УДК 62-5, 681.5.015, 004.942

Исследование способов передачи информации в системах сбора и обработки данных в MATLAB

В. А. Краснокутский, Ю. С. Достлев
ГОУ ВПО «Донецкий национальный технический университет», г. Донецк
kras_v_a@mail.ru

Аннотация

Статья посвящена исследованию применения пакета MATLAB для обработки и анализа данных, поступающих от внешних устройств в реальном времени. Рассмотрены вопросы организации передачи данных через интерфейс последовательного порта. Приведены практические схемы и алгоритмы передачи аналоговых данных для обработки их средствами пакета MATLAB. В качестве устройства связи между внешним устройством и пакетом MATLAB используется платформа Arduino на микроконтроллере ATmega32. Показаны эффективные возможности проектирования систем цифровой обработки данных в интерактивном режиме с использованием простых систем сбора данных в реальном времени.

Введение

MATLAB – язык высокого уровня для научно-технических вычислений [1]. Он объединяет в себе численные расчеты, визуализацию и программирование. Благодаря открытой архитектуре в MATLAB включают дополнительные специализированные пакеты Toolboxes, содержащие наборы функций для различных задач. Пакеты предоставляют полный набор возможностей для разработки, анализа и тестирования моделей аналоговых и цифровых систем, систем связи и передачи информации и др.

MATLAB является прекрасным средством для научных разработок, а также для обучения студентов по специальностям, связанным с информационными технологиями. При решении прикладных задач с помощью MATLAB легко выполняются:

- математические расчеты;
- разработка алгоритмов;
- моделирование;
- анализ данных и визуализация;
- научная и инженерная графика;
- разработка приложений.

При разработке устройств цифровой обработки сигналов удобно использовать реальные сигналы прототипов плат таких устройств [2, 3, 4]. Для этого необходимо обеспечить обмен данными между системой MATLAB и внешним устройством. В MATLAB такая связь осуществляется с помощью интерфейса RS_232 последовательного порта персонального компьютера (ПК) [5]. Однако в современных компьютерах последовательные порты (COM) для подключения внешних устройств практически не используются. Периферийные устройства в настоящее время подключаются через интерфейс USB.

Целью работы является исследование возможностей подключения MATLAB к внешнему устройству и передачи данных между ними. Для выполнения поставленной цели необходимо выполнить ряд задач:

- разработать структуру устройства связи MATLAB с внешним устройством;
- выбрать платформу устройства связи;
- разработать тестовые программы;
- проверить работу системы связи.

Структура устройства связи

Структурная схема устройства связи приведена на рис.1.

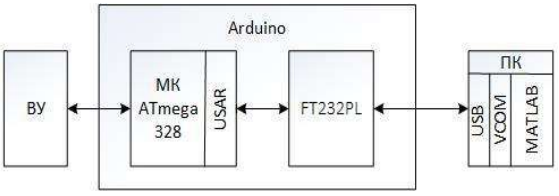

Рисунок 1 – Структурная схема устройства связи

В качестве платформы для подключения внешнего устройства (ВУ) используется Arduino nano [6] Это простая плата для разработки электронных устройств, которая состоит из микроконтроллера (МК) ATmega 328 [7], элементов обвязки для подключения различных схем. Загрузка программы в микроконтроллер осуществляется с помощью программы Boot Loader, которая зашита в памяти

микроконтроллера и не требует внешнего программатора. Устройство программируется через USB из среды программирования Arduino IDE. Среда программирования - это бесплатная программа, которая обеспечивает написание программы, трансляции и загрузки исполняемого файла в микроконтроллер Arduino. Язык программирования устройств Ардуино основан на C/C++, скомпонован с библиотекой AVR Libc и позволяет использовать любые ее функции [8]. Кроме того, разработано множество библиотек, которые работают с большим количеством периферийных устройств, подключаемых к микроконтроллерам. Использование библиотек позволяет во многих случаях скрыть от разработчика особенности аппаратурной и программной реализации функций управления устройством. Но, в то же время, ничем не ограничиваются возможности самостоятельной разработки программ для управления периферийными устройствами.

Микроконтроллер имеет встроенный 10 разрядный аналого-цифровой преобразователь (АЦП), что позволяет вводить и осуществлять предварительную обработку аналоговых сигналов. Связь микроконтроллера ATmega 328 с персональным компьютером осуществляется с помощью последовательного порта USART. Сигналы последовательного порта Tx и Rx поступают на микросхему FT232RL, которая является преобразователем интерфейса USB в последовательный интерфейс USART [9,10]. Для работы с микросхемой FT232RL необходимо в персональный компьютер (ПК) загрузить драйвер, который создает виртуальный последовательный COM - порт, через который осуществляется связь с MATLAB. Драйвер для разных операционных систем доступен на сайте производителя микросхемы. Теперь MATLAB может обращаться к плате Arduino, как к последовательному порту.

Работа MATLAB с последовательным портом

Для работы с последовательным портом необходимо выполнить следующий ряд действий [5]:

– создать объект последовательного порта;

– подключиться к устройству;

– настроить свойства последовательного порта;

– выполнить операции записи и чтения данных;

– отключить устройство.

Объект последовательного порта для определенного COM-порта создается с помощью функции serial('COM4') с указанием номера порта, в нашем случае виртуального.

Используя функцию fopen подключить объект последовательного порта к устройству. Далее следует настройка свойств объекта порта. Важно, чтобы свойства портов MATLAB и Arduino были одинаковыми. Значения свойства присваиваются с помощью функции set или точечной нотации.

Теперь вы можете записывать данные в устройство с помощью функции fprintf или fwrite, а также считывать данные с устройства с помощью функций fgetl, fgets, fread, fscanf или readasync.

Когда объект последовательного порта больше не нужен, вы должны отключить его от устройства с помощью функции fclose, удалить из памяти с помощью функции delete и удалить его из рабочей области с помощью команды clear.

В простом примере обмена данными показаны все рассмотренные действия:

```
s = serial('COM3');
set(s,'BaudRate',9600);
fopen(s);
fprintf(s,'!');
out = fscanf(s);
fclose(s)
delete(s)
clear s
```

В табл.1 и табл. 2 Приведены основные функции их назначение для записи данных в последовательный порт и чтения данных из последовательного порта.

Таблица 1 – Функции записи данных

Имя	Назначение функции
fprintf	Запись текста в устройство
fwrite	Запись двоичных данных в устройство

Таблица 2 – Функции чтения данных

Имя	Назначение функции
fgetl	Чтение одной строки с отбрасыванием терминатора
fgets	Чтение одной строки включая и терминатор
fread	Чтение двоичных данных
fscanf	Считывание данных с устройства и форматирование в виде текста

При работе с последовательным портом необходимо учитывать, что при передаче данных MATLAB не взаимодействует непосредственно с последовательным портом, он передает данные через два буфера (рис. 2). Один буфер - входной, другой - выходной.

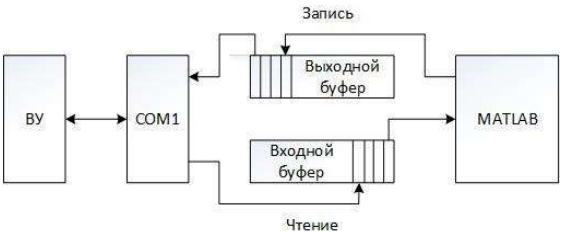

Рисунок 2 – буферизация при передаче данных по последовательному порту

Входной буфер (InputBufferSize) - это компьютерная память, выделенная объектом последовательного порта для хранения данных, которые должны быть считаны с устройства Буфер вывода (OutputBufferSize) - это память компьютера, выделенная объектом последовательного порта для хранения данных, которые должны быть записаны на устройство.

При записи в устройство MATLAB помещает данные в выходной буфер, образуя очередь. Последовательный порт COM выбирает из буфера очередной байт данных с начала очереди, таким образом, реализуя процедуру: первый пришел – первый ушел. Если выходной буфер заполняется быстрее, чем опустошается, то может произойти переполнение буфера и потеря данных.

Чтение данных осуществляется через входной буфер. И в этом случае может возникнуть аналогичная ситуация переполнения буфера с потерей информации.

Значения размеров входного и выходного буфера определяются свойствами InputBufferSize и OutputBufferSize соответственно. Установить размер буферов можно командой set:

Set(s,' InputBufferSize', 1024);
Set(s,' OutputBufferSize', 1024);

По умолчанию значение размеров буферов равно 512. Изменять размер буферов может понадобиться при передаче большого объема данных или при непрерывной передаче данных.

Ввод аналоговых данных в MATLAB

Микроконтроллер ATmega 32P содержит десяти-разрядный АЦП с 8 каналами, который может использоваться для измерения аналоговых сигналов и передачи их значений через последовательный порт в MATLAB. В языке программирования Arduino содержится простая в использовании функция аналогового ввода analogRead(pin). Эта функция считывает преобразованный АЦП сигнал с заданного вывода pin микроконтроллера. При частоте синхронизации микроконтроллера 16 МГц время преобразования составляет 100 мкс что соответствует частоте 10 кГц. Аналоговые сигналы измеряются, как правило, с постоянным интервалом времени (шагом).

Реализовать процесс измерения с заданным шагом можно разными способами, в том числе и теми средствами, которые предлагаются в программном обеспечении Arduino. Это программы millis(), micros(), delay(), delayMicroseconds(). В этих программах задействован таймер/счетчик 0 (Timer/Counter0), который исполняет роль системных часов.

Для ввода аналоговых сигналов необходимо проводить измерение сигналов с постоянным шагом с наименьшими затратами времени на организацию запуска АЦП. Стандартные средства Arduino не позволяют сделать это должным образом. Поэтому, целесообразно разработать специальную программу работы АЦП, минимизирующую «накладные» расходы, связанные с организацией процесса измерения.

Для построения временной диаграммы измерения аналогового сигнала используется таймер/счетчик1 (Timer/Counter1), который запускает АЦП на измерение. АЦП работает в режиме автозапуска по сигналу от таймера/счетчика 1. Сигнал запуска АЦП от таймера вырабатывается, когда код счетчика совпадет с кодом в регистре сравнения OCR1B.

В этом регистре хранится код интервала времени запуска АЦП. После запуска АЦП устанавливает в единицу бит занятости ADSC (ADC Start Conversion). По окончании измерения флаг устанавливается в ноль и происходит ввод кода измеренного напряжения.

Определение момента окончания преобразования АЦП можно осуществлять двумя способами: путем сканирования состояние бита ADSC или с помощью прерываний. Какой использовать способ зависит от конкретной задачи.

Разработка и исследование алгоритмов ввода аналоговых сигналов в MATLAB (рис. 1) осуществлялась на макетной плате Arduino – nano. Для повышения входного сопротивления измерительной схемы и согласования уровней входного сигнала с динамическим диапазоном АЦП разработан усилитель АЦП. В качестве источника входного аналогового тестового сигнала использовался генератор синусоидального сигнала с частотой 100Гц, построенный по схеме Вина.

Пример простой тестовой программы ввода аналоговых данных в MATLAB от Arduino по последовательному каналу представлен в табл. 3. Программа оформлена в виде М-файла и работает совместно с программой Arduino показанной в табл.4.

В программе MATLAB задаются параметры частоты квантования сигнала 4000 Гц и количество точек измерения сигнала.

Таблица 3 – Тестовая программа MATLAB

```
% тестовая прогамма ввода аналоговых сигналов
% в MATLAB  Программа работает совместно с
% matlab_arduino_ADC_Test.ino
Fs=4000;  %  частота квантования в Гц
n_point=256;  % количество точек измерения
s=serial('COM4');
set (s,'BaudRate',115200);
fopen(s);
pause(2);
fprintf(s,'!');
out1=fread(s,n_point,'int16');
d = out1*4.73/1023;
%построение графика
Ts=1/Fs;
t=0:Ts:Ts*(n_point-1);
plot(t,d(1:n_point));
title('Input Signal');
pause
% Преобразование Фурье
Y=fft(d,Nf);
f=((0:Nf/2)/Nf)*Fs;
stem(f,abs(Y(1:(Nf/2)+1)),'.')
title(' БПФ','FontName', 'Arial Unicode MS')
zoom on;
fclose(s);
delete(s);
clear s;
```

Далее создается объект COM – порта с указанием номера порта (виртуального). Важно, чтобы номера портов в MATLAB и Arduino совпадали и были одинаково настроены. В нашем случае они соответствуют стандартным настройкам, принятых по умолчанию, за исключением скорости передачи (BaudRate) – 115200.

Открываем порт функцией fopen(s) и делаем необходимую паузу 2с. Затем MATLAB посылает запрос Arduino на измерение сигнала путем посылки строки с командой '!'.

Далее вводим n_point двоичных данных в формате INT16, преобразуем в формат с плавающей запятой и выводим график измеренного сигнала (рис. 3). После ввода сигнала можно выполнить обработку сигнала, например, с помощью дискретного преобразования Фурье (рис. 4) [11].

Программа Arduino, принимает строку с командой ‘!’, распознает ее и запускает процесс измерения и ввода 256 значений аналоговых данных в буферный регистр. После окончания приема данных они передаются по последовательному каналу в MATLAB. Таким образом, происходит сначала измерение 256 точек, а потом передача в MATLAB для дальнейшего анализа.

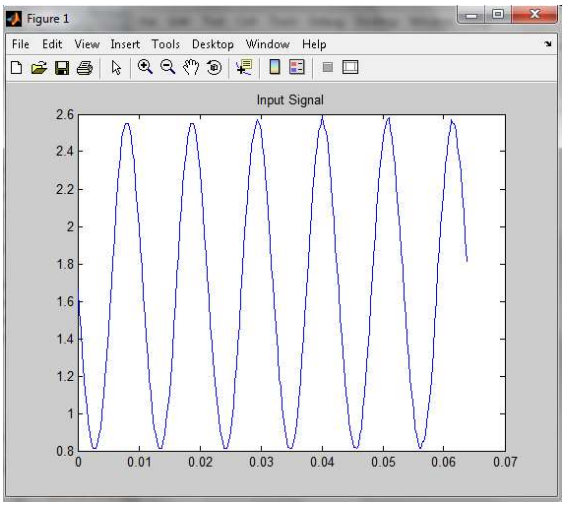

Рисунок 3 –Измеренный сигнал

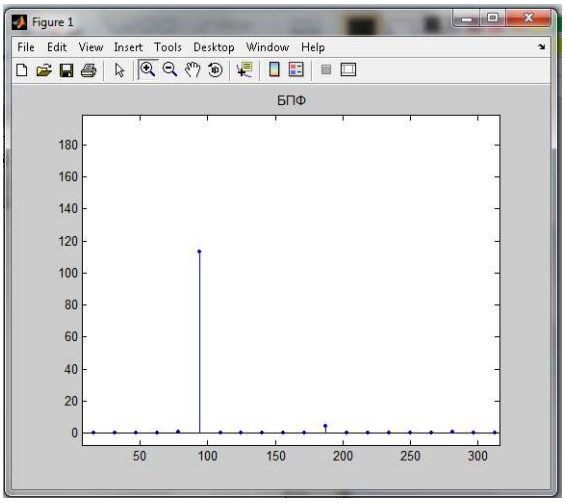

Рисунок 4 – ДПФ измеренного сигнала

Таблица 4 – Тестовая программа Arduino

```
// тестовая прогамма ввода аналоговых сигналов
// в MATLAB  Программа работает совместно с
// matlab_arduino_ADC_Test.m.
#include <LiquidCrystal.h>
#include "avr/io.h"
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
String string2;
unsigned char buf[520];
volatile unsigned int j;
unsigned int i, n_byte;
unsigned int n_point;  // количество измерений
unsigned long int fclk = 16000000;
//параметры теста
unsigned int p0=1;  //Гц-0,кГц - 1
unsigned int p1=4;  // Fs
unsigned int p2=256; //кол.точек изм.
//*****************************************
void setup() {
  Serial.begin(115200);  // открыть последовательный порт
  lcd.begin(16, 2);  // установка LCD
  lcd.print("Matlab ADCint4");
  i = 0;
```

```
  j=0;
  n_point=p2;
  n_byte=n_point*2;
    // Инициализация АЦП
  cli();
  ADMUX = (1<< REFS0);    //VREF = AVCC,Chanel 0
  ADCSRA  = (1<<ADEN)|(1<<ADATE)|(1<<ADPS1);
  ADCSRB    = (1<<ADTS2)|(1<<ADTS0);    //режим
запуска АЦП
  // Инициализация Таймера 1
  OCR1A = 0;
  TCNT1=0;  // Сброс таймера
  TCCR1A = (1<< COM1B0); // Переключение OC1B
  TCCR1B = (1<< WGM12); // Mode CTC
    // Расчет кода регистра сравнения с таймером
  if(p0==0){ OCR1B = fclk/p1-78;}
  else { OCR1B = 16000/p1-78;}
  TCNT1=0; // сброс таймера1
  TIFR1 |=(1 << OCF1B);// сброс флага прерывания
OCF1B
  ADCSRA |= (1<<ADIE);  // разрешение прерывания
АЦП
  sei();  // разрешение глобальных  прерываний
}
void loop() {

   if (Serial.available() > 0) //если буфер не пуст, то
   {
    string2 = Serial.readStringUntil('\n');
        if(string2 == "!")
     {
       j=0;
       TCCR1B |= (1<<CS10);  //  Пуск таймера 1
    // Ожидание измерения n_point точек
       while(j <= (n_byte)) {};
    // Останов Таймера 1
  TCCR1B &= ~(1<<CS12)&~(1<<CS11)&~(1<<CS10);
    // передача n_byte в последовательный порт
       Serial.write(buf,n_byte);
     }
   }
}

// Программа обработки прерывания от АЦП
ISR (ADC_vect)
{
    TCNT1=0; // сброс таймера1
    TIFR1 |=(1 << OCF1B); // сброс флага прерывания
OCF1B
    buf[j] = ADCL; buf[j+1] = ADCH;
    j=j+2;
 }
```

Обеспечение синхронного запуска АЦП осуществляется с помощью таймера 1 (таймер 0 занят для системных нужд Arduino). Запуск АЦП происходит при совпадении кода таймера с кодом, записанным в регистр сравнения таймера OCR1B. При этом АЦП должен работать в соответствующем режиме автозапуска.

После завершения измерения АЦП происходит прерывание, устанавливается флаг прерывания ADINF и управление передается и программе обработки прерывания ISR(ADC_vect). В этой программе осуществляется ввод данных с АЦП и запись их в буферный регистр buf и наращивается счетчик количества проведенных измерений j. Предварительно надо сбросить таймер 1 в ноль так, как он автоматически не сбрасывается в этом режиме. Также необходимо сбросить флаг прерывания таймера OCF1B, хотя прерывания по таймеру запрещены, но установившись в единицу при совпадении кодов таймера и регистра сравнения OCR1B, не даст запустить АЦП в следующем цикле.

Время, затрачиваемое на обращение к программе обработки прерывания порядка 7 мкс. Это время задержки включения таймера на новый цикл измерения. Его можно учесть в коде регистра сравнения OCR1B в виде поправки.

Временная диаграмма работы АЦП с прерываниями показана на рис. 5.

На временной диаграмме счет таймера 1 показан наклонной прямой. Код таймера увеличивается до тех пор, пока не станет равным коду в регистре OCR1B, определяя тем самым период квантования сигнала $T_{кв}$. Сигналом от таймера запускается АЦП и по окончании измерения происходит прерывание и управление передается на программу обработки прерывания, которая читает измеренный код и наращивает счетчик количества измеренных точек. По достижении счетчиком заданного количества измерений останавливается таймер и данные передаются в MATLAB.

В устройстве связи на Arduino используется жидкокристаллический индикатор LCD 1620, на который выводится информация о параметрах ввода сигнала и состоянии прибора.

Для работы с последовательным портом используются библиотечные функции Arduino:

- Serial.begin(115200) - открыть порт;
- Serial.available() – проверка буфера;
- Serial.readStringUntil() – чтение строки;
- Serial.write() – запись двоичных (бинарных) данных.

Все они должны быть согласованы по форматам передаваемых данных с программами MATLAB.

Использование графических пользовательских интерфейсов GUI

Рассмотренный подход в цифровой обработке реальных сигналов обладает большой гибкостью в применении средств MATLAB. Для каждого конкретного исследуемого объекта можно разработать отдельный М-фай, который включает передачу ряда параметров в устройство связи на Arduino: частоту квантования, число измеряемых точек, режимы измерения данных и др. Программа MATLAB должна содержать модули цифровой обработки сигналов, реализующие алгоритмы

ДПФ, различные виды фильтрации (БИХ и КИХ) с возможностью их синтеза и др [12]. Управление может осуществляться с помощью флагов. Установка соответствующего флага в 1 означает наличие опции, 0 – отсутствие. Таким образом, можно подобрать оптимальные алгоритмы обработки сигнала. Этот подход в исследовании сигналов предусматривает изменение в программе и требует определенных усилий в программировании и затрат времени. Для упрощения и наглядности работы с реальными объектами целесообразно использовать графические интерфейсы пользователей (GUI), для разработки которых используется среда разработки графических пользовательских интерфейсов MATLAB GUIDE.

GUIDE предоставляет набор инструментов, значительно упрощающий процесс проектирования и создания GUI. Используя редактор макетов GUIDE легко компоновать GUI, щелкая и перетаскивая компоненты GUI - такие как панели, кнопки, текстовые поля, ползунки, меню и так далее - в область макета. Программирование графического интерфейса пользователя. GUIDE автоматически генерирует M-файл, который управляет работой графического интерфейса пользователя.

На рис. 6 представлен GUI, построенный на основе рассмотренного теста, Он содержит органы управления в виде кнопок, меню, ползунков, текстовых полей, поясняющих назначение органов управления.

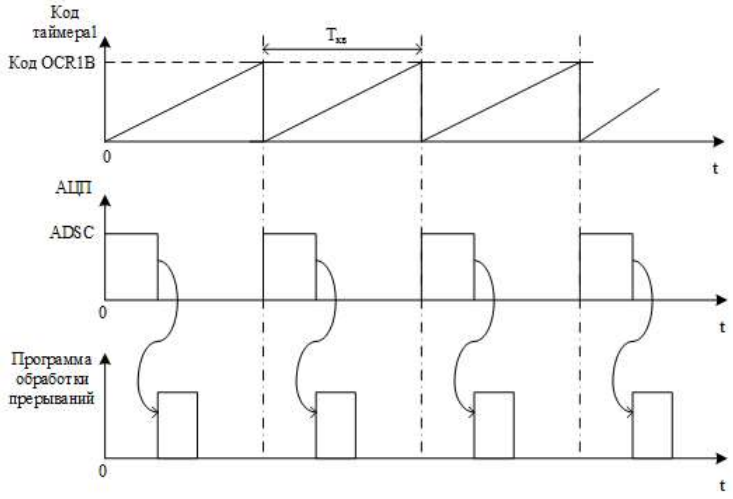

Рисунок 5 – Временная диаграмма работы АЦП по прерываниям

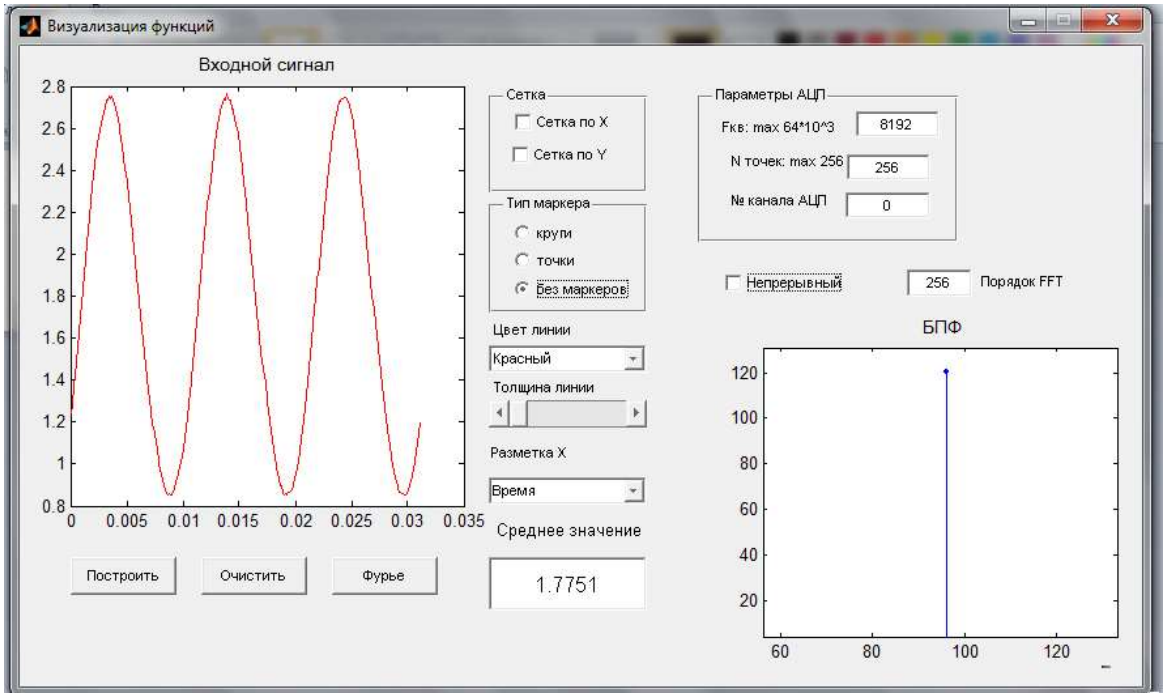

Рисунок 6 – Пример графического интерфейса пользователя (GUI) системы ввода аналоговых сигналов.

Так с помощью окон «Параметры АЦП» задаются параметры работы измерительного устройства: шаг квантования сигнала, количество точек измерения, номер канала АЦП. Также можно задать порядок ДПФ. Для лучшего восприятия графиков измеренных сигналов можно в графическом окне показать сетку по осям X и Y, графики выводить с различными маркерами. С помощью ниспадающего меню «Цвет линии» можно выбрать цвет линий графиков. Толщина линии графиков плавно изменяется с помощью ползунка. Разметка по оси X может быть выполнена по времени или выборкам сигнала.

Запуск системы на измерение сигнала осуществляется по нажатию кнопки «Построить». Параметры измерения сигнала передаются в измерительное устройство на Arduino, происходит его настройка и запускается процесс определения значений заданного количества точек. После окончания измерения массив данных передается в MATLAB и выводится в окно графика сигнала.

ДПФ сигнала можно посмотреть, нажав на кнопку «ДПФ». Очистка окон графиков осуществляется с помощью соответствующей кнопки.

В рассмотренном GUI для обработки сигнала использовался только один алгоритм ДПФ, однако не трудно реализовать и любые другие алгоритмы и выбирать нужные из списка.

В статье не рассмотрены вопросы ввода, отображения, и обработки входных данных в реальном времени в темпе их поступления. Такой режим работы системы требует дополнительных исследований.

В системе введен дополнительный режим работы «Непрерывный», позволяющий постоянно выводить графики сигнала на экран в темпе поступления заданного числа точек измерения.

Выводы

Рассмотренные способы ввода аналоговых сигналов в MATLAB позволяет строить системы анализа и обработки реальных сигналов, используя все вычислительные возможности системы. Использование в системе реального сигнала, а не его модели, позволяет повысить достоверность исследования и сокращает время проектирования систем ЦОС.

Устройство можно использовать в качестве осциллографа и регистратора аналоговых сигналов.

Особенно представляет интерес использование системы в учебном процессе в курсах Микроконтроллеры, Компьютерная электроника, Цифровая обработка сигналов, в постановках междисциплинарных курсовых проектах.

Целью дальнейших исследований может быть анализ временных соотношений при различных способах организации измерения и передачи данных по последовательному каналу. Особый интерес представляет потоковый ввод данных, и их отображение в графическом окне в реальном времени.

Литература

1. Ануфриев, И. Е. MATLAB 7 / И. Е. Ануфриев, А. Б. Смирнов, Е.Н. Смирнова. - СПб.: БХВ - Петербург, 2005. - 1104 с: ил.

2. Краснокутский, В. А. Исследование алгоритмов частотной фильтрации сигналов системы контроля концентрации метана / В. А. Краснокутский, М. В. Выростков // Научные труды Донецкого национального технического университета. Серия "Информатика, кибернетика и вычислительная техника" (ИКВТ-2007). – Донецк: ДонНТУ, 2007. – Вып. 8 (120). - С. 160-168.

3. Краснокутский, В. А. Исследование алгоритмов цифровой фильтрации системы контроля состояния рудничной атмосферы / В. А. Краснокутский, О. В. Гомозов // Наукові праці Донецького національного технічного університету.Серія "Інформатика, кібернетика та обчислювальна техніка". - Донецьк: ДонНТУ, 2008.- Вип. 9 (132). – С. 152–156.

4. Краснокутский, В. А. Цифровая частотная фильтрация сигналов системы контроля концентрации метана / В. А. Краснокутский, М. В. Выростков // Практика и перспективы развития партнерства в сфере висшей школы. Материалы восьмого научно-практического семинара. – Донецк: ДонНТУ, 2007. – Том 3. – С. 148–154.

5. MATLAB® The Language of Technical Computing External Interfaces Version 7 Available at:https://www.mn.uio.no/astro/english/services/it/help/mathematics/matlab/apiext.pdf.

6. Петин, В. А. Практическая энцикло-педия Arduino / В.А. Петин, А. А. Биняковский. - М.: ДМК Пресс, 2017. - 152 с.

7. 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash/ Available at: https://www.microchip.com/content/dam/mchp/documents/MCU08/ProductDocuments/DataSheets/Atmel-7810-Automotive-Microcontrollers-ATmega328PDatasheet.pdf.

8. Прокопенко, В. С. Программирование микроконтроллеров ATMEL на языке С. – К. «МК-Пресс», СПб.: «КОРОНА_ВЕК», 2012. – 320 с., ил.

9. Агуров, П. В. Интерфейсы USB. Практика использования и программирования.- СПб.: БХВ-Петербург, 2006. – 576 с.: ил

10. FT232R USB UART IC Datasheet Version 2.16. Available at: https://ftdichip.com/wp-content/uploads/2020/08/DS_FT232R.pdf.

11. Смит, Стивен. Цифровая обработка сигналов. Практическое руководство для инженеров и научных работников / Стивен Смит; пер. с англ. А. Ю. Линовича. - М.: Додэка-XXI, 2012. - 720 с.

Цифровая обработка сигналов / А. Б. Сергиенко. - СПб.: Питер, 2002. - 608 с: ил.

Краснокутский В.А., Достлев Ю. С. Исследование способов передачи информации в системах сбора и обработки данных в MATLAB. *Статья посвящена исследованию применения пакета MATLAB для обработки и анализа данных, поступающих от внешних устройств в реальном времени. Рассмотрены вопросы организации передачи данных через интерфейс последовательного порта. Приведены практические схемы и алгоритмы передачи аналоговых данных для обработки их средствами пакета MATLAB. В качестве устройства связи между внешним устройством и пакетом MATLAB используется платформа Arduino на микроконтроллере ATmega32. Показаны эффективные возможности проектирования систем цифровой обработки данных в интерактивном режиме с использованием простых систем сбора данных в реальном времени.*

Ключевые слова: *МАТЛАБ, аналоговый сигнал, передача данных, Ардуино, устройство ввода аналоговых сигналов, последовательный порт, графический интерфейс пользователя.*

Krasnokutskiy V., Dostlev Yu. A study of methods of transferring information in data collection and processing systems in MATLAB. *This article explores the application of MATLAB package for processing and analyzing data from external devices in real time. Issues of organizing data transfer through the serial port interface are considered. Practical schemes and algorithms of analog data transfer for their processing using MATLAB package are given. Arduino platform on ATmega32 microcontroller is used as a communication device between an external device and MATLAB package. Efficient capabilities of designing digital data processing systems in interactive mode using simple real-time data acquisition systems are shown.*

Keywords*: MATLAB, analog signal, data transmission, Arduino, analog input device, serial port, graphical user interface, GUI*

Статья поступила в редакцию 08.12.2022
Рекомендуется к публикации профессором Зори С. А.