

Міністерство освіти, науки, молоді та спорту України
Тернопільський національний технічний університет
імені Івана Пулюя
Кафедра приладів і контрольно-вимірювальних систем

Навчально-налагоджувальний стенд **ST841/CPLD (V4.1)**

Методичні вказівки до лабораторних робіт



Проектування пристроїв і вузлів інформаційно-вимірювальних систем
та створення програмного забезпечення на базі навчально-
налагоджувального стенда

Тернопіль 2011

Паламар М.І., Чайковський А.В., Пастернак Ю.В., Стрембіцький М.О. Паламар А.М. Проектування пристроїв і вузлів інформаційно-вимірювальних систем та створення програмного забезпечення на базі навчально-налагоджувального стенда. Методичні вказівки до лабораторних і практичних робіт з дисциплін «Проектування інформаційно-вимірювальних систем», «Мікропроцесори і ЕОМ», «Проектування приладів і систем на основі мікроконтролерів». – Тернопіль: ТНТУ, 2011. – 65 с.

Методичні вказівки містять опис функціональних, принципових схем вузлів лабораторного стенда, порядок завантаження і налагодження програм на ньому, а також індивідуальні завдання до 8 лабораторних робіт з прикладами. Призначені для допомоги у вивченні апаратної частини вимірювально-керуючих систем на основі мікроконтролерів сімейства MCS-51/52, програмованих логічних інтегральних схем (ПЛІС) та створення і налагодження програмного забезпечення для них.

Служать для підвищення ефективності проведення лабораторно-практичних занять і самостійного вивчення дисциплін «Проектування інформаційно-вимірювальних систем», «Мікропроцесори і ЕОМ», «САПР засобів вимірювання», «Основи проектування систем штучного інтелекту», «Автоматизовані системи опрацювання вимірювальної інформації» та ін.

Укладачі: к.т.н., доц. Паламар М.І.
Чайковський А.В.
Пастернак Ю.В.
Стрембіцький М.О.
Паламар А.М.

Методичні вказівки розглянуті та схвалені на засіданні кафедри
Приладів і контрольно-вимірювальних систем

Протокол № від 2011 р.

Одобрено методичною радою факультету ФРК

Протокол № від 2011 р.

СПИСОК СКОРОЧЕНЬ

АЗ	—	апаратні засоби;
АСМ (ASM)	—	асемблер;
АЛБ	—	арифмето-логічний блок;
АЦП	—	аналогово-цифровий перетворювач
БСА	—	блок-схема алгоритму;
ВІС	—	велика інтегральна схема;
ЗПД	—	зовнішня пам'ять даних;
ЗПП	—	зовнішня пам'ять програм;
КПП	—	контролер пріоритетних переривань;
МВР	—	мова високого рівня;
МК	—	мікроконтролер;
МТ	—	мікропроцесорна техніка;
МП	—	мікропроцесор;
ОЗП	—	оперативно запам'ятовуючий пристрій;
ОМЕОМ	—	однокристальна мікроЕОМ;
ПК	—	персональний комп'ютер;
ПЛІС	—	програмована логічна інтегральна схема;
ПП	—	пам'ять програм;
ППЗ	—	прикладне програмне забезпечення;
ППЗП	—	програмований постійний запам'ятовуючий пристрій;
РСФ (RSF)	—	регістр спеціальних функцій;
РПД	—	резидентна пам'ять даних;
РПП	—	резидентна пам'ять програм;
ЦАП	—	цифро-аналоговий перетворювач;
ШІМ	—	широтно-імпульсний модулятор;

Зміст

Вступ	5
Загальна технічна характеристика навчального стенда ST841/CPLD	6
Лабораторна робота №1. Ознайомлення із будовою стенда та архітектурою процесора і системою команд OEOM ADuC841, створення проекту в середовищі Keil. Схема статичного відображення інформації та робота з світлодіодною лінійкою	14
Лабораторна робота №2. Схема статичного відображення інформації та робота з семисегментним індикатором.....	20
Лабораторна робота №3. Схема динамічного відображення інформації.....	24
Лабораторна робота №4. Робота з енкодером. Опитування дискретних датчиків	27
Лабораторна робота №5 Зчитування інформації з матричної клавіатури	31
Лабораторна робота №7. Проектування системи керування кроковим двигуном.....	40
Лабораторна робота №8. Проектування системи керування двигуном постійного струму	46
Лабораторна робота №9. Програмування послідовного інтерфейсу RS-232	52
Список використаних джерел	63
ДОДАТКИ	64

Вступ

Однокристальні мікроконтролери знаходять широке застосування в різноманітних сферах техніки: від вимірювальних приладів, фотоапаратів та відеокамер, принтерів, сканерів, копіювальних апаратів до виробів електронних розваг і будь-якої домашньої техніки.

Мікроконтролери відрізняються від мікропроцесорів за рядом ознак. В першу чергу це їх функціональність. При застосуванні мікропроцесорів для їх роботи потрібні додаткові компоненти як пам'ять, пристрої вводу і виводу даних, генератор тактової частоти і ін. Мікроконтролери сконструйовані таким чином, що всі ці частини зібрані разом на одному кристалі і поміщені в одному корпусі. Для роботи мікроконтролера потрібно мінімум зовнішніх компонентів, тому що вся необхідна периферія вбудована в його середині. Таким чином зменшується апаратна частина і скорочується час при конструюванні нових пристроїв.

З моменту появи перших мікроконтролерів їх складність постійно зростає за рахунок появи нових апаратних рішень та додавання нових команд для програмної підтримки нових функціональних вузлів, призначених для вирішення різноманітних задач.

Проектування вбудованих систем на сучасній елементній базі значно підвищує ефективність розробки за рахунок скорочення часу, мініатюризації, зниження споживаної потужності і збільшення швидкодії і надійності. На сьогоднішній день актуальною є задача придбання навиків розробки інформаційно-керуючих пристроїв на базі мікроконтролерів і програмованих логічних пристроїв, що дозволяють реалізувати алгоритми високої складності.

Розроблений стенд дозволяє ознайомитися із сучасною елементною базою, практично вивчати типові вузли приладів інформаційних вимірювальних систем, алгоритмів роботи, проводити відладку нових програм, організувати взаємодію датчиків з мікроконтролером, а також керувати виконуючими пристроями, такими як двигуни різних типів, електромагнітні виконавчі органи, та інші.

Загальна технічна характеристика навчального стенда ST841/CPLD



Рисунок 1 – Зовнішній вигляд стенда ST841/CPLD

Стенд призначений для освоєння архітектури та методів проектування інформаційно-керуючих систем, систем збору і оброблення інформації на базі найпоширеніших мікроконтролерів сімейства MCS-51/52, а також ПЛІС фірм Altera або Xilinx.

Стенд може бути використаний для:

- вивчення вузлів мікроконтролерних інформаційно-вимірювальних систем, промислових інтерфейсів, пристроїв вводу-виводу;
- виконання курсових, дипломних та науково-дослідних робіт.

- розробки і відлагодження апаратно-орієнтованого програмного забезпечення;
- взаємодії із аналоговими та цифровими первинними перетворювачами та виконавчими органами;

Функціональні можливості:

- мікроконтролер ADuC841 (однотактове ядро, 62 кбайт флеш-пам'яті програм, 4 кбайт флеш-пам'яті даних, 2304 байт оперативної пам'яті, апаратна підтримка інтерфейсів UART, I2C, SPI, восьмиканальний 12-розрядний АЦП, 320кГц, два 12-розрядні цифро-аналогові перетворювачі, джерело опорної напруги, датчик температури);
- програмована логічна матриця CPLD: EPM3032, EPM3064 (Altera), або XCR3032XL, XCR3064XL (Xilinx);
- інвертуючий, неінвертуючий, диференційний та інструментальний підсилювачі;
- пристрої вводу:
 - матрична клавіатура,
 - дискретні кнопки,
 - джойстик-потенціометр,
 - механічний енкодер;
- пристрої відображення:
 - лінійка світлодіодів,
 - 4 семисигментні індикатори,
 - 1 матричний індикатор,
 - індикатор на рідких кристалах;
- звукова підсистема:
 - звуковий рст кодек (TLV320AIC1109),
 - підсилювач звукової частоти TDA7052,
 - динамік, мікрофонний вхід, вихід на телефони;
- шини даних:
 - паралельна 8-бітна шина даних;
 - SPI
 - I2C
 - RS232
 - RS485
- цифрова периферія:
 - годинник реального часу з I2C інтерфейсом DS1307-33;
 - 8 Мбіт флеш-пам'ять із SPI інтерфейсом,
 - генератор прямокутних імпульсів (NE555),
 - генератор тактової частоти;
- виконавчі пристрої:
 - драйвери крокових двигунів (8 ключів);
 - ШІМ-регулятор;
 - Н-міст для двигуна постійного струму.

Будова контролера

Як процесор використовується однокристальний мікроконтролер ADuC841 з такими характеристиками:

- Однотактне 20MIPS ядро архітектури 8052;
- Швидкісний 12-розрядний АЦП;
- Два 12-розрядні ЦАП;
- 62 кбайт вбудованої пам'яті програм;
- 2 кбайти вбудованої пам'яті даних;
- UART, I2C, SPI інтерфейси;

- Програмування та можливість відладки через інтерфейс UART.

Для роботи із стендом необхідне знання архітектури контролера, його регістрів та периферії. Проведемо короткий огляд найбільш важливих вузлів.

ADuC841 є функціонально завершеним контролером інтелектуальних датчиків і включає в себе високоякісний багатоканальний АЦП із самокалібруванням, два ЦАПи, і швидкий (20МГц) з однократовим виконанням команд 8-ми розрядний програмований мікроконтролер з системою команд МК 8051 на однім кристалі. Ядром МК є контролер 8052, що забезпечує пікову продуктивність до 20 MIPS. На кристалі розміщено 62Кбайт Flash пам'яті програм, 4Кбайт Flash пам'яті даних, 256 байт пам'яті з довільним доступом (RAM) і 2Кбайт розширеної пам'яті з довільним доступом (XRAM).

В склад ADuC841 входять також додаткові аналогові пристрої: два 12-розрядні ЦАПи, монітор напруги живлення і джерелом опорної напруги. Додатковими цифровими пристроями є: два 16-розрядні $\Sigma\Delta$ ЦАПи, два 16-розрядні широтно-імпульсні модулятори, сторожовий таймер, лічильник часових інтервалів, три таймери-лічильники, і три порти послідовного вводу-виводу (SPI, I2C, і UART). Заводська прошивка контролера підтримує завантаження програмного забезпечення через послідовний порт UART, а також емуляцію через один контакт пристрою – EA. Далі приведена функціональна схема ADuC841.

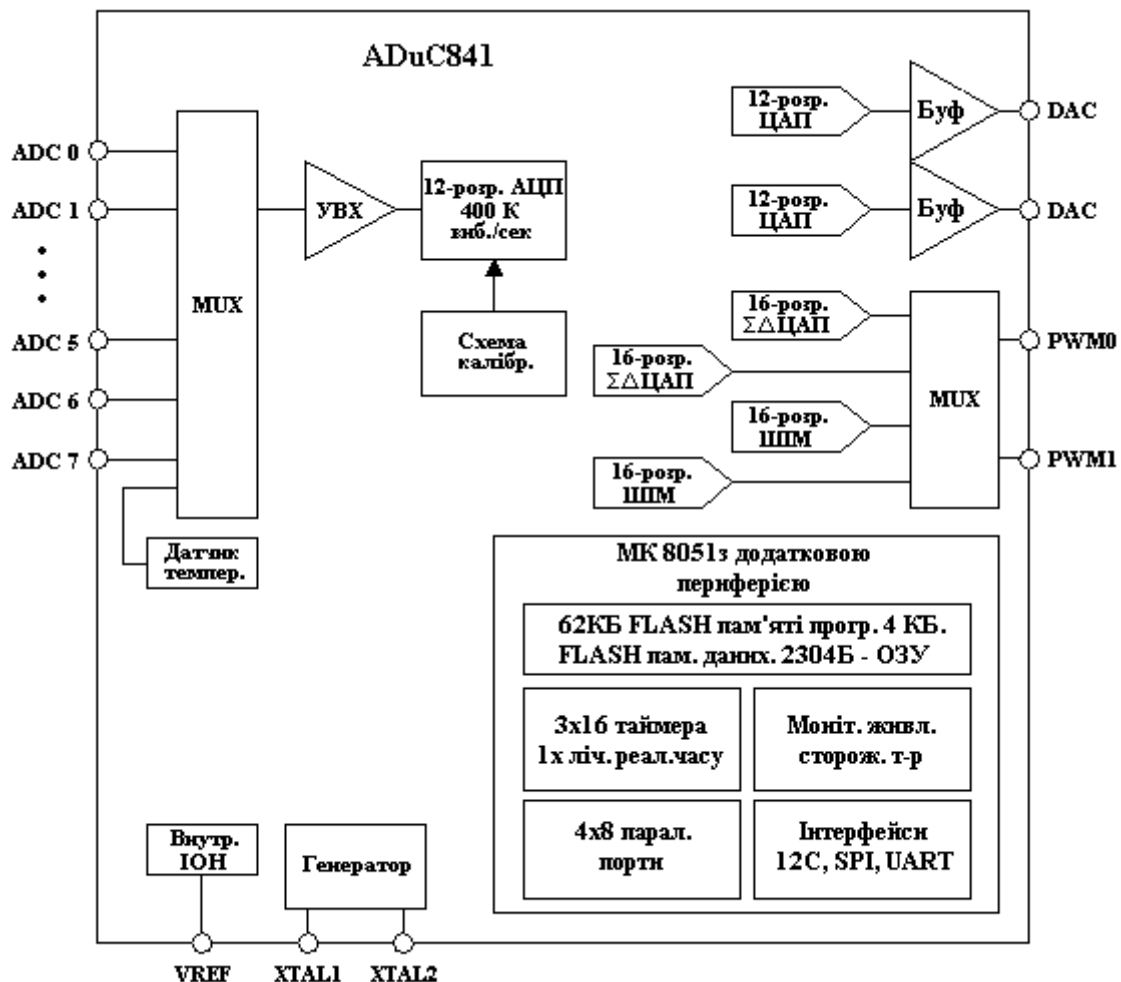


Рисунок 2 – Функціональна схема ADuC841

Периферійні пристрої стенду

Програмована логічна інтегральна схема.

На платі стенду розміщена панелька для встановлення ПЛІС типу EPM3032ALC44, EPM3064ALC44 виробництва Altera а також сумісних з ними XCR3032XL, XCR3064XL

виробництва XILINX. Для завантаження програмного забезпечення в ПЛІС та його відладки призначений роз'єм інтерфейсу JTAG. ПЛІС працює незалежно від МК і має свої периферійні вузли.

Блок операційних підсилювачів для аналогових сигналів.

Блок призначений для підсилення входних сигналів, що подаються на входи АЦП МК і сигналів з виходів ЦАП.

Звуковий кодек.

Звуковий кодек TLV320AIC1109 призначений для перетворення аналогового сигналу звукової частоти в стиснений цифровий формат для подальшої передачі, а також для здійснення зворотного перетворення цифрових даних в звуковий сигнал. Для підключення мікрофона і телефона передбачено роз'єми. Сигнал з виходу кодека може подаватись на розміщені на платі підсилювач і динамік.

Джойстик та механічний енкодер.

На платі установлений потенціометричний джойстик з двома осями і кнопкою. Сигнал з виходів джойстика подається на входи 6 та 7 АЦП МК.

Механічний енкодер призначений для перетворення кутового переміщення в електричний сигнал. Дані пристрої дозволяють проектувати системи з необхідністю ручного вводу інформації (наприклад, маніпулятори, прилади, спецтехніка, блоки керування і т.п.).

Цифрова клавіатура.

Матрична клавіатура застосовується при побудові вузлів вводу даних в інформаційну систему.

Кнопки переривань та скидання процесора.

Кнопки переривань призначені для генерації сигналів переривання процесора. Кнопка скидання призначена для скидання процесора під час програмування або для перезапуску існуючої програми.

Рідкокристалічний дисплей.

На платі установлений знакосинтезуючий рідкокристалічний 4x10 дисплей, підключений до шини даних по чотирибітній схемі. Дисплей має вбудовану підсвітку та регулятор контрасту.

Семисегментний чотиризначний та матричний індикатори, світлодіодна лінійка.

На основі світлодіодів, семисегментних та матричних індикаторів будуються підсистеми відображення інформації. Схема включення семисегментного індикатора дозволяє здійснювати статичну індикацію, матричного – динамічну, знакосинтезуючу.

Інтерфейси стандартів RS485, Ethernet, I2C, SPI, 1-wire та системний інтерфейс.

На платі розміщено драйвери інтерфейсів UART, який побудований на мікросхемі CP2102, що дозволяє під'єднувати стенд до USB входу комп'ютера, RS485, а також Ethernet-драйвер, які узгоджують логічні рівні даних стандартів та TTL рівень а також роз'єми для підключення кабелів чи шлейфів периферійних пристроїв. Системний інтерфейс включає шину даних, адресні лінії, лінії UART, I2C, живлення та загальний провід.

Годинник реального часу DS1307.

Використовується для генерації точних секундних імпульсів, а також як годинник і календар.

Восьмиканальний драйвер крокових двигунів.

Драйвер дозволяє керувати двома уніполярними двигунами з чотирма обмотками невеликої потужності (із споживаним струмом до 2А).

Контрольні точки для підключення осцилографа.

Контрольні точки розміщені у важливих функціональних вузлах і дозволяють проводити візуальне спостереження форми сигналу за допомогою осцилографа.

Генератор імпульсів.

Генератор прямокутних імпульсів зі зноу частоти генерації побудований на основі таймера 555 і може використовуватись для генерації контрольних імпульсів.

Стабілізатори живлення.

На платі встановлено стабілізатори напруг 5В, 3.3В.

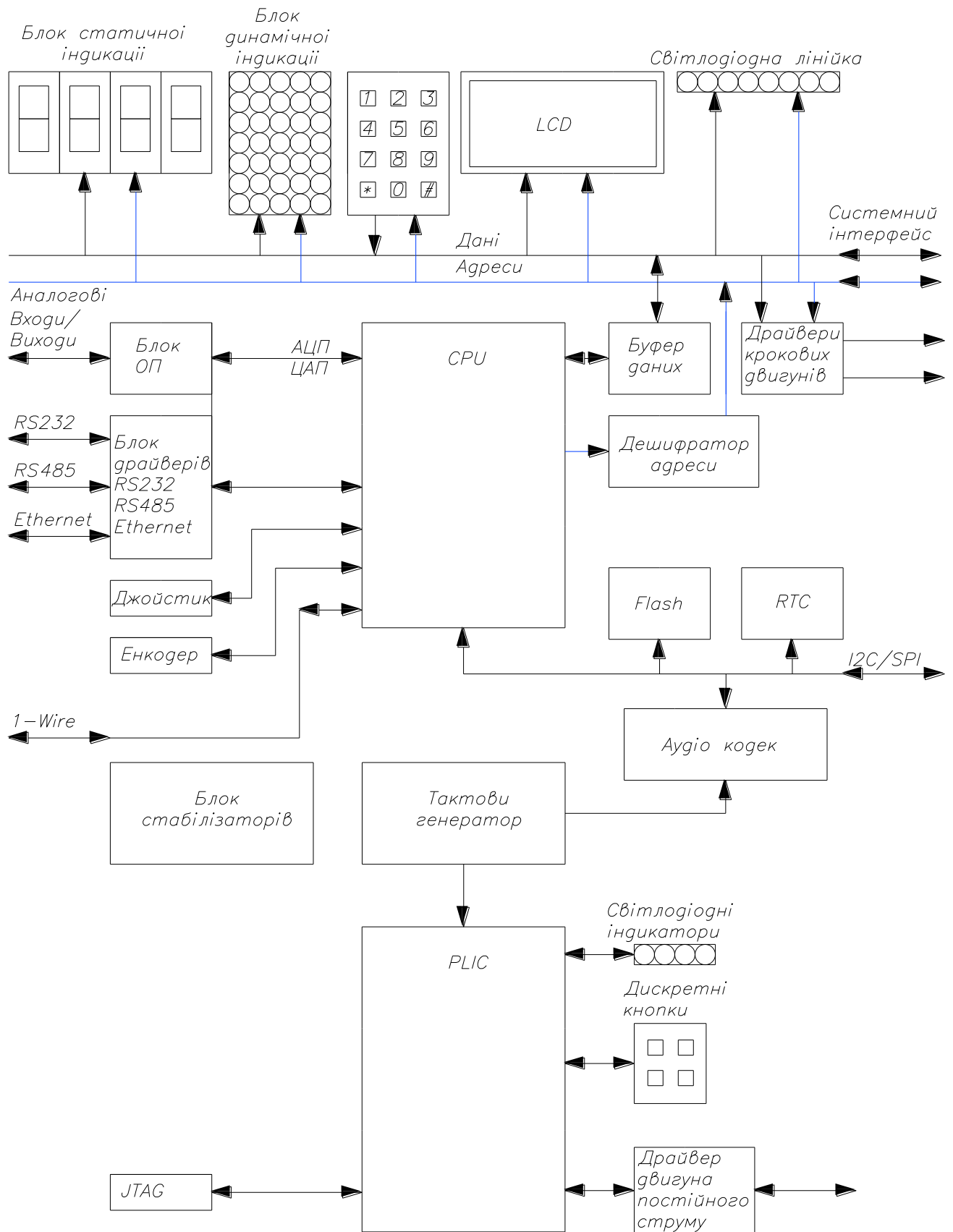


Рисунок 3 – Структурна схема навчального стенду

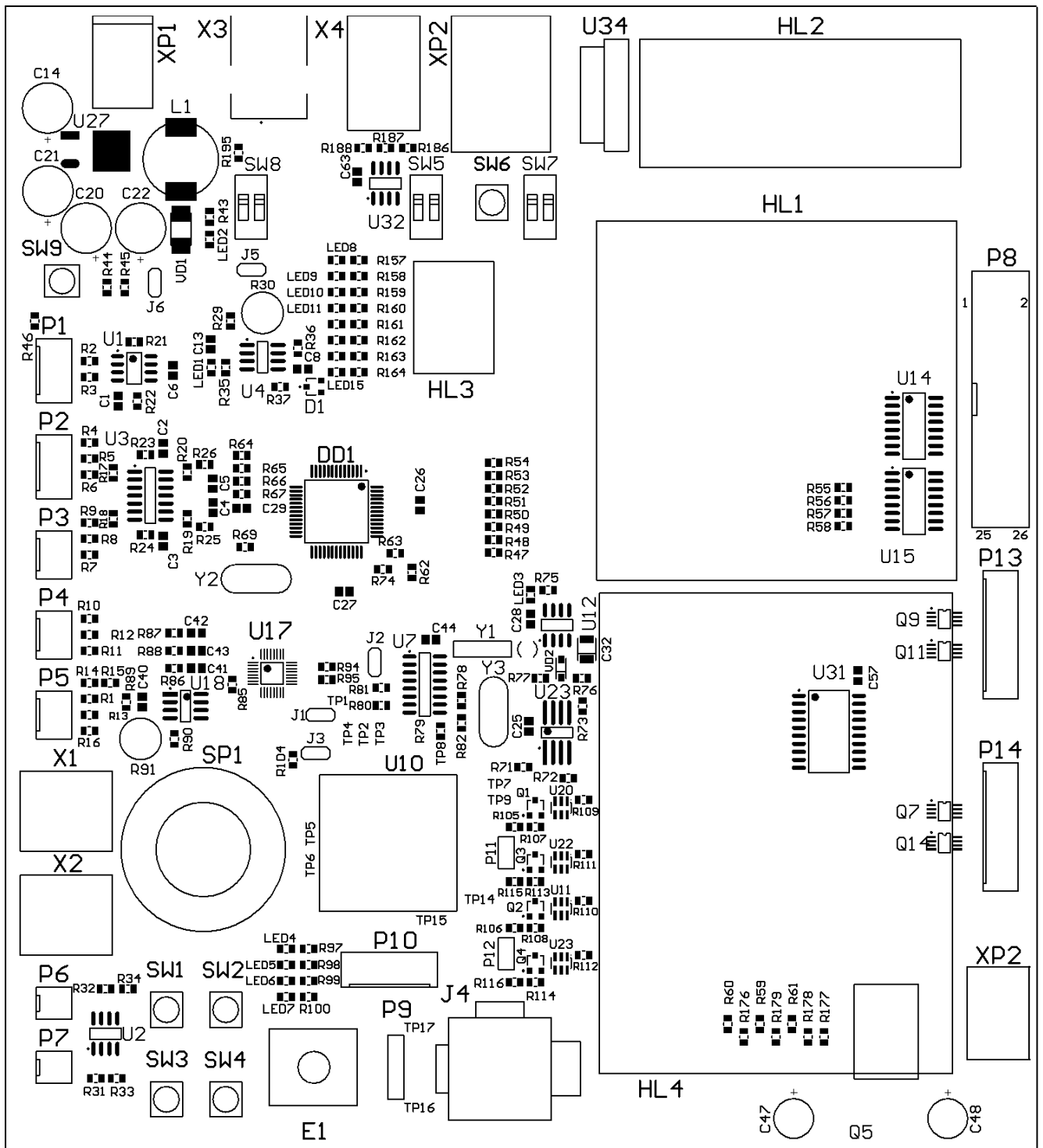


Рисунок 4 – Розміщення елементів станду на платі (вигляд зверху)

Для виконання лабораторних та відлагодочних робіт необхідно знати розміщення та призначення окремих вузлів та елементів керування на платі станду. До елементів керування відносяться перемички та кнопки. Коротко опишемо призначення перемичок, необхідні їх положення та роз'єми для підключення до станду периферійних вузлів.

Таблиця 1 – Положення перемичок

№ пп	Позначення по схемі	Положення	Значення
1,2	JP1, JP2	Розімкнуто	-
		Замкнуто	Ліній P3.4, P3.5 з'єднані з кодеком
3	JP3	Розімкнуто	-
		Замкнуто	Лінія GND з'єднана з 10 виводом ПЛІС
5	JP5	Розімкнуто	-
		Замкнуто	Автоматичне перезавантаження з МК
6	JP6	Розімкнуто	Робода
		Замкнуто	Програмування

Таблиця 2 – Положення перемикачів

№ пп	Позначення по схемі	Положення	Значення
1	SW8	Розімкнуто	-
		Замкнуто	Замкну толінії RX, TX на CP2102 (зв'язок з ком'ютером)
2	SW5	Розімкнуто	-
		Замкнуто	Замкну толінії RX, TX на MAX485ESA (RS-485)
3	SW7	Розімкнуто	-
		Замкнуто	Замкну толінії RX, TX на EM500 (Ethernt модуль)

Таблиця 3 – Призначення роз'ємів

Позначення на платі	Призначення роз'єма
XP1	Живлення стенду
XP2	Підключення двигуна постійного струмі зі зовнішнім живленням
XP3	Інтерфейс Ethernet
X1	Підключення мікрофона до входу аудіо кодека
X2	Підключення навушників до виходу аудіо кодека
X3	Інтерфейс USB
X4	Інтерфейс RS-485
P1	Вхід інструментального підсилювача
P2	Вхід диференційного підсилювача
P3	Вхід не інвертованого підсилювача
P4	Вхід інвертованого підсилювача
P5	Вхід не інвертованого підсилювача зі зміщенням нуля
P6	Вихід ЦАП DAC0
P7	Вихід ЦАП DAC1
P8	Лінії системного інтерфейсу
P9	Інтерфейс 1-wire
P10	Програмування ПЛІС (JTAG)
P11, P12	Вихід Н-моста від ПЛІС
P13, P14	Підключення крокового двигуна

Таблиця 4 – Призначення виводів системного інтерфейсу

№ по схемі	Позначення по схемі	Призначення
1..8	DAT0..DAT7	Шина даних
10..13	CS12..CS15	Адресні лінії
14	SDATA	Лінія MOSI SPI інтерфейсу/ лінія SDA I2C інтерфейсу
15	SCLOCK	Лінія SCK SPI інтерфейсу/ лінія SCL I2C інтерфейсу
16	DIR	Лінія напрямку даних
17,18	T0,T1	Входи таймерів-лічильників
19	INT0	Лінія переривання 0
20	INT1/MISO	Лінія переривання 1/ лінія MISO SPI інтерфейсу
21,22	RXD,TXD	Лінії послідовного інтерфейсу UART
22,23	+5V	Напруга живлення
9,25,26	GND	Спільний вивід

Таблиця 5 – Призначення виводів роз'єму PKI

№ по схемі	Позначення по схемі	Призначення
1	GND	Спільний вивід
2	VCC	Вивід живлення
3	Vo	Регулювання контрасту
4	RS	Вибір регістру команд/даних
5	R/W	Читання/запис (постійно на запис)
6	E	Строб запису
7-10	N/C	Не під'єднано
11-14	DAT4..DAT7	4-розрядна шина даних

Таблиця 6 – Призначення виводів роз'єму для крокового двигуна

№ по схемі	Призначення
1,2	Позитивний полюс живлення
3,4	Обмотка 1
5,6	Обмотка 2
7,8	Обмотка 3
9,10	Обмотка 4

Лабораторна робота №1. Ознайомлення із будовою стенда та архітектурою процесора і системою команд OEOM ADuC841, створення проекту в середовищі Keil. Схема статичного відображення інформації та робота з світлодіодною лінійкою

Тема: Ознайомлення із будовою стенду та архітектурою процесора і системою команд OEOM ADuC841, створення проекту в середовищі Keil, схема статичного відображення інформації та робота з світлодіодною лінійкою.

Мета роботи: Написати програму для роботи зі світлодіодною лінійкою, відладити її в середовищі Keil, завантажити програму в контролер, вивчити можливості апаратної відладки.

Порядок виконання роботи:

1. Вивчити структурну схему стенду, спосіб адресації до периферійних вузлів, призначення окремих елементів схеми, ознайомитись із структурою та системою команд OEOM ADuC841.
2. Ознайомитись із середовищем Keil, навчитись створювати новий проект, ознайомитись із методами програмної відладки.
3. Розробити алгоритм індивідуального завдання до початку заняття.
4. Розробити програму і скомпілювати її.
5. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

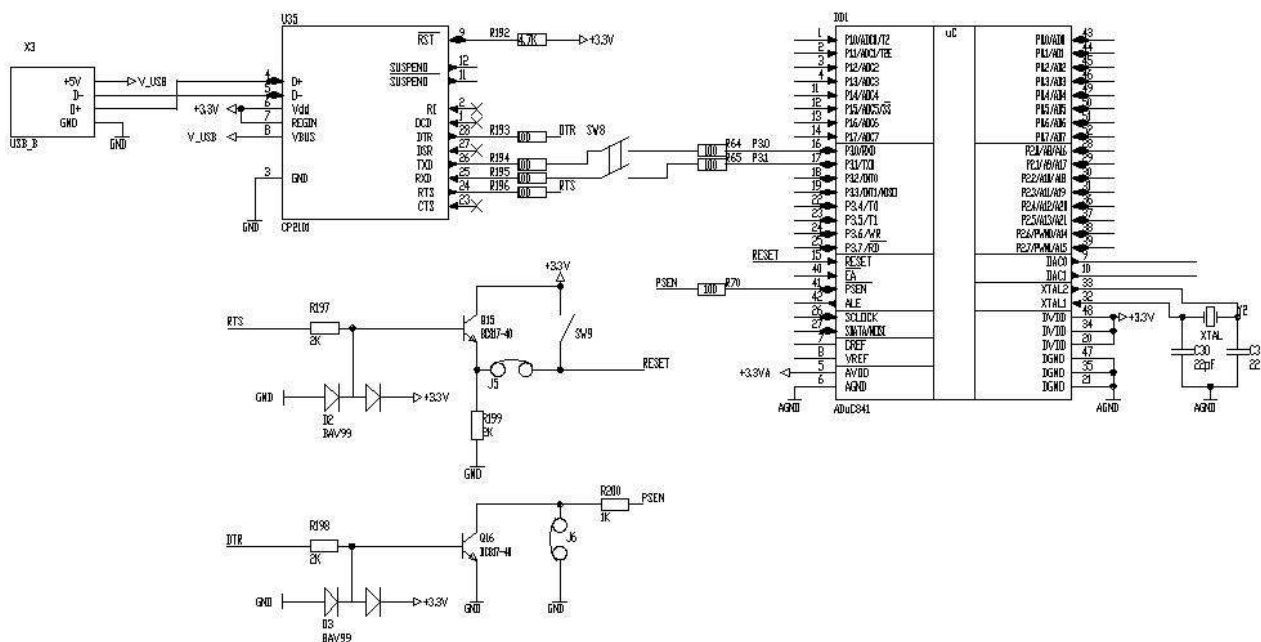


Рисунок 1.1 – Схема для лабораторної роботи №1

Короткі теоретичні відомості

Створення проекту в Keil

- а) запускаємо середовище
- б) меню Project -> NewProject
- в) у вікні, що появилось, вказати директорію, в якій буде збережений проект або створити нову, вказати назву проекту
- г) вибрати тип контролера – Analog Devices -> ADuC841
- д) початковий код до проекту не додавати
- е) створити прив'язку – для цього натиснути кнопку Options for Target:

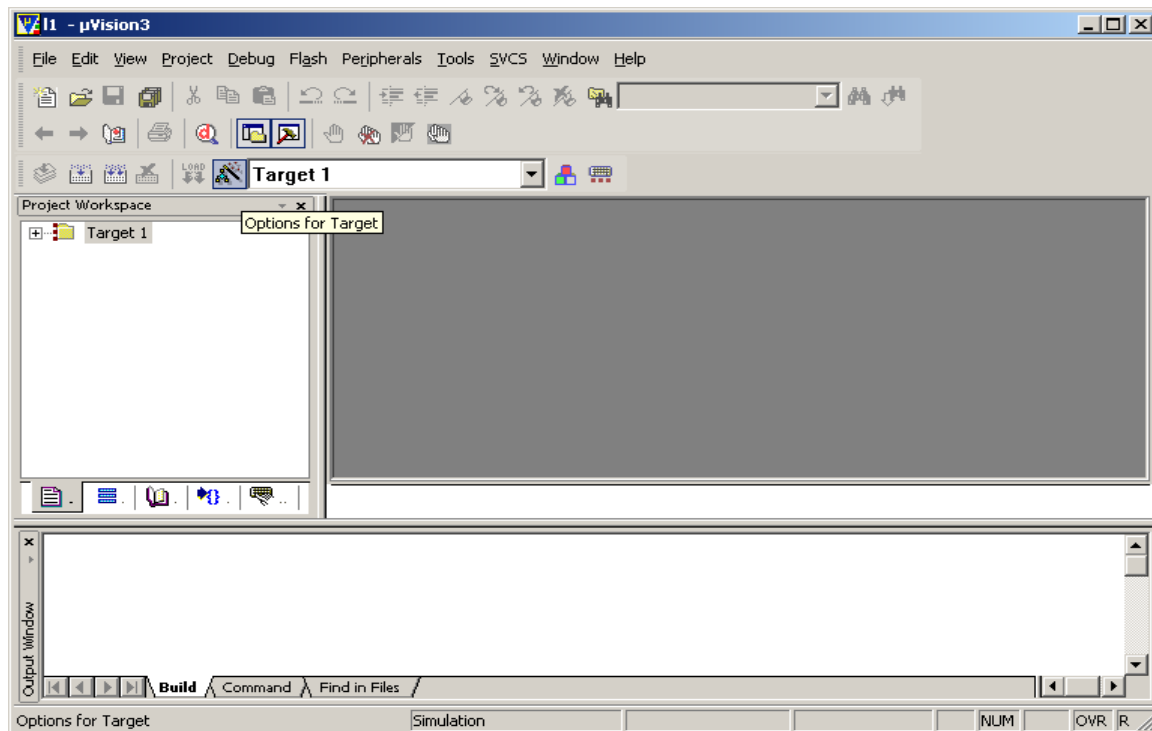


Рисунок 1.2 – Вибір створення прив'язки

У вікні, що появилось, на закладці Target необхідно вказати частоту кварцового резонатора, що використовується – 11,0592 МГц.

На закладці Output слід відмітити пункт Create HEX File.

На закладці Utilities відмітити пункт Use Target Driver for Flash Programming, із списку слід вибрати пункт ADI Monitor Driver, (рисунок 1.2), натиснути кнопку Settings, встановити відповідний порт для зв'язку (COM1), швидкість лінії 9600, пункти RTS і DTR встановити у значення Inactive.

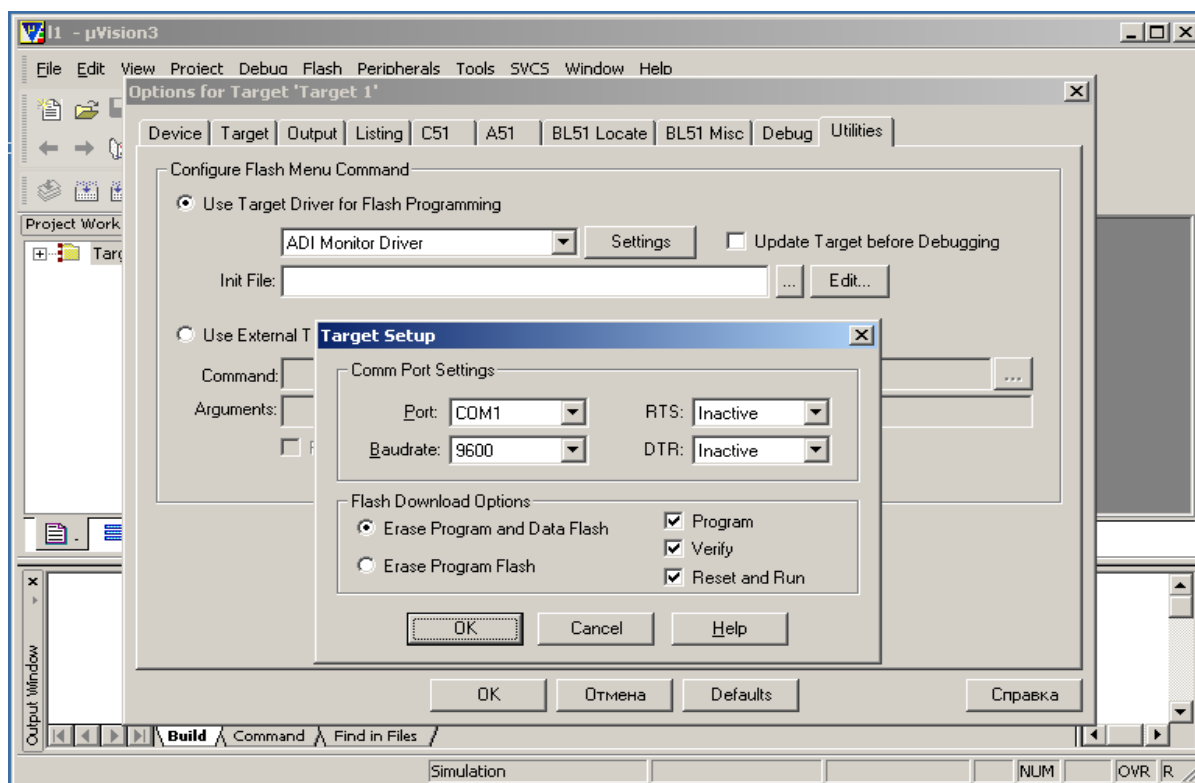


Рисунок 1.3 – Настроювання прив'язки

передачу від контролера до шини значення сигналу DIR повинно бути рівним логічній 1, а для роботи на прийом від шини до контролера – логічному 0.

Покажемо часові діаграми роботи контролера із периферійними пристроями.

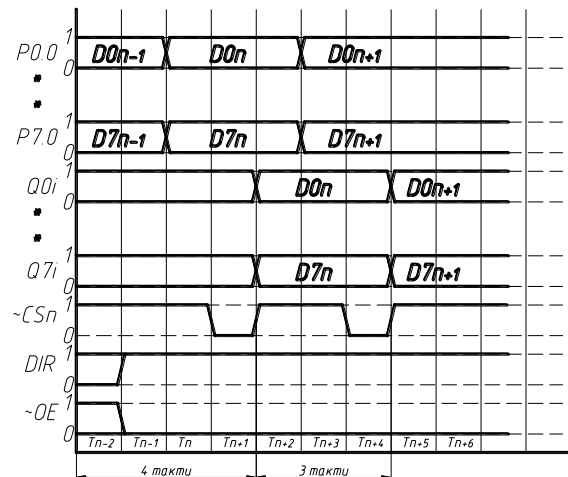


Рисунок 2.1 – Часова діаграма запису двох чисел в регістр периферійного пристрою

Число фіксується регістром (периферійним пристроєм) по зростаючому фронту сигналу $\sim CS$. Послідовність роботи контролера така:

1. Встановлюється напрям роботи буфера (DIR=1) і увімкнення його і дешифраторів ($\sim OE=0$);

2. В порт P0 записується число, яке потрібно записати;

3. В порт P2.0-P2.3 записується адрес периферійного пристрою (регістра);

4. В порт P2.0-P2.3 записується число 0000b.

Таким чином запис числа проходить за 4 такти.

Якщо напрям роботи буфера був встановлений і дешифратори увімкнені, то процес проходить за 3 такти.

При зчитуванні контролером із шини даних потрібно встановити напрям роботи буфера на читання (DIR=1), дати строб $\sim CS$ для відповідного адресу, і зчитати стан порту P0. Слід пам'ятати, що з регістрів, які розміщені на платі стенду, зчитування неможливе, але зчитувати можна із системного роз'єму.

Набрати текст програми, натиснути кнопку Rebuild all target files.

Для того, щоб завантажити відкомпільований файл у стенд необхідно:

1. Замкнути перемичку J6.
2. Перемикач SW8 перевести в положення замкнуто.
3. Натиснути кнопку SW9.
4. Вибрати команду Flash -> Download.

Внаслідок таких дій програма завантажиться в пам'ять програм мікроконтролера і запуститься на виконання.

Приклад програми для лабораторної роботи №1

На світлодіодній лінійці засвітити біжучу точку.

```
// address of LEDs      0x07
// define DATA transfer register
DAT EQU R0
// define ADDRESS transfer register
ADDR EQU R1
// define temporary0 register
Temp0 EQU R2
// Address for PUSH and POP commands
Temp0 EQU 0x02
//Interrupt vector
ORG 0x0000
JMP RESET
```

```

ORG      0x0033
RESET:
MOV      DAT, #11111111b
MOV ADDR, #00000111b
CALL write
MOV A, #0x01
Forever:
    Rr A
    MOV DAT, A
    MOV ADDR, #0x07
    CALL write
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    CALL LDelay
    JMP Forever
// Write to peripheral device subroutine
write: setb P3.6          //Set Data bus buffer to TX
        mov P0, DAT      //moving data to bus buffer
        mov P2, ADDR     //set peripherals address
        nop              //wait
        nop
        mov P2, #0x00    //clock pulse for device latch
        ret              //exit from subroutine
// Delay 17.9 ms
LDelay: MOV      Temp0, #0xFF
LD1:     DEC      Temp0
        PUSH     _Temp0
        MOV      Temp0, #0xFF
        DJNZ     Temp0, $
        POP      _Temp0
        CJNE     Temp0, #0x00, LD1
RET
END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	На світлодіодній лінійці засвітити індикатори (LED 8 LED 15) через один, нижній індикатор засвічений, час засвітки 1с., після цього погасити індикатор, час витримати 1с., повторити процедуру спочатку.
2	На світлодіодній лінійці засвітити індикатори (LED 8 LED 15) через один, нижній індикатор загашений, час засвітки 2с., після цього засвітити всі індикатори, час світіння 1с. дальше повторити процедуру спочатку.
3	На світлодіодній лінійці засвітити індикатори (LED 8 LED 11), потім погасити, процедуру повторювати з частотою 1Гц
4	На світлодіодній лінійці засвітити індикатори (LED 12 LED 15), потім погасити, процедуру повторювати з частотою 0,5Гц
5	На світлодіодній лінійці засвітити біжучу точку, напрямом від LED 8 до LED 15, швидкість переміщення 1 сегмент всекенду
6	На світлодіодній лінійці засвітити біжучу точку, напрямом від LED 15 до LED 8, швидкість переміщення 2 сегмент всекенду
7	На світлодіодній лінійці засвітити індикатори в режимі “інкрементного термометра”, напрямом від LED 15 до LED 8, швидкість наростання 1 сегмент в секенду (початковий стан: засвічено LED 15, черз 1 с. засвічено LED 15 і LED 14, коли засвічено всі світлодіоди повертаємось в початковий стан)
8	На світлодіодній лінійці засвітити індикатори в режимі “інкрементного термометра”, напрямом від LED 8 до LED 15, швидкість наростання 2 сегмент в секенду (початковий стан: засвічено LED 8, черз 0,5с. засвічено LED 8 і LED 9, коли засвічено всі світлодіоди повертаємось в початковий стан)
9	На світлодіодній лінійці засвітити індикатори в режимі “декриментно термометра”, напрямом від LED 15 до LED 8, швидкість спадання 2 сегмент в секенду (початковий

	стан: засвічено всі світлодіоди, через 2 с. засвічено з LED 15 по LED 9, коли всі світлодіоди погаснуть повертаємось в початковий стан)
10	На світлодіодній лінійці засвітити індикатори в режимі “декриментно термометра”, напрямок від LED 8 до LED 15, швидкість спадання 1 сегмент в секунду (початковий стан: засвічено всі світлодіоди, через 1 с. засвічено з LED 14 по LED 8, коли всі світлодіоди погаснуть повертаємось в початковий стан)

Лабораторна робота №2. Схема статичного відображення інформації та робота з семисегментним індикатором

Тема: Схема статичного відображення інформації та робота з семисегментним індикатором .

Мета: Вивчити метод статичного відображення інформації на прикладі семисегментного індикатора.

Порядок виконання роботи:

1. Вивчити принцип статичного методу відображення інформації на прикладі семисегментного індикатора.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скопіювати її.
4. Завантажити програму в стенд, переконатись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

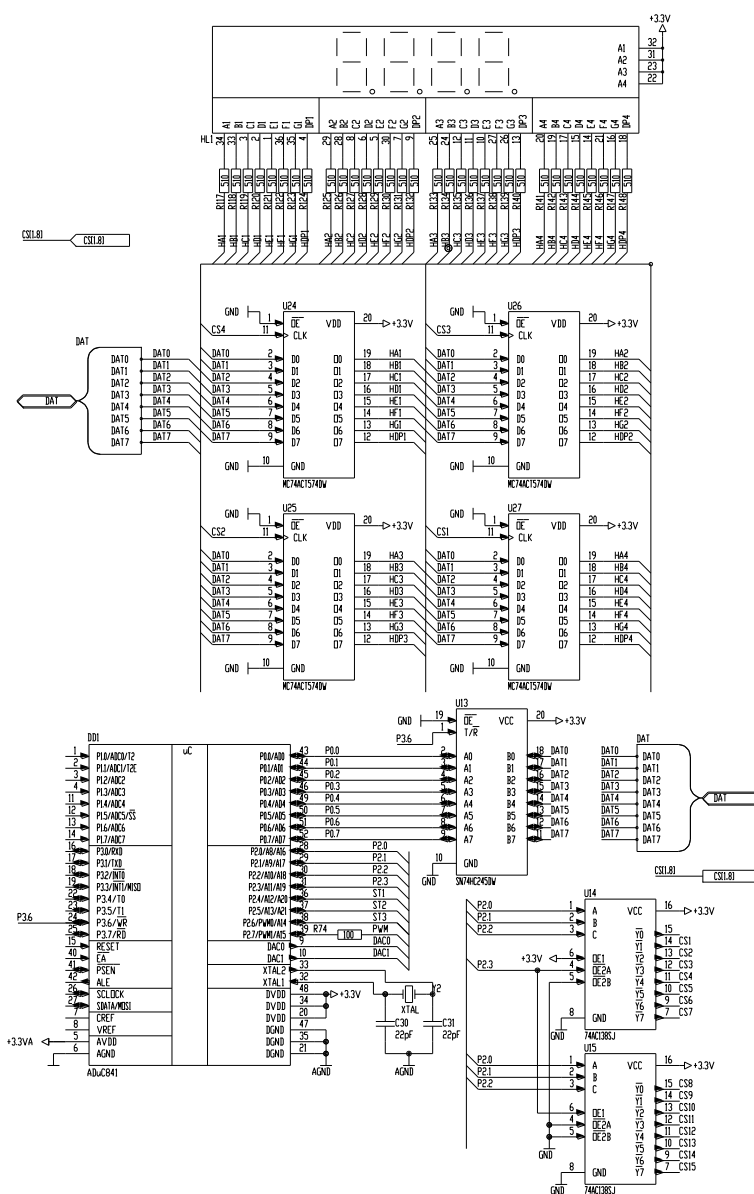


Рисунок 2.1 – Схема для лабораторної роботи №2

Короткі теоретичні відомості

До вузла статичної індикації входять: 4 регістри-фіксатори, струмообмежуючі резистори і чотири цифровий світлодіодний семисегментний дисплей. Схема їх увімкнення показана на рисунку 3.1. Зміст статичної індикації полягає у тому, щоб кожен індикатор постійно висвічувався від свого джерела інформації.

Число, яке виводиться на індикатор, фіксується відповідним регістром. Після запису числа в регістр, шина даних може приймати довільне значення, воно не буде впливати на значення, що висвічується на індикаторі. Хоча застосування статичної індикації і потребує відносно великих апаратних затрат, зате відпадає потреба в постійному переключенні індикаторів, як у випадку динамічної індикації, спрощується програмне забезпечення, контролер звільняється від необхідності постійної почергової передачі чисел на кожен індикатор, як у випадку динамічної індикації.

Коли потрібно засвітити сегмент індикатора, слід встановити відповідний біт регістра, рівний логічному нулю. Для того, щоб погасити відповідний індикатор, у регістр потрібно записати число 0FFh. Оскільки індикатори підключені без дешифраторів, то процес дешифрування слід проводити програмно. Програмне дешифрування дозволяє виводити на індикатор не тільки числа, а й деякі букви та інші символи.

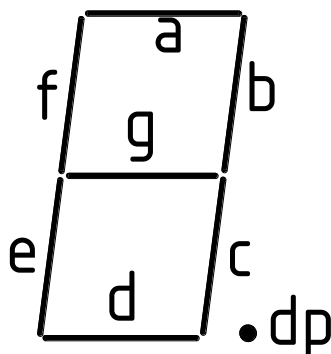


Рисунок 3.1 – Схема позначення сегментів індикатора

Таблиця 3.1 – Дешифрування двійкового коду для семисегментного індикатора

Сегмент/цифра	1	2	3	4	5	6	7	8	9	0
Dat0 (a)	1	0	0	1	0	0	0	0	0	0
Dat1 (b)	0	0	0	0	1	1	0	0	0	0
Dat2 (c)	0	1	0	0	0	0	0	0	0	0
Dat3 (d)	1	0	0	1	0	0	1	0	0	0
Dat4 (e)	1	0	1	1	1	0	1	0	1	0
Dat5 (f)	1	1	1	0	0	0	1	0	0	0
Dat6 (g)	1	0	0	0	0	0	1	0	0	1
Dat7 (dp)	1	1	1	1	1	1	1	1	1	1

Для засвічування десяткової коми (dp, decimal point) потрібно встановити біт 7 дешифратора, рівний 0.

Адреси цифр дисплея розміщені наступним чином:

для запису в перший (справа наліво) розряд адрес рівний 01h;

для запису в другий розряд P2 адрес рівний 02h;

для запису в третій розряд P2 адрес рівний 03h;

для запису в четвертий розряд P2 адрес рівний 04h.

Запис відбувається згідно послідовності, вказаної в лабораторній роботі №2.

Приклад програми для лабораторної роботи №2

На семисегментному індикаторі почергово появляються цифри '1', '9', '8', '8'.

```
dat      EQU      R0
adr      EQU      R1
Temp1    EQU      R2
Temp2    EQU      R3
Temp3    EQU      R4

Segm     EQU      0x02

CSEG
ORG 0x0000
JMP run
ORG 0x0033
run:
    call off
    call ldelay
    call ldelay
    mov dat, #11001111b
    mov adr, #00000100b
    call strob
    call ldelay
    mov dat, #10010000b
    mov adr, #00000011b
    call strob
    call ldelay
    mov dat, #10000000b
    mov adr, #00000010b
    call strob
    call ldelay
    mov dat, #10000000b
    mov adr, #00000001b
    call strob
    call ldelay

petla:
    jmp petla

strob:
    setb P3.6
    mov P0,dat
    mov P2,adr
    mov P2,#0x00

ret
off:
    mov dat, #11111111b
    mov adr, #00000001b
    call strob
    mov dat, #11111111b
    mov adr, #00000010b
    call strob
    mov dat, #11111111b
    mov adr, #00000011b
    call strob
    mov dat, #11111111b
    mov adr, #00000100b
    call strob

ret

Delay:
    mov Temp1, #0xFF
loop:
    dec Temp1
    mov Temp2, #0xFF
    djnz Temp2, $
    cjne Temp1, #0x00, loop

ret

Ldelay:
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay

RET
END
```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	На семисегментному індикаторі засвітити на старших двох розрядах xxH , через інтервал в 1с. погасити число і на двох молодших розрядах число засвітити число xxH , час світіння 1с., після цього повторювати процедуру спочатку.
2	На семисегментному індикаторі засвітити на молодшому розряді число xH , через інтервал 1с, на старшому розряді засвітити число xH (при цьому попереднє число не гаситься), через 1с. вивести на молодший розряд суму чисел, старший розряд погасити, час засвітки 1с., даліше повторити процедуру спочатку.
3	На семисегментному індикаторі засвітити на двох молодших розрядах число $0xH$, через інтервал в 1с. погасити число і на то самому індикаторі засвітити число $x0H$, час світіння 1с., після цього повторювати процедуру спочатку, старший розряд погашено.
4	На семисегментному індикаторі засвітити на молодшому розряді число xH , через інтервал 1с, на старшому розряді засвітити число xH (при цьому попереднє число не гаситься), через 1с. вивести на молодший розряд добуток чисел, старший розряд погасити, час засвітки 1с., даліше повторити процедуру спочатку.
5	На семисегментному індикаторі починаючи з молодшого розряді засвітити число xH , через інтервал в 1с. погасити його і відобразити те саме число на наступному розряді, через 1с., на наступному, коли засвітиться останній індикатор, процедуру повторити з початку.
6	На семисегментному індикаторі засвітити на старшому розряді число xxH , через інтервал 1с, на молодшому розряді засвітити число xH (при цьому попереднє число не гаситься), через 1с. вивести на молодший розряд частку від ділення першого числа на перше, старший розряд погасити, час засвітки 1с., даліше повторити процедуру спочатку.
7	На семисегментному індикаторі на молодшому розряді засвітити число $0H$, через інтервал в 0,5с. інкрементувати дане значення, коли число буде рівне 10, обнулити індикатор і процедуру повторити з початку, решта розрядів індикатора погашено.
8	На семисегментному індикаторі засвітити на старшому розряді число xxH , через інтервал 1с, на молодшому розряді засвітити число xxH , яке має бути менше за перше (при цьому попереднє число не гаситься), через 1с. вивести на молодший розряд різницю від віднімання першого числа від другого чисел, старший розряд погасити, час засвітки 1с., даліше повторити процедуру спочатку.
9	На семисегментному індикаторі на старшому розряді засвітити число $0H$, через інтервал в 0,1с. інкрементувати дане значення, коли число буде рівне 100, обнулити індикатор і процедуру повторити з початку, решта розрядів індикатора погашено.
10	На семисегментному індикаторі на молодшому розряді засвітити число $59H$, через інтервал в 1с. декремнтувати значення, коли число буде рівне 0 засвітити на індикатор $59H$ і процедуру повторити з початку, решта розрядів індикатора погашено.

Лабораторна робота №3. Схема динамічного відображення інформації

Тема: Схема динамічного відображення інформації.

Мета: Вивчити метод динамічного відображення інформації на прикладі матриці світлодіодів.

Порядок виконання роботи:

1. Вивчити принцип динамічного методу відображення інформації на прикладі матриці світлодіодів.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

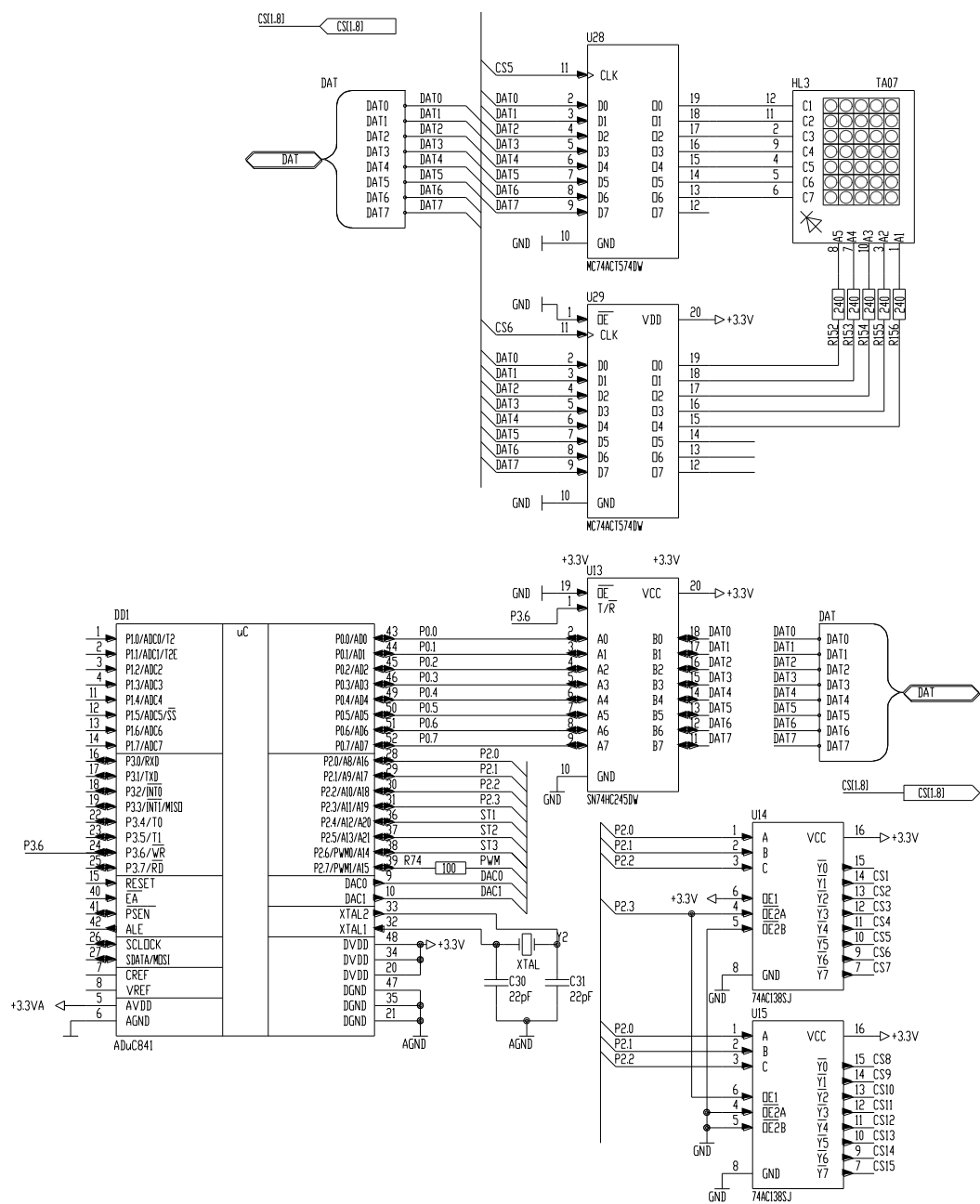


Рисунок 3.1 – Схема для лабораторної роботи №3

Короткі теоретичні відомості

Динамічна знакосинтезуюча індикація реалізована на світлодіодній матриці 5x7 і дозволяє вивести довільний символ, організувати біжучу стрічку та ін. Зміст динамічної індикації полягає у тому, щоби по черзі засвічувати елементи індикатора з такою частотою, при якій око за рахунок деякої інерційності сприйняття не спроможне помітити мерехтіння. Максимум частоти переключення обмежений необхідністю забезпечити такий час світіння елемента, при якому значно не зменшується його яскравість. Елементи індикатора засвічуються записом логічних нулів у відповідні розряди регістру рядків, який підключений до катодів індикатора (на схемі – DD17, адрес регістра 05h) і логічних одиниць у розряди регістру стовбців, підключеного до анодів (на схемі – DD18, адрес регістра 06h). Є зміст виводити знак на індикацію не по елементах (точках), а по рядках. В такому випадку, для того, щоб засвітити у верхньому рядку першу і четверту точку, потрібно в регістр DD17 записати число 11111110b, а в регістр DD18 – число 00001001b.

Приклад програми для лабораторної роботи №3

Засвітити на матриці світлодіодів літеру “S”.

```

dat Equ R0
adr Equ R1
Temp1 Equ R2
Temp2 Equ R3
stec Equ 0x02

Org 0x000
run:
    mov dat,#10111110b
    mov adr,#00000101b
    call writ
;    call delay
    mov dat,#00001110b
    mov adr,#00000110b
    call writ
;    call delay
    call off

    mov dat,#11011101b
    mov adr,#00000101b
    call writ
;    call delay
    mov dat,#00010001b
    mov adr,#00000110b
    call writ
;    call delay
    call off

    mov dat,#11111011b
    mov adr,#00000101b
    call writ
;    call delay
    mov dat,#00010000b
    mov adr,#00000110b
    call writ
;    call delay
    call off

    mov dat,#1110111b
    mov adr,#00000101b
    call writ
;    call delay
    mov dat,#00000001b
    mov adr,#00000110b
    call writ
;    call delay
    call off

    mov dat,#1111011b
    mov adr,#00000101b
    call writ
endl:
END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	На матричний світлодіодний індикатор засвічувати 4 символи, період світіння 1с.

Лабораторна робота №4. Робота з енкодером. Опитування дискретних давачів

Тема: . Робота з енкодером. Опитування дискретних давачів.

Мета: Вивчити принцип роботи енкодера, здійснити опитування дискретних давачів, ознайомитися з системою переривання контролера.

Порядок виконання роботи:

1. Вивчити принцип роботи енкодера, ознайомитися з системою переривання контролера на основі дискретних давачів.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
5. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

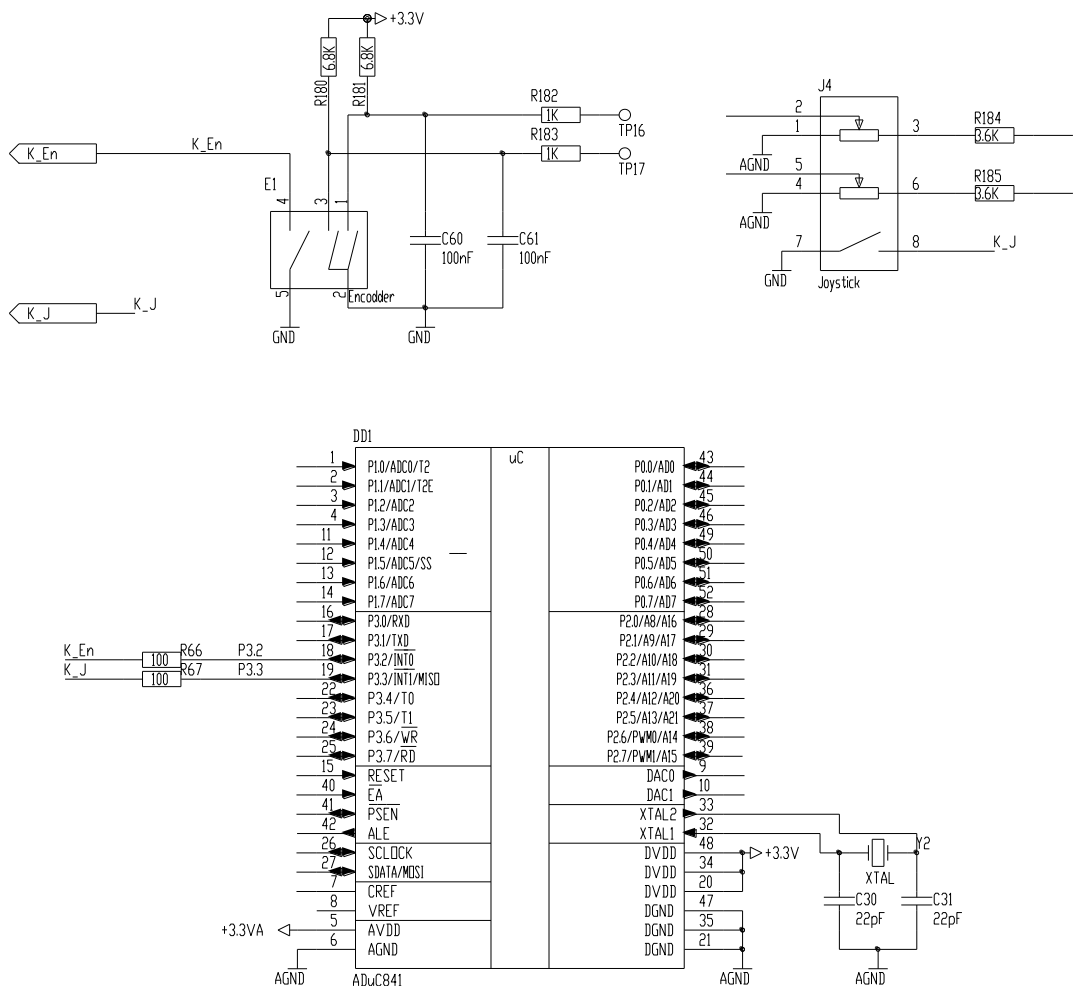


Рисунок 4.1 – Схема для лабораторної роботи №5

Короткі теоретичні відомості

Для введення інформації широко застосовуються кнопкові перемикачі. Сигнал таких перемикачів формується шляхом замикання (розмикання) електричного кола.

В стенді встановлено дві дискретні кнопки K_En і K_J, які під'єднані до входів зовнішніх переривань контролера відповідно до INT0 і INT1. Опитування даних давачів можна здійснювати як з використання переривань так і способом опитування стану окремого виводу потра контролера.

Також для введення інформації використовується механічний енкодер E1.

Енкодер (датчик кута або перетворювач кут-код) – пристрій, який призначений для перетворення кута повороту обертового об'єкта (вала) в електричні сигнали, які дозволяють визначити кут його повороту.

Енкодери розділяються на інкрементні і абсолютні, які можуть забезпечити дуже високу роздільну здатність. В інкрементних датчиках кутового положення поточне положення визначається шляхом підрахунку числа імпульсів від нульової точки.

На виході енкодера є два прямокутних сигнали, які зсунуті відносно один одного, в даному випадку на 90° (рисунок 4.2).



Рисунок 4.2 – Сигнали на виході енкодера

По характеру зсуву можна визначити напрямок обертання ручки вала енкодера. Розрізняється також кількість імпульсів на оберт, для даного типу енкодера – 20.

Приклад програми для лабораторної роботи №6

Написати програму для роботи з енкодером. При повертанні ручки за годинниковою стрілкою збільшувати значення і значення виводити у двійковому вигляді на світлодіодну лінійку, при повертанні прити годинникової стрілки зменшувати значення.

```
$include (mod841)
    DAT    equ    R7
    ADDR   equ    R6

    State data 9

org 0
JMP BEGIN

ORG 030H
BEGIN:
    mov    SP,    #2fh
    mov    State, #0

loop:
    call   ReadEnc
    mov    a, State
    add    a, R7
    mov    State, a
    mov    DAT, a
    mov    ADDR, #7
    call   write
    call   Zatr
    jmp    loop

ZATR:
MOV    R7, #0H
DJNZ R7, $
RET

encA bit 0
encB bit 1
```

```

ReadEnc:
    mov a, #0
    mov c, encA
    rlc a
    mov c, P3.4
    mov encA, c
    rlc a
    cjne a, #1, ReadEnc_NotFirst
;first
    mov c, P3.5
    mov encB, c
ReadEnc_NotFirst:
    cjne a, #2, ReadEnc_NotSecond
;second
    mov a, #0
    mov c, encB
    rlc a
    mov c, P3.5
    rlc a
    cjne a, #1, ReadEnc_NotCW
;CW
    mov R7, #1
    ret
ReadEnc_NotCW:
;CCW
    cjne a, #2, ReadEnc_NotCCW
    mov R7, #255
    ret
ReadEnc_NotCCW:
ReadEnc_NotSecond:
    mov R7, #0
    ret

;====*END*====

write:    setb P3.6                //Set Data bus buffer to TX
        mov P0, DAT                //moving data to bus buffer
        mov P2, ADDR //set peripherals address
        nop                        //wait
        nop
        mov P2, #0x00 //clock pulse for device latch

    ret                                //exit from subroutine

end

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	При натисканні кнопки на енкодері інкрементувати змінну і виводити на молодший розряд статичного індикатора. Змінна приймає значення 0...99, прогарма не повинна реагувати на тривале натискання на кнопку. Початковий стан на статичному індикаторі засвічено 00 на молодшому розряді, старший розряд погашено, при досягненні значення 100 на індикаторі засвічується 00.
2	При натисканні кнопки на енкодері декрементувати змінну, яка відображається на старшому розряді статичного індикатора, при натисканні кнопки на джойстику декрементується змінна, яка відображається на старшому розряді статичного індикатора. Змінні приймають значення 99...00, при досягненні значення -1 засвічується 99, прогарма не повинна реагувати на тривале натискання на кнопки. Початковий стан на статичному індикаторі засвічено 00 на старшому розряді, молодший розряд погашено.
3	При обертанні ручки енкодера в напрямку за годинниковою стрілкою переміщати на світлодіодній лінійці засвічену точку вгору, при обертанні ручки енкодера в напрямку проти годинникової стрілки переміщати точку на світлодіодній лінійці вниз, при досягненні крайнього положення точка переводиться на протилежний край.
4	При обертанні ручки енкодера в напрямку за годинниковою стрілкою переміщати на світлодіодній лінійці погашену точку вниз, при обертанні ручки енкодера в напрямку проти годинникової стрілки переміщати погашену точку вгору, при досягненні крайнього положення точка переводиться на протилежний край.
5	При натисканні кнопки на енкодері переміщувати засвічену точку на матричному

	світлодіодному індикаторі, початковий стан: засвічена точка знаходить в лівому нижньому куті. При натисканні на кнопку енкодера точка переміщується по стовпці вверху, при досягненні верхнього положення точка переміщається на нижній рядок і на один стовпець вправо, при закінченні останнього стовпця, точка повертається в початкове положення
6	При натисканні кнопки на джойстику перемішувати загашену точку на матричному світлодіодному індикаторі, початковий стан: загашена точка знаходить в правому верхньому куті. При натисканні на кнопку джойстика точка переміщується по стовпці вниз, при досягненні нижнього положення точка переміщається на верхній рядок і на один стовпець вліво, при закінченні останнього стовпця, точка повертається в початкове положення.
7	На світлодіодному індикаторі запустити “біжучу точку”, напрямом знизу вгору, швидкість переміщення задавати обертанням ручки енкодера, максимальна швидкість 8 сегментів за 1с., мінімальна швидкість 1 сегмент за 8с., при обертанні ручки енкодера за годинниковою стрілкою швидкість збільшується в двічі відносно встановленої, при обертанні ручки проти годинникової стрілки швидкість зменшується вдвічі відносно встановленої. При досягненні мінімального або максимального значення швидкість у відповідному напрям не міняється.
8	На світлодіодному індикаторі запустити погашену “біжучу точку”, напрямом зверху вниз, швидкість переміщення задавати обертанням ручки енкодера, максимальна швидкість 16 сегментів за 1с., мінімальна швидкість 1 сегмент за 4с., при обертанні ручки енкодера за годинниковою стрілкою швидкість збільшується в двічі відносно встановленої, при обертанні ручки проти годинникової стрілки швидкість зменшується вдвічі відносно встановленої. При досягненні мінімального або максимального значення швидкість у відповідному напрям не міняється.
9	Засвічувати світлодіоди на індикаторі за допомогою дискретних кнопок. При натисканні кнопки на енкодері збільшувати число засвічених індикаторів на один, при натисканні кнопки на джойстику зменшувати число засвічених індикаторів на один. При засвіченні всіх індикаторів програма ігнорує натискання кнопки енкодера, при погашенні всіх індикаторів програма ігнорує натискання кнопки джойстика. Початковий стан: світлодіодна лінійка погашена.
10	При обертанні ручки енкодера за годинниковою стрілкою інкрементувати значення змінної, яка виводиться на молодший розряд семи сегментного індикатора, при обертанні ручки енкодера проти годинникової стрілки декрементувати значення змінної. При досягненні значення 100 змінна приймає значення 00, при досягненні значення -1 змінна приймає значення 99.

Лабораторна робота №5 Зчитування інформації з матричної клавіатури

Тема: Зчитування інформації з матричної клавіатури.

Мета: Вивчити взаємодію контролера із засобами вводу даних, вивід даних на засоби відображення інформації.

Порядок виконання роботи:

1. Вивчити алгоритми зчитування даних із клавіатури.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

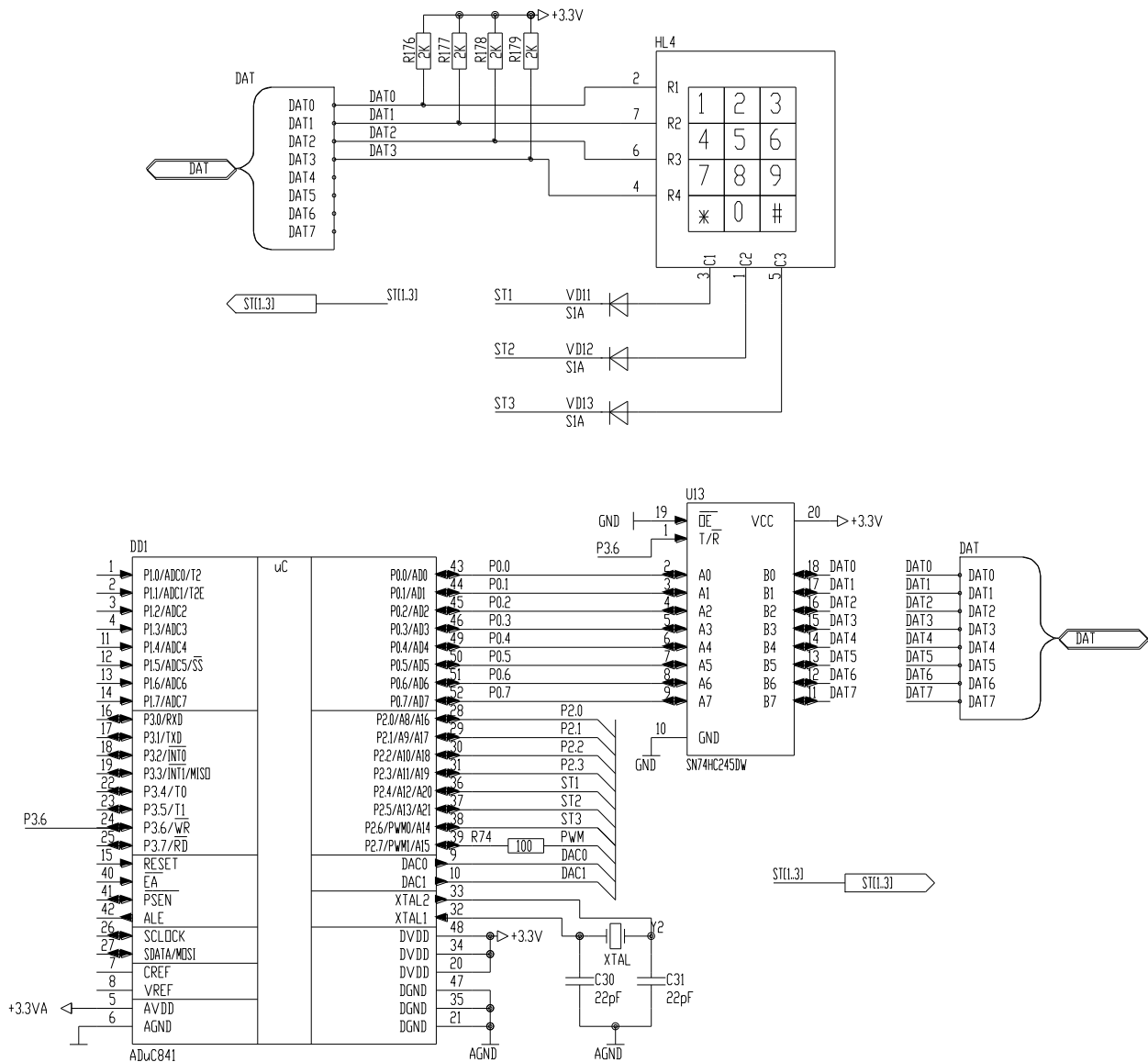


Рисунок 5.1 – Схема для лабораторної роботи №5

Короткі теоретичні відомості

Зчитування із клавіатури відбувається методом сканування по рядках. Для здійснення зчитування із першого стовпця потрібно записати в порт P2 адресу 0EFh, і зчитати дані із старшого півбайта порту P0. Для цього необхідно задати напрям передачі даних буфера DD6 на прийом інформації (встановити вивід DIR в логічний нуль), це значення задається портом мікроконтролера P3.6. Для роботи на прийом від шини до контролера встановити P3.6 в логічний 0. При цьому якщо натиснута кнопка із першого рядка, то значення порту P0.0 буде рівне логічному 0, якщо натиснута кнопка із другого рядка, то P0.1=0 і т.д. Для зчитування із другого стовпця в порт P2 записується адрес 0DFh, третього – 0BFh.

Зчитування із рядків клавіатури слід проводити не частіше ніж через 50мс, через те, що для будь-якого механічного контакту характерне явище вібрації контактної пари, протягом якої контакт декількаразово розмикається-замикається, протягом приблизно 8-12мс.

Приклад програми для лабораторної роботи №5

Програма зчитує дані з клавіатури і засвічує відповідний світлодіод.

```
dat Equ R0
adr Equ R1
Temp1 Equ R2
Temp2 Equ R3
;key1 Equ R4
;key2 Equ R5
stec Equ 0x02

Org 0x000
jmp run

cseg
org 0x033

run:

stovb1:
    mov adr, #01100000b
    mov P2, adr
    clr p3.6
    mov a, p0

next0:
    jb p0.0, next1
    call svit8
    jmp run

next1:
    jb p0.1, next2
    call svit5
    jmp run

next2:
    jb p0.2, next3
    call svit2
    jmp run

next3:
    jb p0.3, stovb2
    call gas1
    jmp run

stovb2:
    mov adr, #01010000b
    mov P2, adr
    clr p3.6
    mov a, p0

next02:
    jb p0.0, next12
    call svit7
    jmp run

next12:
    jb p0.1, next22
    call svit4
    jmp run

next22:
    jb p0.2, next32
    call svit1
    jmp run

next32:
    jb p0.3, stovb3
    call svit7

    call svit6
    call svit5
    call svit4
    call svit3
    call svit2
    call svit1
    jmp run

stovb3:
    mov adr, #00110000b
    mov P2, adr
    clr p3.6
    mov a, p0

next03:
    jb p0.0, next13
    call svit6
    jmp run

next13:
    jb p0.1, next23
    call svit3
    jmp run

next23:
    jb p0.2, next33
    call svit2
    call svit3
    call svit4
    call svit5
    call svit6
    call svit7
    call svit8
    jmp run

next33:
    jb p0.3, stovb1
    call gas1
    jmp run

svit1:
    mov dat, #01111111b
    mov adr, #00000111b
    call writ
    call ldelay

ret

svit2:
    mov dat, #10111111b
    mov adr, #00000111b
    call writ
    call ldelay

ret

svit3:
    mov dat, #11011111b
    mov adr, #00000111b
    call writ
    call ldelay

ret

svit4:
```

```

        mov dat,#11101111b
        mov adr,#00000111b
        call writ
        call ldelay

ret

svit5:
        mov dat,#11110111b
        mov adr,#00000111b
        call writ
        call ldelay

ret

svit6:
        mov dat,#11111011b
        mov adr,#00000111b
        call writ
        call ldelay

ret

svit7:
        mov dat,#11111101b
        mov adr,#00000111b
        call writ
        call ldelay

ret

svit8:
        mov dat,#11111110b
        mov adr,#00000111b
        call writ
        call ldelay

ret

gas1:
        mov dat,#11111111b

```

```
mov adr,#00000111b
call writ
call ldelay
```

	кнопки, на індикаторі відображається 00. Початковий стан: молодший розряд індикатор погашено, на старшому засвічено 00.
6	За допомогою кнопок матричної клавіатури переміщувати засвічену точку по матричному індикаторі, “4” – переміщення на один сегмент вліво, “6” – переміщення на один сегмент вправо, “2” – переміщення на один сегмент вгору, “8” – переміщення на один сегмент вниз. При досягненні країв екрану засвічений індикатор за межі не виходить. Початковий стан: засвічено індикатор по середині матриці.
7	За допомогою кнопок матричної клавіатури переміщувати засвічену точку по діагоналі матричному індикаторі, “1” – переміщення на один сегмент вліво і вгору, “3” – переміщення на один сегмент вправо і вгору, “9” – переміщення на один сегмент вправо і вниз, “7” – переміщення на один сегмент вліво і вниз. При досягненні країв екрану засвічений індикатор за межі не виходить. Початковий стан: засвічено індикатор по середині матриці.
8	На статичному індикаторі засвіти крійний сегмент і переміщувати його по контуру індикатора, швидкість стала – 1 сегмент за секунду. При натисканні кнопки матричної клавіатури “*” сегмент рухається за годинниковою стрілкою, при натисканні кнопки “#” сегмент рухається проти годинникової стрілки, при натисканні “0” рух сегмента припиняється. Початковий стан: засвічено один довільний сегмент.
9	При натисканні кнопки на матричній клавіатурі від “0” до “4” номер натиснутої кнопки відображати на молодшому розряді статичного індикатора, при натисканні кнопки від “5” до “9” відображати номер натиснутої кнопки на старшому розряді статичного індикатора, натискання “*” гасить всі розряди індикатора. Початковий стан індикатор погашено.
10	На світлодіодній лінійці засвічувати індикатори з різною тривалістю світіння в залежності від номера натиснутої клавіші. Так при натисканні “1” засвічується 1 індикатор і мигає з частотою 1Гц, при натисканні кнопки “8”, засвічується 8 індикаторів і частота мигання становить 0,125Гц. Програма реагує лише на цифрові кнопки від “1” до “8”. Початковий нижній індикатор мигає з частотою 1Гц.

Лабораторна робота №6. Аналогово-цифрове перетворення. Робота з джойстиком

Тема: Аналогово-цифрове перетворення. Робота з джойстиком.

Мета: Вивчити методи оцифровування аналогових сигналів.

Порядок виконання роботи:

1. Вивчити алгоритм обробки даних аналого-цифровим перетворювачем.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження програми в стенд.

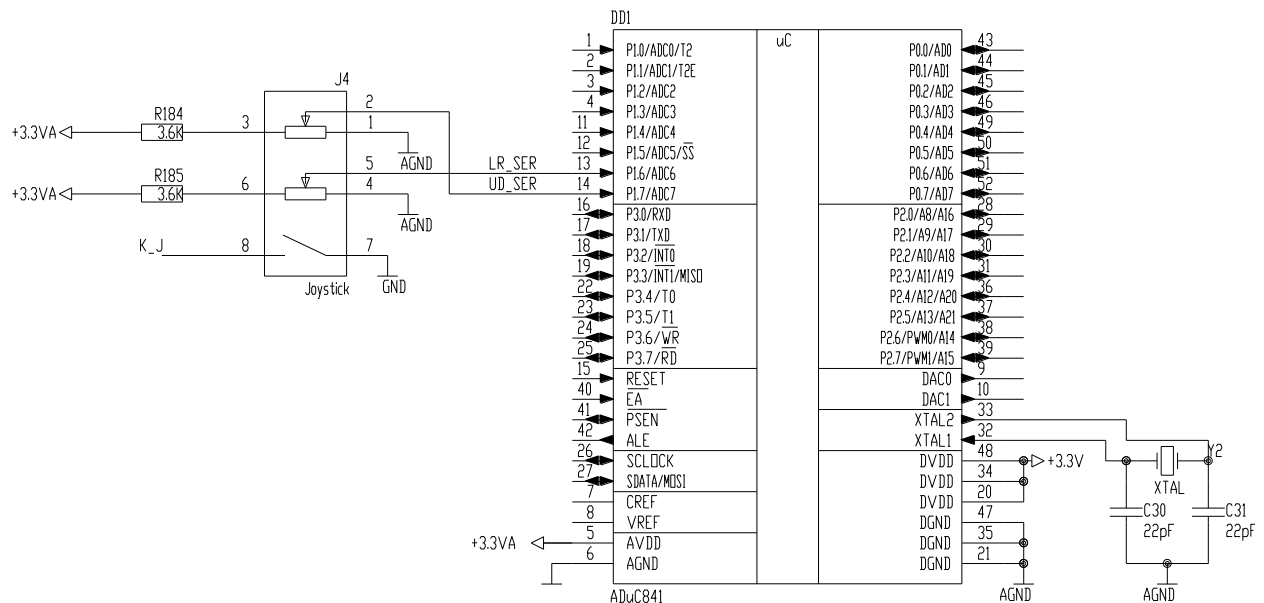


Рисунок 6.1 – Схема для лабораторної роботи №6

Короткі теоретичні відомості

АЦП, аналогово-цифровий перетворювач – пристрій, що перетворює вхідний сигнал в дискретний код (цифровий сигнал). Як правило, АЦП – електронний пристрій, що перетворює напругу в двійковий код.

Розрядність АЦП характеризується кількістю дискретних значень, які перетворювач може видати на виході, вимірюється в бітах. Наприклад, АЦП, здатний видавати 256 дискретних значень (0...255), має розрядність 8 біт, оскільки: $2^8=256$.

Розрядність може бути також визначена в величинах вхідного сигналу і виражена, наприклад, в вольтах. Розрядність за напругою дорівнює напрузі, що відповідає максимальному вихідному коду, який ділиться на кількість вихідних значень.

Мікроконтролер ADuC481 як і інші пристрої компанії ADI має високоточний АЦП і програмований ЦАП.

АЦП керується за допомогою регістрів спеціальних функцій ADCCON1..ADCCON3. Молодших 8 біт результату перетворення зберігаються в регістрі ADCDATAL, старших 4 біти – в молодшій тетраді регістру ADCDATAH. В старшу тетраду контролер записує номер поточного каналу.

Значення регістру ADCCON1 керує режимом запуску та роботи АЦП. Його адреса – 0xEF, початкове значення – 0x40, побітова адресація не підтримується.

Таблиця 6.1 – Біти регістру ADCCON1

Біт	Назва	Призначення
7	MD1	Ввімкнення АЦП. Якщо біт встановлений – АЦП ввімкнено
6	EXT_REF	Біт встановлений – АЦП працює від зовнішнього джерела опорної напруги, Біт скинений – АЦП працює від внутрішнього джерела опорної напруги.
5	CK1	Визначають подільник тактової частоти в частоту АЦП. Частота АЦП не може перевищувати 8.38MHz CK1 CK0 Подільник 0 0 32 0 1 4 1 0 8 1 1 2
4	CK0	
3	AQ1	
2	AQ0	
3	AQ1	Визначають кількість періодів АЦП, що витрачаються на захоплення сигналу пристроєм вибірки та зберігання (апертурний час). Для високоомних джерел сигналу значення мають збільшуватися. AQ1 AQ0 Кількість періодів 0 0 1 0 1 2 1 0 3 1 1 4
2	AQ0	
1	T2C	
0	EXC	
1	T2C	Встановлений біт дозволяє запуск АЦП при переповненні таймера 2
0	EXC	Встановлений біт дозволяє запуск АЦП при низькому рівні на лінії CONVST

Молодша тетрада регістру ADCCON2 визначає обраний канал перетворення, старша керує запуском АЦП. Адреса регістру – 0xD8, початкове значення – 0x00, побітова адресація підтримується.

Таблиця 6.2 – Біти регістру ADCCON2

Біт	Назва	Призначення
7	ADCI	Переривання АЦП. Встановлюється контролером вкінці єдиного АЦП перетворення або вкінці блоку перетворень в режимі DMA
6	DMA	Вмикає режим DMA
5	CCONV	Запускає режим послідовних перетворень. Після закінчення кожного перетворення розпочинається нове, доти, доки не користувач не скине біт DMA.
4	SCONV	Запускає одне перетворення. Після його завершення біт автоматично скидається
3	CS3	Визначає активний канал АЦП CS3 CS2 CS1 CS0 Канал 0 0 0 0 ADC0
2	CS2	
1	CS1	
0	CS0	
		
		0 1 1 1 ADC7
		1 0 0 0 Датчик температури (не менше 1мкс для захоплення)
		1 0 0 1 DAC0 (ЦАП 0)
		1 0 1 0 DAC1 (ЦАП 1)
		1 0 1 1 AGND (аналоговий спільний)
		1 1 0 0 VREF (опорна напруга)

Регістр ADCCON3 керує процедурою калібрування і не розглядається в даній лабораторній роботі.

Перед тим, як запускати аналогово-цифрове перетворення слід ввімкнути АЦП – записати в регістр ADCCON1 байт ініціалізації. Наприклад ввімкнемо АЦП з такими параметрами:

Опора – внутрішня (EXT_REF=0)

Подільник – 2 (CK1=1 CK0=1) ($f_{ADC}=f_{CLK}/2=5.5299\text{МГц}$)

Апертура – 4 періоди (AQ1=1 AQ0=1)

Запуск від спрацювання таймера 2 та зовнішнього сигналу заборонені (T2C=0 EXC=0). Таким чином байт ініціалізації рівний $10111100_2 = 188_{10} = \text{BC}_{16}$. Записуємо його в регістр ADCCON1.

Далі, щоб провести одиничне вимірювання, потрібно записати в ADCCON2 номер каналу і біти запуску. Так для запуску одиничного перетворення з 7 каналу в ADCCON2 слід записати байт 0x17.

Після запуску дані будуть готові через 17-20 тактів АЦП (в залежності від апертурного часу, встановленого бітами AQ1-AQ0). Якщо подільник частоти АЦП рівний двом (біти СК1=1 СК0=1), то це складе 34-40 тактів мікроконтролера. Взяти точний час завершення аналогово-цифрового перетворення можна опитуючи біт ADCI – в кінці циклу перетворення контролер встановить його.

Результат перетворення знаходитиметься в регістрах ADCDATAH:ADCDATAL.

Приклад програми для лабораторної роботи №6

Під час руху джойстика запалюється точка на матриці світлодіодів.

```
ADCCON1  DATA  0EFH  ;ADC CONTROL          cjne R5, #00000000b, vbik1
ADCCON2  DATA  0D8H  ;ADC CONTROL          cjne R7, #00000000b, vboky
ADCDATAL DATA  0D9H  ;ADC DATA LOW BYTE   mov dat1, #00000001b
ADCDATAH DATA  0DAH  ;ADC DATA HIGH BYTE  jmp vbokend
ADCI     BIT     0DFH  ;ADCCON2.7 - ADC INTURRUPT
FLAG
SCONV    BIT     0DCH  ;ADCCON2.4 - SINGLE
CONVERSION ENABLE
;DACCON  DATA  0FDH  ;DAC CONTROL REGISTER
;CFG841  DATA  0AFH  ;GENERAL FLASH/PWM
CONTROL REGISTER

dat      EQU     R0
adr      EQU     R1
Temp1    EQU     R2
Temp2    EQU     R3
dat1     EQU     R4
Segm     EQU     0x02
CSEG
ORG 0x0000
jmp begin
ORG 0x0033
begin:
    mov pl, 0xFF
    mov ADCCON1, #0ECh
    call off
    mov TCON, #11111111b
    mov TMOD, #11001100b

beg1:
    call diagonal
    jmp beg1

vboky:
vbik1:
    call rezult1
    cjne R5, #00000000b, vbik2
    cjne R7, #00001110b, vboky
    mov dat1, #00010000b
    jmp vbokend

vbik2:
    call rezult1
    cjne R5, #01001001b, vbik3
    cjne R7, #00001001b, vboky
    mov dat1, #00001000b
    jmp vbokend

vbik3:
    call rezult1
    cjne R5, #00001000b, vbik4
    cjne R7, #00000111b, vboky
    mov dat1, #00000100b
    jmp vbokend

vbik4:
    call rezult1
    cjne R5, #10010010b, vbik5
    cjne R7, #00000100b, vboky
    mov dat1, #00000010b
    jmp vbokend

vbik5:
    call rezult1

    mov ADCCON2, #6h
    clr ADCI

vbokend:
ret
diagonal:
; JB p3.2, diagonal
; call vboky
run:
kavalok1:
    call rezult2
    cjne R5, #00000000b, kavalok2
    cjne R6, #00000000b, run
    call vboky
    call lampa1

kavalok2:
    call rezult2
    cjne R5, #01001001b, kavalok3
    cjne R6, #00000010b, run
    call vboky
    call lampa2

kavalok3:
    call rezult2
    cjne R5, #10010010b, kavalok4
    cjne R6, #00000100b, run
    call vboky
    call lampa3

kavalok4:
    call rezult2
    cjne R5, #00001000b, kavalok5
    cjne R6, #00000111b, run
    call vboky
    call lampa4

kavalok5:
    call rezult2
    cjne R5, #01001001b, kavalok6
    cjne R6, #00001001b, run
    call vboky
    call lampa5

kavalok6:
    call rezult2
    cjne R5, #01110111b, kavalok7
    cjne R6, #00001010b, run
    call vboky
    call lampa6

kavalok7:
    call rezult2
    cjne R5, #00000000b, kavalok1
    cjne R6, #00001110b, run
    call vboky
    call lampa7

jmp run
ret
rezult1:
    mov ADCCON2, #6h
    clr ADCI
```

```

    setb SCONV
    jnb ADCl,$
    mov b, adcdatal
    mov a, adcdatab
    clr acc.7
    clr acc.6
    clr acc.5
    clr acc.4
    mov Tl0, b
    mov TH0, a
    mov R5, Tl0
    mov R7, Th0
ret
result2:
    mov ADCCON2, #7h
    clr ADCl
    setb SCONV
    jnb ADCl,$
    mov b, adcdatal
    mov a, adcdatab
    clr acc.7
    clr acc.6
    clr acc.5
    clr acc.4
    mov Tl1, b
    mov TH1, a
    mov R5, Tl1
    mov R6, Th1
ret
lampa7:
    mov dat, #10111111b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa6:
    mov dat, #11011111b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa5:
    mov dat, #11101111b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa4:
    mov dat, #11110111b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa3:
    mov dat, #11111011b
    mov adr, #00000101b

```

```

    call strob
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa2:
    mov dat, #11111101b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
lampa1:
    mov dat, #11111110b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000100b
    mov adr, #00000110b
    call strobl
ret
off:
    mov dat, #11111111b
    mov adr, #00000101b
    call strob
;
    mov dat, #00000000b
    mov adr, #00000110b
    call strobl
ret
strob:
    setb P3.6
    mov P0, dat
    mov P2, adr
    mov P2, #0x00
ret
strobl:
    setb P3.6
    mov P0, dat1
    mov P2, adr
    mov P2, #0x00
ret
Delay:
    mov Temp1, #0xFF
loop:
    dec Temp1
    mov Temp2, #0xFF
    djnz Temp2, $
    cjne Temp1, #0x00, loop
ret
Ldelay:
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
    call delay
RET
END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	За допомогою джойстика керувати швидкістю переміщення точки по світлодіодній лінійці. Швидкість переміщення визначається положенням ручки джойстика відносно вертикальної осі, нижнє положення відповідає максимальній швидкості (8 сегментів за 1с.), верхнє положення мінімальна швидкість переміщення засвіченого світло діода (1 сегмент за 1с.). Початковий стан: засвічений світлодіод переміщаєть по лінійці в напрямку знизу вверху з швидкістю відповідно до положення ручки джойстика.
2	За допомогою джойстика відповідно до положення його ручки по горизонтальній осі

	засвічувати світлодіоди. При встановленні ручки джойстика в ліве крайнє положення світлодіодна лінійка погашена, при встановленні ручки джойстика в праве крайнє положення засвічено всі світлодіоди. Початковий стан: засвічена кількість світлодіодів відповідно до положення ручки джойстика.
3	Відображати на статичному семисегментному індикаторі код від перетворення АЦП в десятковому форматі від каналу до якого під'єднано горизонтальну вісь джойстика.
4	Відображати на статичному семисегментному індикаторі код від перетворення АЦП в десятковому форматі від каналу до якого під'єднано вертикальну вісь джойстика.
5	На статичному індикаторі реалізувати біжучу стрічку, на якій по черзі засвічуються символи на 1, 2, 3 і 4 знаковість відповідно починаючи з лівого, виводиться ("2","0","1","1"), зміна символів визначається положенням ручки джойстика відносно горизонтальної осі, ліве крайнє положення – символи змінюються з частотою 1 символ за 1с., крайнє праве положення – символи змінюються з частотою 8 символів за 1с. Початковий стан: символи змінюються з швидкістю відповідно до положення ручки джойстика.
6	На статичному індикаторі реалізувати біжучу стрічку, на якій по черзі засвічуються символи на 4, 3, 2 і 1 знаковість відповідно починаючи з правого, виводиться ("1","1","0","2"), зміна символів визначається положенням ручки джойстика відносно вертикальної осі, верхнє крайнє положення – символи змінюються з частотою 1 символ за 1с., крайнє нижнє – положення символи змінюються з частотою 8 символів за 1с. Початковий стан: символи змінюються з швидкістю відповідно до положення ручки джойстика.
7	На матричному індикаторі засвічувати з певним періодом символ "*" . Тривалість світіння символу залежить від положення ручки джойстика відносно вертикальної осі. Верхнє крайнє положення – час відображення символу 1с., інтервал погашеного індикатора 1с., крайнє нижнє – час відображення символу 8с., інтервал погашеного індикатора 1с. початковий стан: символ відображається з частотою відповідно до положення ручки джойстика.
8	На матричному індикаторі засвічувати з певним періодом символ "#". Тривалість світіння символу залежить від положення ручки джойстика відносно горизонтальної осі. Праве крайнє положення – час відображення символу 1с., інтервал погашеного індикатора 1с., крайнє ліве – час відображення символу 8с., інтервал погашеного індикатора 8с. початковий стан: символ відображається з частотою відповідно до положення ручки джойстика.
9	На матричному індикаторі засвічувати певну кількість рядків відповідну до положення ручки джойстика по вертикальній осі. Верхнє крайнє положення – засвічено всі рядки матричного індикатора, нижнє крайнє положення – весь матричний індикатор погашено. Початковий стан: засвічено певну кількість рядків відповідно до положення ручки джойстика.
10	На матричному індикаторі засвічувати певну кількість стовпців відповідну до положення ручки джойстика по горизонтальній осі. Ліве крайнє положення – засвічено всі стовпці матричного індикатора, праве крайнє положення – весь матричний індикатор погашено. Початковий стан: засвічено певну кількість стовпців відповідно до положення ручки джойстика.

Тема: Проектування системи керування кроковим двигуном.

Порядок виконання роботи:

1. Вивчити режими роботи уніполярного крокового двигуна (повнокроковий, півкроковий, мікрокроковий) та алгоритми задання їх.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконатись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження схеми в стенд.

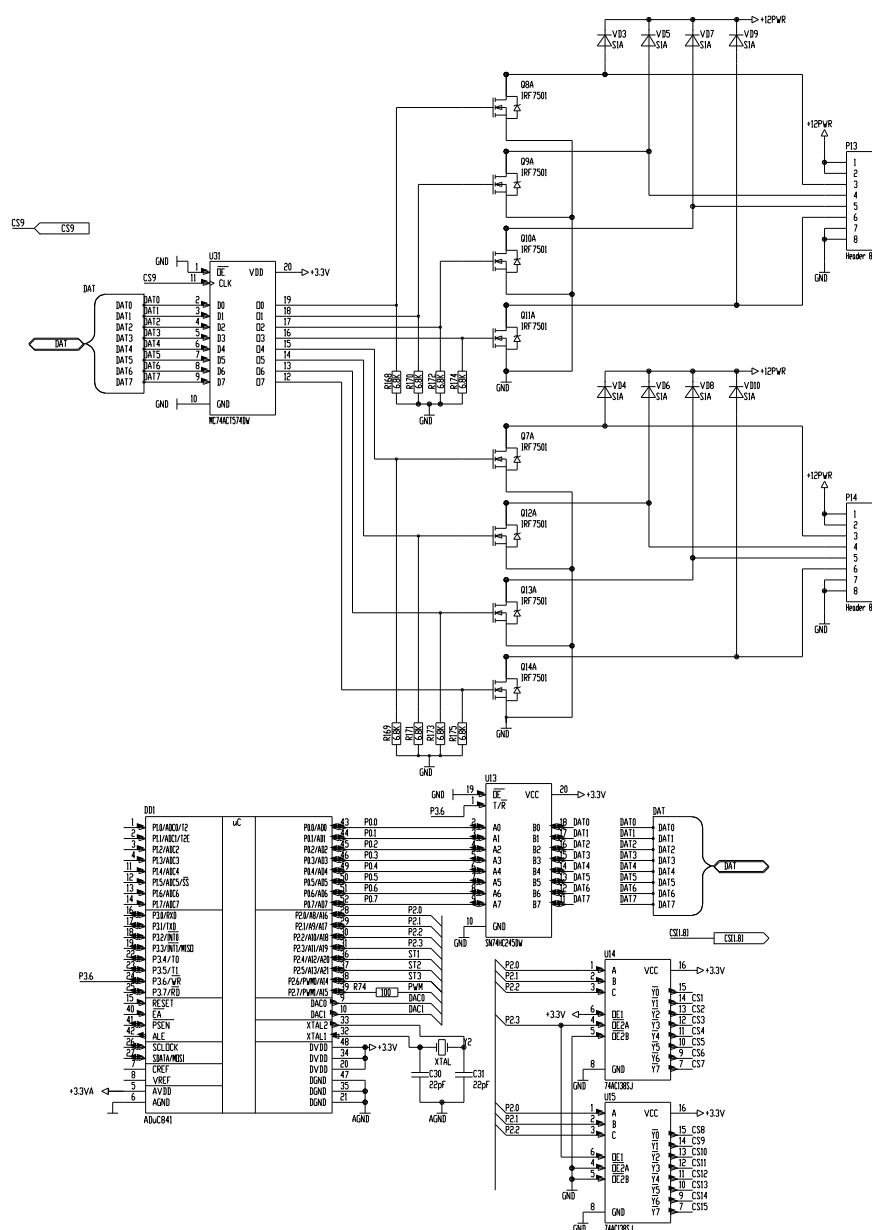


Рисунок 7.1 – Схема для лабораторної роботи №7

Короткі теоретичні відомості

Для керування кроковими двигунами на стенді передбачено додатковий регістр, виходи якого можуть комутувати 8 ключів, зібраних на МОН-транзисторах типу IRF7501. Транзистори дозволяють переключати струм до 2,4А при температурі 25°C або 1,9А при температурі до 70°C. Опір каналу ключа в увімкненому стані приблизно становить 0,135 Ом. Напруга стік-витік до 20В, що цілком достатньо для керування двигуном типу ДШ-200 або аналогічних.

Діоди, увімкнені між стоками (drains) і додатною напругою живлення необхідні для захисту ключів від напруг само-е.р.с., які наводяться в обмотках двигуна під час переключення напруги. Контакти роз'ємів типу WH-8 увімкнено паралельно для збільшення надійності з'єднання і навантажувальної здатності.

Слід пам'ятати, що навіть якщо двигун не обертається, а на одну із обмоток подана напруга, то він споживає значний струм, і існує можливість перегрівання як двигуна, так і блока живлення. Тому якщо використовувати цей режим роботи двигуна (режим утримання), необхідно забезпечити, щоби споживаний струм був у межах норми для даного типу двигуна і не лишати його в такому режимі протягом тривалого часу.

Частоту обертання двигуна можна регулювати як зміною періоду переривання таймера, так і зміною кількості переповнень таймера на один крок. В даній програмі було використано останній спосіб, однак для забезпечення плавного регулювання частоти потрібно використовувати перший, або їх комбінацію.

Приклад програми для лабораторної роботи №8

Скласти програму керування кроковим двигуном. Режим роботи двигуна визначається натисканням кнопок першого рядка клавіатури (наприклад, перша кнопка – пуск/стоп, друга – реверс, третя – збільшення швидкості обертання, четверта – зменшення).

Алгоритм роботи драйвера:

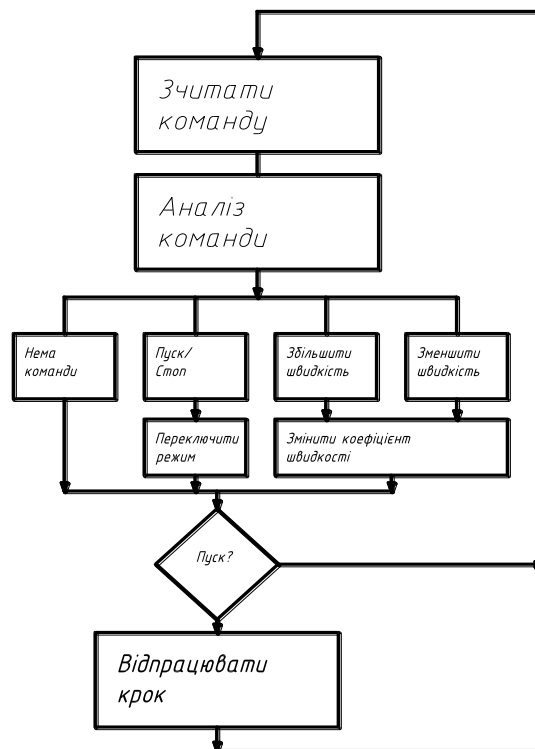


Рисунок 7.2 – Алгоритм основного циклу програми керування кроковим двигуном (обробка переривання таймера 0)

```

sbit  F1          = 0xD1          //Declare flag F1

DAT      equ      R0          //Data transfer register
ADDR     equ      R1          //Address transfer register
ST_CNT   equ      R2          //Steps counter
PS_CNT   equ      R3          //step to step pause

V_cnt    equ      R4
Flags_a  equ      0x004

C_TL0    EQU      0FFh
C_TH0    EQU      00Ah          //07FFh - max stab. vel. of rotation
SM_addr  EQU      009h          //Step motor address

Step_11  EQU      000000001b
Step_12  EQU      000000100b
Step_13  EQU      000000010b
Step_14  EQU      000001000b
Step_21  EQU      000000101b
Step_22  EQU      000000110b
Step_23  EQU      000001010b
Step_24  EQU      000001001b

Pause    EQU      003h

ORG      0x0000
JMP      RESET

ORG      0x000B
JMP      Tim0_ovf

ORG      0x0030
RESET:
    MOV     ST_CNT, #001h //Step 1
    MOV     PS_CNT, #Pause //pause ...
    MOV     V_cnt, #Pause //program flags
    MOV     DAT, #0FFh
    MOV     ADDR, #007h
    CALL    write
    MOV     DAT, #000h
    MOV     ADDR, #SM_addr
    CALL    write
    CALL    Timer0_init
wait:
    JMP     wait

Timer0_init:
    MOV     TH0, #C_TH0          //1,0XX ms
    MOV     TL0, #C_TL0
    MOV     TMOD, #001h
    MOV     TCON, #010h //Run timer
    MOV     IE, #010000010b
    //Enable global interrupt and Timer0 interrupt
    MOV     IP, #000000000b
    //interrupt priority
    RET
;*****
; Step subroutine
Tim0_ovf:
    MOV     TH0, #C_TH0          //1,0XX ms
    MOV     TL0, #C_TL0          //Reload timer 0

    CALL    Read_row1
    MOV     A, DAT
    JB      ACC.4, RUN
    JB      ACC.5, Revers
    JB      ACC.6, V_inc
    JB      ACC.7, V_dec

    JMP     M_CNTRL
RUN:      //starting and feet by turns
    CPL     F0
    JMP     M_CNTRL
Revers:

    JMP     M_CNTRL
V_inc:
    INC     V_cnt
    CJNE    V_cnt, #000h, M_CNTRL

```

```

        DEC V_cnt
        JMP M_CNTRL
V_dec:
        DEC V_cnt
        MOV A, V_cnt
        SUBB A, #003h
        JNC M_CNTRL
        MOV V_cnt, #003h
        JMP M_CNTRL
M_CNTRL:
        DEC PS_CNT          //decrement step to step pause counter
        CJNE PS_CNT, #000h, E_T0_ovf
        // exit from interrupt
        // if pause is full
        MOV A, V_cnt
        MOV PS_CNT, A          //reload step to step pause counter
        JNB F0, E_T0_ovf
step1:
        CJNE ST_CNT, #001h, step2
        MOV DAT, #Step_11
        MOV ADDR, #SM_addr
        CALL Write
        MOV ST_CNT, #002h
        RETI
step2:
        CJNE ST_CNT, #002h, step3
        MOV DAT, #Step_12
        MOV ADDR, #SM_addr
        CALL Write
        MOV ST_CNT, #003h
        RETI
step3:
        CJNE ST_CNT, #003h, step4
        MOV DAT, #Step_13
        MOV ADDR, #SM_addr
        CALL Write
        MOV ST_CNT, #004h
        RETI
step4:
        CJNE ST_CNT, #004h, step1
        MOV DAT, #Step_14
        MOV ADDR, #SM_addr
        CALL Write
        MOV ST_CNT, #001h
        RETI

E_T0_ovf:
        // mov zero to step motor latch
        MOV DAT, #000h
        MOV ADDR, #SM_addr
        CALL write
        RETI
;*****
Read_row1:
        MOV P1, #000h
        MOV P2, #000010000b
        NOP
        NOP
        MOV A, P1
        ANL A, #0F0h
        MOV Dat, A
//Check old-push button
        JB F1, RR1
        CJNE DAT, #000h, RR2
        JMP RRender
RR1:
        CJNE DAT, #000h, RR3
        CLR F1
        JMP RRender
RR2:
        SETB F1
        JMP RRender
RR3:
        MOV DAT, #000h
        JMP RRender
RRender:
RET
write: setb P3.6
        mov P0, DAT //moving data to bus buffer
        mov P2, ADDR //set peripherals address
        nop

```

```

nop
mov P2, #000h      //clock

ret

END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою джойстика при переміщенні його по вертикальній осі і дискретної кнопки. При встановленні ручки джойстика у крайнє нижнє положення двигун зупиняється, при встановленні ручки у верхнє крайнє положення швидкість обертання двигунв максимальна, дискретна кнопка визначає напрямок обертання вала двигуна, при кожному натисканні відбувається реверс. Початковий стан: двигун обертається за годинниковою стрілкою відповідно до положення ручки джойстика
2	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою дискретних кнопок, які знаходяться на джойстику і енкодері. При натисканні кнопки на енкодері швидкість збільшується на один, при досягненні максимальної швидкості подальше натискання на кнопку не змінює швидкості. При натисканні на кнопку на джойстику швидкість зменшується на один, при досягненні нульової швидкості двигун зупиняється, подальше натискання на дану кнопку не приводить до змін. Поточну швидкість відображати на молодшому розряді статичного семисегментного індикатора. Початковий стан: двигун зупинено, на індикаторі відображається поточна швидкість.
3	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою першого стовпця клавіатури наступним чином. Кнопка з цифрою "1" збільшує швидкість до максимуму , кнопка з цифрою "4" міняє напрямок руху на протилежний, кнопка з цифрою "7" зменшує поточну швидкість до мінімуму, кнопка "#" зупиняє двигун, при цьому вмикається режим утримання. На статичному семисегментному індикаторі відображати поточний стан двигуна (реверс – "R", нормальний режим – "F", двигун зупинено – "S"). Початковий стан: двигун зупинено, на індикаторі відображено поточний стан двигуна.
4	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою джойстика при переміщенні його по горизонтальній осі. Центральне положення – двигун зупинено, при зсуванні ручки вліво, двигун обертається за годинниковою стрілкою, швидкість залежить від положення ручки джойстика, при досягненні крайнього лівого положення – швидкість максимальна. При зсуванні ручки право, двигун обертається в напрямку проти годинникової стрілки, швидкість залежить від положення ручки джойстика, при досягненні крайнього правого положення – швидкість максимальна.
5	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою кнопок клавіатури. Натискання кнопки "#" – збільшує швидкість обертання двигуна в напрямку за годинниковою стрілкою до максимуму за 3с. (якщо до цього моменту двигун був зупинений, або обертася в іншомунапрямі), натискання кнопки "*" – збільшує швидкість обертання двигуна в напрямку проти годинникової стрілки до максимуму за 3с. (якщо до цього моменту двигун був зупинений, або обертася в іншомунапрямі), Натискання кнопки "0" – зупиняє двигун.
6	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою кнопок клавіатури. Кожна кнопка на якій зображено цифру вказує час розгону і тормозіння двигуна в секундах (кнопка "0" неактивна). Двигун за певний час виходить на максимальну швидкість і також за певний час приторможує до повної зупинки, даліше знову починає розгін.
7	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою джойстика при переміщенні його по вертикальній. Для даної задачі використовувати джойстика, як трьохпозиційний перемикач, тобто при встановленні ручки джойстика в центральнє положення двигун – зупинено, при встановленні в крайнє нижнє положення двигун починає обертатися в нарядку за годинниковою стрілкою, при цьому швидкість зростає з

	збільшенням часу утримування джойстика в даному положенні. При встановленні в крайнє верхнє положення двигун починає обертатися в напрямку проти годинникової стрілки, при цьому швидкість зростає з збільшенням часу утримування джойстика в даному положенні.
8	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою дискретних кнопок. Так натискання кнопки на енкодері двигун зупиняє і при натисканні кнопки на джойстику двигун обертається в протилежному напрямку як до моменту натискання кнопки на енкодері збільшує поточну швидкість, при досягненні максимуму кнопка ніяких змін не вносить. Початковий стан: двигун зупинено.
9	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою кнопок клавіатури і відобразити задану швидкість (у відносних одиницях) на молодшому розряді статичного семисегментного індикатора. Кнопка “#” зменшує швидкість на один розряд, кнопка “*” збільшує швидкість на один розряд, кнопка “0” зупиняє двигун, подальший пуск двигуна здійснюється за допомогою натискання кнопки “*”. Початковий стан: двигун зупинено, на індикаторі відображається поточна швидкість.
10	Керувати кроковим двигуном, який під'єднаний до роз'єму ... за допомогою джойстика при переміщенні його по вертикальній . Двигун обертається в напрямку за годинниковою стрілкою, нижнє положення ручки джойстика – максимальна швидкість, верхнє положення – двигун зупинено, на реші діапазону встановлення ручки джойстика встановлюється відповідне значення швидкості. При цьому на світлодіодній лінійці представлена швидкість обертання двигуна у вигляді “термометра”.

Лабораторна робота №8. Проектування системи керування двигуном постійного струму

Тема: Проектування системи керування двигуном постійного струму.

Мета: Вивчити взаємодію контролера із пристроями механічного приводу, зокрема керування двигуном постійного струму з використанням апаратного ШІМ-контролера.

Порядок виконання роботи:

1. Вивчити алгоритм керування двигуном постійного струму.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження схеми в стенд.

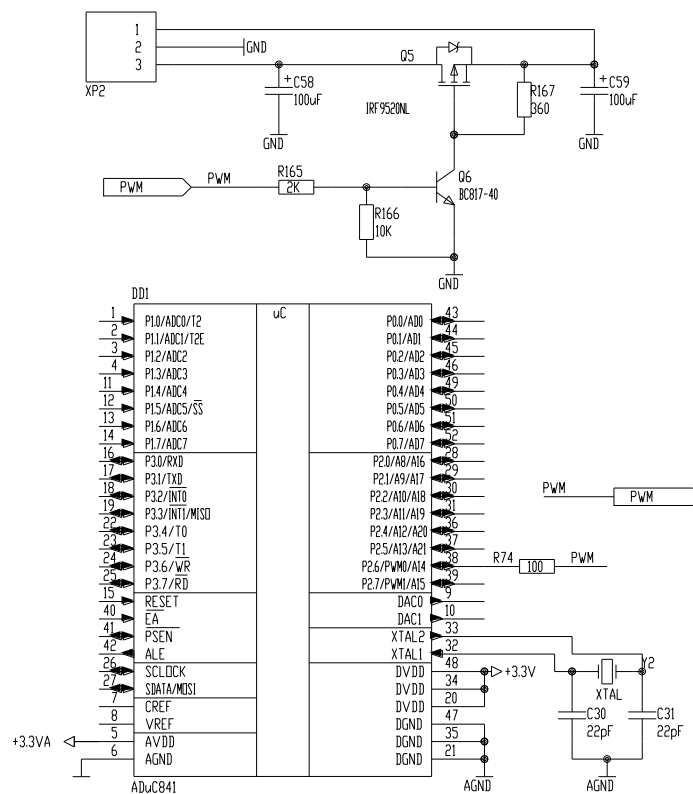


Рисунок 8.1 – Схема для лабораторної роботи №8

Короткі теоретичні відомості

Для керування двигуном постійного струму на стенді передбачений драйвер, який складається з ключа на МОН- транзисторі типу IRF9540, а також біполярного транзистора типу BC817 призначення якого забезпечити чітке перемикання керованого ключа. Дана схема керується виходом контролера Р2.6, який має можливість використання вбудованого апаратного ШІМ-контролера. База біполярного транзистора через високоомний резистор підтягнута до землі, що забезпечу надійне закривання ключа, при відсутньому сигналі керування.

Швидкість обертання двигуна постійного струмі залежить від величини напруги, яка подається йому на вхід. Розглянемо принцип роботи ШІМ модулятора, за допомогою якого будемо змінювати значення вхідної напруги на живлення електродвигуна.

Широтно-імпульсна модуляція (ШІМ) – наближення бажаного сигналу (багаторівневого чи неперервного) до дійсних бінарних сигналів (з двома рівнями – вкл./викл.), таким чином, що в середньому за певний проміжок часу їх значення рівні. Формально, це можна записати так:

$$\int_{t_1}^{t_2} x(t) dt = \sum A \cdot \Delta T_i$$

де: $x(t)$ – бажаний вхідний сигнал в межах від t_1 до t_2 ;

ΔT_i - тривалість i-го ШІМ імпульсу, амплітуда якого A ;

ΔT_i підбирається таким чином, щоб сумарні площі (енергії) обох величин були приблизно рівні за достатньо великий проміжок часу, а також рівне середнє значення за період.

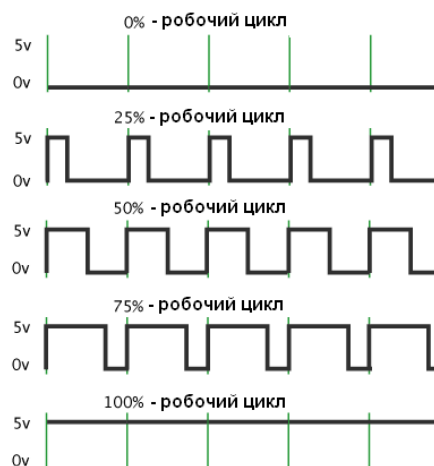


Рисунок 8.2 Зміна процентного заповнення ШІМ

ШІМ, який використовується на ADuC841 є досить гнучким для роботи і може бути налаштований для будь-якого із шести режимів експлуатації. Два з цих режимів дає можливість використовувати ШІМ як $\Sigma - \Delta$ ЦАП з дозволом до 16 біт. Функціональна схема роботи широтно-імпульсного модулятора приведена на рисунку 8.2, потрібно звернути увагу на те що тактування генератора може відбуватися по різному і наведено в таблиці 8.1.

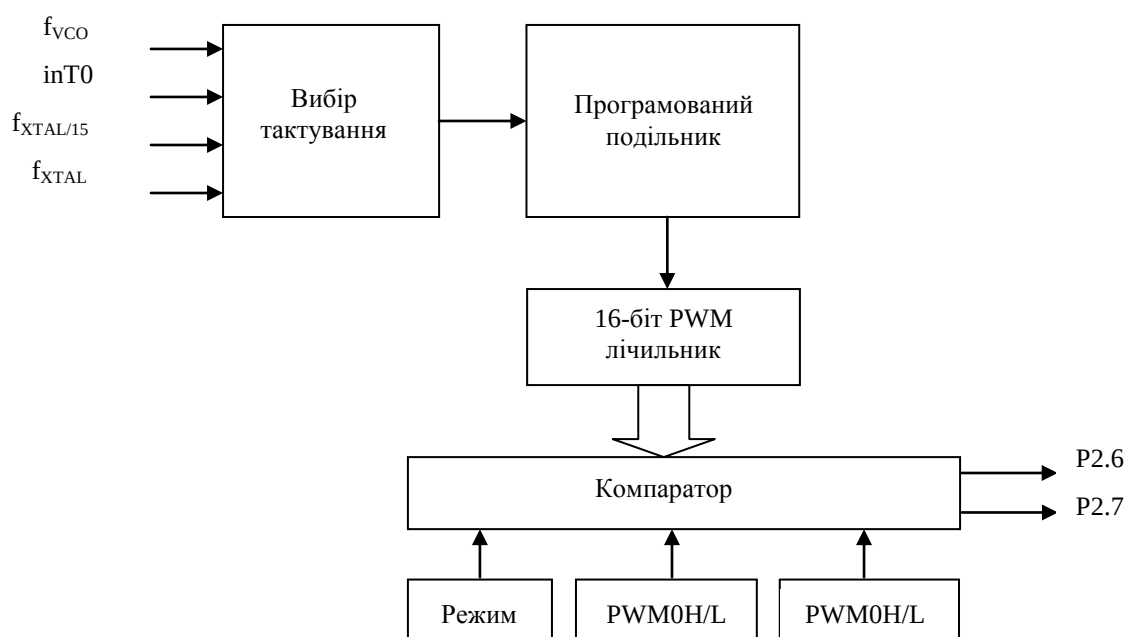


Рисунок 8.3 – Функціональна схема роботи ШІМ ADuC841

Для налаштування і роботи ШІМ використовується п'ять SFR регістрів – це: контроль SFR (PWMCON) і чотири регістри даних SFR (PWM0H, PWM0L, PWM1H і PWM1L).

PWMCON управляє різних режимах роботи ШІМ, а також вибором тактової частоти. PWM0H/L і PWM1H/L – регістрів даних, які задають тривалість періоду роботи ШІМ.

Таблиця 8.1 Біти регістру PWMCON

Біт	Назва	Призначення
7	SNGL	Відключає ШІМ вихід від порта P2.6 або P3.4 і назначає його для цифрового вводу/виводу
6	MD2	Біти режиму роботи ШІМ. MD2 MD1 MD0 Режим роботи ШІМ 0 0 0 0: ШІМ вимкнений 0 0 1 1: Одноканальний ШІМ з виводом на P2.7 або P3.3 0 1 0 2: Двоканальний спарений 8-бітний ШІМ 0 1 1 3: Двоканальний спарений 16-бітний ШІМ 1 0 0 4: Двоканальний NRZ 16-бітний Σ - Δ АЦП 1 0 1 5: Двоканальний 8-бітний ШІМ 1 1 0 6: Двоканальний RZ 16-бітний Σ - Δ АЦП 1 1 1 Не використовується
5	MD1	
4	MD0	
3	CDIV1	
2	CDIV0	
1	CSEL1	
0	CSEL0	
		Визначення тактового сигналу для лічильника ШІМ CDIV1 CDIV0 Значення 0 0 ШІМ лічильник = вибрана частота/1 0 1 ШІМ лічильник = вибрана частота/4 1 0 ШІМ лічильник = вибрана частота/16 1 1 ШІМ лічильник = вибрана частота/64
		Визначення тактового сигналу для ШІМ CSEL 1 CSEL 0 Значення 0 0 частота ШІМ = $f_{XTAL}/15$ 0 1 частота ШІМ = f_{XTAL} 1 0 частота ШІМ = вхідний сигнал на P3.4/T0 1 1 частота ШІМ = $f_{VCO} = f_{OSC}$

Розглянемо роботу ШІМ в 1 режимі. В даному режимі, як частота так і період роботи ШІМ програмується користувачем. Регістри PWM1H/L встановлюють частоту роботи ШІМ, а PWM0H/L здійснюють задання тривалості періоду роботи (рисунок 8.3). Наприклад запис в PWM1H/L числа 65536 дає 16-бітний ШІМ з частотою роботи 168 Гц (при тактовій частоті кварцу 11,0592МГц).

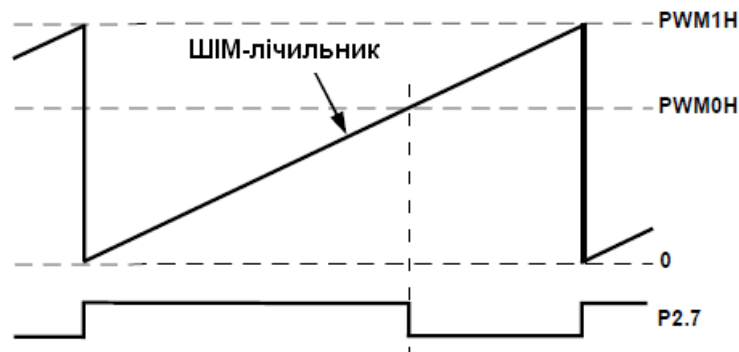


Рисунок 8.4 – ШІМ в режимі 1

Таким чином в даному режимі є можливість налаштувати ШІМ на різну кількість біт, при цьому частота буде визначатися за формулою:

$$f_{PWM} = f_{XTAL} / 2^n$$

де: f_{PWM} – частота роботи ШІМ (Гц);
 f_{XTAL} – частота тактування ШІМ (Гц);
 n – розрядність ШІМ (біт).

Приклад програми для лабораторної роботи №8

Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. Швидкість встановлюється відповідно до положення ручки джойстика, верхнє положення – мінімальна швидкість, нижнє положення – максимальна швидкість, 1 режим роботи, 12- бітний ШІМ, частота 675Гц.

```
$include (mod841)

org 0
JMP Start          ;перехід на мітку Start

ORG 030H

Start:
call Init_PWM      ;виклик ініціалізації ШІМ
call ADCInit       ;виклик ініціалізації АЦП

mov PWM1H, #00fh   ;встановлення розрядності ШІМ 12-біт
mov PWM1L, #0ffh
mov PWM0H, #007h   ;початкове значення ШІМ (50%)
mov PWM0L, #0ffh

BEGIN:             ;мітка початку основного циклу програми
call Measurev      ;виклик процедури вимірювання АЦП
mov PWM0H,R7        ;запис в старший байт ШІМ значення тетради старшого байту з АЦП
mov PWM0L,R5        ;запис в молодший байт ШІМ значення молодшого байту з АЦП

JMP BEGIN          ;перехід на початок циклу

Init_PWM:          ;ініціалізація ШІМ
mov PWMCN, #00010111b ;1-режим, дільник робочої частоти на 4, тактування від кварца контролера
RET

ADCInit:           ;ініціалізація АЦП
mov ADCCON1, #10111100b
ret

Measurev:          ;вимірювання
;In: -
;Out: R7:R6 -- ADC result
;Alters: a, PSW
mov ADCCON2, #00010110b
jnb ADCL, $
mov a, ADCDATAH
anl a, #0fh
mov R7, a
mov R6, ADCDATAH
ret

END
```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою другого рядка матричної клавіатури встановлюються наступні режими роботи двигуна: натискання клавіші “2” – встановлення максимальної швидкості, натискання клавіші “5” – встановлення середнього значення швидкості, натискання клавіші “8” – встановлення мінімальної швидкості, натискання клавіші “0” – зупинка двигуна. На світлодіодній лінійці відображати задану швидкість у вигляді “термометра”. 1 режим роботи, 8- бітний ШІМ,

	частота Початковий стан: двигун зупинено, на світлодіодній лінійці відображається поточне значення швидкості.
2	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою дискретних кнопок задавати швидкість обертання вала двигуна. Кнопка на енкадері збільшує значення швидкості на одиницю, при досягненні максимальної швидкості кнопка стає неактивною, кнопка на джойстику зменшує значення швидкості на одиницю, при встановленні швидкості 0, двигун зупиняється, подальше натискання на кнопку не приводить до змін, встановлене значення швидкості у (%) відображається на статичному семисегментному індикаторі. 1 режим роботи, 6- бітний ШІМ, частота Початковий стан: двигун зупинено, індикатор погашено.
3	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою джойстика, при переміщенні його по горизонтальній осі, задавати швидкість обертання вала двигуна. Крайнє праве положення – максимальна швидкість, крайнє ліве положення – мінімальна швидкість. На світлодіодній лінійці відображати задану швидкість у вигляді “термометра”. 1 режим роботи, 8- бітний ШІМ, частота Початковий стан: вал двигуна обертається відповідно до положення ручки на джойстику, на світлодіодній лінійці відображається встановлене значення швидкості.
4	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою копек матричної клавіатури задавати наступні режими роботи двигуна. Натискання кнопки “#” – збільшує на одиницю швидкість, натискання кнопки “*” – зменшує на одиницю швидкість, натискання кнопки “0” – зупиняє двигун, при досягненні крайніх значень при заданні швидкості відповідні кнопки стають неактивними. 1 режим роботи, 3- бітний ШІМ, частота
5	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою енкадера задавати швидкість обертання вала двигуна. При обертанні ручки енкадері в напрямку за годинниковою стрілкою – швидкість збільшується на одиницю, при обертанні ручки енкадера в напрямку проти годинникової стрілки швидкість зменшується на одиницю. При досягненні крайніх значень при заданні швидкості, повертання ручки енкадері в відповідному напрямі ігнорується. 1 режим роботи, 4- бітний ШІМ, частота Початковий стан: двигун зупинено.
6	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою кнопок матричної клавіатури задавати час розгону двигуна до максимальної швидкості, при досягненні максимальної швидкості двигун зупинити на 10с., і продовжувати процедуру спочатку. Кожна кнопка відповідає часу розгону в десятках секунд. “1” – 10с., “2” – 20с., і т.д., кнопки “#”, “0”, “*” – неактивні. 1 режим роботи, 8- бітний ШІМ, частота Початковий стан: встановлено час розгону двигуна 10с.
7	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою кнопок матричної клавіатури задавати швидкість обертання вала двигуна, кожна кнопка відповідає десятку процентів від максимальної швидкості “1” – 10%, “2” – 20%, і т.д., кнопка “0” – 100%, кнопка “*” – зупинка двигуна. Встановлене значення швидкості у (%) відображається на статичному семисегментному індикаторі . 1 режим роботи, 6- бітний ШІМ, частота Початковий стан: двигун зупинено.
8	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою джойстика, при переміщенні його по горизонтальній осі, задавати швидкість обертання вала двигуна. Крайнє ліве положення – максимальна швидкість, крайнє праве положення – мінімальна швидкість. На світлодіодній лінійці запустити біжучу точку, швидкість переміщення залежить від заданої швидкості (мінімальна швидкість – 0,5 сегмента за 1с., максимальна швидкість –

	8 сегментів за секунду). 1 режим роботи, 4- бітний ШІМ, частота Початковий стан: вал двигуна обертається відповідно до положення ручки на джойстику, біжуча точка переміщується відповідно до встановленого значення швидкості.
9	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою енкодера задавати швидкість обертання вала двигуна. При обертанні ручки енкодері в напрямку проти годинникової стрілки – швидкість збільшується на одиницю, при обертанні ручки енкодера в напрямку за годинниковою стрілкою швидкість зменшується на одиницю. При досягненні крайніх значень при заданні швидкості, повертання ручки енкодері в відповідному напрямі ігнорується. 1 режим роботи, 5 - бітний ШІМ, частота Початковий стан: двигун зупинено.
10	Керувати швидкістю обертання двигуна постійного струму використовуючи вбудований апаратний ШІМ-контролер. За допомогою дискретних кнопок задавати швидкість обертання вала двигуна. Кнопка на енкодері зменшує значення швидкості на одиницю, при досягненні мінімальної швидкості кнопка стає неактивною, кнопка на джойстику збільшує значення швидкості на одиницю, при встановленні максимальної швидкості подальше натискання на кнопку не приводить до змін, встановлене значення швидкості у вигляді “термометра” відображається на світлодіодній лінійці, заповнення лінійки відбувається знизу вгору. . 1 режим роботи, 3- бітний ШІМ, частота Початковий стан: двигун зупинено, індикатор погашено.

Лабораторна робота №9. Програмування послідовного порту, інтерфейс RS-232

Тема: Програмування послідовного порту, інтерфейс RS-232.

Мета: Вивчити принцип передачі даних по інтерфейсу RS-232, зокрема ознайомитися з режимами роботи послідовного інтерфейсу, здійснити налаштування прийому/передачі даних на різних режимах

Порядок виконання роботи:

1. Вивчити принцип роботи послідовного інтерфейсу RS-232.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження схеми в стенд.

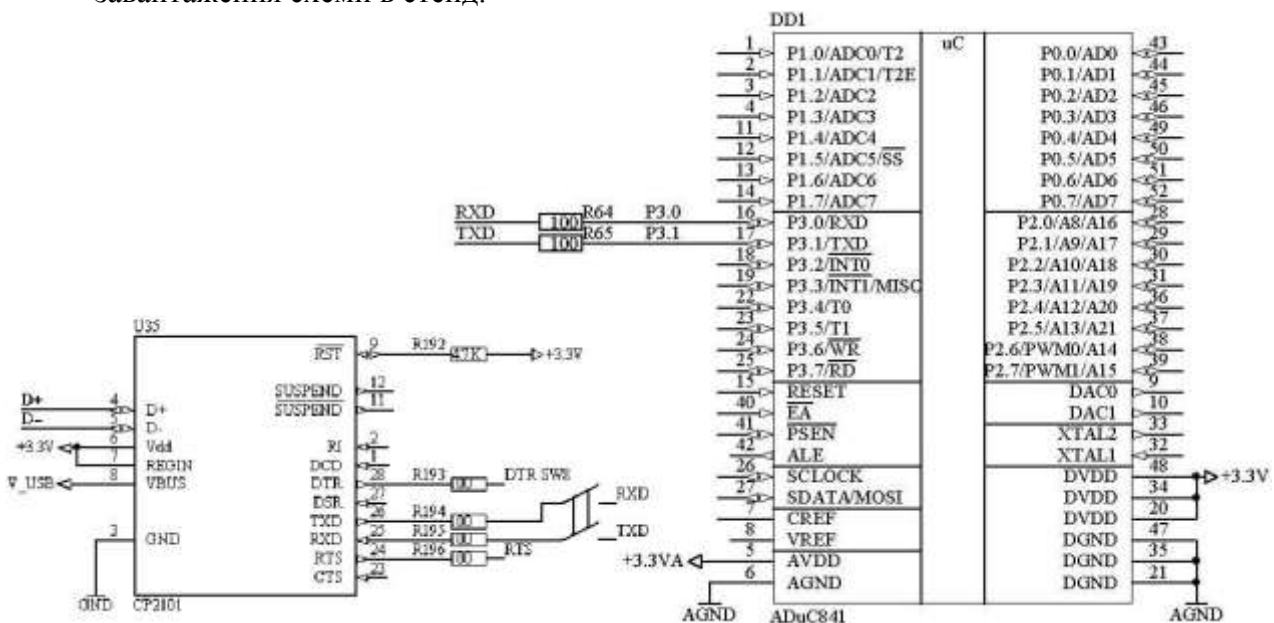


Рисунок 9.1 – Схема для лабораторної роботи №9

Короткі теоретичні відомості

Універсальний асинхронний приймач-передавач (UART – Universal Asynchronous Receiver-Transmitter) – периферійний пристрій мікроконтролера, який дозволяє послідовно передавати та приймати дані по двох однопровідних лініях. UART має можливість працювати в режимі повного дуплексу і не використовувати при цьому додаткової лінії для синхронізації.

Часто UART застосовують у парі з інтерфейсом RS-232 для зв'язку периферійного пристрою з персональним комп'ютером. Сигнали UART та RS-232 відрізняються в основному рівнями логічної одиниці та нуля. Якщо UART використовує рівні стандартні для КМОП чи TTL логіки, то стандарт RS-232 передбачає використання напруги від -3В до -25В для кодування логічної одиниці та напруги від 3В до 25В для кодування логічного нуля.

Передавання даних відбувається пакетами по 8 або 9 біт (молодші біти передаються першими). Синхронізація приймача та передавача відбувається таким чином (рис. 9.2). У спокої лінія утримується передавачем у стані логічної одиниці. Якщо виникає необхідність передати дані то передавач передає нульовий старт біт, вісім або дев'ять інформаційних бітів, та обов'язково одиничний стоп-біт, що повертає лінію в стан очікування. Дев'ятий інформаційний біт можна використати для перевірки цілісності байта.

При прийманні посилання, приймач постійно прослуховує лінію RXD і в момент переходу з одиничного в нульовий стан (прийом старт-біта) внутрішній таймер прийому скидається в нуль, щоб синхронізувати його із таймером передавача. По синхронізованому внутрішньому таймеру контролер за період переривання кожного інформаційного біта тричі опитує лінію RXD, а у вхідний буфер записується значення, отримане шляхом “мажоритарного голосування” – записується значення, отримане принаймні в двох вимірах із трьох. Після прийому всіх інформаційних біт та стоп-біта приймач переходить у стан очікування нового старт-біта.

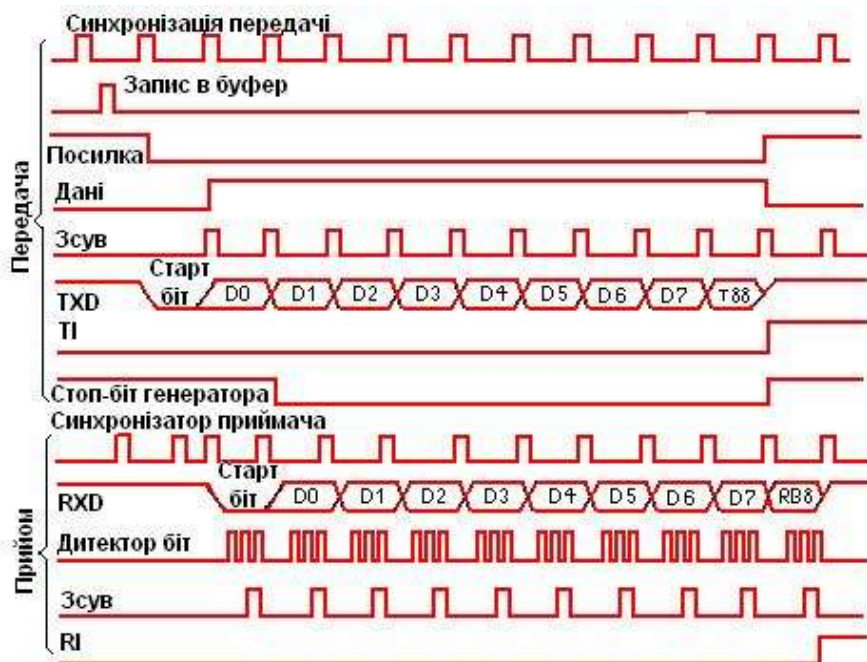


Рисунок 9.2 – Часові діаграми роботи UART

Для успішного обміну приймач і передавач має бути налаштованим на одну і ту ж швидкість передавання та довжину посилання. Допустиме відхилення частоти передавача та приймача визначається з умов недопущення роз синхронізації за час передачі посилання і не може перевищувати 3%.

UART сімейства мікроконтролерів MCS51 (до якого відноситься ADuC841) передбачає роботу в чотирьох режимах, які вибираються регістром *SCON* (табл. 9.2).

В нульовому, синхронному, режимі дані передаються та приймаються через вхід *RXD*, вихід *TXD* передає синхроімпульси. В першому-третьому режимах дані передаються асинхронно виводом *TXD*, а приймаються *RXD*.

- в першому режимі передається старт-біт, 8 біт даних та стоп-біт;
- у другому та третьому режимах передається старт-біт, 9 біт даних і стоп-біт.

Щоб задати частоту передачі даних по UART використовуємо таймер 3, який в мікроконтролері ADuC841, призначений для даної задачі. Для того щоб здійснювати передачу даних з відповідною швидкістю необхідно налаштувати регістр *T3CON*, який відповідає за режим роботи таймера 3 для даного регістра потрібно визначити значення дільника (*DIV*) за формулою:

$$DIV = \frac{\log\left(\frac{f_{CODE}}{16 \times BaudRate}\right)}{\log(2)}$$

де: f_{CODE} – частота тактування контролера;
Baud Rate – швидкість передач (Бод).

При розрахунку заокруглення отриманого результату необхідно здійснити в сторону меншого цілого числа.

Таблиця 9.1 – Біти регістру *T3CON*

Біт	Назва	Призначення
7	T3BAUDEN	T3BAUDEN – 1. Таймер 3 ввімкнено для генерування передачі даних.
6...3		Зарезервовано.
2	DIV2	Бінарний дільник частоти DIV2 DIV1 DIV0 Бінарні дільники
1	DIV1	
0	DIV0	

Також необхідно встановити відповідний дробовий коефіцієнт дільника, який записуємо в регістр *T3FD* і значення розраховуємо за формулою:

$$T3FD = \frac{2 \times f_{CORE}}{2^{DIV-1} \times Baud\ Rate} - 64$$

Отримане значення слід заокруглювати до ближчого цілого числа.

Для прикладу проведемо розрахунок значення регістрів *T3CON* і *T3FD* при передачі даних з швидкістю 115200 Бод, для 11,0592МГц.

$$DIV = \log(11059200 / (16 \times 115200)) / \log 2 = 2,563 = 2$$

$$T3CON = 82H$$

$$T3FD = (2 \times 11059200) / (2^{2-1} \times 115200) - 64 = 32 = 20H$$

Таблиця 9.2 – Біти регістру *SCON*

Біт	Назва	Призначення
7	SM0	Біти вибору режиму роботи UART.
6	SM1	SM0 SM1 Режим роботи: 0 0 0: Синхронний, фіксована швидкість ($f_{clk}/2$). 0 1 1: 8-бітний, асинхронний, змінна швидкість. 1 0 2: 9-бітний, асинхронний, фіксована швидкість ($f_{clk}/32$) або ($f_{clk}/16$). 1 1 3: 9-бітний, асинхронний, змінна швидкість.
5	SM2	Біт дозволу багатопроцесорного зв'язку. - у режимі 0 має бути скинутий; - у режимі 1, якщо встановлений, то RI не спрацює, доки не отримано пакет зі встановленим дев'ятим бітом; - у режимі 2 та 3, якщо встановлений, то RI не спрацює, доки не отримано пакет зі встановленим дев'ятим бітом; - якщо скинутий, то RI спрацює одразу ж після завершення приймання біт даних
4	REN	Біт дозволу приймача. Встановлюється/скидається програмно
3	TB8	Дев'ятий біт даних для передавача
2	RB8	Дев'ятий біт даних для приймача
1	TI	Ознака закінчення передавання байта. Встановлюється UART після завершення передавання бітів даних. Скидається програмно
0	RI	Ознака закінчення прийому байта. Встановлюється після завершення приймання посилення. Скидається програмно

Приклад програми для лабораторної роботи №9

Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролер приймає число, яке відповідає номеру каналу АЦП, змінює дані і пересилає на ПЕОМ два байта даних вимірюного значення аналогово сигналу (старший байт передається першим). При отриманні числа яке не відноситься до номера каналу АЦП,

контролер повертає значення 0000H. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.

```
$include (mod841)

DAT    equ    R0    //дані
ADDR    equ    R1    //адрес

org 0
JMP Start    //перехід на мітку Start

ORG 023H    //вектор переривання від UART
jmp UART    //перехід на опрацювання переривання UART

ORG 030H

Start:
call ADCInit    //виклик ініціалізації АЦП
call Init_UART    //виклик ініціалізації UART
call clear_buff    //очистка даних буферів
setb EA    //дозвіл глобальних переривань
setb ES    //дозвіл переривання від UART

BEGIN:

    cjne R7,#0, measure    //перевірка на наявність прийнятих даних
    JMP BEGIN    //якщо дані не прийняли, то перехід на BEGIN
measure:    //якщо прийняли дані по UART
    call Measurev    //знімаємо значення аналогової величини з відповідного каналу АЦП
    call puch_char    //пересилаємо старший байт даних
    mov A,R6    //перезаписуємо молодший байт даних в акумулятор
    mov R7,A    //перезаписуємо акумулятор в R7
    call puch_char    //передаємо молодший байт даних
    mov R7,#0    //обнуляємо R7 BEGIN

JMP BEGIN    //перехід на

ADCInit:
    mov ADCCON1, #10111100b
    ret

Init_UART:
    mov T3FD, #20h    //за відповідною формулою
    mov T3CON, #82h    //11,0592MHz, 115200bot
    mov SCON, #52h    //SM = 1, TI = 1,
    ret

Measurev:
;In: -
;Out: R7:R6 -- ADC result
;Alters: a, PSW
    mov A, R7    //запис номера каналу для опитування АЦП в акумулятор
    orl A, #00010000b    //логічне «або» вмісту акумулятора з константою (встановлення каналу АЦП)
    mov ADCCON2, A    //запис вмісту акумулятора в регістр налаштування АЦП, запуск вимірювання
    jnb ADCL, $    //очікування завершення перетворення АЦП
    mov a, ADCDATAH    //запис старшого байта вимірюного значення в акумулятор
    anl a, #0fh    //очищення старшої тетради акумулятора
    mov R7, a    //запис старшого байта вимірюного значення в R7
    mov R6, ADCDATAL    //запис молодшого байта вимірюного значення в R6
    ret

write:
    setb P3.6
    mov P0, DAT    //moving data to bus buffer
    mov P2, ADDR    //set peripherals address
    nop
    nop
    mov P2, #000h    //clock
    ret

clear_buff:
    mov A, #7h
    nex_buff:
    mov DAT, #0ffh
    mov ADDR, A
    call write
    dec a
    cjne a, #0h, nex_buff
    ret
```



```

puch_char:
jnb TI,$          //очікування, якщо дані з SBUF не передано
clr TI           //очищення біта TI
mov SBUF, R7     //запис в SBUF значення з R7
ret

UART:
jnb RI,exit_uart //якщо переривання відбулося не при прийомі даних, то виходимо з переривання
clr RI           //інакше очищаємо RI
mov R7, SBUF     //перезаписуємо значення з SBUF в R7
exit_uart:
RETI             //вихід з переривання

END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролеру передається число, яке може мати значення (0...9), і відображається на матричному індикаторі, на інші цифри програма не реагує. Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: індикатор погашено.
2	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролеру передається число, яке може мати значення (1...8), що відповідає швидкості переміщення бігучої точки на лінійці світло діодів (кількість сегментів за секунду), а напрямку LED8...LED15. Швидкість передачі 19200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
3	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. На ПЕОМ контролер передає число, яке відповідає номеру натиснутої клавіші клавіатур, на натискання клаваш “*” і “#” контролер не реагує. Швидкість передачі 57600 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
4	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролер передається число, яке може мати значення (0...9), що відповідає швидкості обертання крокового двигуна в напрямку за годинниковою стрілкою, 0 – двигун зупинено, 1 – мінімальна швидкість, 9 – максимальна швидкість. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
5	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролеру передається число, яке може мати значення (0...9), що відповідає швидкості обертання двигуна постійного струму, 0 – двигун зупинено, 1 – мінімальна швидкість, 9 – максимальна швидкість. Управляти двигуном використовуючи вбудований ШІМ контролера, частота і розрядність довільна. Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
6	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролеру передається двобайтне число, яке може мати значення (0000...9999), прийняте число відображається на статичному семи сегментному індикаторі. Швидкість передачі 19200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: до прийому першого числа по UART статичний семи сегментний індикатор погашено індикатор погашено.
7	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролеру передається число, яке може мати значення (1...8), що відповідає яскравості світіння лінійки світлодіодів LED8...LED15, 1 – мінімальна яскравість, 8 – максимальна яскравість. Швидкість передачі 57600 бод, режим роботи: 8-

	бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
8	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. З ПЕОМ контролер передається число, яке може мати значення (1...99), що відповідає кількості «кроків», які повинен виконати двигун в напрямку проти годинникової стрілки. Нове значення для швидкості не приймається поки двигун не відпрацював попереднє. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
9	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. На ПЕОМ контролер передає два байта даних при натисканні кнопки«*» на матричній клавіатурі стенда, які відповідають за виміряне значення аналогового рівня з 7 каналу АЦП. Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: індикатор погашено.
10	Здійснити передачу даних по UART, на ПЕОМ прийом і передача виконується терміналом. На ПЕОМ контролер передає два байта даних при натисканні кнопки«#» на матричній клавіатурі стенда, які відповідають за виміряне значення аналогового рівня з 6 каналу АЦП. Швидкість передачі 57600 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: індикатор погашено.

Лабораторна робота №10. Програмування послідовного порту, інтерфейс RS-485

Тема: Програмування послідовного порту, інтерфейс RS-485.

Мета: Вивчити принцип передачі даних по інтерфейсу RS-485, зокрема ознайомитися з режимами роботи послідовного інтерфейсу, здійснити налаштування прийому/передачі даних на різних режимах

Порядок виконання роботи:

1. Вивчити принцип роботи послідовного інтерфейсу RS-485.
2. Розробити алгоритм індивідуального завдання до початку заняття.
3. Розробити програму і скомпілювати її.
4. Завантажити програму в стенд, переконавшись в правильності її роботи, при негативному результаті виявити допущені помилки і виправити їх. Повторити завантаження схеми в стенд.

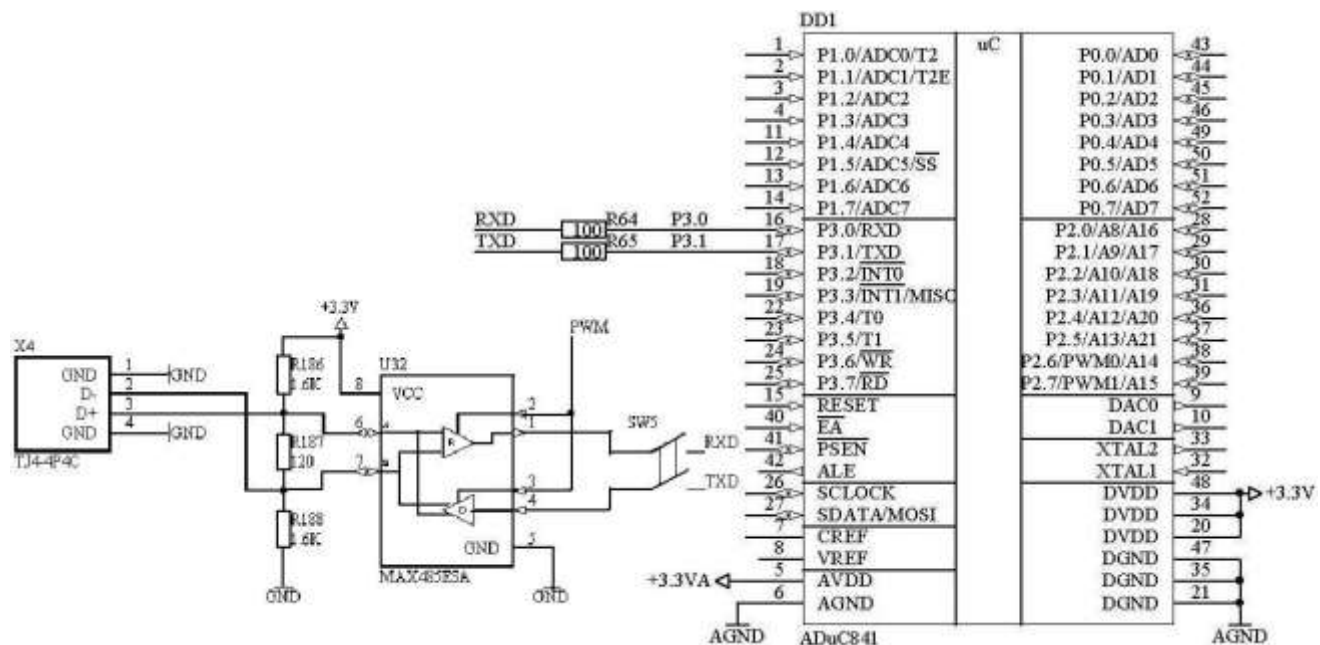


Рисунок 10.1 – Схема для лабораторної роботи №9

Короткі теоретичні відомості

RS-485 (*Recommended Standard 485*), **EIA-485** (*Electronic Industries Alliance-485*) – стандарт фізичного рівня для асинхронного інтерфейсу. Регламентує електричні параметри напівдуплексної багато точкової диференційної лінії зв'язку типу «спільна шина».

Стандарт набув великої популярності і став основою для створення цілого сімейства промислових мереж широко використовуваних в промисловій автоматизації.

Стандарт RS-485 спільно розроблений двома асоціаціями: Асоціацією електронної промисловості (EIA) і асоціацією промислових засобів зв'язку (TIA).

В стандарті RS-485 для передавання і приймання даних використовується одна вита пара провідників (рис. 10.2), інколи супроводжується екранованим оплетенням або загальним проводом. Передавання даних здійснюється за допомогою диференціальних сигналів. Різниця потенціалів однієї полярності між провідниками однієї полярності приймається як логічна одиниця, різниця іншої полярності – ноль (рис. 10.3).

1. Стандарт RS-485 обумовлює лише електричні і часові характеристики інтерфейсу.

2. Стандарт RS-485 не обумовлює:

- параметри якості сигналів (допустимий рівень спотворення, відбивання в довгих лініях);
- типи з'єднань і кабелів;
- гальванічну розв'язку ліній зв'язків;
- протокол обміну.

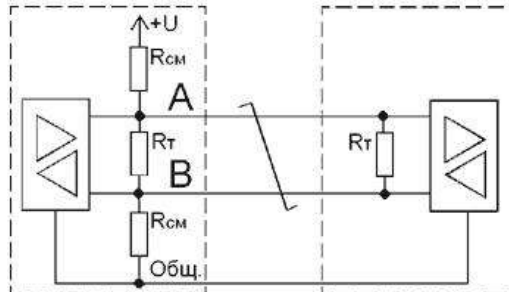


Рисунок 10.2 – Інтерфейс RS-485

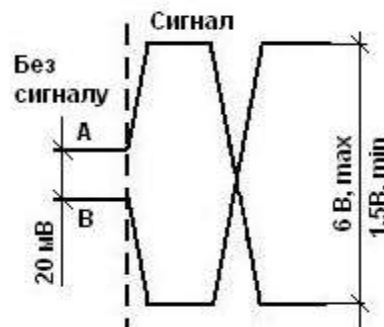


Рисунок 10.3 – Рівні сигналів

При роботі з напівдуплексним інтерфейсом RS-485 (прийом і передавання здійснюється по одній парі провідників з розділенням по часу) можна забути, що UART контролера – повно дуплексний, тобто приймає і передає дані незалежно і одночасно.

Зазвичай під час роботи прийомопередавача RS-485 на передачу, вихід приймача RO (рис. 10.4) переводиться в третій стан і вивід RX контролера (приймач UART) залишається «висячим». В результаті, під час передачі на приймачі UART замість рівня стопового біта «1» буде присутнє невідоме число і будь-яка завада буде прийнята за вхідний сигнал. Тому потрібно або на час передачі відключити приймач UART (за допомогою управляючого регістра), або підтягнути RX до одиниці.

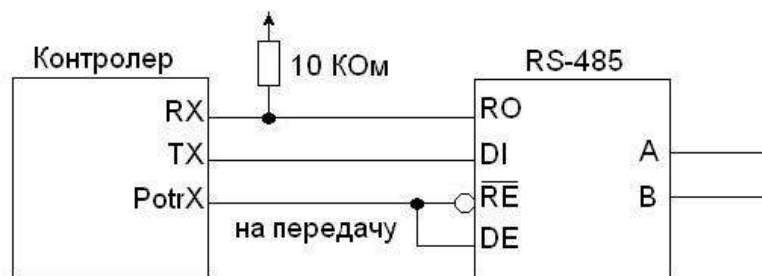


Рисунок 10.4 – Встановлення прийомопередавача RS-485 на передачу

Для передачі даних з по інтерфейсу RS-485 використовуємо вбудований в контролер UART, налаштування швидкості передавання і параметрів описано в Лабораторній роботі №9.

Зауваження. Для здійснення передачі даних необхідно використати мікросхему MAX485, для цього після програмування стенда перевести перемикач SW8 в нижнє положення (відключення від UART контролера драйвера RS-232) і перевести перемикач SW5 у верхнє положення (підключення до UART контролера драйвера RS-485).

Приклад програми для лабораторної роботи №9

Здійснити прийом/передачу даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить число, яке може мати значення (1..8), контролер приймає число і засвічує відповідну кількість світлодіодів (по порядку в напрямку LED15...LED8), при цьому відсилає у відповідь число на одиницю більше від отриманого. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: світлодіодна лінійка погашена до початку прийому першого байта.

```
$include (mod841)
```

```
DAT    equ    R0    //дані
ADDR    equ    R1    //адрес
```

```
org 0
JMP Start                //перехід на мітку Start
```

```
ORG 023H                //вектор переривання UART
jmp UART_ISR            //перехід на опрацювання переривання UART
```

```
ORG 030H
```

```
Start:
call Init_UART          //виклик ініціалізації UART
call clear_buff          //очистка діних буферів
setb EA                 //дозвіл глобальних переривань
setb ES                 //дозвіл переривань від UART
clr P2.7                //налаштування RS-485 на прийом
```

```
mov R7,#00h            //початковий стан прийнятих бат
```

```
BEGIN:
```

```
    cjne R7,#0, write_led    //перевірка на наявність прийнятих даних
    JMP BEGIN                //якщо дані не прийняли, то перехід на BEGIN
write_led:
    mov a,R7                //якщо прийняли дані по UART
    mov DPTR,#DANI_LED      //запис прийнятих даних в акумулятор
    movc A,@A+DPTR          //запис в DPTR таблиці для світлодіодів
    mov DAT,a               //звертання до балиці DANI_LED
    mov DAT,a               //запис заних на вивід
    mov ADDR,#07h           //запис адресу лінійки світлодіодів
    call write              //виклик процедури виведення

    inc R7                  //інкремент отриманого значення прийнятого байта
    setb P2.7               //налаштування RS-485 на передачу
    call puch_char          //пересилання байту даних по UART
    jnb TI,$                //очікування завершення пересилання байта
    clr P2.7                // налаштування RS-485 на прийом
    mov R7,#0               //обнуління прийнятого значення
JMP BEGIN
```

```
Init_UART:
    mov T3FD, #20h          //за відповідною формулою
    mov T3CON,#82h          //11,0592MHz, 115200    bod
    mov SCON, #52h          // SM = 1, TI = 1
ret
```

```
write:
    setb P3.6
    mov P0, DAT              //moving data to bus buffer
    mov P2, ADDR             //set peripherals address
    nop
    nop
    mov P2,#000h             //clock
ret
```

```
clear_buff:
    mov A,#7h
nex_buff:
```

```

mov DAT,#0ffh
mov ADDR,A
call write
dec a
cjne a, #0h, nex_buff
ret

puch_char:
jnb TI,$           //очікування якщо дані з SBUF не передано
clr TI            //очищення біта готовності передачі TI
mov SBUF, R7      //запис в SBUF значення з R7
ret

UART_ISR:
jnb RI,exit_uart  // якщо переривання відбулося не при прийомі даних, то виходимо з переривання
clr RI            //інакше скидаємо ознаку готовності прийнятих біт
mov R7, SBUF      //перезаписуємо значення з SBUF в R7
exit_uart:
RETI

DANI_LED:         //масив даних для світлодіодної лінійки
DB 11111111b      //0
DB 01111111b      //1
DB 00111111b      //2
DB 00011111b      //3
DB 00001111b      //4
DB 00000111b      //5
DB 00000011b      //6
DB 00000001b      //7
DB 00000000b      //8

END

```

Варіанти індивідуальних завдань

№	Текст індивідуального завдання
1	Здійснити прийом даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить число, що може мати значення (1..8), яке відповідає за швидкість переміщення світної точки по світлодіодній лінійці (кількість сегментів в секунду), напрямок LED15...LED8. Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний. Початковий стан: світлодіодна лінійка погашена до початку прийому першого байта.
2	Здійснити передачу даних по RS-485. Контролер передає одnobайтне число, яке відповідає значенню старших 8-біт від 7 каналу АЦП, дані передаються з дискретністю не більше 10Гц. Швидкість передачі 19200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
3	Здійснити прийом даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить число, що може мати значення (1..4), яке відповідає за кількість засвічених світлодіодів на лінійці, які переміщуються з сталою швидкістю 1сегмент в секунду, напрямок LED8...LED15. Швидкість передачі 57600 бод, режим роботи: 8-бітний, асинхронний. Початковий стан: світлодіодна лінійка погашена до початку прийому першого байта.
4	Здійснити передачу даних по RS-485. Контролер передає одnobайтне число, яке відповідає значенню старших 4-біт від 6 каналу АЦП, дані передаються з дискретністю не більше 10Гц. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
5	Здійснити прийом даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить число, що може мати значення (0..255), отримане число відображати на статичному семисегментному індикаторі. Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний. Початковий стан: індикатор погашено до почотку прийому першого числа.
6	Здійснити передачу даних по RS-485. Контролер передає одnobайтне число, яке відповідає за номер натиснутою кнопки на матричній клавіатурі, натискання кнопки «#» і «*» програма ігнорує. Швидкість передачі 19100 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.
7	Здійснити прийом даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить одnobайтне число, старша тетрада якого вказує на координати світної точки на

	матричному індикатор по горизонталі, молодша тетрада вказує на координати світної точки на матричному індикаторі по вертикалі. Нумерація індикатора з лівого нижнього кута (тобто при отриманні цифри 011Н, засвічується нижній лівий сегмент матричного світлодіодного індикатора, при отриманні цифри 057Н, засвічується верхній правий сегмент матричного світлодіодного індикатора). Швидкість передачі 57600 бод, режим роботи: 8-бітний, асинхронний. Початковий стан: матричний світлодіодний індикатор погашена до початку прийому першого байта.
8	Здійснити передачу даних по RS-485. Контролер передає однобайтне число, яке відповідає значенню змінної, дане значення інкрементується на одиницю при натисканні кнопки на енкодері і декриментується при натисканні кнопки на джойстику, при досягненні значення 0 - змінна не зменшується, при досягненні значення 99 - змінна не збільшується. Швидкість передачі 115200 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта. Початковий стан: передається значення змінної.
9	Здійснити прийом даних по RS-485. На вхід драйвера мікросхеми RS-485 приходить число, що може мати значення (0..9), яке відповідає за яскравість світіння світлодіодної лінійки (0 – індикатори погашено, 1 – мінімальна яскравість засвітки, 9 – максимальна яскравість засвітки індикатора). Швидкість передачі 9600 бод, режим роботи: 8-бітний, асинхронний. Початковий стан: світлодіодна лінійка погашена до початку прийому першого байта.
10	Здійснити передачу даних по RS-485. Контролер передає однобайтне число, яке відповідає значенню змінної, яка задається обертанням ручки енкодера, при повороті в напрямку за годинниковою стрілкою змінна інкрементується, при обертанні в напрямку проти годинникової стрілки змінна декриментується, значення змінне відображається у двійковому коді на світлодіодній лінійці. Швидкість передачі 19100 бод, режим роботи: 8-бітний, асинхронний, дозвіл встановлення біта ознаки закінчення передавання байта.

Список використаних джерел

1. FM24C256 256Kb FRAM Serial Memory Rev 1.2 Jan. 2002.
2. <http://www.gaw.ru/html.cgi/txt/lcd/lcm/Bolymin/char/BC1004A.htm>
3. 16-megabit 5-volt Only Serial ® DataFlash AT45D161 <http://www.atmel.com>
4. <http://www.msclub.ce.cctpu.edu.ru/MCS51-PLD/TZ.html>
5. <http://kazus.ru/articles/419.html>
6. Programmable controllers: theory and implementation /L.A. Bryan, E. A. Bryan. TJ223.P76B795 1997. – 1035p.
7. Однокристалльные микро-ЭВМ. Техническое описание и руководство по применению / Г.П. Литвинский., Москва , 1982.

ДОДАТКИ

