СЪЮЗ НА УЧЕНИТЕ В БЪЛГАРИЯ - ПЛОВДИВ

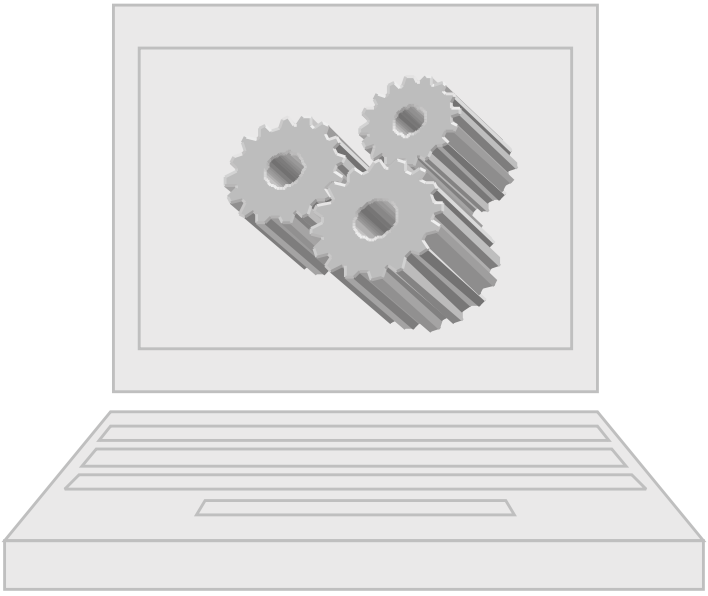
Научни трудове на Съюза на учените Пловдив

Серия В. Техника и технологии,
том. XIX

2021г.
Пловдив

ISSN 1311 - 9419 (Print)
ISSN 2534 - 9384 (Online)

НАУЧНИ ТРУДОВЕ
НА СЪЮЗА НА УЧЕНИТЕ В БЪЛГАРИЯ - ПЛОВДИВ



2021

ПЛОВДИВ

СЪЮЗ НА УЧЕНИТЕ В БЪЛГАРИЯ – ПЛОВДИВ

Научни трудове на Съюза на учените в България – Пловдив

Серия В. Техника и технологии, том XIX

2021

Дом на учените, Пловдив

.....

UNION OF SCIENTISTS IN BULGARIA – PLOVDIV

Scientific Works of the Union of Scientists in Bulgaria - Plovdiv

Series C. Technics and Technologies, Vol. XIX

2021

House of Scientists, Plovdiv

Научни трудове на Съюза на учените в България – Пловдив. Серия В. Техника и технологии. Том XIX, ISSN: 1311-9419 (Print); ISSN 2534-9384 (Online), 2021. Scientific Works of the Union of Scientists in Bulgaria - Plovdiv. Series C. Technics and Technologies. Vol. XIX, ISSN 131 -9419 (Print); ISSN 2534-9384 (Online), 2021.

ПОРТ-МАНИПУЛАЦИЯ НА ATmega328P - ДОМ ЗА ROCKY

Росица Максимова, Красимир Колев
Университет по хранителни технологии – Пловдив

PORT MANIPULATION OF ATmega328P - ROCKY'S HOME

Rositsa Maksimova, Krassimir Kolev
University of Food Technologies – Plovdiv

Abstract: This paper presents techniques for programming of microcontrollers with ATmega328P processor through port manipulation, considering its pros and possible cons. A prototype to a smart electronic system with a compilation of electronic components is proposed. A principal illustration of the system for representing its idea is synthesized. A table is given, describing the components to their respective pins, each with its own role. The source code of the system is provided.

Keywords: Arduino, ATmega328P, microcontrollers, port manipulation.

Въведение

Порт-манипулацията представлява по-бързо манипулиране на входно-изходните пинове на микронконтролера чрез програмиране на ниско ниво на портовите регистри. За нуждите на настоящата публикация е ползвана микроконтролерна развойна платка Arduino Uno Rev3 с ATmega328P AVR микроконтролер. Програмирането чрез така наречената порт-манипулация притежава както своите предимства, така и своеобразни рискове, от които сме се абстрахирали за целите на нашата публикация.

Рискове на порт-манипулацията:

- Програмният код може да представлява затруднение при отстраняване на грешки, поддръжка и разбиране от последващи програмисти;
- Кодът е по-малко преносим. Ако се ползват библиотеки с готови написани функции като digitalRead() и digitalWrite() например, е много по-лесно той да работи на всички микроконтролери от сходна серия, докато при директна манипулация на регистрите може да доведе до грешка поради различни регистри за всеки вид микроконтролер;
- Директният достъп до портовете може да доведе до неволни неизправности. Така например лесно може да се промени серийният пин (RX) от серийния порт чрез невнимателна промяна на бита от 0 към 1 и той да спре да работи.

Както обаче гласят 2 от общо 19-те принципа за програмиране из „кодекса“ на програмния език Python, наречен Zen of Python: “Не оставяйте грешките незабелязани...“, “... освен ако не сте ги скрили умишлено”, ако се предвидят всички заплахи на този вид програмиране и се заобиколят професионално, това би довело до използване на предимствата на порт-манипулацията **(The Zen of Python, 2020)**:

- Изпълнението на кода отнема само няколко микросекунди. Позволява много бързо включване и изключване на пинове в рамките на няколко микросекунди. Кодът, съдържащ се във функциите digitalRead() и digitalWrite() представлява

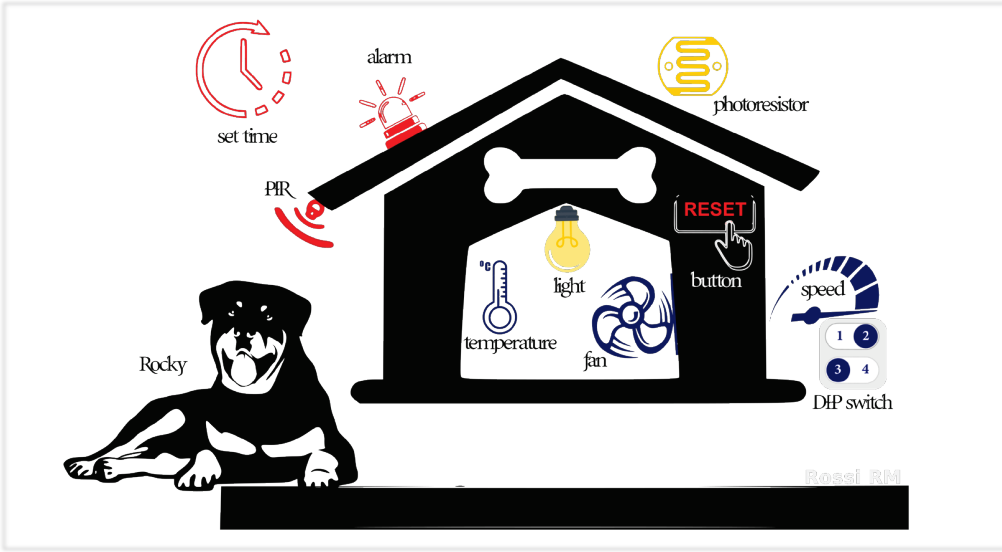
дузина или няколко реда, които се компилират в немалко машинни инструкции. Всяка машинна инструкция изисква един тактов цикъл от 16MHz (за нашия микроконтролер), който може да струва доста време понякога в чувствителни откъм време приложения. Директният достъп до портовете може да свърши същата работа за много по-малко цикли на кварцовия генератор;

- Директният достъп до портовете позволява едновременно задаване на множество изходни пинове, което чрез отделно последователно извикване на функции може да повлияе отново по време изпълнението на дадена програма;
- Програмният код, написан чрез порт-манипулация заема значително по-малко памет, което се явява основен момент при микроконтролерите, тъй като разполагат с ограничена програмна памет. Така компилираният код изисква много по-малко байта, за да се зададат едновременно няколко пина чрез регистрите на портовете, отколкото ако се използва цикъл for, за да се зададе всеки пин поотделно **(Barrett, 2020)**.

ATmega328P е високопроизводителен нискомощен AVR 8-битов микроконтролер, разполагащ с 3 порта (B, C, D), контролирани от три вида регистри (DDR, PORT, PIN) **(Microchip, 2020)**. Той е напълно достатъчен за изпълнение на целите на публикацията, а именно създаване на интелигентна електронна система за обслужване на домашен любимец (в нашия случай куче - Rocky).

Интелигентна електронна система Rocky

За създаването на интелигентна електронна система, обслужваща например домашен любимец е представена идеологична принципна илюстрация на системата на фиг. 1.



Фиг.1. Принципна илюстрация на системата

Идеята за проекта е следната: чрез фоторезистора да се управлява осветление евентуално в помещение, обитавано от домашен любимец. Чрез температурен сензор да се следи температурата в помещението и да се управлява моторче за вентилация (в нашия случай за охлаждане). Скоростта на вентилация се настройва от превключвател с 4 степени. Чрез сензор за движение се следи присъствието на Роки, като при отсъствие повече от зададеното време от стопанина се задейства аларма чрез звънец в случая, която се

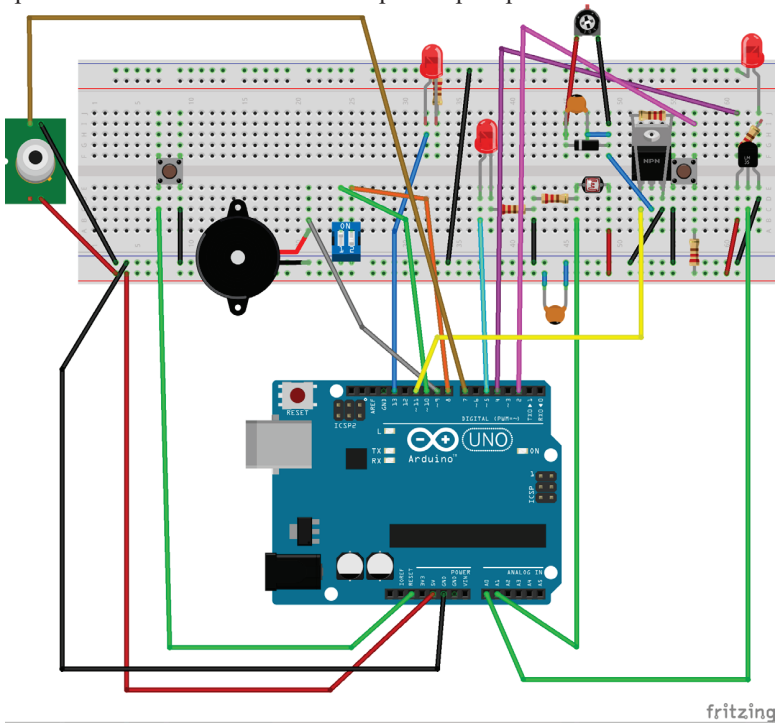
деактивира от засечено движение или чрез бутон за RESET на цялата система на място от стопанина. Съществува и отделен бутон за моментно контролно осветление в помещението.

За нуждите на една такава система са подбрани примерни електронни компоненти, които може да бъдат заменени със свои аналози като марка и модели. В табл. 1 са представени основните компоненти, свързващи се към съответния си пин, както и ролята на всеки пин (Input = Вход, Output = Изход).

Табл. 1. Свързване на компонентите към съответните пинове с определена роля

I/O	ПИН №	КОМПОНЕНТ
Input DDD2	**2**	Бутон №2 (PUSH)
Output DDD4	**4**	Зелен светодиод (LED)
Output DDD5	**5**	Жълт светодиод (LED)
Input DDD7	**7**	Сензор за движение (PIR)
Input DDB0	**8**	Превключвател (DIP Switch)
Output DDB1	**9**	Звънец (Buzzer)
Input DDB2	**10**	Превключвател (DIP Switch)
Output DDB3	**11**	Вентилатор 5V (Fan)
Output DDB5	**13**	Червен светодиод (LED)
Input	**RESET**	Бутон №1 (PUSH)
Input ADMUX0	**A0**	Температурен сензор (TMP36)
Input ADMUX1	**A1**	Фоторезистор

Освен изброените в таблица 1 компоненти, в системата се използват и резистори: 4 бр. 330 Ω, 1 бр. 2.2 kΩ, 1 бр. 1 kΩ, 1 бр. 10 kΩ; диод: 1 бр. 1N4004; транзистор: 1 бр. N-Channel MOSFET и кондензатори: 2 бр. 100 nF. На фиг. 2 е изобразена прототипната принципна схема на свързване на компонентите към микроконтролера Uno.



Фиг.2. Прототипна схема на свързване на системата

Пълният програмен код на системата е предоставен в уебсайта на автора и се намира на адрес https://rossirm.com/portmanipulation.php. Част от кода, базиращ се на принципа на порт-манипулация изглежда по следния начин:

```
…
// Инициализация на АЦП
void initADC() {
  ADMUX &= ~(1<<5); // Зануляване на бита ADLAR = 0, който влияе върху представянето на резултата от АЦП и преобразуване в регистъра за данни (бит ADLAR)
  ADMUX |= 1<<6; // Избор на референтно напрежение = 5V от платката за АЦП (бит REFS0)
  ADMUX &= ~(1<<0); // Избран аналогов канал. Ползване на ADC0, пин A0 (бит MUX0)
  ADCSRA |= (1<<2) | (1<<1) | (1<<0) ; // Задаване на делител 128 между системната тактова честота и собствения такт на АЦП (битове ADPS2:0)
  ADCSRA |= (1<<7) | (1<<3) ; // Активиране на прекъсване и на модула на АЦП (битове ADEN и ADIE)
}
…
// Прекъсване от АЦП
ISR(ADC_vect) {
  if (~ADMUX & (1<<0)) { // Проверка дали BIT 0 = 0
    adcl0=ADCL; // Прочитане и записване на стойността от младшия регистър
    adch0=ADCH; // Прочитане и записване на стойността от старшия регистър
  }
  if (ADMUX & (1<<0)) { // Проверка дали бит 0 = 1
    adcl1=ADCL; // Прочитане и записване на стойността от младшия регистър
    adch1=ADCH; // Прочитане и записване на стойността от старшия регистър
  }
  ADMUX = ADMUX ^ B00000001; // Превключване на каналите на АЦП, мултиплексиране
  ADCSRA |= 1 <<6; // Стартиране на АЦП отново (бит ADSC)
}
…
```

Заключение

Предложеният подход на програмиране чрез манипулиране на портове цели да представи един основен, но и актуален завинаги метод на програмиране, който трябва да се предприема с изключителна увереност, следвайки стриктно документацията. Порт-манипулацията е подходяща в случаи, когато паметта на микроконтролера е оскъдна или е нужно освобождаване на допълнителен цикъл. Така кодът работи по-бързо, отколкото със стандартни функции. Стандартните функции и библиотеки посвоему са налице с причина - по-удобни са за потребителя. Решението и умението са задачи на програмиста. Тази публикация предлага решение на проект именно чрез уменията на авторите ѝ.

Библиография

1. Barrett S. F., Arduino II: Systems, Morgan & Claypool Publishers, 2020.
2. Microchip, ATmega48A/PA/88A/PA/168A/PA/328/P ATmega328P Data Sheet CompleteDS40002061B, Microchip Technology Inc., 2020.
3. PEP 20 - The Zen of Python | Python.org, https://www.python.org/dev/peps/pep-0020/, последно достъпен на 27.10.2020;