



SUPPORT

INTERFACING WITH MATLAB THROUGH RS232

This application note both explains how to configure the power supply for external MATLAB programming and provides a basic scripting example in MATLAB.

Magna-Power Electronics' power supplies can easily integrate into [MathWorks MATLAB](#) for quick script deployment. Scripts written in MATLAB can be compiled to run outside MATLAB as long as the target computer has the [MATLAB Component Runtime](#) installed.

To enable RS232 control on the power supply, the power supply must be configured for External Programming mode (EXT PGM). To set the power supply in external programming mode on either the front panel or RIS panel, using the following steps:

1. While in standby, hit menu on the front panel; "conF" will flash in the voltage display.
2. Hit enter while "conF" is displayed. The "REM SEN" LED will now flash on the right hand side of the power supply.
3. Toggle through the configuration modes using the item button until "EXT PGM" LED flashes.
4. Hit enter while the "EXT PGM" led is flashing to enable external programming.

Magna-Power Electronics' products use Standard Commands for Programmable Instrumentation (SCPI commands) over RS232 for intuitive control over the power supply. A full listing of available commands can be found in section 4.3 of the manual. The following is an example of a MATLAB battery charging test script used to control the power supply:

```
% Magna-Power Electronics, Inc.
% Example MATLAB Integration - Battery Charging Test Script

function battery_test

clear all;
% Print text to command window
display('Now running Battery Test Script');
display('=====');

% Create a serial port object. COM port may vary.
obj1 = serial('COM1', 'BaudRate', 19200);
% Create array to store voltage readings
voltarr = zeros(21,1);
% Connect to created power supply object, obj1
fopen(obj1);

currdate = {date};

fileloc = ['C:\Test Data\battery_test.xls'];

% Send command to the power supply, obj1
% *IDN? verifies power supply model number and serial
fprintf(obj1, '*IDN?');
% Receive response from the power supply, obj1
idn = fscanf(obj1);
```

```

% Print text to command window
display(' ');
display('VOLT 34');
display('CURR 6');

% Set the voltage of the power supply, obj1
fprintf(obj1, 'VOLT 34');
% Set the current of the power supply, obj1
fprintf(obj1, 'CURR 6');

display('OUTP:START');

% Start the power supply output, obj1
fprintf(obj1, 'OUTP:START');

display('Wait 5 min');

pause(300);

display('VOLT 0');

% Set the voltage of the power supply, obj1
fprintf(obj1, 'VOLT 0');

% Measure the voltage at the output of the power supply for 20 mins
for i=1:21
    fprintf(obj1, 'MEAS:VOLT?');
    currvolt = str2num(fscanf(obj1));
    display('MEAS:VOLT? = '+currvolt);
    voltarr(i,1) = currvolt;
    display('Wait 1 min');
    pause(60);
end

% Stop the output of the power supply, obj1
fprintf(obj1, 'OUTP:STOP');
display('STOP');

% Write data to an excel file

xlswrite(fileloc,voltarr,'B19:B39');
xlswrite(fileloc,currdate,'F4:F4');

display(['Data written to: ',fileloc]);

% Disconnect from power supply, obj1
fclose(obj1);
fclose(fid);

% Clean up all objects.
delete(obj1);

```

After you have written and tested the script, use the following command in the MATLAB command window to compile as a MATLAB stand-alone application: `mcc -m filename.m`