

**NAME**

mythen – Dectris Mythen 1K MCA

**INTRODUCTION**

The Dectris Mythen 1K is high-dynamic range, short readout time and high frame rate MCA. `spec` supports single and multiple frame acquisition, optionally gated and or triggered. Frame rates greater than 375 frames per second are possible with `spec`. Connection to the Mythen is over a UDP socket with firmware version 1.x. Either UDP or TCP is available when using Mythen firmware version 2.0 or greater. Multiple units can be configured for use at the same time.

An issue remains with the initial Mythen support in `spec`. During multi-frame acquisition at high frame rates (presets less than 0.1 second), `spec` is locked into a acquisition loop. Currently, not even `^C` will abort the acquisition.

`spec` release 5.09.01-3 added support for most features found in Mythen firmware 2.0. The Mythen can now be configured for either UDP or TCP socket connections. Also, `spec` will display messages associated with the error codes now returned by the Mythen. In addition, `spec` supports the the new Mythen commands for toggling bad-channel interpolation, flat-field correction and rate correction, for setting the dead-time constant *tau*, for loading predefined settings and for optimizing settings to a specified threshold. `spec` also supports the new Mythen command to reset all the settings and parameters.

**CONFIGURATION**

Each Mythen 1K is configured on the MCA- and CCD-type Aquisition screen of the the configuration editor along the following lines:

MCA- and CCD-Acquisition Type Device Configuration

MCA-like	DEVICE	ADDR	<>MODE	<>TYPE
0	YES	mythen	TrimCu	Dectris Mythen (Socket UDP)
1	YES	host:1030		Dectris Mythen (Socket UDP)
2	YES	192.168.0.90	TrimMo	Dectris Mythen (Socket UDP)
3	YES	host:1031	StdCu	Dectris Mythen (Socket TCP)

The `TYPE` can be configured for either UDP or TCP connections (as of `spec` release 5.09.01-3). Mythen firmware below release 2.0 only recognizes UDP connections. The `DEVICE` name is either a resolvable host name or IP address. A port number for the socket connection can be specified. Otherwise, the default port of 1030 will be used.

The `ADDR` field can optionally contain the name of the initial trim or settings file to load. For firmware below version 2.0, the names *TrimCr*, *TrimCu* and *TrimMo* are recognized. If missing, the *TrimCu* trim file will be loaded by default. For firmware at version 2.0 or above, the names *StdCu*, *StdMo*, *HgCr*, *HgCu*, *FastCu* and *FastMo* are recognized by the Mythen. If no name is given, the existing settings on the Mythen will be used, that is, no settings file will be loaded, which will save several seconds for each module.

The `spec` code supports setting the optional controller parameters "inpol" and "outpol" through the configuration editor. Type `p` from the main screen to bring up the optional parameter screen.

**OPERATION**

The Mythen can be started and stopped from `spec` two ways: in parallel with the built-in count functions or manually. If auto-run mode is set with

```
mca_par("auto_run", 1)
```

the MCA will be started and stopped with the standard counting commands used with the `ct-` type macros and with the scan macros. If soft-preset mode is also set with

```
mca_par("soft_preset", 1)
```

the MCA will be programmed with the same count time as the master timer in count-to-time mode. In count-to-monitor mode or if "soft\_preset" is not set, the MCA will be programmed with the preset time set using the `mca_par()` "preset" option below. (Since the Mythen doesn't retain data when stopped using software commands, it is not possible to implement the normal MCA behavior in `spec` where an unset "soft\_preset" means `spec` stops the MCA when the master timer finishes.)

Note, a gate from the master timer or counter can be used to gate the MCA so that the acquisition interval exactly matches the master. If using such a mode, enable gating with the `mca_par()` "gateen" option below.

When using "auto\_run" mode, only one frame will be acquired. In order to acquire multiple frames, use

```
mca_par("run")
```

to start acquisition. As the Mythen can only buffer four data frames, it is essential that `spec` read data as it becomes available. That is done during the automatic hardware polling built into `spec`. It is important not to disable that feature with `spec_par()` or set too long of a polling interval. In addition, one must not run commands during multi-frame acquisition that will interfere with polling, such as creating a subshell with `unix()`.

#### MULTI-FRAME DATA

There are also two ways for `spec` to store the Mythen multi-frame data. In the first method, `spec` will allocate enough storage to hold the data for the number of frames configured. The data can be retrieved using the "frame" or "cframe" options. The "frame" option returns the raw data. The flat-field correction is applied to the data returned using the corrected-frame "cframe" option.

In the second method or reading data, an appropriately configured data array is designated using the "array" option. The data array should be of type `ulong`, the number of columns should be 1280, and the number of rows should be at least as great as the number of frames set with the "frames" option. For example, a data array configured as

```
ulong array data[200][1280]
```

would accommodate 200 frames. The array can also be a shared data array. `spec` will automatically tag the array as "frames" and set the `frame_size` and `latest_frame` elements of the shared data header for the designated array, which will enable optimal display of the data by an auxiliary utility such as *PyMca*.

When the data is placed directly into a data array when using the "array" method, if the "autocorrect" option is enabled, the data will have the flat-field correction applied. If the option is disabled, the raw data is placed in the array.

Both methods for storing the data may be active at the same time. However, to avoid allocating a large amount of memory unnecessarily, if the "array" option is active, `spec` limits the number of frames stored internally, according to the value of the "alloc\_limit" option, which has a default value of 100, a maximum value of 10000 and a minimum value of one.

The standard `mca_get()` function will return the first frame of multi-frame data.

#### PARAMETERS

`spec` can set all of the Mythen 1K operational parameters using the `mca_par()` function, as described in the following section. The parameters "preset", "bits", "frames", "delbef", "delafter", "gateen", "gates", "trigen", "conttrigen", "threshold", "inpol", "outpol", "autocorrect", "alloc\_limit" and "verbose" will be retained in the user state file. For firmware 2.0 or greater, values for the parameters

"badchan\_interpolate", "rate\_correct" and "tau" are also saved in the state file. In addition, "inpol" and "outpol" can additionally be set as optional controller parameters in hardware configuration file. Values set in the configuration file will take precedence to values in the state file.

Before sending the start command to the MCA, any parameters that have been updated will be sent to the Mythen. All parameters will be updated on the next start command after sending a "trimfile" or "settings" command.

## FUNCTIONS

When called with the optional argument, the functions below will set the parameter to the value of the argument. With no arguments, the functions will return the current value of the parameter. It isn't possible for `spec` to read parameter values from the Mythen, so the returned values are those previously set using `mca_par()` during the current session, or the saved values from the user's state file, if not starting fresh.

Note, `spec` does not send commands to change the parameters until the Mythen MCA is started, either using the "run" command or during counting, when "auto\_run" is enabled. In particular, of the following, the only commands that access the Mythen are "reset", "threshold" (only when reading the value), "autosettings", "flatfield", "badchannels", "trimfile", "settings", "send" and "read".

If multiple MCA devices are configured in `spec`, use the `mca_sel()` function to designate the device to use with the `mca_par()` and `mca_get()` functions, or use the `mca_spar()` and `mca_sget()` forms of the functions. See the `mca` help file for more information.

`mca_par("trimfile" [, name])` – Returns the name of the current trim file, or loads the specified trim file. The trim file contains all the initialization parameters for the MCA and is located with the MCA firmware. Each trim file has an associated flat-field file. The flat-field data will be read and stored by `spec` to be used for the optional flat-field correction operation. (Deprecated with version 2.0 firmware.)

`mca_par("flatfield")` – Returns an unsigned long data array containing the flat-field data as read from the detector.

`mca_par("preset" [, seconds])` – Returns or sets the preset value used when manually starting the MCA using the "run" option, below. If the gate is enabled in multi-frame mode, the preset is ignored by the Mythen, but if the frame rate is greater than 10 frames per second, the preset should be set to a value of 0.1 or less so that `spec` will use a more robust method for reading data at high frame rates.

`mca_par("bits" [, value])` – Returns or sets the number of bits per channel for the Mythen to use while acquiring data. Fewer bits allow for slightly faster frame rates in multi-frame mode. Valid values are 4, 8, 16 and 24.

`mca_par("frames" [, number])` – Returns or sets the number of frames to acquire when using a manual start. When using "auto\_run" mode, where the MCA is started and stopped with the master timer, only one frame will be acquired.

`mca_par("latest_frame")` – Returns the frame number of the most recently acquired frame.

`mca_par("delbef" [, seconds])` – Returns or sets the delay between trigger and the first measurement.

`mca_par("delafter" [, seconds])` – Returns or sets the delay between measurements when accumulating multiple frames.

`mca_par("gateen" [, 0|1])` – Returns or sets gated-measurement mode (0 for disabled and 1 for enabled).

`mca_par("gates" [, number])` – Returns or sets the number of gate signals for a gated measurement.

`mca_par("trigen" [, 0|1])` – Returns or sets triggered-measurement mode (0 for disabled and 1 for enabled). A trigger starts the entire multi-frame acquisition or a separate trigger is needed for each frame, depending on the value of the continuous-trigger-enable parameter, "conttrigen".

`mca_par("conttrigen" [, 0|1])` – Returns or sets repeated triggered-measurement mode (0 for disabled and 1 for enabled). With this mode, each frame needs a trigger signal.

`mca_par("threshold" [, value])` – Returns or sets the threshold energy in keV.

`mca_par("inpol" [, 0|1])` – Returns or sets the input polarity for the trigger and gate signals (1 for falling edge and 0 for rising edge). Can be set as an optional parameter in the *config* file.

`mca_par("outpol" [, 0|1])` – Returns or sets the output polarity for trigger and gate signals (1 for active low and 0 for active high). Can be set as an optional parameter in the *config* file.

`mca_par("run")` – Programs the device with updated parameters and starts acquisition.

`mca_par("halt")` – Halts acquisition.

`mca_par("frame", number)` – Returns an unsigned long data array containing the data in frame *number*. Frame numbers start at zero.

`mca_par("cframe", number)` – Returns an unsigned long data array containing the corrected data in frame *number*. Frame numbers start at zero.

`mca_par("array" [, array])` – Returns or sets the name of the data array to be used to receive multi-frame data. The array name can be passed as the array variable or as a string. The array should be type `ulong`, the number of columns should be 1280, and the number of rows should be at least as great as the number of frames.

`mca_par("autocorrect" [, 0|1])` – Returns or sets the mode where data placed directly into the designated data array has the flat-field correction applied. A zero disables autocorrection, while a one enables the mode. With version 2.0 firmware and above, the correction is performed within the Mythen.

`mca_par("autosettings" [, threshold])` – Tells the Mythen to load suitable settings corresponding to the value of *threshold*. Units are keV. If the optional argument is missing, the value set via the "threshold" command is used. If that value is unset, uses the threshold value initially read from the Mythen. (Requires version 2.0 firmware.)

`mca_par("settings" [, name])` – Without the optional argument, returns the name of the current settings designation. Otherwise, sends the Mythen the command to load the settings associated with *name*. (Requires version 2.0 firmware.)

`mca_par("badchan_interpolate" [, 0|1])` – Returns or sets bad-channel interpolation mode. When on, the Mythen replaces values in channels marked as bad with values interpolated from neighboring channels. (Requires version 2.0 firmware.)

`mca_par("rate_correct" [, 0|1])` – Returns or sets rate-correction mode. (Requires version 2.0 firmware.)

`mca_par("badchannels")` – Returns the bad-channels array. (Requires version 2.0 firmware.)

`mca_par("tau" [, value])` – Returns or sets the dead-time constant for rate correction. Units are nanoseconds. (Requires version 2.0 firmware.)

`mca_par("reset")` – Sends the command to reset the Mythen to the default settings. (Requires version 2.0 firmware.)

`mca_par("alloc_limit" [, frames])` – Returns or sets the maximum number of frames to store in addition to the storage used when the "array" option is set (to avoid redundant memory allocation).

`mca_par("verbose", [0|1])` – Returns or sets the verbose flag. When set, spec will display an updated frame count when acquiring multiple frames.

`mca_par("flush")` – Flushes the sockets associated with the connection to the Mythen. This command should not be needed under normal operation.

`mca_par("dump")` – Displays the current values of the Mythen parameters. The following example assumes version 2 firmware:

```

Dectris Mythen 1K SN055
  Settings type ("settings"): StdCu
    Preset time ("preset"): 1 sec
    Bits per channel ("bits"): 24
    Frames ("frames"): 1
    Array for data ("array"): -none-
  Autocorrect ("autocorrect"): disabled
  Alloc. limit ("alloc_limit"): 100 frames
    Verbose ("verbose"): disabled
    Delay before ("delbef"): 0 sec
    Delay after ("delafter"): 0 sec
    Gate ("gateen"): disabled
    Gates per frame ("gates"): 1
    Trigger ("trigen"): disabled
  Trigger repeat ("conttrigen"): disabled
    Threshold ("threshold"): 6.64962 keV
    Input polarity ("inpol"): falling edge (0)
    Output polarity ("outpol"): active high (0)
  Bad channel interpolation ("badchan"): disabled
  Rate correction ("ratecor"): disabled
  Dead time constant ("tau"): 170 nsec

  Auto-run mode ("auto_run"): Off (0)
  Soft-preset ("soft_preset"): Off (0)

```

Note, the parameter values are those spec will use the next time it starts the Mythen. The values may differ from those currently existing on the Mythen device.

`mca_par("send", string)` – Sends the contents of *string* to the Mythen. (Normally not used, available for debugging.)

`mca_par("read", string)` – Sends the contents of *string* to the Mythen and returns the integer response. (Normally not used, available for debugging.)

#### SEE ALSO

mca  
[www.dectris.com](http://www.dectris.com)  
[pymca.sourceforge.net](http://pymca.sourceforge.net)