Soliton Technologies Pvt. Ltd.

System / Software Design Document

Revision History

Ver#	Date	Created By	Reviewed By	Approved By	Revision Details
1.0	17-Mar-2023	Sadhana S.			Initial Version
2.0	04-Apr-2023	Sadhana S.			Final Version
3.0					

1. Objective

1.1 To create a multi-channel-ADC.

1.2 Create a driver vi that generates ANALOG values and converts it to U8 1D digital array.

1.3 Create a GUI where the digital data is plotted and stored, also channel selection is to be made.

S.No.	Req ID	Key pain points	Priority	How is it addressed?
1	REQ001	Returns 1D array of U8 data.	High	The random values generated is converted to digital and into unsigned byte array 1D.
2	REQ002	Driver receives <=16 bits, <=8 channels and sampling rate as input.	High	The ADC chosen in GUI receives the specifications from the ADC types text file and those values are passed to this ADC driver as inputs.
3	REQ003	Changes to ADC to make a new ADC has to be made in the file and not in the LabVIEW source code.	Medium	By clicking Edit Configuration the user can specify the specs to be changed and by clicking Confirm the changes gets applied in the *.txt file as well as the current ADC configuration display.
4	REQ004	GUI is designed to call the driver for every 250ms.	Medium	In the Timeout case it checks if the Start button is True, in that case it goes to that state where it calls the ADC driver.
5	REQ005	Allow the user to select a particular ADC.	Medium	User can choose the ADC from the ADC Type Enum.
6	REQ006	Start/Stop data capture.	High	When START is pressed the ADC driver is called and data acquisition occurs.
7	REQ007	User selects which channel to view.	Medium	When the user unselects a channel then that channel will not be visible and vice-versa.
8	REQ008	Select a data storage path.	Low	When a valid path is chosen and when OK is pressed then data generated gets loaded in the .*txt file.

2. Design Validation

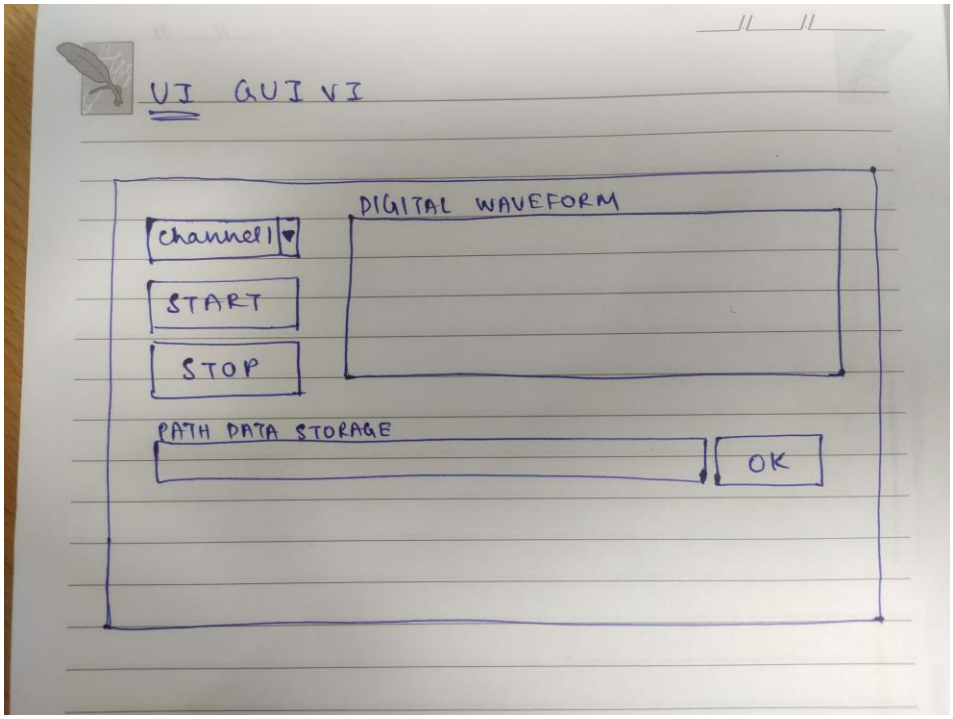
This section deals with the basic design decisions that were made with the basic understanding of the provided problem statement.

2.1. Design Decisions

This section will show how the designs were created from scratch considering the requirements, the process followed in creating them.

2.1.1. Wireframing

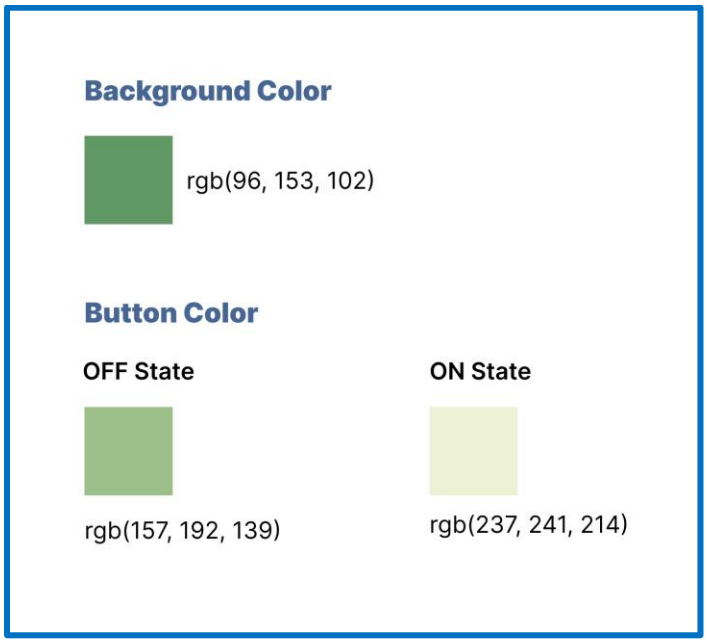
Initially listed out what are the controls and indicators required in the GUI. As the GUI is going to have the UI, created wireframe for GUI to see the alignment and placement of controls and indicators.



This is the wireframe created for GUI. Here I have added the channel selection Enum, START and STOP Boolean buttons, output digital waveform display, path selection for the data storage file.

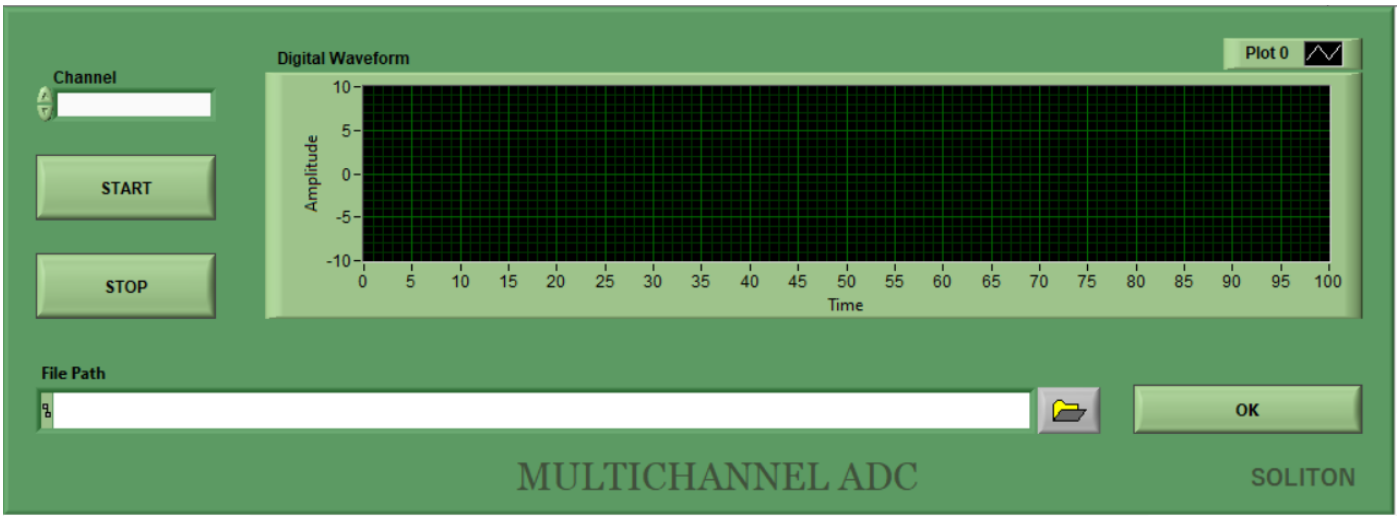
2.1.2. Color Palette

After deciding the controls and its alignments it was now to decide what are the colors to be used in the GUI. So, I created a palette with the colors that are going to be used in the GUI's User Interface.

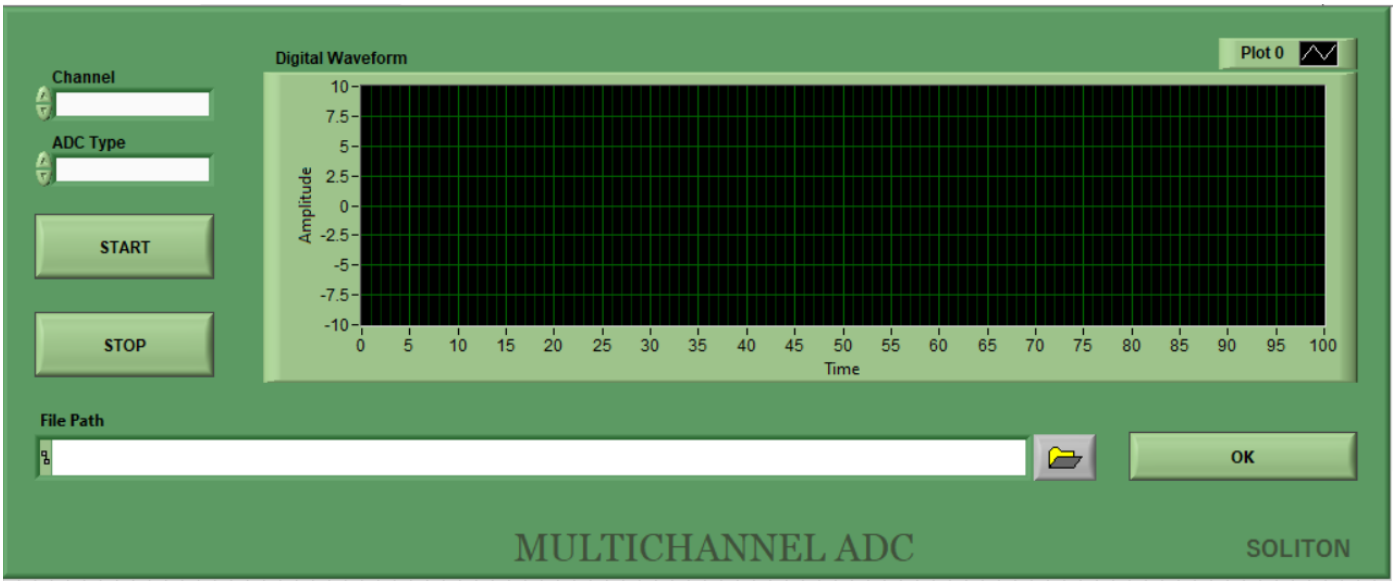


These are the colors that were chosen for the GUI VI. The RGB values are also given. This color was chosen to give a feel of real time machines such as CRO etc... to resemble such looks.

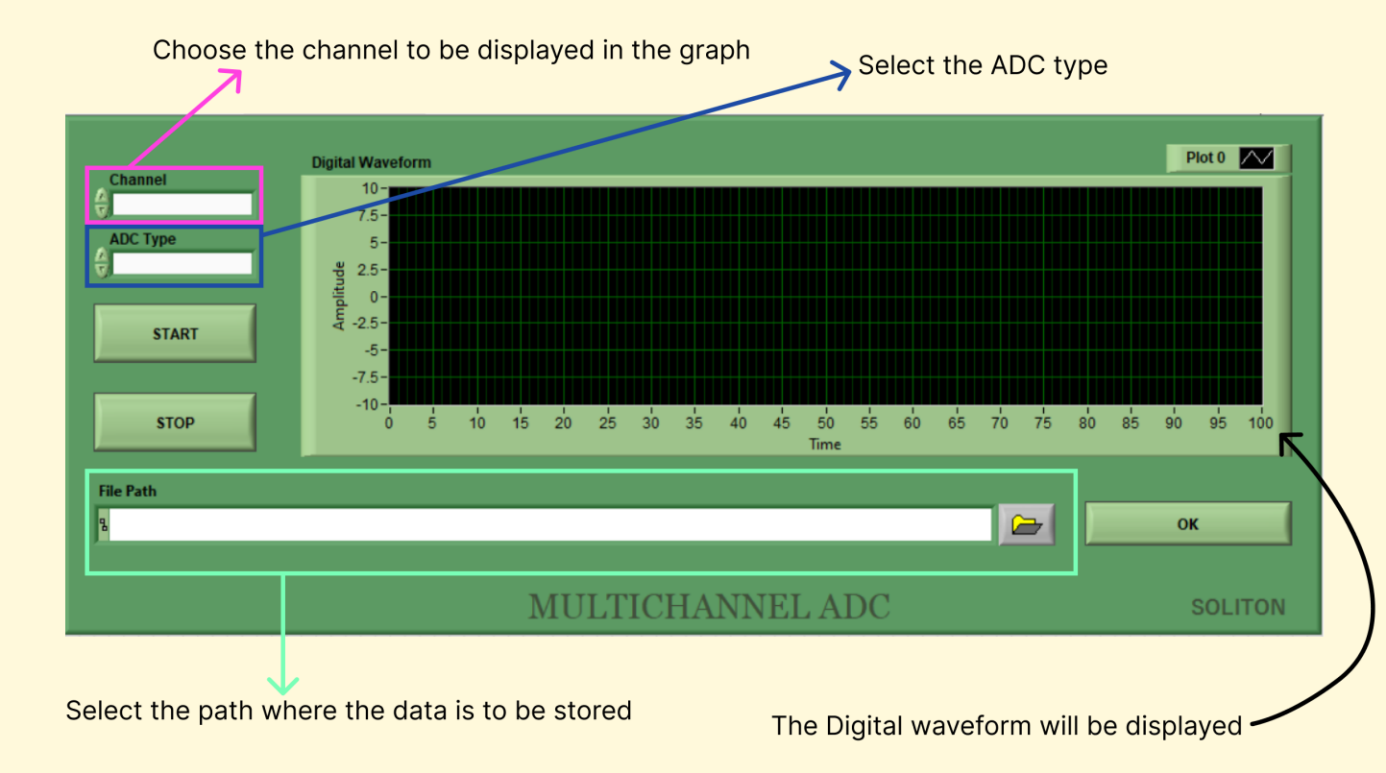
2.1.3. Mockup



This was the initial mockup created for the GUI in LabVIEW. Here I replicated the wireframe and added the colors from the color palette.



UPDATE1: Here I added ADC Type Enum to choose the respective ADC. The ADC specifications will be written in a file and those names are added to this Enum ADC.



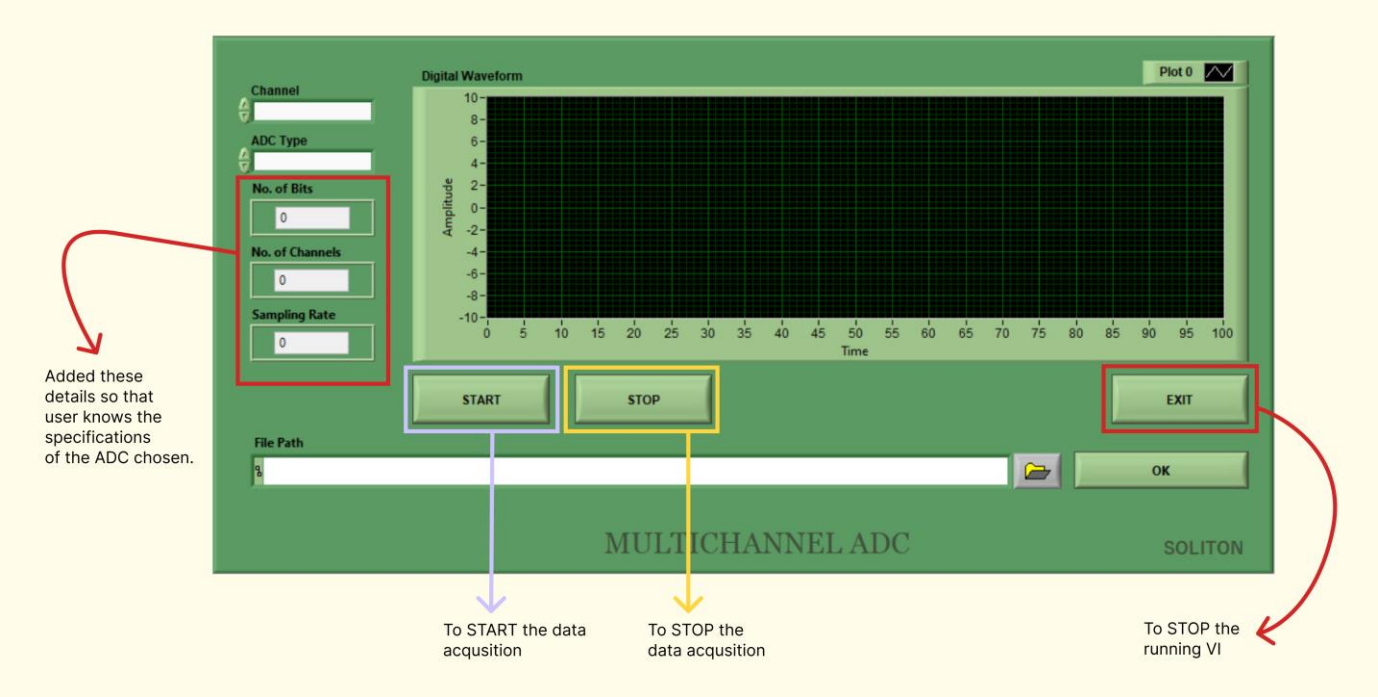
These were the controls that were added to the GUI VI. The channel to be displayed can be chosen, default will be ALL where all the channels value will be displayed. If a particular channel is needed, then that can be chosen from the Enum.

The ADC types are present in the Enum. The user can select the particular ADC needed.

START and STOP will start and stop the data acquisition respectively.

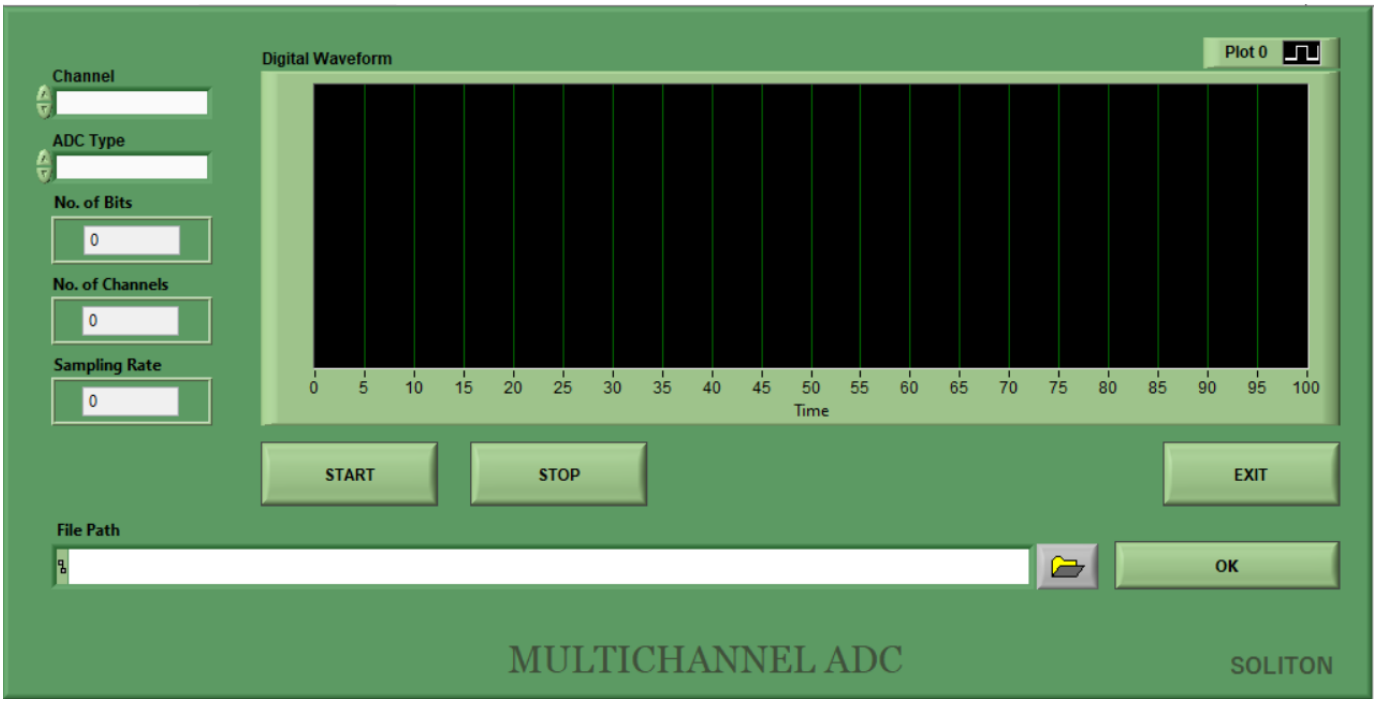
The file in which the data is to be stored can be selected in the file path.

The analog data converted to digital in the ADC driver will be sent to the GUI and will be plotted in the graph.



When an ADC is chosen the corresponding bits, channels and sampling rate will be displayed in the indicator. This was added so that the users can get the idea of the ADC that was chosen.

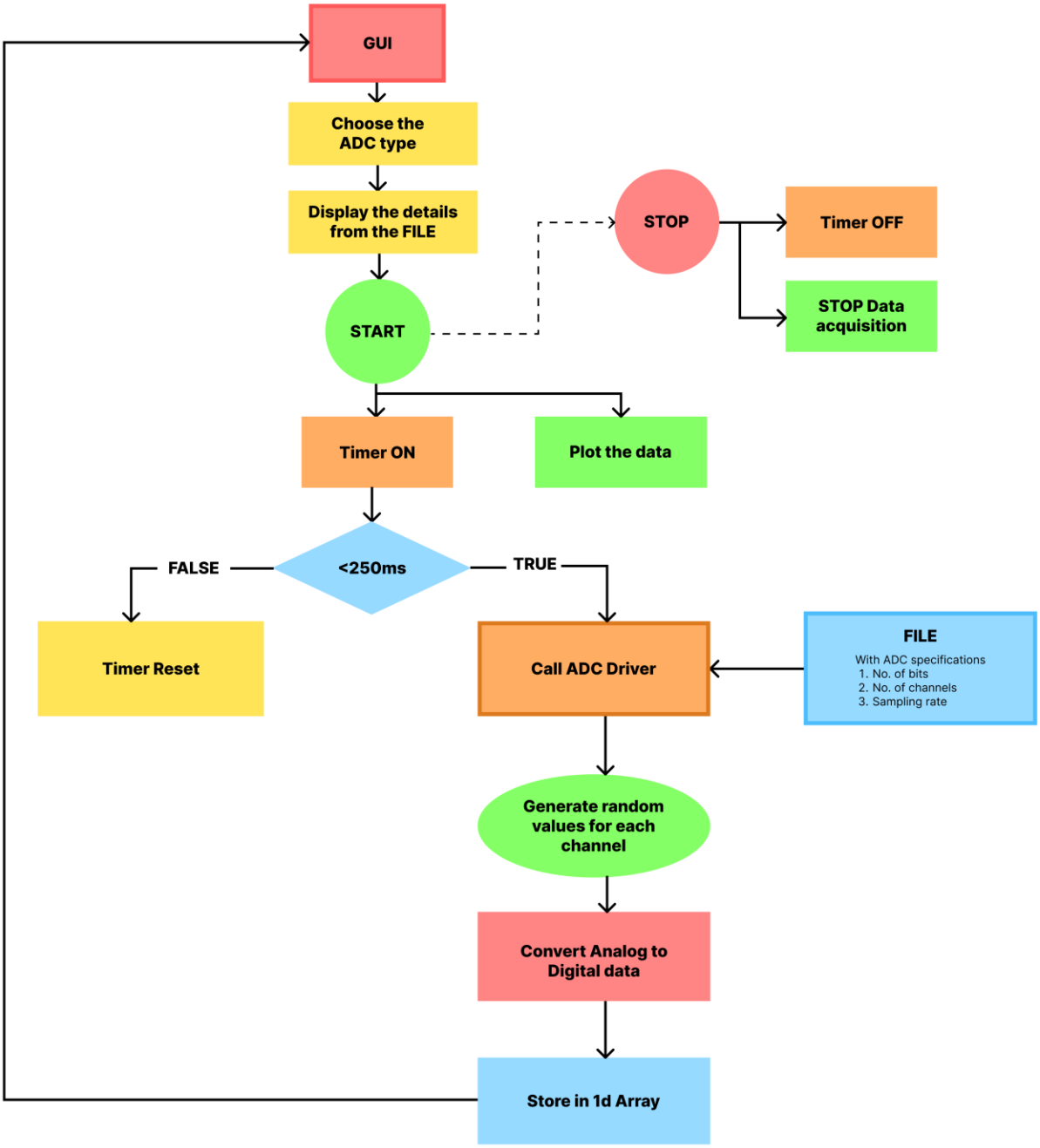
EXIT button was added so that PANEL CLOSE or this button will STOP the VI from running.



This is the final look of the GUI's mockup. The graph was replaced with a digital waveform graph to show that digital data will be displayed.

3. Software Architecture

3.1. Architecture Overview / Diagram



The above flowchart is the rough idea of how the process will take place. Here I have jotted down a rough sketch of the process and tried to break the problem and created the flow with some basic understanding of the problem statement given.

This flowchart will show an overview of approaching the problem.

3.2. ADC Driver Architecture

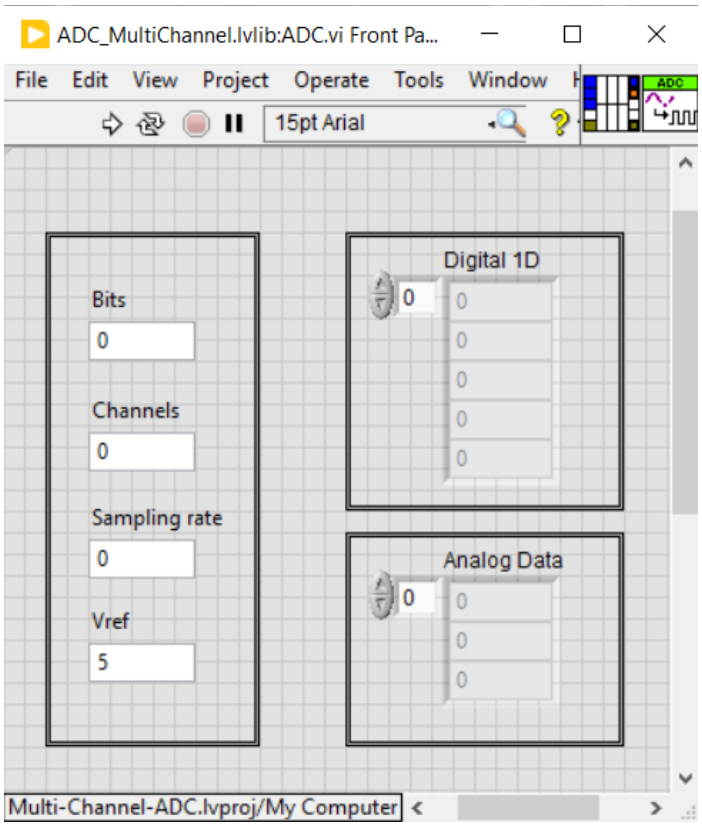
- Here I created a driver vi that generates random values in the range of 0 to Voltage reference for each channel.
- The inputs such as number of channels, number of bits and sampling rate are passed from the GUI which reads the specifications from the file.
- The generated random values are analog values.
- The analog values are converted to digital values by using the formula: **Digital Output = [Analog input voltage * (2^n – 1)]/Reference voltage**
- This value is converted to binary form and then converted to Unsigned byte array.
- By subtracting the ASCII value of zero from the numbers ASCII we can get 1s and 0s.
- If this digital data length is not same as the number of bits, then 0 is added to it.
- This generated 2D array is converted into 1D array so that 1D U8 data can be passed from the driver.

3.3. GUI Architecture

- In the GUI the user can choose the ADC type. On choosing the type the specifications of that ADC will be displayed.
- When START is pressed it goes to that particular event where it calls the ADC driver and data acquisition starts [REQ006].
- After 250ms it comes to the TIMEOUT case where it checks if the START button is still TRUE, in that case it again goes to the state for data acquisition else, it STOPS the data acquisition [REQ004].
- This data is plotted in the graph.
- When a particular channel is unselected the that will not be displayed in the graph. Likewise, channel selection can be made.
- When a text file is chosen and pressed OK the data will be stored in the text file [REQ008].
- If the specification of the current ADC is to be edited then EDIT CONFIGURATION button can be used.
- It will help to alter the number of bits, channels and sampling rate.
- PANEL CLOSE will stop the running vi.

4. Modules Design

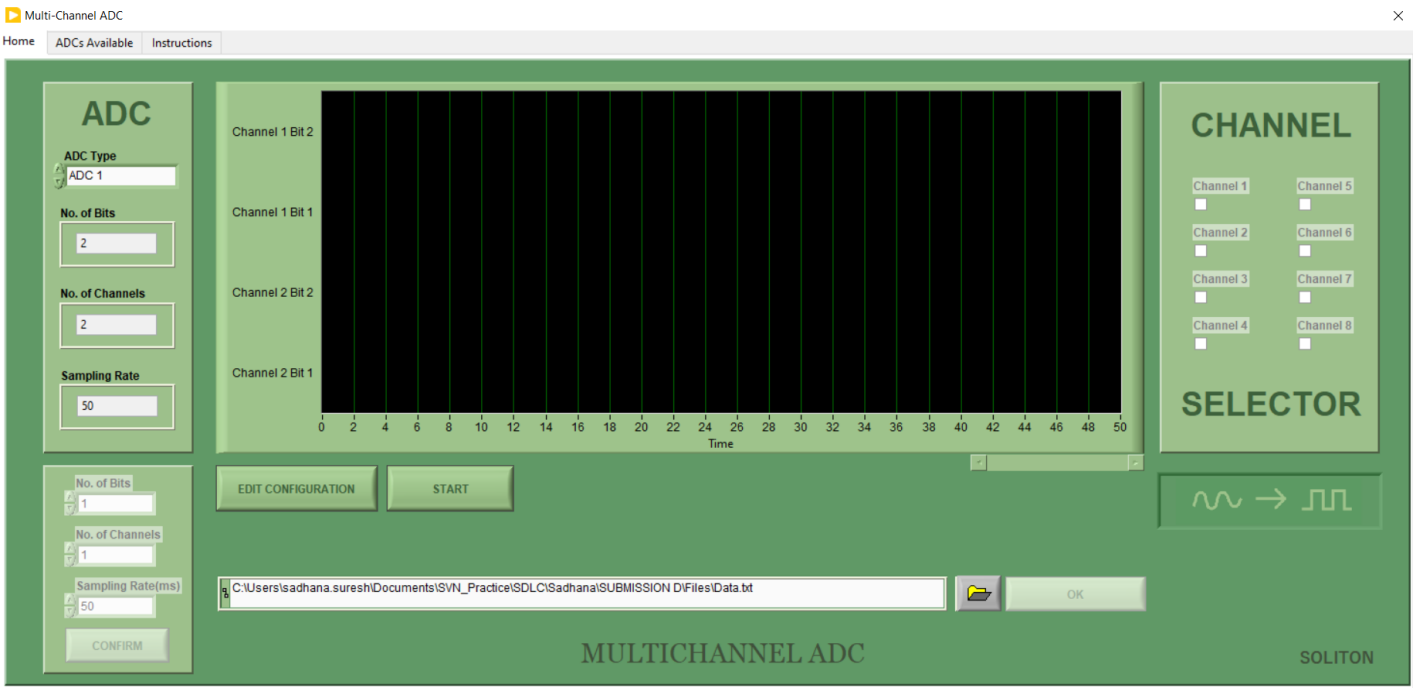
4.1. ADC DRIVER



Interfacing parameters with the module	Input/Output	Datatype
Bits	Input	Numeric
Channels	Input	Numeric
Sampling rate	Input	Numeric
Vref	Input	Numeric
Digital 1D	Output	Numeric Array
Analog Data	Output	Numeric Array

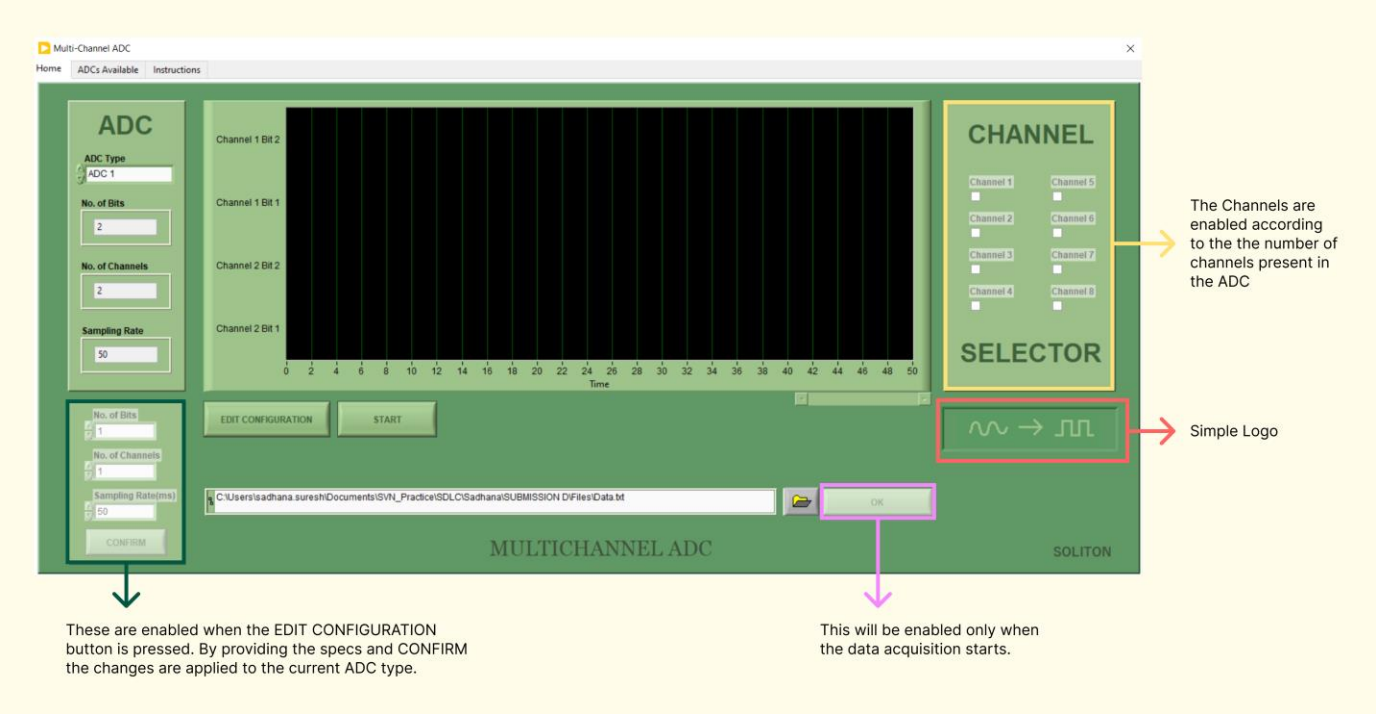
The ADC Driver receives the input data such as number of bits, channels and sampling rate from the GUI according to the ADC chosen [REQ002]. The Vref is made as a constant as the study shows that the Vref is between –5V to 5V. The generated Analog data is displayed as a numeric array indicator along with the digital data that was converted from the analog values. The digital data returned is U8 1D array [REQ001].

4.2. GUI

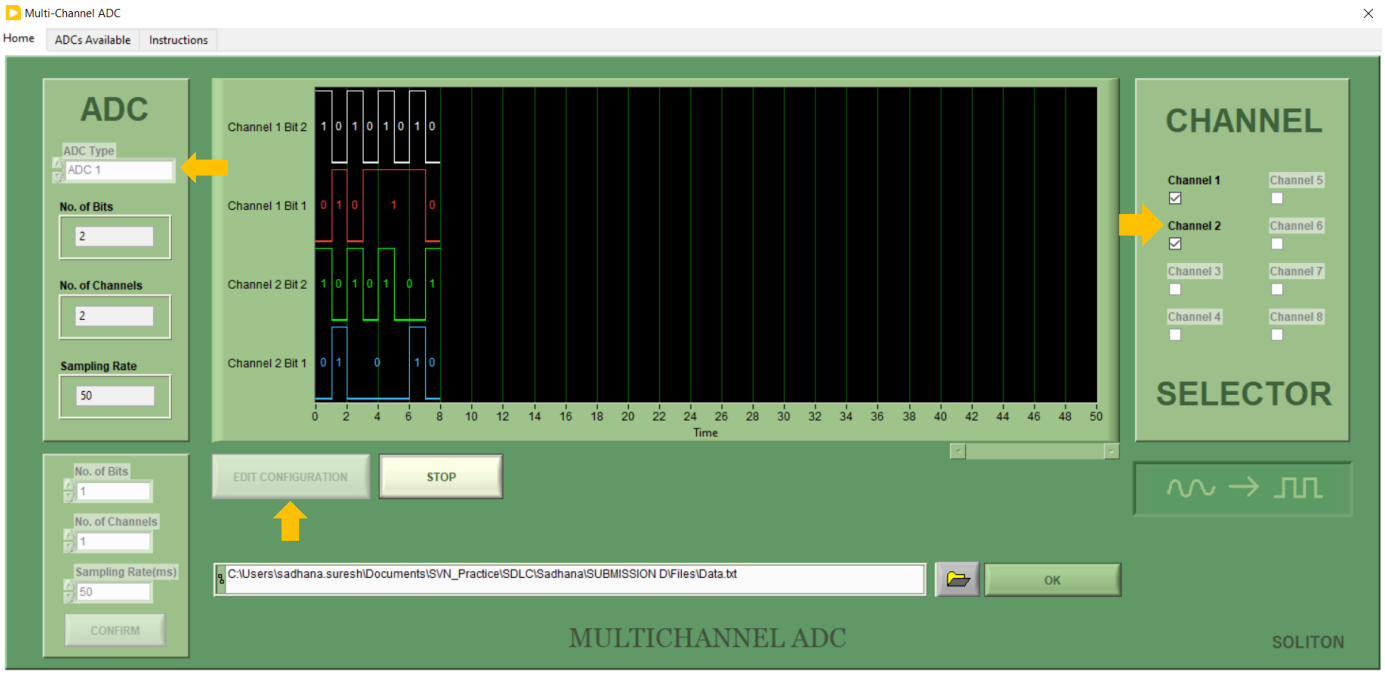


Interfacing parameters with the module	Input/Output	Datatype
ADC Type	Input	Enum
No. of Bits	Output	Numeric
No. Of Channels	Output	Numeric
Sampling Rate	Output	Numeric
Edit Configuration	Input	Boolean
Start	Input	Boolean
File Path	Input	Path
OK	Input	Boolean
Channel 1,2,3,4,5,6,7,8	Input	Boolean
Digital Waveform Graph	Output	Graph
No. of Bits	Input	Numeric
No. of Channels	Input	Numeric

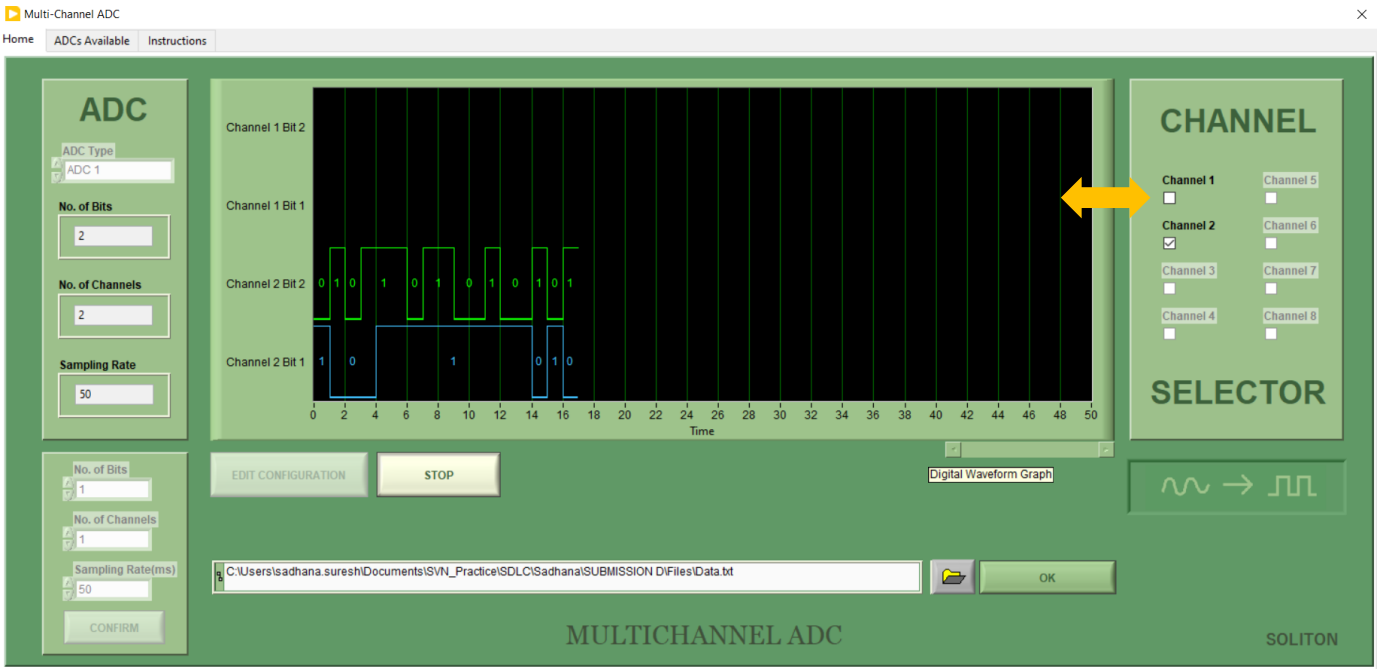
Sampling Rate(ms)	Input	Numeric
Confirm	Input	Boolean



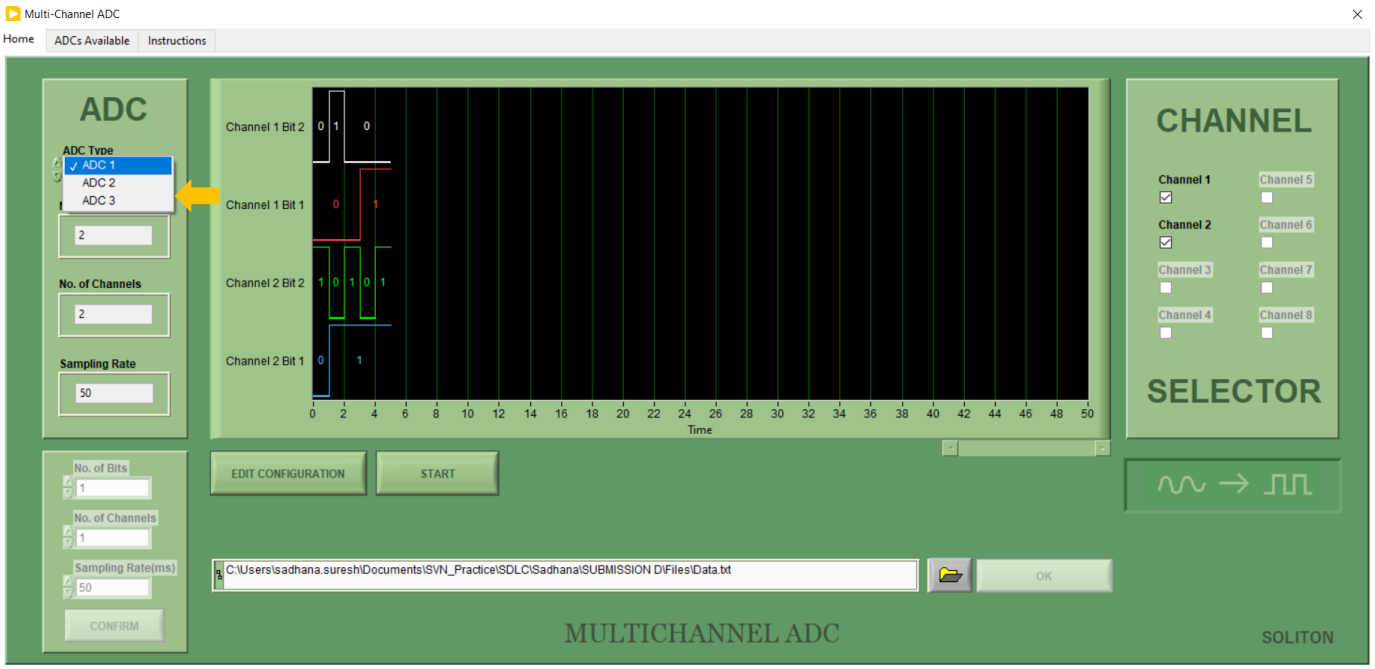
In the initial design the Channels were chose to be an Enum in that case user will be able to see only one channel at a time. If the user wants to see Channel1 and Channel3 but not Channel2 this is not possible. So Channels were made into Boolean so selected channels get displayed.



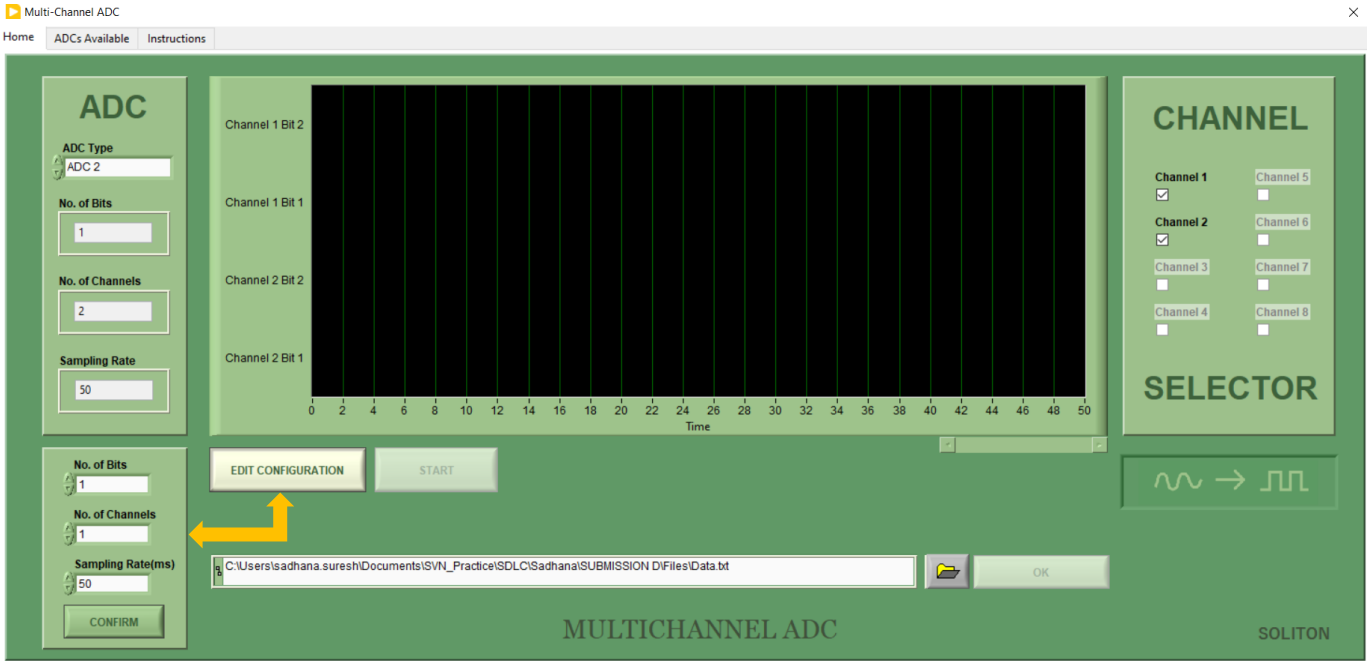
When START is pressed the total number of channels present alone is enabled. Initially the available channels are checked and all the channels gets displayed. Once data acquisition starts the ADC type cannot be changed or its configuration cannot be edited.



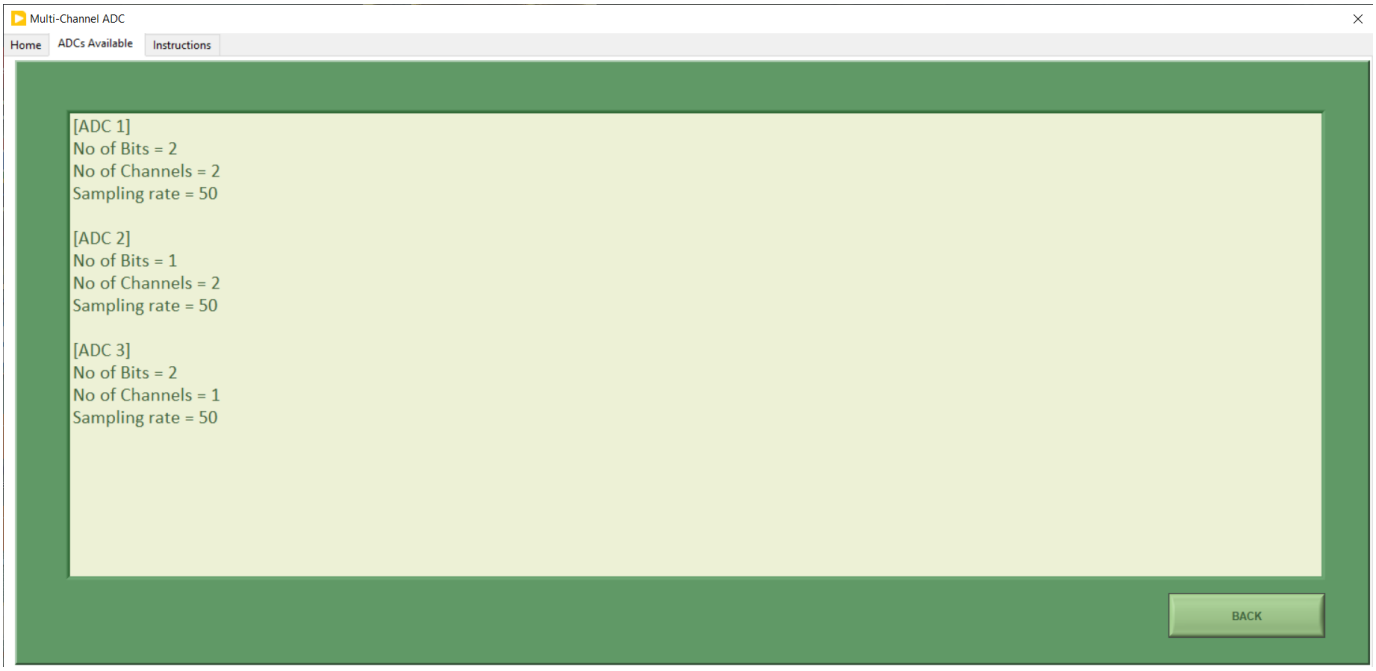
In the above image when the Channel 2 is alone selected only that channel and its bits gets displayed in the graph [REQ007].



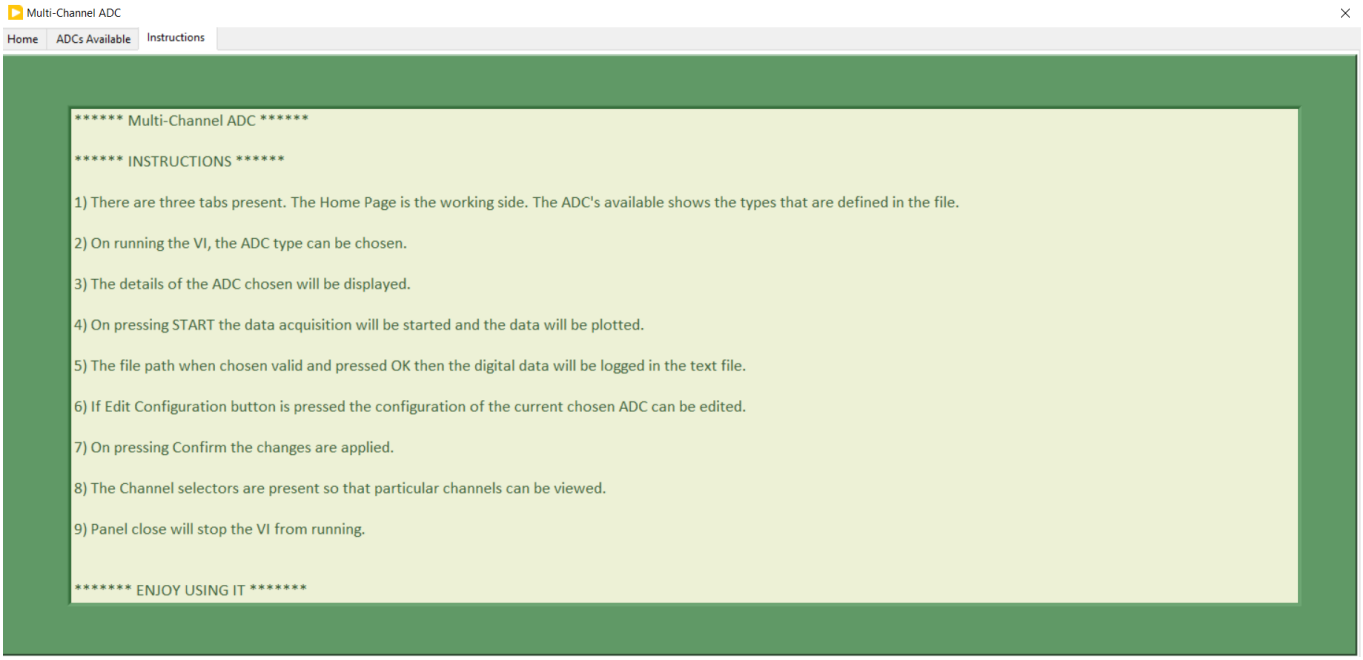
After giving STOP now the ADC types can be switched and the corresponding specifications gets displayed [REQ005].



If Edit Configuration is pressed then the specifications to be edited gets enabled and once Confirmed then the changes gets applied to the current ADC type [REQ003].



By moving to ADCs available page, it will display all the ADCs that are available with its specifications.



Switching to Instructions page, it lists the steps to be followed in operating the Multi-Channel ADC.