# Национальный исследовательский университет «МЭИ» Институт Радиотехники и электроники им. В.А. Котельникова Кафедра Формирования и обработки радиосигналов

#### ОТЧЕТ

# по лабораторной работе №1 «ОСНОВНЫЕ ЭТАПЫ ПРОЕКТИРОВАНИЯ ПРОГРАММ ДЛЯ МИКРОКОНТРОЛЛЕРОВ»

по дисциплине «Цифровая и микропроцессорная техника»

Группа: ЭР-11-21

Бригада №: 2

Выполнил: Тимохин С.А. Проверил: Соловьев Е.Д.

Дата: 13.10.2024

Оценка:

#### 1. ЦЕЛЬ РАБОТЫ

- 1) Знакомство с пакетом MPLAB IDE X;
- 2) Знакомство с архитектурой 8-разрядных микроконтроллеров (МК) на примере МК PIC18F2520;
- 3) Знакомство с этапами проектирования программы для МК на языке «Ассемблер».

#### 2. ДОМАШНЯЯ ПОДГОТОВКА

# 2.1. Выполнение арифметической операции в двоичной системе счисления

$$\begin{split} &\frac{[75+(-132)]}{2} = -28.5\\ &75_{10} = 0000\ 0000\ 0100\ 1011_2\ (\Pi \mathsf{K} = \mathsf{Д}\mathsf{K})\\ &132_{10} = 0000\ 0000\ 1000\ 0100_2\ (\Pi \mathsf{K})\\ &-132_{10} = 1111\ 1111\ 0111\ 1100_2\ (\mathsf{Д}\mathsf{K})\\ &1111\ 1111\ 0111\ 1100_2\ (\mathsf{Д}\mathsf{K}) + 0000\ 0000\ 0100\ 1011_2\ (\mathsf{Д}\mathsf{K}) = result_2(\mathsf{Д}\mathsf{K})\\ &result_2(\mathsf{Д}\mathsf{K}) = 1111\ 1111\ 1100\ 0111_2(\mathsf{Д}\mathsf{K}) = -57_{10} \end{split}$$

Для деления на два необходимо сдвинуть число в двоичном виде вправо на 1 разряд. Старший разряд дополним единицей для сохранения знака. Младший разряд при сдвиге отбрасывается.

$$1111\ 1111\ 1100\ 0111_2 \gg 1 =\ 1111\ 1111\ 1110\ 0011_2 = -29_{10}$$

Погрешность вычисления составляет 0.5

#### 3. ХОД РАБОТЫ

# 3.1 Составление программы, реализующей выполнение арифметического примера

; Выделяем 2 байта в банке памяти под переменные

summand\_1 RES 2

summand\_2 RES 2

summa RES 2

final result RES 2

**MOVLW b'01001011'** ; запись числа 75 в аккумулятор

**MOVWF summand\_1**; запись числа 75 из аккумулятора в переменную

**MOVLW b'101111100'**; запись числа -132 в аккумулятор

**MOVWF summand\_2**; запись числа -132 из аккумулятора в переменную

**MOVF summand\_1, W**; копируем числа из регистра summand\_l в аккум.

ADDWF summand\_2, W ; складываем число, которое находится в данный

; момент в аккумуляторе с числом, которое хранится в регистре summand\_2.

; Записываем результат в аккумулятор.

**MOVWF summa** ; результат суммы копируем из аккумулятора в переменную ; summa.

BSF STATUS,С; сброс флага С

**RRCF summa,F**; сдвиг вправо на один бит (деление на 2)

**DECF summa**; декремент числа в регистре summa

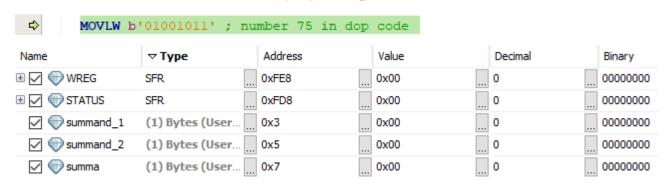
**COMF summa**; инверсия числа в регистре summa

MOVFF summa,final\_result; запись результата выражения в регистр

final\_result

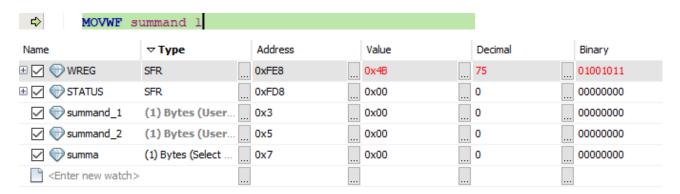
### 3.2 Отладка программы

Шаг 1: Запись числа 75 в аккумулятор



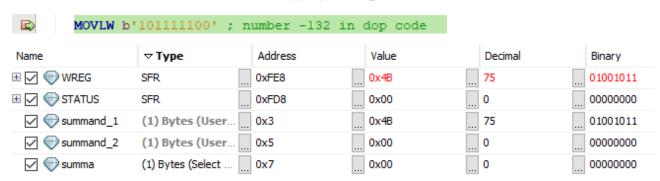
На первом шаге процессор получает инструкцию, но еще ничего не успел записать в аккумулятор. На следующем шаге в аккумуляторе появится число 75.

Шаг 2: Запись числа 75 из аккумулятора в переменную summand\_1



В аккумулятор записалось число 75, и я даю процессору инструкцию записать это число в переменную summand\_1.

Шаг 3: Запись числа -132 в аккумулятор



Видно, что в переменную summand\_1 успешно было записано число 75. Теперь я даю процессору инструкцию записать в аккумулятор число -132.

Шаг 4: запись числа -132 из аккумулятора в переменную

MOVWF s	ummand_2				
Name	<b>▽ Type</b>	Address	Value	Decimal	Binary
	SFR	0xFE8	0x7C	124	01111100
⊕ STATUS	SFR	0xFD8	0x00	0	00000000
✓ ⊕ summand_1	(1) Bytes (User	0x3	0x4B	75	01001011
✓ summand_2	(1) Bytes (User	0x5	0x00	0	00000000
summa 🗸	(1) Bytes (Select	0x7	0x00	0	00000000
Enter new watch:	>	] [			

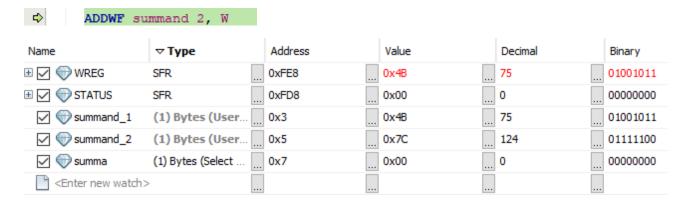
В аккумулятор было записано число -132 (отображается как 124). Даём процессору инструкцию записать число из аккумулятора в регистр summand\_2.

Шаг 5: копируем число из регистра summand 1 в аккумулятор

MOVF s	ummand_1, W				
Name	<b>▽ Type</b>	Address	Value	Decimal	Binary
	SFR	0xFE8	0x7C	124	01111100
	SFR	0xFD8	0x00	0	00000000
✓       summand_1	(1) Bytes (User	0x3	0x4B	75	01001011
✓   Summand_2	(1) Bytes (User	0x5	0x7C	124	01111100
summa 🗸	(1) Bytes (Select	0x7	0x00		00000000
<enter new="" p="" watch:<=""></enter>	>				

В summand\_2 было успешно записано число 124 (-132). Даём процессору инструкцию скопировать число из регистра summand\_1 в аккумулятор.

Шаг 6: сложение чисел в аккумуляторе и регистре summand\_2



В аккумулятор было успешно записано число 75. Даём процессору инструкцию сложить число, которое хранится в аккумуляторе с числом, которое находится в регистре summand\_2. Результат записываем в аккумулятор.

Шаг 7: копирование числа из аккумулятора в регистр summa.

MOVWF :	summa				
Name	<b>∀Туре</b>	Address	Value	Decimal	Binary
	SFR	0xFE8	0xC7	199	11000111
	SFR	0xFD8	0x1A	26	00011010
✓       summand_1	(1) Bytes (User	0x3	0x4B	75	01001011
summand_2	(1) Bytes (User	0x5	0x7C	124	01111100
✓   Summa	(1) Bytes (User	0x7	0x00	0	00000000

Операнды были сложены и результат записан в WREG. В флаг переноса С, было записано значение 26. Даю процессору инструкцию скопировать число из аккумулятора в регистр summa.

Шаг 8: сброс флага переноса

⇔ BSF ST	ATUS,C ; reset :	Elag C			
Name	<b>∀Туре</b>	Address	Value	Decimal	Binary
	SFR	0xFE8	0xC7	199	11000111
	SFR	0xFD8	0x1A	26	00011010
✓ ⊕ summand_1	(1) Bytes (User	0x3	0x4B	75	01001011
✓ ⊕ summand_2	(1) Bytes (User	0x5	0x7C	124	01111100
summa 🗸	(1) Bytes (User	0x7	0xC7	199	11000111
✓	(1) Bytes (Select	0x9	0x00	0	00000000

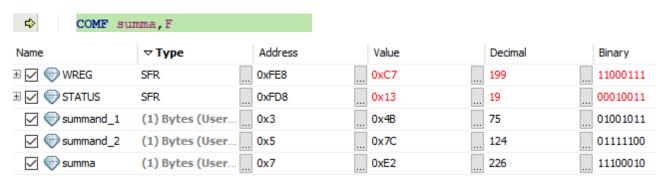
Шаг 9: Сдвиг вправо регистра summa на 1 бит

RRCF su	mma, F				
Name	<b>▽ Type</b>	Address	Value	Decimal	Binary
	SFR	0xFE8	0xC7	199	11000111
	SFR	0xFD8	0x1B	27	00011011
summand_1	(1) Bytes (User	0x3	0x4B	75	01001011
summand_2	(1) Bytes (User	0x5	0x7C	124	01111100
summa 🗸 🤍	(1) Bytes (User	0x7	0xC7	199	11000111

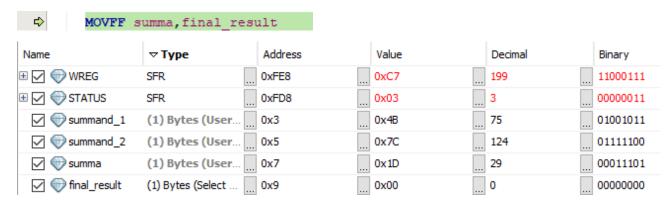
Шаг 10: Декремент числа в регистре summa



Шаг 11: Инверсия числа в регистре summa



Шаг 12: Запись результата в регистр final\_result



Шаг 13: END



# 3.3 Модификация программы

MOVLW d'75'
MOVWF summand\_1
MOVLW d'132'
MOVWF summand\_2

; перевод числа 132 в дополнительный код

**COMF summand\_2,F** 

**INCF** summand\_2,F

**MOVF summand\_1, W**; копируем число из регистра summand\_1 в аккум. **ADDWF summand\_2, W**; складываем число, которое находится в данный ; момент в аккумуляторе с числом, которое хранится в регистре summand\_2. ; Записываем результат в аккумулятор.

**MOVWF summa** ; результат суммы копируем из аккумулятора в переменную ; summa.

 ${f BSF}$  STATUS,C ; сброс флага C

**RRCF summa,F**; сдвиг вправо на один бит (деление на 2)

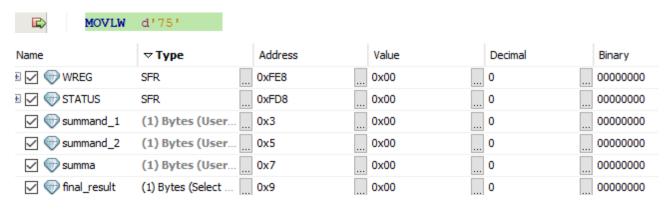
**DECF summa**; декремент числа в регистре summa

COMF summa; инверсия числа в регистре summa

MOVFF summa, final\_result; запись результата выражения в регистр final\_result

### 3.4 Отладка модифицированной программы

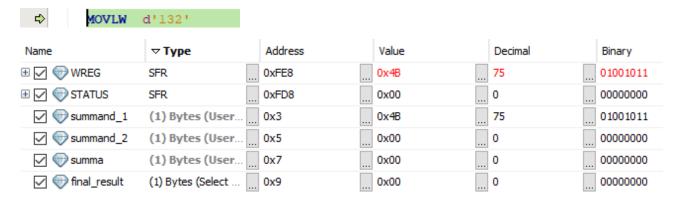
#### Шаг 1:



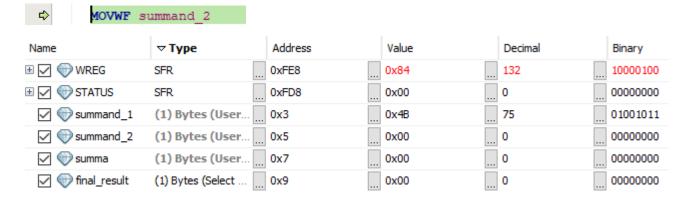
#### Шаг 2:



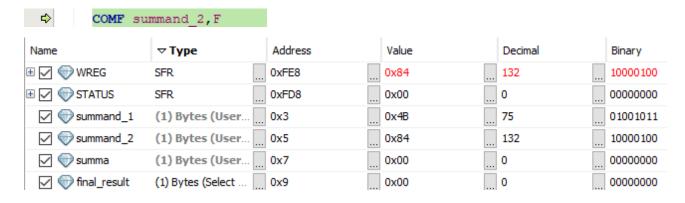
#### Шаг 3:



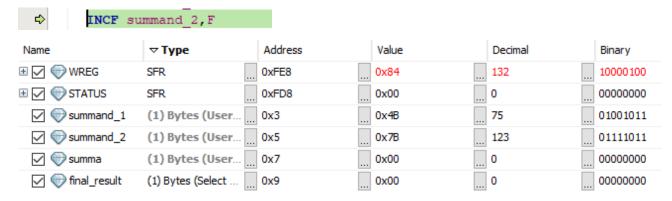
#### Шаг 4:



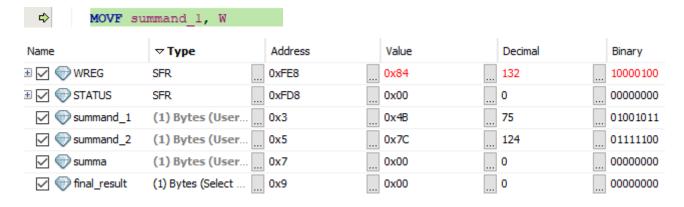
#### Шаг 5:



#### Шаг 6:



#### Шаг 7:



Далее анализ программы имеет результаты аналогичные отладке программы из пункта 3.2, т.к. последующие инструкции программ идентичные.



# 3.5 Пример трансляции команд MOVLW и MOVWF

Команда: MOVLW b'01001011'

# Внутренний формат команды:

Код операции: 0000 1110

Операнд: 0100 1011

Запись в шестнадцатеричном виде: 0Е4В

Команда: MOVWF summand\_1

# Внутренний формат команды:

Код операции: 1011 1110

Операнд: 0x3 (0000 0011) (Адрес регистра summand\_1)

Запись в шестнадцатеричном виде: 6Е03

#### 4. ВЫВОД

Спроектированная программа корректно выполняет арифметическую операцию. В ходе выполнения лабораторной работы были изучены основные ассемблерные инструкции микроконтроллера PIC18F2520 и его архитектура. Я ознакомился с программой MPLAB X IDE и закрепил свои знания о работе с бинарными числами и их представлении в ЭВМ.

Регистр флагов STATUS — это один из основных регистров управления МК, который используется для управления ходом выполнения программы, мониторинга результатов арифметических, логических и иных операций.

Можно обратить внимание, что до какого-то момента выполнения программы регистра флагов STATUS имеет значение 0. После выполнения строки кода **ADDWF summand\_2, W** в регистре **STATUS** появляется число **26** (**0001 1010**). Каждый бит данного записанного числа в регистр STATUS отображает признаки (флаги) выполненной операции.

Номер бита	Флаг	Назначение
7-5	-	Не используются
4	N	Признак знака ( $N=0$ — результат положительный, $N=1$ — результат отрицательный)
3	OV	Переполнение и выход за разрядную сетку
2	Z	Признак нулевого результата
1	DC	Десятичный перенос / коррекция
0	С	Признак переноса (старший доп. Разряд числа)

Таким образом, флаг состояния 26 говорит о том, что результат ненулевой, произошло переполнение, результат отрицательный.

Таким образом, регистр флагов меняет своё значение несколько раз в зависимости от результатов предыдущей операции.