**9 ЛАБОРАТОРНАЯ РАБОТА №9**

**ЦИФРОАНАЛОГОВОЕ ПРЕОБРАЗОВАНИЕ. УПРАВЛЕНИЕ ПЕРИФЕРИЙНЫМИ УСТРОЙСТВАМИ ОТЛАДОЧНОГО СТЕНДА EV8031/AVR**

**Цель работы**

Изучить методы цифро-аналогового преобразования. Ознакомиться с функциональными возможностями и внутренней структурой отладочного стенда EV8031/AVR. Изучить внутреннюю структуру и организацию дополнительной платы расширения. Научиться управлять скоростью вращения электродвигателя постоянного тока (вентилятора) или яркостью свечения ламп по заданному алгоритму.

**9.1 Краткие теоретические сведения**

8.1.1 Цифро-аналоговое преобразование

Цифро-аналоговые преобразователи (ЦАП) вырабатывают напряжение или ток, функционально связанные с управляющим кодом. Применяются ЦАП для формирования выходных аналоговых сигналов цифровых измерительных и вычислительных устройств. Для преобразования двоичного кода в аналоговый сигнал обычно формируются токи, пропорциональные весам разрядов кода, и затем суммируются те из токов, которые соответствуют ненулевым разрядам входного кода.

Применяются в основном два метода ЦАП: суммирование единичных эталонных величин и суммирование эталонных величин, веса которых различаются. В первом при формировании выходной аналоговой величины используется только одна эталонная величина весом в один квант. Во втором методе применяются эталонные величины с весами, зависящими от номера разряда, и в суммировании участвуют только те эталонные величины, для которых в соответствующем разряде входного кода установлена единица.

При использовании на входе двоичного позиционного кода значения всех разрядов поступает одновременно, и работа таких ЦАП описывается выражением:

 (8.1)

где X – аналоговая величина;

аi – коэффициенты соответствующих двоичных разрядов, которые принимают дискретные значения – единица или ноль;

Р – опорный сигнал;

b – число разрядов.

В преобразователях из опорного напряжения формируются эталонные величины, соответствующие значениям разрядов входного кода, которые суммируются и образуют дискретные значения выходной аналоговой величины.

Классификация ЦАП может быть проведена по следующим признакам:

а) способ формирования выходного напряжения – с суммирование напряжений, делением напряжений, суммированию токов;

б) вид выходного сигнала – с токовым выходом, с выходом по напряжению;

в) полярность выходного напряжения – постоянное, переменное.

Основные структуры, используемые в ЦАП интегрального исполнения, – это структуры с суммированием токов:

а) ЦАП – со взвешенными резисторами в цепях эмиттеров;

б) ЦАП – со взвешенными резисторами в цепях нагрузки;

в) ЦАП с лестничной матрицей R=2R в цепях эмиттеров транзисторов источников токов;

г) ЦАП с выходной лестничной матрицей R=2R.

К основным параметрам ЦАП относят:

а) n – число разрядов входного управляющего кода;

б) номинальный выходной ток;

в) время установления выходного сигнала после изменения входного управляющего кода;

г) погрешность полной шкалы;

д) погрешность линейности;

е) дифференциальная нелинейность.

Погрешности ЦАП могут быть выражены в процентах или других относительных единицах, а также в долях кванта.

**9.2 Порядок выполнения работы**

9.2.1 Запустить IDE AVR Studio 4.

9.2.2 Создать новый проект в IDE AVR Studio 4.

9.2.3 В появившемся окне написать программу на языке С или ассемблер с учётом варианта задания, который указан в таблице 8.1.

Таблица 9.1 – Варианты заданий

|  |  |
| --- | --- |
| № | Описание задания |
| 10 | Реализовать алгоритм мигания ламп накаливания: при каждом нажатии на кнопку SW16 изменять длительность (в диапазоне от 5с, до 2с с шагом 1с) удержания низкого и высокого уровней (т.е. подавать на ЦАП код 0 либо код 255) питающего напряжения. Генерацию временных промежутков осуществлять таймером-счётчиком T1. |

9.2.4 Произвести компиляцию проекта.

9.2.5 При наличие сообщений об ошибках или предупреждениях вернуться к предыдущему пункту и внести необходимые исправления. В случае некорректной работы программы выполнить её отладку средствами меню Debug.

9.2.6 Проверить подключение USB-кабеля программатора к одноимённому разъёму системного блока.

9.2.7 Загрузить исполняемый файл проекта в микроконтроллер.

9.2.8 Визуально оценить правильность работы написанной программы.

**9.3 Исходные тексты программы**

#define F\_CPU 7372800L //çàäàåì ÷àñòîòó êâàðöà (7,3728 ÌÃö)

#include <avr/io.h>

#include <avr/iom8515.h>

#include <avr/interrupt.h>

#include <util/delay.h>

#define DAC\_REG 0xF000 //îïðåäåëÿåì àäðåñ, ïî êîòîðîìó îñóùåñòâëÿåòñÿ

#define DARK 0x00

#define BRIGHT 0xFF

volatile unsigned int delay\_nmb = 36000;

volatile char end\_time\_interval = 0;

//Ïðåðûâàíèå ïî ñîâïàäåíèþ À òàéìåðà-ñ÷¸ò÷èêà T1

ISR(INT1\_vect){

if (delay\_nmb == 14399){

delay\_nmb = 35999;

} else {

delay\_nmb -= 7200;

}

}

ISR(TIMER1\_COMPA\_vect){

end\_time\_interval = 1;

}

void hardware\_delay(){

TIMSK = 1 << OCIE1A;

OCR1A = delay\_nmb;

SFIOR = 1 << PSR10;

TCCR1A = 0x00;

TCCR1B = 1 << WGM12 | 1 <<CS12 | 1 << CS10;

while(!end\_time\_interval){

};

end\_time\_interval = 0;

}

int main(void) {

//íà÷àëüíàÿ èíèöèàëèçàöèÿ êîíòðîëëåðà

ACSR= 1<<ACD;

MCUCR= 1<<SRE | 1<<ISC10 | 1<<ISC11;

GICR = 1<< INT1;

sei(); //óñòàíàâëèâàåì ôëàã ãëîáàëüíîãî ðàçðåøåíèÿ ïðåðûâàíèé (áóäåò ðàáîòàòü òîëüêî INT0 è INT1)

volatile unsigned char\* dac\_reg = (unsigned char\*)DAC\_REG;

while(1) { //áåñêîíå÷íûé öèêë (îæèäàåì ïðåðûâàíèÿ êàæäûå 0.25 ñåêóíäû)

\*dac\_reg = BRIGHT;

hardware\_delay();

\*dac\_reg = DARK;

hardware\_delay();

}

return 0;

}

**9.4 Особенности IDE AVRStudio выявленные в ходе выполнения лабораторной работы**

В ходе лабораторной работы никаких новых особенностей IDE AVRStudio не было выявлено.

**Выводы**

В ходе лабораторной работы были изучены принципы работы с цифоро-аналоговым преобразованием в учебно отладочном стенде EV8031/AVR. Была написана программа, которая производит которая управляется яркостью лампы накаливания, по нажатию на кнопку SW16 происходит уменьшение длительности свечения и отсутствия света в лампе накаливания на 1 секунду за каждое нажатие.