonally and it will activate both the 'up' and 'left' signals simultaneously – the PUMA control software can detect and respond to such dual simultaneous signals. However, these diagonal joystick movements are only used for moving the cursor around the screen in the 'Pointer' control mode so are not given specific abbreviations in the scheme described below.

The left and right (set and reset) buttons

The two separate buttons below the joystick are called 'Set' for the left button (the white button in figure 1) and 'Reset' for the right button (the red button in figure 1). This set/reset terminology is used in this manual instead of left/right to make it clear we are talking about the these buttons and not joystick direction movements. However, the written message on the boot up screen of PUMA Control says 'Press LEFT or RIGHT button to begin' (using the word 'button' to distinguish them from joystick actions).

Note that the PCC Arduino can be physically reset (a hard reset) using its own reset button i.e. the reset button that is on the Arduino Nano board inside the PCC case. This can be accessed, e.g. with a match stick, via the hole in the back of the PCC case labelled 'Reset' (see figure 1, right). ***This internal 'Reset' button has nothing to do with the 'Reset' / right button on the front keypad and for the rest of this manual whenever a 'Reset' button or reset action is referred to it only refers to the right button on the joystick keypad on the front panel*** of the PCC.

How to issue a combination key action signal

The 'Set' button can be clicked on its own or in combination with some other button. In this section 'some other button' includes a joystick action as well as the 'Reset' button. The PCC can detect such combination presses and act accordingly. However a combination press must be done in a certain way to be properly recognised.

The correct way to do a combination press is to press the 'Set' button first, then press the other button while the 'Set' button remains depressed. You then release both buttons simultaneously. The exact order of release is not important but it should be approximately simultaneous because if you linger on one or other button then that will register as a second, single, button press.

In order for the PCC to distinguish between a 'Set' only press and a 'Set' + 'Other' combination press, the program waits for a maximum of 500 ms when it detects the 'Set' button first being pressed to see if a second button is going to be pressed as well. If no other button is pressed in this half-second interval then the PCC interprets the action as a solitary 'Set' press and acts accordingly. So you must press the other button within half a second of first depressing the 'Set' button to register a combination press signal.

Note that there is no valid combination signal that comes from clicking any button first except the 'Set' button with the sole exception of the diagonal joystick movements in Pointer control mode described above.

List of allowed input signals from the joystick keypad

In the rest of this manual the following symbolism / terminology is used to mean the following actions:

JS_UP : Move the joystick up

JS_DOWN : Move the joystick down

JS_LEFT : Move the joystick left

JS_RIGHT : Move the joystick right

JS_MID : Click the joystick in (towards the control panel)

SET_BTN :

- SET : Click the Set button on its own
- SETRST : Click the Set button in combination with the Reset button
- SETDOWN : Click the Set button in combination with JS_DOWN
- SETUP : Click the Set button in combination with JS_UP
- SETMID : Click the Set button in combination with JS_MID
- SETLEFT : Click the Set button in combination with JS_LEFT
- SETRIGHT : Click the Set button in combination with JS_RIGHT

RST_BTN : Click the Reset button

As indicated earlier, there is no valid 'RSTSET' action – i.e. there is no valid combination signal that arises from pressing the 'Reset' button first and then the 'Set' button while the 'Reset' button is still pressed. If you try this you will just trigger the action that occurs when pressing the 'Reset' button on its own. Likewise there is no valid combination signal that arises by pressing 'Reset' first in combination with any joystick action. Although such combinations are technically possible, the PUMA Control software is not programmed to respond to them and will react as if 'Reset' were pressed on its own.

Using the above system of single and combined activations, including the diagonal joystick movements for the pointer, the PCC can use this very limited joystick keypad to recognise 17 different input signals from the user.

Control modes

PUMA Control allows various functional aspects of the system to be controlled by the joystick keypad such as focussing the motor, making measurements, etc. Because the single joystick keypad is too limited an input device to allow simultaneous control of all these many features at once, the user selects which functional aspect of the system they want to work with via a menu and then the Joystick keypad will have particular input functions suited to controlling that aspect of the system. Each of these These 'functional aspects of the system' is called a '**control mode**' (CM). There are a total of 6 control modes which will be described in detail in their dedicated sections in this manual. The control modes are:

1. **Z-Motor** : Allows control over the stage focussing motor with various options. Focussing is such a basic aspect of using a microscope that the Z-Motor control mode is the default control mode after boot up and there is also a shortcut key action that allows the user to quickly toggle between any other control mode and the Z-Motor control mode without having to go through the main menu. So a user can easily focus the scope while working, say, with a graticule in Graticule control mode without having to completely leave the graticule mode or remove the graticule from the main display area.

2. **Pointer** : This allows control of an interactive pointer / cursor on the HUD which can be used for pointing, marking objects, counting objects and drawing lines to be measured in length and angle.

3. **Timer** : Set and use a count-up or count-down timer. The time will continue to count time even when you move to a different control mode so it can be used

simultaneously with other modes. Selected quick link shortcuts are also provided to this mode but not as extensively as with the Z-Motor control mode shortcut. A running timer is the only display which persists when the user switches to the menu.

4. **Graticule** : This allows you to superimpose a range of graticules, cross-hairs and scale bars on the field of view via the HUD. Each graticule can be varied in scale and colour.
5. **Plain** : This sets a completely plain field of view on the HUD which can be varied in colour. Use this for illumination effects. When used with the SLM this can act as a programmable colour filter to colour the illumination of the scope.
6. **Aperture** : This allows apertures of various sizes and colours to be generated for use with the SLM. You can cycle between a plain bright field circular aperture (and open and close it in graded steps like an iris), a half-field Schlieren aperture for phase contrast (you can rotate it by 90 degrees as desired) and a central stop aperture of various sizes for dark field microscopy. You can also superimpose the dark field central spot onto either of the two preceding patterns. By varying the colours of the different components you can create Rheinberg filter illumination effects

Software Installation Notes

PUMA Control is an Arduino sketch (.ino file) intended for use on the Arduino Nano v. 3.0 to be compiled and installed using the Arduino IDE v. 1.8.15 or compatible.

It uses a modified version of the Adafruit GFX library where the default font has been customised to draw mirror writing and also to draw custom graphics characters. This custom version is named GFXM and is found in the 'libraries' folder of the PUMA_Control repository in the folder called GFXM_Library. Ensure that the folder 'GFXM_Library' is placed in your Arduino libraries folder as shown below. This customised library is kept as a separate re-named library so that it does not get overwritten each time the standard Adafruit graphics library is updated. This GFXM_Library folder also contains a customised version of the standard Arduino library Print.cpp file which ensures that all writing is right to left - i.e. mirrored.

PUMA Control also uses a slightly modified version of Pawel A. Hernik's "Arduino_ST7789_Fast" library which is called 'ST7789_FastM'. This should be placed in your Arduino libraries folder as shown below.

Make sure you have these modified libraries (all of which are included in the PUMA GitHub distribution) in the appropriate folder for your version of the Arduino IDE to use when compiling and uploading PUMA Control to your Arduino Nano (inside the PCC).

For example, on a Linux OS you may have the following directory structure:

~/Arduino/libraries/ST7789_FastM

~/Arduino/libraries/GFXM_Library

~/Arduino/MySketches/PUMA_Control/PUMA_Control.ino

The Meaning of Audible Signals

PUMA Control has two sources of feedback for the user: Visual feedback via the TFT display (of either the SLM or the AR Projector HUD) and audio feedback via the piezo buzzer in the PCC.

There are 6 characteristic audio feedback signals have been carefully designed to be as helpful and meaningful as possible because not everyone will be using the PCC with a TFT screen device (e.g. the PCC may be used to control of the Z-motor and lamp brightness only).

This chapter describe these signals and what they mean. Each characteristic signal is given a name. These are the names of the audible feedback signals:

SILENCE

ACKNOWLEDGED

PROMPT

FORBIDDEN

TRILL

TIMEOUT

LAMPHOT

In the following descriptions a ‘key action’ or ‘key activation’ refers to any of the button or joystick actions using the joystick keypad of the PCC.

SILENCE

A special case override to the ACKNOWLEDGED signal for certain key actions. Specifically when moving the pointer around or moving focus up or down with the joystick, the ACKNOWLEDGED signal is replaced by SILENCE i.e. no sound and no delay in the loop function that looks for key actions.

ACKNOWLEDGED

This is a single beep. It acknowledges a key action. The total amount of delay introduced by any buzzer signal must be at least 200 ms (regardless of how long the buzzer sounds for) in order to minimise the risk of inadvertent double-key activation.

PROMPT

This is a double beep. It brings the user's attention to a normal state or indicates that they are being prompted to make a decision. For example when boot up is complete (so the system is ready to accept key commands) or when a confirmation question requiring a response is being asked by the system (e.g. 'Do you really want to delete all data points?'). The PROMPT signal is the PCC saying ‘it’s over to you. Your move.’

FORBIDDEN

This is a triple fast beep. It brings the user's attention to an inability to perform the desired action (due, e.g. to a limit having been reached or a condition not met). For example this will occur if a user tries to access motor functions if the stage has not been homed or if a min/max motion limit is reached. It will also sound if you try to undo a drawing mark when there are no more marks left to undo, etc.

TRILL

This is essentially a very fast triple beep. It means the system is entering silent mode (i.e. the mute setting). This is the only meaning for this signal.

TIMEOUT

This is 6 beeps in a row. It means the end of a countdown has been reached (i.e. the audible signal for the timer). This will override the mute setting if the user has requested an audible signal for a count-down (either alone or in combination with a visible signal).

LAMPHOT

This is 4 beeps in a row. It means the lamp is being over-driven with too much current and may be destroyed if immediate action is not taken. This will override the user's mute preference because it is a safety warning. This signal will repeat itself continually until the current is reduced to safe limits. There will be a brief gap between repeats so the user can distinguish it is 4 beeps recurring and not a continuous train of beeps.

Notes

Sometimes a PROMPT or FORBIDDEN signal will be immediately followed by the single beep of ACKNOWLEDGED, at other times it won't. There will be a brief pause between signals. The user may need to get familiar with these signals to appreciate this.

The Boot Up Screen

With all hardware properly connected and lamp brightness potentiometer turned down (or at least to its half way point) switch on the power button on the PCC.

Figure 8. The Boot up screen. The written message says 'Press LEFT or RIGHT button.' (using the word 'button' to distinguish them from joystick actions). However, this manual uses the terms 'Set' for the left button and 'Reset' for the right button. Note that the two buttons are represented as red rectangles at the bottom of the boot up screen with their respective actions written on them in yellow text. The number just to the right of the 'PUMA CONTROL' title is the ammeter reading – in this case '080' indicating that the current drawn by the microscope LED lamp is 80 mA. If no lamp is connected to the PCC this number may not appear (as was the case in figure 5, left). The software version is shown mid-right.

After a few seconds the PROMPT audio signal occurs (see page 28) and the boot up screen appears to let you know the program is ready to use (figure 8). The microscope lamp will turn on immediately – there will be no delay and no need to wait for the boot up because the lamp is powered independently of the software.

In the boot up screen you have 3 options:

1. Press the left button (i.e. 'SET') to initiate a Z-Homing procedure (the Z-stage motor must be homed before you can use it but you are free to use the rest of the software functions without homing the motor - e.g. if you do not have a motorised stage or if you just want to use manual focus). See the section on the Z-Motor control mode for details of the Z-homing procedure.

2. Press the right button (i.e. 'RST_BTN') to take you to the main menu to select a control mode.

3. Press the central joystick button (i.e. 'JS_MID') to mute the device. You will hear the TRILL signal indicating that the device has gone into mute. Press the central joystick button again to undo mute (there will be no audible signal for this – not even the key press ACKNOWLEDGED beep – because the system was in mute when you pressed the JS_MID to re-enable the audio).

Performing any other key action in the boot up screen will have no effect. You will not see the boot up screen again until you next switch the PCC on.

The Menu

The menu allows you to select the mode of operation (called the 'control mode') and set various preferences and settings.

The menu has a maximum of two levels: The main menu and one submenu for each control mode and for some global default settings.

Each menu has a number of '**items**' you can select. Each item has a '**value**' you can change. If a value of an item is the string '[…]' then this means that if you select that item you will be taken into a submenu.

General navigation

You enter the main menu from any control mode by clicking the 'Reset' button.

You navigate the menu with the joystick.

If you have entered a sub-menu, use the 'Reset' button to take you back to the main menu.

If you are already in the main menu the 'Reset' button will take you out of menu mode and into the currently active control mode.

Item select mode

When you first enter a menu you will be in 'item select mode' which enables you to select which menu item to work on (figure 9). This mode is recognised by the fact that there is a white arrowhead on the left HUD panel (pointing to a menu item) but NO green arrow heads surrounding the value of any menu item.

In 'item select mode' you move the arrow head up or down with the joystick till it points to the menu item you want to activate or edit. The arrow cycles around i.e. when it is at the top item, if you move the joystick upwards, the arrow will appear at the bottom-most item and *vice versa*.

When you position the arrowhead at the menu item you want to work on perform a JS_RIGHT action to select the value of that menu item for editing. If the item value shows as '[…]' then selecting that item will take you into a sub-menu and you will be in 'item select mode' for that sub menu.

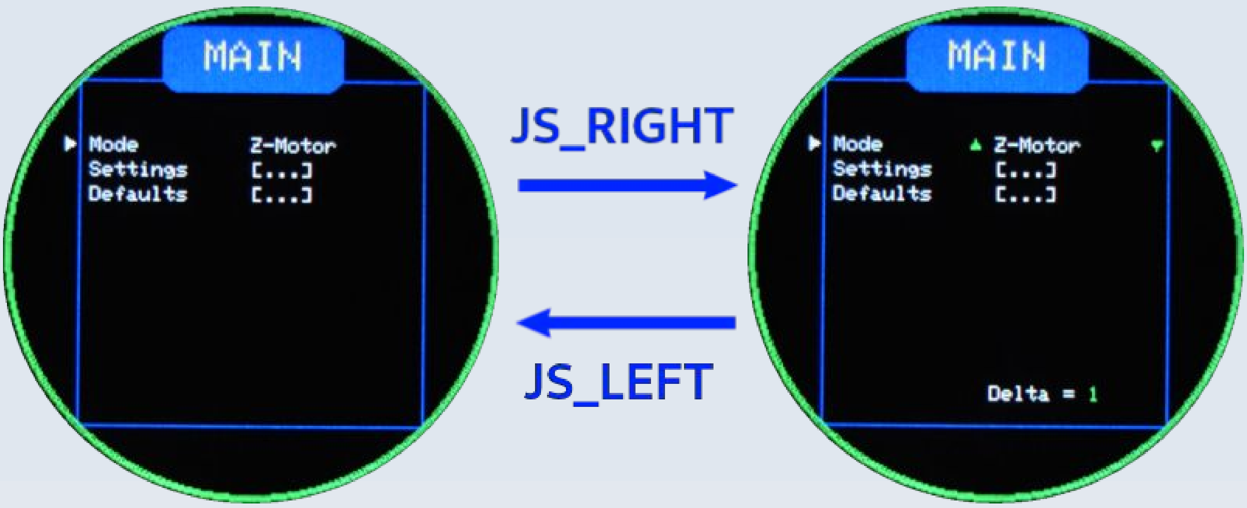


Figure 9. Menu modes and switching between them. Left: '*Item select mode*' on the main menu. The arrowhead is pointing to the first item called 'Mode'. You can move the arrowhead up and down to select a different item using the joystick up/down moves. If you want to edit the selected item then do a JS_RIGHT move on the joystick to enter '*value editing mode*' on that item. Right: Value editing mode on the selected item. This is signified by the appearance of green up/down arrowheads surrounding the item's value, letting you know that you can alter that value my means of up/down movements of the joystick. The appearance of the 'Delta' value below also shows that you are in value editing mode but has no effect when editing non-numerical menu values. To get out of value editing mode back into item select mode just do a JS_LEFT move on the joystick (or press the 'Reset' button to go back a level or exit the menu altogether if you are in the main menu).

Value editing mode

When you select a menu item (whose value is not '[...]') with a JS_RIGHT action then you will see green up/down arrowheads surrounding the selected item's value field and you will be in 'value editing mode' (figure 9). When in value editing mode, the value can be changed by moving the joystick up or down. Also, when you are in 'value editing mode' you will notice, near the bottom of the main square region of the display, the text 'Delta =' in white followed by some integer in green. This delta value is used when entering numbers – more on that shortly.

You can leave value editing mode by one of two ways:

1. Perform a JS_LEFT move on the joystick to take you back to item select mode in the current menu.
2. Press the 'Reset' button to go back one menu level. If you are in a sub-menu this will take you to the main menu (and you will be in in item select mode) or if you are already in the main menu this will take you out of menu mode altogether and into the current control mode.

Editing a menu item value: Strings and symbols

There are two types of value: a fixed list (of strings or symbols) and an integer number.

For fixed lists of text strings or symbols moving the joystick up or down will take you through the options in succession. In some cases the list will cycle around, in others you can't cycle. Once the option you want is displayed simply leave value editing mode and the displayed option will be used as the new value for that menu item.

Editing a menu item value: Numbers

For numbers you can also increment or decrement the value with the joystick (up or down) but you also have the additional control of how much to increment or decrement them by using the 'Delta' value mentioned above. The Delta value tells you how much you will change the current value by with each JS_UP or JS_DOWN key action. Delta can be any one of the following values:

1

10

100

1000

10000

You cycle through these options by clicking the 'Select' button while in value editing mode (the 'Select' button does nothing when you are in item select mode). Delta can be changed when a non-numerical fixed list value is selected for editing but Delta has no effect on non-numerical values.

All numerical values will have limits you cannot go beyond. The limit will be dictated by the nature of the value. For example, the 'Backlight' value can only be between 1 and 255 inclusive.

You can only enter integer values with this method but there is a facility to enter a floating point number by considering the parts either side of the decimal point as separate integers and changing them separately. This is done when entering the calibration factor in the Defaults submenu (see below).

No negative numbers can be input (and none are required) in the current version of PUMA Control.

Editing a menu item value: Colours

Many menu options allow you to select a colour for a particular feature. You may chose from any one of the following: Black, White, Custom, Red, Green, Blue, Yellow, Magenta, Cyan.

The choice is made by moving the joystick up or down to scroll through the list.

Colours are defined by a 16 bit unsigned integer using the Adafruit GFX library colour convention where the least significant 5 bits represent the blue component, the most significant 5 bits represent the red component and the intervening 6 bits are for the green component. A colour therefore has a numerical value between 0 (black) and 65535 (white)

The fixed colour choices with their values are defined as follows (hexadecimal equivalent given in brackets):

BLACK	0	(0x0000)
BLUE	31	(0x001F)
RED	63488	(0xF800)
GREEN	2016	(0x07E0)
CYAN	2047	(0x07FF)
MAGENTA	63519	(0xF81F)
YELLOW	65504	(0xFFE0)
WHITE	65535	(0xFFFF)

The value of the custom colour defaults to 39321 (0x9999) on startup and you can change it to any number (i.e. colour) you like via the 'Defaults' menu (see page 35). However, the program will not allow the custom colour to be equal to one of the fixed colours so the lowest value you can chose is 1 (i.e. 1 more than pure black), the highest value you can chose is 65534 (i.e. 1 less than pure white) and if any other value you chose equals one of the fixed colour values shown above then your chosen value will automatically be incremented by 1.

Selecting multiple custom colours

Each menu item that has a colour property has its own internal colour variable which is separate from the global 'custom colour' variable. When you select 'Custom' as the colour option for any menu item, the current value of the system's global custom colour will be copied into the local colour property of that menu item. That menu item's colour property will not automatically update to a new custom colour value when the system's global custom colour value is changed by the user via the Defaults menu. If you want to update a menu item's custom colour property to the latest one defined by the user then you must go to that menu item and change the colour property away from 'Custom' to some other colour and then back to 'Custom' (or cycle the colour outside the menu using the keypad interface for items that allow colours to be changed outside the menu – see for example page 63).

Although at first sight this may seem to be a limitation, it is in fact a deliberate design choice because this method enables multiple custom colours to be used simultaneously even though only one global custom colour value can be defined at any one time.

As an example of how to do this I will discuss multi-custom-coloured apertures (see page 66 for detail about the Aperture menu and settings). Say you want 3 different custom colours for an aperture: clear aperture colour, background colour and central spot colour. First define the global custom colour value by setting the 'Custom__col' item in the Defaults menu then go to the Aperture menu and select 'Custom' as the colour for your clear aperture colour. Now go back to Defaults and change the value of 'Custom__col' to your next custom colour value. Go to the Aperture menu and select 'Custom' for your background colour. This will only change the background colour value to the new custom colour, the clear aperture colour will retain the previous custom value. Finally go back to Defaults and change the 'Custom__col' value to your third custom choice then go back to the Aperture menu and select 'Custom' as the colour for your central spot – this will only affect the central spot colour. So now you have loaded 3 different custom colours into your aperture settings even though you can only select one global 'Custom__col' value via the Defaults menu.

The Main menu

The main menu has three items: Mode, Settings, Defaults (figure 9).

Mode

Select this item to change the current control mode to one of the following: Z-motor, Pointer, Timer, Graticule, Plain, Aperture.

Details of what each of these modes do will be described elsewhere in this manual. For now note that each of these options has at least one item in a submenu that you can edit. This is the 'Settings' submenu and each control mode has its own 'Settings' submenu.

You access the 'Settings' sub-menu for any control mode in one of two ways:

Simply do a JS_MID while you are in the Mode value. This will take you directly to the settings submenu for that mode.

The 'long way round' is to first do a JS_LEFT to get back to the main menu selection, then do a JS_DOWN to select the 'Settings' item on the main menu then do a JS_RIGHT.

Defaults sub menu

The Defaults submenu has the following items (figure 10):

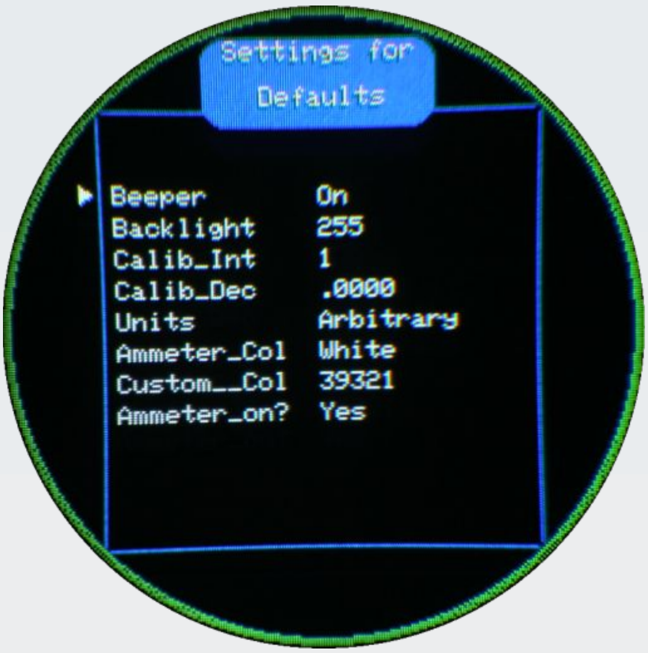


Figure 10. The Defaults sub-menu

Beeper Value options: On, Off

This allows you to toggle the PCC in and out of mute mode. In mute no audio feedback will be provided with two override exceptions:

1. If the user has select an audio signal (including combined audio and visual signal) to mark the end of a count-down then that audio signal will be given even if the system is in mute mode.
2. If the lamp current exceed the upper limit then the audio alarm LAMPHOT will be sounded even if the system is in mute mode.

Backlight Value options: Integer between 1 and 255 inclusive

The brightness of the TFT screen backlight can be adjusted here. This is only relevant to the AR Projector for the HUD because the SLM TFT has no backlight. The dimmest value is 1 and the brightest is 255. You cannot set a value of 0 (backlight off) partly because that would make the menu invisible but also because you can switch off the backlight in other ways while in certain control modes.

Calib_Int Value options: Integer between 0 and 65535 inclusive

The integer part of the calibration factor. See 'Calibrating the HUD' (page 38) for more details.

Calib_Dec Value options: Integer between 0 and 9999 inclusive

The decimal fractional part of the calibration factor. Together with the Calib_Int this forms the floating point number used to calibrate measurements made with the HUD to real world units (the units specified in the next menu item). See 'Calibrating the HUD' (page 38) for more details.

Units Value options: Arbitrary, Millimetres, Nanometres, micrometres

The real world units the HUD is calibrated to by means of the calibration factor. See 'Calibrating the HUD' (page 38) for more details.

Ammeter_Col Value options: see page 33

The colour used for the lamp current meter reading. The default is white. If you select black then the ammeter reading will be invisible but this (and any other colour choice) will be over-ridden if the lamp current exceeds the upper limit. In that situation the lamp meter reading will appear in red (you don't have the option to use red as a standard colour for this).

Custom__Col Value options: An integer between 1 and 65534

The value of the colour to use when a user selects 'Custom' as a colour option for any other colour setting in a menu. In order for the program not to get confused when selecting between colours in internal switch loops, the custom colour value must be different to any of the fixed colours. However, you do not need to consciously worry about this because if you select a value that is equal to one of the fixed colours (see page 33) the program will automatically increment your selected value by 1 (also the range limit 1 to 65534 prohibits you from selecting pure black or pure white).

When you change this value in the Defaults menu it only changes the value of the current global custom colour variable – it will NOT change the colour of any other menu items that you previously set to use 'Custom' as a colour choice. They will retain the custom colour that was in effect the last time you set them to use 'Custom'. This is useful because it means you can set multiple menu items to use multiple different custom colours simultaneously even though you can only have one global 'Custom__col' value set here in the Defaults menu. To see an example of how to use multiple custom colours see page 33.

To to update a menu item that uses 'Custom' as it s colour you must 'alter' that menu option value again (even though that 'alteration' just means re-selecting 'Custom' again from the list of available colour options). While in value editing mode for that menu item you must first use the joystick to move the selection away from 'Custom' to one of the other fixed colours, then move back to 'Custom'. This action will 're-load' the current custom colour setting into that option and it will be used for that item. For items that can have their colour changed interactively outside the menu system by using the keypad (for example, cycling the grid colour as described on page 63), the new Custom value will be loaded the next time the Custom option is selected via this method.

Ammeter_on?   Value options: Yes or No

Toggle between Yes and No using the joystick up/down. This determines whether the PCC monitors the current flowing to the microscopy lamp or not. It is on ('Yes') by default. Turning the ammeter off will mean that you will have no audiovisual warning if the microscope lamp is being over-driven (and so may burnout). The option to turn this off is provided because it is known that the ammeter system used by the PCC is unreliable for battery operation in that as time goes by and the batteries wear down the ammeter current reading rises until eventually a false over-current warning alarm is triggered. This will not happen when an external mains adapter power supply is used. Thus, in all cases where a mains adapter is used it is strongly advised to keep this setting as 'Yes'. See the section on 'Tripping the power regulator' on page 15 for more details.

Calibrating the HUD

Measurements of lengths (including boundary lengths) of microscope specimens can be made using HUD with the Pointer and Graticule control mode features described in their respective chapters. This chapter discusses how to calibrate the HUD so those measurements can be expressed in real world units.

The native pixel resolution of the HUD

The HUD is generated by the AR projector which projects and superimposes an image of an ST7789 TFT computer screen onto the back focal plane of the ocular lens of the microscope. Thus what you see in the HUD are the pixels of this TFT screen.

The TFT screen is composed of 240 pixels horizontally and 240 pixels vertically as a square array but the round field of view of the ocular lens means that you will not see all these pixels, the pixels in the corner regions being out of view. The HUD was designed to operate with oculars that have a 20 mm field of view – that means the image in the back focal plane of the ocular lens is 20 mm in actual diameter (physically there is a 20 mm diameter plane area in the barrel of the ocular that is visible to the observer). The AR projector was designed to project the image of the TFT screen to fill this area in X and Y omitting the corners of the TFT screen in such as way that all the pixels just inside the green circle seen in the boot up and menu displays fits in the field of view of the ocular lens.

So, depending on the actual arrangement of lenses in the AR projector (which can be adjusted by the user when setting up the scope) the visible HUD will consist in a circular area of pixels in a grid, 240 pixels in diameter. All lines drawn and all marks made by the user on the HUD will be made on this pixel grid.

Measurement units

By default, when you first switch on the PCC, all measurement values are given in arbitrary units that represent HUD pixels – so if a line is stated to be 7.35 xm long it means that the distance between the first and last pixel of that line is 7.35 HUD pixels.

Measurement results and units appear in the bottom panel of the HUD in the ‘Pointer’ and ‘Graticule’ control modes (which see for further details). The units of measurement are selected via the ‘Defaults’ sub-menu (see page 35) and when displayed on the HUD the units are represented by two letters, the last one being ‘m’ (for metres). The full list of options and their HUD display abbreviations are:

Arbitrary	xm
Millimetres	mm
Nanometres	nm
micrometres	µm

The default is ‘xm’ but the user can select any of the above options even if they don’t calibrate the scope as described below. Merely selecting a unit of measurement does not automatically make the scope calibrated to that measurement. It is the responsibility of the user to ensure an appropriate calibration factor is entered to make these measurement units mean what they say.

The arbitrary (xm) option is provided so a user can use whatever calibration unit they like and are not restricted to the metric choices above but the user will have to make a note of what ‘xm’ means to them according to their own calibration (for example ‘xm’ could mean ‘hundredths of an inch’ if, for some reason, you wanted to calibrate the scope to that unit).

The need for calibration and factors affecting calibration

As stated above, the initial default at boot up is that all measurement values are given in arbitrary units that represent HUD pixels. However, the size of these pixels have no standard or fixed relationship to the actual size of objects or the real distances between points seen down the microscope because that will depend on the combination of the magnification of the object produced by the objective lens and the precise magnification of the TFT screen produced by the AR projector. So the relationship between HUD pixels and actual observed distances in some real world measurement unit needs to be worked out by a calibration procedure.

It follows that every time you change the objective or change the magnification of the HUD TFT projection image you will need to alter the calibration factor to one that is appropriate for that combination of lens arrangements. Changing the magnification of the TFT screen projection image will occur whenever you alter the distances between the AR projector lenses and its TFT screen by making adjustments to it and also by moving the ocular lens up or down and re-focussing the AR projector image on the new ocular position. Make sure you are aware of this – calibration factors are not just objective-specific.

How to calculate a calibration factor

The calibration factor is the number of real world units per single HUD pixel (e.g. the number of micrometres per HUD pixel).

To calculate a calibration factor, first set up your microscope and AR projector: select and install your objective, focus an image of a test specimen with it and ensure your AR projector is co-focussed and centred and ready to use. Do not change the arrangement of any lenses after doing this (with the sole exception of normal focussing of the stage). The calibration factor you calculate will only be valid for this particular arrangement of objective, ocular and AR projector.

Now place an object of know dimensions on the microscope stage. The most conventional way to do this is with a stage micrometer – i.e. a slide that has a 1 mm mark laser etched onto it with divisions of known length (as in figure 11).

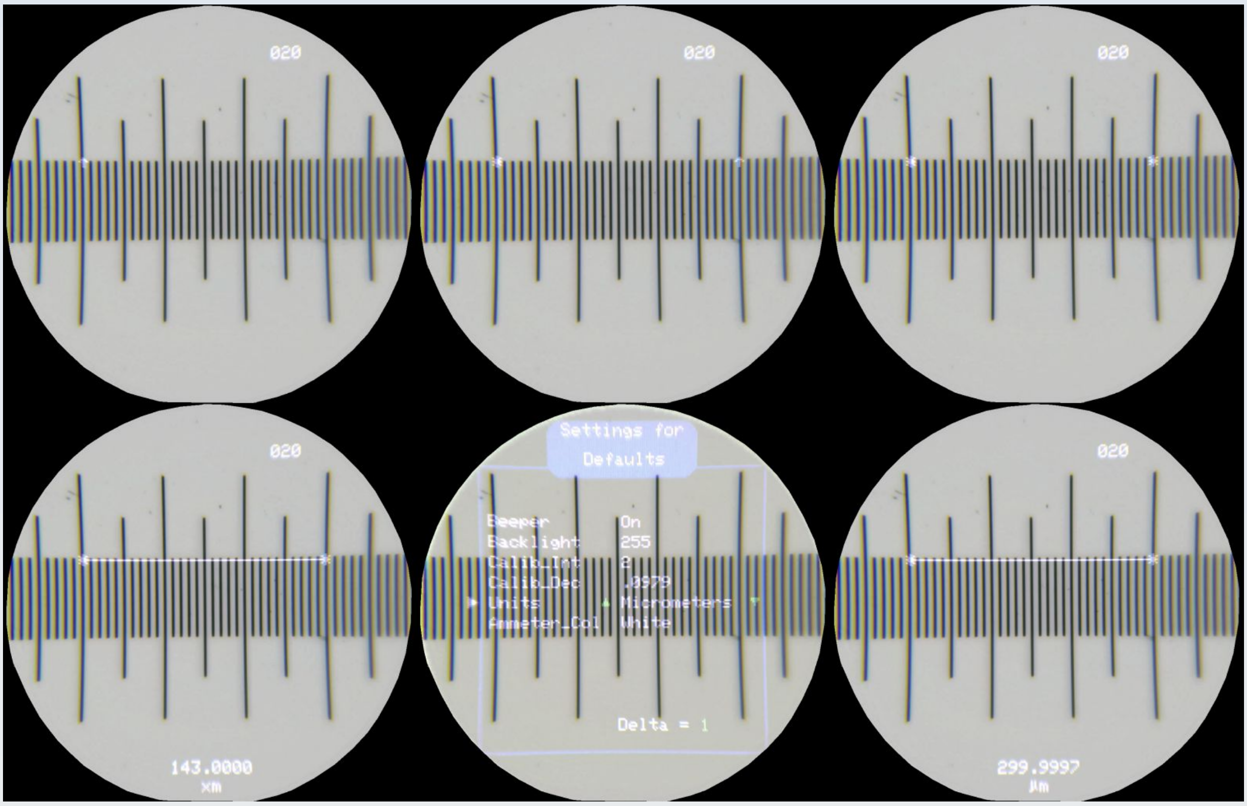


Figure 11. Calibrating the HUD with a stage micrometer which has 1 mm divided into 100 lines so each division line = 10 micrometres. This is a full field of view image through the ocular lens of a PUMA microscope set up with a x40 objective lens and the AR HUD. The procedure from top left through to bottom right is as described in the text. Top row: Left – HUD is in Pointer CM and the cursor (an up-arrow) is placed at one of the divisions of the stage micrometer. Middle – a mark was deposited at the last position of the cursor and it has now been moved to another division some distance away. Right – a mark is deposited there. Bottom row: Left – A line measurement is taken between the two marks and this shows the line to be 143 HUD pixels long. Middle – the calculated calibration factor (2.0979, see text) is entered into the Calib_Int and Calib_Dec items of the Defaults menu and the Units are edited to 'micrometres'. Right – now when the user goes back to the Pointer CM they see that both the measurement value and units have been updated showing a value of 299.9997 micrometres (which is essentially correct because the line spans 30 divisions of the stage micrometer ruler, each division being 10 micrometres long). The number '020' in the top right of the HUD is the lamp ammeter reading (20 mA).

Select the Pointer control mode and draw a straight line on the HUD between two points on this known test object (see page 55 for information on how to do this). Note the length of this line in both real world units (from your knowledge of the test object, call this value X micrometres) and in HUD pixels obtained from the measurement function of the HUD (call this value Y pixels). As the default units used for measurements are HUD pixels Y is just the value of the measurement as shown on the HUD (unless the HUD was previously calibrated in which case you would need to change the calibration back to HUD pixels first by setting the calibration factor to 1.0 or reboot the PCC which will have the same effect). In the example shown in figure 11, Y = 143 HUD pixels.

In the example shown in figure 11, X is 300 micrometres because the line was draw to span 30 divisions of the stage micrometer and each division is 10 micrometres. Note that in figure 11 both the stage micrometer and the drawn line are completely horizontal. This is not necessary – what is necessary is that the drawn line be parallel to the stage micrometer. Both could be at 45 degrees for example – or any other angle so long as they are equal. The pixels in the HUD TFT are square and it is assumed that the microscope optics have no significant astigmatism. This means that there is no need to do separate calibrations for horizontal and vertical directions and so the PUMA Control software has no facility to do separate axis calibrations.

Having got the X and Y measurements you now use a calculator to calculate the calibration factor required to change HUD pixel units into real world units (we are using micrometres in this example) like this:

X= 300 micrometres

Y= 143 HUD pixels

Calibration_factor = X/Y = 300/143 = 2.0979 micrometres per HUD pixel

Inputting the calibration factor and units

Now that you have a calibration factor, you need to input it into the PUMA Control software to allow all future measurements to be presented in these real world units. To do that go to the main menu, select ‘Defaults’ sub-menu and edit the ‘Calib_Int’ and Calib_Dec’ values to represent your calibration factor. This allows the input of a floating point number as two integers. For example if your calibration factor is 2.0979 you would change Calib_Int to be ‘2’ and Calib_Dec to be ‘0979’.

Next select the ‘Units’ menu item and change it to the units your calibration represents (micrometres in this example – see figure 11).

Get out of the menus and back to Pointer control mode and the line you drew should now display with the measurement below updated to calibrated units (figure 11, bottom right). Any further length or distance measurements you make on any specimen (using this same objective and AR Projector setup) – whether by interactive cursor drawing or by a HUD graticule, ruler or scale bar – will have results calibrated to real world units.

Z-Motor

General features

This allows the user to control the focussing Z-motor (figure 7, page Error: Reference source not found). This can be done by moving the joystick up and down with coarse and fine speed options but there are also some advanced functions such as setting up an automatic Z-sweep to start from a given position and also a ‘goto’ function that moves the Z-stage to a specified position.

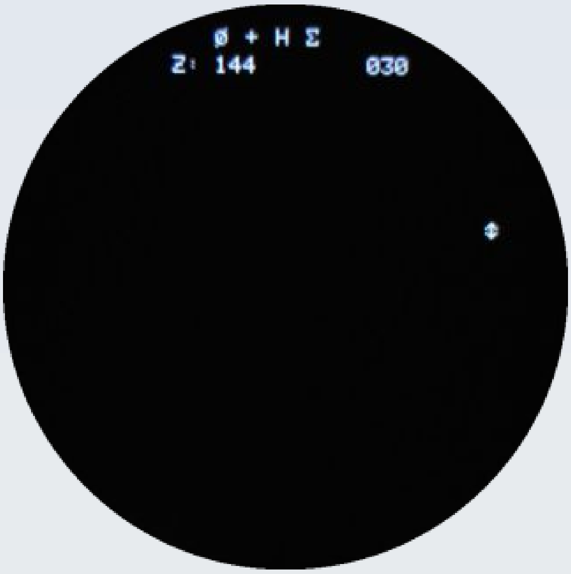


Figure 12. The Z-Motor HUD. This is an example to show where the information and icons for Z-Motor status are displayed on the HUD. The meaning of the following terms are all explained in the text. In this example the Z motor is homed (as indicated by the ‘H’) and is at position 144 steps above the homed (0) position. The motor coils are de-energised (Ø), the user is in coarse (fast) movement mode (+) and is in sweep mode (Σ). The user is in Z-Motor CM (⇕) so key actions will have the effects described in this chapter. The number ‘030’ top right is the lamp ammeter display (it is drawing 30 mA) and is not related to Z-Motor functions.

When the user is in Z-Motor CM the up-down arrow icon (⇕) will display in the right panel of the HUD (figure 12). This is important because information about the Z-motor and stage will usually be present all the time in the top panel of the HUD regardless of what CM is currently active, and users can easily toggle between the Z-Motor CM and other CMs so seeing this indicator lets the user know that when they move the joystick or press any button the actions expected for the Z-Motor control mode will be done.

Z-motor control (i.e. focussing) is such a fundamental process that a special facility is enabled in in other CMs to allow toggling to and from the Z-Motor CM. This toggle is achieved by the SETRST combination key action (see page 23).

All Z-positions are measured in motor steps and cannot be negative. The homed position is 0. This is the lowest position the motor can be moved to. This is also the default Z-position value that is set when the PCC is switched on or rebooted but the Z position displayed has no meaning until the scope has been homed (the user will not be allowed to control the Z-motor until the scope has been homed). The current Z-position is constantly displayed in the top panel of the HUD. As the motor is moved to raise or lower the stage the number of individual motor steps are cumulated into this total Z-position value (steps expended in backlash correction when the motor changes direction are not included in this cumulative count). This position is an 'open loop' value i.e. there is no independent sensor that monitors the stage Z-position. For this reason the accuracy and repeatability of moving the stage to a given position as denoted by this Z-position value will depend partly on the mechanical accuracy of the stage construction, partly on the appropriateness of the backlash correction factor (to be discussed) and partly on the assumption that the motor gear or stage focus is not moved by hand or does not stall or slip at any time.

The PUMA hardware only allows for a lower limit switch. There is no hardware control over how high the stage can be lifted by the motor. To try and compensate for this a software limit has been hard-coded as 80000 steps (= ZPOS_MAX in the source code). This represents about 3.5 mm of Z motion which is almost the physical limit of how much the Z stage can move in its range. Any attempt to move the motor more than this number of steps will not be permitted. Of course, this being a software limit, it is easy to 'fool' the system into going above this (e.g. by manually adjusting the focus gears) but you risk causing damage to the hardware if you do this.

If the user attempts to move the motor beyond one of its permitted motion limits the motion will be aborted and the FORBIDDEN audio signal emitted.

Menu Settings

The Z-Motor CM sub-menu has the following options (see figure 13).

Figure 13. The options in the Z-Motor sub-menu. See text.

Colour

The colour of the Z-Motor related information displayed in the HUD.

Value options: see page 33

Default: White

Z Speed

The delay (in milliseconds) between each change of the motor's coil energisation state (i.e. between individual steps). The lower this value the faster the motor will turn but if you go too fast the motor is likely to stall. Early experiment with prototype showed that significant stalling occurred below 4 ms. If you experience motor stalling in your setup, increasing this value may help remedy the situation at the expense of slower motor revolution.

Value options: Integer in the inclusive range 4 - 65535

Default: 16

Z Delta

The number of motor steps to move with each activation signal. This is the motion resolution. Using a value of 1 step (which translates to a Z movement of approximately

44 nanometres) gives the finest possible control with each up or down activation (an 'activation' is either a manual activation via the joystick or each automatic activation step that occurs as part of a sweep sequence). This resolution is probably too fine for most normal viewing and it will take a long time to move the focus even a very small distance so it is common to use a number greater than 1.

Value options: Integer in the inclusive range 1 - 4294967295

Default: 16

Z Minimum

This is the lowest position that the Z-motor will be allowed to travel to. This is a software lower limit for stage motion which allows the user to set a lower limit that is higher than the homed 0 position (because that level may be too low to prevent unwanted contact of objective with specimen, for example).

Value options: Integer in the inclusive range 0 – 80000

Default: 0

Sweep Start

An automated Z sweep may start from the current Z position of the stage or it may start from a specific Z position. This setting allows the user to set that specific Z position.

Value options: Integer in the inclusive range 'Z Minimum' – 80000

Default: 0

Sweep Steps

The number of steps an automated Z sweep goes through. Each of these steps will move the motor by 'Z Delta' motor steps (see above).

Value options: Integer in the inclusive range 1 – (80000/'Z Delta')

Default: 16

Sweep Delay

The delay in ms between steps in an automated Z sweep.

Value options: Integer in the inclusive range 0 – 4294967295

Default: 0

Backlash

The number of motor steps (individual motor steps, not multiples of 'Z delta') that will be applied whenever the motor changes direction and will not be included in the cumulative total of motor steps that comprise the current Z position. This allows for compensation due to backlash slack in the gear system.

Value options: Integer in the inclusive range 0 – 1000

Default: 640

PowerAlways

A binary flag that dictates whether the motor coils remain energised after motion ceases (1) or not (0). If not then the motor coils are de-energised after each move in order to conserve energy and avoid the risk of coil overheating.

Value options: Integer in the inclusive range 0 - 1

Default: 0

Homing the Z-motor

Homing the stage, homing the scope and homing the Z-motor are all synonymous terms. The Z-motor will not move if the scope has not been homed. A user will not be allowed to toggle into Z-Motor CM from another CM using the shortcut toggle key combination (SETRST) if the scope is not homed. Upon boot up the scope will not be homed so the user must deliberately home the scope if they want to control focus or perform any functions with the Z motor. The scope doesn't home automatically upon boot up because if it homes while a specimen is on the stage this could damage the specimen as well as the objective / scope. This is because the homing process moves the stage as low as possible till the limit switch is triggered and for some objectives this could put the objective through the slide.

The homed status of the scope is displayed in the top panel of the HUD. If a capital 'H' is present it means that the scope has been homed. If it is not present then the scope has not been homed.

Prior to homing, the user must ensure that the limit switch has been appropriately fitted (as per hardware construction guides) otherwise damage to the microscope may result.

The Z-homing process can be initiated in three ways:

1. Deliberately from the boot up screen by pressing the 'Set' button

2. Deliberately when in Z-motor CM by a combination press of SETDOWN (see page 23). This only works if the user is NOT in sweep mode – see below for details.
3. Whenever the Z-Motor CM is entered (from the menu) and the scope is not homed, the user is presented with the notice 'Z MOTOR IS NOT HOMED'. Press ‘Reset’ to exit this warning and go straight to the motor control mode (but you won't be able to use the motor until you go through with homing). Press any other button (i.e. except ‘Reset’) to clear this message and begin the homing sequence.

When homing is initiated:

- You will be advised, via the GUI, to remove the specimen from the stage and perform a JS_MID action (see page 23) to begin homing. The GUI notice actually says ‘PRESS CENTRAL BUTTON’ rather than using the ‘JS_MID’ abbreviation but it means the same thing.
- You can abort the homing process at this point by pressing something other than JS_MID but you wont be able to operate the motor until you have homed the stage. If you have previously homed the stage and are initiating another homing cycle then you will be able to use the motor on the basis of the previous homing.
- Once you perform the JS_MID action homing will commence and cannot be interrupted by any key action (you would have to manually intervene by, for example, switching off the PCC in order to stop this).
- After homing is successfully completed the current Z position displays 0 and the ‘H’ symbol will be displayed in the top panel of the HUD.

Once the motor is homed, it cannot be ‘un-homed’ other than by performing a re-boot of the PCC. However you can re-home the scope at any time. For example if you disturb the Z position by manually moving the focus you can do another homing procedure to reset the current Z position counter to 0.

Sweep mode and ‘Goto’

There are two Z-Motor modes: standard and sweep. You toggle between these two modes by a right move on the joystick (either alone or in combination with the ‘Set’ button). See page 23 for the meaning of the following key action abbreviations.

When in sweep mode SETDOWN triggers a downwards sweep from the current Z position.

When in standard mode the SETDOWN action initiates a homing request.

When in sweep mode SETUP triggers an upwards sweep from the current Z position.

When in standard mode SETUP does nothing.

When toggling between the modes JS_RIGHT just toggles between the modes. However a SETRIGHT combo action does this:

- When in sweep mode it simply toggles to standard mode - the same as if a JS_RIGHT action were made.
- When in standard mode it toggles into sweep mode and causes the scope to first 'goto' the start position for a sweep as set in the menu settings but it does not initiate the sweep itself. This functionality can also be used as a plain 'goto a set position' facility because you can activate it from standard mode then go straight back into standard mode if you wish.

Summary of Z-Motor HUD icons

All these appear in the top HUD panel except the ↕ which appears on the right panel (see figure 12).

↕ You are in Z-Motor control mode

H The Z-motor has been homed

Ø The motor coils are not energised

▲ The motor coils are energised in a pattern that will move the stage upwards when the motor is stepped. The motor may or may not currently be stepping (moving). If the motor is not moving and this icon is displayed it means that the motor was moving in that direction previously and has now stopped but the coils remain energised.

▼ The motor coils are energised in a pattern that will move the stage downwards when the motor is stepped. The motor may or may not currently be stepping (moving). If the motor is not moving and this icon is displayed it means that the motor was moving in that direction previously and has now stopped but the coils remain energised.

+ The system is in coarse focus mode. This means the speed is at the maximum possible setting (a delay of only 4 ms between steps) regardless of the user’s chosen Z speed in the menu.

Σ The system is in sweep mode.

Keypad actions for the Z-Motor CM

See page 23 for the meaning of the key action abbreviations.

JS_UP : Move stage Z-delta motor steps upwards

JS_DOWN : Move stage Z-delta motor steps downwards

JS_LEFT : Toggle between coarse and fine focussing (with icon)

JS_RIGHT : Toggle between standard and sweep mode (with icon)

JS_MID : Toggle the backlight on and off if a homing request has not been initiated, otherwise this begins the homing procedure.

SET_BTN :

- SET : De-energise the motor coils (with icon). The motor will re-energise (and the de-energised icon cleared) whenever the motor is reactivated.
- SETRST : Select the previous non-Menu CM as current control mode.
- SETDOWN : Initiate a request to home the Z motor if in standard mode or start a downward sweep if in sweep mode.
- SETUP : Do nothing if in standard mode or start an upward sweep if in sweep mode.
- SETMID : No action.
- SETLEFT : Click the Set button in combination with JS_LEFT
- SETRIGHT : Toggle between standard and sweep mode (with icon) but if toggling into sweep mode perform a 'goto' action by moving the Z motor to the 'Sweep Start' Z coordinate specified in the settings. Do not initiate the sweep, just go to that position and wait for the user to take the next step.

RST_BTN : Enter the main Menu (or clear the 'Z MOTOR IS NOT HOMED' warning if this is displayed)

Pointer

General features

This allows the user move a cursor around the HUD and, optionally, to deposit marks on the display. The cursor is moved via the up, down, left, right and diagonal motions of the joystick and a mark is deposited at the current location of the cursor by means of a JS_MID key action (see page 23 for an explanation of how to perform this and other key actions referred to in the chapter).

These marks can be used to highlight features, count objects and measure distances and angles. Although this is called ‘Pointer’ CM, the cursor can be any one of a number of shapes, not just arrows.

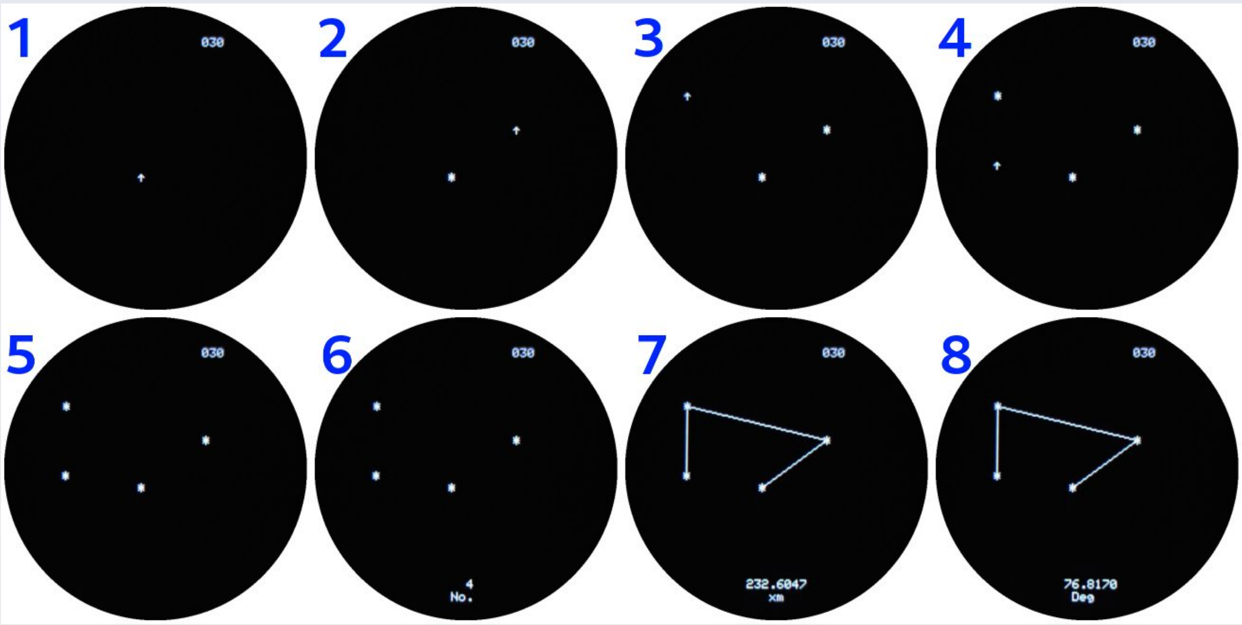


Figure 14. Pointer marking and measurements. This shows a simple sequence of pointer moves (using the † cursor) to 4 positions with a mark (✱) laid down at each position by the user (images in sequence from 1 to 5). Images 6 to 8 show the three types of measurements that can be made with these marks as detailed in the text. The number ‘030’ at the top right of each image is the lamp ammeter reading (30 mA) and is not related to the Pointer functions.

This CM provides a ‘pointer only’ mode (figure 14, images 1 to 5) and 3 additional measurement modes for counting, lengths and angles (figure 14, images 6 to 8). The default mode is pointer only. Other modes are selected (cycled) using the SET button as follows:

1. Pointer: Move a cursor character about with the joystick and deposit marks but no results are displayed i.e. the screen is blank except for the cursor and any deposited marks (and any HUD information from the Z-Motor, Timer and lamp ammeter).
2. Counter: As for the above but the measurement results HUD panel is displayed (the lower panel) showing number of marks deposited (figure 14, image 6).
3. Length: A (poly)line is drawn between all marks laid down by the user and the length of the (poly)line line is displayed in the the measurement HUD panel (figure 14, image 7). This will be in units calibrated by the calibration factor set in the menu (see page 38). You need at least 2 marker points for a line length to be calculable. The FORBIDDEN buzzer signal will sound if you try to read a length from fewer than 2 marks.
4. Angle: A (poly)line is drawn between all marks laid down by the user and the angle between the last two lines drawn is displayed in the the measurement HUD panel (figure 14, image 8). This is in degrees and is the smaller of the two possible angles so will be between 0 and 180 degrees. You need at least 3 marker points for an angle to be calculable. The FORBIDDEN buzzer signal will sound if you try to read an angle from fewer than 3 marks.

The joystick can't be used for both pointer movement and Z-motor focus control simultaneously so a quick toggle between the pointer control mode and the Z-motor control mode is provided by the SETRST combination key action.

The coordinates of all marks laid down by the user are stored in global memory arrays so will be preserved when the user switches over to the menu or any other CM (including the Z-motor CM) so will be preserved when they return to the Pointer CM. Because this global storage uses the limited PSRAM of the Arduino Nano the maximum number of points you can mark is quite limited and, in the current version of PUMA Control, this upper limit is set at 127 marker points.

The marker point character is selected by the user via the Pointer settings menu (see below). Different characters are optimal for different uses. For example, if you want to trace the outline of a complex shape this will require many segments to the poly-line and possible many marker points laid down close to each other. In this scenario it is often best to use the single pixel marker character because these will effectively only show the poly-line and will not complicate the line with larger marker characters closely crowded over each other. You can change the marker character at any time without having to re-draw the poly-line manually – whenever you go to the menu to select a new colour or character and then return to Pointer CM, all previously laid down marker points will be drawn with the newly chosen settings.

To clear all marks and reset the count to perform a SETMID combination key action. You may want to do this prior to measuring a new line or a new angle. You will be prompted prior to deleting all marks because this cannot be undone. Performing the simple JS_MID key action after the prompt will delete the data. Pressing any other button will abort the delete request and return the user to where they were before.

The speed of motion of the cursor can be adjusted by the SETUP or SETDOWN combination key actions. Slowing down the motion of the cursor makes it easier to place it at a precise location.

You can undo and redo marks – in the reverse order to which they were laid down – using the SETLEFT (for undo) or SETRIGHT (for redo) combination key actions. No matter how many marker points you 'undo', you can 'redo' all points that you have 'undone' all the way to the most recently added marker point. For example if you lay down 73 points and you undo all of them, you can redo all 73 points provided you don't add any new marker points. As soon as you add a new marker point then that becomes the 'last point added' and you can't 'redo' any more points beyond that. So in this example if you 'redo' points up to the 50th point and, at that stage, you add a new marker point then you can't continue to redo any of the previous 73 points beyond the 51st point you have just added.

Menu Settings

The Pointer CM sub-menu has the following options (figure 15).

Figure 15 Pointer sub-menu.

Colour

The colour of the cursor (not the markers or other text of the HUD).

Value options: see page 33

Default: White

Pointer

The cursor character (as distinct from the marker point character)

Value options: ↑, X, +, », «, ▼, ▲, ◀, ▶, ←, →, ↓

Default: ↑

Marker Colour

The colour of all markers laid down and also the measurement text.

Value options: see page 33

Default: White

Marker

The character to use when laying down marker points.

Value options: ✱, ·, X, #, O, +

Default: ✱

Measure the length of a line or boundary

All necessary information to do this has been presented above but it may help to go over a complete measurement procedure from start to finish (see figure 16).

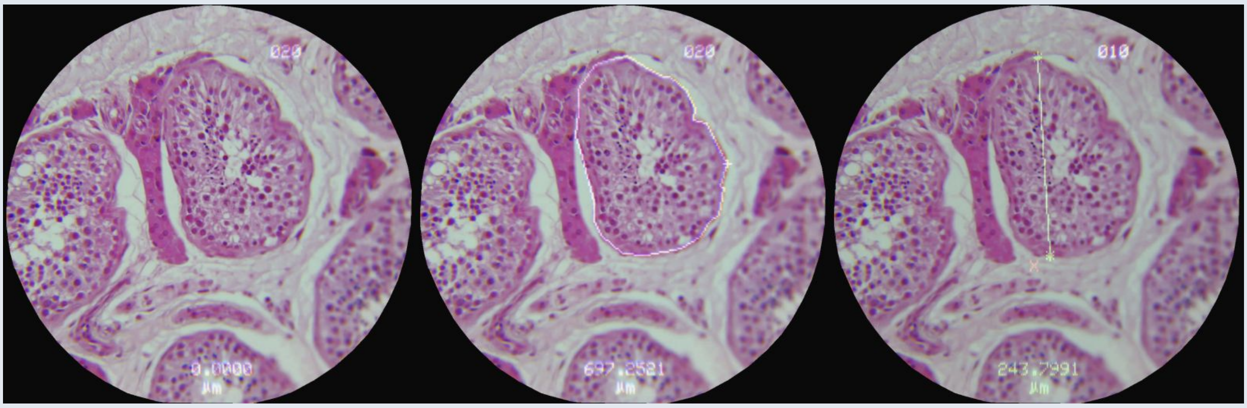


Figure 16. Examples of using Pointer CM to measure a boundary and and line (a diameter) of a seminiferous tubule in an H&E-stained histological section of testis. Left: No marks yet laid down, the lamp ammeter reading shows top right and the measurement below (currently 0) with the symbol for micrometres as units (this HUD was calibrated to micrometres). Middle: The boundary of the selected tubule has been traced by placing multiple marks along its border. In this case the single pixel marker has been selected so all you see is the boundary. All text and graphics are in the default white colour. The boundary measures 697.2521 microns. Right: The previous marker points were deleted and just two marker points laid down at opposite ends of the tubule to measure this diameter line. Note that the colours have been changed via the menu. Green for marks and other text and graphics, red for the cursor. The cursor has been changed to an and the marker character has also been changed to an . The line measures 243.7991 micrometres.

1. Via the menu select your choice of colours and cursor / marker characters
2. When in pointer control mode move the cursor, with the joystick, to one end of the line and perform a JS_MID key action to lay down a marker character.
3. Mover the cursor to the other end of the line and perform a JS_MID key action to lay down another marker character.
4. Press the SET button (on its own) repeatedly until the length measurement result appears on the bottom HUD panel.

That's it. The result will be in whatever units you have previously calibrated the system in (see page 38) or it will be in HUD pixels if the system has not been calibrated.

For boundary measurements the procedure is similar. However in this case you will be creating a poly-line by continuing to lay down marker points at intervals along the boundary of the object to be measured. If you are measuring a closed boundary be sure to lay down a final marker point at the same position as the very first marker point (so the first and last marker point are superimposed on each other).

For tracing and measuring complex lines and boundary shapes that may require many marker points in close proximity to each other (so you can accurately follow tight curves and turns) it will usually be best to select the single pixel marker character (·) because this avoids a messy and overly complex display of overlapping marker characters and will effectively just show the poly-line on its own. This can be seen in figure 16.

Keypad actions for the Pointer CM

See page 23 for the meaning of the key action abbreviations. In Pointer CM there will be no buzzer ACKNOWLEDGED signal when moving the cursor around. Also, the amount of auto-repeat delay for holding the joystick in one position is set by the user by altering the pointer speed as described below.

JS_UP : Move cursor up one pixel

JS_DOWN : Move cursor down one pixel

JS_LEFT : Move cursor left one pixel

JS_RIGHT : Move cursor right one pixel

In Pointer CM you may also move the joystick in any of the four 45 degree diagonal directions to produce a combined movement in the diagonal directions.

JS_MID : Place a mark at the current cursor position and update any count or measurement. If you have initiated a 'delete all marks' action and are being prompted if you really want to delete all, then perform a JS_MID key action to confirm delete.

SET_BTN :

- SET : Cycle between pointer, counter, length and angle modes
- SETRST : Select the Z-motor CM as current control mode.
- SETDOWN : Decrease the cursor motion speed
- SETUP : Increase the cursor motion speed
- SETMID : Reset the marker count to zero (delete all marks).
- SETLEFT : Undo the previous marker
- SETRIGHT : Redo (replace the last marker that was 'undone')

RST_BTN : Enter the main Menu

Timer

General features

This allows the user start/stop a count-down or count up from a start time set via the menu. The start time can also be modified by key actions in the Timer CM (for minutes and seconds only, hours can only be set or modified via the menu). The time on the timer will appear on the right HUD panel with hours on top followed by minutes then seconds at the bottom (figure 17).

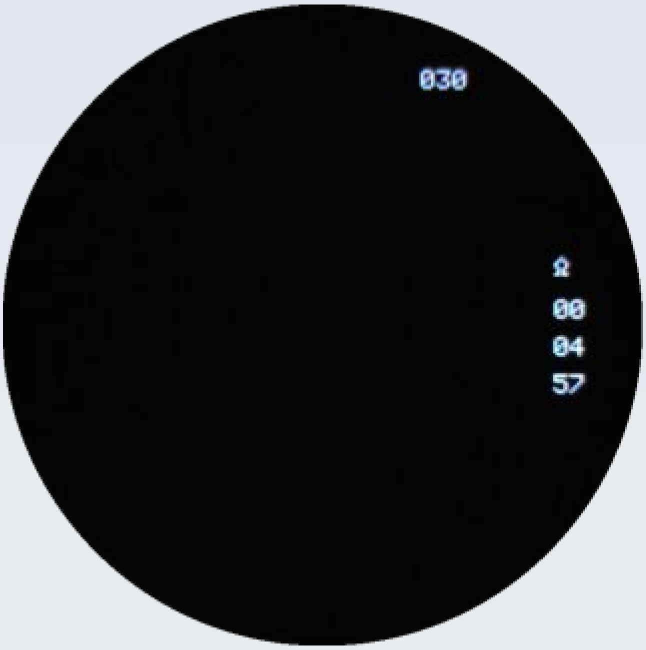


Figure 17. Timer information on the HUD. The icon shows you are in Timer CM. Beneath this is the current time on the timer in hours (top), minutes (middle) and seconds (bottom). The ‘030’ at top right is the lamp ammeter reading (30 mA) and is nothing to do with the timer.

Start / Stop toggle is via the JS_MID key action (see page 23 for an explanation of how to perform this and other key actions referred to in the chapter). If timer is initially started from 00:00:00 then it performs a count-up and start/stop will halt or continue the count up, otherwise it performs a count-down from the current time shown and start/stop will halt or continue this count down.

A reset for 'cold-start' is performed by the SETMID combination key action. This can be done during the running of the timer or when the timer has been stopped. This ‘cold-start’ stops the timer (if it is running) and resets the timer to the start time.

When the timer is not running the start time can be adjusted by the joystick direction buttons. (details below)

The counter is designed to be used in conjunction with other modes so it has been made easy to switch to other modes from the counter. This is done via the JS_UP / JS_DOWN key actions (when the counter is running) and the familiar SETRST combination action toggles in and out of the Z-motor CM (at any time, whether the timer is running or not).

Note that you can switch back and forth between Graticule and Timer modes (and the graticule will persist on the HUD as you do so) but you can only switch from Timer to Pointer mode with a key action and not the other way round (the only way to get into Timer CM mode from Pointer CM is via the main menu). Also, if you switch to Timer CM from Graticule CM (so the graticule persists on the HUD while you are in Timer CM) and you attempt to use the shortcut key action to switch over to Pointer CM then the graticule HUD will be cleared prior to entering the Pointer CM (because Graticule and Pointer graphics cannot be used simultaneously).

Because the timer can be used with other CM and its HUD is visible when other CM are the currently active CM, it has its own icon which appears whenever you are in Timer CM to let you know how to expect the joystick keypad to behave. This is the omega icon (Ω) which appears above the timer hours value in the right HUD panel

For count down only, at the end (when 00:00:00 is reached) the countdown signal is given as determined from the menu settings (audible, flash, both or none).

Menu Settings

The Graticule CM sub-menu has the following options (figure 18).

Figure 18. The Timer sub-menu.

Colour

The colour of the numbers and icon of the HUD.

Value options: see page 33

Default: White

Hours

The number of hours to count down.

Value options: Integer in the inclusive range 0 – 99

Default: 0

Minutes

The number of minutes to count down.

Value options: Integer in the inclusive range 0 – 59

Default: 0

Seconds

The number of seconds to count down.

Value options: Integer in the inclusive range 0 – 59

Default: 0

Signal

The signal to give when the countdown reaches 00:00:00.

Value options: Flash + Beep, Flash only, Nothing, Beep only

Default: Flash + Beep

Keypad actions for the Timer CM

See page 23 for the meaning of the key action abbreviations.

JS_UP : If timer running goto Pointer CM else decrement start minutes by 1

JS_DOWN : If timer running goto Graticule CM else increment start minutes by 1

JS_LEFT : Decrement start time seconds by 1 second (roll over at 00 to 59)

JS_RIGHT : Increment start time seconds by 1 second (roll over at 59 to 00)

JS_MID : Start / stop the timer

SET_BTN :

- SET : Cycle the timer colour
- SETRST : Select the Z-motor CM as current control mode.
- SETDOWN : No Action
- SETUP : No Action
- SETMID : Reset the timer to menu default (stops the timer if running)
- SETLEFT : No Action
- SETRIGHT : Redo (replace the last marker that was 'undone')

RST_BTN : Enter the main Menu

Graticule

General features

This allows the user to superimpose a graticule, cross-hair, ruler or scale bar pattern on the image (figure 19).

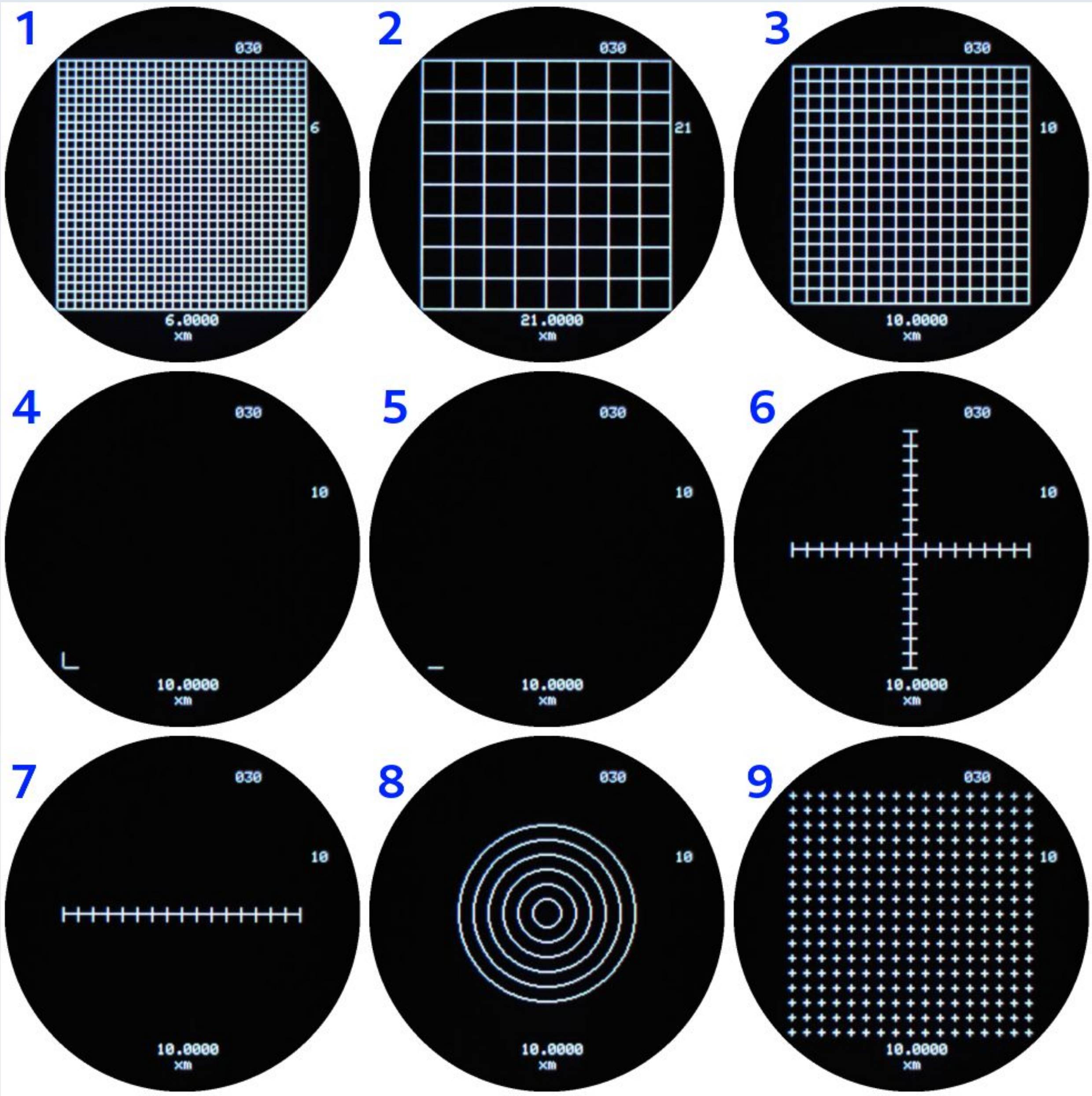


Figure 19. Graticules. 1,2,3 All examples of the same graticule type ('Square grid') but at different scales. The scale is conveyed by the number at the right e.g. '4', '21', '10' . This number denotes the period in HUD pixels so '4' means the lines are 4 HUD pixels apart. The period can be changed by a simple key action (as can the grid colour) – see text for details. The other available grid patterns are shown in 4-9 as follows: 4. Scale bar XY, 5. Scale bar X, 6. Ruler XY, 7. Ruler X, 8. Circles, 9. Cross grid

With simple key actions the type, colour and scale of the superimposed pattern can be changed easily and quickly and it can also be toggled on / off without having to go via the menu. The HUD backlight can also be switched on and off (so as to clear all HUD information from the visual field).

The usual quick toggle over to Z motor control is provided via the SETRST combination key action (see page 23 for an explanation of how to perform this and other key actions referred to in the chapter).

There is also a key action for quick switch-over to the Timer mode so the user can easily go back and forth between timer control and graticule control

The pointer and graticule graphics cannot be used simultaneously because it would be difficult to prevent or remedy overdrawing of the graticule by the interactive cursor within the limits computing resources available in the PCC.

Menu Settings

The Graticule CM sub-menu has the following options (figure 20).

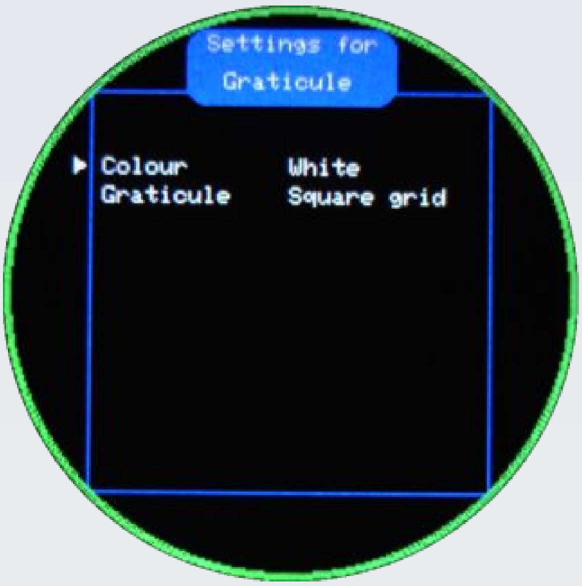


Figure 20. Graticule sub-menu.

Colour

The colour of the lines and text of the HUD.

Value options: see page 33

Default: White

Graticule

The type of graticule / pattern (see figure 19).

Value options: Square grid, Scale bar XY, Scale bar X, Ruler XY, Ruler X, Circles, Cross grid

Default: Square grid. The default line spacing, called the **graticule period**, is 10 HUD pixels but this is not changed from the menu settings but via a key action (see below).

Keypad actions for the Graticule CM

See page 23 for the meaning of the key action abbreviations.

JS_UP : Increase graticule period by one pixel

JS_DOWN : Decrease graticule period by one pixel

JS_LEFT : Cycle visual verbosity: Graticule -> No graticule -> Backlight off

JS_RIGHT : Go into Timer control mode (see Timer chapter for how to get back)

JS_MID : Cycle to the next graticule pattern

SET_BTN :

- SET : Cycle the graticule colour
- SETRST : Select the Z-motor CM as current control mode.
- SETDOWN : No Action
- SETUP : No Action
- SETMID : No Action
- SETLEFT : No Action
- SETRIGHT : No Action

RST_BTN : Enter the main Menu

Plain

General features

This allows the user to fill the HUD screen with a plain colour at the current backlight level.

This is intended to be used for illumination effects. For example, the AR projector is normally used to project an image of its TFT screen to form the HUD. However, with some modification, the AR projector could be used as an ultra low power objective lens (~ x0.8) to project an image of any object placed at the same position as where the TFT screen is normally situated. This Plain CM can then be used to provide background transillumination for the microscope slide. Such a usage requires a modified TFT holder that has a slide holder incorporated which puts the specimen slide in focus and the TFT deliberately out of focus some distance behind the slide (otherwise you would see a textured dotty background due to the pixels of the TFT screen being in focus).

This Plain CM may alternatively be used with the SLM as a custom colour light filter.

Menu Settings

The Plain CM sub-menu has only one options for colour (figure 21) – the backlight brightness is set via the 'Defaults' settings accessed from the main menu.

Figure 21. The Plain sub-menu

Colour

The colour of the plain area.

Value options: see page 33

Default: White

Keypad actions for the Plain CM

See page 23 for the meaning of the key action abbreviations.

JS_UP : No Action

JS_DOWN : No Action

JS_LEFT : No Action

JS_RIGHT : No Action

JS_MID : No Action

SET_BTN :

- SET : Cycle the plain colour
- SETRST : Select the Z-motor CM as current control mode.
- SETDOWN : No Action
- SETUP : No Action
- SETMID : No Action
- SETLEFT : No Action
- SETRIGHT : No Action

RST_BTN : Enter the main Menu

Aperture

General features

This allows the user create a set of variable apertures for the SLM. The basic apertures are bright field, phase contrast (using the Schlieren method) and dark ground (‘dark field’) – see figure 22. The dark field aperture is a central blocking spot and this may be superimposed onto the other two apertures to create a combined aperture pattern (figures 23 ans 24).

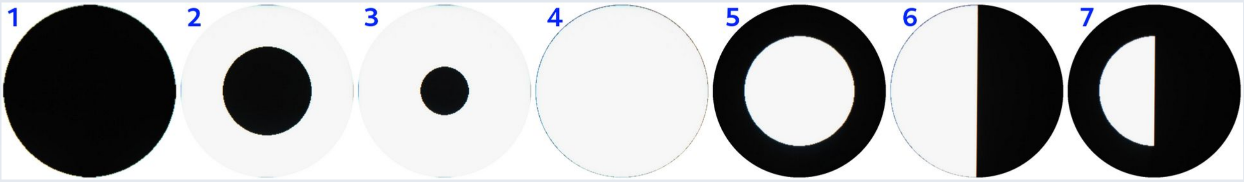


Figure 22 Apertures (individual) in different sizes. 1. Darkfield default aperture. This almost blanks out the whole of the aperture so would need to be reduced in size for most applications as shown in images 2 and 3. 4. Full bright field aperture. 5. Smaller bright field aperture. 6. Full Schlieren phase contrast aperture. 7. Smaller Schlieren phase contrast aperture. The Schlieren apertures are shown in only one orientations (see figure 23 for more).

Asymmetric aperture patterns can be rotated in 90 degree steps (figure 23).

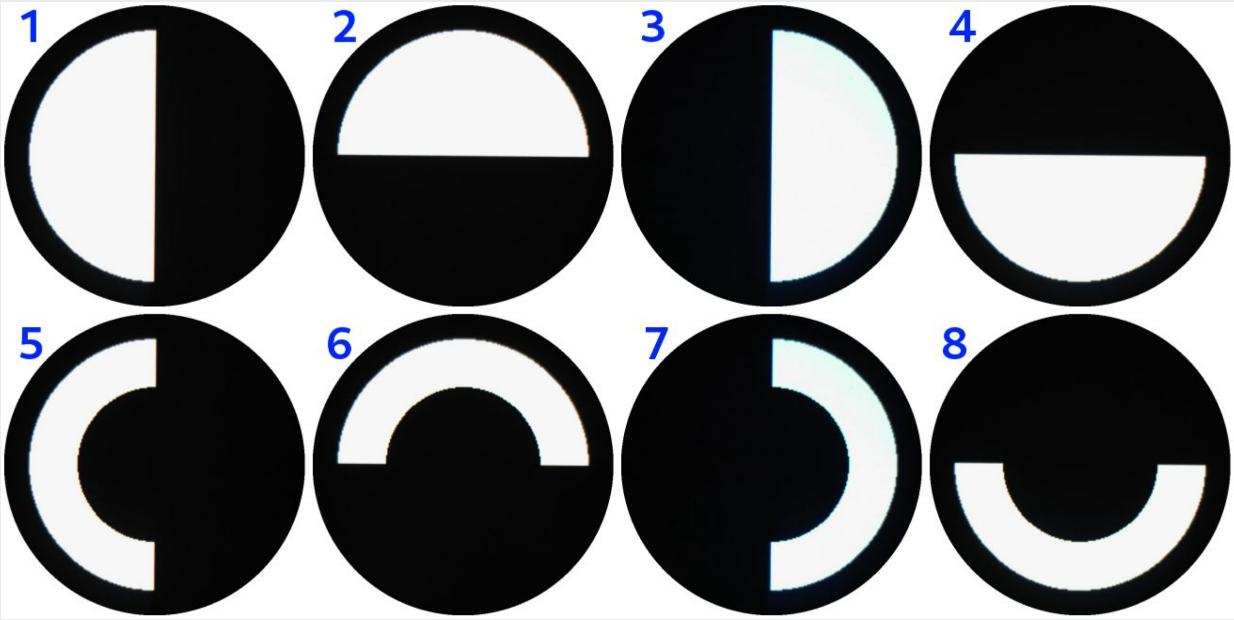


Figure 23 Aperture rotations and combinations using Schlieren (top panel, images 1 to 4) and combined Schlieren and darkfield (bottom panel, images 5 to 8)

Aperture components can be increased and decreased in radius in steps as small as 1 HUD pixel difference in radius between size changes. This step size can be increased and decreased to effect faster/slower coarser/smoother changes in aperture radius. For combined aperture patterns, the radius of each component can be varied independently (figure 24).

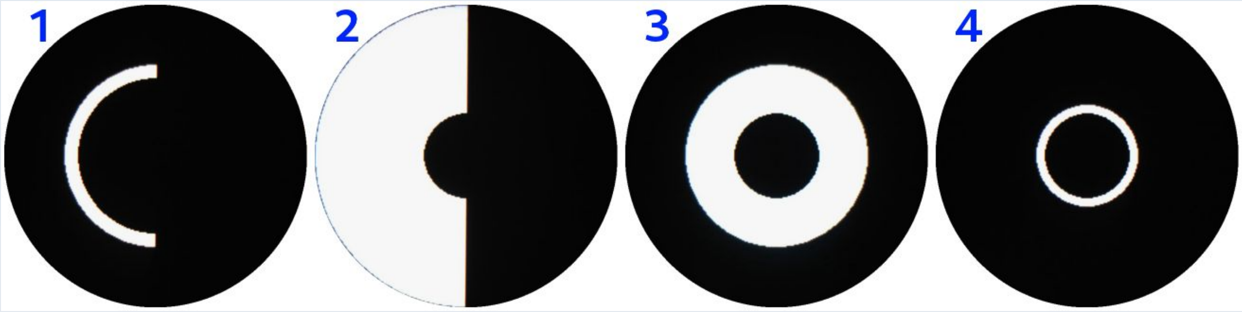


Figure 24. Combined apertures at independently different scales for each component of a combined aperture. 1. Schlieren at small scale combine with darkfield at large scale. 2. The converse of 1. 3. Brightfield at large scale combine with darkfield. 4. Brightfield at a small scale combined with darkfield at the same scale as it was in image 3.

This uses the SLM filter TFT, so the AR TFT cannot be used simultaneously and all HUD information is suppressed with the sole exception of the Z-Motor CM icons: and whenever the user toggles into Z-Motor control mode - as can be done via the usual SETRST combination key action (see page 23 for an explanation of how to perform this and other key actions referred to in the chapter).

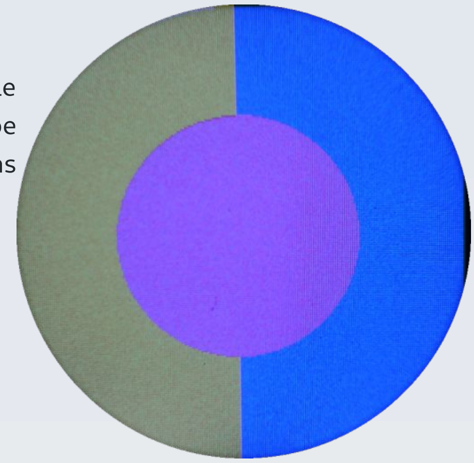
This control mode offers only a very limited set of basic functional apertures and does not do full justice to the capabilities of the SLM. This is because the PCC has very limited program memory and was primarily designed to drive the AR Projector HUD so priority was given to the control modes used for that. Because the SLM is an advanced research features it is anticipated that users will write their own custom driver software for it to make use of its powerful properties. Furthermore, as part of the PUMA project future development, it is also intended to write alternative software for the PCC where the emphasis is on SLM aperture functions (rather than AR HUD functions) and this can be loaded into the PCC as an alternative to PUMA Control if someone wants to make more extensive use of the SLM. Until such alternative software is developed, this Aperture CM was included in the standard PUMA Control software to ensure that at least basic SLM functionality can be experienced with an all-in-one software solution.

Rheinberg filters

The colours of the various components of the aperture default to black and white as shown in the above figures. However, you have the option of setting colours for the clear aperture part (what appears white in the previous figures) and the non-clear aperture part. The central dark field spot is treated as a separate structure for the purpose of colouring so this can be given a separate colour to the rest of the non-clear aperture.

The result is that you can create various coloured shaped filters that can give enhanced pseudo-colour contrast to images without totally obstructing the light. Such filters are known as Rheinberg filters and an example is shown below (figure 25).

Figure 25. One of many possible Rheinberg filters that may be created using the colour options of the Aperture settings



Menu Settings

The Aperture CM sub-menu has the following options (figure 26).

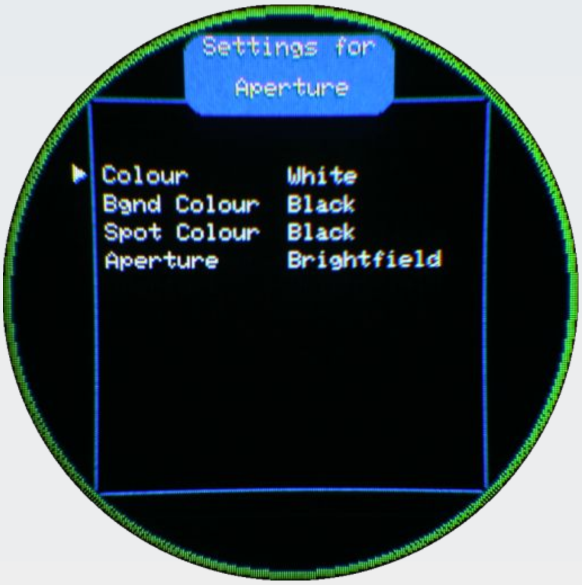


Figure 26 The Aperture sub-menu

Colour

Value options: see page 33

Default: White

The colour of the clear aperture

Bgnd Colour

Value options: see page 33

Default: Black

The colour of any structure that is not the clear aperture or the central dark field spot.

Spot Colour

Value options: see page 33

Default: Black

The colour of the central dark field spot

Aperture

The type of aperture pattern to show (see figure 22).

Value options: Brightfield, Phase cntrst, Darkfield

Default: Brightfield

Keypad actions for the Aperture CM

See page 23 for the meaning of the key action abbreviations.

JS_UP : Increase radius of current aperture

JS_DOWN : Decrease radius of current aperture

JS_LEFT : Increment aperture radius change step size

JS_RIGHT : Decrement aperture radius change step size

JS_MID : Toggle the current aperture on/off (off = fully open plain aperture)

SET_BTN :

- SET : Cycle aperture pattern
- SETRST : Select the Z-motor CM as current control mode.
- SETDOWN : Rotate the display in +90 degree intervals
- SETUP : Rotate the display in -90 degree intervals
- SETMID : Toggle toggle darkfield superposition
- SETLEFT : No Action
- SETRIGHT : No Action

RST_BTN : Enter the main Menu

Index

[End of Document]