МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

НАВЧАЛЬНА МОДЕЛЬ АНАЛОГОВО - ЦИФРОВОГО ПЕРЕТВОРЮВАЧА

(Дипломний проект)

Роман Чаглей

Чернівці, 2008

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

Вступ

1. огляд принципів роботи та будова ацп

1.1 Будова та типи АЦП

1.2 Паралельний порт принтера

1.2.1 Призначення паралельного порта

1.2.2 Загальний опис інтерфейсу Cetronics

1.2.3 Переваги й недоліки інтерфейсу Cetronics

1.2.4 Порядок обміну по інтерфейсу Cetronics

2. функціональнА та електричнА принциповА схемА

2.1 Функціональна схема

2.2 Схема електрична принципова

2.2.1 Вузол буферизації та індикації

2.2.2 Цифро-аналоговий перетворювач

2.2.3 Компаратор

3. конструкторсько-технологічна розробка

3.1 Розробка електричної принципової схеми

3.2 Розробка друкованої плати.

3.3 Розробка збірного креслення

4. Розробка програмного забеспечення

4.1 Алгоритм керування

4.2. Обгрунтування вибору програмних засобів

4.3 Програма для роботи з АЦП

4.3.1 Процедура обрахунку напруги за її двійковим представленням23

4.3.2 Процедура P\_Write\_DR.

4.3.3 Процедура запису P\_Write\_CR.

4.3.4 Процедура читання P\_Write\_SR.

4.3.5 Процедура B\_In

5. Техніко-економічне обгрунтування проекту

5.1. Витрати на сировину і матеріали

5.2. Витрати на закупівельні комплектуючі вироби

5.3. Розрахунок заробітної плати

6. Охорона праці

список використаних джерел..

Додатки

Додаток А. Текст програми

Додаток Б. Тестування перетворювача

Додаток В. Принцип роботи АЦП

ПЕРЕЛІК СКОРОЧЕНЬ

АЦП –аналого-цифровий перетворювач.

ЦАП – цифро-аналоговий перетворювач.

LPT – лінійний порт принтера (Line PrinTer).

ОС – операційна система.

# Вступ

В пояснюючій записці розглядається розробка на тему „Навчальна модель аналогово-цифрового перетворювача”. Розробка виконана для кафедри „Комп’ютерні системи та мережі” Чернівецького національного університету імені Юрія Федьковича і може використовуватись для проведення лабораторних робіт з дисципліни “Пристрої зв’язку з об’єктом”.

1. огляд принципів роботи та будовИ ацп

1.1 Будова та типи АЦП

Аналого-цифрові перетворювачі (АЦП) є пристроями, які приймають вхідні аналогові сигнали і генерують відповідні їм цифрові сигнали, придатні для обробки мікропроцесорами та іншими цифровими пристроями.

Принципово не виключена можливість безпосереднього перетворення різних фізичних величин в цифрову форму, проте це завдання вдається вирішити лише в окремих випадках через складність таких перетворювачів. Тому в даний час найбільш раціональним визнається спосіб перетворення різних по фізичній природі величин спочатку у функціонально пов'язані з ними електричні, а потім вже за допомогою перетворювачів напруга-код в цифрові. Саме ці перетворювачі мають зазвичай на увазі, коли говорять про АЦП.

Процедура аналого-цифрового перетворення безперервних сигналів, яку реалізують за допомогою АЦП, є перетворення безперервної функції часу U(t), що описує початковий сигнал, в послідовність чисел {U'(tj)}, j=0,1,2:, віднесених до деяких фіксованих моментів часу. Цю процедуру можна розділити на дві самостійні операції. Перша з них називається дискретизацією і полягає в перетворенні безперервної функції часу U(t) в безперервну послідовність {U(tj)}. Друга називається квантуванням і полягає в перетворенні безперервної послідовності в дискретну {U'(tj)}.

У основі дискретизації безперервних сигналів лежить принципова можливість уявлення їх у вигляді зважених сум

,



де аj - деякі коефіцієнти або відліки, що характеризують початковий сигнал в дискретні моменти часу; fj(t) - набір елементарних функцій, використовуваних при відновленні сигналу по його відліках.

Аналогово-цифрові перетворювачі

послідовні

паралельні

Послідовно-паралельні

Послідовного приближення

Послідовного рахунку

Інтегруючі

Багатоконтактні

Багатоступінчасті

Конвеєрні

Рис. 1.1. Класифікація АЦП за методами перетворення

В даний час відоме велике число методів перетворення напруга-код. Ці методи істотно відрізняються один від одного потенційною точністю, швидкістю перетворення і складністю апаратної реалізації. На рис.1.1 представлена класифікація АЦП по методах перетворення.

У основу класифікації АЦП покладена ознака, вказуюча на те, як в часі розгортається процес перетворення аналогової величини в цифрову. У основі перетворення вибіркових значень сигналу в цифрові еквіваленти лежать операції квантування і кодування. Вони можуть здійснюватися за допомогою послідовної, паралельної або послідовно-паралельної процедур наближення цифрового еквівалента до перетворюваної величини.

В паралельних АЦП здійснюють квантування сигналу одночасно з допомогою набору компараторів, включених паралельно джерелу вхідного сигналу. На рис.1.2 показана реалізація паралельного методу перетворення для 3-х розрядного числа.

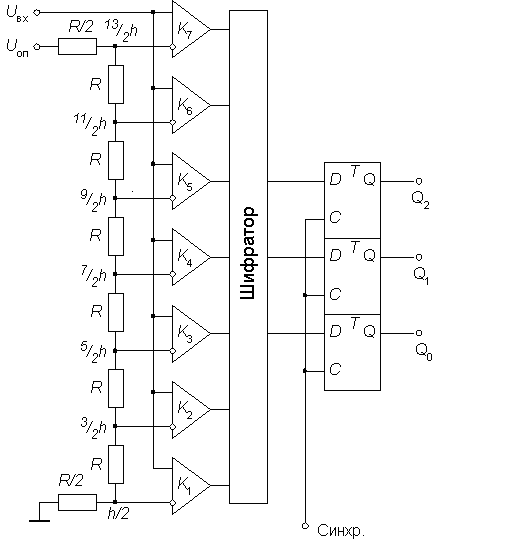


Рис. 1.2. Схема параллельного АЦП

За допомогою трьох двійкових розрядів можна представити вісім різних чисел, включаючи нуль. Необхідно, отже, сім компараторів. Сім відповідних еквідистантних опорних напруг утворюють за допомогою резистивного дільника.

Завдяки одночасній роботі компараторів паралельний АЦП є найшвидшим. Наприклад, восьмирозрядний перетворювач типу МАХ104 дозволяє отримати 1 млрд. відліків в секунду при часі затримки проходження сигналу не більше 1,2 нс. Недоліком цієї схеми є висока складність. Дійсно, N-розрядний паралельний АЦП має 2N-1 компараторів і 2N узгоджених резисторів. Наслідком цього є висока вартість (сотні доларів США) і значна споживана потужність. Вищезгаданий МАХ104, наприклад, споживає близько 4 Вт.

Типовим прикладом послідовних АЦП з одиничними наближеннями є наведений пристрій (рис.1.3), який складається з компаратора, лічильника і ЦАП. На один вхід компаратора поступає вхідний сигнал, а на іншій - сигнал зворотного зв'язку з ЦАП.

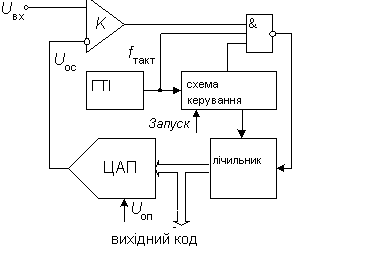


Рис 1.3. Структурна схема АЦП послідовного рахунку: К - компаратор, ГТІ – генератор тактових імпуьсів

Робота перетворювача починається з приходу імпульсу запуску, який включає лічильник, що підсумовує число імпульсів, що поступають від генератора тактових імпульсів ГТІ. Вихідний код лічильника подається на ЦАП, що здійснює його перетворення в напругу зворотного зв'язку Uос. Процес перетворення продовжується до тих пір, поки напруга зворотного зв'язку порівняється з вхідною напругою і перемкнеться компаратор, який своїм вихідним сигналом припинить надходження тактових імпульсів на лічильник. Перехід виходу компаратора з 1 в 0 означає завершення процесу перетворення. Тобто вихідний код пропорційний вхідній напрузі у момент закінчення перетворення.

Час перетворення АЦП цього типу є змінним і визначається вхідною напругою. Його максимальне значення відповідає максимальній вхідній напрузі і при розрядності двійкового лічильника N і частоті тактових імпульсів fтакт дорівнює

### tпр.макс = (2N-1)/ fтакт.

Наприклад, при N=10 і fтакт=1 Мгц значення tпр.макс=1024 мкс, що забезпечує максимальну частоту вибірок близько 1 кГц.

Статична похибка перетворення визначається сумарною статичною похибкою використовуваних ЦАП і компаратора. Частоту рахункових імпульсів необхідно вибирати з урахуванням завершення перехідних процесів в них.

При роботі без пристрою вибірки-зберігання апертурний час співпадає з часом перетворення. Як наслідок, результат перетворення дуже сильно залежить від пульсацій вхідної напруги. За наявності високочастотних пульсацій середнє значення вихідного коду нелінійно залежить від середнього значення вхідної напруги. Це означає, що АЦП даного типу без пристрою вибірки-зберігання придатні для роботи з постійними напругами або такими, що повільно змінюються (за час перетворення змінюється не більш, ніж на значення кванта перетворення).

Таким чином, особливістю АЦП послідовного рахунку є невелика частота дискретизації, що досягає декількох кілогерц. Перевагою АЦП даного класу є порівняльна простота будови, яка визначається послідовним характером виконання процесу перетворення.

Послідовнопаралельні АЦП є компромісом між прагненням отримати високу швидкодію і бажанням зробити це по можливості меншою ціною. Послідовнопаралельні АЦП займають проміжне положення по роздільній здатності і швидкодії між паралельними АЦП і АЦП послідовного наближення. Послідовнопаралельні АЦП підрозділяють на багатоступінчаті, багатоконтактні і конвеєрні.

1.2 Паралельний порт принтера

1.2.1 Призначення паралельного порта

Паралельний порт принтера призначений для зв’язку комп’ютера з принтером, сканером, а також з нестандартними пристроями. Порт принтера також називається лінійним портом принтера LPT, а його сигнали відповідають інтерфейсу Centronics.

1.2.2 Загальний опис інтерфейсу Centronics

Паралельний принтерний порт – це 25-контактна гніздова частина роз’єму D-типу. Існують три адреси, пов’язані з цим портом (base, base+1, base+2). Фактичну базову адресу (base) можна знайти в супроводжуючій документації до комп’ютера або з програми тестування комп’ютера.

Звичайні базові адреси – 378h або 3BCh.

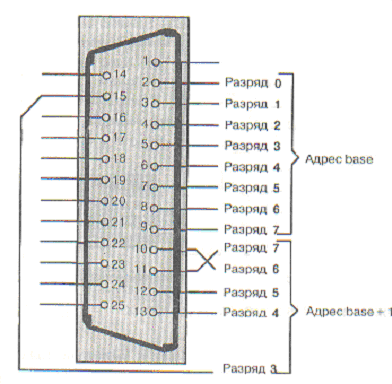


Рис. 1.4. Призначення контактів LPT-порта по розрядам

Даний порт має 8 розрядів виводу даних і декілька розрядів для ліній керування. Деякі з останніх використовуються тільки для виводу керуючої інформації, деякі – для вводу, і деякі можуть бути запрограмовані як для вводу, так і для виводу. В якості логічних рівнів використовуються ТТЛ-рівні напруги (0 і 5 В). По базовій адресі виводу „розташовуються” контакти з другого по восьмий з наймолодшим розрядом на контакті 2 наявні 5 бітів вводу за адресою base+1, розведення контактів для цих розрядів представлене на рис. 1.4.

1.2.3 Переваги та недоліки інтерфейсу Centronics

Інтерфейс Centronics завдяки простоті сполучення і зручності програмування широко використовується для підключення до комп'ютера нестандартних зовнішніх пристроїв.

По-перше, можливості реалізації різних протоколів інформаційного обміну з пристроєм через паралельний порт невеликі. Дійсно, невелика кількість сигнальних ліній інтерфейсу і можливості його програмування не дозволяють реалізувати обмін по перериваннях чи прямий доступ до пам'яті. Практично приходиться обмежуватися програмно-керованим обміном.

Крім того, оскільки інтерфейс Centronics є програмно-керованим, швидкість інформаційного обміну не може бути особливо велика і виявляється прямо зв'язаною зі швидкодією комп'ютера.

Є також обмеження на довжину лінії зв'язку пристрою, підключеного до інтерфейсу Centronics. Він повинен розташовуватися на відстані не більш 1,5-2 метрів від комп'ютера.

Ще однією особливістю інтерфейсу Centronics є відсутність шин живлення (є тільки "земля"). Це означає, що пристрій, що сполучається, повинен використовувати зовнішнє джерело живлення.

Основною перевагою інтерфейсу Centronics є його стандартність – він є на кожному комп'ютері і на всіх комп'ютерах працює однаково (правда, з різною швидкістю). Для підключення зовнішнього пристрою до паралельного порту не потрібно відкривати системний блок комп'ютера, що для багатьох користувачів може стати проблемою. Треба тільки приєднати кабель до роз’єму на його задній стінці.

Можна також відзначити таку перевагу інтерфейсу Centronics, як простота його програмування на будь-якому рівні. У більшості мов програмування присутні процедури взаємодії з принтером, що легко використовувати і для програмування нестандартного пристрою. А оскільки з точки зору програмування Centronics являє собою три програмно доступних регістри, не викликає ускладнень і написання програм нижчого рівня.

1.2.4 Порядок обміну по інтерфейсу Centronics

Основним призначенням інтерфейсу Centronics є підключення до комп'ютера принтерів різних типів. Тому розподіл контактів роз’єму, призначення сигналів, програмні засоби керування інтерфейсом орієнтовані саме на це використання. У той же час за допомогою даного інтерфейсу можна підключати до комп'ютера й інші зовнішні пристрої, що мають роз’єм Centronics, а також спеціально розроблені пристрої.

Основною перевагою використання Centronics для підключення пристроїв у порівнянні з ISA є значно менший ризик вивести комп'ютер з ладу. Головний недолік цього підходу — значно менша швидкість обміну. Призначення 36 контактів роз’єму Centronics приведено в табл. 1.1

Таблиця. 1.1

Призначення контактів роз’ємів Centronics (I – вхідний сигнал комп'ютера, O — вихідний сигнал)

|  |  |  |  |
| --- | --- | --- | --- |
| Контакт роз’єму комп’ютера | Коло | I/O\* | Контакт роз’єму принтера |
| 1 | 2 | 3 | 4 |
| 1 | -STROBE | O | 1 |
| 2 | D0 | O | 2 |
| 3 | Dl | O | 3 |
| 4 | D2 | O | 4 |
| 1 | 2 | 3 | 4 |
| 5 | D3 | O | 5 |
| 6 | D4 | O | 6 |
| 7 | D5 | O | 7 |
| 8 | D6 | O | 8 |
| 9 | D7 | O | 9 |
| 10 | -ACK | I | 10 |
| 11 | BUSY | I | 11 |
| 12 | PE | I | 12 |
| 13 | SLCT | I | 13 |
| 14 | -AUTO FD | O | 14 |
| 15 | -ERROR | I | 32 |
| 16 | -INIT | O | 31 |
| 17 | -SLCTIN | O | 36 |
| 18...25 | GND | - | 16,17,19...30,33 |

\* 0 – виведення, 1 – введення даних.

2. ФУНКЦІОНАЛЬНА ТА ЕЛЕКТРИЧНА ПРИНЦИПОВА СХЕМА

2.1 Функціональна схема

Модель аналогово-цифрового перетворювача складається з таких функційних блоків:

* Буфер та блок індикація регістру DR.
* Буфер регістру СR.
* Буфер регістру SR.
* LPT-порту ком’ютера.
* ЦАП регістру DR.
* ЦАП регістру CR.
* Ключ.
* Компаратор.
* Регулятора вхідної напруги.

Працює схема наступним чином (рис 2.1) :

Напруга Uвх, яку потрібно перетворити у цифровий вигляд, подається на ключ, який відкриває проходження Uвх до компаратора або закриває Uвх, відкриваючи проходження напруги від ЦАП регістру СR (якщо АЦП працює в режимі перевірки). Далі Uвх подається на компаратор, де вона порівнюється з напругою, яка подається з ЦАП регістру DR. Напруга на ЦАП формується так: сигнали з регістру DR подаються в буфер та блок індикації регістру DR, а також на дільник напруги.

DR

Буфер та

блок індикація DR

DR

ЦАП

DR

CR

Буфер

CR

ЦАП

CR

SR

Буфер

SR

Uвх

Компаратор

Ключ

LPT порт

Рис. 2.1. Схема електрична функціональна АЦП

Буфер використовується для забезпечення надійності роботи пристою. Блок індикації забезпечує спостереження за проходженням цифрового сигналу. В ЦАП регістру DR цифровий сигнал перетворюється в напругу, яка поступає на один з двох входів компаратора. Якщо напруга, яка надходить з ЦАП регістру DR, стає більшою Uвх на вході компаратора, то подається відповідний сигнал в буфер регістру SR, а далі в регістр SR і припиняється порівняння. Вважається, що Uвх рівна напрузі з виводу ЦАП регістру DR, цифровий вигляд Uвх відповідає цифровому сигналу регістру DR.

В режимі тестування замість Uвх подається напруга з ЦАП регістру CR, де вона формується так само, як в ЦАП регістру DR: способом поданням на вхід ЦАП різних комбінацій цифрового сигналу.

2.2 Схема електрична принципова

2.2.1 Вузол буферизації та індикації

Вузол буферизації та індикації складається з резисторів, які обмежують напругу, що подається на вхід мікросхем тригерів Шмітта (які нормілізують напругу до 0 чи 5 В), а також світлодіодів, що дозволяють побачити стан сигналу.

Резистори R1, R2, R3, R4 обмежують напругу, яка подається з з 2-го по 5-й контакт LPT-порта. Напруга подається на мікросхеми DD1 та DD2 на 13 і 9 ніжки (тобто на тригери Шмітта). Тут сигнали інвертуються і нормалізуються до логічного 0 чи 1 (тобто строго 0 чи 5 В). Резистори R11, R12, R13, R14 обмежують напругу (тобто яскравість світіння) світлодіодів відповідно VD0, VD1, VD2, VD3.

Резистори R5 по R8 також обмежують напругу, яка подається з з 1-ї, 14-ї, 31-ї, 36-ї ніжки LPT-порта і разом з DD2 та DD3 утворюють буфер, що забеспечує надійність мікросхеми від стрибків напруги.

2.2.2 Цифро-аналоговий перетворювач

Даний аналого-цифровий перетворювач в своєму складі має два ЦАП. 1-й ЦАП складається з:

* діодів V1, V2, V3, V4, які забеспечують одностороннє проходження сигналу;
* резисторів R15, R16, R17,R18, що формують на виході різне значення напруги завдяки поданню на вхід резистора 0 чи 5В; резистори мають різний опір;
* резистора R19, що з’єднаний з лінією заземлення.

2-й ЦАП за складом аналогічний і складається з:

* діодів V5, V6, V7, V8, які забеспечують одностороннє проходження сигналу;
* резисторів R20, R21, R23,R24, що формують на виході різне значення напруги завдяки поданню на вхід резистора 0 чи 5В; резистори мають різний опір.
* резистора R22, що з’єднаний з лінією заземлення.

2.2.3 Компаратор

Компаратор складається з мікросхеми DD4. На вхід компаратора подається вимірювана напруга з регулятора змінного опору R28 або з 2-го ЦАП, де формується заздалегідь відома напруга, на 10 ніжку мікросхеми DD4 через резистор R26. Вимірювана напруга порівнюється з напругою, яка формується на 1-му ЦАП. Якщо напруга з 1-го ЦАП стає більшою вимірюваної напруги, то на 8-му ніжку DD4 подається логічний 0. Рівень логічного 0 забеспечується трігерами Шмітта де він два рази інвертується. Далі сигнал зчитується в компютер через 32-гу ніжку LPT-порта. Перш ніж сигнал попаде на SR, він пройде через буферизуючий резистор R9. Тобто, якщо на інверсний вхід компаратора подається напруга менша, ніж на прямий, то на його виході буде +5 В, а інакше – 0 В.

3. конструкторсько-технологічна розробка

3.1 Розробка електричної принципової схеми

Принципова електрична схема пристрою була розроблена з допомогою спеціального програмного забезпечення Protel 99, що використовується в конструкторських цілях. Створення принципової схеми проходило в 2 етапи:

- створення бібліотеки елементів згідно з стандартів;

- побудова самої схеми з використанням створеної раніше елементної бази.

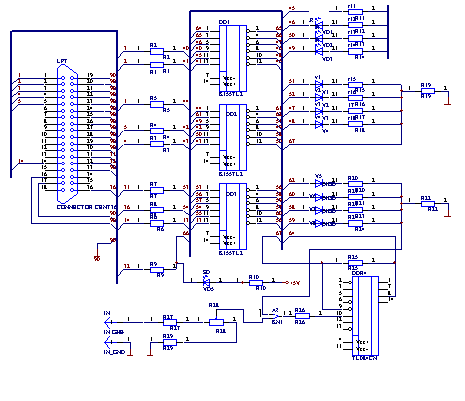


Рис 3.1. Схема електрична функціональна

Елементи розміщувалися за власним розсудом, але з узгодженням певних питань з керівником даного проекту.

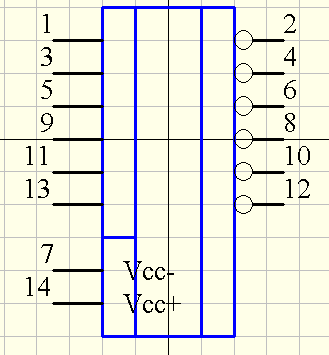
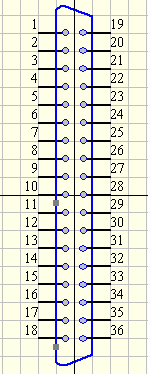


Рис 3.2. Елемент порт Сetronics.

Рис 3.3. Мікросхема К155ТЛ2.

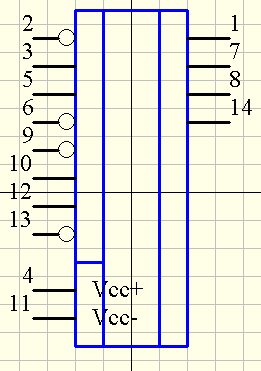


Рис 3.4. Мікросхема TL084CN.

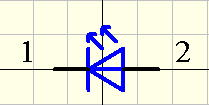
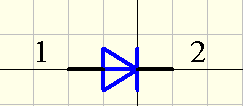


Рис 3.5. Діод.

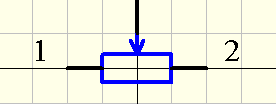


Рис 3.6. Світлодіод.

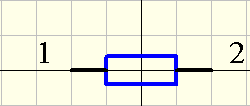


Рис 3.7. Резистор.

Рис 3.8. Реостат.

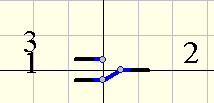


Рис 3.9. Перемикач.

Розглянемо роботу ЦАП, який створений у вигляді дільника напруги (рис. 3.10).

Rc

Ud

DR0

R0

DR1

Rz



R1



DR2

R2



DR3

U\_DR

R3



Рис 3.10. Електрична схема дільника напруги

Вихідна напруга дільника рівна:

,

де Rc – загальний опір резисторів і діодів, Ud – напруга регістрів DR0 – DR3, U\_DR – напруга на виході ЦАП.

3.2 Розробка друкованої плати

Система Protel 99 SE має потужний інструмент синхронізації, який полегшує передачу даних з етапу проектування принципової електричної схеми в редактор друкованих плат і навпаки.

Синхронізатор автоматично отримує інформацію про компонент і параметри звязків з принципової схеми , знаходить відповідному компоненту топологічні посадочні місця в бібліотеці елементів друкованих плат і розміщує їх на кресленні друкованої плати, а потім додає лінії зв’язків між з’єднаними виводами компонентів.

3.3 Розробка збірного креслення

Компоновка пристрою - одна із найбільш складних задач конструювання. Ручний процес компоновки радіодеталей та допоміжних перемикачів, ручок регулювання, гнізд займає багато часу, оскільки приходиться розглядати декілька можливих варіантів і вибрати оптимальний для заданих умов застосування приладу і процесу його виробництва. Під час компоновки приладу потрібно враховувати склад елеменої бази, зручність у користуванні, забезпечення високої ремонтопридатності і безпеки від зовнішніх впливів і внутрішніх дестабілізуючих факторів.

Компоновка дозволяє провести оцінку електромагнітних і теплових зв′язків, розрахувати кінематичні зв′язки, оцінити основні конструкторсько-технологічні рішення і розрахувати основні показники якості конструкції. За результатами компоновки складається компоновочна схема - збірне креслення.

4. РОЗРОБКА ПРОГРАМНОГО ЗАБЕСПЕЧЕННЯ

4.1 Алгоритм керування

Для роботи розробленого пристрою необхідно, щоб на 4 входи пристрою DR подавалися сигнали (5 або 0 В). Завдяки певним комбінаціям сигналів на ЦАП утворюється різна напруга. Кількість комбінацій сигналу, який може перебувати в двох станах вираховується за формулою:

,

де k – кількість комбінацій, m – кількість станів сигналу, n – розрядність сигналу.

Одже сигнал може перебувати в двох станах 0 або 5 В. Розрядність рівна 4. Отже кількість різних комбінацій буде рівна кількості різних напруг, які зможе сформувати ЦАП, рівна:

.

Необхідно також, щоб на 4 входи пристрою CR також подавались сигнали аналогічно як і на DR. Це необхідно тому, що в пристрої передбачена можливість працювати в режимі тестування, коли подається наперед відома напруга для визначених комбінацій сигналу.

Крім того, в алгоритмі керування необхідно не тільки посилати дані на пристрій, а й зчитувати їх. Зчитати потрібно сигнал з виходу компаратора, який повідомить що U\_DR>Uвх. Після одержання від компаратора сигналу потрібно припинити формування комбінацій сигналу на вхід ЦАП, а сформовану комбінацію прийняти за цифровий код напруги Uвх.

4.2 Обгрунтування вибору програмних засобів

В якості операційної системи вибрана система Windows XP сімейства Windows NT. Це аргументовано тим, що Windows найбільш розповсюджена і доступна, а також досить легка у користуванні.

В якості середовища програмування використано Delphі. Середовище Delphі включає в себе повний набір візуальних інструментів для швидкої розробки додатків (RAD – Rapid Application Development), що дозволяє виконати розробку інтерфейсу користувача, який задовольняє всім вимогам замовника. VCL – бібліотека візуальних компонентів, включає в себе стандартні об’єкти побудови інтерфейсу користувача, об’єкти керування даними, графічні об’єкти, об’єкти мультимедіа, діалоги і об’єкти керування файлами, керування DDE і OLE.

Інтегроване середовище програмування Delphі володіє всіма необхідними параметрами, які притаманні іншим системам даного класу:

- візуальна побудова інтерфейсів (Vіsual User-іnterface buіlder) дає можливість швидко створювати додатки візуально, просто обираючи компоненти з відповідної палітри;

- Delphі повністю підтримує такі програмні концепції як інкапсуляція, наслідування, поліморфізм та керування подіями;

Перевагами середовища Delphі в порівнянні з іншими аналогічними засобами розробки програм є значно простіше його використання, що дозволяє виконувати подальшу модернізацію програми іншими програмістами з мінімальними витратами часу на дослідження синтаксичних та логічних особливостей даної програми. Окрім цього, Delphі володіє досить великим об’ємом документації, як створеної розробниками Delphі, так і сторонніми виробниками, що значно полегшує розробку програм. Отже, за співвідноенням можливості-простота-ціна та враховуючи побажання замовників до мови програмування, інтегроване середовище програмування Delphі є оптимальним вибором для вирішення поставленої задачі.

Для роботи з LPT-портом використовується DLL-бібліотека IO.dll стороннього виробника, яка дозволяє працювати з портами. З даної бібліотеки використовується дві функції та одна процедура. Функція IsDriverInstalled : Boolean повертає TRUE у разі успішної установки драйвера LPT-порта. Функція PortIn(Port : Word) : Byte зчитує дані з порта по вказаній адресі, яка задається значенням змінної Port. Процедура PortOut(Port : Word; Data : Byte) записує в порт один байт, який вказується в змінній Data за адресою порта Port.

4.3 Програма керування АЦП

Загальний вигляд головної форми програми представлено на рис.

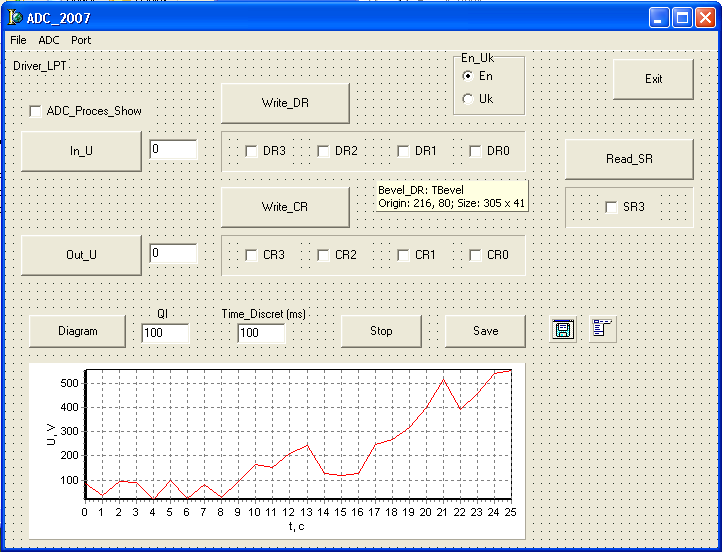


Рис.4.1. Екранна форма програми керування АЦП

Кнопка Write\_DR дозволяє записувати в порт принтера стан чотирьох бітів регістру DR (біти DR0, DR1, DR2, DR3).

Кнопка Read\_SR зчитує з порта принтера значення регітру SR (біт SR3).

Кнопка Write\_СR дозволяє записувати в порт принтера стан чотирьох бітів регістру DR (біти СR0, СR1, СR2, СR3).

Кнопка In\_U перетворює напругу, яка подається на плату у двійковий код DR0 – DR3.

Кнопка Out\_U записує у порт двійковий код, який відповідає введеній на формі напрузі.

За допомогою кнопки Diagram зчитується напруга QI разів і отримане значення будується у вигляді графіка. Проміжок часу між зчитуванням рівний Time\_Discret (Додаток Б).

4.3.1 Процедура обрахунку напруги за її двійковим представленням

Процедура p\_Init\_U встановлює звязок між напругою Ur і двійковою комбінацією, яка буде передана на LPT-порт, а далі в ЦАП, і з якої ЦАП сформує напругу Ur.

p\_Init\_U – процедура без параметрів. В процедурі оголошено пять змінних: nDr та bOut типу Integer, Rc1,Rc,Ur типу Real.

Тіло процедури починається з циклу, де bOut міняється від 0 до qB. Глобальна змінна qB приймає значення 15. В циклі виконуються наступні дії:

* процедура Trans\_bOut\_mDR(bOut) перетворює bOut з десяткової системи числення в двійкову і записує порозрядно в масив mDR[], який є глобальним для всьго модуля;
* змінній Rc1 присвоюєм значення 0;
* відкриваєм цикл, де nDR міняється від 0 до qDR. Глобальна змінна qDR приймає значення 3. В цьому циклі змінній Rc1 присвоюємо значення Rc1:=Rc1+(1/(mR[nDr]+Rdiod))\*mDR[nDr], де mR[i] – масив значень, елементи якого відповідають реальним значення опорів 4-х резисторів ЦАП. Rdiod – змінна, значення якої рівне значеню опору діодів, що підключені до резисторів послідовно. Отже з законом Ома загальний опір при паралельному включенні n опорів, буде рівний:

.

Доданок (1/(mR[nDr]+Rdiod)) множиться на елемент масиву mR[i], що приймає двійкове значення 0 чи 1. Тобто, якщо в двійковій комбінації певний розряд рівний 0, то струм не протікає, а отже і не враховується при обчисленні опору. Одже, в даному циклі обраховується обернена величина до опору Rc;

* змінній Rc, що відповідає опору Rc присвоюєм значення 0. Перевіряється, чи значення Rc1 більше 0, тоді Rc присвоюємо 1/Rc1. Значення 1/Rc1 рівне значенню опору при паралельному включені;
* для обрахування напруги на виході ЦАП при поданні відповідних двійкових комбінацій на вхід неохідно також врахувати значення опору Rz. Змінна Rz буде рівна цьому опору. Напруга Uz на виході ЦАП буде обчислена в процедурі наступним чином U\_DR=(Rz/(Rz+Rc))\*U\_A. Змінна U\_A рівна значенню напруги, що подається на вхід дільника (3.7 В).
* Значення обрахованої напруги Ur на виході ЦАП запишеться у масив mUr[bOut] (таблиця 4.1).

4.3.2 Процедура DR P\_Write\_DR

Процедура P\_Write\_DR записує байт даних в регістр DR.

P\_Write\_DR – процедура без параметрів. В процедурі оголошено три змінних: bOut типу byte, pDr,nDR типу integer.

Змінній bout привласнюєм спочатку 0, pDr – 1. В циклі з масиву mDR[i] з 4-х молодших розрядів формуємо десяткове число, яке потрібно записати в регістр DR( bout:=bOut+mDR[nDR]\*pDr; pDr:=pDr\*2;).

За тілом циклу записуємо в регістр DR десяткове число, що знаходиться в змінній bout. Для запису використовується процедура PortOut(Adr,bOut) з бібліотеки IO.dll. Ця процедура записує байт даних bOut в порт за адресою Adr. В даному випадку Adr рівне $378.

4.3.3 Процедура P\_Write\_CR

P\_Write\_CR аналогічна процедурі P\_Write\_DR, але дані беруться не з масиву mDR, а з масиву mCR. Дані з масиву mCR інвертуються, крім 2-го елементу масиву і записуються в масив mCR1. Надалі в процедурі відбуваються аналогічні дії як в процедурі P\_Write\_DR. При цьому використовується не масив mDR, а mCR1. Також в процедурі PortOut(Adr,bOut) параметр Adr прийме інше значення.

4.3.4 Процедура читання P\_Read\_SR

Процедура читає дані з регістру SR, використовуючи функцію PortIn(Adr) зовнішньої бібліотеки IO.dll. Функція PortIn(Adr) читає байт даних з LPT-порта за адресою Adr і записує в змінну b. Значення b записується в регістр al. Виконується обнулення всіх бітів крім 4-го біта. Одержаний результат записується в елемент масиву mSR[3].

4.3.5 Процедура B\_In

В циклі у порт записується значення змінної bOut від мінімального значення 0 до максимального 15. Якщо зчитаний сигнал SR[3]=0, то це означає, що напруга з ЦАП більша за вхідну. В такому випадку вхідній напрузі буде відповідати цифрове значення bOut, або в двійковому вигляді масив бітів mDR(Рис 4.2., Додаток В).

bOut=0

Перетворення bOut в массив mDR

Запис bOut в порт з адресою Adr\_DR

Зчитування з порта SR[3]

SR[3]=0

Bout<qB

Ні

Так

U\_In=U\_DR

Ні

Так

Bout=Bout+1

Вивід

mDR[0] - mDR[3]

Рис 4.2. Алгоритм програми керування

Таблиця 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| DR3 | DR2 | DR1 | DR0 | U\_DR, В |
| 0 | 0 | 0 | 0 | 0,06 |
| 0 | 0 | 0 | 1 | 0,72 |
| 0 | 0 | 1 | 0 | 1,2 |
| 0 | 0 | 1 | 1 | 1,55 |
| 0 | 1 | 0 | 0 | 1,63 |
| 0 | 1 | 0 | 1 | 1,88 |
| 0 | 1 | 1 | 0 | 2,03 |
| 0 | 1 | 1 | 1 | 2,21 |
| 1 | 0 | 0 | 0 | 2,22 |
| 1 | 0 | 0 | 1 | 2,34 |
| 1 | 0 | 1 | 0 | 2,42 |
| 1 | 0 | 1 | 1 | 2,53 |
| 1 | 1 | 0 | 0 | 2,57 |
| 1 | 1 | 0 | 1 | 2,64 |
| 1 | 1 | 1 | 0 | 2,70 |
| 1 | 1 | 1 | 1 | 2,78 |

Зв’язок між напругою на виході ЦАП та її цифровим представленням

5. ТЕХНІКО-ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ПРОЕКТУ

Для оцінки техніко-економічного рівня приладу визначимо собівартість, яка обумовлює всі грошові витрати підприємства на виготовлення та реалізацію приладу.

Вартість будь якого радіоелектронного виробу залежить від вартості матеріалів, обладнання та затрат праці на виготовлення деталей і проведення збірно-регулювальних робіт. Ці складові і визначають заводську собівартість пристрою. Розрахунок собівартості на стадії проектування проведемо за калькуляційними статтями витрат. Розглянемо порядок підрахунку всіх калькуляційних статей.

5.1 Витрати на сировину і матеріали

У цьому пункті розраховуються витрати на сировину та матеріали, які використовувались при виготовленні приладу з врахуванням транспортних затрат. Результати цих розрахунків приведені в табл. 5.1. Таким чином, як видно з цієї таблиці, загальні витрати на сировину та матеріали для одного виробу становлять 17,5 грн.

Таблиця 5.1

Витрати на сировину і матеріали

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Найменування,  марка | Профіль,  розмір,мм | Одиниці  виміру | Норма на  прилад | Ціна за  одиницю ваги, грн. | Сума,  грн. |
| Склотекстоліт  СФ2-35-1.5 | 1,5\*125\*175 | кг | 0,050 | 100 | 5 |
| Припой ПОС-61 |  | кг | 0,07 | 43 | 3 |
| Флюс ФСК |  | кг | 0,045 | 133 | 6 |
| Провід ТП-10 |  | м | 1,00 | 3,5 | 3,5 |
| Всього |  |  |  |  | 17,5 |

5.2 Витрати на закупівельні комплектуючі вироби

Підрахуємо витрати на закупівельні комплектуючі вироби, якими є всі електрорадіоелементи. Результати розрахунків затрат приведені в табл. 5.2.

Таблиця 5.2

Витрати на комплектуючі радіоелементи

|  |  |  |  |
| --- | --- | --- | --- |
| Найменування,  Марка | Ціна за одиницю, грн. | Кількість | Сума,  Грн. |
| Резистори |  |  |  |
| МЛТ | 0,20 | 28 | 5,6 |
| Мікросхеми |  |  |  |
| К155ТЛ2 | 2,5 | 3 | 7,5 |
| TL084CN | 2,5 | 1 | 2,5 |
| Діоди |  |  |  |
| КС147А | 1,20 | 4 | 4,80 |
| Світлодіоди |  |  |  |
| АЛ307 | 1 | 6 | 6 |
| Роз’єми |  |  |  |
| 36 контактів | 15,00 | 1 | 15,00 |
| Блок живлення | 20 | 1 | 20 |
| Шнур LPT-порта | 10 | 1 | 10 |
| Всього |  |  | 71,4 |
| Транспортно-заготівельні витрати (6%) |  |  | 4,284 |

5.3. Розрахунок заробітної плати

Розрахунок заробітної плати виробничих працівників проводиться за даними тарифних розцінок. Результати розрахунку зведені в табл. 5.3.

Таблиця 5.3

Розрахунок заробітної плати

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № п/п | Види робіт | Розряд робіт | Трудоємність робіт, год | Тарифна ставка | Заробітна плата | | |
|  |  |  |  |  | пряма | додаткова | всього |
| 1 | Виготовлення друкованих плат | 3 | 1,0 | 3,15 | 3,15 | 0,63 | 3,78 |
| 2 | Сверління | 3 | 0,8 | 2,40 | 1,92 | 0,38 | 2,30 |
| 3 | Установка елементів | 3 | 1,0 | 2,00 | 2,00 | 0,40 | 2,40 |
| 4 | Пайка | 3 | 1,5 | 1,14 | 1,71 | 0,34 | 2,05 |
| 5 | Зборка | 3 | 1,0 | 1,95 | 1,95 | 0,39 | 2,34 |
| 6 | Монтаж | 3 | 1,2 | 1,10 | 1,32 | 0,26 | 1,58 |
| 7 | Регулювання | 4 | 1,5 | 3,20 | 4,80 | 0,96 | 5,76 |
| 8 | Контроль | 4 | 0,9 | 2,00 | 1,80 | 0,36 | 2,16 |
|  | Всього |  |  |  |  |  | 22,37 |

Основна заробітна плата виробничих працівників при виготовленні одного виробу по даних підрахунках становитиме 22,37 грн.

6. ОХОРОНА ПРАЦІ

Законодавчо-правовi аспекти по охоронi працi та охоронi навколишнього середовища дає конституцiя та основи законодавства про працю. Головнi положення про охорону працi закрiпленi конституцiєю та основами законодавства про працю. Конституцiя установлює гарантоване право на працю.

Пiд умовами працi розумiється сукупнiсть факторiв виробничого середовища, якi здiйснюють вплив на здоров’я та працездатність людини. Дослiдження умов працi показали що, факторами виробничого середовища в процесi працi є: санiтарно-гiгiєнiчний стан, який визначає зовнiшнiй стан мiкроклiмату в робочiй зонi; механiчнi коливання; випромiнювання; температура; освiтленiсть та iншi, як результат впливу використовуваного обладнання, матеріалів, технологічних процесiв; психологiчнi елементи; робоча позицiя, фiзичне навантаження, нервово-психологiчне напруження, яке обумовлене самим процесом працi; естетичнi елементи: оформлення виробничих примiщень, обладнання робочого мiсця, робочого iнструменту та iншi.

Паяння – це з’єднання деталей при допомозi припою. Hайчастiше використовуються олов’яно-свинцевi припої. Процес паяння супроводжується забрудненням виробничого середовища, робочих поверхонь, одягу та шкiри рук працюючих свинцем, який є досить токсичною речовиною. Це може привести до отруєння органiзму, викликати змiни у складi кровi, вплинути на нервову систему.

В примiщеннях, де здiйснюється паяння припоєм, який вмiщує свинець, для запобiгання попадання свинцю в органiзм не рекомендується зберiгати особистi речi, приймати їжу, палити, а також прати необхідно робочий одяг вдома. Робоче мiсце для паяння обладнується мiсцевою витяжною вентиляцiєю, що забезпечує концентрацiю свинцю в робочiй зонi не бiльшу за гранично допустиму – 0,01 мг/м .

Для запобiгання опiкiв та заляпування рук чи одягу робiтникiв припоєм, при паяннi повиннi використовуватись спецiальнi серветки для видалення лишнього припою з жала паяльника, а також пiнцет. При монтажних роботах, які пов’язані із ризиком забруднення чи пошкодження очей, передбачена видача робітникам захисних окулярiв.

Для попередження професійних хвороб, необхідно після закінчення роботи сполоснути руки однопроцентним оцтовим розчином, далі вимити їх гарячою водою з милом, прополоскати рот, почистити зуби та прийняти душ.

При травленні застосовуються рiзнi шкiдливi хiмiкати – азотна, фтористоводнева, соляна, сірчана, фосфорна кислоти, рiзнi луги та iншi речовини. При роботі з кислотами на виробництвi можливi нещаснi випадки: отруєння випарами та газами, що видiляються, хiмiчнi опiки при попаданнi кислот на поверхню тiла чи в очi.

Вся робота з агресивними речовинами повинна проводитись в спецiальних шафах з ефективною витяжною вентиляцiєю. Робiтники, які працюють з такими речовинами повиннi бути забезпеченi вiдповiдним спецодягом по затвердженим нормам. Для захисту очей необхiдно використовувати окуляри в гумовiй оправi. Вiдпрацьованi кислоти обов’язково повиннi бути нейтралiзованi у вiдповiдностi з технiчною iнструкцiєю.

Небезпечним для людини є струм промислової частоти 50 Гц силою бiльше 0,1 А. Напруга в електричних колах 40 В та більше вважається небезпечною для життя. Електричний струм при напрузi 220 В може викликати важкi травми i смерть.

Для запобiгання враження електричним струмом необхiдно дотримуватися наступних правил технiки безпеки:

- не можна доторкатися до електричних проводiв, контактних зажимiв, клем електрообладнання, слiд пам’ятати, що вони можуть бути пiд напругою;

* необхiдно слiдкувати, щоб на обслуговуваному обладнаннi було надiйне заземлення i не було вiдкритих ділянок пiд напругою;
* не залишати без нагляду працююче обладнання, при необхiдностi вiдлучитись обов’язково вiдключають обладнання вiд мережi;
* при наявностi враження електричним струмом здiйснити першу медичну допомогу потерпiлому, сповiстити про те, що сталося в медпункт та керiвнику робiт.

Освiтлення достатнє i повинно вiдповiдати СH i П II- 4 -79. Потрiбна освiтленiсть визначається характером виконуваної роботи i дiючими санiтарними нормами. При природньому i штучному освiтленi робочi мiсця розмiщенi таким чином, щоб свiтло падало злiва. Такi умови вважаються оптимальними.

Розмiщення для електроустаткування і електропроводки iзоляцiї забезпечують повну безпеку працюючого.

Шкiдливi речовини, якi видiляються при роботi, видаляються за допомогою мiсцевої вентиляцiї. Розмiщення пристроїв i трубопроводiв для вiдведення повiтря виключає можливiсть травматизму.

Зовнiшнє оформлення робочих мiсць в виробничих примiщеннях вiдповiдає вимогам виробничої естетики. Кольори для фарбування поверхонь виробничих примiщень i устаткування, які розмiщені в колi зору працюючого, повинні бути свiтло-голубими, що забезпечує кращу видимiсть деталей i сприятливо дiє на нервову систему людини.

В комплекс заходів по створенню високопродуктивної працi входять i психофiзiологiчнi заходи, якi забезпечують безперервний ріст продуктивностi працi при мiнiмальних витратах фiзичної i нервової енергiї працюючого, i направленi на створення оптимального режиму працi та вiдпочинку із врахуванням конкретного процесу виробництва.

Виробничий процес збирання, монтажу та регулювання приладу вiдноситься по ступеню пожежної небезпечностi до категорiї Д.

Причинами пожежi можуть бути: коротке замикання, перевантаження провiдникiв мережi, виникнення великих перехiдних опорiв, якi порушують нормальну роботу технологiчного обладнання. Тому передбачаються пристрої, що забезпечують вiд’єднання при короткому замиканнi як всiєї мережi, так i окремих її ділянок.

Електроосвiтлення створює пожежну небезпеку при перегрiванні провiдникiв та iзоляцiї.

Для захисту провiдникiв вiд механiчних та хiмiчних пошкоджень їх прокладають в гумових або металевих трубках, якi мають внутрiшню iзоляцiю.

При русi гумової стрiчки транспортеру можуть виникати електростатичнi заряди. При незначнiй вологостi повiтря можливий iскровий розряд мiж частинами обладнання, або розряд на землю.

Енергiя такої iскри незначна, але може бути достатньою для пiдпалення горючої або вибухонебезпечної сумiшi. Таким чином, виникаючi в виробничих умовах електростатичнi заряди, можуть бути джерелами пожежi.

Уникнути небезпеки виникнення електростатичних зарядiв можна наступними методами:

1) заземлення виробничого обладнання та ємностей для зберiгання вогненебезпечних рiдин;

2) збiльшення вологостi повiтря або використання антистатичних домішок до основного продукту.

Джерелами пожежi можуть бути легкозаймистi рiдини: спирт, бензин, ацетон та ін. тому їх запас на виробничій ділянці не повинен перевищувати потреби на одну робочу змiну. Для зберiгання цих рiдин та їх вiдходiв повинна бути передбачена спецiальна тара - алюмiнієвi бiдони з притертими кришками, з чiткою назвою рiдини та попереджувальним надписом "ВОГHЕHЕБЕЗПЕЧНО!".

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Архангельський А.Я. Програмирование в Delphi 7. – М.: Бином, 2004. 1152 с.
2. Бобровский С. Delphi 6 и Kylix: Библиотека программиста. – СПб.: Питер, 2002. – 560 с.
3. Глинський Я.М., Анохін В.Є., Ряжська В.А. Паскаль. Turbo Pascal і Delphi. 4-те вид. – Львів: Деол, СПД Глинський, 2003. – 144 с.
4. Программирование портов в Delphi. - amberv.narod.ru/delphi1.html
5. Зубчук В.И. Справочник по цифровой схемотехнике. – К.: Техника, 1990. – 448 с.
6. Новиков Ю.В., Калашников О.А., Гуляэв С.Э. Разработка устройств сопряжения для персонального компьютера типа IBM PC. – М.: ЭКОМ, 1997. – 224 с.
7. Гук М. Аппаратные интерфейсы ПК. Энциклопедия. – СПб.: Питер, 2002. – 528 с.
8. Ковальчук В.М. Основи економічної теорії. – Тернопіль: 2003. – 203 с.

ДОДАТКИ

Додаток А

Текст програми

unit ADC\_2007\_U;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ExtCtrls, TeeProcs, TeEngine, Chart, Series, Menus;

Type TF\_ADC = class(TForm)

B\_Write\_DR: TButton;

B\_Read\_SR: TButton;

L\_Driver\_LPT: TLabel;

CheckBox\_DR3: TCheckBox;

CheckBox\_DR2: TCheckBox;

CheckBox\_DR1: TCheckBox;

CheckBox\_DR0: TCheckBox;

Bevel\_DR: TBevel;

CheckBox\_CR3: TCheckBox;

CheckBox\_CR2: TCheckBox;

CheckBox\_CR1: TCheckBox;

CheckBox\_CR0: TCheckBox;

Bevel\_CR: TBevel;

B\_Write\_CR: TButton;

CheckBox\_SR3: TCheckBox;

Bevel\_SR: TBevel;

B\_In\_U: TButton;

B\_Out\_U: TButton;

E\_In\_U: TEdit;

E\_Out\_U: TEdit;

B\_Exit: TButton;

B\_Diagram: TButton;

E\_QI: TEdit;

L\_QI: TLabel;

Chart1: TChart;

Series1: TFastLineSeries;

B\_Stop: TButton;

E\_Time\_Discret: TEdit;

L\_Time\_Discret: TLabel;

B\_Save: TButton;

SaveDialog1: TSaveDialog;

CheckBox\_ADC\_Proces\_Show: TCheckBox;

RadioGroup\_Language: TRadioGroup;

MainMenu1: TMainMenu;

File1: TMenuItem;

Save1: TMenuItem;

ADC1: TMenuItem;

InU1: TMenuItem;

OutU1: TMenuItem;

Port1: TMenuItem;

WriteDR1: TMenuItem;

Write\_CR1: TMenuItem;

procedure B\_Write\_DRClick(Sender: TObject);

procedure B\_Read\_SRClick(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure B\_Write\_CRClick(Sender: TObject);

procedure B\_Out\_UClick(Sender: TObject);

procedure B\_In\_UClick(Sender: TObject);

procedure B\_ExitClick(Sender: TObject);

procedure B\_DiagramClick(Sender: TObject);

procedure B\_StopClick(Sender: TObject);

procedure B\_SaveClick(Sender: TObject);

procedure RadioGroup\_LanguageClick(Sender: TObject);

procedure Save1Click(Sender: TObject);

procedure InU1Click(Sender: TObject);

procedure OutU1Click(Sender: TObject);

procedure WriteDR1Click(Sender: TObject);

procedure Write\_CR1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

CONST

QI\_Max=10000;

var

F\_ADC: TF\_ADC;

qDR,qSR,qCR:integer;

Adr\_DR,Adr\_CR,Adr\_SR:word;

qB:integer;

U\_Out,U\_In:real;

U\_A,U\_min:real;

// U\_Out\_Max,U\_In\_Max:real;

RegStop:integer;

QI:integer;

Rdiod,Rz:real;

mR:array[0..8] of real;

mUr:array[0..16] of real;

mDR,mSR,mCR,mCR1:array[0..8] of byte;

mx,my:array[1..qI\_Max] of real;

implementation

{$R \*.dfm}

function IsDriverInstalled : Boolean; stdcall; external 'io.dll';

procedure PortOut(Port : Word; Data : Byte);stdcall; external 'io.dll';

function PortIn(Port : Word) : Byte;stdcall; external 'io.dll';

// CREATE FORM ============

Procedure Trans\_bOut\_mDR(bOut:byte);

{ Процедура перетворює байт bOut в набір з 8 бітів,

які записуються у масив mDR[0..7], де біт mDr[0] - молодший}

Var b1,b2,b3:byte;

i:integer;

Begin

b3:=bOut;

for i:=0 to 7 do

begin

b1:= b3 div 2;

b2:=b3 mod 2;

b3:=b1;

mDR[i]:=b2;

end;

// Примітка. Відображати біти на екрані наступним чином: D7 D6 D5 D4 D3 D2 D1 D0

End;

Procedure p\_Init\_U;

Var nDr,bOut:integer;

Rc1,Rc,Ur:real;

Begin

for bOut:=0 to qB do

begin

Trans\_bOut\_mDR(bOut);

Rc1:=0;

for nDR:=0 to qDR do

begin

Rc1:=Rc1+(1/(mR[nDr]+Rdiod))\*mDR[nDr];

end;

rc:=0;

if Rc1>0 then Rc:=1/Rc1;

Ur:=(Rz/(Rz+Rc))\*U\_A;

if bOut=0 then Ur:=U\_min;

mUr[bOut]:=Ur;

// ShowMessage(' bOut= '+IntToStr(bOut)+' Rc= '+FloatToStrF(Rc,ffFixed,6,2)+' Ur= '+FloatToStrF(Ur,ffFixed,6,2));

end; // bOut

End;

procedure TF\_ADC.FormCreate(Sender: TObject);

begin

qDR:=3;

qCR:=3;

qSR:=1;

Adr\_DR:=$378;

Adr\_SR:=$379;

Adr\_CR:=$37A;

qB:=15;

U\_A:=3.7;

U\_min:=0.14;

RDiod:=300;

mR[3]:=2200;

mR[2]:=4700;

mR[1]:=8200;

mR[0]:=15000;

Rz:=4600;

p\_Init\_U;

if IsDriverInstalled then

begin

L\_Driver\_LPT.Caption:='Driver Ok!';

end;

end;

// EXIT ================

procedure TF\_ADC.B\_ExitClick(Sender: TObject);

begin

Application.Terminate;

end;

// == DR ==== BEGIN ===================

Procedure p\_Write\_DR;

Var bOut:byte;

pDr,nDR:integer;

Begin

bout:=0;

pDr:=1;

for nDR:=0 to qDR do

begin

bout:=bOut+mDR[nDR]\*pDr;

pDr:=pDr\*2;

end;

PortOut(Adr\_DR,bOut);

End;

procedure TF\_ADC.B\_Write\_DRClick(Sender: TObject);

Var nDR:integer;

begin

for nDR:=0 to qDR do

begin

mDR[nDR]:=0;

end;

if CheckBox\_DR0.Checked then mDR[0]:=1;

if CheckBox\_DR1.Checked then mDR[1]:=1;

if CheckBox\_DR2.Checked then mDR[2]:=1;

if CheckBox\_DR3.Checked then mDR[3]:=1;

p\_Write\_DR;

end;

// == DR ==== END ===================

// === CR ==== BEGIN ===================

Procedure p\_Write\_CR;

Var bOut:byte;

pCr,nCR:integer;

Begin

// Invers

for nCR:=0 to qCR do

begin

if nCR<>2 then

begin

mCR1[nCR]:=0;

if mCR[nCR]=0 then mCR1[nCR]:=1;

end;

if nCR=2 then mCR1[nCR]:=mCR[nCR];

end;

bout:=0; pCr:=1;

for nCR:=0 to qCR do

begin

bout:=bOut+mCR1[nCR]\*pCr;

pCr:=pCr\*2;

end;

PortOut(Adr\_CR,bOut);

End;

procedure TF\_ADC.B\_Write\_CRClick(Sender: TObject);

Var nCR:integer;

begin

for nCR:=0 to qCR do

begin

mCR[nCR]:=0;

end;

if CheckBox\_CR0.Checked then mCR[0]:=1;

if CheckBox\_CR1.Checked then mCR[1]:=1;

if CheckBox\_CR2.Checked then mCR[2]:=1;

if CheckBox\_CR3.Checked then mCR[3]:=1;

p\_Write\_CR;

end;

// === CR ==== END ===================

// == SR == BEGIN ===============

Procedure p\_Read\_SR;

Var b,b1:byte;

Begin

b:=PortIn(Adr\_SR);

asm

mov al,b

and al,00001000b

mov b1,al

end;

mSR[3]:=0;

if b1>0 then mSR[3]:=1;

End;

procedure TF\_ADC.B\_Read\_SRClick(Sender: TObject);

begin

p\_Read\_SR;

if mSR[3]=0 then CheckBox\_SR3.checked:=false;

if mSR[3]=1 then CheckBox\_SR3.checked:=true;

end;

// == SR == END ===============

// Out\_U BEGIN =============

Procedure Tran\_bOut\_mCR(bOut:byte);

{ Процедура перетворює байт bOut в набір з 8 бітів,

які записуються у масив mCR[0..7], де біт mCr[0] - молодший}

Var b1,b2,b3:byte;

i:integer;

Begin

b3:=bOut;

for i:=0 to 7 do

begin

b1:= b3 div 2;

b2:=b3 mod 2;

b3:=b1;

mCR[i]:=b2;

end;

// Примітка. Відображати біти на екрані наступним чином: D7 D6 D5 D4 D3 D2 D1 D0

End;

procedure TF\_ADC.B\_Out\_UClick(Sender: TObject);

Var bOut:byte;

Ur:real;

begin

U\_Out:=StrToFloat(E\_Out\_U.Text);

for bOut:=0 to qB do

begin

Ur:=mUr[bOut];

// ShowMessage(' bOut= '+IntToStr(bOut)+' Ur= '+FloatToStrF(Ur,ffFixed,6,2));

if Ur>U\_Out then break;

end;

// bOut:=round(r1);

Tran\_BOut\_mCR(bOut);

p\_Write\_CR;

if mCR[0]=1 then CheckBox\_CR0.Checked:=true

else CheckBox\_CR0.Checked:=false;

if mCR[1]=1 then CheckBox\_CR1.Checked:=true

else CheckBox\_CR1.Checked:=false;

if mCR[2]=1 then CheckBox\_CR2.Checked:=true

else CheckBox\_CR2.Checked:=false;

if mCR[3]=1 then CheckBox\_CR3.Checked:=true

else CheckBox\_CR3.Checked:=false;

end;

// Out\_U END =============

// In\_U BEGIN ===================

procedure TF\_ADC.B\_In\_UClick(Sender: TObject);

Var bOut:byte;

Ur:real;

begin

U\_In:=0;

for bOut:=0 to qB do

begin

Trans\_bOut\_mDR(bOut);

PortOut(Adr\_DR,bOut);

if mDR[0]=1 then CheckBox\_DR0.Checked:=true

else CheckBox\_DR0.Checked:=false;

if mDR[1]=1 then CheckBox\_DR1.Checked:=true

else CheckBox\_DR1.Checked:=false;

if mDR[2]=1 then CheckBox\_DR2.Checked:=true

else CheckBox\_DR2.Checked:=false;

if mDR[3]=1 then CheckBox\_DR3.Checked:=true

else CheckBox\_DR3.Checked:=false;

p\_Read\_SR;

Ur:=mUr[bOut]; // oporna for Dr3 - DR0

if mSR[3]=0 then

begin

U\_In:=Ur;

break;

end;

if bOut=qB then

begin

if mSR[3]>0 then U\_In:=U\_A;

end;

if CheckBox\_ADC\_Proces\_Show.checked then

ShowMessage(' bOut= '+IntToStr(bOut)+' Ur= '+FloatToStrF(Ur,ffFixed,6,2));

// Sleep(5);

end;

E\_In\_U.text:=FloatToStrF(U\_In,ffFixed,4,2);

end;

// In\_U END ===================

procedure TF\_ADC.B\_DiagramClick(Sender: TObject);

Var i:integer;

x,y:real;

Time\_Discret:integer;

begin

RegStop:=0;

CheckBox\_ADC\_Proces\_Show.Checked:=false;

QI:=StrToInt(E\_QI.Text);

if QI>QI\_Max then

begin

ShowMessage('QI>QI\_Max= '+IntToStr(QI\_Max));

QI:=QI\_Max;

end;

Time\_Discret:=StrToInt(E\_Time\_Discret.Text);

Chart1.Series[0].Clear;

for i:=1 to QI do

begin

B\_In\_UClick(Sender);

x:=(i-1)\*Time\_Discret/1000;

y:=U\_In;

mx[i]:=x;

my[i]:=y;

// if i=1 then y:=y+0.0001;

Chart1.Series[0].AddXY(x,y);

Sleep(Time\_Discret);

Application.ProcessMessages;

if RegStop=1 then break;

end;

end;

procedure TF\_ADC.B\_StopClick(Sender: TObject);

begin

RegStop:=1;

end;

procedure TF\_ADC.B\_SaveClick(Sender: TObject);

Var flName:string;

tx:TextFile;

i:integer;

begin

if SaveDialog1.Execute then

begin

flName:=SaveDialog1.FileName+'.txt';

AssignFile(tx,FlName);

ReWrite(tx);

for i:=1 to QI do

begin

write(tx,mx[i]:7:3);

writeln(tx,my[i]:7:3);

end; // i

CloseFile(tx);

end;

end;

procedure TF\_ADC.RadioGroup\_LanguageClick(Sender: TObject);

begin

if RadioGroup\_Language.ItemIndex=1 then

Begin

B\_Write\_DR.Caption:='Записати\_DR';

B\_Write\_CR.Caption:='Записати\_CR';

B\_In\_U.Caption:='Вхід\_U';

B\_Out\_U.Caption:='Вихід\_U';

B\_Diagram.Caption:='Діаграма';

B\_Stop.Caption:='Стоп';

B\_Save.Caption:='Зберегти';

B\_Exit.Caption:='Вихід';

B\_Read\_SR.Caption:='Читати\_SR';

CheckBox\_ADC\_Proces\_Show.Caption:='ADC\_Показати\_Процес';

L\_Time\_Discret.Caption:='Проміжки\_Часу (мс)';

MainMenu1.Items[0].Caption:='Файл';

MainMenu1.Items[0].Items[0].Caption:='Зберегти';

MainMenu1.Items[1].Caption:='ADC';

MainMenu1.Items[1].Items[0].Caption:='Вхід\_U';

MainMenu1.Items[1].Items[1].Caption:='Вихід\_U';

MainMenu1.Items[2].Caption:='Порт';

MainMenu1.Items[2].Items[0].Caption:='Записати\_DR';

MainMenu1.Items[2].Items[1].Caption:='Записати\_CR';

end;

if RadioGroup\_Language.ItemIndex=0 then

Begin

B\_Write\_DR.Caption:='Write\_DR';

B\_Write\_CR.Caption:='Write\_CR';

B\_In\_U.Caption:='In\_U';

B\_Out\_U.Caption:='Out\_U';

B\_Diagram.Caption:='Diagram';

B\_Stop.Caption:='Stop';

B\_Save.Caption:='Save';

B\_Exit.Caption:='Exit';

B\_Read\_SR.Caption:='Read\_SR';

CheckBox\_ADC\_Proces\_Show.Caption:='ADC\_Proces\_Show';

L\_Time\_Discret.Caption:='Time\_Discret (ms)';

MainMenu1.Items[0].Caption:='File';

MainMenu1.Items[0].Items[0].Caption:='Зберегти';

MainMenu1.Items[1].Caption:='ADC';

MainMenu1.Items[1].Items[0].Caption:='In\_U';

MainMenu1.Items[1].Items[1].Caption:='Out\_U';

MainMenu1.Items[2].Caption:='Port';

MainMenu1.Items[2].Items[0].Caption:='Write\_DR';

MainMenu1.Items[2].Items[1].Caption:='Write\_CR';

end;

end;

procedure TF\_ADC.Save1Click(Sender: TObject);

begin

B\_Save.Click;

end;

procedure TF\_ADC.InU1Click(Sender: TObject);

begin

B\_In\_U.Click;

end;

procedure TF\_ADC.OutU1Click(Sender: TObject);

begin

B\_Out\_U.Click;

end;

procedure TF\_ADC.WriteDR1Click(Sender: TObject);

begin

B\_Write\_DR.Click;

end;

procedure TF\_ADC.Write\_CR1Click(Sender: TObject);

begin

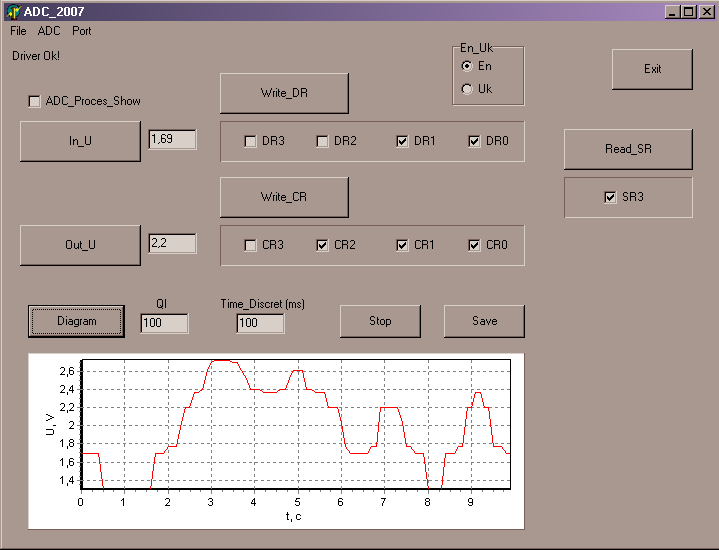
B\_Write\_CR.Click;

end;

end.

Додаток Б

Тестування перетворювача



Додаток В

Принцип роботи АЦП

0

1

2

3

4

5

Uвх

U, В

bOut

U\_DR