Arm® Control Tool User Guide

Version r0p1-00eac0 / Revision r0p1

Revision Information

Date	Issue	Confidentiality	Change
23/03/2018	EAC	Confidential	Final Release
29/03/2019	EAC	Confidential	YUV Conversion support

Confidential Proprietary Notice

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20348

Confidentiality Status

This document is Confidential. This document may only be used and distributed in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Product Status

The information in this document is Final, that is for a developed product.

Contents

1 Introduction 10
2 System requirements 11
2.1 Control tool GUI 11
2.2 Control tool sever 11
3 Access overview 12
3.1 Hardware Access 12
3.1.1 Direct hardware Access 12
3.1.2 Stream hardware access 13
3.2 Channels overview 13
4 Running the control tool server 16
4.1 Command listen 16
4.2 Command direct 16
4.3 Getting basic information 17
4.4 Preset configuration files 18
4.5 Command line options 19
4.5.1 Options to set up access 19
4.5.2 Channel options 20
4.5.3 HTTP endpoint options 21
4.5.4 Miscellaneous options 22
5 Graphical user interface 23
5.1 GUI layout 23
5.2 Actions menu 24
5.3 Sections 25
5.4 Controls 25
5.4.1 Buttons 25
5.4.2 Data Types 25
5.4.3 Data Mode 26
5.5 Utility extensions 26

5.5.1 Image converter 26
5.5.2 Calibrations converter 27
5.5.3 JSON to BIN Converter 27
5.5.4 Pipeline Visualisation 27
5.6 Miscellaneous 28
5.6.1 Force reload hardware configuration 28
5.6.2 View list of the recent errors 28
5.6.3 View current status of the running server 28
5.6.4 Running in limited offline mode 28
5.6.5 Getting help about command line options 28
5.6.6 Troubleshooting GUI issues 29

List of Tables

Table 1. Hardware access command line options 12
Table 2. Available hardware and software channels 14

Preface

This preface introduces the Arm® Control Tool User Guide. It contains the following sections:

- About this book.
- Feedback.

About this book

This book is for the Arm Control Tool version 3.3.x.

Implementation obligations

This book is designed to help you implement an Arm product. The extent to which the deliverables may be modified or disclosed is governed by the contract between ARM and Licensee. There may be validation requirements, which if applicable will be detailed in the contract between Arm and Licensee and which if present must be complied with prior to the distribution of any silicon devices incorporating the technology described in this document. Reproduction of this document is only permitted in accordance with the licences granted to Licensee.

Arm assumes no liability for your overall system design and performance, the verification procedures defined by Arm are only intended to verify the correct implementation of the technology licensed by Arm, and are not intended to test the functionality or performance of the overall system. You or the Licensee will be responsible for performing any system level tests.

You are responsible for any applications which are used in conjunction with the Arm technology described in this document, and to minimize risks adequate design and operating safeguards should be provided for by you. Arm's publication of any information in this document of information regarding any third party’s products or services is not an express or implied approval or endorsement of the use thereof.

Product revision status

The rmpn identifier indicates the revision status of the product described in this book, for example, r1p2, where:

rm Identifies the major revision of the product, for example, r1.

pn Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is for system designers, system integrators, and verification engineers who are designing a System-on-Chip (SoC) device that uses the Mali-IV009 Image Signal Processor (ISP).

It is assumed that the readers of this document are familiar with ISP hardware and ISP software.

Using this book

This book is organized into the following sections:

Section	Description
Introduction	This section provides a brief introduction of the Control Tool.
System requirements	This section provides a list of the Control Tool system requirements.
Access overview	This section provides information about the ways in which you can achieve hardware and software access.
Running the control tool server	This section provides information on how to run the Control Tool server.
Graphical user interface	This section provides information about the Control Tool user interface and its various options.

Typographic conventions

Font/text type	What it means
italic	Introduces special terminology, denotes cross-references, and citations.
bold	Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.
`monospace`	Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.
`monospace italic`	Denotes arguments to monospace text where the argument is to be replaced by a specific value.
`monospace bold`	Denotes language keywords when used outside example code.
`<and>`	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: `MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>`

Additional reading

Information published by Arm and by third parties.

See http://infocenter.arm.com for access to Arm documentation.

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to errata@arm.com. Give:

- The title.
- The number ARM-EPM-137445.
- The page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note: Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

1 Introduction

Arm ISP hardware and ISP software have many settings that can be controlled, observed and tuned. It is very important to understand and monitor what is going on in the system and to have a way to change the state and save it for future re-use or analysis.

The Arm Control Tool provides a fast and reliable means for communication with the ISP hardware and ISP software in real-time to enable you to monitor and control various settings.

The Arm Control Tool is a standalone user-space application consisting of the following parts:

- Control Tool server (the act-server executable)
- Control Tool web-based graphical user interface (GUI)

The Control Tool server is a command line application which can be built for various target platforms. It works as middle-ware between the user interface and the back-end represented as ISP hardware and optionally ISP software. The server communicates with the back-end by means of channels (which are implemented by drivers).

The GUI is a web-based thin client and works in the following web-browsers only:

- Mozilla Firefox (latest ESR which is 52.1.2 or later);
- Chromium or Google Chrome (58.0 or later).

The Control Tool server application can be launched on the target system (if it has Linux OS running and network interface), or on user system (when the latter can be connected to the target device via supported cables). The Control Tool client web GUI always works in a supported web browser on the user's system. In accesses Control Tool server by HTTP URL which can be found in the server console log output after it is started.

If the Control Tool server application is started on user's system, it can be accessed via the URL `http://127.0.0.1:8000/` or `http://localhost:8000/` provided that the default port number 8000 is used.

2 System requirements

This section provides information about the minimum system requirements of the Control Tool GUI and the Control Tool Server.

2.1 Control tool GUI

- Minimum screen resolution supported: 1000 x 600.
- Mozilla Firefox (latest ESR which is 52.1.2 or later) supported.
- Chromium or Google Chrome (58.0 or later) supported.

2.2 Control tool sever

Supported platforms:

- Linux x86_64 (a desktop system) (x86_64-linux-gnu),
- Microsoft Windows x64 (x86_64-win32),
- Microsoft Windows x86 (i686-win32),
- Google Android AArch64,
- Various Arm and AArch64 based embedded Linux systems.

3 Access overview

Both the ISP hardware and ISP software can be accessed in different ways which depend on which connection types are supported by the target system and on how the software part was built.

3.1 Hardware access

Hardware access can be either:

- direct hardware access
- stream hardware access

3.1.1 Direct hardware Access

To access the hardware directly it is usually not required to have the software running, however the Control Tool GUI will only display the hardware settings.

`--hw-acc=direct` is the default value for this option.

In this case:

- a hardware channel is required, this channel can be one of the following: i2c, devmem, or mdcmdio;
- the software, if present and running, can be accessed indirectly through the hardware channel or through its own channel.

The command line options are explained in the following table:

Software Access	SW Channels (--fw-channel)	Command Line Option	Details
None	(software is not available)	`--fw-acc=none`	no access to software
Hardware Buffer	(software is not available)	`--fw-acc=hw`	only hardware channel
Stream Access	`chardev, socket, uart`	`--fw-acc=stream`	support required in software
Software Buffer	`sharedmem`	`--fw-acc=sw`	support required in software

Table 1. Hardware access command line options

Note: ***For historical reasons software access and channel are denoted with `fw` family of command line options; additionally, the stream software access relates to the type of driver, however in fact it is direct access to software.***

For no access to software command line option `--fw-acc=none` must be used.

When using option `--fw-acc=hw`, software must be built with 1K buffer support.

3.1.2 Stream hardware access

When the hardware access is set to stream (`--hw-acc=stream`), the Control Tool server sends all the requests to the hardware through the software channel. This configuration may be used when direct access to the hardware is not an option (for example, not supported by the target system implementation).

When using stream hardware access, the presence of runnable software is required for using the Control Tool.

In this case:

- all hardware channels are unavailable.
- a software channel is required, this channel can be one of the following: `chardev`, `socket`, `uart`, or `sharedmem`.
- the software can be accessed only via its own channel.

When running the system in this way, hardware registers are controlled through the software which acts like middleware.

Software access	SW channels (--fw-channel)	Command line option	Details
Software Access	`chardev, socket, uart`	`--fw-acc=stream`	support required in software
Software Buffer	`sharedmem`	`--fw-acc=sw`	support required in software

3.2 Channels overview

A channel is an abstract description of physical connection between the Control Tool server and the back-end system. Each channel can be used by enabling a corresponding driver in the Control Tool server. There are two types of channels:

- **Hardware channels**: are used to access the hardware directly and can be used for indirect access to software.
- **Software channels**: are used to access the software directly and can be used for indirect access to hardware.

The following table lists the available channels:

Hardware channels	Software channels
`i2c`	`chardev`
`devmem`	`socket`
`mdcmdio`	`uart` `sharedmem`

Table 2. Available hardware and software channels

- The I2C driver (`--hw-channel=i2c`) is used to communicate with the hardware through a FTDI USB cable.
- The DevMem driver (`--hw-channel=devmem`) is used when the hardware registers are mapped to certain address space in virtual memory.
- The MDCMDIO driver (`--hw-channel=mdcmdio`) is similar to I2C though it allows work via the MDCMDIO protocol.
- The CharDev driver (`--fw-channel=chardev`) provides a way to communicate with the special character device (for example, /dev/ac_isp) created by the software at kernel level.
- The Socket driver (`--fw-channel=socket`) allows communication with software through TCP socket which needs to be supported by the software itself.
- The UART driver (`--fw-channel=uart`) is used to access the software through UART protocol via USB cable (needs to be supported by the software itself).
- The Shared Memory driver (`--fw-channel=sharedmem`) provides communication with the software through a given memory window allocated by software at user-space level.

Channel	Type	Server runs on…	Meaningful command line
i2c	Hardware	Desktop or laptop	--hw-channel=i2c
devmem	Hardware	Target Linux or Android	--hw-channel=devmem

Channel	Type	Server runs on…	Meaningful command line
mdcmdio	Hardware	Desktop or laptop	`--hw-channel=mdcmdio`
socket	Software	Target Linux or Android	`--fw-acc=stream --fw-channel=socket`
chardev	Software	Target Linux or Android	`--fw-acc=stream --fw-channel=chardev`
sharedmem	Software	Target Linux or Android	`--fw-acc=stream --fw-channel=sharedmem`
uart	Software	Desktop or laptop	`--fw-acc=stream --fw-channel=uart`

The driver availability on the supported operating systems is shown below.

Driver	Desktop	Embedded	Windows	Android
i2c	yes	no	yes	no
devmem	no	yes	no	yes
mdcmdio	yes	no	yes	no
socket	yes	yes	yes	no
chardev	no	yes	no	yes
sharedmem	No	Yes	no	no
uart	yes	no	yes	no

There may be three combinations:

- both the hardware and the software are accessed through a single hardware channel,
- the hardware and software are accessed directly and independently via their own channels,
- both the hardware and the software are accessed through a single software channel.

Caution: ***Do not use software drivers for hardware channels.***
Do not use hardware drivers for software channels.

4 Running the control tool server

To start working with the Control Tool you should start the Control Tool server command line application. It should be invoked from command line terminal. The behaviour of the server application is controlled by command line options (parameters).

This section describes the available options. In general, command line usage of the Control Tool server is usually in the following format:

```
./act-server [ <query> | [ [<options>] [<command>] ] ]
```

where query is one of `--help`, `--channels`, `--devices`, or `--version`.

A command is used to set the regime in which the Control Tool server should work. Available commands are as follows:

- listen starts a HTTP server providing access to the server via the web-based GUI (this is the default command).
- direct gives raw command-line access to hardware registers and software API only.

Note: Only one command may be used at a time.

4.1 Command listen

The listen command makes the Control Tool server operate as a HTTP endpoint. It requires:

- a content directory (which is part of the Control Tool distribution).
- an `ArmControl.xml` file which contains information about hardware registers mapping.

To access software API and calibrations you also need a `command.json` file which describes software interfaces.

4.2 Command direct

The direct command turns the Control Tool server into a command prompt which provides direct access to the hardware registers and software API (software calibrations and file transfer are not available).

Note: This regime is recommended for advanced users only.

Running the command direct does not require any Web GUI files, that is, you do not need to have a content directory beside the Control Tool server executable.

After starting a direct command, you may type h (and press **Enter**) to get instant reference about available commands:

```
command (h for help): h

*     general commands:

      - h -  print help message

      - q - quit from application

*     hw r {offset} {size} - read bytes from hardware at <offset>:

*     hw w {offset} {bytes} -  write bytes to hardware at <offset>:

      - offset - address to read from: dec or hex (0x prefix) 32-bit
      number

      - size - number of bytes to read: dec or hex (0x prefix) 32-bit
      number

      - bytes - hex low-case digits: e.g.  01aaff22

*     api r {sec} {cmd} - read from software api

*     api w {sec} {cmd} {value} -  write value to ISP Driver api

      - sec -  section:  8-bit number

      - cmd - command:  8-bit number

      - value - value to set: 32-bit number
```

To quit, type q and press **Enter**.

4.3 Getting basic information

You can use the following commands to display some basic information on the screen.

The Control Tool server displays this information and exits the command.

- A brief explanation of all available command line options can be displayed by invoking:

  ```
  ./act-server --help
  ```

- To list all available channels (drivers) on your platform run:

  ```
  ./act-server --channels
  ```

- To get version information use:

```
./act-server --version
```

4.4 Preset configuration files

The easiest way to use the Control Tool server with certain parameters is to use --preset option with a path to a text file containing settings for the parameters values you wish to use.

```
./act-server --preset=custom-settings.set
```

A preset file has simple key = value format (often referred to as property file format):

```
name-1           = value 1 # comment

another-name-2   = value 2
```

where names of parameters correspond to names of command line options.

Example preset file:

```
http-port     = 8214      # use custom port

# hw-ac       = direct    # not needed as it is default value

hw-channel    = devmem    #   other parameters for this driver are below

hw-mem-offset = 0x7a000000

hw-mem-size   = 0x400000

fw-acc        = none      # disable access to software

log           = error     # suppress most of console log messages except errors
```

Using such a preset file is equivalent to providing the following command line:

```
./act-server --http-port=8214 --hw-channel=devmem --hw-mem-
offset=0x7a000000 --hw-mem-size=0x400000 --fw-acc=none

--log=error
```

Note that commands with the equal to sign and those without the equat to sign are equivalent. For example, the following two commands are equivalent:

```
./act-server --param=value

./act-server  --param  value
```

4.5 Command line options

In most cases there are two channels available:

- The hardware channel to communicate with hardware of back-end system.
- The software channel to work with software which runs on the hardware.

When running the Control Tool server, you should select the drivers for the channels by using command line parameters `--hw-channel` for the hardware channel and `--fw-channel` for the software channel.

For example, to see the list of available devices for a given channel run the cfollowing command:

```
./act-server --hw-channel i2c --devices
```

If you connect your hardware through a FTDI cable you are running the Control Tool on a desktop system (either Windows or Linux). In this case the i2c channel should be used. It is the default channel, so you do not need to specify anything:

```
./act-server

INFO > BusManager:  driver i2c has been successfully initialized in sync
mode... INFO > BusManager: device i2c has been successfully terminated

INFO > ACTServer: FTDI ... device will be used

INFO > HTTPServer: please open http://127.0.0.1:8000/ in firefox or chrome
```

4.5.1 Options to set up access

To set the hardware or software access use the options `--hw-acc` and `--fw-acc` respectively.

For example, to use direct hardware access (default) and access to software via hardware buffer use the following command:

```
./act-server --hw-acc direct --fw-acc hw
```

This sets the Control Tool server to directly access the hardware and to communicate with the software via the hardware buffer by means of a hardware channel. A software channel is not used in this case.

For the `--hw-acc` option the available values are: direct for direct access (default) and stream for access hardware indirectly through a software channel.

The `--fw-acc` option can take one of the following values.

- Use option `--fw-acc=none` to block any attempts to access the software (useful, when software is not supposed to run on the target system).
- Use option `--fw-acc=hw` (along with option `--hw-acc=direct`) to access the software via the hardware buffer (also known as 1K buffer).
- Use option `--fw-acc=sw` (in conjunction with `--hw-acc=direct` or `--hw-acc=stream`) to access the software directly by means of the software buffer (use sharedmem driver)
- use option `--fw-acc=stream` (in conjunction with `--hw-acc=direct` or `--hw-acc=stream`) to access the software directly via the socket, uart, or chardev drivers

When `--fw-acc=hw` software access (that is, access through the hardware buffer) is used for the software you must set the values of the buffer offset and size (in bytes) by using command line options `--hw-buf-offset` and `--hw-buf-size` respectively.

These values may be taken from the `ArmControl.xml` file by the Control Tool client if they are available, but they can be specified manually as well.

For example:

```
./act-server --fw-acc hw --hw-buf-offset 0x888800 --hw-buf-size 1024
```

Note: The offset and size should both be specified in bytes.

4.5.2 Channel options

Some channels require extra parameters to be provided before they can work. Some of these parameters have default values.

All drivers require the name of the device which physically represents the channel.

- For hardware channels use the `--hw-dev` parameter.
- For software channels use the `--fw-dev` parameter.

For example, if you want to use the UART connection via a USB cable, you should invoke the Control Tool server with a command line such as the following:

```
./act-server --hw-acc stream --fw-acc stream --fw-channel uart --fw-dev
/dev/ttyUSB0
```

Note: In most cases the name of the device is inferred automatically. However, in some cases you may need to provide the device name yourself. Referring to system logs may help in such cases.

The following drivers require an address in some type of memory and/or the size of the corresponding memory window:

- `devmem` (hardware driver)
- `mdcmdio` (hardware driver)
- `sharedmem` (software driver)

Use the following options to set this up:

- for hardware drivers use `--hw-mem-offset` and `--hw-mem-size` options (note that options start with --hw-).
- for software driver use `--fw-mem-offset` and `--fw-mem-size` options (note that options start with --fw-).

For example:

```
./act-server --hw-channel devmem --hw-mem-offset 0x7a000000 --hw-mem-size 0x400000
```

Note: The offset and size should be specified in bytes.

The `uart` software driver requires the baud rate to be provided. Use option `--fw-baud-rate` to set this up when connecting to the software through UART channel.

4.5.3 HTTP endpoint options

These options are for the listen command only.

By default, the Control Tool server starts as a HTTP endpoint through which you can access the web-based GUI. There are several options to control the behaviour of the HTTP server.

Use the `--http-port` option to provide a custom port number (the default is 8000).

By default, the Control Tool server listens to all local interfaces.

If this is an issue (for security reasons etc.) you may use the `--http-address` option to specify a certain IP address.

For example, giving the 127.0.0.1 address allows connections only from localhost.

The default value is 0.0.0.0 which means the Control Tool accepts connections from everywhere.

4.5.4 Miscellaneous options

To control the amount of console log output, use the `--log` option which can take one of the following values:

- 0 or n or none: for no logging output.
- 1 or e or error: for showing only errors.
- 2 or w or warn: show both errors and warning only.
- 3 or i or info: show all messages except debug ones (this is the default value).
- 4 or d or debug: used to enable extensive logging.

For example:

```
./act-server --log debug
```

The console log output contains information about errors which can be used to understand the reason of a problem and troubleshooting.

Note: If you want to submit a bug report, please attach the corresponding log output as well.

5 Graphical user interface

The graphical user interface is a Web based thin client. It can be accessed when the Control Tool server runs with the listen command (default). Once started, the Control Tool server prints a URL which may be used to open the GUI web page. By default, you may use the address http://127.0.0.1:8000 if the Control Tool server is run locally (that is, on the user's computer).

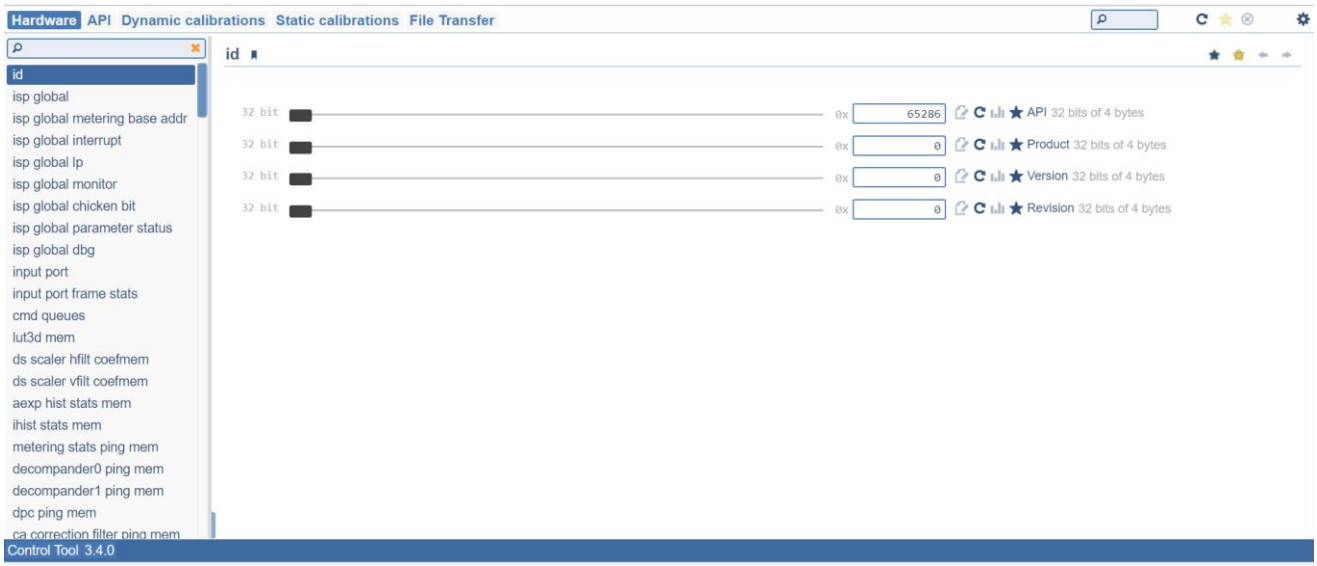
When the Control Tool is accessed for the first time it prompts you to upload an `ArmControl.xml` file. After uploading this file, the normal GUI is displayed.

The settings menu in right upper corner may then be used to load a `command.json` file (use the option Upload new software API). Alternatively, you may activate the tab API which also displays a dialogue box to upload the `command.json` file for the software API.

Both, the hardware XML file and the software API `command.json` file can be replaced in runtime by activating the menu items **Upload New XML** file and **Upload New Software API** respectively.

5.1 GUI layout

The following figure shows the Control Tool user interface layout.



The Control Tool GUI contains a header bar with a tab for each of the available groups of registers and functions that you can access and control. The range of available tabs is determined by which `command.json` file (if any) you upload.

By default, the only available tabs are Hardware (for accessing hardware registers) and API (for working with software API functions). Only the Hardware tab is available when no software API file has been loaded.

The header bar also contains global text search for items and action buttons including the main menu button in the top right corner.

Apart from the Hardware tab, the other tabs and their contents are generated dynamically from the `command.json` file. This affects the menu items present in the main menu as well. Each tab has its own list menu in the left sidebar. Each menu item opens its corresponding section in the main content panel. The menu search bar at the top of each left pane menu can be used to quickly locate menu items by their name in the menu.

5.2 Actions menu

The Actions menus can be accessed by clicking the gear icon in the top right corner.

- **Edit Control Tool UI Settings**: allows you to edit the Control Tool GUI settings (some settings require reload of the web page to take effect).
- **Load HW Registers**: upload the `settings.json` file to apply values from this file to the corresponding hardware registers (values are applied in arbitrary order).
- **Load HW Registers (staggered)**: same as previous item, but the values are applied in order they appear in the file.
- **Save All HW Registers** and **Save Visible HW Registers**: use these actions to save values of hardware registers to a JSON file.
- **Save API Values** and **Save Visible API Values**: use these menu items to save values of the software API.
- **Save Calibrations**: actions are available when `command.json` file is loaded, they may be used to download current values of the software calibrations.
- **Image Converter**: opens SD card converter tool which can be used to convert images from and to different formats.
- **Calibrations to C Source**: opens a utility which provides means for converting software calibrations to a C source file.
- **Calibrations to Binary Data**: opens a utility which allows converting software calibrations to a binary file (the correct `command.json` file is required as well).
- **Pipeline Visualisation**: this menu item opens utility for image pipeline visualization based on the `model.xml` file.

- **Upload New XML file**: upload new `ArmControl.xml` file for accessing hardware registers.
- **Upload New Software API**: upload new `command.json` file for accessing software API, calibrations, and File Transfers.
- **Discard Software API**: drop the currently loaded `command.json` file.
- **Clear User Settings**: discard all local user settings (favourites and the Control Tool UI settings).
- **About Control Tool**: gives information about the Control Tool (through this option you can see the current server configuration as well).

5.3 Sections

In each section the content panel displays the title of the section and a bookmark button followed by a list of the controls for that section.

The bookmark icon near the title of the section can be used to add this section to your favourites in the left side list menu. Items on your favourites list appear at the top of the menu with a bookmark icon beside them. Clicking the favourite menu item's bookmark icon removes the item from the favourites list.

5.4 Controls

A control contains several means that allow the user to interact with a specific control item. The controls may be in a different format depending on their data type and access mode. Each control item provides view and edit (if applicable) access to various controls such as hardware registers, software API functions, software calibrations and file transfers.

5.4.1 Buttons

To the right of each control there are several buttons for interacting with this control. Which buttons are enabled depends on the type of control and its access mode. The complete set of buttons is as follows:

5.4.2 Data types

Data type will determine the format of the input method for the control. There are currently three main data types supported in Control Tool for hardware registers.

- **Array**: These controls are displayed as a list of input boxes with one for each element of the array. The array dimensions are listed beside the input boxes.
- **Look up tables (LUT)**: controls are same as for the Arrays.

- **Bytes**: This data type is displayed as a single input box with a range slider. You can change this value either by moving the slider or by editing the text input box.
- **Bit**: This data type is represented as a checkbox. This is for parameters that can either be switched on or off.

The Software API provides two types of data.

- **Integer value**: This data type is displayed as a single input box with a range slider.
- **Enumeration value**: This type is used for control items which a meant to take only certain named values. The control for each such item is a sequence of radio buttons. Software calibrations are represented as Arrays of values and can be accessed in a way similar to hardware arrays. Software file transfers have buttons for upload and download files.

Note: All array-like controls have the option to display the contents of the array graphically.

5.4.3 Data mode

Each control has a data mode of either read-write (RW), read-only (RO), or write-only(WO). This mode affects the behaviour of the control.

- **Read-Write Mode**: Most of the controls are readable and writable. This means you can edit the current value of the control and save it to the board as well as being able to read the current value from the board.
- **Read Only Mode**: Controls that are set to read only mode can read the current value from the board for their parameter but are not able to edit the value. All input elements are therefore disabled for controls set to this mode.
- **Write Only Mode**: These controls can receive input from the user but cannot read their value from the board. The input elements of the control are all enabled but its reload button is disabled. You may enable and disable support for write-only registers in the Settings.

5.5 Utility extensions

5.5.1 Image converter

You can access the image converter by clicking **Main Menu > Image Converter** on the GUI.

The Image File Converter (available from the Control Tool) is used to convert captured images to PPM (P6, lossless) and JPEG (lossy) formats which then can be opened in any image viewing programme. It is possible to choose the resulting JPEG quality.

The following input formats are supported:

- RGB
- RAW
- YUV

5.5.2 Calibrations converter

You can access the Calibrations Converter by clicking **Main Menu > Calibrations to C Source** on the GUI.

The calibrations converter (available from the Control Tool) is used to convert saved JSON software calibrations to C source code.

5.5.3 JSON to BIN converter

You can access the JSON to BIN Converter by clicking **Main Menu > Calibrations to Binary Data** on the GUI.

The JSON to BIN converter (available from the Control Tool) is used to convert saved JSON calibrations to binary format. You should use proper `command.json` file as well.

5.5.4 Pipeline visualisation

You can access the Pipeline Visualisation by clicking **Main Menu > Pipeline Visualisation** on the GUI.

The pipeline visualisation tool enables you to generate a diagram of an ISP pipeline.

This is done by clicking Choose File and selecting the `model.xml` file for the pipeline you wish to draw (please note this is not the same as `ArmControl.xml` file).

Once you have uploaded the file, click draw to generate the diagram.

Once the pipeline visualisation is generated you can navigate around the pipeline using scroll bars. Use the zoom slider to zoom in and out.

The pipeline visualisation generator automatically selects a main root which is draws the rest of the diagram from. To redraw with a different component as the main root, you can either click the component or enter its ID in to the Main Root ID input and click Draw again.

5.6 Miscellaneous

5.6.1 Force reload hardware configuration

If you need force re-loading of the `ArmControl.xml` file, reload the control tool URL.

For example: http://localhost:8000/reload.

5.6.2 View list of the recent errors

The status bar at the bottom edge of GUI shows status information and enables you to open a list of errors in a pop up window.

5.6.3 View current status of the running server

You can see the current status of the Control Tool by clicking **Right Menu > About** or accessing the `/about` URL.

For example: `http://localhost:8000/about`

This may be useful for troubleshooting to see how exactly the Control Tool is configured.

5.6.4 Running in limited offline mode

You can run the Control Tool in offline mode with the command line

```
./act-server --hw-channel=offline
```

Note: ***The offline mode enables you to work only with hardware registers and software API. Calibrations and file transfers are not supported in the offline mode.***

The offline driver uses debug registers, so if your `ApicalControl.xml` does not have them, you should specify the offset with the `--dbg-offset` switch (with any value):

```
./act-server --hw-channel=offline --dbg-offset=0
```

5.6.5 Getting help about command line options

Use the `--help` switch to obtain a brief description of the options available for running the Control Tool.

5.6.6 Troubleshooting GUI issues

If you are experiencing any issues with the Control Tool GUI then try clearing your local user settings by clicking the gear icon in the top right corner of the screen and selecting Clear User Settings from the main menu.

If you are still experiencing problems then you should try manually removing the local storage to see if this resolves the problem. This can be done by pressing f12 in your browser to open the developer tools, clicking the console tab and typing `localStorage.clear()` into the console.

Note: This clears all other settings set by other web pages which use same URL address as you use for accessing the Control Tool.