

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

МІКРОПРОЦЕСОРИ

Методичні вказівки до лабораторних робіт з дисципліни
«Мікропроцесори в системах та пристроях» для студентів фаху
7.090701 «радіотехніка» усіх форм навчання

2010

МІКРОПРОЦЕСОРИ. Методичні вказівки до лабораторних робіт з дисципліни "Мікропроцесори в системах та пристроях" для студентів фаху 7.090701 «радіотехніка» усіх форм навчання.

/ Укл. В.І. Мисленков - ЗНТУ. - Запоріжжя, 2010. – 66 с.

Укладачі: ст. викладач В.І. Мисленков,

Рецензент: ст. викладач Ю.Л. Мейстер

Відповідальний
за випуск: зав. лабораторією В.П. Колесніков

Затверджено:
на засіданні кафедри
«радіотехніки та телекомунікацій»
Протокол № 10__ від 22 06__ 2010

ЗМІСТ

Вимоги до оформлення звіту з лабораторної роботи.....	5
<u>1 Лабораторна робота №1</u>	
<u>Визначення продуктивності процесора INTEL8031</u>	<u>5</u>
1.1 Порядок виконання лабораторної роботи.....	5
1.2 Короткі теоретичні відомості.....	5
1.3 Схема програми, що використовує таймер для виміру часу	8
1.4 Контрольні питання.....	8
<u>2 Лабораторна робота №2</u>	
<u>Програмування паралельного інтерфейсу K580BB55.</u>	
<u>Схеми відображення інформації</u>	<u>9</u>
2.1 Порядок виконання лабораторної роботи.....	9
2.2 Короткі теоретичні відомості.....	9
2.3 Приклад програми для статичної індикації	13
2.4 Варіанти індивідуальних завдань.....	14
2.5 Контрольні питання.....	15
<u>3 Лабораторна робота №3</u>	
<u>Обробка частотних і часових сигналів</u>	<u>16</u>
3.1 Порядок виконання лабораторної роботи.....	16
3.2 Короткі теоретичні відомості.....	16
<u>3.3 Приклад програми виміру частот</u>	<u>19</u>
3.4 Варіанти індивідуальних завданні	21
3.5 Контрольні питання.....	22
<u>4 Лабораторна робота №4</u>	
<u>Обмін інформацією через послідовний порт</u>	<u>22</u>
4.1 Порядок виконання роботи	23
4.2 Короткі теоретичні відомості.....	23
4.3 Варіанти завдань.....	25
4.4 Контрольні питання.....	26
5 Лабораторна робота № 5	26
Архітектура ADSP-2181	26
5.1 Завдання.....	26
5.2 Порядок виконання роботи	26
5.3 Контрольні питання.....	30
6 Лабораторна робота № 6	31
Арифметика ЦПОС з фіксованою крапкою	31
6.1 Завдання.....	31

6.2	Порядок виконання роботи	312
6.3	Контрольні питання.....	355
7	Лабораторна робота № 7	355
Учебний	стенд	ADSP2181
		EZ-KIT
		Lite
		355
7.1	Завдання	366
7.2	Порядок виконання роботи	366
7.3	Контрольні питання.....	39
8	Лабораторна робота №8	39
	Мова асемблера ЦПОС ADSP-2181.....	39
8.1	Завдання	39
8.2	Порядок виконання роботи	40
8.3	Контрольні питання.....	422
9	Лабораторна робота № 9	433
	Математичні обчислення на ЦПОС ADSP-2181	433
9.1	Завдання	433
9.2	Порядок виконання роботи	433
9.3	Контрольні питання.....	477
10	Лабораторна робота № 10.....	477
	Проектування FIR-фільтрів на ЦПОС ADSP-2181.....	477
10.1	Завдання	477
10.2	Порядок виконання роботи	488
10.3	Контрольні питання.....	511
	Перелік посилань	52
	Додаток А	533
	Мікропроцесор	ADSP-2181
		534
	Додаток Б.....	633
	Опис стенда ADDS-2181 EZ-KITLite.....	633

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ З ЛАБОРАТОРНОЇ РОБОТИ

Звіт повинен містити:

- титульний лист із назвою теми, номером завдання та інших звичайних атрибутів;
- мету лабораторної роботи;
- завдання, видане на лабораторну роботу;
- матеріали, необхідні для представлення результатів роботи (текст програми, таблиці, ілюстрації);
- зміст контрольних запитань;
- матеріали, необхідні для представлення відповіді на контрольне запитання;
- висновки.

1 ЛАБОРАТОРНА РОБОТА №1 ВИЗНАЧЕННЯ ПРОДУКТИВНОСТІ ПРОЦЕСОРА INTEL 8031

Мета роботи: навчитися складати тестові послідовності. Навчитися оцінювати ефективність мікропроцесора.

1.1 Порядок виконання лабораторної роботи

1.1.1 Скласти тестову послідовність команд.

1.1.2 Вимірити час виконання тестової послідовності за допомогою секундоміра. Для цього необхідно вставити тестову послідовність у цикл як затримку між миготінням індикатора і вимірити період миготіння. Потім обчислити продуктивність. Виміри зробити 5-6 разів, обчислити середнє і похибку.

Вимірити час виконання тестової послідовності за допомогою вбудованого таймера – скласти відповідну програму і вивести на індикатор час (у машинних циклах) чи продуктивність у MIPS.

1.2 Короткі теоретичні відомості

1.2.1 Ручний метод вимірювання продуктивності процесора

Одиницею виміру продуктивності комп'ютера є час: комп'ютер, що виконує той же обсяг роботи за менший час є більш швидким. Час

виконання будь-якої програми виміряється в секундах. Часто продуктивність виміряється як швидкість появи деякого числа подій у секунду.

Для виміру часу роботи процесора використовується спеціальний параметр - час ЦП (CPU time), що не включає час чекання вводу/ виводу чи час виконання іншої програми.

Таким чином, щоб можна було визначити продуктивність процесора необхідно якимось чином вимірити час, витрачений на виконання тестової послідовності команд. Звичайно, можна було б засікти час для системи команд, що досить довго виконується, користаючись секундоміром, але це не зовсім коректний спосіб, бо залежить від часу реакції експериментатора. Логічніше доручити це комп'ютеру.

1.2.2 Вимірювання продуктивності за допомогою апаратного таймера

У 8051 є апаратний таймер. Якщо використовувати його, то можна запропонувати наступний алгоритм:

- ініціалізація таймера;
- запуск таймера;
- виконання тестової послідовності програм;
- зупинка таймера;
- вивід умісту таймера на дисплей.

Таким чином, ми одержуємо час виконання в тактах таймера (1/12 частоти осцилятора). Тепер знаючи частоту осцилятора можна розрахувати продуктивність у секундах чи у MIPS.

Цікаво порівняти результати цього і ручного виміру продуктивності.

1.2.3 Таймер/лічильник Intel 8051

8051 має два 16-бітних регістри таймера/лічильника: Timer0 і Timer1. Усі вони можуть бути сконфігуровані для роботи або в режимі таймера, або в режимі лічильника.

У режимі таймера, значення регістру збільшується на 1 кожен машинний цикл. Тому його можна вважати лічильником машинних циклів. Оскільки машинний цикл складається з 12 періодів осцилятора, швидкість рахунку складає 1/12 частоти осцилятора.

У режимі лічильника, значення регістра збільшується на 1 у відповідь на перехід 1-в-0 на відповідному вхідному контакті T0. Коли

високий рівень на ньому в одному циклі змінюється низьким рівнем у наступному циклі, значення лічильника збільшується. Розпізнавання переходу 1-в-0 займає два машинних цикли (24 періоди осцилятора), максимальна частота лічильника визначається як $1/24$ частоти осцилятора. Це не накладає обмеження на час перебування зовнішнього сигналу на вході, але щоб переконатися, що даний рівень визначений хоча б раз до його зміни, він повинний утримуватися як мінімум один повний машинний цикл.

Доповнення: таймер 0 і таймер 1 мають чотири режими роботи.

1.2.4 Таймер 0 і Таймер 1

Ці таймери/лічильники є й у 8051. Функція “Timer” чи “Counter” визначається контрольним бітом C/T# у СФР TMOD. Ці два таймери/лічильники мають чотири режими роботи, що визначаються парами бітів (M1, M2) у TMOD. Режими 0, 1 і 2 однакові для обох таймерів/лічильників. Режим 3 розрізняється.

1.2.5 Режим 0 (MODE 0)

Кожен таймер у режимі 0 – це 8-бітний лічильник з 32-х кратним попереднім дільником частоти.

У цьому режимі, регістр таймера конфігурується як 13-бітний регістр. Коли лічильник переповняється, він виставляє прапор переривання таймера TF1. Рахунковий вхід з'єднаний з таймером, коли TR1=1 і або GATE=1, або INT1# = 1. (Установка GATE у 1 дозволяє контролювати таймер зовнішнім сигналом INT1#, що допомагає проводити пульсові виміри). TR1 – це контрольний біт у регістрі TCON, GATE - у TMOD.

13-бітний регістр складається з усіх 8-ми бітів TH1 і молодших 5-ти бітів TL1. Верхні три біти TL1 не визначені і повинні ігноруватися. Установка прапора виконання (TR1) не дозволяє очищати регістри.

Режим 0 однаковий для таймера 0 і таймера 1. Треба замінити TR0, TF0 і INT0# відповідними сигналами таймера 1. Два біти GATE різні: один для таймера 1 (TMOD.7) і один для таймера 0 (TMOD.3)

GATE	C/T#	M1	M0	GATE	C/T#	M1	M0
-------------	-------------	-----------	-----------	-------------	-------------	-----------	-----------

Таймер 1

Таймер 0

Рисунок 1.1- Формат слова стану

Якщо встановлений GATE, таймер/лічильник “х” дозволений, тільки поки на виводі “INTх#” високий рівень і встановлений контрольний вивід “TRх”. Якщо GATE не встановлений, таймер “х” дозволений увесь час, поки встановлений контрольний біт “TRх”.

Режим 1: THx TLx - 16-розрядний лічильник.

Режим 2: TLx – 8-бітний лічильник. При переповненні завантажуються значенням із THx (THx = const).

1.3 Схеми програми, що використовує таймер для виміру проміжку часу

```

                ORG 00H
                JMP Start
                ORG 33H
Start:          mov TMOD,#01H      ;ініціалізація т/л 0
                setb TR0           ;ввімкнути таймер 0
                                   ;послідовність команд

                clr TR0            ;зупинка таймера 0

                                   ;вивід умісту регістрів таймера
                                   ;(TH0,TL0) на індикатори

                jmp $
                END

```

1.4 Контрольні питання

- 1.4.1 Визначення MIPS.
- 1.4.2 Визначення MFLOPS.
- 1.4.3 Критерії використання MIPS та MFLOPS.
- 1.4.4 Нормалізація MFLOPS.
- 1.4.5 Тест SPEC.
- 1.4.6 Поняття продуктивності комп'ютера.

2 ЛАБОРАТОРНА РОБОТА №2

ПРОГРАМУВАННЯ ПАРАЛЕЛЬНОГО ІНТЕРФЕЙСУ K580BB55. СХЕМИ ВІДОБРАЖЕННЯ ІНФОРМАЦІЇ

Мета роботи: вивчення схем динамічної і статичної індикації. Розробка програм для ОЕОМ КР1816ВЕ31 для відображення цифрової інформації на пристроях динамічного і статичного типу, а також на одиничних індикаторах.

2.1 Порядок виконання лабораторної роботи

2.1.1 Вивчити принцип роботи різних методів відображення.

2.1.2 Розробити алгоритм для виконання індивідуального завдання до початку лабораторного заняття.

2.1.3 Розробити програму для виконання індивідуального завдання.

2.1.4 Увести програму індивідуального завдання на персональному комп'ютері.

2.1.5 Завантажити програму в стенд ОЕОМ. Переконавшись в правильному виконанні індивідуального завдання, при негативному результаті здійснити зміну алгоритму або програми. Повторити завантаження програми в стенд ОЕОМ

2.1.6 Роздрукувати листинг правильно працюючої програми.

2.1.7 Відповісти на контрольні питання.

2.2 Короткі теоретичні відомості

2.2.1 Склад мікросхеми K580BB55

До складу інтегральної мікросхеми K580BB55 програмувального паралельного інтерфейсу входить:

- пристрій керування вводом-виводом;
- двох-направлений буфер даних;
- порти А і В, 8-ми розрядні регістри із шинними формувальниками;
- порт С, два 4-х розрядних регістри із шинними формувальниками;
- регістр керуючого слова.

Порт А і старша половина порту С може бути об'єднані в групу А. Порт В і молодша половина порту С може об'єднуватися в групу В.

Для забезпечення працездатності інтегральної мікросхеми необхідно записати керуюче слово.

Керуюче слово ініціалізації визначає режим роботи та напрямок передачі даних.

Таблиця 2.1-Формат керуючого слова ініціалізації

Біт							
D7	Визначає установку режимів роботи каналів D7-1, або роботу ППІ в режимі скидання/установки окремих розрядів						
D6	0	Режим 0	0	Режим 1	1	Режим 2	Група А
D5	0		1		0		
D4	0	Вивід РА(7-0)		1	Ввід І РА(7-0)		
D3	0	Вивід РС(7-4)		1	Ввід РС(7-4)		
D2	0	Режим 0		1	Режим 1		
D1	0	Вивід РВ(7-0)		1	Уведення1РВ(7-0)		
D0	0	Вивід РС(3-0)		1	Уведення1 РС(3-0)		

2.2.2 Системи відображення інформації

Найпростішими приладами відображення інформації в цифрових пристроях є світлодіоди і цифрові індикатори.

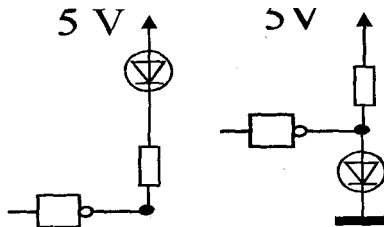


Рисунок 2.1 - Включення одиничних індикаторів

У напівпровідникових світлодіодах використовується властивість р-п переходу випромінювати світло у видимій частині спектра при протіканні через нього прямого струму ($I_p=5-20\text{ма}$, $U_p=2-3\text{В}$). Варіанти включення індикаторів на рис.2.1.

Для відображення цифрової інформації найбільше поширення одержали семи-сегментні індикатори, у яких зображення цифри складають із семи лінійних світлодіодних сегментів розташованих у виді цифри 8. На основі світлодіодів і семи сегментних індикаторів будуються підсистеми відображення інформації. При побудові підсистем відображення інформації розрізняють два підходи - динамічна і статична індикація.

Статична індикація полягає в постійному підсвічуванні індикаторів HL1-п від одного джерела інформації рис.2.2.

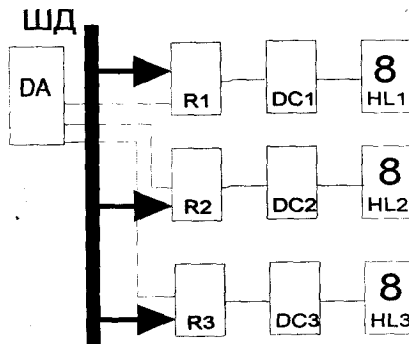


Рисунок 2.2 - Структурна схема статичної індикації

DA - дешифратор адреси, необхідний для вибірки відповідного регістра.

R1-R3- регістри в яких тимчасово зберігається значення коду числа для відображення (відповідний регістр вибирається DA).

DC1-DC3 – семи-сегментні дешифратори, що перетворюють подвійний код у семи-сегментний код.

HL1-HL3 – семи-сегментні індикатори.

ШД - шина даних, по ній здійснюється передача даних на індикацію.

У такій системі кожен індикатор HL1-п підключається через власний дешифратор DC1-п і регістр-клямку RG1-п до шини даних, вибірка регістрів RG1-п проводиться за допомогою селектора адреси СА. Апаратні витрати при такій організації складають n пар (регістр + дешифратор) при n десяткових розрядах індикатора.

У стенді адресація реалізована за допомогою дешифратора – мікросхеми 153ЗИД7, через вихідні сигнали CS2, CS3. Так як адреси індикаторів знаходяться в полі адресації зовнішньої пам'яті даних, то для вибірки використовується сигнал WR із процесора. Мікросхеми

індикації КР490ИП2 містять у собі регістр і семи-сегментний дешифратор.

Сутність динамічної індикації полягає в почерговому циклічному підключенні кожного індикатора НГ1-п до джерела інформації через загальну шину даних, рис 2.3.

Вибірка індикатора здійснюється дешифратором DA. У регістрі RD зберігається цифровий код, призначений для відображення. У регістрі RA зберігається адреса індикатора.

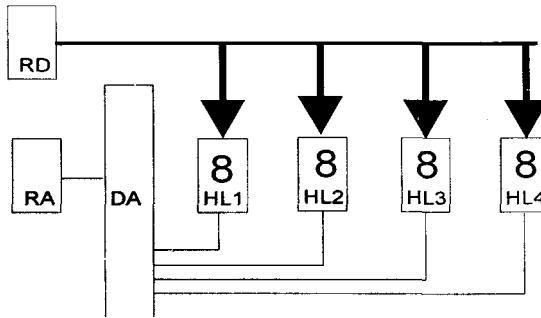


Рисунок 2.3 - Структурна схема динамічної індикації

RD - регістр даних для тимчасового збереження відображуваного числа або символу.

RA - регістр адреси для тимчасового збереження двійкового коду адреси обраного індикатора.

DA - для перетворення адреси на позиційний код.

HL1 – HL4- семи-сегментні індикатори.

При такому включенні значно зменшуються апаратні витрати, але необхідно забезпечити достатній час світіння одного індикатора, для того щоб не зменшувалася яскравість. Також необхідно забезпечити таку частоту перебору індикаторів, щоб не було помітне мерехтіння. Переваги такого способу помітні при кількості десяткових розрядів індикації більше 5.

На стенді статична індикація реалізована на чотирьох статичних семи-сегментних двійково-десяткових індикаторах (НГ0 – НГ3). Звертання до них виконується, як до комірок пам'яті з адресами A000h, B000h. Передача даних на індикатори здійснюється з OEOM по шині даних AD(0-7). Сигнали вибірки мікросхем індикації - CS3, CS2.

Динамічна індикація реалізована за допомогою двох семи-сегментних індикаторів HG5, HG6. Керування динамічною індикацією здійснюється за допомогою елементів DD4, DD3.4, DD3.5 (лінія даних A,B, C, D, E, F, G, H, - PBO, PB1, PB2, PB3, PB4, PB5, PB6, PB7), сигнали надходять з порту PB мікросхеми паралельного приймача-передавача DD17 (див. схему електричну стенда), сигнали вибірки відповідного індикатора надходять з виходу PC7 порту PC мікросхеми DD17 до транзисторів VT2 і VT3.

Також на платі розширення встановлена матриця 5x7 світлодіодов HG7, і лінійка світлодіодов HL1-HL8. Світлодіоди запалюються записом логічних одиниць у відповідні розряди порту PA мікросхеми паралельного приймача-передавача DD17.

Керування світлодіодною матрицею здійснюється по лініях PA0-PA4 паралельного приймача-передавача DD17 і P1.0-P1.6 DD2. Наприклад, для того щоб засвітити крапку з координатами [1;1] необхідно встановити рівень логічної одиниці на лінії PA.0, і рівень логічного нуля на лінії P1.0.

2.3 Приклад програми для статичної індикації

З частотою 1 Гц відобразити на статичному індикаторі число 04:

	ORG 0	
	Jmp Continue	
	ORG 33H	
Continue:	mov A,#04h	;записати в A число 04
	mov DPTR,#0A000h	;установити у DPTR
		;адресу інд. HG0, HG1
	movx @DPTR,A	;засвітити на інд.
		;HG0, HG1 число 04
	mov DPTR,#0B000h	;установити в DPTR адресу
		;інд. HG2, HG3
	movx @DPTR,A	;засвітити на інд. HG2,HG3
		;число 04
	mov R1,#0FFh	;часова затримка
		;на 2 регістрах з декрементом
Cl:	mov R2,#0FFh	;у вкладеному циклі,
C3:	djnz R2, C3	;визначає час світіння
	djnz R1, Cl	;індикаторів
	mov A,#0FFh	;записати в A число FFh
	mov DPTR,#0A000h	;установити в DPTR адресу
		;інд.HG0, HG1
	movx @DPTR,A	;згасити індикатори HG0, HG1

	mov DPTR,#0B000h	;установити в DPTR адресу ;інд. HG2, HG3
	movx @DPTR,A	;згасити індикатори HG2, HG3
	mov Rl,#OFFh	;часова затримка на 2 регістрах
C2:	mov R2,#OFFh	;із декрементом у вкладеному ;циклі
C4:	djnz R2, C4	;визначає час гасіння
	djnz Rl, C2	;індикаторів
	jmp Continue	;перехід на початок програми
	END	

2.4 Варіанти індивідуальних завдань

Таблиця 2.1 - Таблиця завдань до лабораторної роботи №2

№	Текст індивідуального завдання
1	Занести в регістр R1 число XX, віднімаючи від числа одиницю відображати на Д інд. HG5, HG6 отримане значення до нуля з частотою 1 Гц. З частотою 0,5 Гц засвічувати HL1, HL2.
2	Занести в В ДД число X0, у регістр Rl - XX, число з В відображати на С інд. HG1 з частотою 1 Гц, число з Rl відображати на Д інд. HG5, HG6 з частотою 0,25 Гц.
3	Засвітити HL1-HL4. Занести в регістр В ДД число 0X, у регістр R5 – X0, два розряди суми (десятки й одиниці) по черзі відображати на С інд. HG2, HG1 і на Д інд. HG5, HG6 з частотою 1 Гц.
4	Занести в регістр R6 ДД число XX, у регістр R5 - XX, у регістр R0 -XX, послідовно відображати ці числа по одному розряду на Д інд. HG5, HG6, С інд. HG4, HG3, HG2, HG1.
5	Поперемінно засвічувати HL5 - HL8. Занести в регістр R2 ДД число 0X, у регістр Rl – X0, суму чисел відобразити на Д інд. HG5, HG6.
6	Занести в регістр А ДД число 0X, у регістр R2 – X0, число з А відобразити на С інд. HG3, число з регістру R2 відображати на Д інд. HG6 з частотою 0.25 Гц.
7	Занести в А ДД число XX, у регістр Rl - XX, молодші два розряди суми числа відобразити на Д інд. HG5, HG6, старшу тетраду – на С інд. HG3, плавно засвітити HL1-HL8.
8	Занести в регістр R6 ДД число XX, відобразити його на С інд. HG3, HG1.
9	Занести в регістр В ДД число XX, у регістр R3 - XX, різницю чисел відобразити на Д інд. HG6, HG5.

10	Засвітити HL3. Занести в А ДД число XX, у регістр R5 – X0, число з А відобразити на С інд. HG2, HG1, число з R5 відобразити на Д інд. HC5.
11	Занести в регістр R0 ДД число XX, поперемінно відображати молодшу і старшу тетради на Д інд. HG5, HG6 з частотою 0,25 Гц.
12	Занести в регістр R2 ДД число 0X, у регістр R5 - X0, суму чисел відобразити на Д інд. HG5, HG6.
13	Занести в регістр В ДД число, з частотою 2 Гц виводити це число на С інд. HG2, HG3 і одночасно на Д інд. HG5, HG6.
14	Поперемінно засвічувати HL1-HL4, HL5-HL8. Занести у комірку з адресою 0010h зовнішньої пам'яті OEOM ДД число 0X, у регістр R3 - XX, суму чисел відобразити на Д інд. HG5, HG6.
15	Занести в регістр R1 ДД число 0X, у регістр R3 - XX, молодші два розряди суми відобразити на Д інд. HG5, HG6 з повільним (протягом 5 сек.) загасанням цього числа, старшу - на С інд. HG2.

Перелік використаних скорочень Д інд. – динамічний індикатор; С інд. – статичний індикатор.

2.5 Контрольні питання

2.5.1 Призначення та програмування паралельного інтерфейсу K580BV55. Призначення і сфера застосування пристроїв зі статичним і динамічним методом відображення .

2.5.2 Розрахунок часу регенерації для динамічного методу відображення .

2.5.3 Обґрунтування необхідності застосування різних методів відображення даних.

2.5.4 Схемо-технічні рішення для побудови схем відображення інформації.

2.5.5 Схеми включення одиничних індикаторів.

2.5.6 Включення рідинно-кристаливих індикаторів.

2.5.7 Включення газорозрядних індикаторів.

3 ЛАБОРАТОРНА РОБОТА № 3 ОБРОБКА ЧАСТОТНИХ І ЧАСОВИХ СИГНАЛІВ

Мета роботи: вивчення методів частотного перетворення. Здобуття навичок виміру частоти, періоду, тривалості дискретних сигналів за допомогою програмувальних лічильників/ таймерів ОЕОМ 1816ВЕ31, а також з використанням зовнішнього еталонного генератора.

3.1 Порядок виконання лабораторної роботи

3.1.1 Вивчити методи частотного перетворення (апаратні та за допомогою програмувальних лічильників/таймерів).

3.1.2 Вивчити методи виміру часових інтервалів.

3.1.3 Розробити алгоритм для виконання індивідуального завдання.

3.1.4 Розробити програму для виконання індивідуального завдання до початку лабораторного заняття.

3.1.5 Увести програму індивідуального завдання на персональному комп'ютері.

3.1.6 Завантажити програму в стенд ОЕОМ . Переконатися в правильному виконанні індивідуального завдання , змінити значення вимірюваної частоти, повторити перетворення, при негативному результаті здійснити зміну алгоритму або програми.

3.1.7 Роздрукувати листінг правильно працюючої програми.

3.1.8 Відповісти на контрольні питання викладача.

3.2 Короткі теоретичні відомості

У системах автоматичного керування часто приходится вимірювати такі величини як: частота f , період T , тривалість сигналу t , зсув фаз ϕ . Для цього застосовується перетворення частота-код. У залежності від того який саме параметр потрібно вимірити застосовують різний підхід.

При вимірі частоти f_x проводиться підрахунок імпульсів вхідного сигналу, на протязі фіксованого відрізка часу рис.3.1.

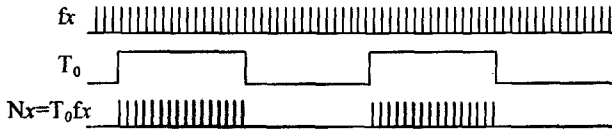


Рисунок 3.1-Часові діаграми

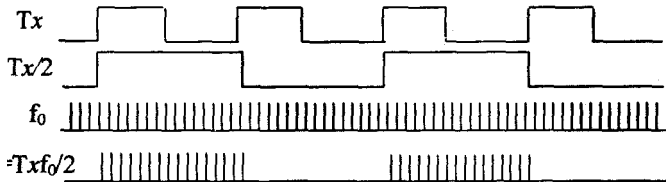


Рисунок 3.2 - Вимір періоду проходження методом підрахунку кількості імпульсів відомої частоти

Такий метод прийнятний для виміру частот більше 100 Гц. Верхня межа обмежується швидкодією елементів схеми і розрядністю лічильників.



Рисунок 3.3 - Вимір зсуву фаз

Вимір частот нижче 100 Гц замінюється виміром періоду T_x . При цьому проводиться підрахунок імпульсів фіксованої частоти f_0 за інтервал рівний або кратний вимірюваному періоду рис.3.2.

У такий же спосіб вимірюється тривалість імпульсу T_d відмінність складається лише в тім, що при вимірі тривалості імпульсу не треба поділяти частоту на два.

При вимірі зсуву фаз часовий інтервал t , формують шляхом кон'юнкції сигналів f_{1x} і f_{2x} рис.3.3, отриманий інтервал часу вимірюють методом описаним вище.

Схеми, що реалізують такий алгоритм приведені на рис. 3.4.

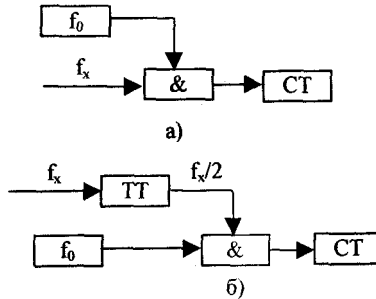


Рисунок 3.4 - Схеми для виміру частоти: а) низької, б) високої

У лабораторному стенді при наявності плати розширення до входу ОЕОМ Т0 підключений постійний генератор частоти, а до входу Т1 підключений генератор зі змінною частотою. Сигнали цих генераторів можна спостерігати на осцилографі, підключеному до ВНС рознімання. Для спостереження сигналу Т0 необхідно замкнути перемичку J1, а для Т1 перемичку J3.

Таймери / лічильники (т/л) ОЕОМ призначені для підрахунку зовнішніх подій, для одержання програмно керованих часових затримок, виконання функцій ОЕОМ, що задають час. До складу блоку т/л входять:

- два 16-ти розрядних регістри т/л 0, т/л 1;
- 8-ми розрядний регістр режимів т/л (TMOD);
- 8-ми розрядний регістр керування (TCON);
- схема інкременту;
- схема фіксації INT0, INT1, T0, T1;
- схема керування прапорами;
- логіка керування т/л.

Два 16-розрядних регістри т/л 0 і т/л 1 виконують функцію збереження вмісту рахунка. Кожний з них складається з пари 8-ми

розрядних регістрів, відповідно TH0, TL0 і TH1, TL1. Причому регістри TH0, TH1 - старші, регістри TL0, TL1 - молодші 8 розрядів. Кожний з 8-ми розрядних регістрів може бути використаний, як регістр загального призначення, якщо т/л не використовуються (біт TR0 для т/л 0 і біт TR1 для т/л 1 у регістрі керування TCON дорівнює "0").

Регістр режимів т/л (TMOD) призначений для прийому і збереження коду, що визначає:

- один з 4-х можливих режимів роботи кожного т/л;
- роботу у виді таймера чи лічильника;
- керування т/л від зовнішнього виводу.

3.3 Приклад програми виміру частоти

```
$nolist
$include(c:\asm51\compile\mod51)
$list                                ;Вимір частоти в Гц. Розрахована на зовнішню
                                    ;частоту T0 6817 Гц, підрахунок ведеться за 1 с,
                                    ;результат у виді XXXX Гц
mode equ 55h                        ;Настроює T1 counter, T0 counter
str  equ 50h                         ;Запис у TCON дозволяє рахунок
ORG 0000h
BEG: MOV T1L,#00h                    ;Обнуління
     MOV TH1,#00h                    ;т/л 1
     MOV TL0,#0f0h                    ;Попереднє встановлення т/л0
     MOV TH0,#0E4h                    ;Розрахована на 1с
     MOV TMOD,#MODE                   ;ініціалізація режимів таймерів
     MOV TCON,#STR                     ;Запуск рахунка
     JNB TF0,$                         ;Перевірка прапора переповнення т/л0
     CLR TR1                           ;Якщо відбулося переповнення, то
     CLR TR0                           ;зупинити рахунок т/л 1 і т/л 0
     MOV RO,TL1                        ;Зчитування результату виміру
     MOV R1,TH1                        ;RO містить молодшу частину, а R1 старшу
     CALL IND                           ;Виклик підпрограми індикації
     MOV R3,#01h                       ;Часова затримка
M     MOV R4,#01h                       ;на двох регістрах
     DJNZ R4,$                          ;з декрементом
     DJNZ R3,M                          ;у вкладеному циклі
     JMP BEG                           ;Перехід на наступний цикл виміру
IND:                                  ;Підпрограма індикації, робить переклад
                                    ;результату із двійкової системи у двійково -
```

;десяткову та його відображення

```

        JMP Ml
L:      DB 01h,02h,04h,08h,16h,32h,64h,28h,56h,12h,24h,48h, 96h
HI:     DB 00h,00h,00h,00h,00h,00h,00h,01h,02h,05h,10h,20h,40h
Ml:     MOV R2,#00h

        MOV R6,#00h
        MOV R7,#00h
NACHALO_L:
        MOV A,RO
        JNB ACC.0,AGAIN_L
        MOV A,R2
        MOV DPTR,#L
        MOVC A,@A+DPTR
        ADD A,R6
        DA A
        MOV R6,&
        JNB PSW. 7,PLOS_NEXT_L
        MOV A,R7
        ADD A,#01H
        DA A
PLUS_NEXT_L:
        MOV R7,A
        MOV A,R2
        MOV DPTR,#HI
        MOVC A,@A+DPTR
        ADD A,R7
        DA A
        MOV R7,A
AGAIN_L:
        MOV A,RO
        RRC A
        MOV RO,A
        INC R2
        CJNE R2,#08h, NACHALO_L
NACHALO_HI:
        MOV A,R1
        JNB ACC.0,AGAIN_HI
        MOV A,R2
        MOV DPTR,#L
        MOVC A,@A+DPTR
        ADD A,R6
        DA A
        MOV R6,A
        JNB PSW. 7,PLUS_NEXT_HI

```

```

MOV A,R7
ADD A,#01H
DA A
MOV R7,A
PLUS_NEXT_HI: MOV A,R2
MOV DPTR,#HI
MOVC A,@A+DPTR
ADD A,R7
DA A
MOV R7,A
AGAIN_HI: MOV A,R1
RRC A
MOV R1,A
INC R2
CJNE R2,#0Dh,NACHALO_HI
MOV A,R6
MOV DPTR,#0B000h ;Виведення на індикацію
MOVX @DPTR,A ;молодшої частини коду
MOV A,R7
MOV DPTR,#0A000h ;Виведення на індикацію
MOVX @DPTR,A ;старшої частини коду
RET
END

```

3.4 Варіанти індивідуальних завданні

Таблиця 3.1 - Таблиця завдань до лабораторної роботи №3

№	Текст індивідуального завдання
1	Вимірити частоту ГФЧ (T0) відобразити на С Інд.
2	Вимірити частоту ГПЧ (T1) відносно ГФЧ відобразити на С Інд.
3	Вимірити частоту ГПЧ (T1) щодо частоти роботи процесора, відобразити на С Інд..
4	Вимірити різницю частот ГФЧ і ГПЧ відобразити на Д інд.
5	З використання внутрішніх т/л забезпечити плавне загоряння числа 55 на С інд.
6	По черзі відображати 48 на індикаторах HG1,2 і HG3,4 інтервалом 1с. Часові інтервали формувати ГФЧ.
7	Підрахувати і відобразити на Д інд. кількість натискань S11 за 10с. Часові інтервали формувати т/л ОЕОМ.

8	Відображати на індикаторах HG2 числа від 1 до 9 у плинні 1с, і паузою між відображеннями 1с. Тимчасові інтервали формувати т/л ОЕОМ.
9	При натисканні кнопки S10 відобразити числа 1,9,9,9 на індикаторах HG1,2,3,4 на 1с.
10	При натисканні кнопки S3 відобразити числа 1,2,3,4 на індикаторах HG1,2,3,4 на 2с.
11	Підрахувати кількість натискань кнопки S3 за 5с. Часові інтервали формувати з допомогою ГФЧ.
12	Вимірити час між натисканнями кнопок S10 і S4. Час в сек. відобразити на С інд.
13	При натисканні кнопки S4 відобразити в сек. час її натискання на індикаторах HG1,2.
14	Вимірити період ГФЧ (T0) відобразити на С інд.
15	Вимірити період ГПЧ (T1) відобразити на С інд.

3.5 Контрольні питання

3.5.1 Методи і типи частотного і часового перетворення.

3.5.2 Параметри частотного перетворення.

3.5.3 Фактори, що впливають на похибку частотного перетворення.

3.5.4 Поняття дозвільної здатності частотного перетворення.

3.5.5 Вимірювання періоду.

3.5.6 Характеристика перетворення, нелінійність частотного перетворення.

3.5.7 Апаратні реалізації частотного і часового перетворення.

3.5.8 Приклади практичного застосування частотного і часового перетворення.

4 ЛАБОРАТОРНА РОБОТА №4 ОБМІН ІНФОРМАЦІЄЮ ЧЕРЕЗ ПОСЛІДОВНИЙ ПОРТ

Мета роботи: навчитися робити елементарні операції передачі інформації через послідовний порт, складати і налагоджувати прості багатопроцесорні системи типу «хост-контролер».

4.1 Порядок виконання роботи

4.1.1 Скласти програму для 8031 для прийому/передачі інформації через послідовний порт відповідно до варіанта.

4.1.2 Скласти програму серверної частини для передачі по COM порту для ПК

4.1.3 Запустити програму на стенді і на ПК, домогтися коректної взаємної роботи.

4.2 Короткі теоретичні відомості

Область даних BIOS містить список до чотирьох адрес COM портів.

- COM1 адаптер декодує порти від 3f8H до 3ff.

- COM2 адаптер декодує порти від 2f8H до 2ff.

COM1 викликає переривання 4 (IRQ 4, оброблений INT 0с вектор)

COM2 викликає переривання 3 (IRQ 3, оброблений INT 0b вектор).

Опис портів:

2f8H

Запис — буфер порту передавача. 8 бітів символу, що посилається.

Читання — регістр буфера приймача. 8 бітів отриманого символу.

Запис: (коли DLAB = 1) молодший байт дільника.

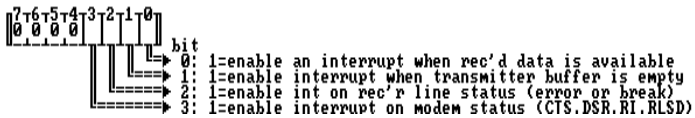
Після OUT 3fb, 80H зберігає молодший байт дільника, що, разом зі старшим байтом (порт 3f9h) складає 16-розрядне значення.

BAUD	DIVISOR	BAUD	DIVISOR
110	1040	1200	96
150	768	2400	48
300	384	4800	24
600	192	9600	12

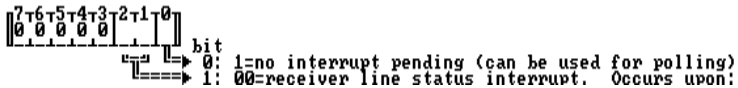
Це встановлює швидкість у бодах, як показано вище.

3f9H

Запис — старший байт дільника (коли DLAB = 1; тобто, після OUT 3fb, 80H)

**3fa**

Запис — регістр дозволу переривань.



Читання — розпізнавальний регістр переривань. Коли переривання відбувається, потрібно зчитати значення регістру, щоб знайти, що викликало це переривання.

Контроль переповнення по парності, чи помилка кадрової синхронізації або переривання:

Скидається зчитуванням стану рядка (порт 3fd)

01 = отримані дані доступні

Скидається, зчитуванням буфера приймачів (порт 3f8H)

10 = трансмітер буфер порожній

Скидається, записом буфера трансмітерів (порт 3f8H)

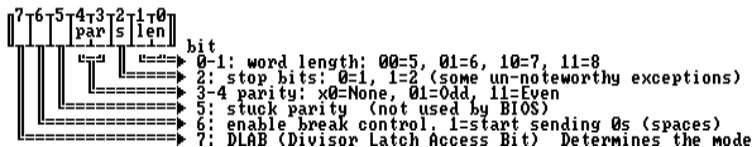
11 = стан модему. Відбувається на: Clear To Send Data, Ready,

RI, чи RLS.

Скидається зчитуванням стану модему (порт 3fe)

3fbH

Читання/запис — регістр керування лінією.

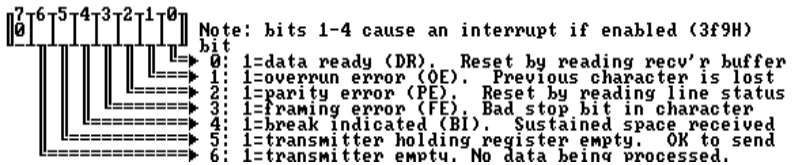
**3fc**

Запис - регістр керування модему.

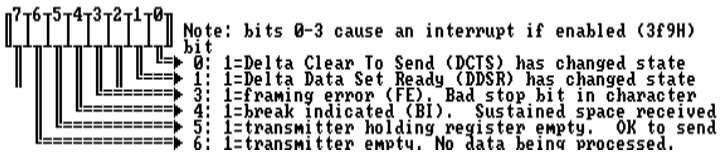


3fdH

Читання - регістр стану лінії.

**3feH**

Читання - регістр стану модему.

**4.3 Варіанти завдань**

Таблиця 4.1-Вихідні дані до роботи №4

№	Текст індивідуального завдання
1	Уведіть із клавіатури ПК і передайте з ПК 2 байти у двійково-десятьковому форматі через COM порт зі швидкістю 9600 і виведіть їх на статичні індикатори стенда.
2	Уведіть із клавіатури ПК і передайте з ПК 2 байти у двійково -десятьковому форматі через COM порт зі швидкістю 4800 і виведіть на статичні індикатори стенда їх суму.
3	Передайте через COM порт зі швидкістю 4800 номер натиснутої клавіші клавіатури стенда (від 1 до 3) і виведіть його на екран ПК.
4	Передайте через COM порт зі швидкістю 2400 байт і виведіть його на екран ПК.
5	Передайте через COM порт зі швидкістю 1200 10 байт і виведіть їх на екран ПК
6	Уведіть із клавіатури ПК і передайте з ПК 2 байти у двійково-десятьковому форматі через COM порт зі швидкістю 9600 і виведіть їх на статичні індикатори стенда.
7	Передайте з ПК серію з 10 байт $f(i)=\text{int}(48 \cdot \sin(\pi i/10)+48)$ у двійково-десятьковому форматі через COM порт зі швидкістю 2400 і виведіть їх по мірі прийому на статичні індикатори стенда. (Посилка значень через 0.5 сек).

8	Передайте через COM порт зі швидкістю 1200 номер натиснутої клавіші клавіатури стенда (від 1 до 3) і виведіть його на екран ПК.
9	Передайте через COM порт зі швидкістю 4800 байт і виведіть його на екран ПК.
10	Передайте через COM порт зі швидкістю 9600 5 байт – значенням 10-i-3 – і виведіть їх на екран ПК.

4.4 Контрольні питання

4.4.1 Як здійснюється програмування порту?

4.4.2 Як здійснюється передача даних?

4.4.3 Як задається швидкість передачі даних?

4.4.4 Як задається швидкість передачі даних з OEOM на ПК?

5 ЛАБОРАТОРНА РОБОТА № 5 АРХІТЕКТУРА ADSP-2181

Мета роботи: вивчити архітектуру і технічні характеристики ЦПОС ADSP-2181. Придбати практичні навички роботи з програмою ADSP-2181 Navigator. Ознайомитися з технічною документацією, що відноситься до сімейства ЦПОС ADSP-21xx.

5.1 Завдання

5.1.1 Вивчення пристрою генераторів адреси ЦПОС сімейства ADSP-21XX.

5.1.2 Вивчення принципу роботи блоків ALU і MAC сигнальних процесорів сімейства ADSP-21XX.

5.1.3 Вивчення принципу роботи блоку Shifter ЦПОС сімейства ADSP-21XX.

5.1.4 Вивчення пристрою контролера BDMA сигнальних процесорів сімейства ADSP-218x.

5.1.5 Вивчення принципу роботи порту IDMA ЦПОС сімейства ADSP-218x.

5.1.6 Вивчення пристрою програмного автомата ЦПОС сімейства ADSP-21XX.

5.2 Порядок виконання роботи

5.2.1 Вивчити пристрій генераторів адреси ЦПОС сімейства ADSP- 21XX.

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [003] ;
- у вікнах [004] , [005] , [006] , [007] містяться необхідні теоретичні відомості;
- запустити приклад 1. Вивчити порядок формування адреси в циклічному буфері DAG1;
- у вікні [009] описані розходження в пристрої DAG1 і DAG2;
- самостійно розібрати приклад роботи генератора адреси при інших початкових значеннях, що зберігаються в регістрах L, I та M (завдання видає викладач). Результат роботи оформити у вигляді таблиці (табл. 5.1).

Таблиця 5.1 – Формування адреси в циклічному буфері

№ циклу	Регістр L	Регістр I	Регістр M

5.2.2 Вивчення принципу роботи блоків ALU і MAC сигнальних процесорів сімейства ADSP-21XX:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe) ;
- відкрити вікно [017] ;
- у вікнах [018] і [019] містяться необхідні теоретичні відомості;
- запустити приклад 3. Вивчити порядок виконання операції додавання в ALU;
- відкрити вікно [021] ;
- у вікнах [022] і [023] містяться необхідні теоретичні відомості;
- запустити приклад 4. Вивчити порядок виконання операції множення-накопичення в MAC;
- запустити приклад 5. Вивчити порядок виконання операції множення-накопичення в MAC для випадку насичення;

– самостійно розібрати приклад роботи блоку MAC при виконанні частини програми, заданої викладачем. Результат роботи оформити у вигляді таблиці (табл. 5.2);

Таблиця 5.2 – Виконання операції множення-накопичення в блоці MAC

№ циклу	MX	MY	MR2	MR1	MR0	MF

5.2.3 Вивчення принципу роботи блоку Shifter ЦПОС сімейства ADSP-21XX.

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [045] ;
- у вікнах [046] , [047] і [050] містяться необхідні теоретичні відомості;
- запустити приклад 10. Вивчити порядок виконання операції зрушення в блоці Shifter;
- відкрити вікно [049], що містить відомості про операцію визначення максимальної експоненти блоку даних;
- запустити приклади 11, 12;
- відкрити вікно [052], що містить відомості про операцію нормалізації даних;
- самостійно розібрати приклад роботи блоку Shifter при виконанні ділянки програми, заданої викладачем. Результат роботи оформити у вигляді таблиці (табл. 5.3).

Таблиця 5.3 – Виконання операцій у блоці Shifter

№ циклу	SI	SE	SB	SR1	SR0

5.2.4 вивчення пристрою контролера BDMA сигнальних процесорів сімейства ADSP-218x:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [060];

- у вікнах [060] , [064] , [065] , [066] і [067] містяться необхідні теоретичні відомості;
- запустити приклад 13. Вивчити порядок читання 8-розрядних даних, вирівняних методом LSB;
- запустити приклад 14. Вивчити порядок читання 8-розрядних даних, вирівняних методом MSB;
- запустити приклад 15. Вивчити порядок читання 16-розрядних даних;
- запустити приклад 16. Вивчити порядок читання 24-розрядних даних;
- порівняти чотири методи читання даних з використанням контролера BDMA. Результати порівняння звести в (табл. 5.4) ;

Таблиця 5.4 – Виконання операцій читання контролером BDMA

Метод читання даних	Особливості
8-Bit LSB Align Read	
8-Bit MSB Align Read	
16-Bit Read	
24-Bit Read	

- запустити приклад 17. Вивчити порядок запису 8-розрядних даних, вирівняних методом LSB;
- запустити приклад 14. Вивчити порядок запису 8-розрядних даних, вирівняних методом MSB;.
- запустити приклад 15. Вивчити порядок запису 16-розрядних даних;
- запустити приклад 16. Вивчити порядок запису 24-розрядних даних;
- порівняти чотири методи запису даних з використанням контролера BDMA. Результати порівняння занести в (табл. 5.5).

Таблиця 5.5 – Виконання операцій запису контролером BDMA

Метод запису даних	Особливості
8-Bit LSB Align Write	

8-Bit MSB Align Write	
16-Bit Write	
24-Bit Write	

5.2.5 Вивчення принципу роботи порту IDMA ЦПОС сімейства ADSP-218x:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [118];
- у вікнах [119] , [120] і [121] містяться необхідні теоретичні відомості;
- запустити приклад 21. Вивчити порядок роботи ЦПОС разом з хост-контролером;
- скласти схему протоколу обміну даними через порт IDMA.

5.2.6 Вивчення пристрою програмного автомата ЦПОС сімейства ADSP-21XX:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [027];
- у вікнах [028] , [029] , [030] , [031] , [032] , [033] та ін. містяться необхідні теоретичні відомості;
- запустити приклад 7. Вивчити принцип роботи програмного автомата;
- запустити приклад 6. Вивчити метод апаратної організації циклу в ЦПОС;
- скласти блок-схему програмного автомата ЦПОС і описати призначення його основних частин.

5.3 Контрольні питання

5.3.1 Області використання сигнальних процесорів.

5.3.2 Особливості архітектури сучасних ЦПОС.

5.3.3 Основні принципи роботи контролерів переривань і ПДП.

5.3.4 Технічні характеристики ЦПОС сімейства ADSP-218x.

5.3.5 Порівняльна характеристика різних класів мікропроцесорів.

5.3.6 Класифікація архітектури обчислювальних пристроїв.

5.3.7 Приклад використання ЦПОС у складі аналогового модему.

5.3.8 Приклад використання ЦПОС у складі мобільного телефону.

5.3.9 Приклад використання ЦПОС у складі звукової карти ПК;

5.3.10 Приклад використання ЦПОС у складі системи ЦОС;

8.3.11 Переваги і недоліки використання програмного автомата в складі сигнального процесора.

5.3.12 Архітектура сигнальних процесорів з фіксованою крапкою фірми Analog Devices.

6 ЛАБОРАТОРНА РОБОТА № 6 АРИФМЕТИКА ЦПОС З ФІКСОВАНОЮ КРАПКОЮ

Мета роботи: вивчити формати представлення чисел у ЦПОС ADSP-2181 і правила їх використання у різних додатках ЦОС. Придбати практичні навички, необхідні для проектування систем цифрової обробки сигналів. Ознайомитися з технічною документацією, що відноситься до наявних інструментальних програм.

6.1 Завдання

6.1.1 Програми додавання, віднімання і множення дробових чисел (подвійна точність).

6.1.2 Програми додавання, віднімання і множення дробових чисел (потрійна точність).

6.1.3 Програми множення цілих чисел (подвійна і потрійна точність).

6.1.4 Програма розподілу чисел одинарної точності без знаку.

6.1.5 Програма розподілу чисел одинарної точності зі знаком.

6.1.6 Програма мовою Сі для перетворення форматів представлення даних.

6.2 Порядок виконання роботи

6.2.1 Програми додавання, віднімання і множення дробових чисел (подвійна точність):

- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab2). Скопіювати в нього файли asm21.exe, ld21.exe, spl21.exe;
- у каталозі \Sources знайти заготовки програм (файли dra.dsp, dps.dsp і dpm.dsp). Скопіювати їх у свій робочий каталог;
- за допомогою текстового редактора змінити текст програм відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модулі, що виконуються, (dra.exe, dps.dsp і dpm.dsp) ;
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму dra.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- завантажити в інтегроване середовище програму dps.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- завантажити в інтегроване середовище програму dpm.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- програми додавання, віднімання і множення дробових чисел (потрійна точність) ;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab2). Скопіювати в нього файли asm21.exe, ld21.exe, spl21.exe;
- у каталозі \Sources знайти заготовки програм (файли tra.dsp, tps.dsp і tpm.dsp). Скопіювати їх у свій робочий каталог;

- за допомогою текстового редактора змінити текст програм відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модулі, що виконуються, (tra.exe, tps.exe і tpm.exe) ;
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму tra.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- завантажити в інтегроване середовище програму tps.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- завантажити в інтегроване середовище програму tpm.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним.

6.2.2 Програми множення цілих чисел (подвійна і потрійна точність):

- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab2). Скопіювати в нього файли asm21.exe, ld21.exe, spl21.exe;
- у каталозі \Sources знайти заготовки програм (файли idpm.dsp і itpm.dsp). Скопіювати їх у свій робочий каталог;
- за допомогою текстового редактора змінити текст програм відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модулі, що виконуються, (idpm.exe і itpm.exe) ;
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- Завантажити в інтегроване середовище програму idpm.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;

- завантажити в інтегроване середовище програму `itpm.exe` і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним.

6.2.3 Програма розподілу чисел одинарної точності (без знака):

- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, `C:\Work\lab2`). Скопіювати в нього файли `asm21.exe`, `ld21.exe`, `spl21.exe`;
- у каталозі `\Sources` знайти заготовку програми (файл `sdivq.dsp`). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм `ASM21` і `LD21` одержати модуль, що виконується, (`sdivq.exe`);
- запустити програму `Visual DSP Debugger`. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму `sdivq.exe` і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;
- за допомогою програми `SPL21` розщепити програму `sdivq.exe` для запису в ППЗУ. Формат задається викладачем.

6.2.4 Програма розподілу чисел одинарної точності (зі знаком):

- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, `C:\Work\lab2`). Скопіювати в нього файли `asm21.exe`, `ld21.exe`, `spl21.exe`;
- у каталозі `\Sources` знайти заготовку програми (файл `sdivs.dsp`). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм `ASM21` і `LD21` одержати модуль, що виконується, (`sdivs.exe`);

- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;

- завантажити в інтегроване середовище програму sdivs.exe і виконати її покрокове налагодження. Записати зміни значень у регістрах ЦПОС ADSP-2181. Порівняти отриманий результат з очікуваним;

- За допомогою програми SPL21 розщепити програму sdivs.exe для запису в ППЗУ. Формат задається викладачем.

6.2.5 Програма мовою Сі для перетворення форматів представлення даних. Завдання видає викладач.

6.3 Контрольні питання

6.3.1 Властивості додаткового коду представлення дробових і цілих чисел.

6.3.2 Формати 1.15 і 16.0.

6.3.3 Основні арифметичні операції, які можна виконати на ЦПОС сімейства ADSP-21XX.

6.3.4 Операція нормалізації дробових чисел. Як вона реалізована в сигнальному процесорі ADSP-2181.

6.3.5 Динамічний діапазон представлення чисел.

6.3.6 Призначення і використання програми ASM21.

6.3.7 Призначення і використання програми LD21.

6.3.8 Призначення і використання програми SPL21.

6.3.9 Етапи розробки програм для ЦПОС сімейства ADSP-21XX.

6.3.10 Структура програми мовою асемблера ADSP-21XX.

6.3.11 Особливості використання пам'яті даних і пам'яті програм ЦПОС ADSP-2181.

6.3.12 Порівняння можливостей програм SIM21 і Visual DSP Debugger.

7 ЛАБОРАТОРНА РОБОТА № 7 УЧБОВИЙ СТЕНД ADDS2181 EZ-KIT LITE

Мета роботи: вивчити пристрій і принцип дії учбового стенда ADDS2181 EZ-KIT Lite. Придбати практичні навички роботи з ним. Ознайомитися з методикою, що рекомендується виробником розробки

систем обробки сигналів і методами налагодження програм, що рекомендує виробник.

7.1 Завдання

7.1.1 Налагодження програми мовою асемблера ADSP-2181.

7.1.2 Налагодження програми мовою Сі для ADSP-2181.

7.1.3 Програма мовою асемблера для налагодження в середовищі Visual DSP.

7.1.4 Програма мовою асемблера для стенда ADDS-2181 EZ-KIT Lite.

7.1.5 Розробка архітектурного файлу системи.

7.1.6 Підготовка файлу для прошивання ППЗУ.

7.2 Порядок виконання роботи

7.2.1 Налагодження програми асемблера ADSP-2181:

- одержати завдання у викладача;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab3);
- скопіювати в нього файли asm21.exe і ld21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;
- за допомогою програми EZ-KIT Host завантажити модуль, що виконується, у пам'ять програм сигнального процесора і виконати його. Використовувати кнопку RESET для виконання апаратного скидання стану;
- за допомогою програми Visual DSP Debugger виконати покрокове виконання тієї ж самої програми. Скласти таблицю зміни значень у регістрах процесора;
- зміст командного файлу і таблиці зміни значень у регістрах процесора представити в звіті.

7.2.2 Налагодження програми мовою СІ для ADSP-2181:

- одержати завдання у викладача;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab3);

- скопіювати в нього файли asm21.exe, ld21.exe, g21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти .BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;
- за допомогою програми EZ-KIT Host завантажити модуль, що виконується, у пам'ять програм сигнального процесора і виконати його. Використовувати кнопку RESET для виконання апаратного скидання стану;
- за допомогою програми Visual DSP Debugger виконати покрокове виконання тієї ж самої програми. Скласти таблицю зміни значень у регістрах процесора;
- зміст командного файлу і таблиці зміни значень у регістрах процесора представити в звіті.

7.2.3 Програма мовою асемблера для налагодження в середовищі Visual DSP:

- одержати завдання у викладача;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab3);
- скопіювати в нього файли asm21.exe і ld21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти .BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;
- за допомогою програми EZ-KIT Host завантажити модуль, що виконується, у пам'ять програм сигнального процесора і виконати його. Використовувати кнопку RESET для виконання апаратного скидання стану;
- за допомогою програми Visual DSP Debugger виконати покрокове виконання тієї ж самої програми. Скласти таблицю зміни значень у регістрах процесора;
- зміст командного файлу і таблиці зміни значень у регістрах процесора представити в звіті.

7.2.4 Програма мовою асемблера для стенда ADDS-2181 EZ-KIT Lite:

- одержати завдання у викладача;

- створити на диску свій робочий каталог (наприклад, C:\Work\lab3).
- скопіювати в нього файли asm21.exe і ld21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти .BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;
- за допомогою програми EZ-KIT Host завантажити модуль, що виконується, у пам'ять програм сигнального процесора і виконати його Використовувати кнопку RESET для виконання апаратного скидання стану;
- за допомогою програми Visual DSP Debugger виконати покрокове виконання тієї ж самої програми. Скласти таблицю зміни значень у регістрах процесора;
- зміст командного файлу і таблиці зміни значень у регістрах процесора представити в звіті.

7.2.5 Розробка архітектурного файлу системи:

- одержати завдання у викладача;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab3) ;
- скопіювати в нього файл bld21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти .BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;
- скласти таблицю розподілення ресурсів системи;
- зміст командного файлу і таблиці представити у звіті.

7.2.6 Підготовка файлу для прошивання ППЗУ:

- одержати завдання у викладача;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab3);
- скопіювати в нього файл spl21.exe;
- за допомогою текстового редактора скласти текст програм відповідно до завдання;
- скласти .BAT-файл пакетного виконання завдання;
- у пакетному режимі виконати завдання;

- одержати 16-річний дамپ розщепленої версії модуля програми, що виконується;
- зміст командного файлу, вихідний текст програми і дамп її версії для запису в ППЗУ представити в звіті.

7.3 Контрольні питання

- 7.3.1 Склад стенду ADDS-2181 EZ-KIT Lite. Призначення його блоків.
- 7.3.2 Технічні характеристики стенда ADSP-2181 EZ-KIT Lite.
- 7.3.3 Призначення і використання програми EZ-KIT Lite Host.
- 7.3.4 Призначення і використання програми EZ-KIT Lite Monitor.
- 7.3.5 Доступні способи налагодження ПО для ЦПОС ADSP-2181.
- 7.3.6 Призначення і використання програми Visual DSP.
- 7.3.7 Призначення і використання програми BLD21.
- 7.3.8 Призначення файлу опису архітектури (.SYS).
- 7.3.9 Особливості використання мови Сі при написанні програм для сигнальних процесорів.
- 7.3.10 Призначення і використання програми G21.
- 7.3.11 Склад і використання Run-time Library для мови Сі.
- 7.3.12 Структура програми мовою Сі для ЦПОС ADSP-2181.

8 ЛАБОРАТОРНА РОБОТА № 8 МОВА АСЕМБЛЕРА ЦПОС ADSP-2181

Мета роботи: вивчити інструкції асемблера процесора ADSP-2181. Придбати практичні навички написання програм для нього і виконання їх на симуляторі Visual DSP Debugger.

8.1 Завдання

- 8.1.1 Скласти програму по заданій блок-схемі алгоритму (рис. 8.1, а).
- 8.1.2 Скласти програму по заданій блок-схемі алгоритму (рис. 8.1, б).

8.1.3 Скласти програму по заданій блок-схемі алгоритму (рис. 8.1, в).

8.1.4 Складання програми по заданій таблиці зміни значень у регістрах процесора.

8.2 Порядок виконання роботи

8.2.1 Складання програми по заданій блок-схемі алгоритму:

- вивчити задану блок-схему алгоритму;
- у каталозі \Sources знаходиться файл sample.dsp, що містить приклад програми, що працює по заданому алгоритму;
- створити на диску свій робочий каталог, наприклад, C:\work\lab4. Скопіювати в нього файли sample.dsp, asm21.exe, ld21.exe і файл опису архітектури;
- у текстовому редакторі змінити зміст програми sample.dsp у відповідності зі своїм бажанням (творча робота) ;
- виконати компіляцію і компонування нової версії файлу sample.exe;
- запустити відладчик Visual DSP і завантажити в інтегроване середовище свою версію файлу, що виконується, sample.exe;
- зробити покрокове виконання програми із середовища симулятора ЦПОС ADSP-2181. Одночасно скласти і заповнити таблицю зміни значень регістрів процесора;
- скласти блок-схему алгоритму програми, заповнивши блоки, що пустують, новим змістом (інструкціями асемблера) ;
- у результаті лабораторної роботи повинні бути отримані всі дані, необхідні для опису роботи програми sample. Записати загальну кількість кроків (машинних тактів) виконання програми.

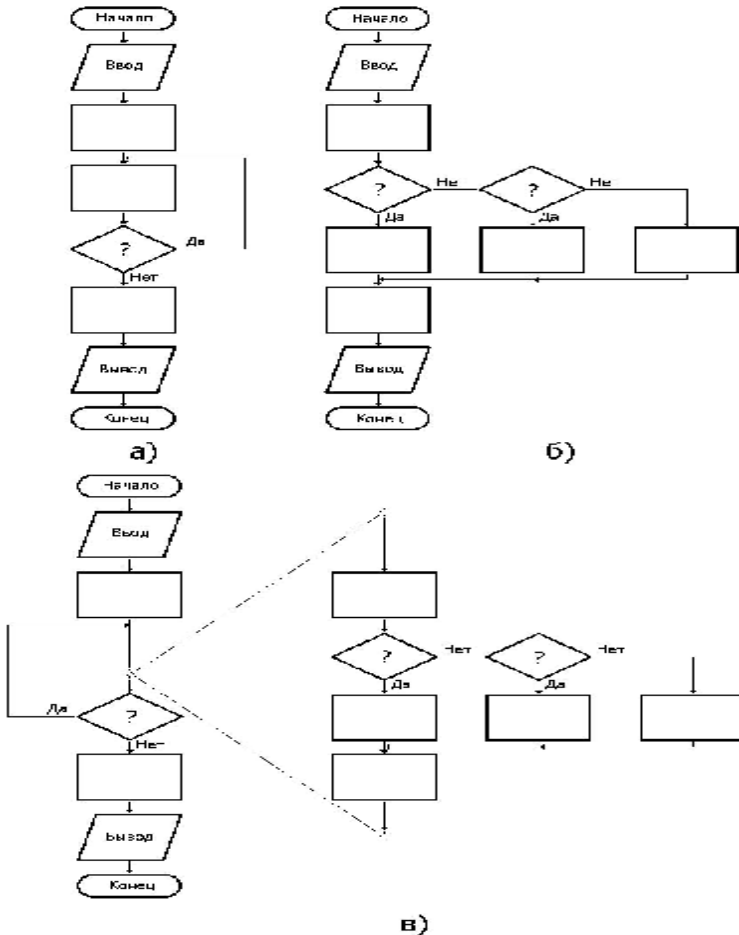


Рисунок 8.1 – Варіанти завдань

8.2.2 Складання програми по заданій таблиці зміни значень у регістрах процесору:

- одержати завдання у викладача. Вивчити задану таблицю зміни значень регістрів процесора;
- скласти блок-схему алгоритму програми, заповнивши блоки необхідним змістом (інструкціями асемблера);
- створити на диску свій робочий каталог, наприклад, C:\work\lab4. Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;

- у текстовому редакторі створити новий документ, що містить текст програми мовою асемблера ЦПОС ADSP-2181. Зберегти його у вигляді текстового файлу sample.dsp;
- виконати компіляцію і компонування файлу sample.exe;
- запустити налагоджувальник Visual DSP і завантажити в інтегроване середовище свою версію файлу, що виконується, sample.exe;
- зробити покрокове виконання програми із середовища симулятора ЦПОС ADSP-2181. При необхідності зробити потрібні зміни у вихідному тексті і заново виконати компіляцію і компонування програми sample.exe;
- у результаті лабораторної роботи треба отримати версію коректно працюючої програми і всі дані, необхідні для опису ходу її роботи. Записати загальну кількість кроків (машинних тактів) виконання програми.

8.3 Контрольні питання

- 8.3.1 Призначення регістрів процесора ADSP-2181.
- 8.3.2 Пряма, лінійна непряма і циклічна непряма адресація в ЦПОС сімейства ADSP-21XX.
- 8.3.3 Пам'ять програм і пам'ять даних процесора ADSP-2181.
- 8.3.4 Переривання ЦПОС ADSP-2181. Регістри керування перериваннями.
- 8.3.5 Організація циклів у ЦПОС сімейства ADSP-21XX.
- 8.3.6 Призначення регістрів ASTAT, SSTAT і MSTAT.
- 8.3.7 Виклик підпрограм і обробка переривань у процесорах сімейства ADSP-21XX.
- 8.3.8 Регістри таймера-лічильника.
- 8.3.9 Послідовні синхронні порти. Керування.
- 8.3.10 Інструкції мови асемблера для блоків ALU, MAC і Shifter.
- 8.3.11 Багатофункціональні інструкції.
- 8.3.12 Інструкції керування ходом програми і пересилання даних асемблера ASM21.

9 ЛАБОРАТОРНА РОБОТА № 9 МАТЕМАТИЧНІ ОБЧИСЛЕННЯ НА ЦПОС ADSP-2181

Мета роботи: вивчити теоретичні зведення, що відносяться до теми лабораторної роботи. Придбати практичні навички проектування програм середньої складності для ЦПОС ADSP-2181.

9.1 Завдання

- 9.1.1 Програма апроксимації функції $y = \sin(x)$.
- 9.1.2 Програма апроксимації функції $y = \sqrt{x}$.
- 9.1.3 Програма апроксимації функції $y = \arctan(x)$.
- 9.1.4 Програма апроксимації функції $y = \log_{10}(x)$.
- 9.1.5 Програма апроксимації функції $y = \ln(x)$.
- 9.1.6 Програма компадування по А-закону (G.911).

9.2 Порядок виконання роботи

9.2.1 Програма апроксимації функції $y = \sin(x)$:

- повторити необхідні для роботи факти з курсу вищої математики;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;
- у каталозі \Sources знайти заготовку програми (файл sin.dsp). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (sin.exe);
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму sin.exe і виконати її. Порівняти отриманий результат з очікуваним;

- використовуючи подібний підхід, скласти таблицю значень функції $\sin(x)$ із кроком 15 градусів.

9.2.2 Програма апроксимації функції $y = \sqrt{x}$:

- повторити необхідні для роботи факти з курсу вищої математики;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;
- у каталозі \Sources знайти заготовку програми (файл sqrt.dsp). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (sqrt.exe);
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- Завантажити в інтегроване середовище програму sqrt.exe і виконати її. Порівняти отриманий результат з очікуваним;
- використовуючи подібний підхід, скласти таблицю значень функції квадратного кореня для цілих чисел на відрізьку [1; 100] з довільним кроком (не менш 20 значень).

9.2.3 Програма апроксимації функції $y = \arctan(x)$:

- повторити необхідні для роботи факти з курсу вищої математики;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;
- у каталозі \Sources знайти заготовку програми (файл arctan.dsp). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (arctan.exe);

- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму arctan.exe і виконати її. Порівняти отриманий результат з очікуваним;
- використовуючи подібний підхід, скласти таблицю значень функції арктангенса для цілих чисел на відрізку [-50; 50] з довільним кроком (не менш 20 значень).

9.2.4 Програма апроксимації функції $y = \log_{10}(x)$:

- повторити необхідні для роботи факти з курсу вищої математики;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;
- у каталозі \Sources знайти заготівку програми (файл log.dsp). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (log.exe);
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму log.exe і виконати її. Зверити отриманий результат з очікуваним;
- використовуючи подібний підхід, скласти таблицю значень функції десяткового логарифма для цілих чисел на відрізку [1; 100] з довільним кроком (не менш 20 значень).

9.2.5 Програма апроксимації функції $y = \ln(x)$.

- повторити необхідні для роботи факти з курсу вищої математики;
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;

- у каталозі \Sources знайти заготовку програми (файл log.dsp). Скопіювати її у свій робочий каталог;
- за допомогою текстового редактора змінити текст програми відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (log.exe);
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму log.exe і виконати її. Порівняти отриманий результат з очікуваним;
- використовуючи подібний підхід, скласти таблицю значень функції натурального логарифма для цілих чисел на відрізку [1; 100] з довільним кроком (не менш 20 значень).

9.2.6 Програма компонування по А-закону (G.911):

- вивчити необхідні для роботи факти, що містяться в стандарті G.911 організації ITU-T (колишня CCITT);
- одержати завдання у викладача. За допомогою калькулятора розрахувати очікуваний результат;
- створити на диску свій робочий каталог (наприклад, C:\Work\lab5). Скопіювати в нього файли asm21.exe, ld21.exe і файл опису архітектури;
- у каталозі \Sources знайти заготовки програм компонування і експандування (файли a_compre.dsp і a_expand.dsp). Скопіювати їх у свій робочий каталог;
- за допомогою текстового редактора змінити вихідний текст програми a_compre.dsp відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (a_compre.exe);
- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму a_compre.exe і виконати її. Порівняти отриманий результат з очікуваним;
- за допомогою текстового редактора змінити вихідний текст програми a_expand.dsp відповідно до завдання;
- за допомогою програм ASM21 і LD21 одержати модуль, що виконується, (a_expand.exe);

- запустити програму Visual DSP Debugger. Налаштувати режим симуляції процесора ADSP-2181;
- завантажити в інтегроване середовище програму a_expand.exe і виконати її. Порівняти отриманий результат з очікуваним;
- зробити висновки про доцільність використання такого способу збільшення динамічного діапазону представлення рівня сигналу.

9.3 Контрольні питання

- 9.3.1 Числові ряди. Апроксимація функції синуса.
- 9.3.2 Числові ряди. Апроксимація функції квадратного кореня.
- 9.3.3 Числові ряди. Апроксимація функції арктангенса.
- 9.3.4 Числові ряди. Апроксимація функції логарифма.
- 9.3.5 Векторна алгебра. Додавання матриць.
- 9.3.6 Векторна алгебра. Множення матриць.
- 9.3.7 Лінійна імпульсно-кодова модуляція.
- 9.3.8 Стандарт G.911. Компадування по А-закону.
- 9.3.9 Стандарт G.911. Компадування по μ -закону.
- 9.3.10 Масштабування. Вибір формату представлення ІКМ-даних.
- 9.3.11 Використання файлів для збереження даних.
- 9.3.12 Диференціальна імпульсно-кодова модуляція.

10 ЛАБОРАТОРНА РОБОТА № 10 ПРОЕКТУВАННЯ FIR-ФІЛЬТРІВ НА ЦПОС ADSP-2181

Мета роботи: вивчити методику проектування цифрових фільтрів з кінцевою імпульсною характеристикою (FIR-фільтрів). Придбати практичні навички цифрової фільтрації. Ознайомитися з принципами ЦОС, що використовуються при проектуванні цифрових фільтрів.

10.1 Завдання

- 10.1.1 Низькочастотний нерекурсивний фільтр. Проектування методом вікна.

10.1.2 Високочастотний нерекурсивний фільтр. Проектування методом вікна.

10.1.3 Смуговий нерекурсивний фільтр. Проектування методом вікна.

10.1.4 Низькочастотний нерекурсивний фільтр. Проектування методом Ремеза.

10.1.5 Високочастотний нерекурсивний фільтр. Проектування методом Ремеза.

10.1.6 Смуговий нерекурсивний фільтр. Проектування методом Ремеза.

10.2 Порядок виконання роботи

10.2.1 Проектування нерекурсивного фільтра методом вікна:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);
- відкрити вікно [125];
- запустити приклад 22. Ознайомитися з принципом роботи FIR-фільтра;
- запустити приклад 23. Ознайомитися зі змістом програми цифрової фільтрації для ЦПОС ADSP-2181;
- одержати завдання на проектування у викладача;
- створити на диску свій робочий каталог, наприклад, C:\work\lab6. Скопіювати в нього файли todat.com, FIR.m, firdemo.dsp, asm21.exe, ld21.exe, файл опису архітектури і файли дані програми FIRDEMO;
- за допомогою програми MatLAB розрахувати порядок і значення коефіцієнтів фільтра методом вікна;
- за допомогою програми todat.com перетворити десяткові значення коефіцієнтів фільтра у формат 16.0. У результаті на диску повинен з'явитися файл data.dat, що містить коректні значення коефіцієнтів нерекурсивного фільтра;
- у текстовому редакторі змінити зміст програми firdemo.dsp відповідно до отриманих результатів;
- виконати компіляцію і компонування нової версії файлу FIRDEMO, що використовує як файл даних отриманий файл коефіцієнтів фільтра;
- скопіювати в робочий каталог програми EZ-KIT Lite Host свій модуль, що виконується, firdemo.exe і файл data.dat. Виключити комп'ютер;
- зібрати схему експерименту (рис. 13.1). Ввімкнути комп'ютер. При необхідності виконати налаштування звукової карти

таким чином, щоб вона забезпечувала одно-часний ввід / вивід аудіоданих (режим full-duplex) ;

- запустити програму SpectraLAB. Налаштувати режим Real Time (Mode -> Real Time);

- Запустити програму EZ-KIT Lite Host, де активізувати демонстраційну програму цифрової фільтрації (Filter). Вибирати номер фільтра, що відповідає розробленому;

- запустити вбудований генератор звуку програми SpectraLAB (Utilities -> Generator). Налаштувати бажаний режим роботи генератора і максимальний рівень сигналу на лінійному виході звукової карти;

- переключачи через <Alt> + <Tab> можна в реальному масштабі часу змінювати режими роботи запущених програм. Мета – одержати на екрані програмного спектрографа чітку картину зміни спектра сигналу на вході і виході стенда EZ-KIT Lite (плюс АЧХ звукової карти). Це – результат цифрової фільтрації;

- за допомогою натискання <PrintScreen> зробити "знімки" зображення, які можна використовувати для ілюстрації звіту. Крім них, у звіт додати список значень коефіцієнтів FIR-фільтра.

10.2.2 Проектування нерекурсивного фільтра методом Ремеза:

- запустити програму ADSP-218x Navigator (файл 218xNavigator.exe);

- відкрити вікно [125];

- запустити приклад 22. Ознайомитися з принципом роботи FIR-фільтра;

- запустити приклад 23. Ознайомитися зі змістом програми цифрової фільтрації для ЦПОС ADSP-218;

- одержати завдання на проектування у викладача;

- створити на диску свій робочий каталог, наприклад, C:\work\lab6. Скопіювати в нього файли todat.com, FIR.m, firdemo.dsp, asm21.exe, ld21.exe, файл опису архітектури і файли дані програми FIRDEMO;

- за допомогою програми MatLAB розрахувати порядок і значення коефіцієнтів фільтра методом Ремеза;

- за допомогою програми todat.com перетворити десяткові значення коефіцієнтів фільтра у формат 16.0. В результаті на диску повинен з'явитися файл data.dat, що містить нові значення коефіцієнтів нерекурсивного фільтра;

- у текстовому редакторі змінити зміст програми firdemo.dsp відповідно до отриманих результатів;

- виконати компіляцію і компонування нової версії файлу FIRDEMO, що використовує як файл даних отриманий новий файл коефіцієнтів фільтра;
- скопіювати в робочий каталог програми EZ-KIT Lite Host модуль, що виконується, firdemo.exe і файл data.dat. Виключити комп'ютер;
- зібрати схему експерименту (рис. 10.1). Ввімкнути комп'ютер. При необхідності виконати налаштування звукової карти таким чином, щоб вона забезпечувала одночасний ввід/вивід аудіоданих (режим full-duplex);
- запустити програму SpectraLAB. Налаштувати режим Real Time (Mode -> Real Time);
- запустити програму EZ-KIT Lite Host, де активізувати демонстраційну програму цифрової фільтрації (Filter). Вибирати номер фільтра, що відповідає розробленому;
- запустити вбудований генератор звуку програми SpectraLAB (Utilities -> Generator). Налаштувати бажаний режим роботи генератора і максимальний рівень сигналу на лінійному виході звукової карти;
- переключаючи через <Alt> + <Tab> можна в реальному масштабі часу змінювати режими роботи запущених програм. Мета – одержати на екрані програмного спектрографа чітку картину зміни спектра сигналу на вході і виході стенда EZ-KIT Lite (плюс АЧХ звукової карти). Це –результат цифрової фільтрації;
- за допомогою натискання <PrintScreen> зробити "знімки" зображення на екрані, які можна використовувати для ілюстрації звіту. Крім них, у звіт додати список значень коефіцієнтів FIR-фільтра.

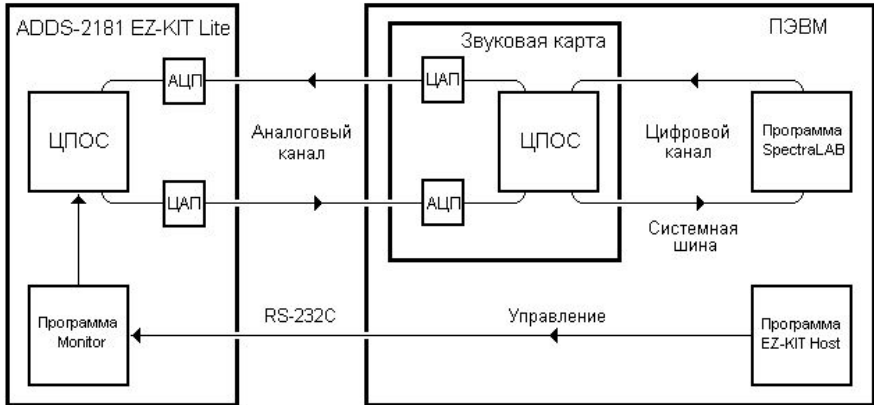


Рисунок 10.1 - Схема лабораторного станда

10.3 Контрольні питання

10.3.1 Класифікація пристроїв фільтрації. Области використання аналогових і цифрових фільтрів.

10.3.2 Згортка сигналу функції відгуку у часовій області.

10.3.3 Ряд Фур'є. Дискретний ряд Фур'є.

10.3.4 Перетворення Фур'є. Дискретне перетворення Фур'є.

10.3.5 Принцип дії і властивості FIR-фільтра.

10.3.6 Теорема Найквіста-Котельникова.

10.3.7 Принцип дії і властивості FIR -фільтрів.

10.3.8 Основні характеристики FIR-фільтра.

10.3.9 Проектування FIR-фільтра. Метод вікна.

10.3.10 Проектування FIR-фільтра. Метод Ремеза.

10.3.11 Блок-схема пристрою цифрової фільтрації.

10.3.12 Організація програми FIRDEMO.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бондарев В.Н., Трестер Г., Чернега В.С. Цифровая обработка сигналов: методы и средства: Учеб. пособие для вузов.- Севастополь: Изд-во СевГТУ, 1999.
2. Руководство пользователя по сигнальным микропроцессорам семейства ADSP-21XX/ Пер. с англ. О.В. Луневой; Под ред. А.Д. Викторова; Санкт-Петербургский государственный электротехнический университет.- Санкт-Петербург, 1997.
3. Шрюфер Э. Обработка сигналов: цифровая обработка дискретизированных сигналов: Учебник/ Под ред. В.П. Бабака.- К.: Либідь, 1995.
4. Гук М. Интерфейсы ПК: справочник – СПб: Питер Ком, 1999. – 416 с.
5. ADSP-21XX Family EZ-KIT Lite Reference Manual, Analog Devices, 1995.
6. 21xx Family Tools Release Notes (for the ADSP-21xx DSP), Analog Devices, 1998
7. Visual DSP Debugger Tutorial (for the ADSP-21xx Family DSP), Analog Devices, 1998.
8. Visual DSP Debugger Guide and Reference, Analog Devices, 1998.
9. ADSP-2181 EZ-KIT Lite Evaluation System Manual, Analog Devices, 2001.
10. ADSP-21XX Family User's Manual, Prentice Hall, 1994.
11. Mixed-signal and DSP Design Techniques, Analog Devices, 2000.
12. Digital Signal Processing Applications Using the ADSP-21XX Family, Vol.1, Prentice Hall, 1992.
13. Digital Signal Processing Applications Using the ADSP-21XX Family, Vol.2, Prentice Hall, 1994.
14. Analog Devices' Data Sheets: AD1847 (Rev.B, 1996), ADSP-21XX Family Development Tools (Rev.A, 1995.)
15. FIR-filters without tears // Analog Dialogue #17-2, 1983.

Додаток А Мікропроцесор ADSP-2181

А.1 Внутрішня будова мікропроцесора ADSP-2181

Мікропроцесор **ADSP-2181** є базовою моделлю серії. ADSP-2181 має 81920 (80K) байт вбудованої пам'яті статичного типу (16384 (16K) 24-розрядних слів ПП і 16384 (16K) 16-розрядних слів ПД). Структурно-функціональна організація мікропроцесора ADSP-2181 зображена на рис.А.1.

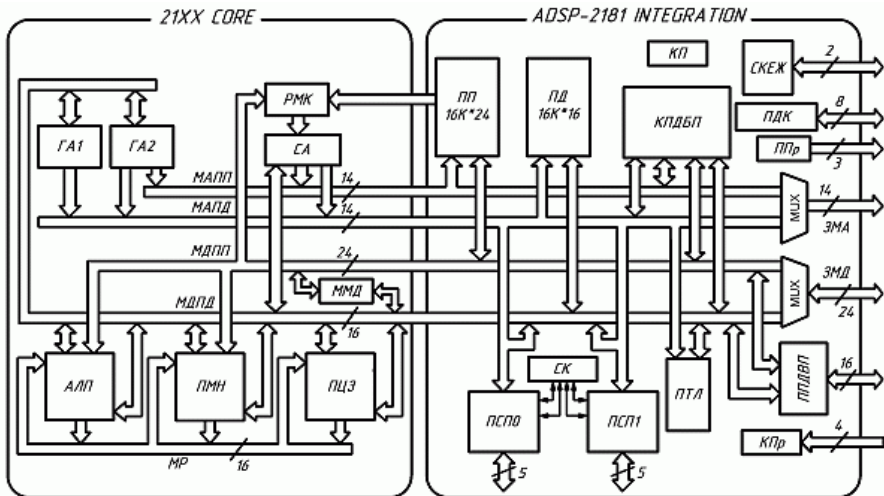


Рисунок А.1 – Структурно-функціональна організація ЦПОС ADSP-2181

Мікропроцесор ADSP-2181 об'єднує на кристалі дві підсистеми: загальне для всіх моделей серії обчислювальне ядро 21XX Core і архітектурні розширення (ADSP-2181 Integration). Склад і характеристики архітектурних розширень залежать від моделі мікропроцесора. В таблиці А.1 перелічені основні структурні елементи мікропроцесора ADSP-2181.

Обчислювальне ядро мікропроцесора містить три операційних блока: АЛП, ПМН і ПЦЗ. За допомогою магістралі результату (МР) проміжні результати з виходу одного блоку можуть бути передані на вхід іншого. Паралельна робота операційних блоків не підтримується.

Наявність незалежних генераторів адреси даних (ГA1, ГA2) і СА дозволяє значно скоротити час виконання команд переміщення даних і керування програмою (цикли, виклики підпрограм, умовні переходи та ін.). Глибина вкладеності апаратно реалізованих циклів не повинна перевищувати чотирьох. Автомат не може обслуговувати рекурсивні виклики підпрограм, число яких перевищує розмір стеку.

Таблиця А.1 – Архітектурні елементи ADSP-2181

Назва (укр.)	Назва (англ.)	Призначення
ГA1, ГA2	DAG1, DAG2	Генератори адреси даних
АЛП	ALU	Арифметично-логічний пристрій
ПМН	MAC	Пристрій множення / накопичування
ПЦЗ	Shifter	Пристрій циклічного зсуву
МАПП	PMA bus	Магістраль адреси пам'яті програм (14 біт)
МАПД	DMA bus	Магістраль адреси пам'яті даних (14 біт)
МДПП	PMD bus	Магістраль даних пам'яті програм (24 біт)
МДПД	DMD bus	Магістраль даних пам'яті даних (16 біт)
МР	R bus	Магістраль результату (16 біт)
ММД	Bus exchange	Міст магістралей даних
РМК	Instruction register	Регістр команд
СА	Program sequencer	Системний автомат
ПП	PM	Пам'ять програм (48Кбайт)
ПД	DM	Пам'ять даних (32Кбайт)
ПСПО, ПСП1	SPORT0, SPORT1	Програмовані синхронні порти
СК	Companding circuit	Схема компандування (ITU-T: G.711)
КПДБП	BDMA	Контролер ПДП байтової пам'яті
ПТЛ	Timer	Програмований таймер-лічильник
СКЕЖ	Power down circuit	Схема керування електроживленням
ПДК	Programmable I/O	Програмовані дискретні канали (8 біт)
ППр	Programmable flags	Програмовані прапори
ЗМА	Address bus	Зовнішня магістраль адреси (14 біт)
ЗМД	Data bus	Зовнішня магістраль даних (24 біт)
ППДВП	IDMA	Порт ПДП внутрішньої пам'яті (16 біт)
КІр	Interrupt controller	Контролер переривань

А.2 Арифметично - логічний пристрій

Арифметично-логічний пристрій призначений для виконання основних адитивних, логічних, бітових операцій і примітиву ділення. Набір регістрів АЛП складається з двох копій регістрів AX0, AX1, AY0, AY1, AR, AF (загалом – дванадцять 16-розрядних регістрів). В табл.А.2 перелічені операційні регістри АЛП.

MF (16 біт) і MR2 (8 біт). Загалом – 16 регістрів. Ввімкнення/вимикання тінювого набору регістрів виконується командами ENA SEC_REG і DIS SEC_REG відповідно.

Додатковий 8-розрядний регістр MR2 використовується для збереження результату виконання операції множення з накопиченням від втрат, пов'язаних з переповненням. Використання регістру MR2 захищає результат навіть тоді, коли виникло $2^8 = 256$ переповнень розрядної сітки.

В табл. А.3 перелічені регістри даних ПМН.

Таблиця А.3 – Регістри ПМН

Регістр / адреса	RGP	DREG	Біт	Призначення
MX0	00	0010	16	Молодше слово операнда X
MX1	00	0011	16	Старше слово операнда X
MY0	00	0110	16	Молодше слово операнда Y
MY1	00	0111	16	Старше слово операнда Y
MR0	00	1011	16	Перше слово результату
MR1	00	1101	16	Друге слово результату
MR2	00	1110	8	Третє слово результату
MF	-	-	16	Проміжний результат

Структурно-функціональна організація ПМН приведена на рис.А.3.

В ПМН використовується матричний спосіб виконання операції множення, який дозволяє отримати результат впродовж одного МЦ. Після множення 16-розрядних операндів отримують 32-розрядний результат. В залежності від формату даних, що перемножувались, результат множення Р може бути записаний в регістр результату MR двома способами:

- якщо виконувалось множення цілих чисел (формат 16.0), то Р просто копіюється в MR;
- якщо перемножувались дробові числа (формат 1.15), то в MR заноситься результат множення, зсунутий на один розряд вліво (див.рис.А.4).

Спосіб розміщення результату ПМН залежить від стану прапору M_MODE (четвертий біт регістру MSTAT).

А.4 Пристрій циклічного зсуву

Пристрій циклічного зсуву призначений для виконання операцій зсуву, нормалізації та знаходження порядку чисел у форматі 1.15.

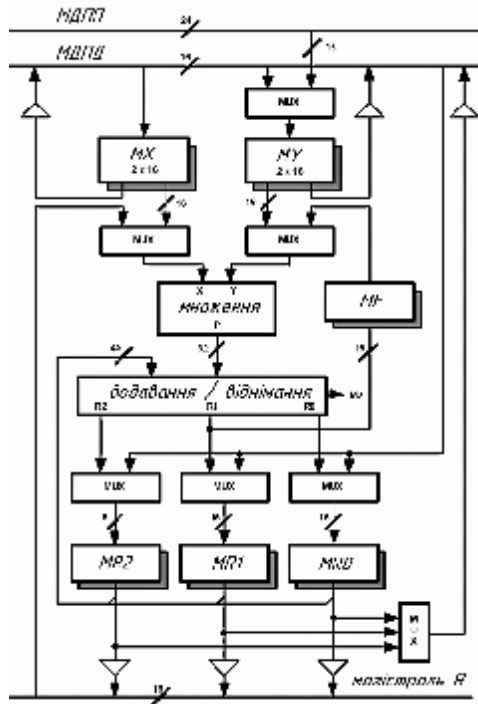


Рисунок А.3 – Пристрій множення / накопичення ADSP-2181

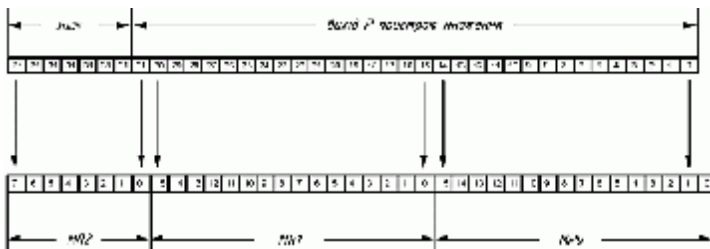


Рисунок А. 4 – Результат множення дробових чисел (формат 1.15)

Набір регістрів ПЦЗ складається з двох копій регістрів SI, SR0, SR1 (16 біт), SE (8 біт) і SB (5 біт). Загалом – 10 регістрів. Включення / виключення тінювого набору регістрів виконується командами ENA SEC_REG і DIS SEC_REG відповідно. В табл.А.4 перелічені регістри даних ПЦЗ.

Таблиця А.4 – Регістри ПЦЗ

Регістр / адреса	RGP	DREG	Біт	Призначення
SI	00	1000	16	Вхідний операнд
SE	00	1001	8	Порядок зсуву
SB	11	0110	5	Зсув блоку операндів
SR0	00	1110	16	Перше слово результату ПЦЗ
SR1	00	1111	16	Друге слово результату ПЦЗ

Структурно-функціональна організація ПЦЗ приведена на рис.А.6. Операційний пристрій зсуву складається з 16 регістрів розрядністю 32 біти. До цього регістрового файлу вводиться 16-розрядне значення, положення якого завдають керуючий 8-розрядний код C і сигнали HI / LO. Данні в ПЦЗ надходять через регістр SI.

Регістр результату SR має розрядність 32 біти, він складається з двох 16-розрядних регістрів SR0, SR1. Він має зворотній зв'язок з логічним пристроєм OR/PASS для забезпечення 32-розрядних операцій зсуву. Якщо в команді вказана операція SR OR, нове значення результату записується після логічного АБО з попереднім змістом регістру SR.

Регістр SE (8 біт) містить значення порядку зсуву. Негативне значення порядку призводить до зсуву даних вліво, позитивне значення – до зсуву вправо. Значення цього регістру може бути визначено після виконання операції визначення порядку EXP (SFT25 – SFT30) або задано безпосередньо.

Регістр SB (5 біт) призначений для знаходження максимального порядку блоку чисел з плаваючою крапкою (операції SFT31, SFT32).

Після зсуву всі біти праворуч від вхідного значення в регістрі результату заповнюються нулями, а біти ліворуч – додатковим бітом X. Значення біту X залежить від команди, що виконується. Це може бути самий старший біт значення регістру SI (ASHIFT), біт AC регістру ASTAT (зсув значення AR, в якому зберігається результат операції, яка призвела до переповнення (модифікатор (HIX))) або нуль (LSHIFT).

Модифікатори (HI) і (LO) вказують місце положення вхідного 16-розрядного значення в 32-розрядному регістровому файлі ПЦЗ. Наприклад, якщо треба виконати зсув 32-розрядного слова вправо.

– для старших 16 біт вхідного значення виконується операція ASHIFT з модифікатором (HI). Запис результату – без операції SR OR;

– для молодших 16 біт вхідного слова виконується операція LSHIFT з модифікатором (LO). Запис результату – з операцією SR OR.

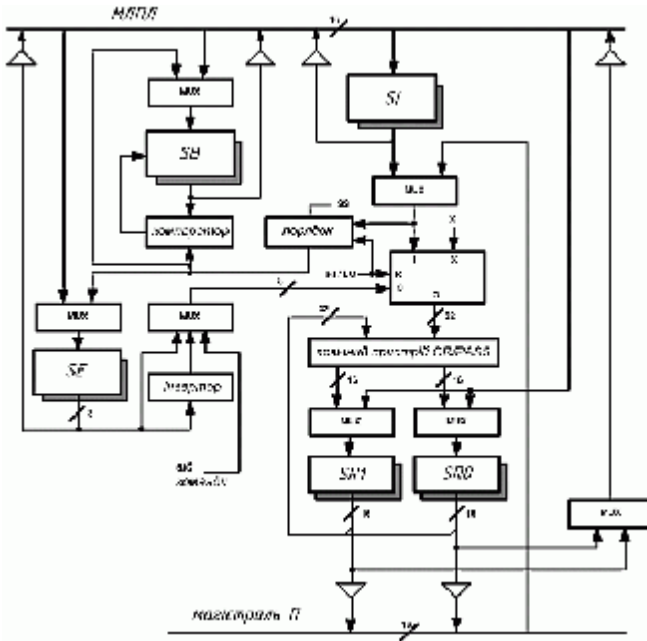


Рисунок А.6 – Пристрій зсуву ADSP-2181

А.5 Генератори адреси

Майже усі ЦПОС, у тому числі ADSP-2181, широко використовують непряму адресацію тому, що для вирішення переважної більшості задач ЦОС часто необхідно використовувати буфери – масиви однорідних даних. Використання непрямої адресації в ADSP-2181 забезпечують генератори адреси ГА1, ГА2. За їх допомоги можливо швидко отримати доступ до елементів масиву даних.

Генератор адреси ГА1 генерує тільки адреси даних, що зберігаються у ПД, але забезпечує додаткову функцію – розміщення бітів слова адреси у зворотному порядку (біт-інверсна адресація). Генератор ГА2 може формувати адресу даних для обох просторів пам'яті – ПД і ПП.

Структурно-функціональна організація ГА приведена на рис.А.7.

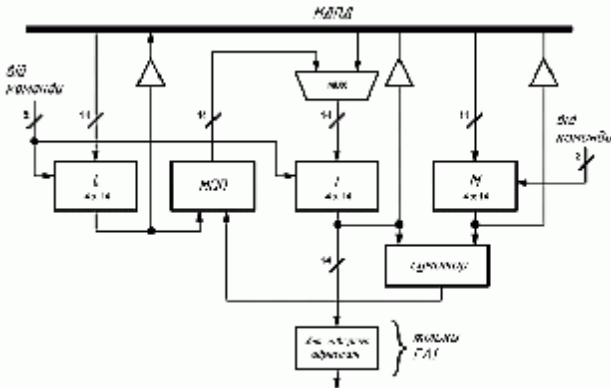


Рисунок А.7 – Генератор адреси ADSP-2181

А.6 Системний автомат

Системний автомат керує процесом виконання команд програми. Він відповідає за апаратну підтримку основних режимів роботи мікропроцесора:

- послідовне виконання команд;
- виконання циклів DO UNTIL (у тому числі вкладених);
- реалізацію умовних і безумовних переходів (JUMP);
- обробку викликів переривань (у тому числі вкладених);
- обслуговування викликів підпрограм CALL;
- повернення керування після виконання підпрограми RTS;
- повернення керування з процедури обслуговування переривання RTI;
- керування виконанням багатofункціональних команд;
- апаратне опитування прапору FI;
- включення / виключення режиму економічного електроживлення IDLE;

- включення / виключення режиму зменшення частоти МЦ (IDLE n);

- включення / виключення режиму GO, таймеру та ін.

Структурно-функціональна організація системного автомату приведена на рис.А.8. На вхід мікропрограмованого автомату (МПА) СА надходять логічні умови (ЛУ), які вказують місцезнаходження адреси наступної команди програми. Ця адреса може зберігатися в одному з п'яти джерел:

- пристрій інкременту лічильника команд (PC);
- стек лічильника команд (стек PC);
- регістр команд ;
- контролер переривань ;
- індексний регістр ГА1.

Лічильник команд PC має розрядність 14 біт і містить адресу поточної команди. На лінійних ділянках програми значення PC нарощується пристроєм інкременту. Лічильник команд зв'язаний зі стеком PC, в якому може зберігатися до шістнадцяти 14-розрядних слів. Під час виконання викликів підпрограм, організації циклів DO UNTIL або обслуговування переривання, до стеку лічильника команд автоматично записується вихідне значення пристрою інкрементора (адреса наступної команди програми). По закінченні роботи цикл, підпрограма та процедура обслуговування переривання повертають керування основній програмі і зміст стеку лічильника команд автоматично відновлюється.

Значення, яке зберігається на вершині стеку PC ідентифікує псевдорегістр TOPPCSTACK. За його допомогою воно може бути прочитане / записане програмними засобами.

Лічильник CNTR використовується циклом DO UNTIL і має розрядність 14 біт. Якщо потрібно організувати N повторів виконання циклічної ділянки програми, до регістру CNTR записують число N-1. Після запису у регістр CNTR, його значення автоматично завантажується до стеку лічильника циклу.

Стек лічильника призначений для зберігання не більше чотирьох 14-розрядних слів – лічильників вкладених циклів. На вершині циклу міститься лічильник циклу, який має найбільший рівень вкладеності. Для того, щоб змінити значення поточного лічильника циклу, у програмі можна використовувати спеціальний регістр завантаження лічильника циклу OWRCNTR. Після запису у регістр OWRCNTR, його значення не завантажується до стеку лічильника циклу, а передається у регістр CNTR.

Стек циклів складається з чотирьох 18-розрядних регістрів. Він містить адреси останніх команд циклічних ділянок програми (14 біт)

та код логічної умови закінчення тимчасово затриманих циклів (4 біти). Код ЛУ закінчення циклу TERM передається у контексті машинної команди організації циклу DO UNTIL.

Під час передачі / повертання керування основній програмі мікропроцесор відновлює зміст регістрів стану, зі стеку PC вилучає значення лічильника команд і продовжує роботу. Стек станів може зберігати не більше дванадцяти 25-розрядних слів. Кожне слово стеку станів складається зі змісту основних регістрів стану ASTAT (8 біт), MSTAT (7 біт) та IMASK (10 біт).

Інформацію про стан стеків СА можна прочитати з регістру SSTAT. В табл.1.1 наведений опис його прапорів. Переповнення будь-якого стеку означає постійну втрату даних. Тому активний стан прапорів переповнення регістру SSTAT фіксується.

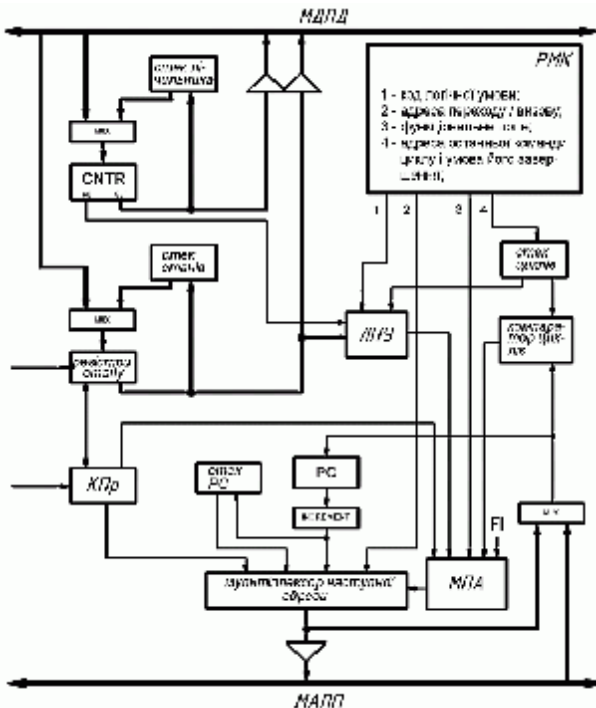


Рисунок А.8 – Системний автомат ADSP-2181

Додаток Б

Опис стенда ADDS-2181 EZ-KIT Lite

Апаратна частина учбового стенда являє собою друковану плату розміром 3.5 x 5.5 дюймів. На ній знаходяться ПЦОС ADSP-2181, ППЗУ, звуковий кодек AD1847, різні додаткові ланцюги і рознімання (рис. Б.1). Він може використовуватися як платформа для розробки нових програм для ADSP-2181.

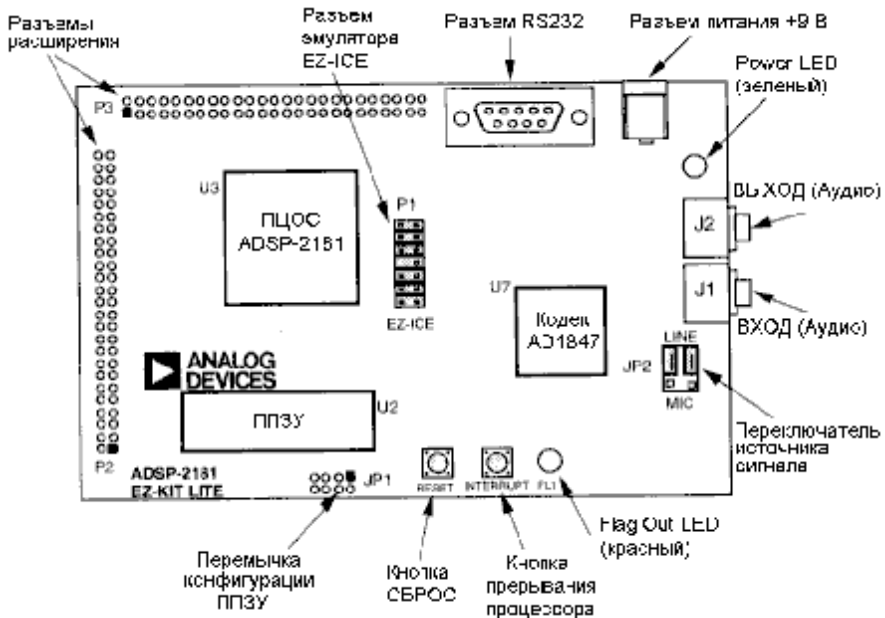


Рисунок Б.1— Зовнішній вигляд плати стенда ADDS-2181 EZ-KIT Lite

Стенд EZ-KIT Lite - приклад мінімальної реалізації системи ЦОС на процесорі ADSP-2181. ППЗУ з'єднано з процесором через порт BDMA. Цей інтерфейс використовує тільки вісім з 24 ліній для передачі даних (D8-D15). Вісім інших каналів даних (D16-D23) використовуються для передачі додаткових розрядів адреси. Це дозволяє ADSP-2181 адресувати до 32 Мбіт (4 Мбайт) пам'яті. Гніздо на друкованій платі може бути використане для установки мікросхем ППЗУ ємністю від 256 до 8 Мбіт. Перемичка JP1 забезпечує

необхідний спосіб керування висновками 3 і 30 ці гнізда, у залежності від мікросхеми ППЗУ, яка використовується. Процесор, підключений так, що після скидання системи він автоматично завантажує програму з ППЗУ і починає її виконувати. Порт IDMA процесором не використовується. Усі сигнали IDMA доступні на розніманні P3.

Кодек AD1847 з'єднаний із ПЦОС через SPORT0. Цей високошвидкісний синхронний послідовний порт передає дані, команди керування та інформацію про стан між ADSP-2181 і кодеком. Є можливість відключення кодека, якщо послідовний порт планується використовувати для іншої мети. Для цього може використовуватися сигнал CODECDIS, доступний на розніманні P3. Коли цей сигнал має низький логічний рівень, кодек заблокований, а сигнали знаходяться в стані високого імпедансу.

Канал SPORT1 використовується для зв'язку з хост-машиною через інтерфейс RS-232 (U5). Висновки Flag In і Flag Out сигналізують про прийом/передачу даних. Прийняті дані також йдуть до IRQ1, так що процесор може знайти їх без опитування стану висновку Flag In. Програма прийому/передачі ПЦОС емулює роботу універсального асинхронного приймача-передавача (УАПП, англ. UART) для забезпечення відповідного протоколу асинхронного послідовного зв'язку зі швидкістю передачі 9600 біт/сек.

Таблиця Б.1 – Технічні характеристики стенда ADDS-2181 EZ-KIT Lite

Основні характеристики	
Процесор	ADSP-2181KS-133 (частота 33 МГц, зовнішнє джерело синхроімпульсів 16,667 МГц)
Аналоговий інтерфейс	Звуковий стерео кодек AD1847
Аналогові входи	Одна пара лінійних стерео входів (рівень до 2 В)
	Одна пара мікрофонних стерео входів (рівень до 20 мВ)
Аналоговий вхід	Одна пара лінійних стерео виходів (рівень до 1 В)
Джерело постійного струму	8-10 В, 300 мА
Умови експлуатації	Температура 0-700°C, відносна вологість 10-90 %
Призначення рознімів	
J1	Вхід звуковий, стерео (3,5 мм).
J2	Вихід звуковий, стерео (3,5 мм).
J3	Цифрове 9-контактне рознімання (інтерфейс RS-232 для зв'язку з ЕОМ).
J4	Штирькове рознімання напруги +9В (зовнішній діаметр 5,5 мм (+ джерело напруги), внутрішній діаметр 2 мм (- джерело напруги).
JP1	Використовується для конфігурування ППЗУ (8 контактів).
JP2	Використовується для налаштування рівня вхідного сигналу, що надходить на рознімання J1 (лінійний вхід або мікрофон). Має 6 контактів, розділених на дві групи (правий і лівий канал). Середній контакт кожної групи приєднаний до входу Line 1 кодека AD1847. Для більш точного регулювання рівнем вхідного сигналу можна використовувати додатковий ланцюг узгодження.
P1	14-контактний роз'єм для підключення внутрішньо-схемного емулятора EZ-ICE. Вихід 7 відіграє роль ключа (повинен бути вилучений).
P2, P3	50-контактні рознімання розширення. Можуть використовуватися для розширення функціональних можливостей стенда або для тестування.
U2	Гніздо для встановлення мікросхеми ППЗУ. Можуть бути використані мікросхеми 27C512 (ємність 64 Кбайт), 27C010 (128Кбайт), 27C256 (32Кбайт), 27C020 (256Кбайт), 27C040 (512Кбайт) або 27C080 (1Мбайт). Тип установлюваної мікросхеми ППЗУ задається за допомогою JP1.
R2 8	Місце для резистора з нульовим опором (перемички). Якщо цей резистор встановлений, то процесор одержує можливість

	самостійно робити скидання апаратної частини стенда під керуванням програми. Для цього в програмі для ADSP-2181 необхідно використовувати можливість конфігурування PF0.
R2 9	Місце для перемички. Якщо її встановити і видалити X3 і C37, то кодек зможе використовувати сигнал ADSP-2181 CLKOUT замість власного кварцового генератора 24.576 МГц. У цьому випадку, для узгодження процесора і кодека, необхідно замінити X1 на більш низьке значення частоти.
Призначення перемикачів	
S1	Кнопка скидання. Натискання цієї кнопки призводить до скидання ADSP-2181 і AD1847. Має захист від деренчання контактів.
S2	Кнопка переривання. Натискання цієї кнопки змушує процесор виконати процедуру переривання IRQE (якщо воно передбачено в програмі і вектор переривання не змінив своє значення). Має захист від деренчання контактів.