Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет ИУ «Информатика и системы управления»

Кафедра ИУ-3 «Информационные системы и телекоммуникации»

Методические указания
по выполнению домашнего задания 2
«Библиотечная подпрограмма»
по дисциплине «Микропроцессорные устройства обработки сигналов»

Для студентов, обучающихся по направлениям
2304002468, 2304007468 и 2302010065

Составитель проф. каф. ИУЗ, д.т.н. В.С. Выхованец

Москва, 2020

Содержание

1 Общие указания

Целью выполнения домашнего задания является изучение библиотечной подпрограммы цифровой обработки сигналов, заданной в теме домашнего задания.

Для выполнения домашнего задания предварительно необходимо ознакомиться с учебным пособием [1, с. 102-165] и технической документацией [2], найти и изучить описание заданной библиотечной подпрограммы.

На следующем этапе выполнения домашнего задания необходимо в литературе, например [3], найти и привести в отчете те теоретические сведения, которые необходимы для понимания метода решения задачи индивидуального задания. Далее в отчете описывается алгоритм, реализуемый библиотечной функцией.

На последнем этапе выполнения домашнего задания необходимо изучить исходный текст заданной библиотечной функции, прилагаемый к описанию библиотеки цифровой обработки сигналов. Результатом изучения исходного текста должны быть комментарии к операторам программы, содержащие ссылки на теоретический материал и указания на используемые технические средства микропроцессора. Прокомментированный текст библиотечной функции приводится в приложении к отчету по домашнему заданию.

Итогом выполнения домашнего задания является оформление отчета. Срок выполнения домашнего задания – три недели с момента выдачи темы индивидуального задания.

2 Требования к отчету

Оформление отчета по домашнему заданию осуществляется в соответствии с ГОСТ 2.105-95 [3], список литературы по ГОСТ Р 7.0.5–2008 [4]. Отчет должен содержать:

- титульный лист;
- содержание;
- постановка задачи;
- теоретические сведения;
- интерфейс библиотечной функции;
- описание алгоритма;
- выводы;
- список литературы;
- приложение с текстом библиотечной функции.

3 Пример выполнения домашнего задания

3.1 Постановка задачи

Темой домашнего задания является изучение алгоритмов и исследование способов их реализации, выполняющих канальное шифрование сигналов в цифровой форме. Для выполнения домашнего задания необходимо:

- ознакомиться с рекомендованной литературой;
- привести в отчете теоретические сведения по методам обработки сигналов;
- описать алгоритм обработки сигналов и данных;
- прокомментировать текст библиотечной функции;
- оформить отчет по домашнему заданию.

Основным результатом домашнего задания является комментированный текст библиотечной функции на языке ассемблера, а также описание выполняемого ею алгоритма цифровой обработки сигналов и данных.

3.2 Теоретические сведения

Канальное шифрование – способ шифрования, при котором шифруются данные, проходящие через канал связи, а также данные о маршрутизации сообщений и форматах их представления. Канальное шифрование применяется при передаче конфиденциальных данных по незащищенным каналам связи. При шифровании выполняется обратимое преобразование данных с целью их сокрытия от неавторизованных получателей и, в это же время, предоставление авторизованным получателям возможности доступа к этим данным в незашифрованном виде.

Шифром называется пара алгоритмов, реализующих прямое и обратное преобразование сообщений. Эти алгоритмы применяются над данными с использованием ключа, который задает выбор конкретного преобразования из совокупности возможных преобразований данных, реализуемых заданным алгоритмом.

Ключи для шифрования и для расшифровывания могут отличаться, а могут быть одинаковыми. В симметричных криптосистемах для шифрования и расшифрования используется один и тот же ключ. Алгоритм и ключ выбирается заранее и известен обеим сторонам.

Основной операцией симметричного шифрования является гаммирование, или поразрядное сложение по модулю два криптографической гаммы с блоком исходных данных, которое используется для получения шифротекста. Криптографическая гамма вычисляется по некоторому известному алгоритму на основе ключа (см. ГОСТ 28147-89).

3.3 Интерфейс библиотечной функции

Для выполнения операции гаммирования используется функция gamma, имеющая следующий прототип на языке С:

```
void gamma(DATA *x, DATA *y, DATA *r, ushort nx),
```

где `ushort` – тип данных, определенный как `unsigned short`; `DATA` – тип данных, являющийся синонимом `int`; `x` – указатель на вектор исходных данных; `y` – указатель на вектор криптографической гаммы, `r` – указатель на вектор, выделенный вызывающей функцией для размещения результатов гаммирования; `nx` – число элементов в векторах `x`, `y` и `z`. Возвращаемое значение у функции отсутствует.

Векторы входных и выходных данных функции представляют собой одномерные массивы `x`, `y` и `z`, элементами которых являются 16-разрядные числа (целые или дробные), пронумерованные в порядке увеличения их адресов в памяти данных (рисунок 3.1).

Адрес	
x+0	x[0]
x+1	x[1]
x+2	x[2]
x+3	x[3]
...	...
x+nx-1	x[nx-1]

Рисунок 3.1 – Вектор `x` функции gamma

Длина векторов `nx` задается в формате N16 – беззнаковом 16-разрядном целочисленном формате. Сами массивы, а также их элементы выравнены по границе слова. Разрешается использование одного и того же указателя для задания любых двух векторов. Использование одного и того же указателя для всех трех векторов приведет к обнулению заданного вектора.

Перекрытие массивов не допускается, так как результат вызова функции становится не соответствующий ее спецификации.

3.4 Описание алгоритма

Функция gamma выполняет гаммирование двух входных векторов `x` и `y`, а результат записывает в выходной вектор `r`. Реализуемый функций gamma алгоритм описан на языке программирования С и приведен на листинге 3.1.

Листинг 3.1 – Алгоритм гаммирования

```
#define  DATA     int
#define  ushort   unsigned short
void gamma(DATA *x, DATA *y, DATA *r, ushort nx)
{
```

Листинг 3.1 – Алгоритм гаммирования

```
    ushort i;
    for(i = 0; i < nx; i++)
         r[i] = x[i] ^ y[i];
}
```

В теле основного цикла функции выполняется извлечение очередных элементов векторов `x` и `y`, их поразрядное сложение по модулю 2 и запись результата в соответствующий элемент выходного вектора. Для организации цикла используется целочисленная переменная `i`, которая задает индекс элементов векторов.

Комментированный текст библиотечной функции gamma приведен в приложении А.

3.5 Выводы

В домашнем задании 2 приведено описание библиотечной функции гаммирования, используемой для шифрования данных в открытых каналах связи. Для метода шифрования данных на основе криптографического гаммирования разработан алгоритм на языке С и приведен комментированный текст его реализации библиотечной функцией на языке ассемблера для микропроцессора цифровой обработки сигналов TMS320C5515.

Список литературы

[1] Микропроцессорные устройства обработки сигналов [Текст] / В.С. Выхованец, Н.А. Демин, Е.И. Мозговая, С.И. Назарова, Д.А. Рожкова, Е.С. Шапкина; Под ред. В.С. Выхованца. – М.: МГТУ им. Н.Э. Баумана, 2014. – 177 с.

[2] TMS320C55x DSP Library: Programmer's Reference. – Texas Instruments, 2009. – 144 p.

[3] ГОСТ 2.105-95. Единая система конструкторской документации. Общие требования к текстовым документам. – М.: Изд-во стандартов, 2012. – 26 с.

[4] ГОСТ Р 7.0.5–2008. Библиографическая ссылка. Общие требования и правила составления. – М.: Изд-во стандартов, 2009. – 23 с.

Приложение А

Библиотечная функция gamma

```
;*****************************************************************
; Версия XX.YY.ZZ
;*****************************************************************
; Функция gamma.
; Микропроцессор C55xx.
; Описание: криптографическое гаммирование данных.
; Алгоритм:
;   for(i = 0; i < nx; i++)
;      r[i] = x[i] ^ y[i];
; Использование:
;   gamma (
;        DATA *x,                  /* AR0 – адрес первого вектора */
;        DATA *y,                  /* AR1 – адрес второго вектора */
;        DATA *r,                  /* AR3 – адрес результата*/
;        ushort nx                 /* T0 – длина векторов */
;   );
;*****************************************************************
        .ARMS_off                       ;Сигнальный режим адресации
        .CPL_on                         ;Режим компилятора
        .mmregs                         ;Разрешение MMR-регистров
;-----------------------------------------------------------------
; Описание структуры стека данных
;-----------------------------------------------------------------
LOC_SZ  .set    1                       ;Объем локальных переменных
SAV_SZ  .set    1                       ;Объем сохраняемых регистров
RET_SZ  .set    1                       ;Размер адреса возврата
ARGS    .set    LOC_SZ+SAV_SZ+RET_SZ    ;Смещение аргументов
ARG_SZ  .set    0                       ;Объем аргументов
;-----------------------------------------------------------------
; Макроопределения для используемых регистров
;-----------------------------------------------------------------
        .asg    AR0, x_ptr              ;Первый вектор
        .asg    AR1, y_ptr              ;Второй вектор
        .asg    AR2, r_ptr              ;Результирующий вектор
        .asg    BRC0, outer_cnt         ;Счетчик наружнего цикла
;-----------------------------------------------------------------
; Тело функции
;-----------------------------------------------------------------
        .def    _gamma
        .text
_gamma:
; Сохранение регистров статуса в стеке
        PSH     mmap(ST2_55)
        PSH     mmap(ST1_55)
; Выделение в стеке локальной памяти
        ASUB    #LOC_SZ, SP
; Сохранение в локальной памяти регистров
        MOV     outer_cnt, @0
; Конфигурирование микропроцессора
        OR      #04020h, mmap(ST1_55) ;Установка CPL и C54CM
        AND     #07F00h, mmap(ST2_55) ;Сброс ARMS и ARxLC
; Установвить счетчик повторения наружного блока
        SUB     #1, T0                ; T0 = nx-1
        MOV     T0, outer_cnt         ; BRC0 = nx-1
; Начало наружного блока команд
```

```
        RPTBLOCAL loop                ;Команда повторения блока
        MOV     *x_ptr+, AC0          ;Чтение элемента x(i)
        XOR     *y_ptr+, AC0          ;Гаммирование с y(i)
loop:   MOV     LO(AC0), *r_ptr+      ;Запись результата
; Восстановить регистры из локальной памяти
        MOV     @0, outer_cnt
; Освободить локальную память
;       AADD    #LOC_SZ, SP
; Восстановление регистров статуса из стека
        POP     mmap(ST1_55)
        POP     mmap(ST2_55)
; Возврат из функции
        RET
;-----------------------------------------------------------------------
```