This version (04 May 2020 08:18) was ***approved*** by AndreiD1994.

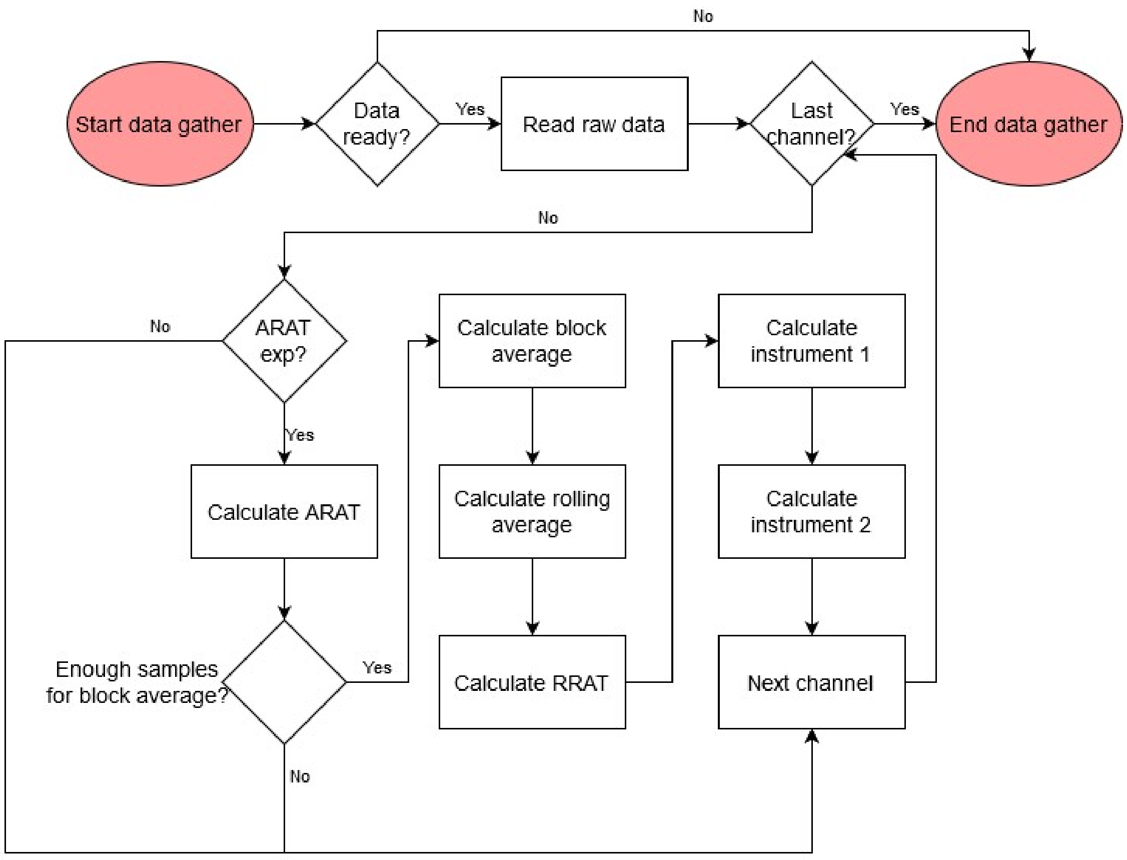
Optical Platform Demo (w/ EVAL-CN0503-ADRZ)

The **ADuCM3029_demo_cn0503** project provides a solution to get multiple liquid parameters (for example turbidity, pH, fluorescence, etc.) using the **EVAL-CN0503-ARDZ** and the **EVAL-ADICUP3029**. It uses a a complete multimodal sensor front end, stimulating up to eight LEDs and measuring the return signal on up to eight separate current inputs using photo-diodes. The board is controlled via a command line interface (CLI), Python scripts that provide high level functions or a Python GUI that communicates with the firmware via **serial terminal**.

General Description/Overview

The **ADuCM_demo_cn0503** project uses **EVAL-CN0503-ARDZ** to provide a method to determine properties of liquids, for example turbidity, fluorescence and pH. To do this the application must configure the ADPD4101 sensor, read the data from it and perform calculations on the data to improve **SNR** and pre-processes it to units as close to the desired measurement as possible.

The application configures the device to stimulate **LED**s and read the **PD**s in 4 time slots, each corresponding to one of the optical paths present on the board. It enables 2 ADC channels per time slot to measure the light beam before and after passing through the substance. This means that a packet of 8 samples of data come from the **ADC** once per sampling period.



The samples can then be plugged into a user defined equation that can include any of the samples and any random constant to generate a ratio, which the application defines as the **'absolute ratio'**. Before being displayed or used any further the **absolute ratio** is filtered by a **block average filter** and a **rolling average filter** to digitally increase the **SNR** and take out unwanted frequencies. The bandwidth of the **rolling average filter** can be set by the user. After getting the **absolute ratio** that describes the liquid that is now in the vat, it can be used to compare it to the **baseline measurement** using either 1 - (absolute ratio / baseline) or (absolute ratio / baseline) equations. The user has to input the **baseline** for each optical path as well as the equation from these options. The application defines this result as the **'relative ratio'**. The application then has 2 more options of processing the information in the form of two **5th order polynomials** in which the **relative ratio** can be inserted as the variable. The coefficients of each of the **polynomial** members are also input by the user via the **CLI**. To be noted that the result of the first equation is the variable for the second equation. This gives the user a high degree of flexibility in getting data out of the platform.

Demo Requirements

The following is a list of items needed in order to replicate this demo.

- Hardware
  - EVAL-ADICUP3029
  - EVAL-CN0503-ARDZ
  - Mirco USB to USB cable
  - PC or Laptop with a USB port
  - 3D printed mechanical fixtures described in the Hardware User Guide
- Software
  - AduCM3029_demo_cn0503 demo application
  - CrossCore Embedded Studio (2.8.0 or higher)
  - ADuCM302x DFP (3.2.0 or higher)
  - ADICUP3029 BSP (1.1.0 or higher)
  - Serial Terminal Program
    - Such as Putty or Tera Term

Setting up the Hardware

1. Set up the the **EVAL-CN0503-ARDZ** as shown in the Hardware User Guide.
2. Connect the board to the **EVAL-ADICUP3029** via the Arduino headers. Fix Me! add picture
3. Connect a micro- USB cable to P10 connector of the EVAL-ADICUP3029 and connect it to a computer. The final setup should look similar to the picture below. Fix Me! add picture

Configuring the Software

The **ADuCM3029_demo_cn0503** does not need any software configuration. It can be built and run as is.

Outputting Data

A serial terminal is an application that runs on a PC or laptop that is used to display data and interact with a connected device (including many of the Circuits from the Lab reference designs). The device's UART peripheral is most often connected to a UART to USB interface IC, which appears as a traditional COM port on the host PC/ laptop. (Traditionally, the device's UART port would have been connected to an RS-232 line driver / receiver and connected to the PC via a 9-pin or 25-pin serial port.) There are many open-source applications, and while there are many choices, typically we use one of the following:

- Tera Term
- Putty
- Real Term

Before continuing, please make sure you download and install one of the above programs.

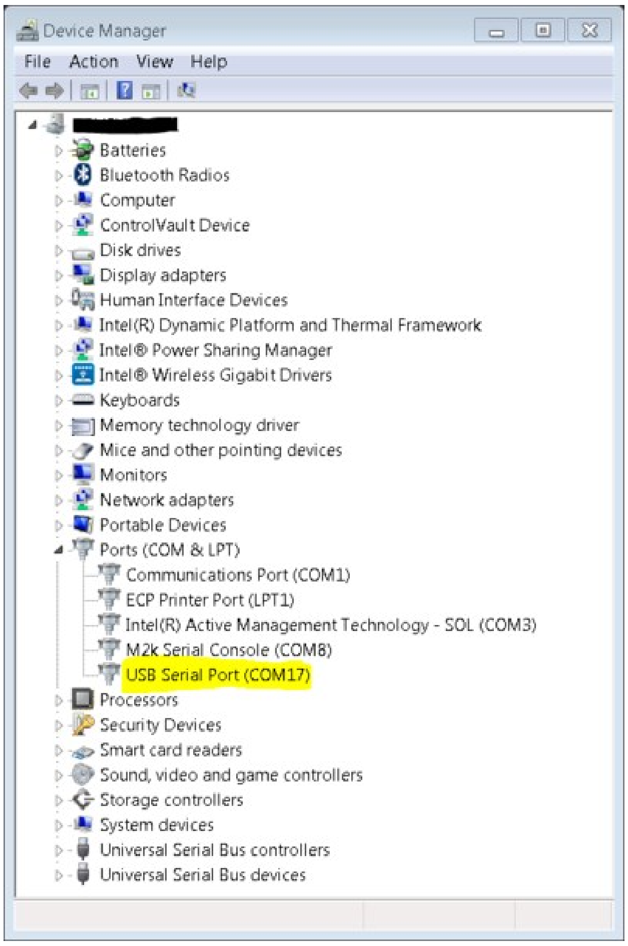
There are several parameters on all serial terminal programs that must be setup properly in order for the PC and the connected device to communicate. Below are the common settings that must match on both the PC side and the connected UART device.

1. **COM Port** - This is the physical connection made to your PC or Laptop, typically made through a USB cable but can be any serial communications cable. You can determine the COM port assigned to your device by visiting the device manager on your computer. Another method for identifying which COM port is associated with a USB-based device is to look at which COM ports are present before plugging in your device, then plug in your device, and look for a new COM port.
2. **Baud Rate** - This is the speed at which data is being transferred from the connected device to your PC. These parameters must be the same on both devices or data will be corrupted. The default setting for most of the reference designs in 115200.
3. **Data Bits** - The number of data bits per transfer. Typically UART transmits ASCII codes back to the serial port so by default this is almost always set to 8-Bits.
4. **Stop Bits** - The number of “stop” conditions per transmission. This usually set to 1, but can be set to 2 for redundancy.
5. **Parity** - Is a way to check for errors during the UART transmission. Unless otherwise specified, set parity to “none”.
6. **Flow Control** - Is a way to ensure that data lose between fast and slow devices on the same UART bus are not lost during transmission. This is typically not implemented in a simple system, and unless otherwise specified, set to “none”.

In many instances there are other options that each of the different serial terminal applications provide, such as **local line echo** or **local line editing**, and features like this can be turned on or off depending on your preferences. This setup guide will not go over all the options of each tool, but just the minor features that will make it easier to read back data from the connected devices.

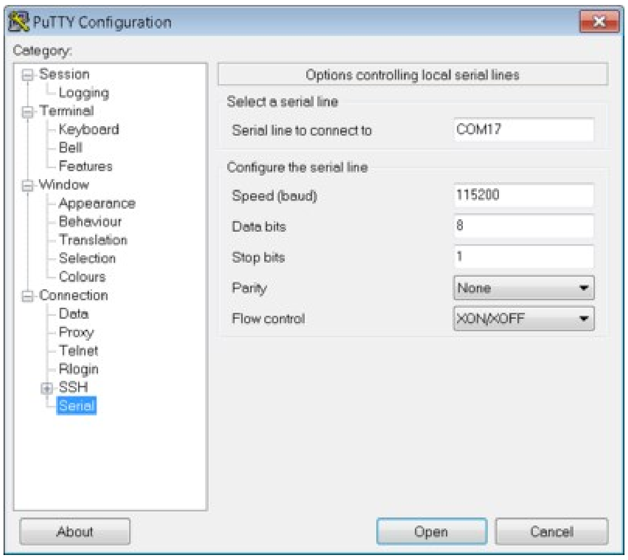
Example setup using Putty

1. Plug in your connected device using a USB cable or other serial cable.
2. Wait for the device driver of the connected device to install on your PC or Laptop.
3. Open your device manager, and find out which COM port was assigned to your device.



[redacted]
- Batteries
- Bluetooth Radios
- Computer
- ControlVault Device
- Disk drives
- Display adapters
- Human Interface Devices
- Intel(R) Dynamic Platform and Thermal Framework
- Intel® Power Sharing Manager
- Intel® Wireless Gigabit Drivers
- Keyboards
- Memory technology driver
- Mice and other pointing devices
- Monitors
- Network adapters
- Portable Devices
- Ports (COM & LPT)
  - Communications Port (COM1)
  - ECP Printer Port (LPT1)
  - Intel(R) Active Management Technology - SOL (COM3)
  - M2k Serial Console (COM8)
  - USB Serial Port (COM17)
- Processors
- Security Devices
- Smart card readers
- Sound, video and game controllers
- Storage controllers
- System devices
- Universal Serial Bus controllers
- Universal Serial Bus devices

4. Open up your serial terminal program (Putty for this example)
5. Click on the serial configuration tab or window, and input the settings to match the requirements of your connected device. The default baud rate for most of the reference designs is 115200. Make sure that is the selected baud rate as well.



Options controlling local serial lines

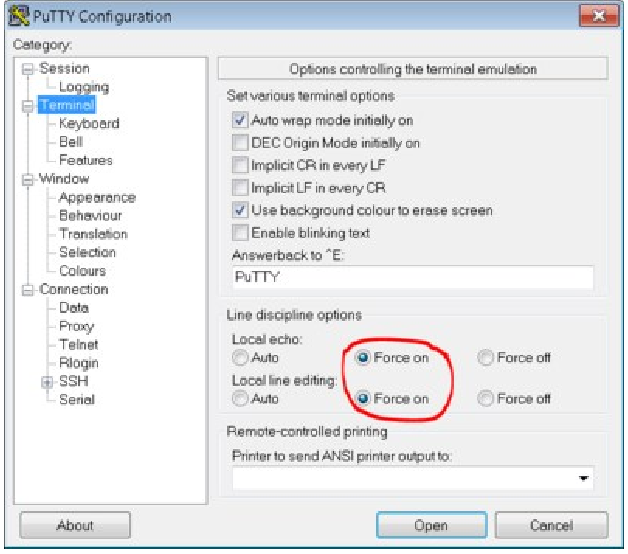
Select a serial line

Serial line to connect to	COM17

Configure the serial line

Speed (baud)	115200
Data bits	8
Stop bits	1
Parity	None
Flow control	XON/XOFF

6. Ensure that local echo and line editing are enabled, so that you can see what you type and are able to correct mistakes. (Some devices may echo typed characters - if so, you will see each typed character twice. If this happens, turn off local echo.)


PuTTY Configuration

Category:
- Session
  - Logging
- Terminal
  - Keyboard
  - Bell
  - Features
- Window
  - Appearance
  - Behaviour
  - Translation
  - Selection
  - Colours
- Connection
  - Data
  - Proxy
  - Telnet
  - Rlogin
  - SSH
  - Serial

Options controlling the terminal emulation

Set various terminal options
- [✓] Auto wrap mode initially on
- [ ] DEC Origin Mode initially on
- [ ] Implicit CR in every LF
- [ ] Implicit LF in every CR
- [✓] Use background colour to erase screen
- [ ] Enable blinking text

Answerback to ^E: PuTTY

Line discipline options
- Local echo: Auto [✓ Force on] Force off
- Local line editing: Auto [✓ Force on] Force off

Remote-controlled printing

Printer to send ANSI printer output to:

About | Open | Cancel

7. Click on the open button, and as long as your connected device and serial terminal program are setup the same, than you should see data displaying.

Hint: If you see nothing in the serial terminal, try hitting the reset button on the embedded development board.

Available commands

Typing **help** after the application has started will display the list of commands:

Function	**Command**	**Description**	**Example**
Application commands			
	HELP	Display available commands.	
	REG? XXX	Read an ADPD register. *<XXX>* = register address in hexadecimal	REGA - read register 0x10A
	REG XXX YYYY	Write an ADPD register. *<XXX>* = register address in hexadecimal *<YYYY>* = register new value in hexadecimal	REG 10A 301 - write 0x301 to register 0x10A
	MODE?	Read the data display mode. Return a 4 character code: *CODE* - raw data is displayed; *ARAT* - absolute ratio is displayed; *RRAT* - relative ratio is displayed; *INS1* - data processed for instrument 1 is displayed; *INS2* - data processed for instrument 2 is displayed.	
	MODE XXXX	Set the data display mode. *XXXX* = 4 character code to describe the data display mode. The options are: *CODE* - raw data is displayed; *ARAT* - absolute ratio is displayed; *RRAT* - relative ratio is displayed; *INS1* - data processed for instrument 1 is displayed; *INS2* - data processed for instrument 2 is displayed.	MODE CODE - set display mode to raw codes
	STREAM X	Start calculating and displaying a stream of data. *X* = If 0 or not present enter continuous streaming mode; if X>0 stream data for X values.	STREAM 10 - stream for 10 values of the output.
	IDLE?	Queries the idle condition.	
	IDLE x	If X=0 or not listed only the terminal stream is terminated, but the application continues sampling. If X=1 stop sampling altogether.	IDLE 0 - stop streaming, but keep sampling.
	ALRM?	Query alarm status; if returned 0 no alarm is present; If bit 0 of the returned value is 1 instrument 2 value is below lower threshold; If bit 1 of the returned value is 1 instrument 2 value is above higher threshold.	
	DEFn? XXXX	Query operation parameters. *n* = ID of the required optical path (can be from 0 to 8). *XXXX* = the operation parameter to be queried. The following options exist: *ARAT* = absolute ratio expression in reverse polish notation (RPN); *RFLT* = the digital low pass filter bandwidth applied to the absolute ratio measurements; *ALRM* = the high and low alarm threshold for the INS2	DEF ARAT - query the abslute ratio expression for optical path 0

Function Application commands	Command	Description	Example
		measurements; *RATB* = baseline ratio used in calculating the relative ratio; *INS1* = the coefficients of the fifth order polynomial used to calculate the first instrument measurements; *INS2* = the coefficients of the fifth order polynomial used to calculate the second instrument measurements.	
	BOOT	Perform a software reset.	
	ODR?	Query the Output Data Rate of the STREAM command.	
	ODR XX.XX	Sets Output Data Rate for the STREAM command. *XX.XX* = The new ODR in Hz. It can have only certain values and will round down to them: ODR options between 5 Hz and 0.01 Hz and can be calculated by 5/ODR = integer number.	ODR 2.5 - set output data rate to 2.5 SPS
	RATMASK?	Query the active channels mask. Returns a hexadecimal 8 bit number that has 1 for every active channel and 0 for every hidden channel. For example if 1 - only optical path 0 is turned on; if 3 - Optical paths 0 and 1 are turned on and the rest are hidden; etc.	
	RATMASK XX	Set the active channels mask. *XX* = New optical path mask.	RATMASK F - turn on optical paths 0, 1, 2 and 3 and leave the rest(4, 5, 6 and 7) off
	FL_HELP	Display the tooltip for the flash commands.	
	PCB-LEDn YY.Y XXX.X	Do an automatic calibration of the specified PCB LED current so that the corresponding ADC channels returns a percentage of the saturation. *n* = LED ID from the PCB.(1, 2, 3 or 4) *YY.Y* = percentage of ADC saturation. *XXX.X* = maximum LED current. If left not specified will default at 338 mA.	PCB-LED1 45.8 190.5 - calibrate LED 1 so that the cresponding optical path(s) return 45.8% from the ADC saturation, but limit the current to 190.5 mA.
	DEFn args	**Set operation parameters.** ***n* = ID of the required optical path (can be from 0 to 8).** ***args* = operation parameters and their new value.**	
	DEFn ARAT RPN	Set the absolute ratio expression; *RPN* = new ARAT expression.	DEF0 ARAT A2A1/ - set the optical path 0 absolute ratio to be calculated as (slotA_ch2 / slotA_ch1);
	DEFn RFLT XX.XX	Set the digital filter bandwidth, in Hz, used for the absolute ratio measurement; *XX.XX* = new digital filter bandwidth in Hz.	DEF1 RFLT 0.5 - set the optical path 1 digital filter bandwidth to 0.5 Hz. Not all values are possible, query the parameter to get the actual value set;
	DEFn ALRM XXXX YYYY	Set high and low alarm thresholds for instrument 2 measurements; *XXXX* = new alarm high threshold; *YYYY* = new alarm low threshold;	DEF0 ALRM 25 15 - set optical path 0 high alarm threshold to 25 and low alarm threshold to 15;
	DEFn RATB X.XXXXX	Set baseline ratio for to calculate relative ratio of the signal path; *X.XXXXX* = new baseline ratio.	DEF2 RATB 2.3 - set the baseline ratio as 2.3 for signal path 2;
	DEFn INS1 X.XXE+XX Y.YYE+YY …	Set the fifth order polynomial coefficients for calculating instrument 1, ordered from lowest to highest power;	DEF0 INS1 0 1 0.54 1.23e+0 5.48e-4 44.22 - set coefficients for optical path 0;
	DEFn INS2 X.XXE+XX Y.YYE+YY …	Set the fifth order polynomial coefficients for calculating instrument 2, ordered from lowest to highest power;	DEF4 INS2 0 1 0.54 1.23e+0 5.48e-4 44.22 - set coefficients for optical path 4;
Flash commands			
	FL_CLEARBUF	Clear the software buffer.	
	FL_LOAD X	Load the software buffer with data from flash configuration page. *x* = 0 to load from User Update page; 1 to load from the Manufacture Default page.	
	FL_PROGRAM X	Program the flash configuration page with data from the software buffer. *x* = 0 to program the User Update page; key to program the Manufacture Default page.	
	FL_ERASE X	Erase the flash configuration page. *x* = 0 to erase the User Update page; key to erase the Manufacture Default page.	
	FL_APPLY	Settings currently in the software buffer are applied to the application and device.	
		Save a specific device or application parameter into the software buffer. *cmd* = register or application command to input the relevant	

Function Application commands	Command	Description	Example
	FL_WRITE CMD	parameter. For register commands the buffer contains an array of register-value pairs that will be written in order to the device. Calling this function with a register command will add a new register-value pair at the end of the array. The only application command supported by this command is the def command. Calling the DEF command will update that value into the software buffer. It can be later applied to the application.	FL_WRITE REG 10A 3355 - write the register 0x10A to the value 0x3355. FL_WRITE DEF2 RFLT 0.01 - set filter bandwidth of ration 2 to 0.01Hz
	FL_READ CMD	Read a specific device or application parameter value from the software buffer. *cmd* = register or application command to input the relevant parameter. For register commands the buffer contains an array of register-value pairs that will be written in order to the device. Calling this function with a regiter command will display the value of the first occurrence of the register. The only application command supported by this command is the def command. Calling the DEF command will read that value from the software buffer.	FL_READ REGA - return the first value of the register 0x10A FL_READ DEF RFLT - display the setting of the ration 2 filter bandwidth from the software buffer.

Fix Me! - ADD CLI PICTURE

Obtaining the Source Code

We recommend not opening the project directly, but rather import it into CrossCore Embedded Studios and make a local copy in your workspace.

The source code and include files of the **ADuCM3029_demo_cn0503** can be found here:

AduCM3029_demo_cn0503 at Github

AduCM3029_demo_cn0503.HEX

How to use the Tools

The official tool we promote for use with the EVAL-ADICUP3029 is CrossCore Embedded Studio. For more information on downloading the tools and a quick start guide on how to use the tool basics, please check out the Tools Overview page.

Importing

For more detailed instructions on importing this application/demo example into the CrossCore Embedded Studios tools, please view our How to import existing projects into your workspace section.

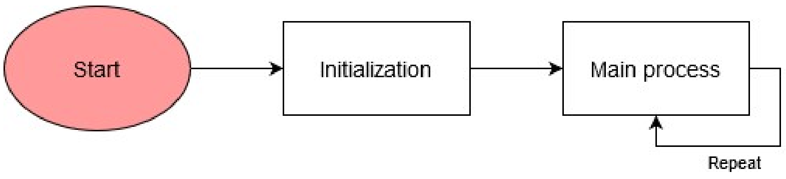
Debugging

For more detailed instructions on importing this application/demo example into the CrossCore Embedded Studios tools, please view our How to configure the debug session section.

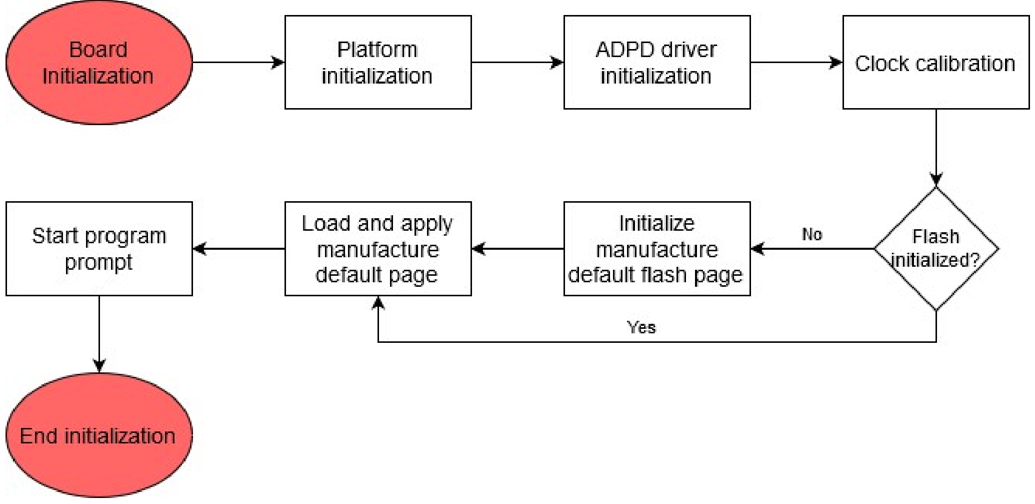
Project Structure

The program is composed of two main parts:

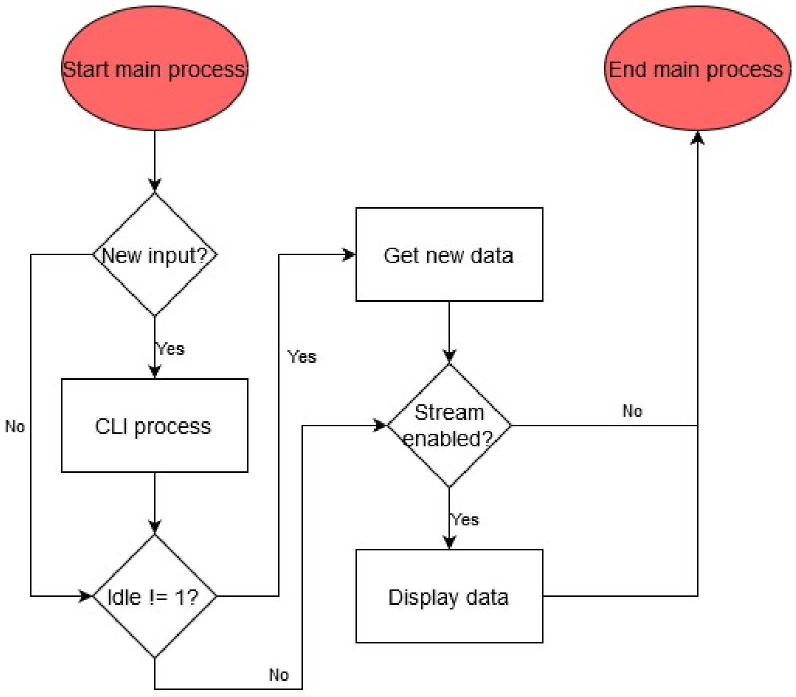
1. Initialization routine.
2. Main process.



The platform initialization includes the carrier clock and power initialization as well as initialization for **DMA**, **GPIO**, **SPI**, **I2C**, **UART** and **flash** cores. Then the program initializes the ADPD device driver with default values and applies this configuration to the chip. After this it will perform the clock calibration for the ADPD device and will apply the configuration saved in the flash pages in the following order: manufacturer page and user page. If the manufacturer page is not initialized it will initialize it. Applying the user page after the manufacturer default will only change the specific configurations mentioned in the user page, leaving the rest untouched.



The main process of the application runs into a loop and is responsible for taking data out of the device and displaying it and for implementing the user **CLI** using the serial **UART** interface.



End of Document