Diff Ctrl EPICS IOC User Guide

Version 3.0

August/2023

Brazilian Synchrotron Light Laboratory (LNLS)
Brazilian Center for Research in Energy and Materials (CNPEM)

About this manual

This manual provides an overview of the Diff Ctrl EPICS Soft IOC. It is assumed that the reader is familiar with the basics of EPICS.

Contents

1 Introduction

The Diff Ctrl EPICS IOC is a Soft IOC that controls the position of 2 ’blades’ or ’edges’ that move in opposite directions. This IOC aims to meet the needs of different applications that could have this description. In case of Sirius, it is used for Slits and Scrapers at the accelerator. Since both systems use DMC 30017 Galil controllers, the PVs accessed by the Diff Ctrl IOC in order to command the motors of a given device are provided by (two) instances of the Galil DMC 30017 EPICS IOC support by LNLS.

The Diff Ctrl IOC repository is `https://github.com/lnls-dig/diff-ctrl-epics-ioc`.

2 Installation

Shortcut It is possible to run a *docker image* containing the IOC and all of its dependencies. Go to the Running the IOC from a docker container section to see how to do it.

2.1 Installing without docker

Before everything Make sure you have EPICS Base 3.15 and the following EPICS modules installed (older versions might work but are not recommended):

- autosave R5-9
- seq 2-2-6
- calc R3-7
- busy R1-7

Clone the IOC repository and checkout the desired commit (in case you are not using master).

```
git clone https://github.com/lnls-dig/diff-ctrl-epics-ioc.git <install
   location>
cd <install location>
git checkout <commit>
```

Create a *RELEASE.local* file inside the *configure/* directory and add the paths to EPICS Base and external modules. An example file is available at *configure/RELEASE.local.example*.

```
echo ’EPICS_BASE=<EPICS Base path>’ > configure/RELEASE.local
echo ’AUTOSAVE=<Autosave path>’ >> configure/RELEASE.local
echo ’SNCSEQ=<seq path>’ >> configure/RELEASE.local
echo ’CALC=<calc path>’ >> configure/RELEASE.local
echo ’BUSY=<busy path>’ >> configure/RELEASE.local
```

From the IOC <TOP> directory, run make:

```
make
```

3 Running the IOC without docker

After successfully building the IOC, from the IOC top directory, run:

```
cd iocBoot/iocDiffCtrl &&
./runDiffCtrl.sh <options>
```

The available options can be listed by passing any invalid option to the startup script, such as:

```
./runDiffCtrl.sh -h
```

or can be consulted in the README.md file at the IOC <TOP> directory or project page https://github.com/lnls-dig/diff-ctrl-epics-ioc.

4 Running the IOC from a docker container

Make sure you have docker installed and that you understand the basics of creating, running and acessing a container, and downloading an image.

4.1 Downloading the image (IOC + dependencies)

The image named **lnlsdig/diff-ctrl-epics-ioc** contains the compiled IOC and dependencies, and can be downloaded to your machine by running:

```
docker pull lnlsdig/diff-ctrl-epics-ioc:<IOC_TAG>
```

where *<IOC_TAG>* should be replaced by the desired IOC version.

The IOC version list can be found at https://hub.docker.com/r/lnlsdig/diff-ctrl-epics-ioc/tags.
After the IOC image is downloaded, you can instante it by running a container that uses it.

4.2 Running a docker container with that image

One way to run the container is to do:

```
docker run -it --network host --restart always --name <CONTAINER_NAME>
   lnlsdig/diff-ctrl-epics-ioc:<IOC_TAG> -P <PREFIX1> -R <PREFIX2> <
   other options>
```

Which is going to run the container starting at its default entrypoint, that is, the IOC startup script. The parameter <IOC_TAG> should be replaced by the IOC version. All the arguments after it are passed to the IOC startup script when run in this way. The <PREFIX1> and <PREFIX2> are the IOC EPICS PVs prefix, combined as follows: $PV\_PREFIX = PREFIX1 + PREFIX2$. The other startup script arguments should be provided as well. They are listed in the IOC README.md file.

If you run this command before downloading the image, the image is automatically downloaded, provided there is internet connection.

Below you find an explanation for each flag we used in the above command.

- -it: interactive (STDIN open) and allocate pseudo-TTY. Since we did not change the container entrypoint, your terminal is going to connect to the IOC shell.
- –network host: configures the container to use the host's network stack inside the container. It is the simplest configuration to use, although it provides no isolation.

- –restart always: restarts the container automatically when it exits.
- –name <CONTAINER_NAME>: configures the container name as <CONTAINER_NAME>.

In order to stop the docker container, i.e., the IOC, run:

```
docker stop <CONTAINER_NAME>
```

If you want to delete the container, after stopping it, run:

```
docker rm <CONTAINER_NAME>
```

You can also run the contaier with the *–rm* flag to clean up the container and remove the file system when it exits.

5 Device Reference Frame

The IOC control follows the beam coordinate system. This means the X axis increments in value from right to left (greater beam energy) and the Y axis increments in value from bottom to top.

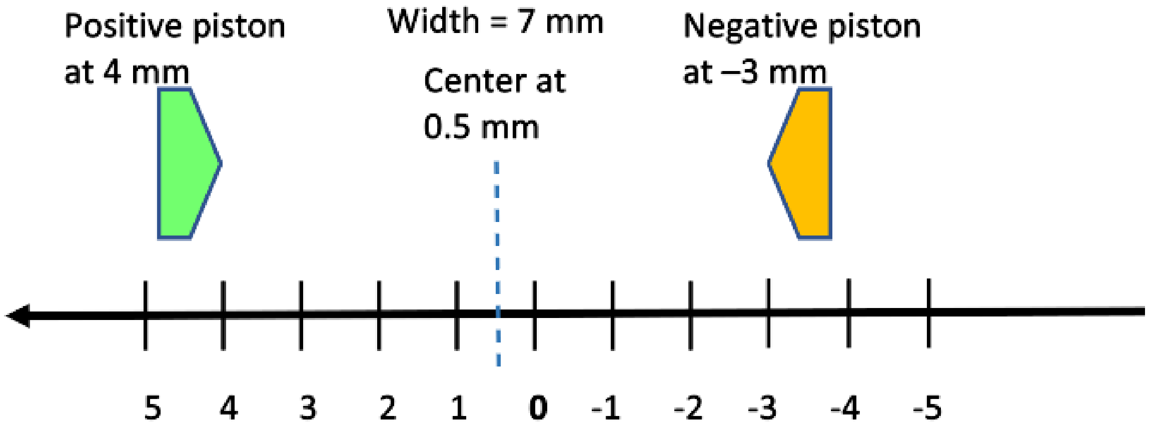
The origin of the X and Y axis is considered to be at the device center. If the beam passes through the 0 position, this means it is perfectly centered with respect to the device.

The device blade, or edge, that when retreated is positioned in the negative region of the X axis (or Y axis) is refered to as the *Negative Edge*.

The device blade, or edge, that when retreated is positioned in the positive region of the X axis (or Y axis) is refered to as the *Positive Edge*.

The figure 1 illustrates the positioning of the blades (edges) considering some coordinates.

Figure 1: Device reference frame overview

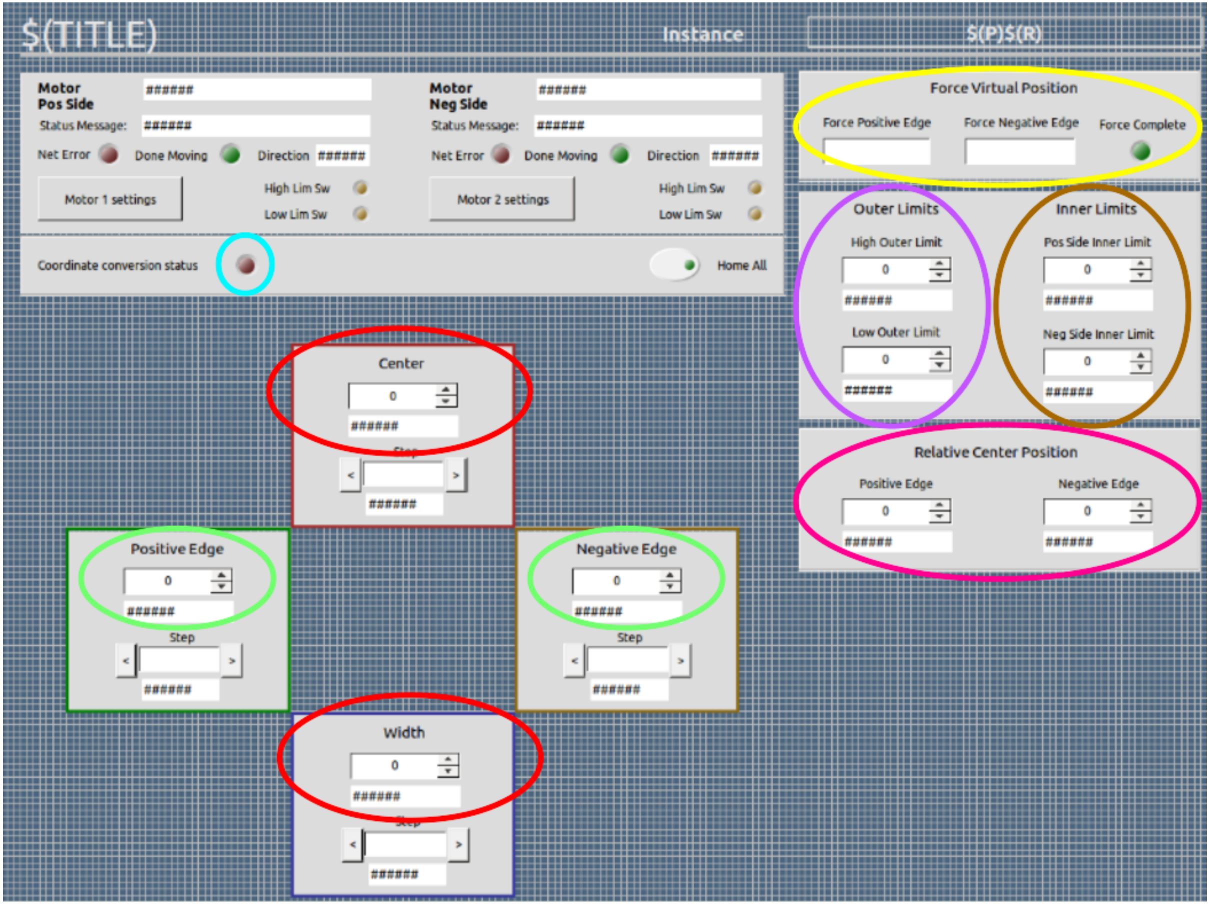


6 Edge Control Overview

Following a common way of implementing Slit control, the user can control the device by either specifying each blade position (in the device reference frame) or provide the slit center position and width. Figure 1 illustrates these parameters.

Figure 2 shows the Diff-Ctrl CS-Studio OPI with the main features circled. The green circles show where the *Negative* and *Positive* edge positions can be specified. The red circles show where the gap center and width should be specified.

Figure 2: OPI overview



7 Edge Position Calibration

The readback position of the blades can be calibrated to account for encoder offsets. The yellow circle in Figure 2 shows the *Force Position* fields. These fields allow the user to force a desired readback position to a blade (or edge). When a value is set, the *Force Complete* LED turns off while the force operation is in progress. When the force operation is complete, the LED turns on again. Only one edge position can be forced at a time. The value should be specified in engineering units.

The *Force Position* command can only compensate offsets to the encoder values. It cannot compensate for deviations in the device motion, since this is encoded in the motor-edge conversion function applied for each edge. To understand how to compensate for fixed offsets in edge position, see section Reference Center Calibration.

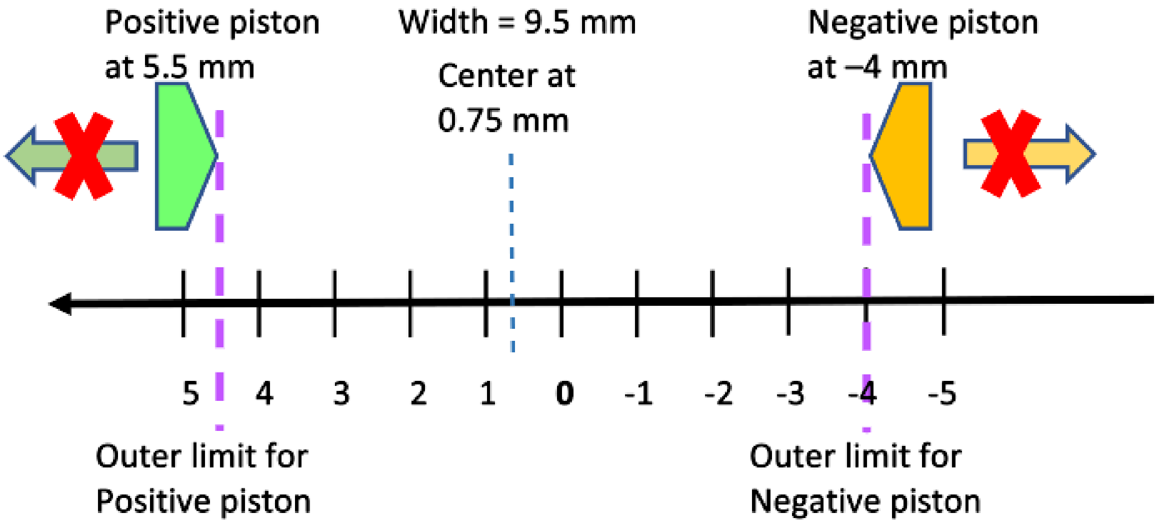
8 Configuring Software Limits

This IOC should control a device with 2 actuators. This means there can be 4 limits to the device motion, that is, 2 for each blade.

The limits to the blade retreat motion are referred to as *Outer Limits*, since they are associated with limiting the opening of the device gap. The limits to the blades "advance" motion are referred to as *Inner Limits*, since they refer to limits for the gap closing. The "retreat" and "advance" terms refer only to the motion of parts that can interact with the beam, that is, the blades themselves. Any coordinate conversion between motors and blade position should not be considered here.

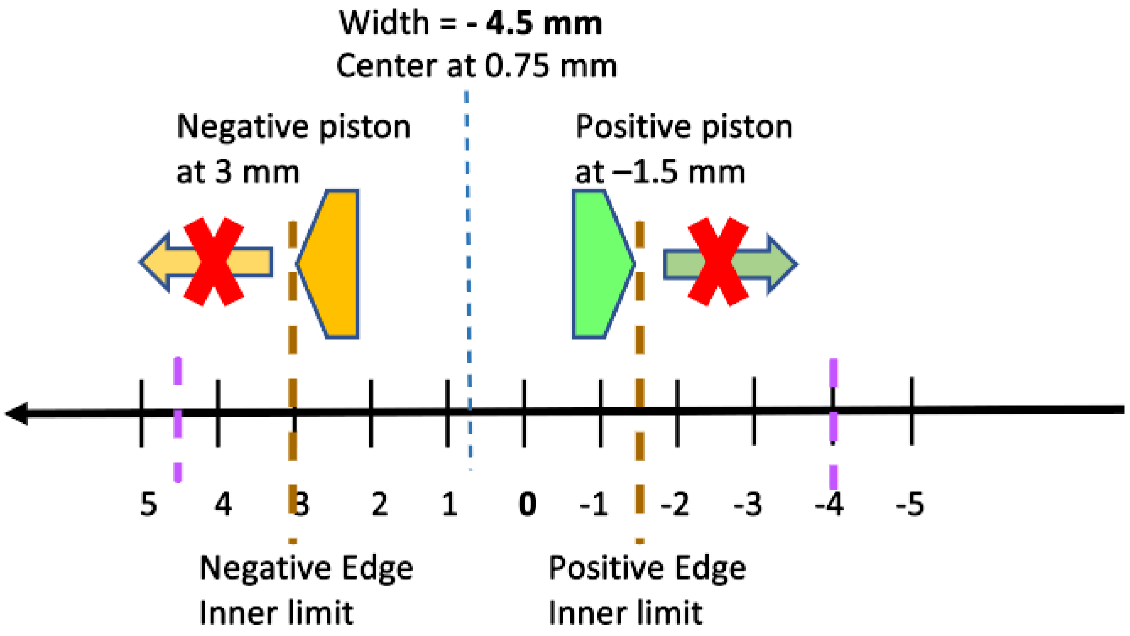
The Figure 3 illustrates how the outer limits configuration affects motion.

Figure 3: Outer limits



The Figure 4 illustrates how the inner limits configuration affects motion.

Figure 4: Inner limits



In Figure 2, the *Outer Limits* configuration fields are circled in purple, while the *Inner Limits* configuration fields are circled in brown.

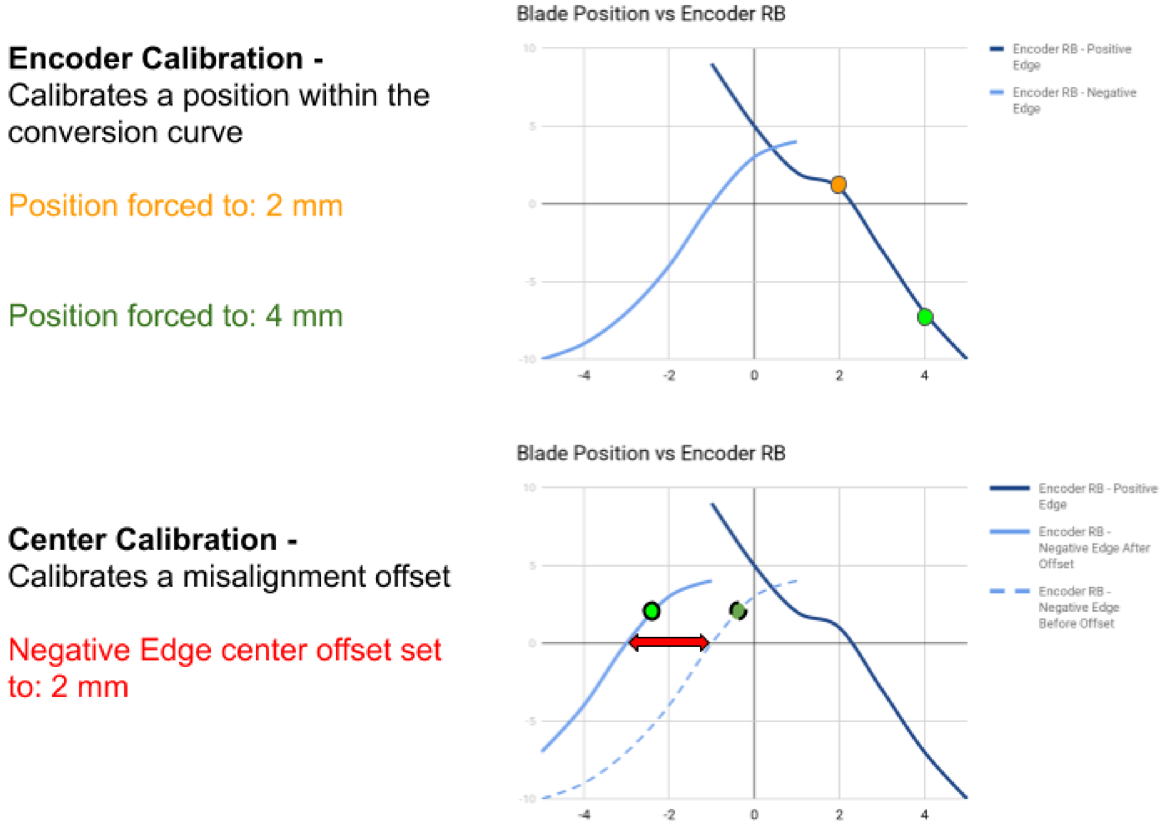
9 Reference Center Calibration

In addition to the calibration of the offset of encoder readings, it is also possible to configure an additional offset for edge position. When a position is forced for encoder calibration, the resulting encoder offset is considered when converting the motor position into the blade position through the selected coordinate conversion function. The *Center Offset* can be used to shift the blade position reading, ignoring the underlying conversion function between motor and blade coordinates.

The Figure 5 shows the difference between the *Reference Center Offset* and the *Encoder Offset*. While the *Encoder Offset* shifts the position along the conversion function curve, the *Reference Center Offset* shifts the blade position directly away or towards the device center position, while keeping the encoder readback unchanged. This means that each blade coordinate conversion function input is kept unchanged.

The *Center Offset* is useful for compensating offsets due to device misalignment (the device center does not match the vacuum chamber center) without the need to modify the conversion functions code.

Figure 5: Offset types



The Figures 6 and 7 illustrate how the *Center Offset* value affects the position values. Greater offset values virtually ”retreat” the edge with respect to the device center. Figure 7 shows offsets configured for blades that have moved beyond the center position (the effect is the same, but retreating puts the edge closer to the center). Negative offset values do the opposite, ”moving” the blades in the advancing direction.

Figure 6: Reference Center Example - blades retreated

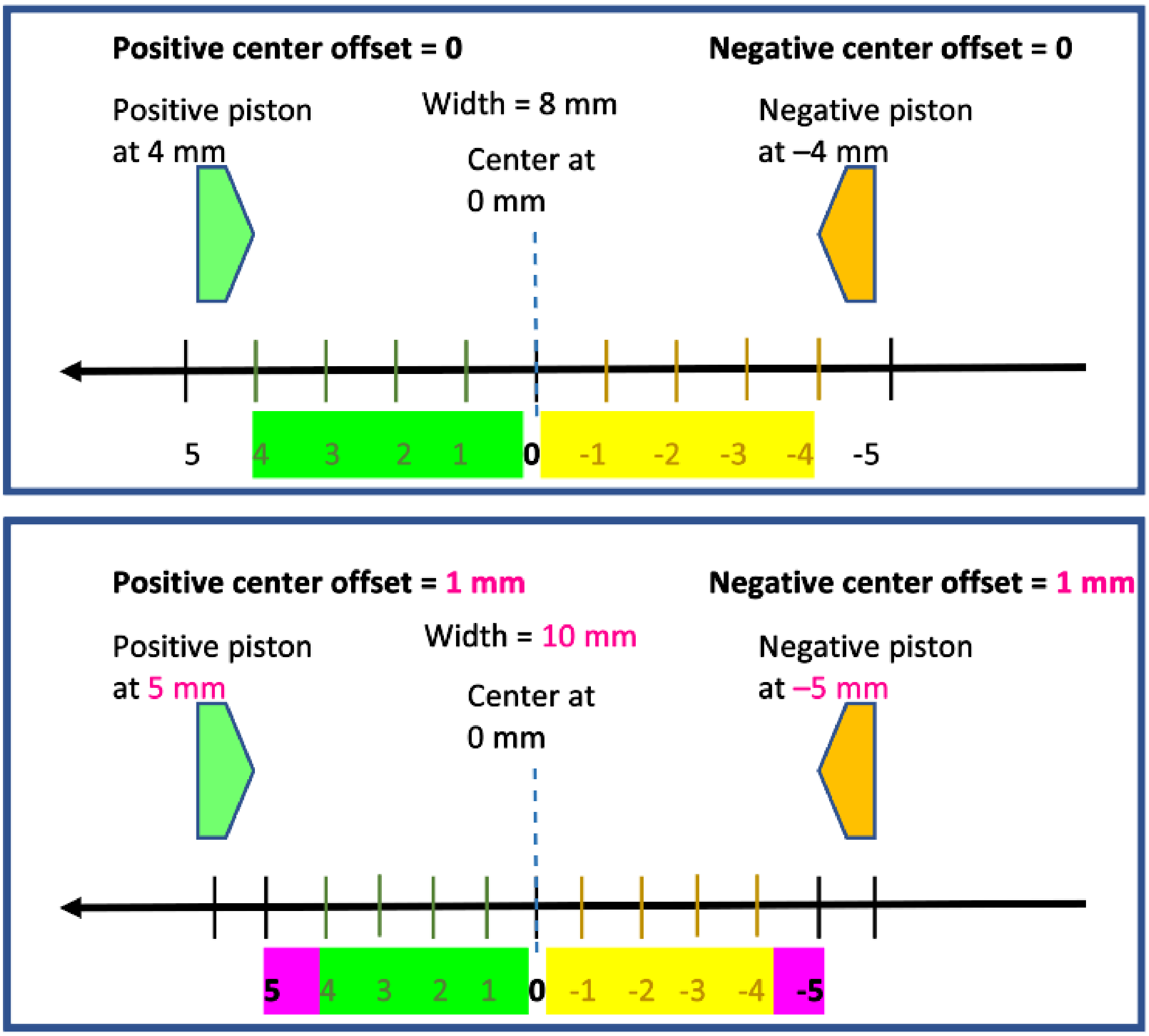
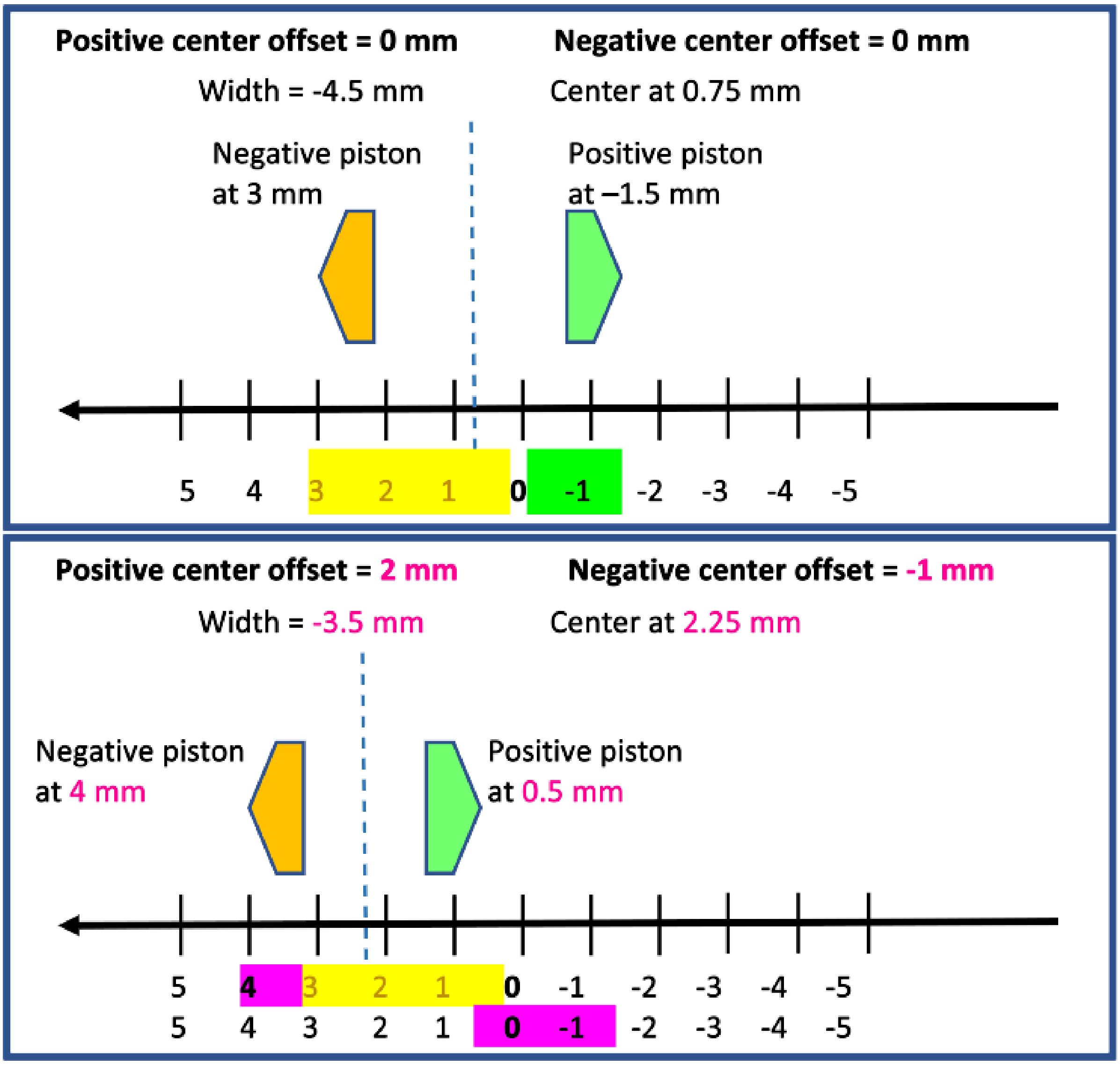


Figure 7: Reference Center Example - blades advanced past center



10 Motor vs Blade Reference Frame Conversion

In order to support a wider variety of motion types, the commanded position for a device blade is converted into a motor controller set point position by a custom conversion function associated to an EPICS Array Subroutine record (aSub).

Different functions can be used for each blade, and for set point and readback conversion.

Currently, there are 3 sets of functions available, each associated to a device type that can be selected at IOC startup. For each device type, the startup script selects a different set of functions and argument values:

1. slit: Uses a dummy conversion for both set points and readbacks. The blade position is the same as the "motor" position.
2. scraper-h: Uses interpolation of a table specific to the Sirius Horizontal scraper to map blade and "motor" positions. The table contains a limited range of encoder readings/ motor set points. Values outside range are not converted.
3. scraper-v: Uses interpolation of a table specific to the Sirius Vertical scraper to map blade and "motor" positions. The table contains a limited range of encoder readings/ motor set points. Values outside range are not converted.

11 How to Add a New Device Type

This section explains how to create a new device type and configure its own conversion functions (see also Motor vs Blade Reference Frame Conversion).

11.1 The conversion function signature

The conversion functions employed are run by subroutine records in the IOC record database. Therefore, the conversion subroutines should follow the structure required by the aSub record, that is, its input arguments and output values should be read/written from/to the record.

The record *Input A* contains the position value to be processed:

- For command conversion: the input is the device blade position. This value should be converted to a motor position to be sent to the controller IOC.
- For readback conversion: the input is the encoder reading. This value should be converted to a device blade position.

The converted value should be written to the record *VALA* field (value A).

If the conversion fails, the subroutine should return a value different from zero.

The record inputs B - L can be used for passing parameters to the functions at startup.

11.2 Add the new conversion function(s)

1. At the *<TOP>/diffCtrlApp/src/convFnc.c* file:
   (a) Add the new function definition to it.
   (b) At the end of the file, add the line:

```
epicsRegisterFunction(<function name>);
```

2. Alternatively, a different source file can be created, but then it must be added to the *src/* directory *Makefile*. Likewise, if additional header and source files are needed, they must be added to the *Makefile*.

3. At the *<TOP>/diffCtrlApp/src/convFnc.dbd* file:

   (a) For each convertion function created, add the following line:

```
function(<function name>)
```

11.3 Add a new device type

1. To add a new device type option at IOC startup, edit the *<TOP> /iocBoot/iocDiffCtrl/parseCMDOpts.sh* file:

   (a) Add the new device for the *d* option, as in the example.

```
"slit") DEVICE="SLIT" ;;
<new device option>) DEVICE=<some device string> ;;
...
```

   (b) Add the new device type in the *usage* message, so people know it is available.

11.4 Associate the new device and convertion functions

1. To associate the new device type to the new convertion functions, edit the *<TOP> /iocBoot/iocDiffCtrl/runDiffCtrl.sh* file:

   (a) Add the convertion functions that should be used for each device blade, for conversion of the commanded blade position to ”motor” position and from encoder value to blade position readback. In addition, there are 11 optional parameters that can be passed to the conversion functions at IOC startup: from CONV_INPB to CONV_INPL. The parameter values are passed to all 4 functions. The following example shows a new device being added after the Slit device configuration:

```
    SLIT)
        CONV_FUNC_NEG_CMD="slitConvFnc";
        CONV_FUNC_NEG_RB="slitConvFnc";
        CONV_FUNC_POS_CMD="slitConvFnc";
        CONV_FUNC_POS_RB="slitConvFnc";
        ;;

    <some device string>)
        CONV_FUNC_NEG_CMD=<cmd function for negative edge>;
        CONV_FUNC_NEG_RB=<RB function for negative edge>;
        CONV_FUNC_POS_CMD=<cmd function for positive edge>;
        CONV_FUNC_POS_RB=<RB function for positive edge>;
        CONV_INPB=<parameter 1>
        CONV_INPC=<parameter 2>
        ...
        ;;
```

2. When appropriate, add the new device type to the *README.md* and *documentation.*

12 Process Variables Description

Each IOC instance should add a prefix to the process variables indicating which device it controls.

The placeholders <P_NAME> and <N_NAME> should be replaced by the corresponding values defined in iocBoot/iocDiffCtrl/runDiffCtrl.sh by the environment variables POSITIVE_EDGE_NAME and NEGATIVE_EDGE_NAME, respectively.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
PV_NAME	DATA_TYPE	DESCRIPTION
<N_NAME>MotionCtrl-Cte	String (char[40])	Prefix used for device *negative edge* motion controller.
<P_NAME>MotionCtrl-Cte	String (char[40])	Prefix used for device *positive edge* motion controller..
<N_NAME>DoneMov-Mon	Enum: No (0) Yes (1)	Indicates when the *negative edge* is done moving, i.e., when it has completely finished its motion.
<P_NAME>DoneMov-Mon	Enum: No (0) Yes (1)	Indicates when the *positive edge* is done moving, i.e., when it has completely finished its motion.
<N_NAME>SlitPos-SP	Float: min -1e6, max 1e6	Command the *negative edge* to move to the specified distance, in engineering units, from the *reference axis center*. More information on the device reference frame convention can be found in section Device Reference Frame. This PV does not necessarily indicate the current edge position, for that, the readback value should be used.
<N_NAME>SlitPos-RB	Float	Shows the *negative edge* distance, in engineering units, from the *reference axis center*. This value is updated whenever the edge encoder registers a movement.
<P_NAME>SlitPos-SP	Float: min -1e6, max 1e6	Command the *positive edge* to move to the specified distance, in engineering units, from the *reference axis center*. More information on the device reference frame convention can be found in section Device Reference Frame. This PV does not necessarily indicate the current edge position, for that, the readback value should be used.
<P_NAME>SlitPos-RB	Float	Shows the *positive edge* distance, in engineering units, from the *reference axis center*. This value is updated whenever the edge encoder registers a movement.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
Center-SP	Float	Command both the *positive and negative edges* movement so that the midpoint between them is located at the specified distance, in engineering units, from the *reference axis center*. More information on the device reference frame convention can be found in section Device Reference Frame. This PV does not necessarily indicate the current center position, for that, the readback value should be used.
Center-RB	Float	Shows the distance of the midpoint between the *positive and negative edges*, in engineering units, from the *reference axis center*. This value is updated whenever the encoder of one of the edges registers a movement.
Width-SP	Float	Command both the *positive and negative edges* movement so that the distance between them is set to the specified value, in engineering units. They edges are moved in such a way that keeps the *Center-RB* position unchanged. More information on the device reference frame convention can be found in section Device Reference Frame. This PV does not necessarily indicate the current width between the edges, for that, the readback value should be used.
Width-RB	Float	Shows the distance between the *positive and negative edges*, in engineering units. This value is updated whenever the encoder of one of the edges registers a movement.
CoordConvErr-Mon	Enum: Ok (0), Error (1)	Error status of the coordinate conversion between encoder reading and actual edge position. If the error status is 1, then the conversion function failed for the given commanded position or for the current encoder reading. This can be due to invalid input, for instance, an out of range position, or to some error in the function itself.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
<N_NAME>SlitInnerLim-SP	Float: min -1e6, max 1e6	The inner limit associated to the *negative edge*, in engineering units. This is the limit position when moving this edge toward the reference center from the retreat position. The *negative edge* retreat position is usually negative, but if the edge keeps advancing, it eventually arrives at a positive position. Thus, if the *negative edge* inner limit is positive, the *negative edge* can move beyond the reference center, until it reaches the specified limit. If it is negative, the edge can move towards the center until it reaches the specified limit.
<N_NAME>SlitInnerLim-RB	Float	The readback value of the *negative edge* inner limit.
<P_NAME>SlitInnerLim-SP	Float: min -1e6, max 1e6	The inner limit associated to the *positive edge*, in engineering units. This is the limit position when moving this edge toward the reference center from the retreat position. The *positive edge* retreat position is usually positive, but if the edge keeps advancing, it eventually arrives at a negative position. Thus, if the *positive edge* inner limit is negative, the *positive edge* can move beyond the reference center, until it reaches the specified limit. If it is positive, the edge can move towards the center until it reaches the specified limit.
<P_NAME>SlitInnerLim-RB	Float	The readback value of the *positive edge* inner limit.
<N_NAME>SlitOuterLim-SP	Float: min -1e6, max 1e6	The outer limit associated to the *negative edge*, in engineering units. This is the limit retreat position for the *negative edge*, i.e., the most negative position this edge can have.
<N_NAME>SlitOuterLim-RB	Float	The readback value of the *negative edge* outer limit.
<P_NAME>SlitOuterLim-SP	Float: min -1e6, max 1e6	The outer limit associated to the *positive edge*, in engineering units. This is the limit retreat position for the *positive edge*, i.e., the most positive position this edge can have.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
<P_NAME>SlitOuterLim-RB	Float	The readback value of the *positive edge* outer limit.
<P_NAME>EdgeCenter-SP	Float	The distance between the *positive edge* and the reference center. A greater distance value shifts the available positions for the *positive edge* in the positive direction (greater edge position values). Smaller distance values do the opposite, including negative values.
<P_NAME>EdgeCenter-RB	Float	The readback value of the distance between the *positive edge* and the reference center.
<N_NAME>EdgeCenter-SP	Float	The distance between the *negative edge* and the reference center. A greater distance value shifts the available positions for the *negative edge* in the negative direction (smaller edge position values). Smaller distance values do the opposite, including negative values.
<N_NAME>EdgeCenter-RB	Float	The readback value of the distance between the *negative edge* and the reference center.
Force<N_NAME>SlitPos-Cmd	Float	Force a given position reading, in engineering units, to the *negative edge* (does not move the edge). This is useful for calibration of the edge positions. For instance, if the change of an enconder shifts the absolute readings, but a known position can be restored (by other measurement or by a reference such as a limit switch), then the known position can be forced. The IOC does this by setting an offset to the motion controller's encoder readings.
Force<P_NAME>SlitPos-Cmd	Float	Force a given position reading, in engineering units, to the *positive edge* (does not move the edge). This is useful for calibration of the edge positions. For instance, if the change of an enconder shifts the absolute readings, but a known position can be restored (by other measurement or by a reference such as a limit switch), then the known position can be forced. The IOC does this by setting an offset to the motion controller's encoder readings.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
ForceComplete-Mon	Enum: No (0), Yes (1)	Completion status of the force operation. After a position reading is forced to some of the edges, the IOC takes some time to apply the configuration. When the new offset configuration is finished, this PV goes to 1.
Home-Cmd	Enum: Off (0), On (1)	Command both edges to the retreat position. The *positive edge* moves in the positive direction, until it hits a limit switch, while the *negative edge* moves to the negative direction, until it hits a limit switch.
<P_NAME>SlitStep-SP	Float: min -1e6, max 1e6	The step size, in engineering units, to use *positive edge* when an increment or decrement *positive edge* position command is issued. See Inc<P_NAME>Slit-Cmd and Dec<P_NAME>Slit-Cmd.
<P_NAME>SlitStep-RB	Float	The readback value of the *positive edge* step.
<N_NAME>SlitStep-SP	Float: min -1e6, max 1e6	The step size, in engineering units, to use when an increment or decrement *negative edge* position command is issued. See Inc<N_NAME>Slit-Cmd and Dec<N_NAME>Slit-Cmd.
<N_NAME>SlitStep-RB	Float	The readback value of the *negative edge* step.
CenterStep-SP	Float: min -1e6, max 1e6	The step size, in engineering units, to use when an increment or decrement *center* position command is issued. See IncCenter-Cmd and DecCenter-Cmd.
CenterStep-RB	Float	The readback value of the *center* step.
WidthStep-SP	Float: min -1e6, max 1e6	The step size, in engineering units, to use when an increment or decrement *width* command is issued. See IncWidth-Cmd and DecWidth-Cmd.
WidthStep-RB	Float	The readback value of the *width* step.
Inc<P_NAME>Slit-Cmd	Enum: Off (0), On (1)	Increment the *positive edge* position by the step specified by <P_NAME>SlitStep-RB.
Dec<P_NAME>Slit-Cmd	Enum: Off (0), On (1)	Decrement the *positive edge* position by the step specified by <P_NAME>SlitStep-RB.

Table 1: Application Process Variables

Name	**Data Type**	**Description**
Inc<N_NAME>Slit-Cmd	Enum: Off (0), On (1)	Increment the *negative edge* position by the step specified by <N_NAME>SlitStep-RB.
Dec<N_NAME>Slit-Cmd	Enum: Off (0), On (1)	Decrement the *negative edge* position by the step specified by <N_NAME>SlitStep-RB.
IncCenter-Cmd	Enum: Off (0), On (1)	Increment the *center* position by the step specified by CenterStep-RB.
DecCenter-Cmd	Enum: Off (0), On (1)	Decrement the *center* position by the step specified by CenterStep-RB.
IncWidth-Cmd	Enum: Off (0), On (1)	Increment the *width* by the step specified by WidthStep-RB.
DecWidth-Cmd	Enum: Off (0), On (1)	Decrement the *width* by the step specified by WidthStep-RB.