A31315 DLL PROGRAMMING USING MATLAB ON WINDOWS OS

By Vijay Peddoju
Allegro MicroSystems

INTRODUCTION

This application note describes procedures for programming the A31315 DLL using MATLAB on Windows OS.

Hardware Required

- Sensor Test Kit: ASEK-20
- Daughter board: TED-0002820
- Granddaughter board, either:
  - □ TED-0002931 for 8-pin SOIC (single-die version); or
  - □ TED-0002932 for 14-pin TSSOP (dual-die version)
- Sensor: A31315 device with the option for SENT/PWM or analog signal output
- PC with USB connection

PROCEDURE

When setting certain parameters in the MATLAB program, it is helpful to know certain application-specific and A31315 device-specific information, such as output mode (analog, SENT/PWM) and axis selection (XY, XZ).

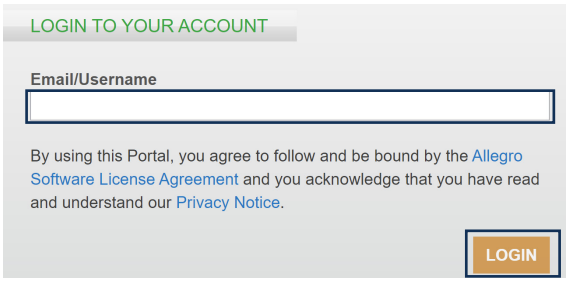
Perform each portion of this procedure in order:

1. Check/Update the ASEK-20 firmware.
2. Download the A31315 DLL.
3. Check the hardware connections.
4. Program the device using MATLAB.

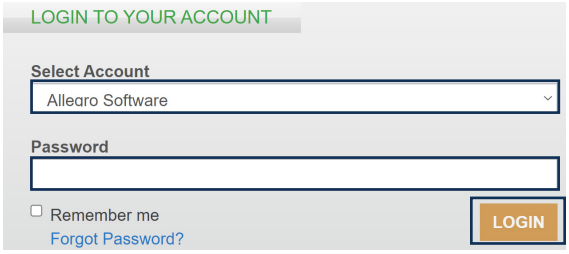
Check/Update ASEK-20 Firmware

The firmware on the ASEK-20 must be compatible with the software on the A31315. This may not be the case, even if the products were purchased together. Check for firmware/software updates and download the latest firmware/software as provided in this section.

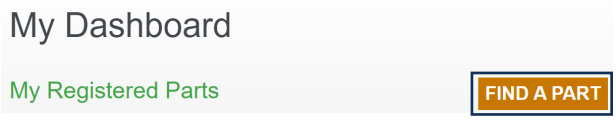
1. If not already registered, create an account at https://registration.allegromicro.com/login.
2. Login to https://registration.allegromicro.com/login.
   A. Enter your email/username and click "login"



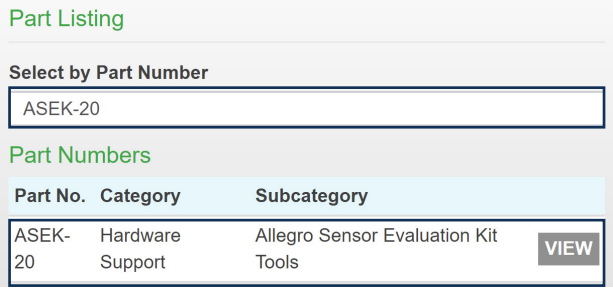
   B. In the "Select Account" field, select "Allegro Software"; then enter your password and click "login".



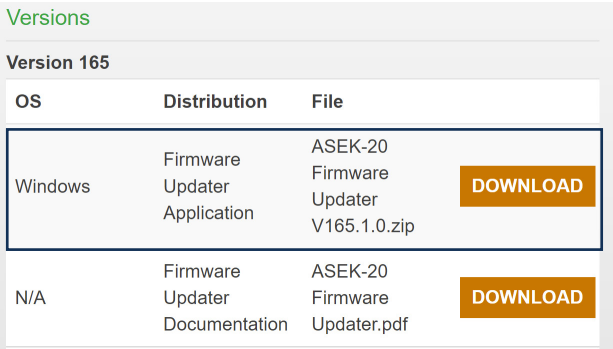
3. Click on "Find a Part".



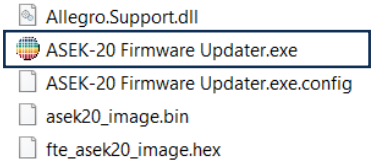
4. In the "Select By Part Number" field, type "ASEK-20", then click "View".



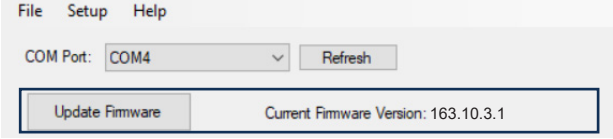
5. Download the most recent version of the "Firmware Updater Application" file (ZIP file).



6. Extract the downloaded files and run the Firmware Updater Application (EXE file).



7. The current firmware version will be shown on the Firmware Updater Application GUI. If the "Current Firmware Version" is not sufficient, click "Update Firmware".

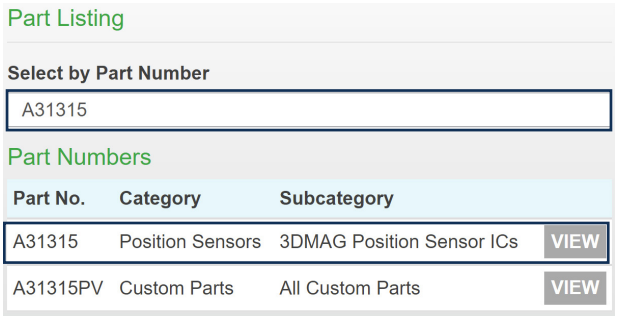


Download A31315 DLL and Programming Application

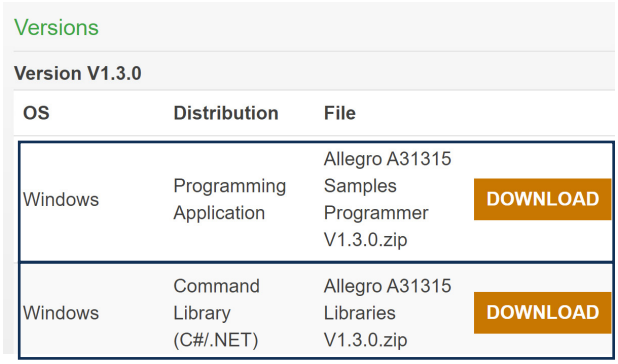
Download the A31315 DLL (and programming application, if needed) from the Allegro MicroSystems website as follows:

8. Log in to https://registration.allegromicro.com/login, and navigate to the "Available Parts" as provided in Step 1 through Step 3 in "Check/Update ASEK-20 Firmware".

9. In the "Select By Part Number" field, type "A31315", then click "View".



10. Download the most recent version of the "Command Library" file (ZIP file); if not already installed, also download the "Programming Application" file (ZIP file).



11. Extract and save the files:

    - Save the DLL files to the location on the PC where the MATLAB script files will be stored.

Check Hardware Connections

12. Ensure the granddaughter board is applied to the daughter board in the correct orientation:
    - □ When the sensor is being inserted into the socket, the round mark that indicates Pin 1 must be oriented to the top left corner and the writing on both boards must be aligned in the same orientation (otherwise the boards will not connect).
    - □ If the orientation of the device is incorrect, an ASEK error will be shown in the command prompt section of MATLAB while running the MATLAB program; however, this will not damage the device.
    - □ Do not unplug the daughter or granddaughter board from the ASEK-20 while it is powered.

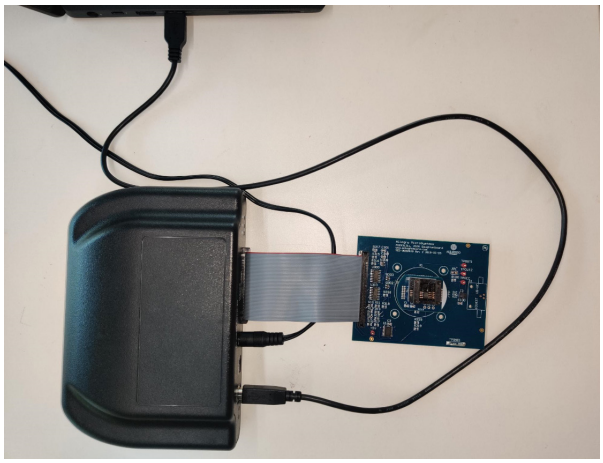


Figure 1: Hardware setup.

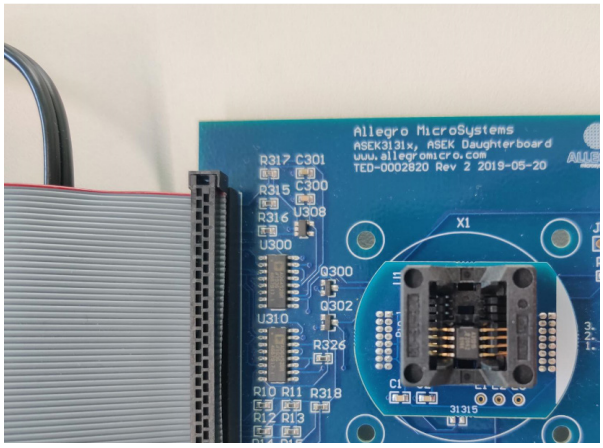


Figure 2: Daughter board (TED-0002820) and granddaughter board (TED-0002932, shown here).

Program the Device Using MATLAB

13. Before running the MATLAB script:
    A. Copy the code on the following pages into a MATLAB script in the same folder as the DLL files. (The full MATLAB code is also included as an appendix following these instructions.)
    B. Set the COM port in the script. (Use either the device manager or the MATLAB "seriallist" command to find the COM port).
14. Connect the ASEK-20 with a USB cable to a Windows PC.
15. Power the ASEK-20 with a 5 V power supply.
16. Initialize setup as follows:
    A. Select the communication method.
    B. Select the A31315 socket option.
    C. Select the used COM port.

```
%% Script to showcase the A31315 DLL communication
% Author: Vijay Peddoju
% Co-Author: Till Ostermann
% Date: 2021-5-14
% Software prerequisites:
% - Matlab (tested with R2020a)
% - requires the Windows Command Library V0.7.1 or higher (download via
% registration.allegromicro.com after registration)
% - ASEK20.dll, ASEKBase.dll and ASEK20_A31315.dll files need to be accessible
% Hardware prerequisites:
% - ASEK20 with a firmware 164.10.3.1 or latest version(firmware updater download
% available via registration.allegromicro.com after registration)
% - Connected TED-0002820 daughter board
% - Connected A31315 device (using grand daughter board)
% ASEK Initialization: might need to close Matlab and restart in case of any
%failure

close; clear; clc;

% User selectable parameters
% The user can choose the communication method to the A31315 device. There
% are two ways to communicate with the device:
% Over voltage and over drive

%% ASEK initialization

ASEK20 = NET.addAssembly( [which('ASEK20_A31315.dll')] );
A31315 = ASEK20.AssemblyHandle.CreateInstance( 'Allegro.ASEK.ASEK20_A31315' );

%seriallist function shows a list with available COM-Ports

ASEK_com = 'COMxx';
A31315.SetCommunicationPort(ASEK_com); % Set Port where ASEK is connected
```

17. Set up the proper protocol for communication between the ASEK-20 and the A31315 by specifying the relevant parameters. In this example code, Manchester communication between the ASEK-20 and the A31315 has been configured with support for the analog output mode. In the case of a long wire harness or a large capacitance on V_{out}, it can be beneficial to reduce the Manchester communication speed. The minimum available speed is 4000 (bit/s).

```
% Setup the device with the proper protocol for communication

A31315.IsAnalogOutput = true;
A31315.SetNumberOfDie(1);
A31315.InitializeDeviceManchester();

A31315.CommunicationEnableMethod = 3; % 0 = overvoltage, 3 = overdrive
A31315.CommunicationEnableVoltage = 10.300;
A31315.SetManchesterHighVoltage(4);
A31315.SetManchesterLowVoltage(0.3);
A31315.ASEK20InternalCapacitor =1;
A31315.SetManchesterInputSamplingThreshold(1.5);
A31315.SetManchesterCommunicationSpeed(4000);


A31315.SetVcc(5.0,  hex2dec('86'), hex2dec('8298A6B6'));
disp( ['Init TED-0002820 and A31315 successful. Connected on connected via ' ASEK_com '.']);
```

18. Read the temperature and angle data from the sensor EEPROM and enable the appropriate output mode (analog of SENT/PWM).

```
%% Measure section
% Read temperature form sensor

temp = (double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary,...
    hex2dec('A7'), 31, 16))  / 128) + 25;

% read digital angle output

angle_dig =
double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary, ...
    hex2dec('A7'), 15, 0));
angle_dig_0_360 = angle_dig * 360.0 / (2^16);

% read digital 1D channels
fi_number = fi(0,1,16,15)
fi_number.hex = dec2hex( A31315.Read_volatile_chan_a_16b );
ChanA = fi_number.double*1000 % 1000G device
fi_number.hex = dec2hex( A31315.Read_volatile_chan_b_16b );
ChanB = fi_number.double*1000 % 1000G device
```

19. Read the digital angle value: The "ana_range_sel" parameter, which specifies the DAC scaling, depends on the load circuitry. The formula in the code determines the analog angle value from 0 degrees to 360 degrees.

```
%read analog angle output for A31315LOLATR-**-S-AR-10 devices
%Check the device type to set it's output mode.

angle_ana = A31315.ReadOutputVoltage();
vcc_ana = A31315.GetVcc();

ana_range_sel = A31315.Read_eeprom_ana_range_sel();
low_clamp_percent = [0 4 5 6 7 8 10 15];
high_clamp_percent = [100 96 95 96 93 92 90 85];

angle_ana_0_360 = (angle_ana - vcc_ana * low_clamp_percent(ana_range_sel+1)/100) / ...
    ((high_clamp_percent(ana_range_sel+1) - ...
    low_clamp_percent(ana_range_sel+1))/100 * vcc_ana ) * 360;
```

20. Read the angle from the sensor and plot a graph in MATLAB.

```
%% Demo, read digital output

angle_out = [];

% Set high speed mode

A31315.HoldCommunicationEnable=true;

for i=1:50

    angle_out(i) =
double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary, ...
        hex2dec('A7'), 15, 0));
    angle_out(i) = angle_out(i) * 360.0 / (2^16);
    plot(angle_out);
    drawnow

end

% End high speed mode

A31315.HoldCommunicationEnable=false;
```

21. Write to the device by setting the parameters for the output modes (analog or SENT/PWM). For more information regarding these parameters, refer to the datasheet.

```
%% Set analog output mode parameters
% applies only for A31315LOLATR-**-S-AR-10 devices

if true
    A31315.IsAnalogOutput = true;
    A31315.Write_eeprom_ana_range_sel(6) % Refer data sheet for more ranges
    A31315.Write_eeprom_ana_bw_sel(1)
    %15kHz, 1: 30kHz
    A31315.Write_eeprom_bw_sel(0)
end


%% Set SENT/PWM output parameters
% applies only for A31315LOLATR-**-S-SE-10 devices

if true
    A31315.Write_eeprom_dig_out_sel(2)
    A31315.Write_eeprom_dig_out_data_rate(0)
    % Set frame rate when using SENT (no effect on PWM)
    A31315.Write_eeprom_sent_frame_rate(2)

end
```

22. Access the GUI by running the "Allegro A31315 Samples Programmer.exe" file. (If not previously installed, this file was downloaded and extracted in Steps 10 and 11).
23. Reprogram the device by loading all parameters from the A31315 GUI to a CSV file.

```
%% Write EEPROM dump to device

if true
    % Adjust the location of a potential CSV file (same as exported from GUI)
    % to upload it with the script to the device.
    % Load EEPROM to a CSV file, only use if the device should be reprogrammed
    % CSV file must have the same format as the csv export from the A31315 GUI

    EEP = readtable([pwd '\EEP_31315.csv']);

    % Set high speed mode
    A31315.HoldCommunicationEnable=true;

    %Parameters in Registry 0x15,0x3c,0x3D,0x3E, 0x3F
    %which are not included in the present DLL, hence not programmable

    N = {'chan_a_dis','chan_b_dis','chan_orthog', ...
        'spare_0x15','spare_0x21','spare_0x3b','spare_0x3c','spare_0x3d', ...
        'spare_0x3e','spare_0x3f'} ;

    % Program EEPROM to device

    for idx = 1:length(EEP.Var1)
        if ~(strcmp(EEP.Var1{idx},N))
            A31315.(['Write_eeprom_' EEP.Var1{idx}])(EEP.Var2(idx));
            disp([ 'Program: ' EEP.Var1{idx} ])
        else
            disp([ 'Not program: ' EEP.Var1{idx} ])
            continue
        end
    end

    % End high speed mode

    A31315.HoldCommunicationEnable=false;
end
```

24. Read the parameters that are to be reprogrammed and change their values as needed.

```
%% Read write partial registers and re-program the parameters by changing their value

if true

    % For reading full register
    % example: A31315.ReadMemory(A31315.MemoryAccessType_Primary, hex2dec('37'))
    % For reading partial register

    Angle_gain = A31315.ReadPartialRegister(A31315.MemoryAccessType_Primary, hex2dec('33'), ...
        15,0);

    % Write ang_gain (default 1), To be checked again
    % read first old value

    ang_gain = fi(1,0,16, 10);
    ang_gain.dec = num2str( A31315.ReadPartialRegister(A31315.MemoryAccessType_Primary, ...
        hex2dec('33'), 15,0) );

    ang_gain.double=1;
    A31315.WritePartialRegister(A31315.MemoryAccessType_Primary, hex2dec('33'), ...
        str2num( ang_gain.dec ), 15,0)
    A31315.CloseCommunicationPort
end
```

APPENDIX: FULL MATLAB SOURCE CODE

```
%% Script to showcase the A31315 DLL communication
% Author: Vijay Peddoju
% Co-Author: Till Ostermann
% Date: 2021-5-14
% Software prerequisites:
% - Matlab (tested with R2020a)
% - requires the Windows Command Library V0.7.1 or higher (download via
% registration.allegromicro.com after registration)
% - ASEK20.dll, ASEKBase.dll and ASEK20_A31315.dll files need to be accessible
% Hardware prerequisites:
% - ASEK20 with a firmware > 163.1.0 (firmware updater download available via
% registration.allegromicro.com after registration)
% - Connected TED-0002820 daughter board
% - Connected A31315 device (using grand daughter board)
% (WKUP set to high or lpm_en_c=0)
% ASEK Initialization: might need to close Matlab and restart in case of a failure

close; clear; clc;

% User selectable parameters
% The user can choose the communication method to the A31315 device. There
% are two ways to communicate with the device:
% Over voltage and over drive

%% ASEK initialization
% Configure ASEK connection
% must explicitly provide full path to DLL on net.addAssembly command

ASEK20 = NET.addAssembly( [which('ASEK20_A31315.dll')] );
A31315 = ASEK20.AssemblyHandle.CreateInstance( 'Allegro.ASEK.ASEK20_A31315' );

% seriallist function shows a list with available COM-Ports

ASEK_com = 'COMxx';
A31315.SetCommunicationPort(ASEK_com); % Set Port where ASEK is connected

% Setup the device with the proper protocol for communication
% Configure Manchester communication between ASEK-20 and A31312 device
% In case of long wire harnesses or a large capacitance on Vout, it can be
% beneficial to reduce the Manchester communication speed. The minimum
% available is 4000 (bit/s).

A31315.IsAnalogOutput = true;
A31315.SetNumberOfDie(1);
A31315.InitializeDeviceManchester();

A31315.CommunicationEnableMethod = 3; % 0 = overvoltage, 3 = overdrive
A31315.CommunicationEnableVoltage = 10.300;
A31315.SetManchesterHighVoltage(4);
A31315.SetManchesterLowVoltage(0.3);
A31315.ASEK20InternalCapacitor =1;
A31315.SetManchesterInputSamplingThreshold(1.5);
A31315.SetManchesterCommunicationSpeed(4000);

% Unlock and give message for successful unlock
% Set voltage and send unlock code
A31315.SetVcc(5.0,  hex2dec('86'), hex2dec('8298A6B6'));
disp( ['Init TED-0002820 and A31315 successful. Connected on connected via ' ASEK_com '.']);

%% Measure section
% Read temperature form sensor

temp = (double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary,...
    hex2dec('A7'), 31, 16))  / 128) + 25;

% read digital angle output

angle_dig =
double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary, ...
    hex2dec('A7'), 15, 0));
angle_dig_0_360 = angle_dig * 360.0 / (2^16);
```

```
% read digital 1D channels
fi_number = fi(0,1,16,15)
fi_number.hex = dec2hex( A31315.Read_volatile_chan_a_16b );
ChanA = fi_number.double*1000 % 1000G device
fi_number.hex = dec2hex( A31315.Read_volatile_chan_b_16b );
ChanB = fi_number.double*1000 % 1000G device

%read analog angle output for A31315LOLATR-**-S-AR-10 devices
%Check the device type to set the device output mode.

angle_ana = A31315.ReadOutputVoltage();
vcc_ana = A31315.GetVcc();

ana_range_sel = A31315.Read_eeprom_ana_range_sel();
low_clamp_percent = [0 4 5 6 7 8 10 15];
high_clamp_percent = [100 96 95 96 93 92 90 85];

angle_ana_0_360 = (angle_ana - vcc_ana * low_clamp_percent(ana_range_sel+1)/100) / ...
    ((high_clamp_percent(ana_range_sel+1) - ...
    low_clamp_percent(ana_range_sel+1))/100 * vcc_ana ) * 360;

%% Demo, read digital output

angle_out = [];

% Set high speed mode

A31315.HoldCommunicationEnable=true;

for i=1:50

    angle_out(i) = double(A31315.ReadPartialRegister(Allegro.ASEK.MemoryAccessType.primary, ...
        hex2dec('A7'), 15, 0));
    angle_out(i) = angle_out(i) * 360.0 / (2^16);
    plot(angle_out);
    drawnow

end

% End high speed mode

A31315.HoldCommunicationEnable=false;

%% Set analog output mode parameters
% applies only for A31315LOLATR-**-S-AR-10 devices

if true
    A31315.IsAnalogOutput = true;
    A31315.Write_eeprom_ana_range_sel(6) % Refer data sheet for more ranges
    A31315.Write_eeprom_ana_bw_sel(1)
    %15kHz, 1: 30kHz
    A31315.Write_eeprom_bw_sel(0)
end


%% Set SENT/PWM output parameters
% applies only for A31315LOLATR-**-S-SE-10 devices

if true
    A31315.Write_eeprom_dig_out_sel(2)
    A31315.Write_eeprom_dig_out_data_rate(0)
    % Set frame rate when using SENT (no effect on PWM)
    A31315.Write_eeprom_sent_frame_rate(2)

end
```

```
%% Write EEPROM dump to device

if true
    % Adjust the location of a potential CSV file (same as exported from GUI)
    % to upload it with the script to the device.
    % Load EEPROM to a CSV file, only use if the device should be reprogrammed
    % CSV file must have the same format as the csv export from the A31315 GUI

    EEP = readtable([pwd '\EEP_31315.csv']);

    % Set high speed mode
    A31315.HoldCommunicationEnable=true;

    %Parameters in Registry 0x15,0x3c,0x3D,0x3E, 0x3F
    %which are not included in the present DLL, hence not programmable

    N = {'chan_a_dis','chan_b_dis','chan_orthog', ...
        'spare_0x15','spare_0x21','spare_0x3b','spare_0x3c','spare_0x3d', ...
        'spare_0x3e','spare_0x3f'} ;

    % Program EEPROM to device

    for idx = 1:length(EEP.Var1)
        if ~(strcmp(EEP.Var1{idx},N))
            A31315.(['Write_eeprom_' EEP.Var1{idx}])(EEP.Var2(idx));
            disp([ 'Program: ' EEP.Var1{idx} ])
        else
            disp([ 'Not program: ' EEP.Var1{idx} ])
            continue
        end
    end

    % End high speed mode

    A31315.HoldCommunicationEnable=false;
end

%% Read write partial registers and re-program the parameters by changing their value

if true

    % For reading full register
    % example: A31315.ReadMemory(A31315.MemoryAccessType_Primary, hex2dec('37'))
    % For reading partial register

    Angle_gain = A31315.ReadPartialRegister(A31315.MemoryAccessType_Primary, hex2dec('33'), ...
        15,0);

    % Write ang_gain (default 1), To be checked again
    % read first old value

    ang_gain = fi(1,0,16, 10);
    ang_gain.dec = num2str( A31315.ReadPartialRegister(A31315.MemoryAccessType_Primary, ...
        hex2dec('33'), 15,0) );

    ang_gain.double=1;
    A31315.WritePartialRegister(A31315.MemoryAccessType_Primary, hex2dec('33'), ...
        str2num( ang_gain.dec ), 15,0)
    A31315.CloseCommunicationPort
end
```

Revision History

Number	Date	Description	Responsibility
–	April 26, 2022	Initial release	Vijay Peddoju

955 PERIMETER ROAD • MANCHESTER, NH 03103 • USA
+1-603-626-2300 • FAX: +1-603-641-5336 • ALLEGROMICRO.COM