Лабораторна робота № 4

Дослідження аналого-цифрових інформаційних перетворювальних каскадів

Мета роботи: вивчення особливостей функціонування аналого-цифрових перетворювачів та їх взаємодії з мікроконтролерними пристроями керування; ознайомлення з пристроями відображення виміряної інформації.

Короткі теоретичні відомості

Аналого-цифрові перетворювачі (АЦП) це пристрої, які приймають вхідні аналогові сигнали та генерують відповідні до них цифрові сигнали, які придатні для обробки мікропроцесорами та іншими цифровими пристроями.

Процедура аналого-цифрового перетворення неперервних сигналів, яку реалізовують за допомогою АЦП, це перетворення неперервної функції часу , яка описує вхідний сигнал, у послідовність чисел , що віднесені до деяких фіксованих моментів часу. Цю процедуру можна розділити на дві самостійні операції: дискретизацію і квантування.

Відома велика кількість методів аналого-цифрового перетворення. Ці методи суттєво відрізняються один від одного потенційною точністю, швидкістю перетворення та складністю апаратної реалізації. На рис. 1 наведена класифікація АЦП за методами перетворення.

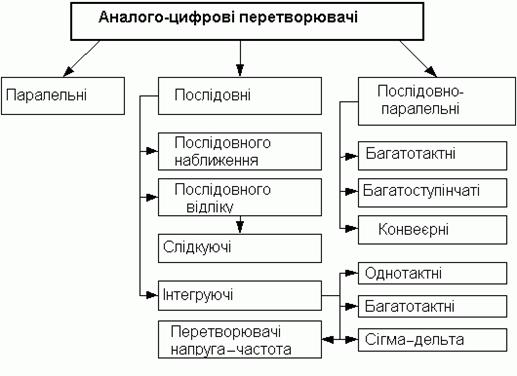


Рис. 1. Класифікація методів аналого-цифрового перетворення

В основу класифікації АЦП покладено ознаку, яка вказує на те, як в часі розгортається процес перетворення аналогової величини в цифрову. В основі перетворення вибіркових значень сигналу в цифрові еквіваленти лежать операції квантування та кодування. Вони можуть проводитись за допомогою або послідовної, або паралельної, або послідовно-паралельної процедур наближення цифрового еквівалента до перетворюваної величини.

Розглянемо детальніше найбільш поширені типи АЦП.

Функціонування аналого-цифрового перетворення за методом послідовного підрахунку можна проілюструвати за допомогою структурної схеми на рис. 2.

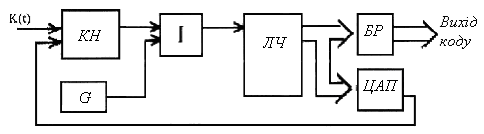


Рис. 2. АЦП послідовного підрахунку

До складу схеми входять: генератор тактових сигналів (G), компаратор напруги (КН), схема І, лічильник (ЛЧ), буферний регістр (БР), цифро-аналоговий перетворювач (ЦАП). Схема працює наступним чином. На вхід перетворювача подається аналоговий сигнал , який підключається до одного з входів компаратора напруги КН. На другий вхід компаратора подається еталонна напруга , яка формується на виході ЦАП під управлінням колового слова на виході ЛЧ. Компаратор формує на своєму виході сигнал або логічної одиниці, або логічного нуля в залежності від того, яке значення більше. Якщо , на виході компаратора формується одиниця, яка дозволяє проходження імпульсів з тактового генератора через схему І на лічильний вхід лічильника ЛЧ. На виході лічильника йде процес підрахунку цих імпульсів в двійковому коді від 20 до 2n-1. Двійковий код з ЛЧ подається на вхід ЦАП, на виході якого формується ступінчатий сигнал . Кожна сходинка цього сигналу відповідає за рівнем інтервалу дискретизації . Сигнал порівнюється із сигналом  і в момент, коли  стає меншим за , на виході компаратора формується сигнал логічного нуля. Схема І закривається, лічильник зупиняє підрахунок і набраний двійковий код переписується у вихідний буферний регістр БР для видачі користувачу.

Метод безпосереднього зчитування реалізовується за допомогою так званого АЦП паралельної дії. Такий перетворювач має лінійку 2n-1 компараторів напруги, перші входи яких запаралелені і на них подається сигнал . На інші входи підключені виходи подільника еталонної напруги. Виходи компараторів під’єднані до перетворювача одиничного коду в двійковий. Процес перетворення здійснюється за один такт, причому на виході лінійки компараторів до компаратора, який зафіксує  буде послідовність одиниць, а далі послідовність нулів одиничного коду. Структурна схема перетворення зображена на рис. 3

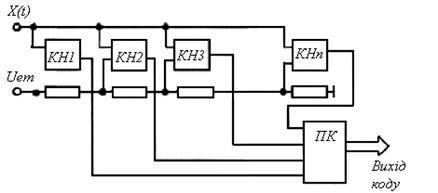


Рис. 3. АЦП безпосереднього зчитування

Найбільше поширення отримав метод порозрядного зрівноваження, який забезпечує час перетворення від 1 мкс до 1 мс. Структурна схема перетворювача зображена на рис. 4, а часова діаграма – на рис. 5.

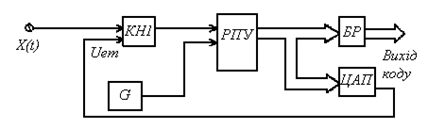


Рис. 4. АЦП порозрядного врівноваження

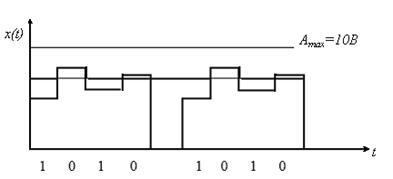


Рис. 5. Часова діаграма перетворення АЦП за методом порозрядного зрівноваження при *n*=4

Схема працює так. На вхід АЦП подається вхідний сигнал , який порівнюється з еталонним сигналом , що формується на виході ЦАП. ЦАП складається із сукупності 3*n* еталонних джерел сигналів, які управляються за допомогою спеціального регістра порозрядного зрівноваження (РПУ). Перетворення проходить за *n* часових тактових інтервалів. Причому на першому такті РПУ примусово вмикає в роботу перший розряд ЦАП. Значення першого розряду еталонних величин на виході ЦАП дорівнює половині діапазону перетворення сигналу. Потім в кінці першого тактового інтервалу компаратор проводить порівняння  з . Якщо <, то примусово увімкнений старший розряд ЦАП залишається ввімкненим до закінчення процесу перетворення. Це забезпечується під управлінням певного сигналу на виході компаратора (1чи 0). Якщо ж >, то перший розряд вимикається на початку другого такту. На початку другого такту в роботу примусово вмикається другий розряд ЦАП і знову проводиться порівняння  з . Процедура повторюється доти, поки всі *n* розрядів ЦАП не візьмуть участі у процесі зрівноваження. В результаті на виході АЦП формується код, що відповідає вхідному сигналу.

Відомо, що недоліком послідовних АЦП є низька завадостійкість результатів перетворення. Дійсно, вибірка миттєвого значення вхідної напруги, переважно включає доданок у вигляді миттєвого значення завади. Згодом при цифровій обробці послідовності вибірок ця складова може бути подавлена, однак на це потрібен час та обчислювальні ресурси. Переважно у АЦП вхідний сигнал інтегрується або неперервно, або у певному часовому діапазоні, тривалість якого зазвичай вибирається кратною періодові завади. Це дозволяє в багатьох випадках приглушити заваду ще на етапі перетворення. Платою за це є понижена швидкодія інтегруючих АЦП.

Спрощена схема АЦП, який працює в два основних такти (АЦП двотактного інтегрування), наведена на рис. 6.

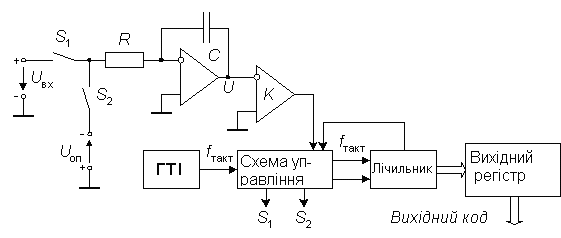


Рис. 6. Спрощена схема АЦП двотактного інтегрування

Перетворення проходить протягом двох стадій: стадії інтегрування та стадії підрахунку. На початку першої стадії ключ S1 замкнутий, а ключ S2 розімкнутий. Інтегратор інтегрує вхідну напругу . Час інтегрування вхідної напруги  постійний. Як таймер використовується лічильник з коефіцієнтом підрахунку , так, що .

До моменту закінчення інтегрування вихідна напруга інтегратора складає

,

де  – середнє за час *t*1 значення вхідної напруги.

Після закінчення стадії інтегрування ключ S1 розмикається, а ключ S2 замикається та опорна напруга  надходить на вхід інтегратора. При цьому вибирається опорна напруга, протилежна за знаком вхідній напрузі. На стадії підрахунку вихідна напруга інтегратора лінійно зменшується за абсолютною величиною, як показано на рис. 7.

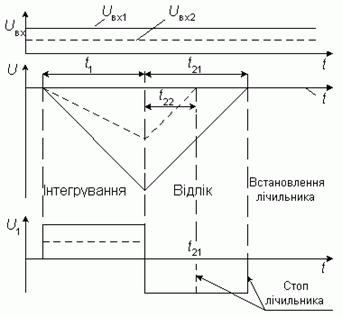


Рис. 7. Часові діаграми АЦП двотактного інтегрування

Стадія підрахунку закінчується, коли вихідна напруга інтегратора переходить через нуль. При цьому компаратор К переключається та підрахунок зупиняється. Діапазон часу, у якому проходить стадія підрахунку, визначається рівнянням

.

Далі, виконавши прості математичні дії і врахувавши, що:

,

де *n*2 – вміст лічильника після закінчення стадії підрахунку, отримаємо результат

.

З цієї формули випливає, що відмітною рисою методу багатотактного інтегрування є те, що ні тактова частота, ні постійна інтегрування *RC* не впливають на результат. Необхідно тільки, щоб тактова частота протягом часу *t*1+*t*2 залишалася постійною. Це можна забезпечити при використанні простого тактового генератора, оскільки істотні часові чи температурні дрейфи частоти відбуваються за час який більший, ніж час перетворення.

Основними параметрами і характеристиками АЦП є наступні:

- число розрядів *п* вихідного коду;

- роздільна здатність *h* – мінімальний квант вхідної напруги, за якої вихідний код змінюється на одиницю молодшого розряду;

- нелінійність (*δЛ* – максимальне відхилення вихідного коду від розрахункового значення у всьому діапазоні шкали;

- абсолютна похибка *δА* – найбільше відхилення вихідного коду від розрахун­кового в кінцевій точці шкали;

- час перетворення *tпр* – інтервал від моменту початку перетворення до по­яви на виході сталого коду; часто замість *tпр* швидкодія АЦП характеризу­ється частотою перетворення;

- діапазон і полярність вхідної напруги, число джерел живлення, струм споживання, можливість спільної роботи з мікропроцесорами.

Завдання до виконання

1. Відповідно до варіанту завдання вибрати мікроконтролер та розробити для нього програмне забезпечення на мові програмування високого рівня.
2. З використанням вибраного мікроконтролера реалізувати віртуальну модель аналого-цифрового перетворювального каскаду та перевірити його роботу.

Варіанти для вибору мікроконтролерів

|  |  |
| --- | --- |
| **№ варіанту** | **Тип мікроконтролера** |
| 1 | ATMEGA128 |
| 2 | ATMEGA1281 |
| 3 | ATMEGA162 |
| 4 | ATMEGA165 |
| 5 | ATMEGA168 |
| 6 | ATMEGA169 |
| 7 | ATMEGA32 |
| 8 | ATMEGA324P |
| 9 | ATMEGA325 |
| 10 | ATMEGA3250 |
| 11 | ATMEGA329 |
| 12 | ATMEGA48 |
| 13 | ATMEGA64 |
| 14 | ATMEGA644 |
| 15 | ATMEGA645 |
| 16 | ATMEGA8 |
| 17 | ATMEGA8515 |
| 18 | ATMEGA88 |
| 19 | ATMEGA1280 |
| 20 | ATMEGA1284P |
| 21 | ATMEGA164P |
| 22 | ATMEGA3290P |
| 23 | ATMEGA640 |
| 24 | ATMEGA6450 |
| 25 | ATMEGA649 |
| 26 | ATMEGA6490 |
| 27 | ATMEGA8535 |

# Виконання:

Реалізована схема



# Код реалізації:

# Main.c

#include "main.h"

char adc\_string[1000];

void port\_init(void)

{

PORTD=0x00;

DDRD=0xFF;

}

int main(void)

{

unsigned int adc\_value;

float n;

port\_init();

LCD\_init();

ADC\_Init ();

LCD\_clear();

set\_position(0,0);

while(1)

{

adc\_value = ADC\_convert();

n = (adc\_value \* 5.0 / 1024.0);

set\_position(0,0);

*sprintf*(adc\_string, "%.2u", adc\_value);

str\_lcd(adc\_string);

}

}

# LCD.c

#include "lcd.h"

void send\_half\_byte(unsigned char c)

{

c<<=4;

e1;

*\_delay\_us*(50);

PORTD&=0b00001111;

PORTD|=c;

e0;

*\_delay\_us*(50);

}

void send\_byte(unsigned char c, unsigned char mode)

{

if (mode==0) rs0;

else rs1;

unsigned char hc=0;

hc=c>>4;

send\_half\_byte(hc);

send\_half\_byte(c);

}

void send\_char\_lcd(unsigned char c)

{

send\_byte(c,1);

}

void set\_position(unsigned char x, unsigned y)

{

char adress;

adress=(0x40\*y+x)|0b10000000;

send\_byte(adress, 0);

}

void LCD\_init(void)

{

*\_delay\_ms*(15); //Ждем 15 мс (стр 45)

send\_half\_byte(0b00000011);

*\_delay\_ms*(4);

send\_half\_byte(0b00000011);

*\_delay\_us*(100);

send\_half\_byte(0b00000011);

*\_delay\_ms*(1);

send\_half\_byte(0b00000010);

*\_delay\_ms*(1);

send\_byte(0b00101000, 0); //4 bit mode

*\_delay\_ms*(1);

send\_byte(0b00001100, 0);

*\_delay\_ms*(1);

send\_byte(0b00000110, 0); //left mode

*\_delay\_ms*(1);

}

void LCD\_clear(void)

{

send\_byte(0b00000001, 0);

*\_delay\_us*(1500);

}

void str\_lcd (char str1[])

{

wchar\_t n;

for(n=0;str1[n]!='\0';n++)

send\_char\_lcd(str1[n]);

}

# ADC.c

#include "adc.h"

void ADC\_Init(void)

{

ADCSRA |= (1<<ADEN) // Access using ADC

|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)| (1 << ADATE);//Frequency Devider

ADMUX |= (1<<REFS1)|(1<<REFS0); //Source, enable on the port ADC0

}

unsigned int ADC\_convert (void)

{

unsigned int result = 0;

ADMUX = (ADMUX & 0XF0) | (0X0F & 0); // Enable conversion on selected ADC channel

*\_delay\_us*(400);

ADCSRA |= (1<<ADSC); // Start ADC Conversion

while((ADCSRA & (1<<ADIF))); // Wait till conversion is complete

result = ADC; // Read the ADC Result

ADCSRA |= (1 << ADIF); // Clear ADC Conversion Interrupt Flag

return result;

}