Advantech AE Technical Share Document

Date	2022 / 07 / 26	**Related Product**	CODESYS
Category	■ FAQ □ SOP		
Abstract	How to use Labview to acquire ADC data at high speed?		
Keyword	ADC、CODESYS、Data Connection、Labview		
SR#			

Revision History

Date	Version	Author	Reviewer	Description
2022 / 05 / 26	V1.0	Tenjin.Lin	Owen.Chang	CODESYS V3.5 SP1720.2、LabView2016、Visual Studio Community 2010 UP、OS:WIN10
2022/11/01	V2.0	Tenjin.Lin	Owen.Chang	CODESYS V3.5 SP1720.3、LabView2016、Visual Studio Community 2010 UP、OS:WIN10

1 Problem Description & Architecture:

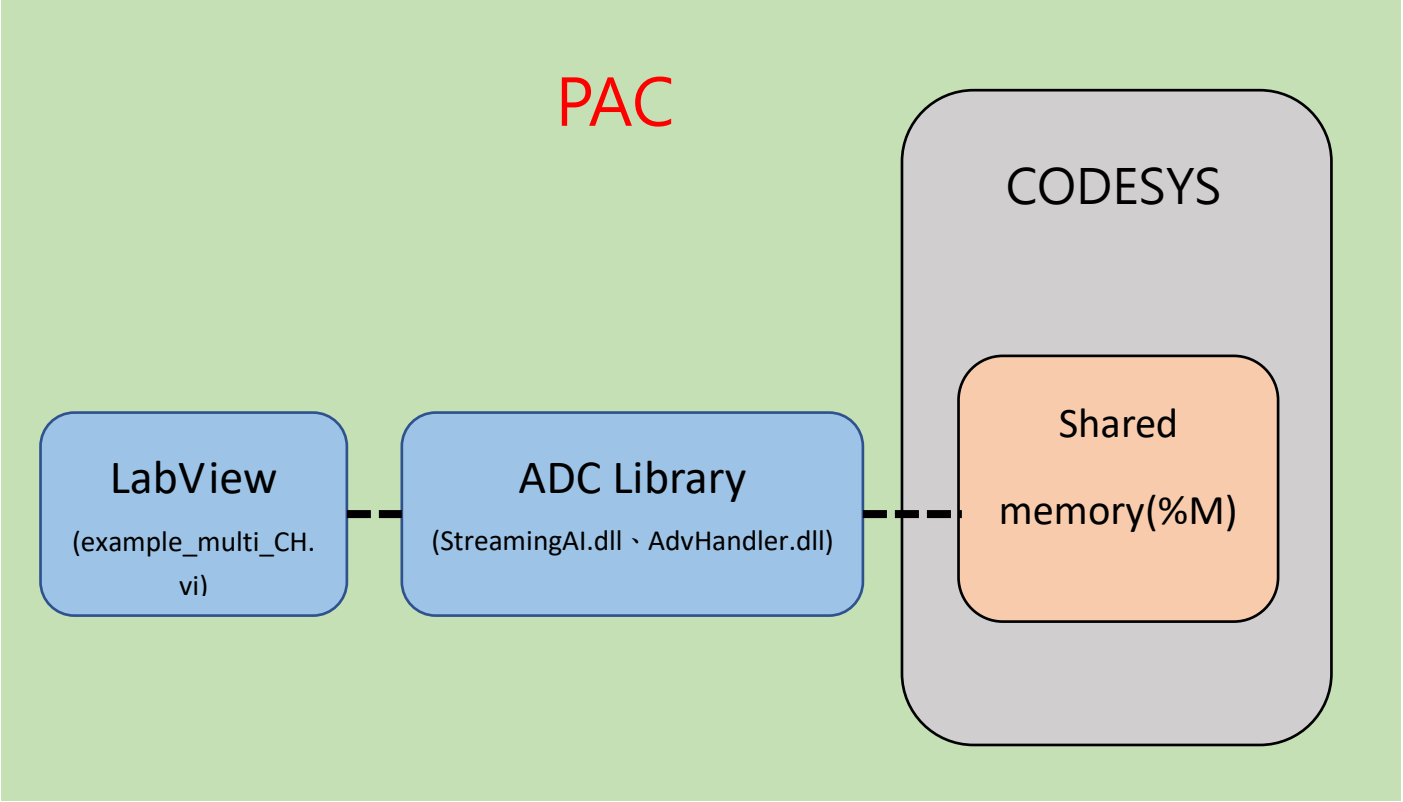
This article introduces how to use Labview to acquire ADC data at high speed(Advantech Data Connection), and introduces how to use the sample program.

- CODESYS Project: Streaming_multi-CH.project
- ADC dll: StreamingAI.dll、AdvHandler.dll
- LabView : streamingAI_MultiCh.vi、streamingAI_MultiCh_analysis.vi、streamingAI_singleCh.vi

2 Brief Solution - Step by Step:

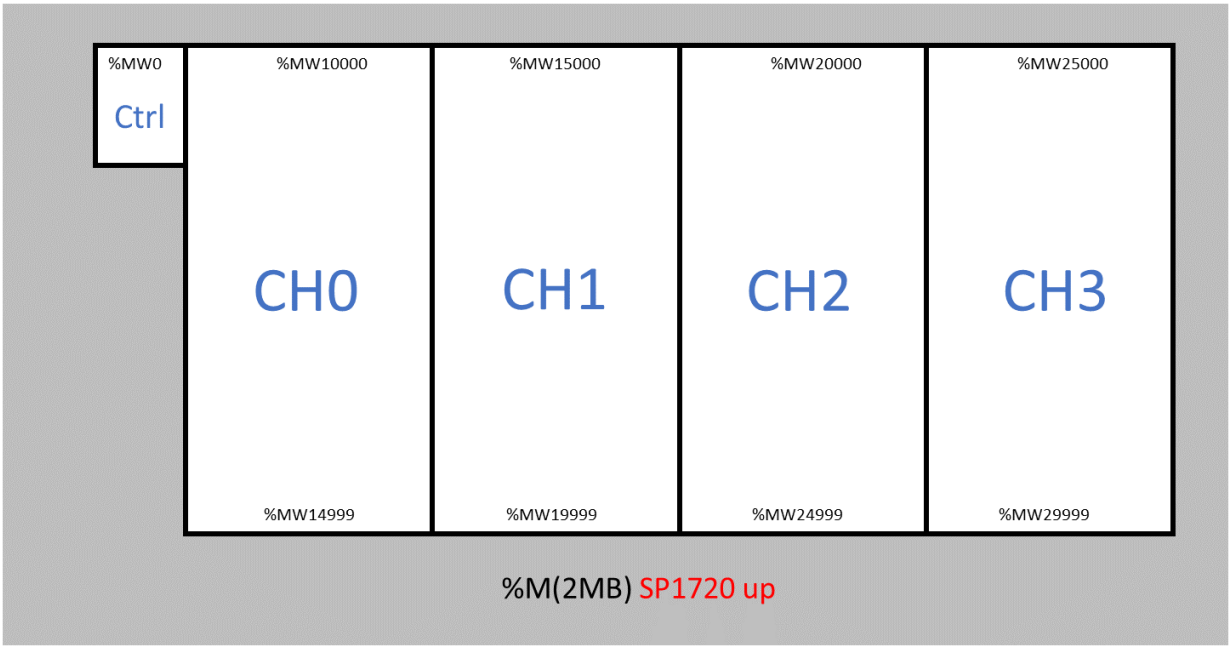
2.1 Architecture introduction

Following is the structure of the ADC in the controller. LabView uses the ADC library to retrieve data from CODESYS RTE.



2.2 On the CODESYS RTE side: our RD creates a mechanism.

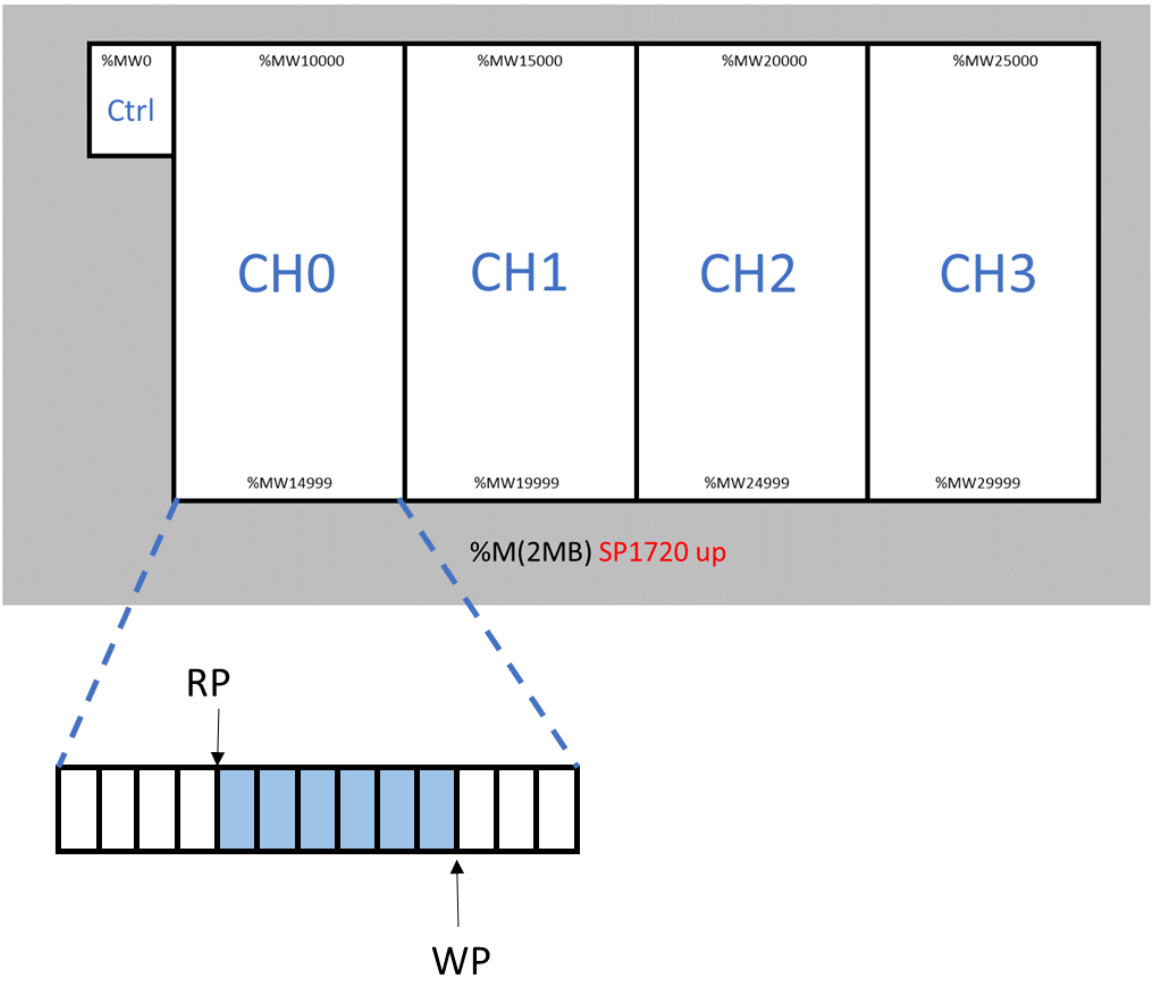
In the temporary storage area %M of CODESYS, we set the first position %MW0 according to the number of CHs of the data source, and the control status area "ChannelBufferControl "is placed on the second Word, and each storage area can be set the space size of the block is currently 5000 words by default, and the actual storage location starts from %10000M and extends later.



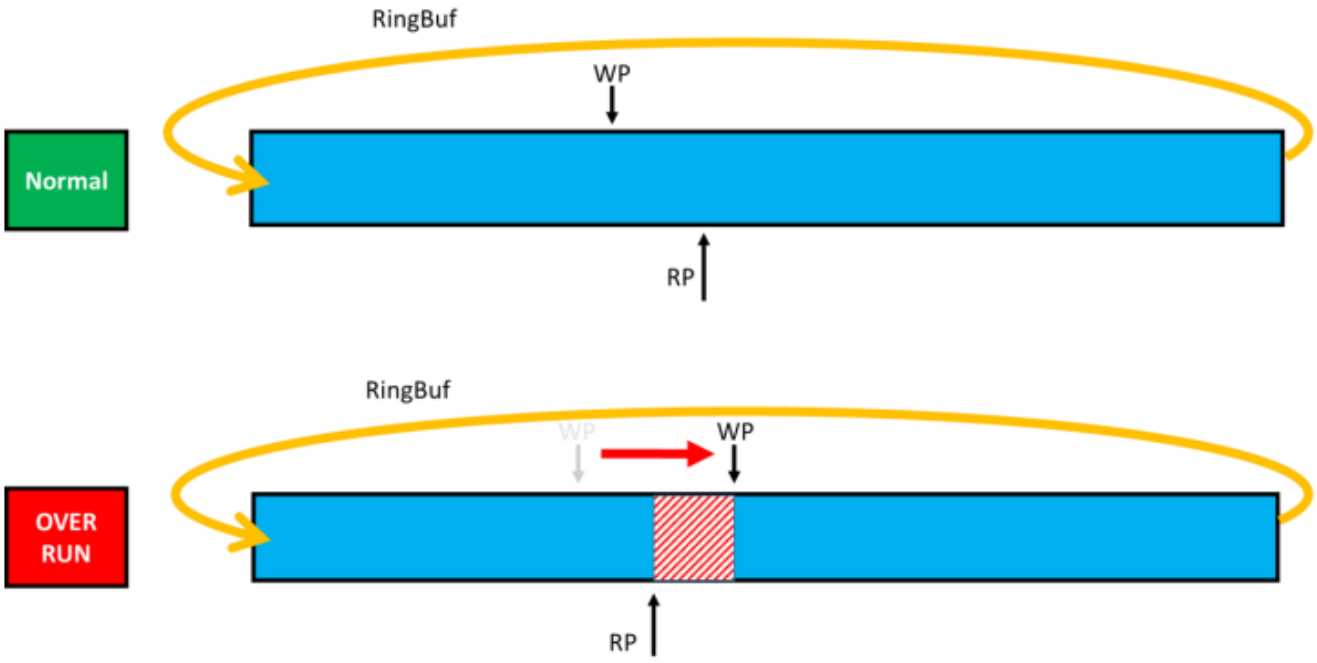
2.3 The operation method of data storage is: fill in the AI value of %M in each cycle, read and put the AI value in the corresponding memory location of each channel (such as %W10000, %W15000, %W20000, %W25000), then increment the value of WP and write the timestamp at the same time. If the software reads the data (ADC), it will also read the value and change the value of RP, which is given by the host. Since the amount of data written and read is the same every time, the starting position of each channel is fixed, and it is possible to know where to start writing and reading in the next cycle, so the mechanism, all channels have independent control bit.

WP: Write Position= Indicates that Codesys fills AI in the position of %M

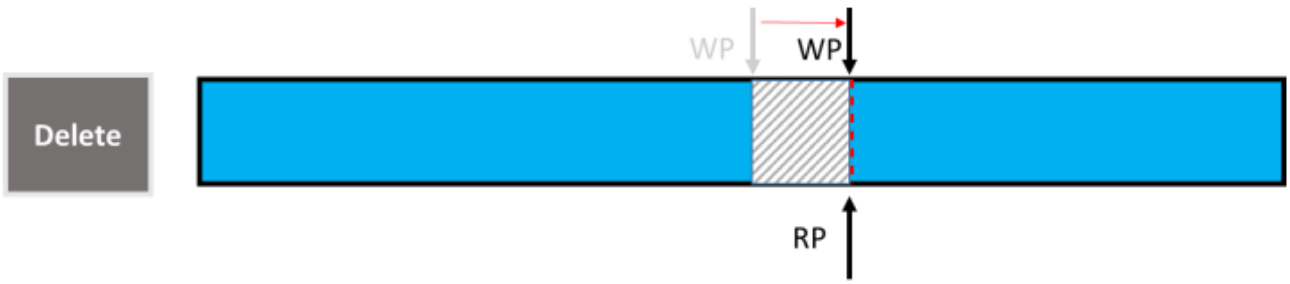
RP: Read Position = indicates the position read by the upper software



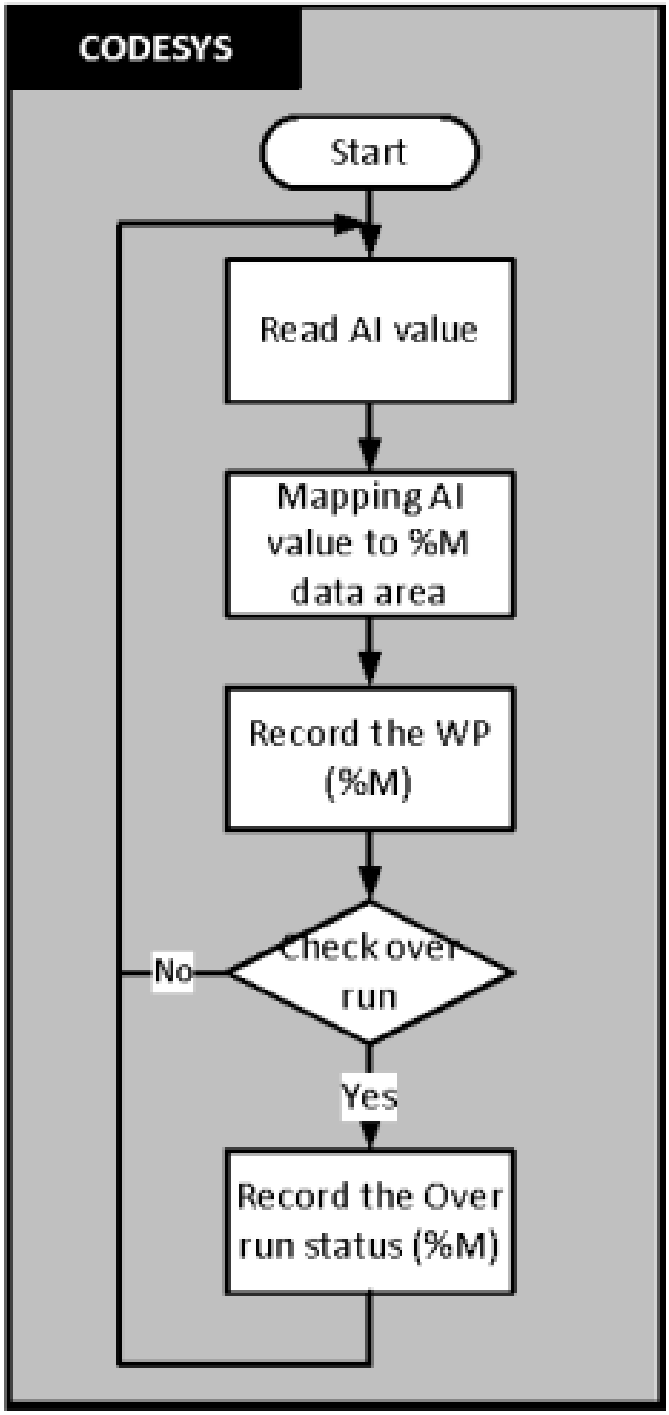
2.4 OverRun flag mechanism: If the old data is not read, the new data is imported and overwritten, which will cause data errors, this error is OverRun. This error occurs because the memory space is a Ring Buffer. When the WP data writing speed exceeds the RP reading speed, data overwrite occurs, triggering the OverRun mechanism. Therefore, the overflow flag is set on the RTE as Display Status This is designed to remind the user to speed up the retrieval speed of the upper-level program.



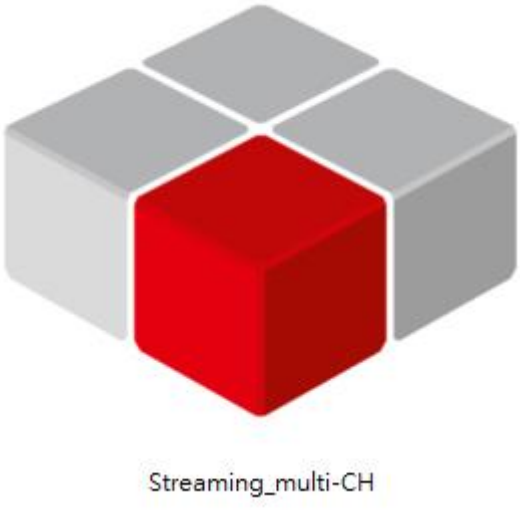
2.5 OverRun occurred: The upper-level disposal is left to the user to decide. The current example is to discard the excess part when the RP catches up with the WP and pull down the overRun flag.



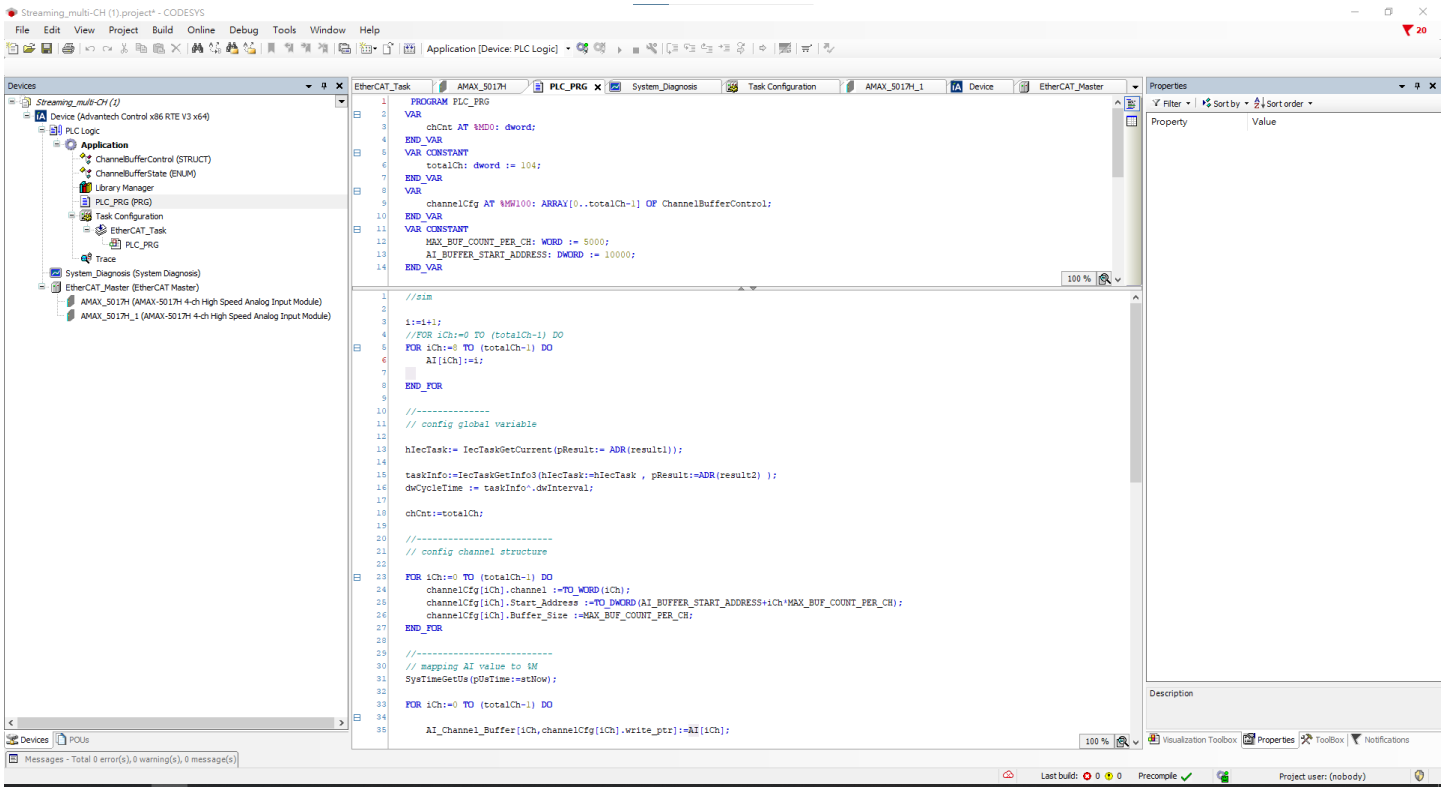
3 CODESYS actual operation:



3.1 Open CODESYS Project



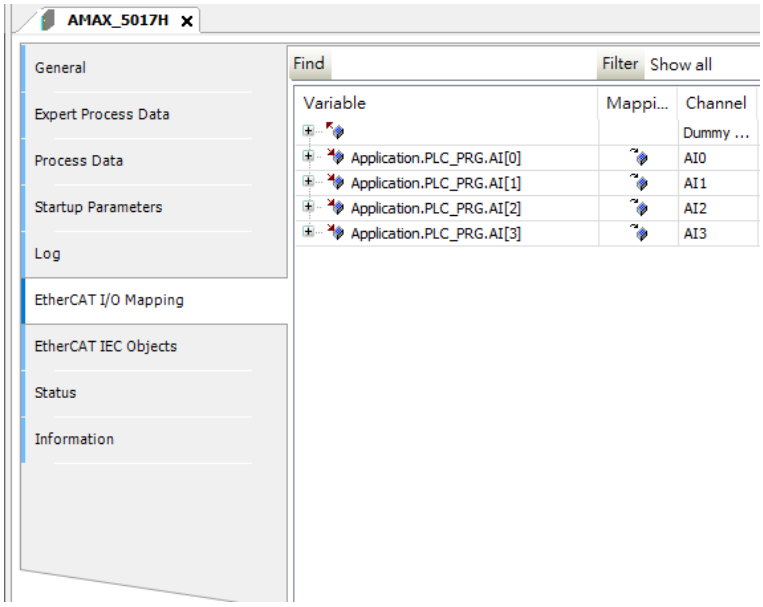
3.2 After opening the project, you can see that there are already many sample codes



3.3 AI arrays represent data acquisition channels. The default setup procedure for the samples is to generate artificial data in I+1 mode. (Read AI value)

You can map AI[0]~AI[3] to terminal block IO and comment 1~7 lines of code, using the actual data source.

```
1   //sim
2
3   i:=i+1;
4
5   FOR iCh:=12 TO (totalCh-1) DO
6       AI[iCh]:=i;
7   END_FOR
8
9   chCnt:=totalCh;
```



3.4 Configure the channel structure, and set the channel number(iCh), data storage location, and Buffer space size here. The value you just set in the parameter setting will be quoted here.(Configure channel)

Parameter:

1	VAR	**chCnt**	%MD0	dword	
2	VAR CONSTANT	**totalCh**		dword	104
3	VAR	**channelCfg**	%MW100	ARRAY[0..totalCh-1] OF ChannelBufferControl	
4	VAR CONSTANT	**MAX_BUF_COUNT_PER_CH**		WORD	5000
5	VAR CONSTANT	**AI_BUFFER_START_ADDRESS**		DWORD	10000

Program:

```
//--------------
// config global variable

hIecTask:= IecTaskGetCurrent(pResult:= ADR(result1));

taskInfo:=IecTaskGetInfo3(hIecTask:=hIecTask , pResult:=ADR(result2) );
dwCycleTime := taskInfo^.dwInterval;

chCnt:=totalCh;

//--------------------------
// config channel structure

FOR iCh:=0 TO (totalCh-1) DO
    channelCfg[iCh].channel :=TO_WORD(iCh);
    channelCfg[iCh].Start_Address :=TO_DWORD(AI_BUFFER_START_ADDRESS+iCh*MAX_BUF_COUNT_PER_CH);
    channelCfg[iCh].Buffer_Size :=MAX_BUF_COUNT_PER_CH;
END_FOR

//--------------------------
```

3.5 The code function below is used to map the AI value to %M and move the WP to that position after placing the value, while miscalculating the write time on each write. (mapping AI value to %M+Record the WP (%M)).

```
//--------------------------
// mapping AI value to %M
SysTimeGetUs(pUsTime:=stNow);

FOR iCh:=0 TO (totalCh-1) DO

    AI_Channel_Buffer[iCh,channelCfg[iCh].write_ptr]:=AI[iCh];
    AI_Channel_Buffer_Time[iCh,channelCfg[iCh].write_ptr]:=TO_DWORD(stNow);
    channelCfg[iCh].write_ptr_tick:=(stNow);


    IF channelCfg[iCh].write_ptr>=MAX_BUF_COUNT_PER_CH THEN
        channelCfg[iCh].write_ptr:=0;
    ELSE
        channelCfg[iCh].write_ptr:=channelCfg[iCh].write_ptr+1;
    END_IF

END_FOR
```

WP: Write Position= Indicates that Codesys fills AI in the position of %M.

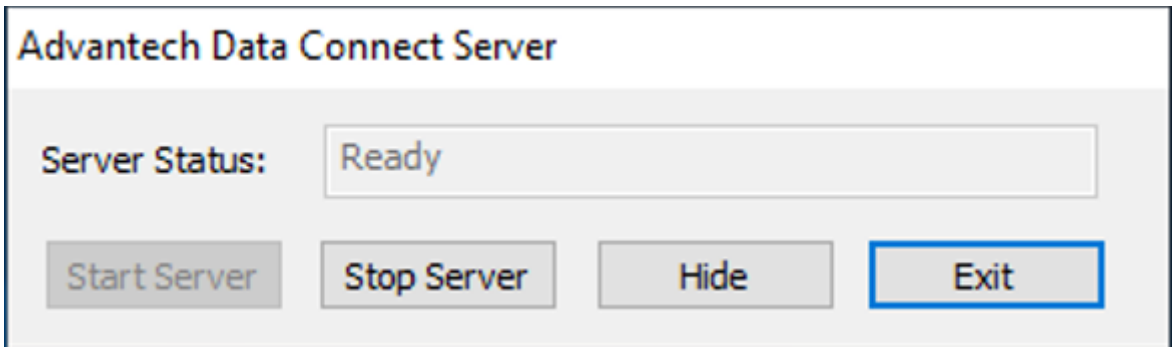
RP: Read Position = indicates the position read by the upper software.

3.6 This section of the program is to judge whether WP has caught up with RP, because if it catches up, the OverRun state will appear, and if OverRun appears, it will notify the upper program that OverRun occurs.

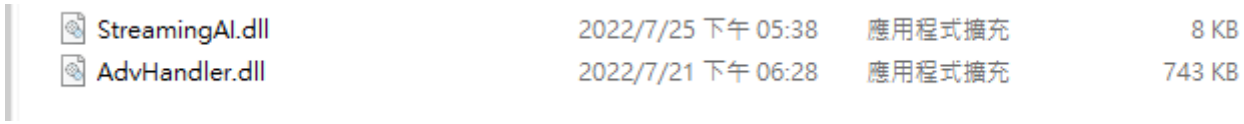
```
//---------------------------
// handle over run

FOR iCh:=0 TO (totalCh-1) DO
    IF ((channelCfg[iCh].Status AND SHL(DWORD#1,ChannelBufferState.isOverRun)) >0 ) THEN
        ;
    ELSE
        // get water level
        IF channelCfg[iCh].write_ptr>=channelCfg[iCh].read_ptr THEN
            waterLevel[iCh]:=channelCfg[iCh].write_ptr-channelCfg[iCh].read_ptr;
        ELSE
            waterLevel[iCh]:=MAX_BUF_COUNT_PER_CH  + channelCfg[iCh].write_ptr - channelCfg[iCh].read_ptr;
        END_IF

        IF (waterLevel[iCh]=MAX_BUF_COUNT_PER_CH-1) THEN
            channelCfg[iCh].Status:=channelCfg[iCh].Status OR SHL(DWORD#1,ChannelBufferState.isCatchup);
        END_IF

        IF ((channelCfg[iCh].Status AND SHL(DWORD#1,ChannelBufferState.isCatchup)) >0 ) THEN
            IF (channelCfg[iCh].write_ptr>channelCfg[iCh].read_ptr) THEN
                channelCfg[iCh].Status:=channelCfg[iCh].Status OR SHL(DWORD#1,ChannelBufferState.isOverRun);
                channelCfg[iCh].Status:=channelCfg[iCh].Status AND( NOT(SHL(DWORD#1,ChannelBufferState.isCatchup)));
            END_IF
        END_IF
    END_IF


END_FOR
```

4 Start ADC server

The ADC server will be bundled with the RTE version, and the version must be SP1720.2 or above.



5 At the same time, two DLLs (StreamingAI.dll, AdvHandler.dll) must be introduced. The main functions of these two DLLs are to read and write to the ADC server, and provide APIs for the upper software to use.



5.1 StreamingAI mainly provides five APIs for upper-layer programs to use.

{init() is used for initialization,

resetAI() is used to reset the positions of WP and RP in a single channel,

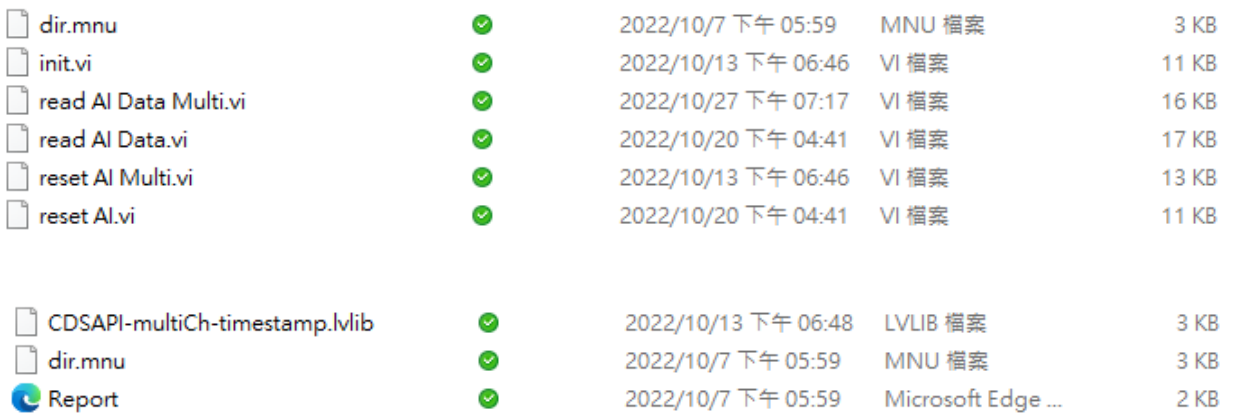
resetAIMulti() is used to reset the positions of WP and RP in multiple channels,

readAIData () for single-channel read data,

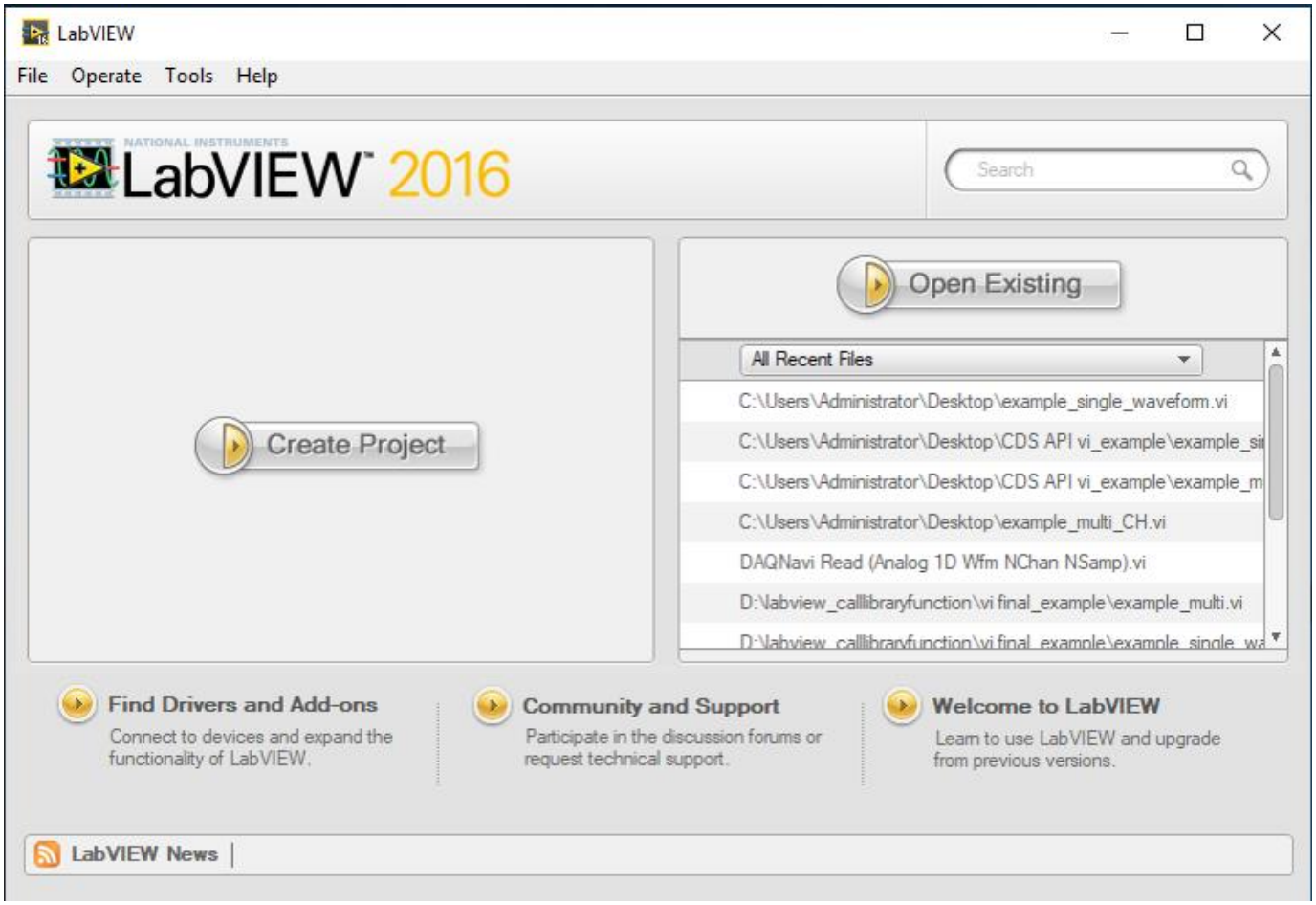
readAIDataMulti () for multi-channel read data.}

```
CDS_API void init();
CDS_API unsigned long resetAI(unsigned long i_iCh);
CDS_API unsigned long resetAIMulti(unsigned long* i_iChList, int i_iChTotal);
CDS_API unsigned long readAIData(unsigned long i_iCh,int i_iSampleTotal, void* o_pValue, int* pLength,
        long long* o_pStartTick, unsigned long* o_dwCycleTime);
CDS_API unsigned long readAIDataMulti(unsigned long* i_iChList, int i_iChTotal, int i_iSampleTotal,
        unsigned short** o_pValueArray, int* pLengthArray, long long* o_pStartTickArray, unsigned long* o_dwCycleTime);
```

5.2 At the same time, VI files are provided in the example. These five VI files are used to call the above five APIs so that LabView can use them. In this example, these five APIs are compiled into a CDSAPI-multiCh-timestamp.lvlib, which is convenient for users to call.



6 Open LabView2016

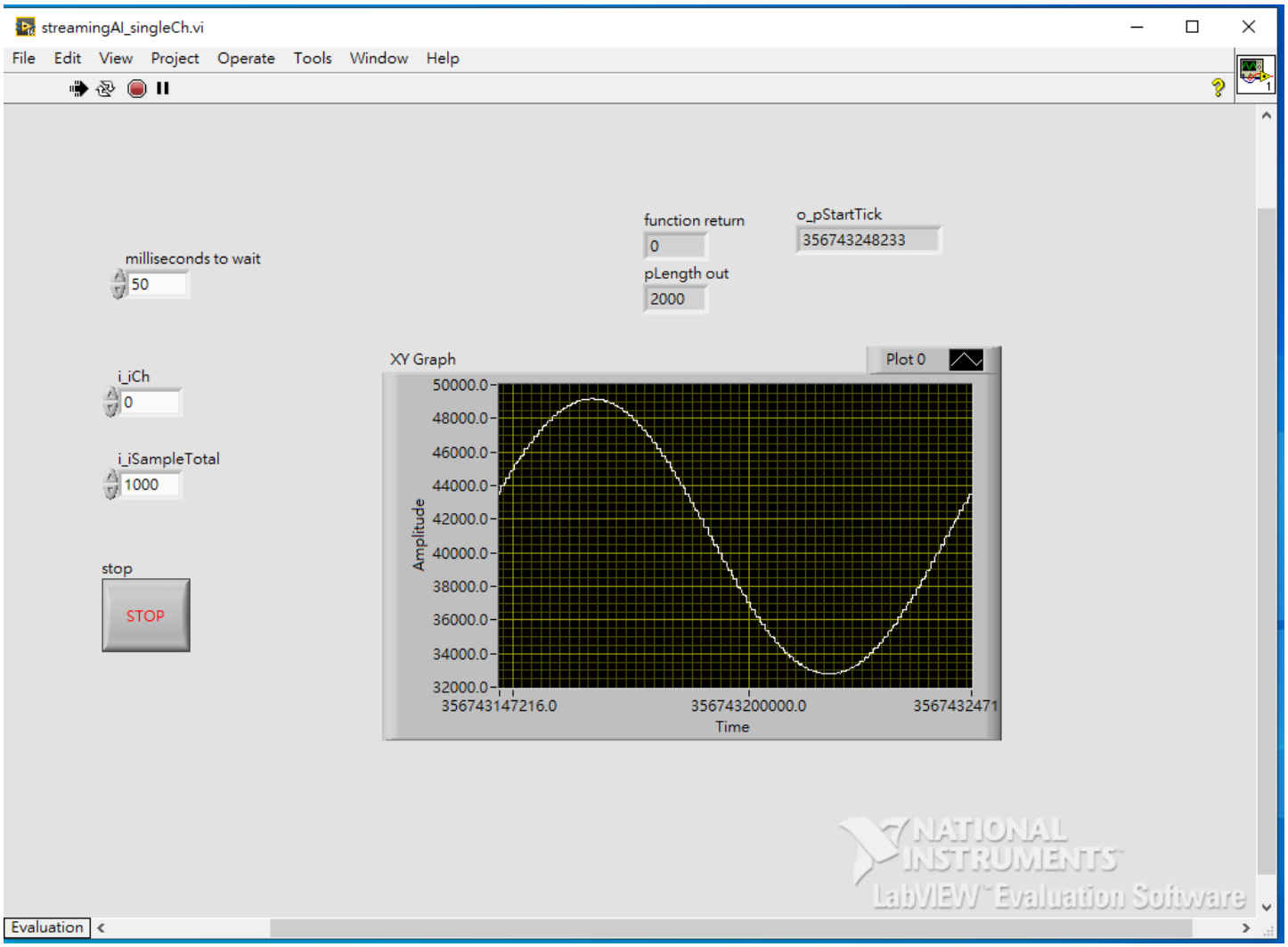


6.1 We provide three examples streamingAI_MultiCh.vi、streamingAI_MultiCh_analysis.vi and streamingAI_singleCh.vi, these three examples are multi-CH and single-CH, and there is another example with an analysis function.

streamingAI_MultiCh.vi	2022/10/27 下午 07:19	VI 檔案	25 KB
streamingAI_MultiCh_analysis.vi	2022/10/28 下午 04:51	VI 檔案	30 KB
streamingAI_singleCh.vi	2022/10/27 下午 07:21	VI 檔案	21 KB

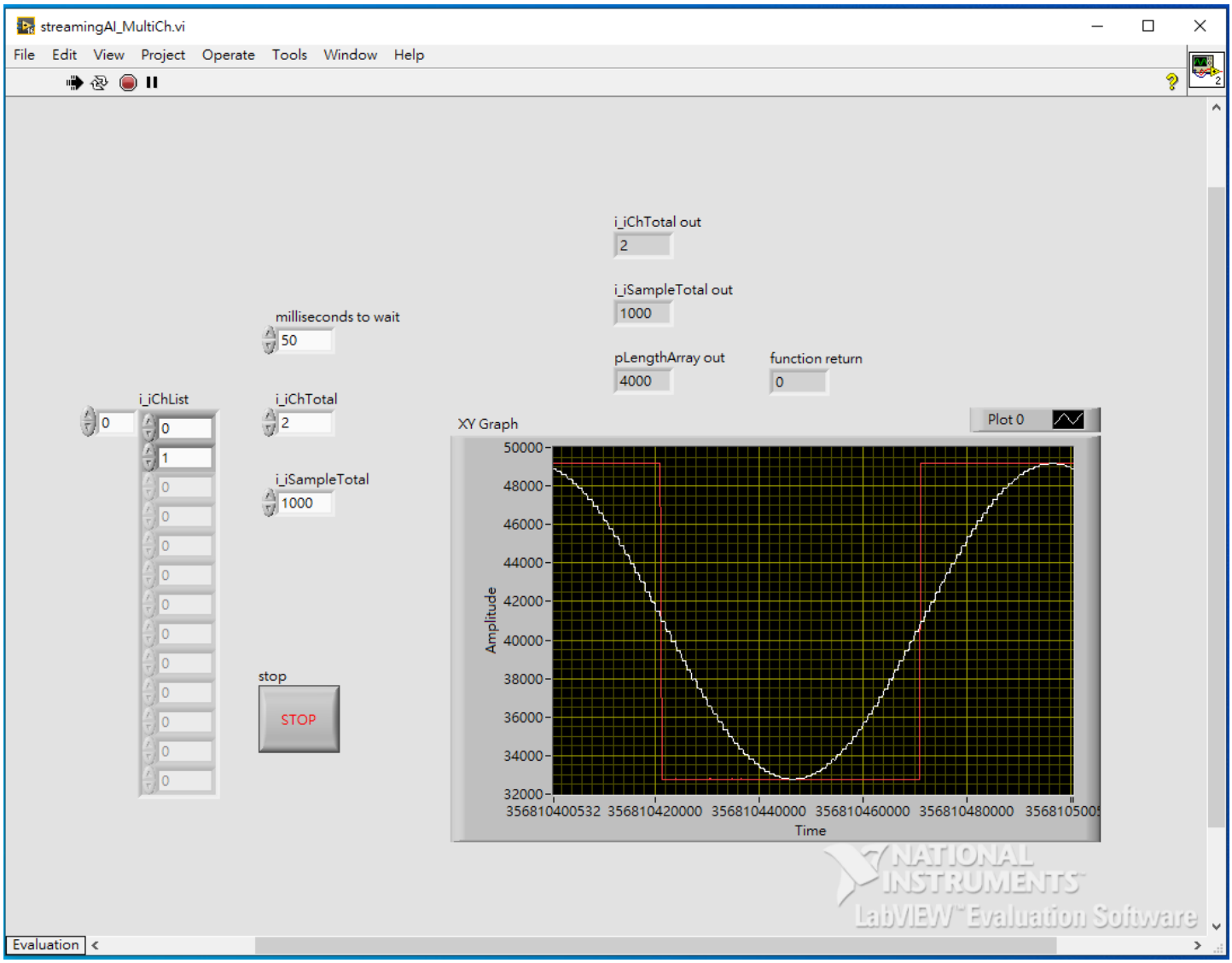
6.2 streamingAI_singleCh.vi

This is an example of a single channel and drawing, mainly to set the iCh to be read, the amount of data i_iTotal, and the scan time millseconds to wait.



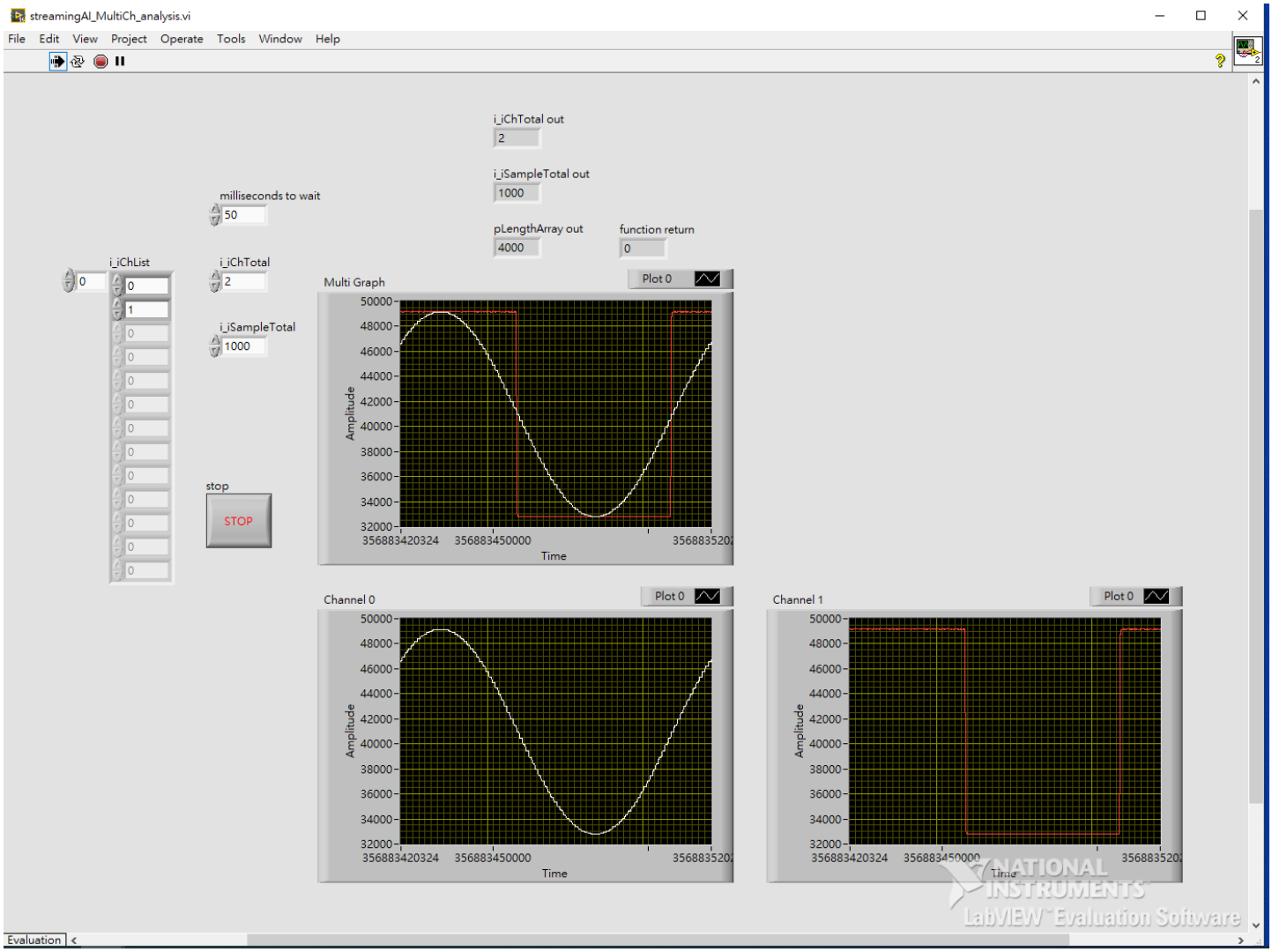
6.3 streamingAI_MultiCh.vi

This is an example of multi-channel data, mainly to set the iCh to be read, the amount of data i_iTotal, and the scan time millseconds to wait.



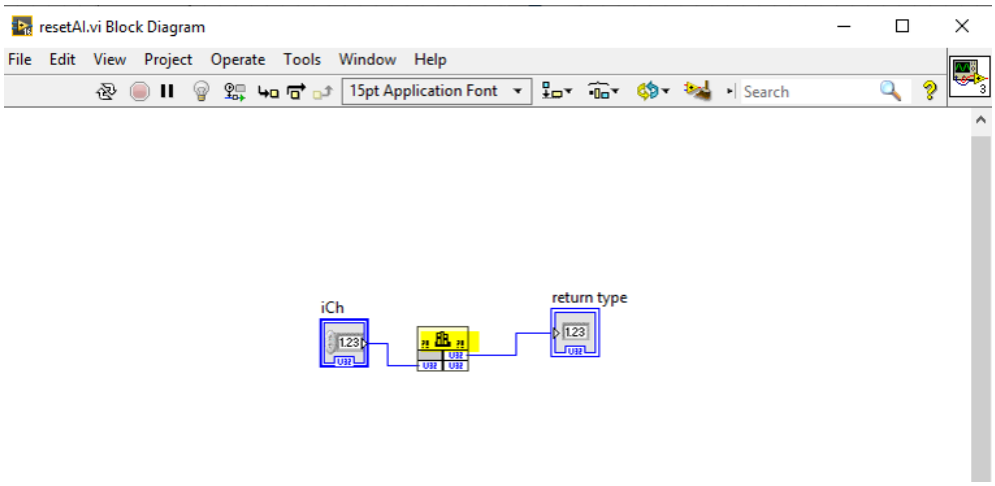
6.4 streamingAI_MultiCh_analysis.vi

This is an example of multi-channel data, and the measurement results of each channel are displayed separately. Mainly set the iCh to be read, the amount of data i_iTotal, and the scan time in milliseconds.

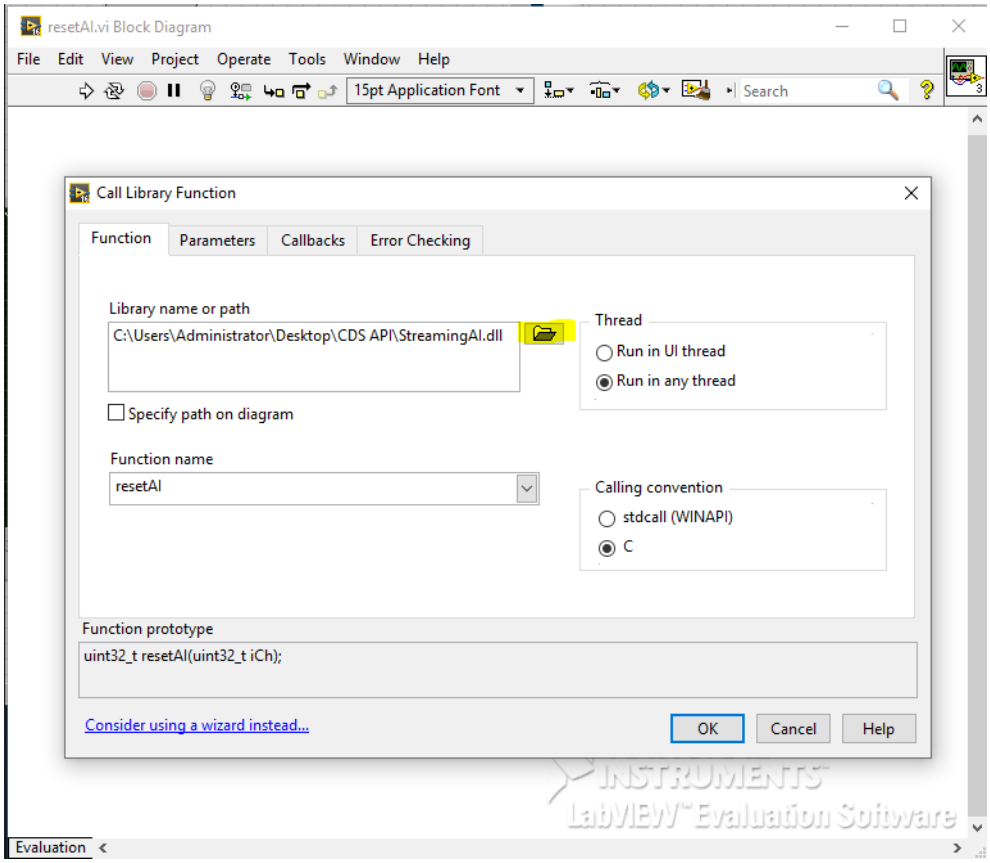


6.5 If you want to re-introduce the DLL method

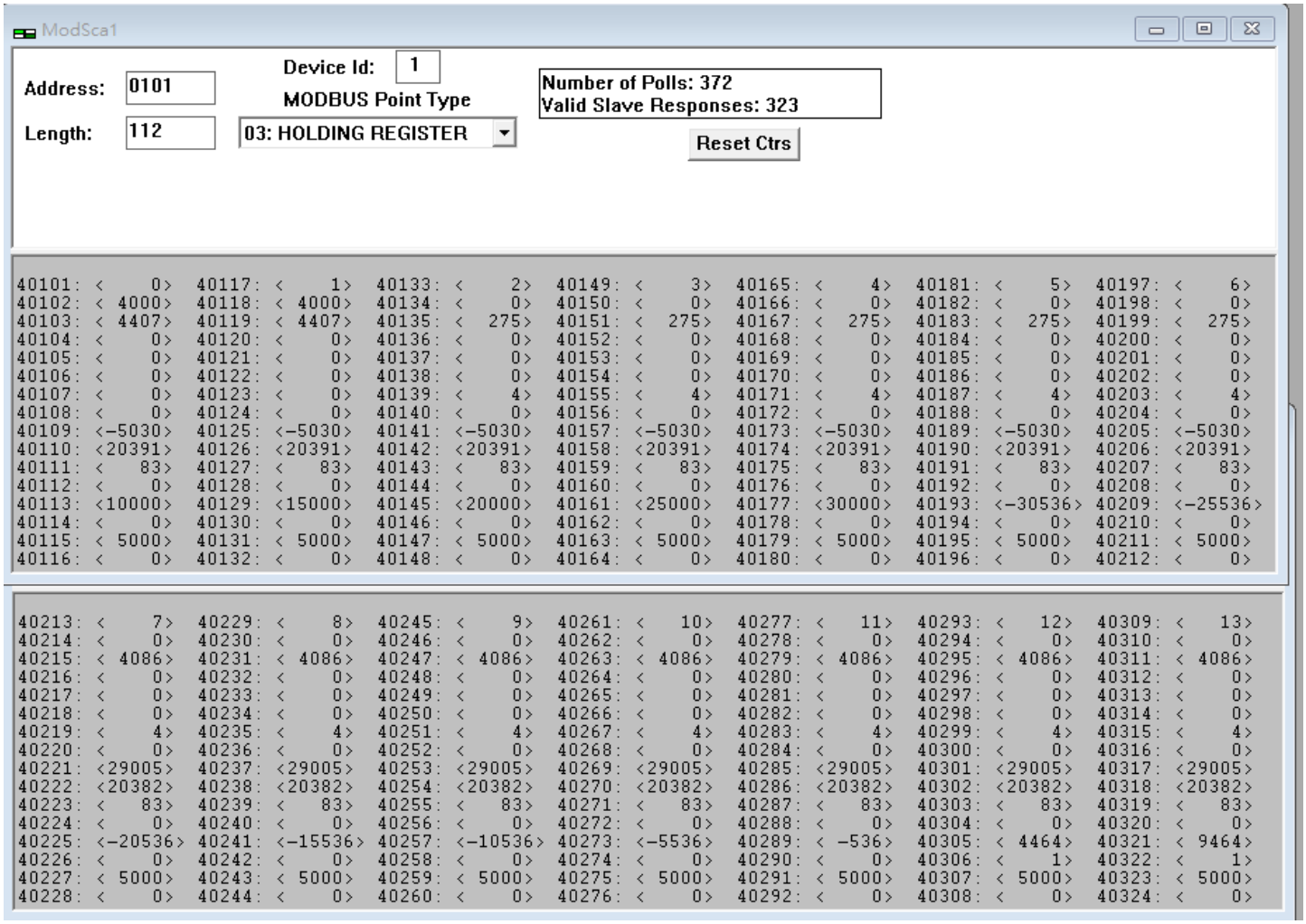
6.5.1 Open the "Block Diagram" of vi and the double click the object which mark in yellow below.



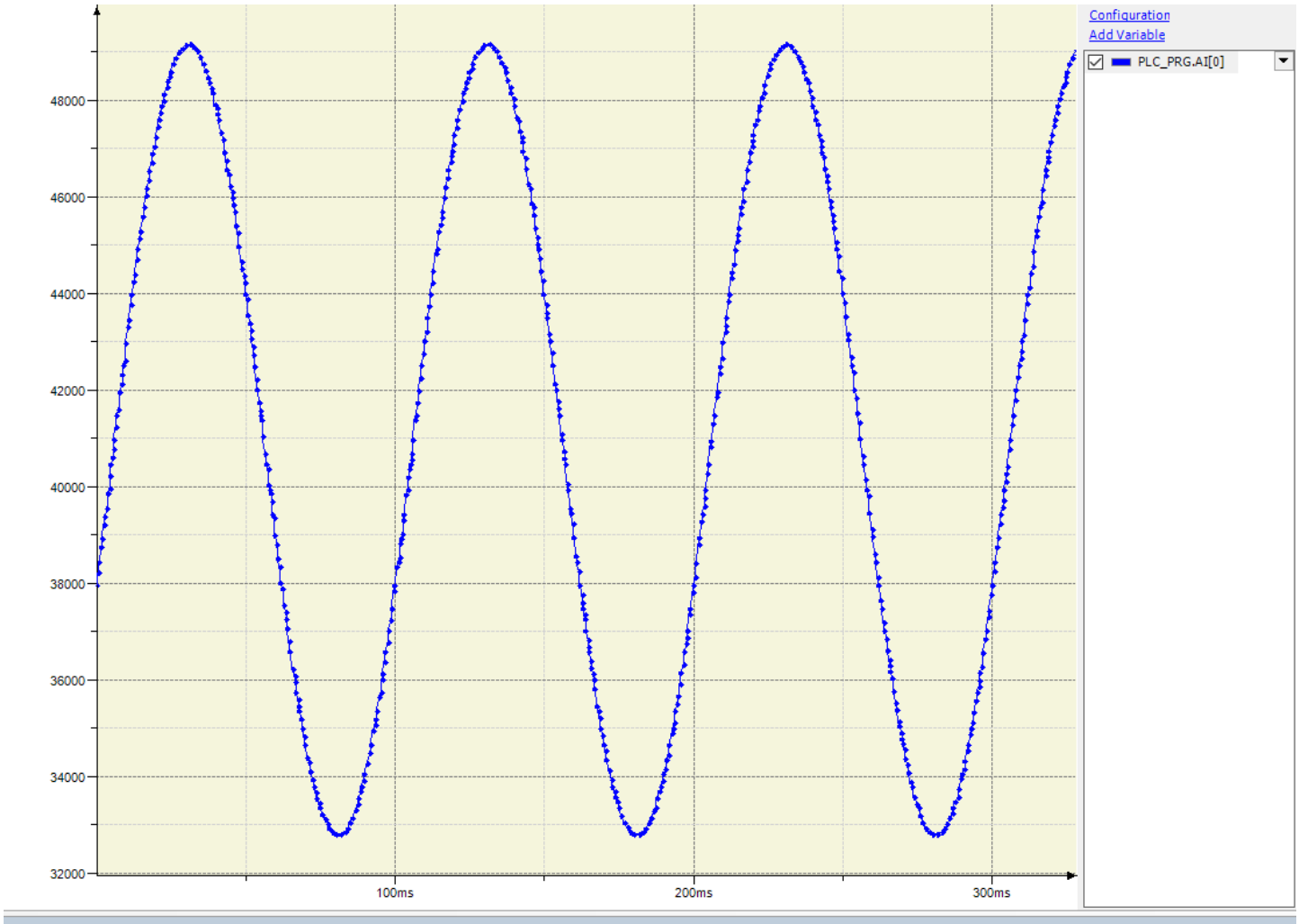
6.5.2 Click the path selector and then to choose the StreamingAI.dll where you saved.



6.5.3 You can use the third-party MODBUS TCP program to know the current status of ADC operation.



6.5.4 On the CODESYS side, you can use CODESYS Trace to view the current status.



Reference:

(1) CODESYS Online Help