# 使用C#通过AdsNotification的方式

# 实现批量数据的采集

1. **概述**

针对很多客户在开发过程中需要对过程数据进行较高速率的采集（如1ms、500μs等），并通过通讯的方式将数据上传至C#上位机进行数据存储及后续分析，为保证数据的连续性和可靠性，编写此样例程序。

此样例程序上位机部分为winform客户端程序，采用visual studio2019开发，目标框架.Net Framework4.7.2。

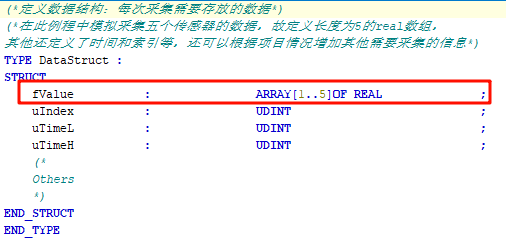
PLC程序包含TC3和TC2两份源代码。

1. **程序功能设计**

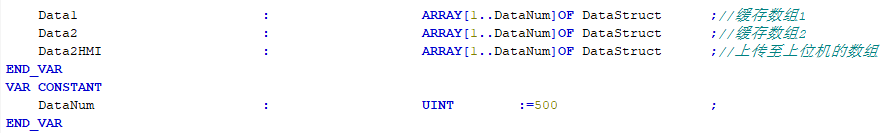
样例程序模拟采集5个传感器的数据，1ms采集一次，C#上位机接收采集数据并进行数据处理（显示等）。

1. **PLC实现逻辑**

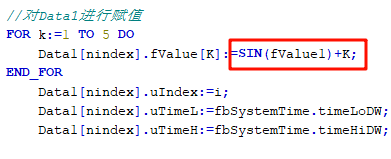
* 将PLC的任务周期固定为1ms（亦即程序代码1ms执行一次<实际代码执行所用时间：根据实际代码量定:譬如100us）；
* 定义数据结构，如下图所示，即每次采集需要存放的数据，例如：传感器采集数据、时间信息、索引值等（该数据结构可根据实际采集的数据进行调整）；



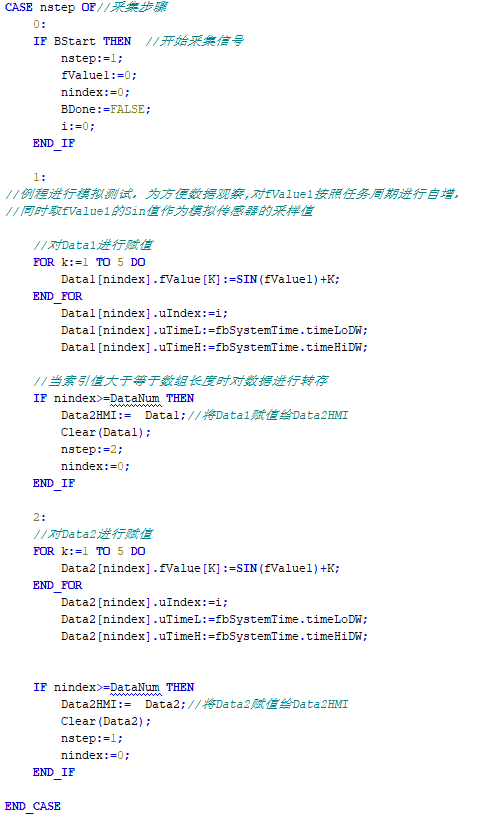
* 定义数组，此处定义两个缓存数据Dta1和Data2，用于交替存储每1ms采集的数据，同时定义Data2HMI数组用于和上位机进行数据交互；此处将数组长度定义为500，即每个数组存放500ms的数据；



* 数据处理设定逻辑：由于本例程进行模拟测试，为方便数据观察,对fValue1按照任务周期进行自增（fValue1:=fValue1+0.01;），同时在每个任务周期取fValue1的Sin值作为模拟传感器的采样基准值，5个传感器的采样值则按照索引在fValue1的Sin值的基础上分别进行偏置；数据结构中的时间数据，在PLC程序中进行了赋值，但是在C#上位机上未体现对其数据的处理，在此主要体现结构体数组采集的方式；



* 交替缓存逻辑，通过以下程序实现数据交替缓存的逻辑，主要逻辑：开始采集后，先将数据放入Data1中进行缓存，缓存500ms后，将Data1中的数据一次性赋值给Data2HMI；同时，数据切换到Data2进行缓存，当缓存500ms后，再将Data2中的数据一次性赋值给Data2HMI；依次往复；Data2HMI则用于PLC和上位机进行数据交互。



1. **C#上位机实现逻辑**

采用ADS通知AdsNotification的方式与PLC中的**Main.Data2HMI**进行数据交互；C#客户端发送ADS请求，订阅Data2HMI数据，当Data2HMI数据发生改变的时候，通知C#客户端程序，C#客户端通过回调函数接收并处理数据。（ADS详细通讯过程在此不进行详述）

* 根据PLC 的数据结构，定义C#数据结构：在此部分需要先按照PLC中**DataStruct**的数据结构定义结构体，再按照**Main.Data2HMI**数组长度定义结构体数组，定义方式如下：



* 注册事件处理函数：在程序初始化中进行ADS通知事件的注册，如下图所示：

文本

描述已自动生成

* 根据PLC变量进行通知注册：点击C#界面的“订阅”按钮实现对**Main.Data2HMI、Main.BDone**两个变量的通知订阅，其中核心是对**Main.Data2HMI**的通知订阅，**Main.BDone**为采集完成标志，用于辅助完成一些其他处理逻辑（可根据情况是否需要）如下图所示：

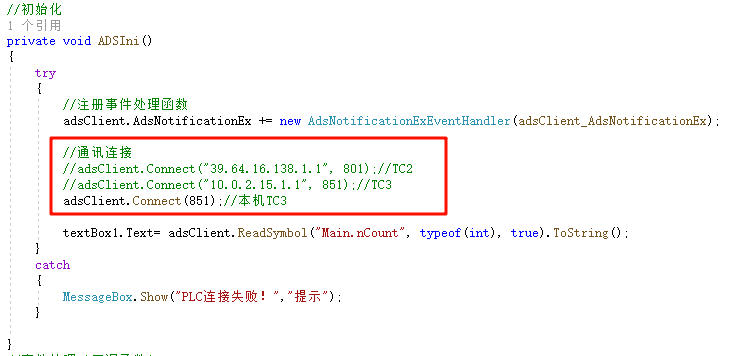


1. **实现效果**

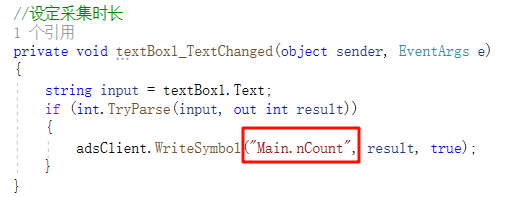
在此样例程序中，用正弦曲线模拟需要采样的5个传感器的数值，上位机接收采集的数据并通过实时曲线进行显示，通过观察曲线的状态可判断数据的连续性，通过曲线的点数可以判断数据的稳定性。

程序使用如下：

* 运行PLC程序；
* 修改C#通讯连接参数；



* 运行C#程序；
* 点击订阅按钮完成对PLC数据的通知注册；
* 设定采集时长；

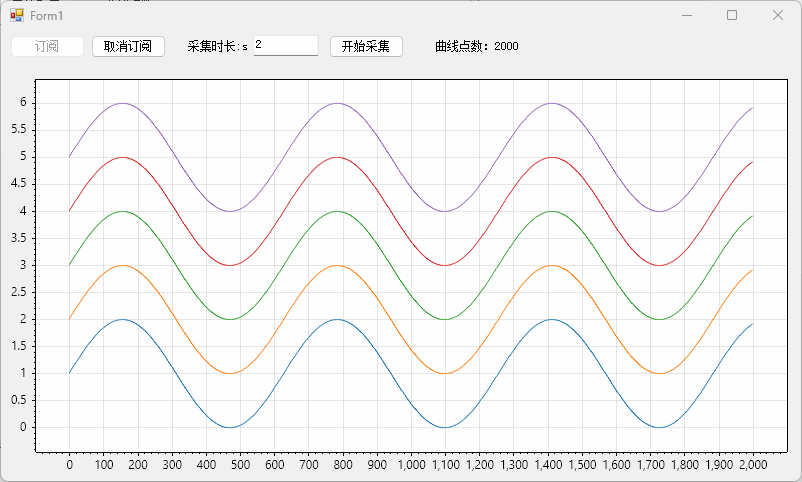


* 点击开始采集按钮；

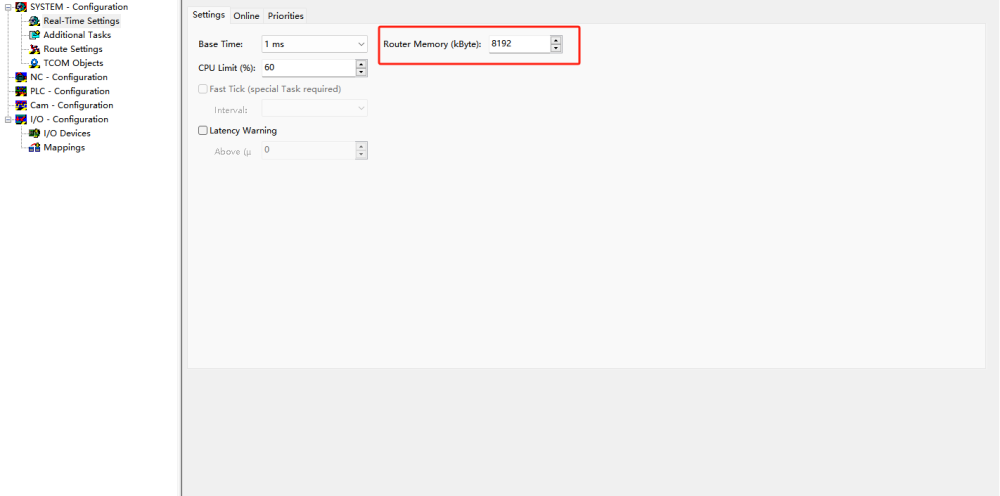
图形用户界面, 文本, 应用程序

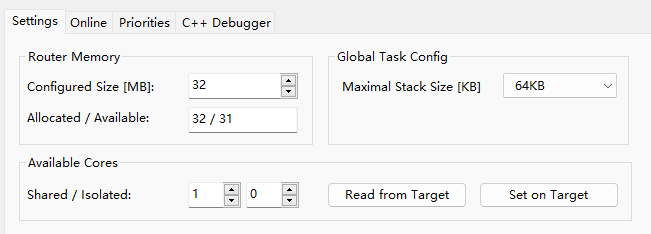
描述已自动生成

* 曲线点数显示接收的单条曲线的点数，样例程序是1ms一次，因此曲线点数应是采集时长的1000倍，根据点数可以判断数据的可靠性，是否存在数据丢包的情况；



1. **其他问题**
   * 1. 此样例程序只实现了基本的数据采集、接收、处理（显示）逻辑，程序并没有包含完整的错误处理等机制，在实际使用时需注意；
     2. 对于采集数据量很大的情况，需要注意以下问题：
     3. Router Memory：特别是TC2，根据数据量的大小可能需要调整该数值，否则可能无法有效注册数据的通知；





* + 1. 上位机数据处理过程：在数据量大的情况下，上位机的数据处理可能会比较耗时（取决于数据处理的复杂程度），如果回调很频繁，需要修改上位机的处理逻辑，避免数据回调处理不过来；
    2. 总之，对于数据量较大的情况需要对上下位机的程序进行合理的搭配，调整回调时间及通讯参数，同时可能还需要注意网络传输的情况。