МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Санкт-Петербургский государственный технологический институт
(технический университет)»
СПбГТИ(ТУ)

УГНС	09.00.00	Информатика и вычислительная техника
Направление подготовки	09.03.01	Информатика и вычислительная техника
Направленность (профиль)		Системы автоматизированного проектирования
Форма обучения		очная
Факультет		Информационных технологий и управления
Кафедра		Систем автоматизированного проектирования и управления
Учебная дисциплина		**Вычислительные системы, сети и телекоммуникации**
Курс II		Группа 494

КУРСОВОЙ ПРОЕКТ

Тема: **Разработка программ преобразования форматов двоичных данных и сортировок в машинных кодах с помощью эмулятора на ПК**

Задача: **Составить программу формирования массива целых положительных однобайтных двоичных чисел без знака, соответствующих исходному массиву символов**

Студент	_____	Гусев А.А.
Руководитель, доцент, к.т.н.	_____	Макарук Р.В.
Оценка за курсовой проект	_____	_____

Санкт-Петербург
2021

Содержание

Введение

Курсовой проект состоит из аналитической и практической частей. В первой необходимо подготовить теоретический материал на тему: «Двоично-десятичное кодирование. Арифметические действия над ДД-кодами. Стандарты кодирования текстов». Также необходимо привести примеры арифметики с ДД-кодами на основе чисел из таблицы, приведенной в индивидуальном задании к курсовому проекту. Практическая часть заключается в разработке алгоритма и программной реализации на эмуляторе микро-ЭВМ СМ-1800 задачи преобразования массива констант в массив целых положительных однобайтных двоичных чисел без знака. Заданные значение констант:

Таблица 1 — Заданные значения констант

Адрес$_{16}$	**Константа$_{16}$**	**Адрес$_{16}$**	**Константа$_{16}$**
5000	39 37	500A	33 30
5002	33 38	500C	31 34
5004	31 32	500E	39 39
5006	30 30	5010	30 37
5008	34 35	5012	36 31

Будем рассматривать эти коды как массив кодов КОИ-7, только что введенных с клавиатуры двузначных десятичных чисел (например, 33 38 – это число 38_{10}).

Необходимо составить программу формирования массива целых положительных однобайтных двоичных чисел без знака, соответствующих исходному массиву символов (с адреса 5000_{16}). Результирующий массив записать с адреса 7000_{16}. Программу располагать в памяти с ячейки 4000_{16}.

Целью данного курсового проекта является изучение архитектуры МикроЭВМ с помощью эмулятора СМ1800, команд МикроЭВМ, написание алгоритма, программного кода и его реализация.

1 Аналитическая часть

1.1 Двоично-десятичное кодирование

Двоично-десятичный код (ДД код) — форма записи рациональных чисел, когда каждый десятичный разряд числа записывается в виде его четырёхбитного двоичного кода. Таким образом, каждая тетрада двоично-десятичного числа может принимать значения от 0000_2 (0_{10}) до 1001_2 (9_{10}). При помощи 4 бит можно закодировать 16 цифр. Из них используются 10. Остальные 6 комбинаций в двоично-десятичном коде являются запрещёнными.

Таблица 2 — Соответствия двоично-десятичного кода и десятичных цифр

Двоично-десятичный код				**Десятичный код**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Запрещенные комбинации применяются в основном в телефонной связи. В данном случае помимо десятичных цифр кодируются специальные символы при помощи запрещенных комбинаций.[3]

Таблица 3 — Использование запрещенных комбинаций

Двоично-десятичный код				**Спец. символы**
1	0	1	0	*
1	0	1	1	#
1	1	0	0	+
1	1	0	1	—
1	1	1	0	,
1	1	1	1	Символ гашения

Преимуществами данного вида кодирования информации можно считать упрощенный вывод чисел на индикацию — вместо последовательного деления на 10 требуется просто вывести на индикацию каждый полубайт, а также отсутствие потери точности для дробных чисел при переводе в десятичный формат представления и наоборот. Стоит отметить, что производить операции умножения и деления на 10, а также округления чисел значительно проще при работе с двоично-десятичным кодом. В основном использовать данное представление чисел используется в калькуляторах благодаря ряду приведенных преимуществ.

Однако, двоично-десятичный код имеет и недостатки. Среди них: требование большей памяти, а также некоторое усложнение арифметических операций.

1.2 Арифметические операции над ДД-кодами

Операции двоично-десятичной арифметики выполняются в два этапа:

- двоично-десятичные числа обрабатываются как двоичные коды.
- выполняется коррекция результата с целью получения двоично-десятичного числа.

Обработка двоично-десятичных чисел на компьютере выполняется побайтно. При этом надо учитывать следующие флаги: F-перенос между байтами, AF-вспомогательный перенос (между двоично-десятичными цифрами в байте)[4].

Сложение:

Операция сложения выполняется в два этапа: сложение и коррекция. После сложения в тетрадах может оказаться недопустимая комбинация или из тетрады может возникнуть перенос. Алгоритм коррекции состоит из двух шагов:

Если AF=1 или младшая тетрада меньше 9, но больше 15, то к ней прибавляется код 6. Возникающий перенос прибавляется к старшей тетраде.

Если CF=1 или старшая тетрада содержит недопустимую комбинацию, то к результирующему байту прибавляется код 6[4].

Ниже приведен пример сложения двух чисел, взятых из таблицы индивидуального задания, представленных двоично-десятичном формате.

$A = 97_{10} = 1001\ 0111$

$B = 99_{10} = 1001\ 1001$

A + B =

```
  1001 0111
+
  1001 1001
----------
```

0001 1001 0110

В данном примере числа складываются тетрадами или полубайтами, начиная справа. Как можно заметить, младший результирующий полубайт должен был получиться 0000 с переносом единицы в старший полубайт, однако, как было отмечено ранее, при переполнении тетрады (то есть при числе > 15_{10}) к ней прибавляется код 6_{10} (0110). Во второй тетраде переполнения не происходит. Единица переносится в тетраду выше. Результат уже состоит из трёх тетрад 0001 1001 0110 - 196_{10}

Вычитание:

Данная операция также выполняется в два этапа: вычитание и коррекция, обусловленная возможностью заёма при вычитании или получения недопустимых значений результата.

Эта коррекция тоже состоит из двух частей:

- Если AF=1 или младшая тетрада содержит недопустимую комбинацию, то из нее вычитается код 6 (прибавляется код 10).Флаги AF и CF устанавливаются в соответствии с правилами установки флагов при вычитании.
- Если CF=1 или старшая тетрада содержит недопустимую комбинацию, то из нее так же вычитается код 6 (прибавляется код 10)[4].

A = 97_{10} = 1001 0111

B = 39_{10} = 0011 1001; Вдоп = 1100 0111 — обратная + 1

A — B =

```
  1001 0111
+
  1100 0111
---------------
  0101 1000
```

Вычитание в двоично-десятичных кодах производится путём складывания прямой записи числа A и дополненной записи числа B, которая строится следующим образом: необходимо инвертировать все полубайты числа и к результату прибавить единицу (например, пусть x = 42_{10}, $x_{пр}$ = 0010 1010 => $x_{обр}$ = 1101 0101 => $x_{доп}$ = 1101 0110). В данном примере в младшей тетраде получается запрещенное значение (1110), поэтому необходимо вычесть код 6 (или же прибавить код 10, что является дополненной записью кода 6). После этого младшая тетрада становится равна 1000 = 8_{10}. Результат 99_{10} - 39_{10} = 58_{10} = 0101 1000.

1.3 Стандарты кодирования текстов

Результатом необходимости стандартизации представления текстовой информации явилась кодировка ASCII — стандартная американская кодировка для обмена информацией.

Несмотря на то, что существует стандарт, несовместимые или частично совместимые с ним варианты кодировок продолжают существовать. Основные проблемы, связанные с кодировками, возникли в тот момент, когда компьютеры распространились за пределы англоязычных стран, а затем и стран с латинским алфавитом. Появилась проблема совмещения латинского и национального алфавита в одной кодировке. Она состоит в том, что текст, который создан в одной кодировке, при использовании другой представляет собой набор символов, лишенных всякого смысла[3].

Программисты помнят машины линий СМ и ДВК (советские аналоги американской фирмы DEC), в которых использовались семибитовая кодировка КОИ-7. другими словами, с ее помощью можно было представить не более 128 символов, многие из которых нельзя было переопределить. В результате программист должен был выбирать один из трех вариантов одной и той же кодировки:

- латинский — со строчными и заглавными буквами
- кириллический — со строчными и заглавными буквами
- смешанный — с заглавными латинскими и русскими буквами

Непосредственный перенос текста с ДВК (КОИ-7 в трех вариациях) на PC (ASCII) был невозможен без специальных средств преобразования кодов.

Что касается принятой для PC восьмибитовой (256 символов) кодировки ASCII, то и здесь поначалу применялось не менее трех вариантов расположения букв кириллицы. В конце концов выжил вариант, известный как CP 866[3].

UNIX принес с собой кодировку DEC КОИ-8 и ее кириллический вариант KOI-8r, который, кстати считается фактическим стандартом для передачи русскоязычной информации и ее представления в сети.

b7	0	0	0	0	1	1	1	1
b6	0	0	1	1	0	1	1	1
b5	0	1	0	1	0	1	0	1

B7	B6	b5	b4	b3	b2	b1	No п.п	0	1	2	3	4	5	6	7
			0	0	0	0	0	ПУС	АР1	ПР	0	ю	п	Ю	П
			0	0	0	1	1	НЗ	СУ1	!	1	а	я	А	Я
			0	0	1	0	2	НТ	СУ2	"	2	б	р	Б	Р
			0	0	1	1	3	КТ	СУ3	#	3	ц	с	Ц	С
			0	1	0	0	4	КП	СУ4	$	4	д	т	Д	Т
			0	1	0	1	5	КПМ	НЕТ	%	5	е	у	Е	У
			0	1	1	0	6	ДА	СИН	&	6	ф	ж	Ф	Ж
			0	1	1	1	7	ЗВ	КБ	'	7	г	в	Г	В
			1	0	0	0	8	ВШ	АН	(	8	х	ь	Х	Ь
			1	0	0	1	9	ГТ	КН	)	9	и	ы	И	Ы
			1	0	1	0	10	ПС	ЗМ	*	:	й	з	Й	З
			1	0	1	1	11	ВТ	АР2	+	;	к	ш	К	Ш
			1	1	0	0	12	ПФ	РИ4	,	<	л	э	Л	Э
			1	1	0	1	13	ВК	РИ3	-	=	м	щ	М	Щ
			1	1	1	0	14	ВЫХ	РИ2	.	>	н	ч	Н	Ч
			1	1	1	1	15	ВХ	РИ1	/	?	о	ъ	О	DEL

Рисунок 1 — Таблица КОИ-7

Нужно было спасать положение в плане совместимости таблиц кодировки. Поэтому, со временем были разработаны новые обновлённые стандарты. В настоящее время наиболее популярной является кодировка под названием UNICODE. В ней каждый символ кодируется с помощью 2— х байт, что соответствует 216=62536 разным кодам.

0020	0 0030	@ 0040	P 0050	` 0060	p 0070	00A0	° 00B0	À 00C0	Ð 00D0	à 00E0	ð 00F0
! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071	¡ 00A1	± 00B1	Á 00C1	Ñ 00D1	á 00E1	ñ 00F1
" 0022	2 0032	B 0042	R 0052	b 0062	r 0072	¢ 00A2	² 00B2	Â 00C2	Ò 00D2	â 00E2	ò 00F2
# 0023	3 0033	C 0043	S 0053	c 0063	s 0073	£ 00A3	³ 00B3	Ã 00C3	Ó 00D3	ã 00E3	ó 00F3
$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074	¤ 00A4	´ 00B4	Ä 00C4	Ô 00D4	ä 00E4	ô 00F4
% 0025	5 0035	E 0045	U 0055	e 0065	u 0075	¥ 00A5	µ 00B5	Å 00C5	Õ 00D5	å 00E5	õ 00F5
& 0026	6 0036	F 0046	V 0056	f 0066	v 0076	¦ 00A6	¶ 00B6	Æ 00C6	Ö 00D6	æ 00E6	ö 00F6
' 0027	7 0037	G 0047	W 0057	g 0067	w 0077	§ 00A7	· 00B7	Ç 00C7	× 00D7	ç 00E7	÷ 00F7
(0028	8 0038	H 0048	X 0058	h 0068	x 0078	¨ 00A8	¸ 00B8	È 00C8	Ø 00D8	è 00E8	ø 00F8
) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079	© 00A9	¹ 00B9	É 00C9	Ù 00D9	é 00E9	ù 00F9
* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A	ª 00AA	º 00BA	Ê 00CA	Ú 00DA	ê 00EA	ú 00FA
+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B	« 00AB	» 00BB	Ë 00CB	Û 00DB	ë 00EB	û 00FB
, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C	¬ 00AC	¼ 00BC	Ì 00CC	Ü 00DC	ì 00EC	ü 00FC
- 002D	= 003D	M 004D	] 005D	m 006D	} 007D	- 00AD	½ 00BD	Í 00CD	Ý 00DD	í 00ED	ý 00FD
. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E	® 00AE	¾ 00BE	Î 00CE	Þ 00DE	î 00EE	þ 00FE
/ 002F	? 003F	O 004F	_ 005F	o 006F	007F	¯ 00AF	¿ 00BF	Ï 00CF	ß 00DF	ï 00EF	ÿ 00FF

Рисунок 2 — Фрагмент таблицы символов UNICODE

2 Практическая разработка

2.1 Блок - схема алгоритма

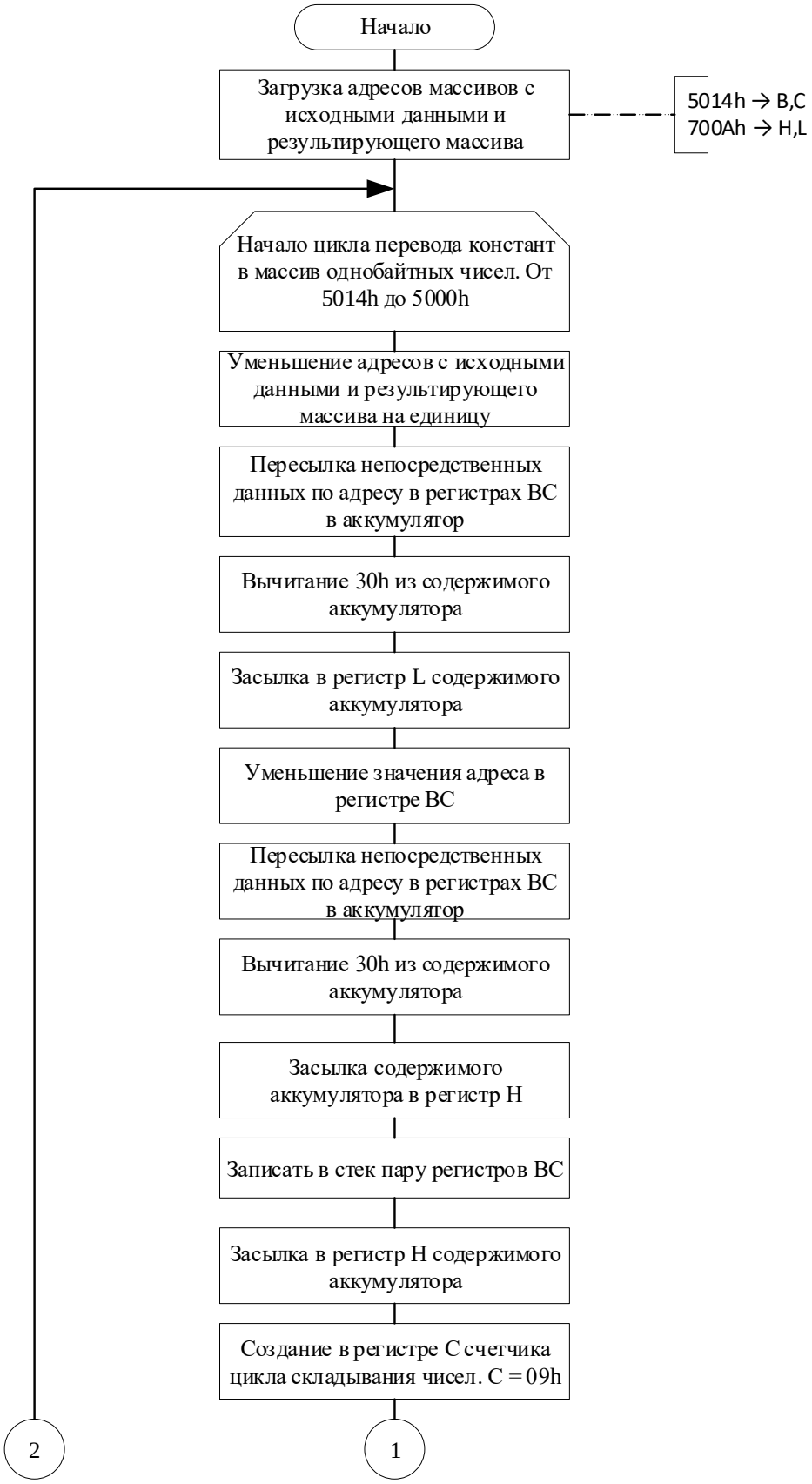

Рисунок 3, лист 1 — Блок-схема основного алгоритма программы

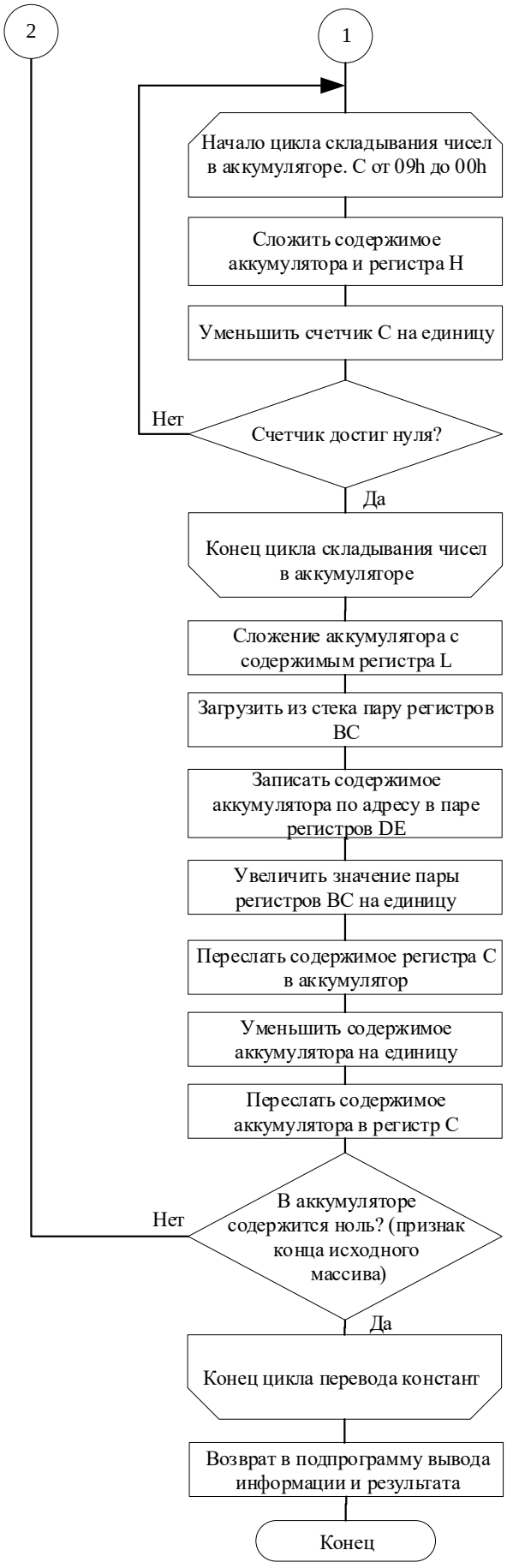

Рисунок 3, лист 2 — Блок-схема основного алгоритма программы

2.2 Распределение памяти и листинг программы с комментарием

В ходе разработки программы для выполнения индивидуального задания было необходимо решить ряд дополнительных задач, а именно:

- Организовать вывод приветствия и информации об авторе программы на экран ВТА;
- Обеспечить вывод на экран ВТА исходного массива и результата работы программы, а также сопроводительного текста.

Для решения данных задач была разработана подпрограмма расположенная в памяти начиная с адреса 3000h.

Таблица 4 — Листинг подпрограммы проекта, предназначенной для ввода текстовых констант

Адрес$_{16}$	**Код$_{16}$**	**Код ассемблера**	**Комментарий**
3000*	21 20 30	LXI H, 3020h	В пару регистров H,L загружается адрес памяти 3020h для работы процедуры ввода TTIO.
3003	CD 55 00	CALL 55h	Вызывается подпрограмма монитора TTIO – ввод символа в аккумулятор с эхом на консоль.
3006	FE 30	CPI 30h	Введенный по процедуре TTIO в аккумулятор символ, сравнивается с кодом 30h.
3008	CA 10 30	JZ 3010h	Переход, если флаг Z установлен, причем флаг нуля устанавливается после работы команды CPI только в том случае, когда с консоли процедурой TTIO будет введен символ нуля. В противном случае, когда перехода по адресу 3010h не произойдет, и будет выполняться команда, следующая за командой JZ.
300B	77	MOV M, A	Пересылка введенного TTIO символа по адресу, содержащемуся в регистрах H, L.
300C	23	INX H	Инкремент пары регистров H, L, т.е. увеличение значения адреса памяти на единицу.
300D	C3 03 30	JMP 3003h	Безусловный переход по адресу 3003H.
3010	AF	XRA A	Обнуление аккумулятора.
3011	77	MOV M, A	Пересылка нулевого содержимого аккумулятора в память по адресу, содержащемуся в регистра H, L, т.е. завершение текста нулевым (пустым) байтом. Это требование процедуры TTCLF для обнаружения конца выводимого текста.
3012*	01 20 30	LXI B, 3020h	Загрузка в пару регистров B, C начального адреса, по которому записан символьный текст для процедуры вывода TTCLF.

Продолжение таблицы 4

Адрес$_{16}$	**Код$_{16}$**	**Код ассемблера**	**Комментарий**
3015	CD 4F 00	CALL 4Fh	Вызов процедуры Монитора. Вывод строки текста с переходом на новую строку, начиная с адреса в паре B,C до нулевого байта. Пользователь проверяет правильность ввода.
3018	C3 40 00	JMP 40h	Выход в Монитор в режиме ожидания ввода команды Монитора.

Стоит отметить, что в проекте используется несколько текстовых констант, поэтому данная программа используется неоднократно. Поэтому в приведенной таблице команды по адресу 3000h и 3012h отмечены символом звездочки. Для повторного использования кода программы в ходе работы над проектом было необходимо изменить 2 и 3 байты команд для изменения стартовой ячейки ввода текстовой константы. Таким образом текст располагается в дампе памяти друг за другом, при этом различные между собой константы разделены нуль-терминатором.

Дамп памяти с 3020h до 30A5h занимает набор текстовых констант. Начиная с адреса 30A8h расположена программа вывода на экран текстовых констант, исходный данных, вызова основной функции и печати результата работы.

Таблица 5 — Листинг подпрограммы проекта для вывода информации и вызова основного алгоритма.

Адрес$_{16}$	**Код$_{16}$**	**Код ассемблера**	**Комментарий**
30A8	CD 49 00	CALL 49h	Вызов процедуры монитора. Переход на новую строку.
30AB	01 20 30	LXI B, 3020h	Загрузка в пару регистров B, C начального адреса, по которому записан символьный текст для процедуры вывода TTCON.
30AE	CD 4C 00	CALL 4Ch	Вызов процедуры монитора. вывод строки текста, начиная с адреса, записанного в B,C до нулевого байта
30B1	CD 49 00	CALL 49h	Вызов процедуры монитора. Переход на новую строку.
30B4	01 00 50	LXI B, 5000h	Загрузка в пару регистров B, C начального адреса, по которому записан исходный массив для процедуры вывода TTCON.
30B7	CD 4C 00	CALL 4Ch	Вызов процедуры монитора. вывод строки текста, начиная с адреса, записанного в B,C до нулевого байта
30BA	CD 00 40	CALL 4000h	Вызов основного алгоритма программы

Продолжение таблицы 5

Адрес$_{16}$	**Код$_{16}$**	**Код ассемблера**	**Комментарий**
30BD	CD 49 00	CALL 49h	Вызов процедуры монитора. Переход на новую строку.
30C0	01 88 30	LXI B, 3088h	Загрузка в пару регистров B, C начального адреса, по которому записан символьный текст для процедуры вывода TTCON.
30C3	CD 4C 00	CALL 4Ch	Вызов процедуры монитора. Вывод строки текста, начиная с адреса, записанного в B,C до нулевого байта
30C6*	1E 0A	MVI E, 0Ah	Загрузка регистра E числом 10. Организация счётчика для цикла
30C8	01 00 70	LXI B, 7000h	Засылка в пару B, C адреса 1-го числа.
30CB	0A	LDAX B	Загрузка аккумулятора содержимым ячейки памяти, адрес которой содержится в регистрах B,C. Является меткой **M1**.
30CC	CD 61 00	CALL 61h	Вызов процедуры монитора. Вывод двух шестнадцатеричных цифр из A (Аккумулятор)
30CF	03	INX B	Продвижение адреса BC + 1 → BC
30D0	1D	DCR E	Уменьшение содержимого счетчика (регистр E)
30D1	C2 CB 30	JXZ 30CBh	Переход к метке **M1** по флагу Z=0
30D4	76	HLT	Останов

В ходе работы над проектом требовалось предусмотреть и обработать особые ситуации. В числе таковых — вывод результатов работы программы. В силу того, что результат — это массив чисел в 16-ричной системе счисления, то его вывод на экран необходимо обеспечить в том же виде. Стандартные процедуры Монитора, используемые при выводе текстовых констант и исходного массива, не удовлетворяют требованиям по выводу в консоль для данной ситуации в силу того, что:

- Процедура TTCON воспринимает массив чисел как символы кодировки КОИ-7 и перед выводом перевод содержимое байта в соответствующий символ[2].
- Результирующий массив содержит число 00h, что для вышеуказанной процедуры является символом конца строки (нуль-терминатор) и, как следствие, вывод завершается до окончания элементов массива.

Для решения возникшей проблемы был организован цикл вывода массива поэлементно с адреса 30C6h. Для этого организовывается счетчик по количеству элементов массива, затем в регистры B, C записывается адрес начального элемента и непосредственные данные записываются в аккумулятор и далее при помощи процедуры Монитора OUTHX производится вывод элементов массива на экран. Цикл повторяется заново до тех пор, пока не будет достигнут конец массива, а именно, пока регистр E не будет равен 00h[1].

Таблица 6 — Листинг основного алгоритма программы

Адрес$_{16}$	**Код$_{16}$**	**Код ассемблера**	**Комментарий**
4000	01 14 50	LXI B, 5014h	Загрузка в пару BC непосредственные данные 5014 (следующий адрес после последнего байта массива шестнадцатиразрядных констант)
4003	11 0A 70	LXI D, 700Ah	Загрузка в пару DE непосредственные данные (следующий адрес после последнего байта результирующего массива двоичных чисел)
4006	1B	DCR D	Декремент DE (уменьшение адреса)
4007	0B	DCX B	Декремент BC (уменьшение адреса)
4008	0A	LDAX B	Загрузить младший байт константы из ячейки с адресом BC (из массива шестнадцатиразрядных констант) в аккумулятор
4009	DE 30	SBI 30h	A=A-30 (перевод в двоичное число)
400B	6F	MOV L, A	Переслать из A в L
400C	0B	DCX B	Декремент BC
400D	0A	LDAX B	Загрузить старший байт константы из ячейки с адресом BC (из массива шестнадцатиразрядных констант)
400E	DE 30	SBI 30h	A=A-30 (перевод в двоичное число)
4010	67	MOV H, A	Переслать из A в H
4011	C5	PUSH B	Записать в стек пару регистров BC
4012	7C	MOV A, H	Переслать из H в A
4013	0E 09	MVI C 09h	Записать в C данные 09h (организация счетчика не основного цикла)
4015	84	ADD H	Сложить A с H
4016	0D	DCR C	Декремент C (шаг счетчика на вычитание)
4017	C2 15 40	JNZ 4015h	Перейти по адресу 4015 если Z=0
401A	85	ADD L	A=A+L (получаем искомое двоичное число)
401B	C1	POP B	Загрузить из стека пару регистров BC
401C	12	STAX D	Записать A по адресу DE (запись результата)
401D	0C	INX C	Инкремент BC
401E	79	MOV A, C	Переслать из C в A
401F	3D	DCR A	Декрементировать A
4020	4F	MOV C, A	Переслать из A в C
4021	C2 06 40	JNZ 4006h	Перейти по адресу 4006 если Z=0 (цикл закончится, когда в A будет нули, иначе, когда все константы будут переведены в двоичные числа)
4024	C3 BD 30	JMP 30BDh	Возврат в подпрограмму для вывода результата

2.3 Результаты тестирования программы

При запуске программы, на экране Монитора выводятся следующие данные:

```
КУРСОВОЙ ПРОЕКТ ВЫПОЛНИЛ СТУДЕНТ 2-ГО КУРСА 494 ГРУППЫ ГУСЕВ АНТОН. ВАРИАНТ # 3
ИСХОДНЫЙ МАССИВ:
9738120045301499076 1
РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ: 61260C002D1E0E63073D_
```

Рисунок 4 — Результат запуска программы

По адресу 5000h лежит массив исходных данных:

```
?D5000/5013
5000: 39 37 33 38 31 32 30 30 34 35 33 30 31 34 39 39
5010: 30 37 36 31
```

Рисунок 5 — Исходные данные программы

По адресу 7000h лежит массив результатов:

```
?D7000/7009
7000: 61 26 0C 00 2D 1E 0E 63 07 3D
```

Рисунок 6 — Результат работы программы

Таким образом, исходному массиву соответствует:

Таблица 7 — Соответствие исходных данных и результатов

Константа	**Результат$_{16}$**
39 37	61
33 38	26
31 32	0C
30 30	00
34 35	2D
33 30	1E
31 34	0E
39 39	63
30 37	07
36 31	3D

Для того чтобы быстро проверить результат можно воспользоваться таблицей КОИ-7 и следующим алгоритмом:

Целочисленное деление на 16 несколько раз пока в частном не получим цифру меньшую 16, затем записываем эту цифру и приписываем все остатки целочисленного деления в обратном порядке с учетом значений чисел из таблицы соответствия чисел.

Например: 33 30 по таблице КОИ-7 — это число 30_{10}. 30 делим на 16 получаем в результате **1** и **14** в остатке. 1 уже не разделить на 16, т.к. она уже меньше 16. Число 14 в шестнадцатеричной системе счисления — это E_{16}. Таким образом получаем ответ 33 30 = 30_{10} = $1E_{16}$.

3 Описание средств вычислительной техники

При разработке программы был использован персональный компьютер следующими техническими характеристиками:

Процессор: Intel Core i5-7400 3.0GHz;

Оперативная память: DDR4 16Гб;

Видеокарта: Nvidia GeForce GTX1060 Dual, 6 GB видеопамяти;

Жесткий диск: 1Тб;

Клавиатура, компьютерная «мышь».

Стандартное программное обеспечение:

Microsoft Windows 10 Сборка 19041, Microsoft Office Word 2019

Практическая разработка данного курсового проекта выполнялась на эмуляторе микроЭВМ СМ-1800 v3.02.

Выводы

Курсовой проект выполнен полностью в соответствии с Заданием. В процессе работы было подробно изучено двоично-десятичное кодирование информации и арифметические операции с ДД-кодами, а также изучен способ представления информации в виде кодов КОИ-7. Получен навык структурирования и написания программ на языке ассемблера, изучены способы взаимодействия с памятью и процесс работы машины при выполнении программы.

При выполнении курсового проекта возникал ряд особых ситуаций, описанных в части практической разработки программ. Для решения данных ситуаций требовалось повторно возвращаться к теоретической части для проработки подхода и алгоритма решения поставленной задачи.

Стоит отметить, что программа может усовершенствована путём добавления возможности ввода массива исходных данных вручную с клавиатуры, при этом явно указывая на размер вводимого массива.

Список литературы

1. Песков И.А., Макарук Р.В. Разработка программ в машинных кодах и на языке ассемблер : практикум для студентов очной формы обучения. – СПб.: СПбГТИ(ТУ), 2019. – 76с.

2. Стандартное программное обеспечение. Монитор: Методические указания.— СПб.: СПбГТИ(ТУ), 2006. – 23с.

3. Хорошевский, В. Г. Архитектура вычислительных систем : учебное пособие / В.Г. Хорошевский. – 2— е изд. – М.: Изд— во МГТУ им. Баумана, 2008. – 520с.

4. Горбоненко В.Д. Арифметические основы цифровой техники / В.Д. Горбоненко – Ульяновск: УлГТУ, 2007. – 27с.