S E S 2 0 1 4
Tenth Anniversary Scientific Conference with International Participation
SPACE, ECOLOGY, SAFETY
12 – 14 November 2014, Sofia, Bulgaria

УСТРОЙСТВО ЗА СЪБИРАНЕ НА ДАННИ НА БАЗАТА НА МИКРОКОНТРОЛЕР PIC18F2550

Константин Методиев

Институт за космически изследвания и технологии – Българска академия на науките
e-mail: komet@space.bas.bg

Ключови думи: *микроконтролер PIC18F2550, протокол I^2C, WinAPI, mikroC, MS Visual Studio*

Резюме: *В настоящия доклад е разгледан прототип на устройство за събиране на данни на база микроконтролер PIC18F2550. Прототипът е в минимална примерна конфигурация и включва сензор, цитирания микроконтролер и персонален компютър, чрез който се събират данните. Устройството се свързва с компютъра през сериен порт и се разпознава като Human Interface Device (HID). Разгледани са принципната схема, както и разработеното програмно обезпечаване. Софтуерът за микроконтролера е разработен в среда MikroC for PIC (фирма Mikroelektronika, Сърбия), а този за компютъра – в среда Visual Studio (фирма Microsoft, САЩ). Проведени са експерименти, целящи да демонстрират част от възможностите на микроконтролера за провеждане на изследователска работа.*

A DATA lOGGING DEVICE BASED UPON THE PIC18F2550 MCU

Konstantin Metodiev

Space Research and Technology Institute – Bulgarian Academy of Sciences
e-mail: komet@space.bas.bg

Key words: *PIC18F2550 MCU, protocol I^2C, WinAPI, mikroC, MS Visual Studio*

Abstract: *In the paper hereby, prototype of a data logging device based upon the PIC18F2550 microcontroller unit (MCU) is discussed. The device has a minimum exemplary configuration including a sensor, the MCU mentioned and a PC by means of which the data are being collected. The device is hooked up into the PC through a serial port. It is identified as a Human Interface Device (HID). What is considered further is both the principal circuit and the relevant software. The software meant to the MCU is developed within mikroC environment (Mikroelektronika, Serbia) whilst the software intended to log data is developed by means of Visual Studio (Microsoft, USA). Several experiments have been carried out in order to verify whether the MCU is capable of carrying out scientific experiments.*

1. Теоретична постановка

За разлика от шина RS-232, за която форматът на приемане и предаване на данните не е определен точно, USB е host контролирана шина със собствен протокол. Host устройството управлява комуникацията по шината. Въз основа на това всяка USB транзакция се разделя на следните три елемента: пакет Token, пакет Data и пакет Handshake, [1]. Двоичният еквивалент на всеки пакет е 4-битов, като той се допълва с още 4 бита отляво (инвертирана стойност, т.е. ~(PID)), за да се коригират евентуални грешки по време на транзакцията.

Данните, съставени от пакети, се предават по USB шината по четири начина: обемен (bulk), прекъсване (interrupt), на равни интервали (isochronous) и контролен (control) трансфер. Прекъсването се използва за предаване на малки количества данни. Целта е данните да се обменят по най-бързия възможен начин. Терминът „прекъсване“ в случая няма нищо общо с хардуерното прекъсване при четене/запис на периферни устройства на PC. Размерът на данните е от 1 до 8 байта за ниска скорост, от 1 до 64 байта за пълна скорост и до 1024 байта

за висока скорост на обмен. В настоящото изследване се използва този начин за трансфер на данни на пълна скорост между компютъра и микроконтролера.

Енумерацията е процес на детекция, идентификация и зареждане на драйвери за USB устройство, [2]. След като периферно устройство се включи към host по USB, се изпълнява следната процедура:

- Детекция на устройството. Осъществява се чрез обмен на данни по линии D+ и D–.
- Определяне на скоростта на обмен на данни. Съществуват три режима на работа: ниска скорост (1.5 Mbps), при който се използва 1k5 съгласуващ резистор към високо ниво към линия D–; пълна скорост (12 Mbps), при който се използва 1k5 съгласуващ резистор към високо ниво към линия D+; висока скорост (480 Mbps), при който се използва 1k5 съгласуващ резистор към високо ниво към линия D+. За тази скорост се дефинират две различни състояния J и K в зависимост от напрежението U между D+ и D– респ. U >= +300mV и U <= –300 mV.
- Прочитане на дескриптора на HID устройството. След като се установи връзката и скоростта на обмен на данни, host ще се опита да прочете дескрипторния файл с използване на адрес по подразбиране. Ако операцията е успешна, следва reset на устройството и задаване на уникален адрес. В този дескрипторен файл се намират променливите Vendor ID и Product ID.
- Прочитане на конфигуриращия дескриптор. Този дескриптор има фиксирана дължина от 9 байта. Той дава специфична информация като брой на поддържаните интерфейси и максималната мощност, която HID устройството се очаква да консумира.
- Прочитане на интерфейсния дескриптор. Този дескриптор също има фиксирана дължина от 9 байта. Той дава информация за броя на използваните крайни точки.
- Зареждане на драйвери. Когато HID устройството е напълно идентифицирано, host зарежда драйвери за управлението му. Съчетаването на определено устройство и драйвер зависи от стойностите на променливите Vendor ID и Product ID. След инсталация на драйверите, настройките се записват в регистрите, така че при повторно включване на устройството, драйверът се зарежда автоматично.

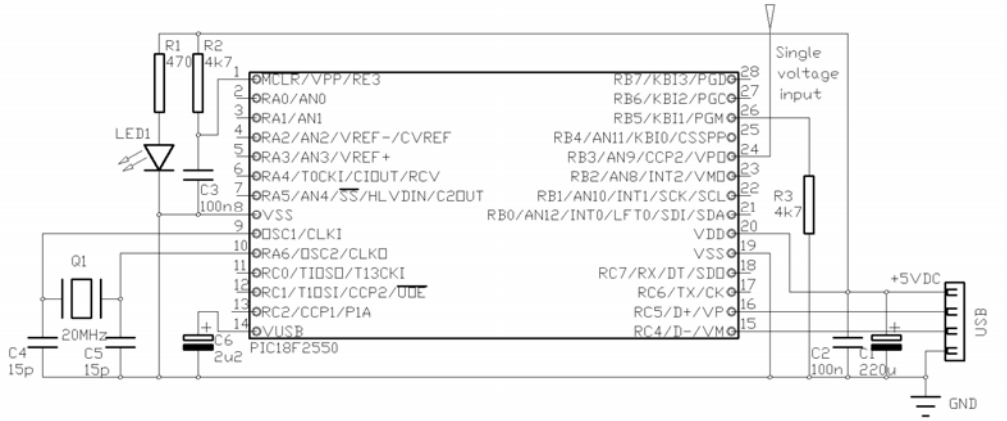
Функциите на Windows API са разположени в различни dll и lib файлове, документацията на които се намира в Windows DDK и Platform SDK. Всяка библиотека за динамично свързване (dll разширение) се съпровожда от един или няколко библиотечни и заглавни файла (респ. lib и h разширение). Библиотечните файлове премахват необходимостта приложението да използва указател към функция от dll файла. Заглавният файл съдържа прототипи на функции, структури и константи, които приложението може да използва непосредствено.

Съществуват следните видове API, посредством които се извършва комуникация между периферно устройство и host, [3]. HID API е сложен интерфейс, чрез който дадена програма има достъп до всяко периферно устройство, както като вход, така и като изход, с изключение на мишка и клавиатура, които разработчиците на Microsoft умишлено са блокирали. DirectInput API осигурява относително разбираем интерфейс за сметка на гъвкавостта. Поддържат се ограничен брой USB HID устройства, например такива за игри. RawInput е нов API и е въведен с Windows XP. Създаден е в отговор на желанието на разработчиците да получат директен достъп до клавиатура и мишка.

2. Материали и методи

Принципната схема на свързване на микроконтролер PIC18F2550 към USB порт е показана на фиг. 1. Кондензатори C3 и C6 са важни с оглед осигуряване стабилността на връзката по USB. Пропускането на тези кондензатори в схемата или само промяна в капацитетите водят до периодично включване и изключване на микроконтролера. Съгласуващ резистор към ниско ниво R3 не позволява на пин 26 (PGM) да активира режим на програмиране ISP когато бит LVP = 1. Резисторът е полезен когато в микроконтролера е зареден bootloader. В този случай пин 25 се свързва към земя.

В разработената програма под Win32 се използват функциите на HID API за комуникация с микроконтролер PIC18F2550. Използван е компилатор Miscosoft Visual Studio 2013. Кодът за програмата в среда OS Windows е адаптиран от скриптове на Jan Axelson, [4], под условията на споразумение GNU General Public License. Подробно описание на използваните функции може да се намери в [5].

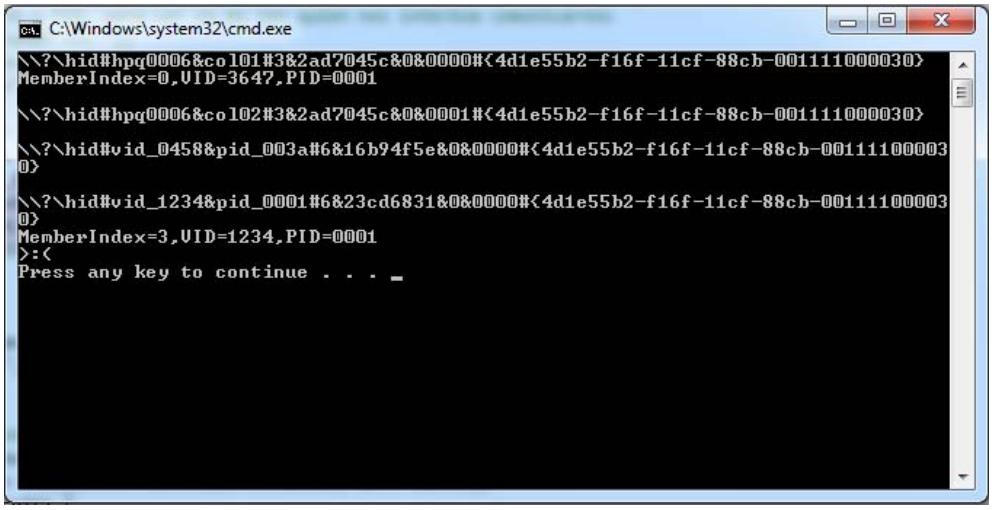


Фиг. 1. Свързване на микроконтролера към USB интерфейс

Алгоритъмът за откриване и комуникация е следния. Функция SetupDiGetClassDevs дава информация за всички устройства от даден клас (в случая HID), които текущо са включени в host устройството, [6]. Първият параметър е т. нар. GUID (Globally Unique Identifier), представляващ уникален за устройството референтен номер. Обикновено GUID се изобразяват като 32-битов шестнадесетичен код по групи, разделени с тирета: {4d1e55b2-f16f-11cf-88cb-001111000030}, фиг. 2. Стойността на параметъра се получава след обръщане към функция HidD_GetHidGuid в библиотека hid.dll. Тази функция резервира блок в оперативната памет, съдържащ масив с данни за всяко включено устройство. Масивите трябва да бъдат подредени в списък (enumeration), за което се използва функция SetupDiEnumDeviceInterfaces в библиотека setupapi.dll. Обръщението към тази функция попълва структура DeviceInterfaceData с детайли за всяко устройство в списъка и връща указател към нея.

Следващата стъпка е обръщане към функция SetupDiGetDeviceInterfaceDetail. Информацията, получавана от тази функция, съдържа път към устройството, който в последствие се използва като аргумент във функция CreateFile. Пътят, освен всичко друго, съдържа променливите VID (Vendor ID) и PID (Product ID), които в последствие се използват за идентификация на дадено устройство.

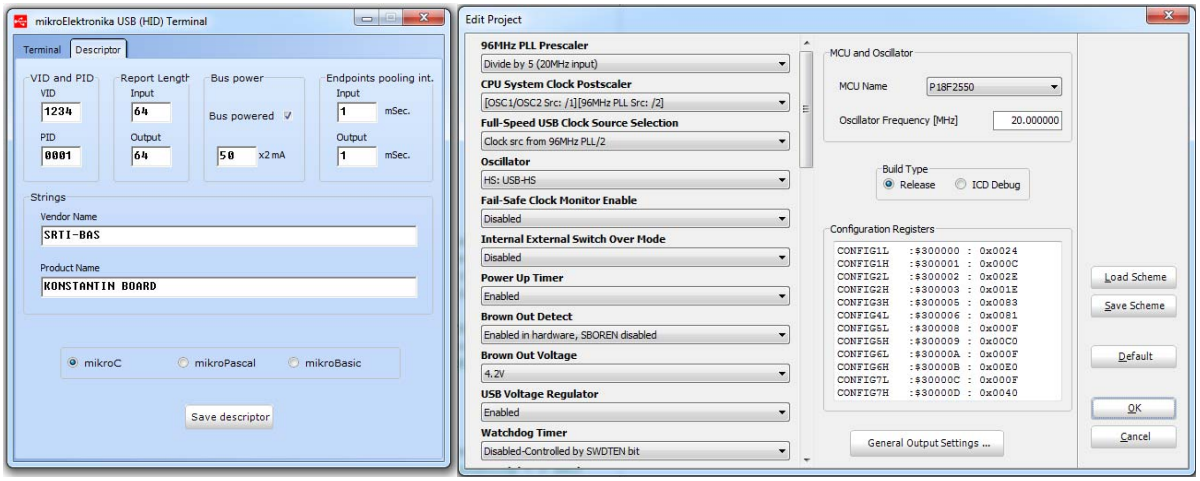
На фиг. 2 са показани стойностите на променливите VID = 1234 и PID = 0001. Те се виждат след спецификатора hid. Пътят се съхранява в структура Psp_Device_Interface_Detail_Data. Последната стъпка е създаване на handle (unsigned long) за четене и писане чрез обръщане към функция CreateFile поотделно. Първият аргумент на функцията е именно полето Psp_Device_Interface_Detail_Data->DevicePath = 23cd6831.



Фиг. 2. Примерна идентификация на включени HID устройства

Създаването на дескрипторен файл в среда MikroC for PIC се осъществява с HID Terminal, който се предоставя с компилатора, фиг. 3. В полета Input се въвеждат дължините на входния и изходен масив символни данни. Тази дължина трябва да съвпада с въведената големина на входния и изходен символни масиви в програмата за микроконтролера.

Максималната дължина е 64 байта. Дескрипторният файл се включва към проекта и се компилира заедно с основния сорс – код, [7]. Настройките на проекта са показани на фиг. 3. Точното въвеждане на посочените стойности на полетата е важно, в противен случай енумерацията е неуспешна.

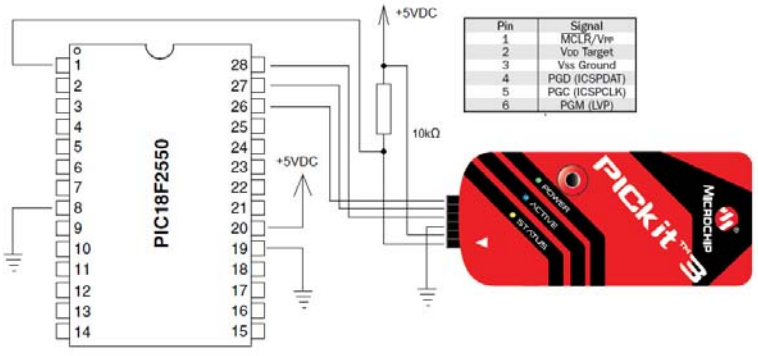


Фиг. 3. Диалог за създаване на дескрипторен файл (ляво) и настройки на проекта в среда MikroC

В микроконтролера е вградена схема за конвертиране на честота от 4 MHz до 96 MHz. Нарича се Phase Locked Loop – PLL. Тази честота се дели на 2, за да се получи работната тактова честота за PIC18F2550, а именно 48 MHz. Микроконтролерът предлага две възможни стойности за тактова честота: 6 MHz (ниска скорост) и 48 MHz (висока скорост). Високата скорост е ошумена и скъпа, което мотивира Microchip да проектира схема PLL. В този случай в падащо меню CPU system Clock Postscaler се избира опция System Clock Postscaler Selection >> 96MHz PLL Src: /2.

Честотата на свързания кристален осцилатор е 20 MHz. Тъй като схема PLL получава на входа 4 MHz, то в падащо меню 96 MHz PLL prescaler се избира опция Divide by 5 (20 MHz input). Честотата на осцилатора се въвежда допълнително в поле Oscillator Frequency [MHz]: 20.000000. Видът на осцилатора High Speed (HS) се избира от меню Oscillator. Накрая задължително се включва USB регулаторът на напрежение от меню USB Voltage Regulator. Регулаторът работи с напрежение 3.3V, вграден е в микроконтролера и е важен за правилната комуникация. Възможно е също това напрежение да се подвежда от външен източник през пин Vusb. Ако се избере вграденият регулатор обаче, пин Vusb задължително се свързва към маса чрез кондензатор C6 на фиг. 1. Самият кондензатор е необходим за регулатора, но капацитетът му е висок и затова не е целесъобразно да бъде вграден в чипа.

Програмирането на микроконтролер PIC18F2550 се извършва с помощта на софтуерен пакет MPLAB X и програматор PICKit3 на фирмата Microchip. В средата на MPLAB X се създава проект Prebuilt (Hex, Loadable Image) Project. В следващите стъпки на диалога се указва вида на устройството и програматора, а така също се посочва пътя до hex файла. Последният е получен в среда mikroC и представлява компилираната програма. Схемата на свързване на програматора към микроконтролера е показана на фиг. 4.



Фиг. 4. Свързване на програматор PICKIT3 и микроконтролер PIC18F2550

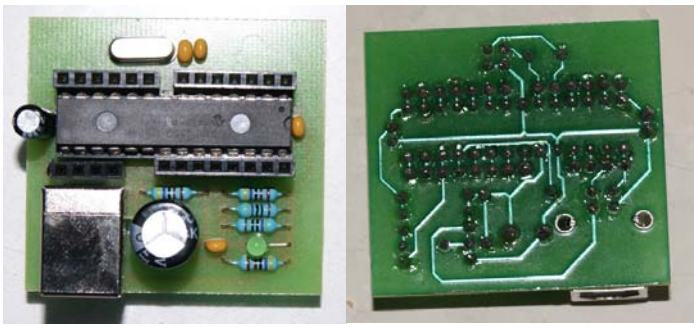
Общият вид на използвания хардуер е показан на фиг. 4, 5 и 6. Микроконтролер PIC18F2550, фиг. 4, е в корпус sDIP-28. Сензор Honeywell 40PC006G2A, фиг. 5, е температурно компенсиран преобразувател (transducer) на статично налягане. Захранва се с +5 VDC, а полезният сигнал е в обхвата от 0 до +4.5 VDC единично напрежение. Обхватът на сензора е от 0 до +6 psi gauge. Сензорът възприема само надналягане, [8].

Сензор TC74 (опция A2-5.0VAT), фиг. 5, е цифров термометър с обхват от –40 до +125°C. Захранва се от +5 VDC. Изходният сигнал на сензора е в съответствие с протокол I^2C. Адресът за писане в режим Slave е 0b10010100 (0x94), а този за четене – 0b10010101 (0x95), [9].

Сензор DS1307, фиг. 5, е часовник за реално време, измерващ час и дата. Захранва се от +3.3 VDC. Изходният сигнал на сензора е в съответствие с протокол I^2C. Адресът за писане в режим Slave е 0b11010000 (0xD0), а този за четене – 0b11010001 (0xD1), [10]. Сензорите TC74A2-5.0VAT и DS1307 са свързани последователно към пинове SDA и SCL. Използват се по един съгласуващ резистор към високо ниво от 1k към всяка шина.

На фиг. 6 е показана развойната платка за микроконтролер PIC18F2550. Практическата схема е генерирана в среда ISIS Proteus от принципната схема, показана на фиг. 1. Всички пинове на микроконтролера са изведени на гнездови рейки с изключение на захранване и маса. Към схемата е запоен кристален осцилатор с честота 20 MHz, като капацитетите на стабилизиращите кондензатори C4 и C5 са по 15 pF, [11].

Фиг. 5. Общ вид на преобразувател 40PC006G2A, термометър TC74A2–5.0VAT и часовник DS1307



Фиг. 6. Страна елементи (ляво) и страна спойки на устройството за събиране на данни

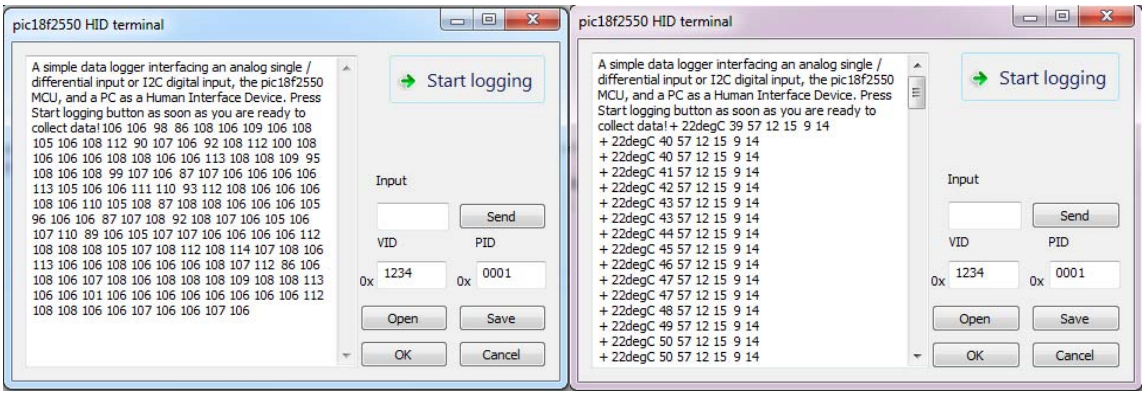
3. Резултати

За проверка пригодността на устройството за събиране на данни, фиг. 6, бяха проведени измервания с преобразувател 40PC006G2A, свързан с аналогов канал AN9 (пин 24) на микроконтролера. Програмата, записана в чипа, изпраща по USB измерените нива на квантуване (integer) под формата на interrupt transfer. Стойностите на измерения сигнал варират в обхвата 0 – 1023 и се записват автоматично на хард диска, като старите данни се запазват (append).

На фиг. 7 е показан screenshot от програмата за събиране на данни. Интерфейсът е modeless dialog box. В полета VID и PID се въвеждат съответните стойности за свързаното HID устройство. Тези стойности са въведени предварително в дескрипторния файл чрез HID Terminal, фиг. 3 и логично трябва да съвпадат. Програмата предлага възможност за ръчно записване и четене на текстови файлове.

След малки корекции същата програма беше използвана за събиране на данни от цифрови сензори TC74A2–5.0VAT и часовник DS1307, свързани последователно по канали SDA и SCL. Примерен screenshot е показан на фиг. 7, дясно. Интервалът на събиране на данни

се задава програмно в Win32, в случая 30 ms за сензор 40PC006G2A и 750 ms за TC74A2–5.0VAT и DS1307.



Фиг. 7. Интерфейс на програмата за събиране на данни от сензор 40PC006G2A (ляво) по аналогов канал и сензори DS1307 и TC74 по протокол I^2C

4. Дискусия

Наблюдават се малки смущения при измерването на фиксирана стойност на налягането чрез сензор 40PC006G2A и АЦП на микроконтролера. На фиг. 7 са показани нивата на квантуване при отворен към атмосферата щуцер на преобразувателя. Установено бе, че и съседните на входния аналогови канали се влияят от постъпващия измерван сигнал. Това наблюдение, както и малката разрядност на АЦП, са причина авторът да не препоръчва използването на модул A/D за измерване на физични величини с висока точност, както и да не се включват сензори с аналогов изход на съседни канали. При измерване със сензор 40PC006G2A е необходимо и предварително да се извърши калибровка с еталонна стойност на налягането. Формулата за пресмятане на измереното налягане във функция на нивата на квантуване може да се кодира допълнително или в програмата за събиране на данни или в third party софтуер, например MS Excel. Тази формула умишлено не е въведена в микроконтролера. Причината е малката разрядност на чипа (8 бита), което води до неточност в аритметичните изчисления и най-вече в действие деление. Освен това технологичното време между две последователни измервания по такъв начин се съкращава значително.

Предимство на цифров протокол I^2C е възможността за свързване на повече от един сензор по канали SDA и SCL. Програмата се обръща към даден сензор по избор посредством адреса за писане. Калибровка в случая не е необходима, тъй като това е ангажимент на фирмите производители на сензорите. Единственото изискване в случая е еднократното сверяване на часовник DS1307 по програмен път.

Алтернатива на модул A/D на микроконтролера е АЦП ADS1015 на фирмата Adafruit, [12]. Изделието цифрова с разрядност от 12 бита, като големината на полезния сигнал може да се програмира. Предлагат се 4 аналогови входа за единично напрежение или 2 за диференциално. Обръщението към всеки аналогов вход става по програмен път посредством активиране на съответни битове в конфигурационния регистър. Резултатите се предават на микроконтролера по протокол I^2C. Калибровка в случая, очевидно, е необходима.

В Интернет се предлагат безплатно програми, с помощта на които се проследява сесия на трансфер на данни по USB протокол. Повечето от тях извършват четене и/или запис през сериен COM порт. В процеса на разработване на софтуера бяха използвани като помощни два програмни продукта, проследяващи трансфер на данни между PC и HID устройство. Тези продукти са алтернатива на представената в доклада програма за събиране на данни. Така например YAT Yet Another Terminal е проект, разработен от частно лице, който се предлага безплатно в Source Forge. С помощта на програмата се тестват серийни комуникации. Поддържат се протоколи RS-232/422/423/485, а така също и TCP-Client/Server/AutoSocket, UDP и USB Serial/HID, [13]. Програмата USBTrace, предлагана от фирмата SysNucleus, представлява USB анализатор. С нейна помощ се проследява трафика между PC и host контролери, хъбове и устройства, включително и HID. Програмата няма безплатен лиценз, [14].

Основното предимство на предложеното устройство е неговата ниска себестойност, както и простота на принципната схема. С помощта на съвременните програмни среди вече е възможно да се разработи необходимият софтуер, както за микроконтролера, така и за PC, без

да се познава хардуерът в дълбочина. Това позволява по-широк кръг от потребители да подготвят самостоятелно апаратура за провеждане на измервания на физични величини.

Литература:

1. I b r a h i m, D., Advanced PIC Microcontroller Projects in C, From USB to ZIGBEE with the PIC 18F Series, ISBN 978-0-7506-8611-2, 2008, Elsevier Ltd.
2. Simplified Description of USB Device Enumeration, Technical Note TN113, 2009, FTDI Ltd.
3. http://at.or.at/hans/research/nime/hid/apis.html
4. http://www.lvr.com/hidpage.htm
5. A x e l s o n, J., USB Complete: Everything You Need to Develop Custom USB Peripherals, Third Edition, ISBN13 978-1-931448-03-1, ISBN10 1-931448-03-5, 2005, Lakeview Research
6. http://www.developerfusion.com/article/84338/making-usb-c-friendly/
7. K i r o l l o s, M. W., HID USB Communication (Pic18f2550 with PC), Ain Shams University, 2013
8. Pressure Sensors 40PC Series, Miniature Signal Conditioned, Interactive Catalog, Honeywell
9. TC74, Tiny Serial Digital Thermal Sensor, Datasheet, Microchip
10. DS1307 64 x 8 Serial Real-Time Clock, Datasheet, Dallas Semiconductor
11. PIC18F2455/2550/4455/4550 Data Sheet, 2006, Microchip Technology Inc.
12. http://www.adafruit.com/product/1083
13. http://sourceforge.net/projects/y-a-terminal/
14. http://www.sysnucleus.com/