General Responses:

@error, "Missing one or more necessary parameters. <command name> command failed.";
Note that commas and semicolons between a pair of double quotes will be ignored. <command name> is a formal name of the command. ("Sync Time Date" instead of "sync_time_date", for example.)
@error, "Unrecognized command [<command>]";
The command is not recognized. <command> will be the first token specified from between the at sign (@) and the first comma.

Play Schedule

Format:
@play_schedule,<schedule file name>;
Description:
Plays the schedule specified by the first parameter, which is a file name. The file name should contain no path information, should be a file with the ".shd" extension, and can optionally be contained within double quotes. The file should already exist in the Schedules folder on the MX7 machine. Anything else that was previously playing stops.
Example:
@play_schedule,"demo.shd";
Responses:
@ok;
@error, "Confirm.txt could not be opened. Play Schedule command failed.";
@error, "Specified file is not a schedule. Play Schedule command failed.";

Play Set

Format:
@play_set,<set file name>;
Description:
Plays the set specified by the first parameter, which is a file name. The file name should contain no path information, should be a file with the ".set" extension, and can optionally be contained within double quotes. The file should already exist in the Sets folder on the MX7 machine. Anything else that was previously playing stops.
Example:
@play_set,"demo.set";
Responses:
@ok;
@error, "Confirm.txt could not be opened. Play Set command failed.";
@error, "Specified file is not a set. Play Set command failed.";

Play Media

Format:
@play_media,<media file name>;
Description:
Plays the video clip specified by the first parameter, which is a file name. The file name should contain no

path information and can optionally be contained within
double quotes. Anything else that was previously
playing stops.
Example:
@play_media,"Nature Scene.mpg";
Responses:
@ok;
@error, "Confirm.txt could not be opened. Play Media command failed.";
@error, "Specified file is not a recognized media file. Play Media command failed.";

Stop

Format:
@stop;
Description:
Stops whatever is currently playing, and simply displays
a black screen.
Example:
@stop;
Responses:
@ok;
@error, "Confirm.txt could not be opened. Stop command failed.";

Get Media List

Format:
@get_media_list;
Description:
Requests that the MX return a list of all files
that exist in the machine's Media directory.
Example:
@get_media_list;
Response:
@media_list,<file name>,<file name>, ... , <file name>;
Response Details:
Returns a command named "media_list", followed by a
comma-separated list of each file name. Each file is
contained in double quotes.
Response Example:
@media_list,"Lake.mpg","Forest.mpg","Plains.mpg";

Get Set List

Format:
@get_set_list;
Description:
Requests that the MX return a list of all files
that exist in the machine's Sets directory.
Example:
@get_set_list;
Response:
@set_list,<file name>,<file name>, ... ,<file name>;
Response Details:
Returns a command named "set_list", followed by a
comma-separated list of each file name. Each file is
contained in double quotes.
Response Example:
@set_list,"demo.set","Saturday.set","Sunday.set","Failsafe.set";

Get Schedule List

Format:

```
@get_schedule_list;
```

Description:

Requests that the MX return a list of all files that exist in the machine's Schedules directory.

Example:

```
@get_schedule_list;
```

Response:

```
@schedule_list,<file name>,<file name>, ... ,<file name>;
```

Response Details:

Returns a command named "schedule_list", followed by a comma-separated list of each file name. Each file is contained in double quotes.

Response Example:

```
@schedule_list,"Spring.shd","Summer.shd","Fall.shd","Winter.shd";
```

Get Active

Format:

```
@get_active;
```

Description:

Requests the file name of the file that is currently being played, if any.

Example:

```
@get_active;
```

Response:

```
@active_file,<type>,<file name>;
```

Response Details:

Returns the type of the active file (the first parameter) and the file name of the active file (the second parameter). The type can be one of the following (without double quotes): "media", "set", "schedule", or "none". The file name is contained in double quotes, and as with all other file names, does not contain any path information, just the file name itself. If the type is "none", then the file name parameter will simply be empty double quotes ("").

Response Example:

```
@active_file,set,"Saturday.set";
```

Response Example:

```
@active_file,none,"";
```

Version

Format:

```
@version;
```

Description:

Requests the current version of the backend software.

Example:

```
@version;
```

Response:

```
@ok, 6.44;
```

Login

Format:

```
@login,<password>;
```

Description:

Checks the password against one stored on the backend

machine, and approves or rejects the login request.
Example:
@login, badpassword;
Responses (remove quotes):
" V "
" EBP "

Logout
Format:
@logout;
Description:
Example:
@logout;
Responses (remove quotes):
" V "

Get Schedule File
Format:
@get_schedule_file,<filename>;
Responses:
Initial Response:
@send_file,"<filename>";
Waits for further command:
@ready;
Sends File using the Zmodem protocol
Error Responses:
@error, "File '<filename>' could not be opened.";
@error, "Specified file is not a schedule. Get Schedule File command failed.";

Get Set File
Format:
@get_set_file,<filename>;
Responses:
Initial Response:
@send_file,"<filename>";
Waits for further command:
@ready;
Sends File using the Zmodem protocol
Error Responses:
@error, "File '<filename>' could not be opened.";
@error, "Specified file is not a set. Get Set File command failed.";

Get Media File
Format:
@get_media_file,<filename>;
Responses:
Initial Response:
@send_file,"<filename>";
Waits for further command:
@ready;
Sends File using the Zmodem protocol
Error Responses:
@error, "File '<filename>' could not be opened.";
@error, "Specified file is not a media file. Get Media File command failed.";

Get Event File

```
    Format:
        @get_event_file,<filename>;
    Responses:
        Initial Response:
            @send_file,"<filename>";
        Waits for further command:
            @ready;
        Sends File using the Zmodem protocol
        Error Responses:
            @error, "File '<filename>' could not be opened.";
```

Get Settings File

```
    Format:
        @get_settings_file,<filename>;
    Responses:
        Initial Response:
            @send_file,"<filename>";
        Waits for further command:
            @ready;
        Sends File using the Zmodem protocol
        Error Responses:
            @error, "File '<filename>' could not be opened.";
```

Get File

```
    Format:
        @get_file,<filename>;
    Responses:
        Initial Response:
            @send_file,"<filename>";
        Waits for further command:
            @ready;
        Sends File using the Zmodem protocol
        Error Responses:
            @error, "File '<filename>' could not be opened.";
```

Send Schedule File

```
    Format:
        @send_schedule_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
            @error, "Failed to receive '<filename>'";
            @error, "Specified file is not a schedule.  Send Schedule File
command failed.";
```

Send Set File

```
    Format:
        @send_set_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
```

```
            @error, "Failed to receive '<filename>'";
            @error, "Specified file is not a set.  Send Set File command
failed.";
```

Send Media File

```
    Format:
        @send_media_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
            @error, "Failed to receive '<filename>'";
            @error, "Specified file is not a media file.  Send Media File
command failed.";
```

Send Event File

```
    Format:
        @send_event_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
            @error, "Failed to receive '<filename>'";
```

Send Settings File

```
    Format:
        @send_settings_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
            @error, "Failed to receive '<filename>'";
```

Send File

```
    Format:
        @send_file,<filename>;
    Responses:
        Initial Response:
            @ready,"<filename>";
        Receives File using the Zmodem protocol
        Error Responses:
            @error, "Failed to receive '<filename>'";
```

Remove Unused Media

```
    Format:
        @remove_unused_media;
    Description:
        Goes through the entire \Media folder, looking for files
        that are not in the currently playing schedule,
        and deletes them.  Presently, it only does so for files
        that are considered some form of video (MPEGs, WMVs,
        Flash, etcetera).  Still images (as well as animated
        GIFS)are ignored.
    Example:
```

```
        @remove_unused_media;
    Responses:
        @ok;
```

Start Instant Crawl

```
    Format:
        @start_instant_crawl,<filename>,<dwell>,<color-
key>,<transparency>,<speed>,<location>,<background>;
    Description:
        Starts a crawl instantly, hiding all other crawls and
        rolls, using the Rich Text Format file specified as the
        crawl text with formatting.  The file should be located
        in the Media folder.  This crawl will run for up to
        <dwell> seconds, or until a @STOP_INSTANT_CRAWL instant
        event is given.  It uses <color-key> as the color to
        turn into 100% transparency, in the format 0xBBGGRR,
        where BB is blue in hexadecimal (00 to FF), GG is green,
        and RR is red.  <transparency> is a number between 0.0
        and 1.0, indicating how translucent the overall crawl is.
        0.0 is fully opaque, and 1.0 is fully translucent.
        <speed> is an integer that describes the number of pixels
        that the crawl moves per frame.  Specify a negative number
        to get left-to-right movement.  <location> is a number
        between 0.0 and 1.0 that describes the vertical location of
        the crawl, 0.0 being the top of the screen and 1.0 being
        the bottom of the screen.  <background>, specified similarly
        to <color-key>, specifies the color to use behind the text.
        If <background> and <color-key> are the same, then the
        background will be fully transparent.
    Example:
        @start_instant_crawl,Tornado.rtf,600,0x000000,0.00,3,1.0,0x000000;
    Example:
        @start_instant_crawl,Go Home.rtf,14400,None,0.00,3,1.0,0x0000FF;
    Responses:
        @ok;
```

Stop Instant Crawl

```
    Format:
        @stop_instant_crawl;
    Description:
        Stops an instant crawl, if one is currently playing.
    Example:
        @stop_instant_crawl;
```

Start Instant File

```
    Format:
        @start_instant_file,<filename>,<dwell>;
    Description:
        Starts playing a file instantly, and continues to play it
        until its dwell time ends or a @STOP_INSTANT_FILE event
        is given.  The file can be a schedule file, a set file, or
        any valid media file, and should already be on the machine
        in its proper folder.  The currently playing schedule will
        be completely stopped and replaced by this new instant
        file, and when this instant file stops, the previous
        schedule will resume playing from the very beginning.  The
        dwell time is in seconds; if this number is 0 or -1, then
```

the instant file will play indefinitely.
Example:
@start_instant_file,Flooding.set,-1;
Example:
@start_instant_file,Hello.mpg,300;
Responses:
@ok;

Stop Instant File
Format:
@stop_instant_file;
Description:
Stops an instant file, if one is currently playing, and resumes whatever had been playing previously, starting from the beginning.
Example:
@stop_instant_file;
Responses:
@ok;

Instant Event Command Serial
Format:
@send_serial_command,<com-port>,<flow-control>,<baud-rate>,<parity>,<data-bits>,<stop-bits>,<data>;
Description:
Commands the backend to immediately send out a serial event. The details of the <data> parameter are complex. That information will be detailed elsewhere.
Responses:
@ok;

Instant Event Command TCP
Format:

@send_tcp_command,<port>,<address>,<ignored>,<ignored>,<ignored>,<ignored>,<data>;
Description:
Commands the backend to immediately send out a TCP message. The details of the <data> parameter are complex. That information will be detailed elsewhere.
Responses:
@ok;

Instant Event Command UDP
Format:

@send_udp_command,<port>,<address>,<ignored>,<ignored>,<ignored>,<ignored>,<data>;
Description:
Commands the backend to immediately send out a UDP message. The details of the <data> parameter are complex. That information will be detailed elsewhere.
Responses:
@ok;

Reboot Machine
Format:

```
        @reboot_machine;
    Description:
        This command will terminate the program, and reboot the
        machine.  The program should be in the Startup directory
        of the Start Menu, and should resume like normal after
        the bootup process.
    Example:
        @reboot_machine;
    Responses:
        @ok;
```

Restart MX5

```
    Format:
        @restart_mx5;
    Description:
        Stops the backend executable, if it is running, and then
        starts it up again (regardless of if it was previously
        running or not.)
    Example:
        @restart_mx5;
    Responses:
        @ok;
```

Manage Cache

```
    Format:
        @manage_cache;
    Description:
        Will manage commands and files associated with those
        in the Cache directory.  This is a somewhat complex
        topic and will be detailed elsewhere.
    Example:
        @manage_cache;
    Responses:
        @ok;
```

Sync Time and Date

```
    Format:
        @sync_time_date,<month>/<day>/<year> <24-hour>:<minute>:<second>;
    Description:
        Sets the backend system's time and date.  All parameters
        are numbers; no month names allowed.
    Example:
        @sync_time_date,12/06/1982 23:49:13;
    Responses:
        @ok;
```

Display Power

```
    Format:
        @display_power,<state>;
    Description:
        Sets the display power to be either on or off by toggling sync. Valid
        values for <state> are literally either on or off.  If
        anything else is specified, a state of on is assumed.
    Example:
        @display_power,on;
    Example:
        @display_power,off;
```

```
Responses:
    @ok;
```