СЕКЦИЯ

УСТРОЙСТВА И СИСТЕМЫ ПАМЯТИ

Руководитель – к.т.н., с. н.с. **ЦЫРКИН В.В.**

ВОПРОСЫ ИСПОЛЬЗОВАНИЯ ПАМЯТИ ПРИ РЕЛИЗАЦИИ АЛГОРИТМА ЦИФРОВОЙ ОБРАБОТКИ РАДИОЛОКАЦИОННОЙ ИНФОРМАЦИИ

Сергеев В.В, Серов Е.М, Цыркин В.В.

ОАО «Концерн радиостроения «Вега»

Современный уровень цифровой техники позволяет практически полностью оптимизировать характеристики обработки сигналов и одновременно кардинально уменьшить массу и габариты аппаратуры, изъять из РЛС громоздкую аппаратуру аналоговой обработки сигналов.

Факторами, ограничивающими применение цифровой обработки сигналов РЛС (ЦОС), являются высокие требования к быстродействию, динамическому диапазону (разрядности) и точностным характеристикам аналого-цифровых преобразователей (АЦП).

Исходя из результатов анализа характеристик радиолокационных сигналов в станциях дозора в (1) был предложен рассматриваемый вариант построения ЦОС. Основные принципы и взаимосвязи аппаратуры ЦОС приведены в (2).

Структурная схема цифровой обработки радиолокационной информации приведена на рис. 1.

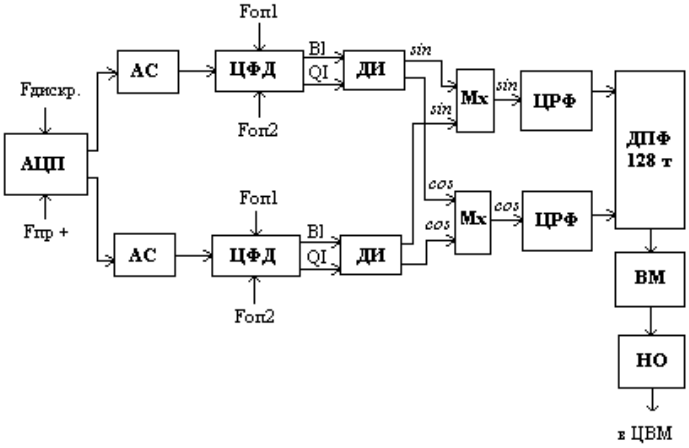


Рис. 1 Структурная схема цифровой обработки радиолокационной информации

АЦП – аналого-цифровой преобразователь
АС - аподизация в стробе
ЦФД - цифровой фазовый детектор
ДИ -децимация и интегрирование
ЦРФ – цифровой режекторный фильтр
ДПФ 128т – дискретный преобразователь Фурье (128 точек)
ВМ - вычислитель модуля
НО – некогерентная обработка
Мх – мультиплексор 2 →1
Данная структурная схема реализована на быстродействующем 14-ти разрядном АЦП AD9244 ф. Analog Devices, США и программируемых логических интегральных схемах (ПЛИС)EP1K100 ф. Altera, США

Аналого-цифровой преобразователь

Главной целью преобразования сигнала в цифровую форму непосредственно на промежуточной частоте является возможность последующего выполнения цифрового разложения сигнала на две квадратурные составляющие. При этом обеспечивается полная идентичность квадратурных каналов по коэффициенту передачи и точности соблюдения фазового сдвига в $\pi/2$ между квадратурами, практически недостижимые при аналоговом исполнении на двух фазовых детекторах. Ключевыми вопросами при аналого-цифровом преобразовании (АЦП) являются допустимые ограничения разрядной сетки, величина эквивалентных входных шумов устройства выборки-хранения (УВХ) и значение частоты дискретизации.

Разрядная сетка АЦП должна обеспечивать требование линейности преобразования в динамическим диапазоне входных сигналов, достигающим 90 дБ над средним значением внутренних шумов приемного тракта. Кроме того, цена младшего разряда и величина собственных шумов АЦП должны обеспечивать снижение суммарного значения вносимых шумов (дискретизации и УВХ), после согласованной фильтрации с выхода фазового детектора, до уровня, не превышающего минус 10 дБ относительно шумов приемного тракта.

Величина ослабления вносимых АЦП шумов после усредняющих цифровых фильтров в квадратурах сигнала зависит от соотношения частоты дискретизации и эквивалентной полосы частот фильтра по закону квадратного корня. При использовании линейного накопителя выборок в качестве усредняющего фильтра величина ослабления вносимых АЦП шумов будет равна квадратному корню из числа эффективно накапливаемых выборок. При использовании накопителя с взвешиванием амплитуд выборок в пределах накапливаемой пачки (аподизации) по закону, близкому к Sin^2, число эффективно накапливаемых выборок составит примерно половину от их полного числа.

Процедура накопления приводит к росту разрядной сетки и более подробному описанию квадратур сигнала, что эквивалентно случаю преобразования после некоего «идеального аналогового фазового детектора» с помощью АЦП с увеличенным числом разрядов (в сторону младших разрядов с уменьшением их цены). Эквивалентное, с точки зрения подробности описания входного сигнала, нарастание диапазона линейного преобразования также равно квадратному корню из числа эффективно накапливаемых выборок.

Аподизация в стробе

Входные сигналы представляют собой периодическую последовательность коротких импульсов с заполнением на промежуточной частоте, приходящих на обработку с произвольной задержкой во времени. В этой связи обработка должна происходить независимо по квантам времени, согласованным с длительностью импульса. Для снижения энергетических потерь сигнала при произвольном положении входного импульса относительно кванта обработки формируется две последовательности квантов, сдвинутых на половину их ширины.

Разбиение на короткие временные кванты эквивалентно нелинейной процедуре умножения входного сигнала на прямоугольную огибающую (стробированию), коэффициенты умножения хранятся в ПЗУ информационной емкостью 12 Кбит , которое реализовано во внутренней памяти ПЛИС. Если частота заполнения импульса не кратна частоте их повторения, то, при не очень высокой промежуточной частоте относительно длительности импульса, помимо истинной частоты сигнала, в окне между гармониками частоты повторения появится так называемая «зеркальная» составляющая.

Цифровой фазовый детектор

Для подавления каждой sin-составляющей и cos-составляющей в спектре, полученных в результате преобразования входного сигнала, и получения частоты выходных выборок, соответствующих одному элементу дальности используется разбиение последовательности выборок на пачки по 120 выборок, аподизация выборок в пачке по закону :

$W(k)=\sin^2(k*\pi/120)$, где W(k) - функция аподизации, k - номер выборки в пачке , k = 0 … 119;

входные выборки с частотой дискретизации (разрядность – 14р, включая знак);

Выходом фазового детектора являются sin-составляющая сигнала и cos-составляющая сигнала.

Выходы обоих дециматоров объединяются на мультиплексорах 2 →1.

Цифровой режекторный фильтр

Для ослабления влияния мощных помех должен быть реализован режекторный фильтр, коэффициенты которого выбираются по команде оператора.

Рекурсивный цифровой режекторный фильтр 4-го порядка построен на основе каскадного соединения двух фильтров 2-го порядка , выполненных по формуле:

Y[i] = B0*X[i]+B1*X[i-1]+B2*X[i-2]-A1*Y[i-1]-A2*Y[i-2];

Каждый каскад имеет свой набор коэффициентов: B0,B1,B2,A1,A2, который реализован на внутреннем ПЗУ ПЛИС. Поскольку фильтрация осуществляется по одноименным номерам выборок в разных зондированиях, выборки запоминаются во внутреннем ОЗУ ПЛИС информационной емкостью 20 Кбит и сопровождаются кодом частоты повторения КFп.

Начальные условия на работу фильтра не накладываются, так как исключение переходного процесса фильтра из дальнейшей обработки обеспечивается временной работой вычислителя БПФ. Каждая квадратурная составляющая фильтруется независимо, но с идентичными коэффициентами.

После процедуры фильтрации информация поступает в формате реального времени (последовательная упаковка) :

0j[0i....(D-1)i] .... (N-1)j[0i ... (D-1)i]

где j - номер зондирования, i - номер дальности

Для дальнейшей обработки информацию нужно перестроить в следующем виде :

0i[0j ... (N-1)j] ... (D-1)i[0j ... (N-1)j]

Перестройка информации осуществляется во внешнем ОЗУ информационной емкостью 128 К слов x 16 р , выполненном на двух микросхемах IDT71024S – 15 фирмы IDT, США с параметрами: информационная емкость 128 К x 8 р, быстродействие – 15 нс.

Дискретное преобразование Фурье

Дискретное преобразование Фурье осуществляется два раза по одной неоднозначной дальности, чтобы в дальнейшем в обнаружителе выполнить критерий два из двух. Производится вычисление 128-точечного БПФ. Для улучшения качества спектрального анализа перед вычислением БПФ необходимо иметь возможность наложения на временные выборки весовых функций.

A[i]=Fc[i] * Fa[i] , где Fc[i] - выборки сигнала,

Fa[i] - выборки функции аподизации,

где i - номер выборки от 0 до 127.

Функция аподизации : $Fa2 = \sin^2(\pi * i / 128)$ для $127 \geq i \geq 0$

Для хранения промежуточных данных каждой итерации используется двух портовое ОЗУ информационной емкостью 40 Кбит, реализованное в ПЛИС.

Вычисление модуля и формирование выходного массива

На выходе ДПФ квадратурные составляющие комплексных амплитуд следуют по частотам в естественной последовательности: 0 , ... , (N-1) , где N - размер преобразования. Вычисление модуля ведется по приближенной формуле:

M = 2Re + Im , если Re ≥ Im и

M = 2Im + Re , если Im > Re

Для работы обнаружителя выходной массив должен быть сформирован в следующей последовательности:

0f[0i ... (D-1)i] ... (N-1)f[0i ... (D-1)i],

где f - частотный канал ,

i -канал дальности ,

N -размер преобразования .

Формирование выходного массива с требуемой информационной емкостью 128 Кслов x 8 р осуществляется в ОЗУ, выполненном также на микросхеме IDT71024S – 15 фирмы IDT, США

Обнаружители.

Узкополосный обнаружитель (УО) состоит из вычислителя порога П(f) и схемы сравнения с ним Порог вычисляется для каждого частотного канала на основе i выборок данного канала . Пороги соседних частотных каналов не коррелированы.

Исполнительный порог вычисляется по формуле:

$$П(f) = [(\sum_{i=0}^{D-3} A(f,i))/(D-3)] * Кп ,$$

где f - номер текущей составляющей БПФ [0….(N-1)] , N=128,

i - номер текущей дальности [0….(D-3)] , D = 64

Кп- коэффициент порога.

Факт превышения амплитуды сигнала A(f,i) исполнительного порога П(f) происходит при выполнении условия: A(f,i)>П(f) [1]

При обнаружении отметки в одном из каналов пеленгационной пары ПП1 или ПП2 для дальнейшей обработки берется также одноименная ей по скорости (V) и дальности (D) парная отметка из другого канала и формируется двух лучевая отметка с амплитудами A1(i) и A2(i). Для запоминания скорости по каждой дальности используется двух портовое ОЗУ информационной емкостью 12 Кбит, реализованное в ПЛИС.

Широкополосный обнаружитель (ШО) состоит из двух некогерентных накопителей, реализованных на ОЗУ ПЛИС, осуществляющих в каждом канале дальности суммирование амплитуд в двух полосах частот (f = a - b, f = c - d, где f -номер текущей составляющей БПФ) с последующим усреднением, вычислителей порогов и схем сравнения с ними. Пороги для них определяются по формулам:

$$П1=[(\sum_{i=0}^{D-3} Acp_1(i))/(D-3)]*Кп_1, \quad \text{где } Acp_1(i)=(\sum_{f=a}^{b} A(f,i))/21;$$

$$П2=[(\sum_{i=0}^{D-3} Acp_2(i))/(D-3)]*Кп_1, \quad \text{где } Acp_2(i)=(\sum_{f=c}^{d} A(f,i))/21;$$

i – номер текущей дальности, f – номер текущей составляющей БПФ,

$Кп_1$- коэффициент порога.

Факт обнаружения происходит при выполнении следующих условий:

$Acp_1(i) > П1$ [2],

$Acp_2(i) > П2$ [3].

При обнаружении отметки в одном из каналов пеленгационной пары ПП1 или ПП2 для дальнейшей обработки берется также одноименная ей по скорости (V) и дальности (D) парная отметка из другого канала и формируется двух лучевая отметка с амплитудами A1(i) и A2(i).

После обнаружения отметок составляется формуляр , который через ОЗУ информационной емкостью 128 К слов х 8 р, выполненное на микросхеме IDT71024S , выдается в ЦВМ первичной обработки

Литература

1. Силкин А.Т., Чернышев М.И, Янушевский Г.Д. , Тарасов В.Л, Сергеев В.В. Перспективы применения систем на кристалле в радиолокационных комплексах. - Наукоемкие технологии № 8, 9, 2004 г., т. 5, стр. 60 – 6 .

2. Верба В.С, Гандурин В.А, Трофимов А.А. Бортовая РЛС для перспективного многофункционального авиационного комплекса разведки, оповещения и управления (МАК РОУ) с цифровой АФАР. - Наукоемкие технологии № 8, 9, 2004 г., т. 5, стр. 110 – 116.

QUESTIONS OF USE MEMORY FOR REALIZATION ALGORITHM DIGITAL PROCESSING RADAR-TRACKING INFORMATION

In the report the block diagram and realization of algorithm of digital processing of the information on modern element base with application foreign programmed logic integrated circuits is considered.

УСТРОЙСТВА СЧИТЫВАНИЯ СИГНАЛОВ ЗАПОМИНАЮЩЕГО УСТРОЙСТВА НА ЦЦИЛИНДРИЧЕСКИХ МАГНИТНЫХ ДОМЕНАХ

Колонина Е.Г., Ульянов А.А., Косов В.И., Савельев А.И.

Московский Государственный Текстильный Университет им. А.Н. Косыгина

Использование доменной памяти, позволяющей создание и оформление изделий текстильной промышленности, приводились в нескольких статьях в частности в докладе: Доменная память для систем управления текстильным оборудованием, опубликованном в сборнике трудов Российского научно-технического общества Радиоэлектроники, электроники и связи имени А.С. Попова, М., 2004 [1].

В данном докладе предлагается использование усилителей считывания для доменной памяти, позволяющих повышенную достоверность воспроизведения сигналов.

Данное запоминающее устройство (ЗУ) было построено на цилиндрических магнитных доменах (ЦМД), выполненных на микросборках типа К1602 РЦ2.

В данной работе с целью повышения надежности считывания сигналов из доменного ЗУ предлагается два разработанных устройства. Первое из них содержит усилитель считывания, информационных вход которого подключен к выходу сумматора, подсоединенного к выходу инвертора, а вход инвертора подключен к выходу одного из ограничителей. Вход этого ограничителя подсоединен к входу другого ограничителя и к выходу эмиттерного повторителя. Вход эмиттерного повторителя подключен к выходу предварительного усилителя считывания, вход которого является информационным входом устройства. Вход тактового запуска устройства подключен к входу блока задержки, выходы которого подсоединены к входам одновибраторов. Выходы одновибраторов подключены к входам элементов ИЛИ, выход которых также подключен к входу стробирования усилителя считывания. Выход усилителя считывания подсоединен к счётному входу счётчика, вход сброса которого является входом сброса устройства. Выход счётчика подключен к входам группы элементов сравнения, входы другой группы которого подсоединены к выходам регистра начальной

установки. Входы регистра являются входами установки устройства в исходное состояние. Выход устройства подключен к выходу элемента сравнения, а выход ограничителя подсоединен к входу блока задержки. Выход блока задержки подключен к входу усилителя-формирователя, соединенного со вторым входом сумматора [2].

При считывании на входе устройства и на входе предварительного усилителя возникает разнополярный сигнал. Далее разнополярный сигнал считывания поступает на эмиттерный повторитель, с выхода которого он подается на первый и на второй входы ограничителей. За счет этого на выходе первого ограничителя сигнал имеет отрицательные составляющие, которые подаются на вход инвертора, а на выходе инвертора получают дополнительный сигнал. С выхода второго ограничителя сигнал, имеющий только положительную полярность, поступает на первый блок задержки и далее на усилитель-формирователь с целью восстановления параметров сигнала, прошедшего через блок задержки. Затем оба преобразованных сигнала подаются на входы сумматора. Поэтому на выходе сумматора будет получен более мощный сигнал, который и поступает на первый вход усилителя считывания, а на выход стробирующего усилителя считывания приходят сигналы стробирования с выхода элемента ИЛИ, полученные в разное время с одновибраторов. Длительность сигналов стробирования задается одновибраторами, а начало их следования с помощью блока задержки, на вход которого с выхода устройства подается запускающий сигнал. Затем сформированный сигнал “1 “ или “0” с выхода усилителя считывания подается на вход счётчика (предварительно в начале каждого такта считывания счётчик сигналов с входа устройства устанавливается в исходное состояние).

Код числа, зафиксированный в счётчике, поступает на входы одной группы элемента срабатывания, на входы другой группы которого подается код числа с регистра начальной установки. Значение кода числа в регистре начальной установки записывается по входу устройства и зависит от типа доменной микросборки, от особенностей ее работы и уровня помех, от числа импульсов стробирования (каждый полученный сигнал может стробироваться неоднократно), от статистических данных и вероятности воспроизведения считанного сигнала для конкретной доменной памяти. Кроме того, значение этого кода может быть определено в процессе отладки памяти путем оперативного его изменения.

Таким образом, в предполагаемом устройстве при выделении сигнала используются как положительные, так и отрицательные составляющие полезного сигнала, что увеличивает общую мощность сигнала и отношение сигнала к помехе, а, следовательно, предлагаемое устройство имеет повышенную надежность. Кроме того, надежность устройства возрастает за счет возможности выделения сигнала в сравнении с начальным, наперед заданным кодом для предотвращения ошибки, к примеру, при искажении сигнала в отдельные моменты времени при считывании.

Второй разработкой прелагается блок считывания для запоминающего устройства, ЦМД с повышенной надежностью воспроизведения сигналов [3].

Устройство содержит коррелятор, первые входы которого являются входами устройства. Вторые входы коррелятора подключены к выходу одного из эталонных элементов памяти и к входам первого дополнительного коррелятора. Второй вход первого дополнительного коррелятора подключен к входам третьего дополнительного коррелятора и к выходу другого эталонного элемента памяти. Вход первого эталонного элемента памяти соединен с входом второго эталонного элемента памяти и с выходом формирователя считывания сигналов, вход которого подключен к первому выходу тактового генератора, второй выход которого подсоединен к входу формирователя записи сигналов. Выход формирователя записи подключен к первым входам эталонных элементов памяти. Выход первого дополнительного коррелятора подключен к входу дополнительного предварительного усилителя воспроизведения и к первому входу сумматора, второй и третий входы сумматора подключены соответственно к выходу второго и третьего дополнительных корреляторов, а выход – к входу дискриминатора. Выход дискриминатора подключен ко вторым входам усилителей воспроизведения. Выходы усилителей воспроизведения подключены к входам числового регистра, а выходы – к входам предварительных усилителей воспроизведения. Первые входы усилителей воспроизведения подключены к выходам корреляторов. Вторые входы предварительных усилителей подсоединены к выходу дополнительного усилителя воспроизведения.

При считывании с выхода тактового генератора поступает сигнал на вход формирователя считывания сигналов. Одновременно происходит считывание сигналов из накопителя, которые поступают на первые входы корреляторов, на вторые входы которых приходит сигнал с выхода эталонного элемента памяти. С этого же выхода поступает сигнал на первый и второй входы первого дополнительного коррелятора и на один из входов второго коррелятора, на другой вход которого подается сигнал с выхода эталонного элемента памяти. С выхода эталонного элемента памяти сигнал поступает па первый и второй входы третьего дополнительного коррелятора. За счет этого при считывании на выходах корреляторов формируются информационные сигналы чтения, которые поступают на первые входы предварительных усилителей воспроизведения. На первом, втором и третьего корреляторах и соответственно формируются корреляционные сигналы первого эталонного элемента памяти, первого и второго элементов памяти и второго эталонного элемента памяти. Эти корреляционные сигналы поступают на соответствующие входы сумматора, с выхода которого

сумматорный сигнал подается на вход дискриминатора. На дискриминаторе формируется уровень дискриминации, который зависит от формы сигналов чтения и от условий считывания. Сформированный уровень дискриминации подается на вторые входы усилителей воспроизведения, на первые входы которых поступают сигналы с выходов предварительных усилителей воспроизведения. За счет того, что на входы предварительных усилителей воспроизведения подаются импульсы строба с выхода дополнительного предварительного усилителя воспроизведения, которые представляют собой сигнал автокорреляции первого эталонного элемента памяти, обеспечивается оптимальное время стробирования. Сигналы чтения с усилителей воспроизведения подаются на входы числового регистра.

Литература

1.Косов В.И. и др., Доменная память для систем управления текстильным оборудованием, труды Российского научно-технического общества радиотехники, электроники и связи имени А.С. Попова, М., 2004.

2.Савельев А.И., Авторское свидетельство СССР №1294164, 1987

3.Савельев А.И., Авторское свидетельство СССР №959155, 1982

THE DEVICES OF SENSING SIGNALS OF THE REMEMBERING DEVICE ON CYLINDRICAL MAGNETIC DOMAINS

Possibilities and devices are considered, allowing sensing signals of the remembering device, running for cylindrical magnetic domains by use of full energy of the reproduced signal and special methods of the discriminations and strobing.

О СТРУКТУРНЫХ ПРИНЦИПАХ ПОСТРОЕНИЯ ПОСТОЯННОЙ ПАМЯТИ ИНФОРМАЦИОННЫХ СИСТЕМ

Крепкая Е.В., Терехова А.И., Иванов А.М.

Московский государственный текстильный университет им. А.Н.Косыгина

Неуклонная тенденция снижения стоимости полупроводниковых БИС ПЗУ, увеличения и быстродействия, надежности и простая технология изготовления позволили с новых позиций взглянуть на них как на эффективные средства создания специализированных процессоров и средства реализации табличных методов обработки информации. В традиционных применениях ПЗУ было пассивным устройством. Однако преобразование ПЗУ в активное устройство осуществляется посредством записи в его структуру закона (таблицы) обработки информации. При этом принципы построения и функционирования ПЗУ изменяются незначительно.

Построение ПЗУ различной информационной емкости производится посредством объединения серийно выпускаемых БИС памяти таким образом, чтобы получить необходимые информационную емкость и разрядность. При этом способы объединения БИС памяти в определенные структуры существенно влияют на время выборки информации и потребляемую мощность.

Наиболее широко используются следующие способы построения ПЗУ с наращиванием информационной емкости: по разрядам (R), по словам (W), как по словам, так и по разрядам (RW). В структуре R объединяются одноименные адресные входы и входы выборки БИС памяти. В этой структуре время выборки информации всего устройства практически не изменяется по сравнению с временем выборки информации БИС памяти, а потребляемая мощность увеличивается в M/m раз, где M – разрядность слова устройства, m – разрядность слова БИС памяти, к – число БИС памяти в устройстве, n – разрядность адресного слова (рис. 1,а).

В структуре W объединяются одноименные адресные входы и разрядные выходы БИС памяти, а их входы выборки кристалла CS подключаются к выходам дополнительного дешифратора. В этой структуре время выборки информации устройства увеличивается по сравнению с временем выборки информации типовой БИС памяти из-за последовательного включения дешифратора и за счет увеличения паразитной емкости, возникающей при объединении выходов нескольких БИС памяти. Потребляемая мощность устройства со структурой W при равном числе используемых БИС памяти практически не изменяется (рис. 1,б).

В структуре RW БИС памяти образуют матричную структуру, имеющую L строк и K столбцов. Одноименные адресные входы всех БИС памяти объединяются, входы выборки кристалла БИС памяти одного столбца матричной структуры также объединяются и объединяются одноименные разрядные выходы БИС памяти одной строки матричной структуры. Для выборки одного столбца БИС памяти применяется дополнительный дешифратор. В этой структуре время выборки информации устройства увеличивается так же, как

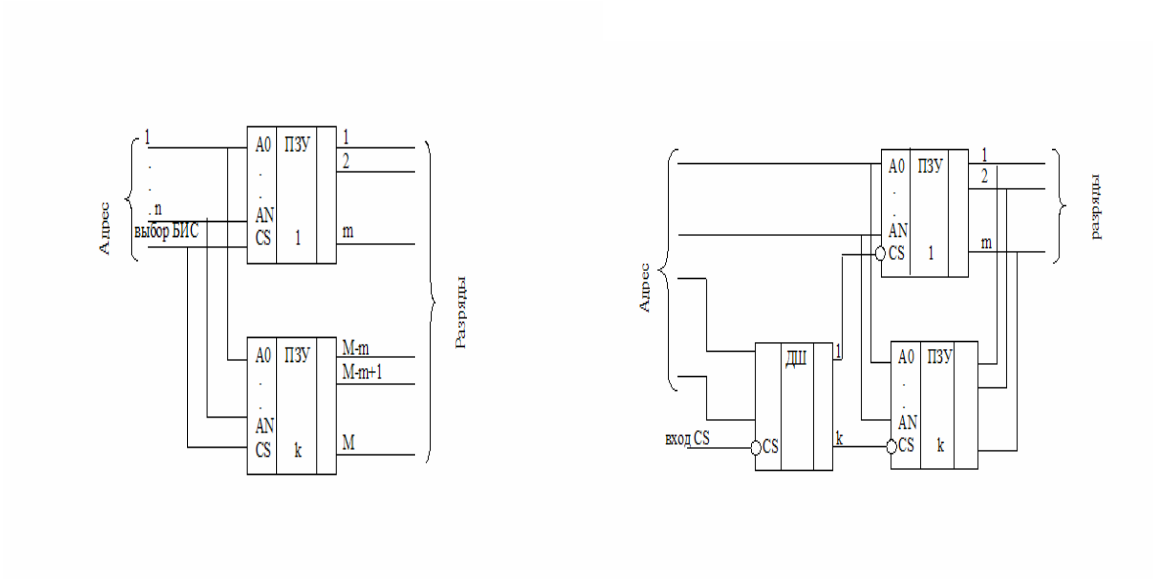
и в структуре W, с той лишь разницей, что столбцов матричной структуры будет несколько. Возрастает и потребляемая мощность (рис.2).

Наличие у БИС памяти нескольких входов разрешения выборки кристалла (например CS1, CS2, CS3) упрощает систему дешифрации устройства .

В случае коррекции информации одного слова, распределенного по нескольким БИС памяти, необходимо менять или группу БИС памяти или все БИС памяти одного столбца (рис. 2). Уменьшение времени коррекции, ее стоимости, сокращения числа заменяемых БИС памяти можно достичь за счет того, что в БИС памяти хранить не один или несколько разрядов слова, а все разряды слов определенного подмассива информации.

Для иллюстрации этого рассмотрим пример. Пусть устройство памяти с традиционной структурой имеет информационную емкость 4К слов по 16 разрядов. Для построения устройства используются БИС памяти с организацией 4Кх1. Общее число БИС памяти равно 16.

При использовании тех же БИС памяти для построения ПЗУ с полноразрядным считыванием слов из каждой БИС памяти в последней окажутся записанными 256 слов по 16 разрядов каждое, т.е. организация БИС памяти изменится с 4Кх1 на 0,25Кх16. Очевидно, информационная емкость устройства как в первом, так и во втором случаях будет одна и та же (4Кх1х16 и 0,25Кх16х16). Примером последней структуры является структура ПЗУ с информационной емкостью 8 слов на 8 разрядов (рис. 3).



а б

Рис. 1

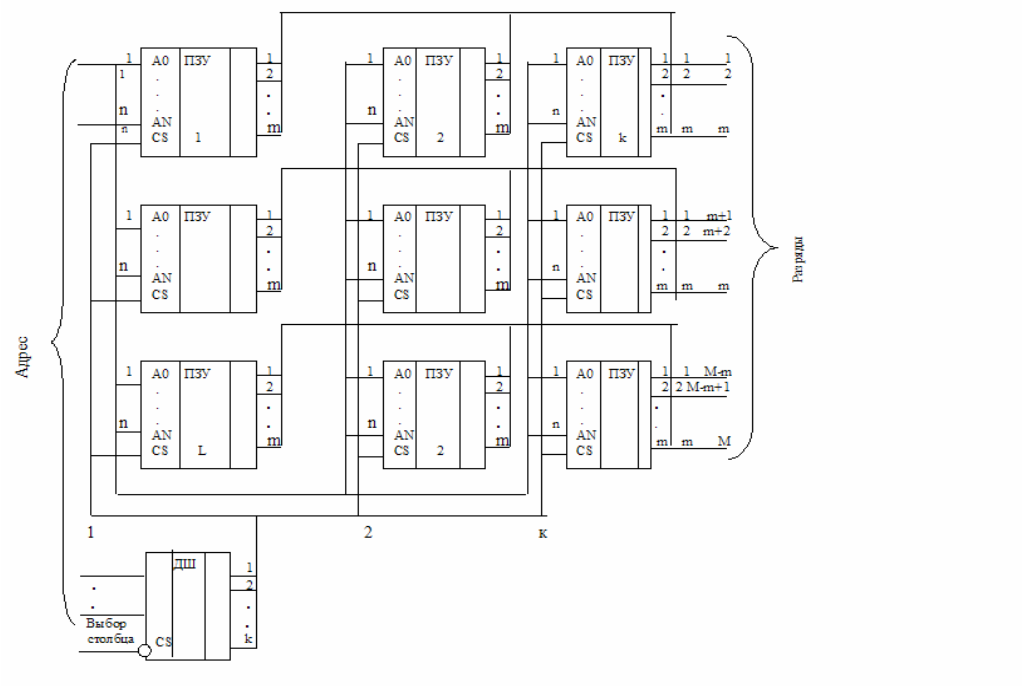


Рис. 2

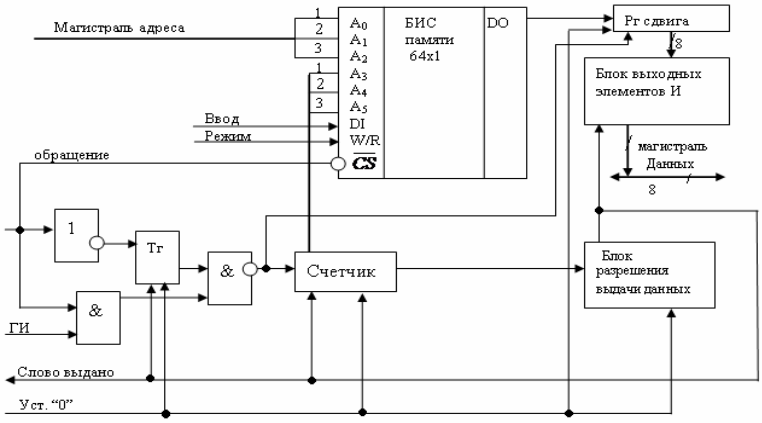


Рис. 3

Аналогичным образом можно организовать считывание полноразрядных слов из БИС памяти с организацией побайтового считывания и формирования полноразрядных слов из считанных байтов. Таким образом подобные структуры позволяют хранить в каждой типовой БИС памяти полноразрядные слова и считывать информацию из каждой БИС памяти или побитно или побайтно. При этом коррекция ранее записанной информации осуществляется заменой только той БИС памяти, в которой подмассив информации имеет ошибки.

Литература

1. Полупроводниковые БИС запоминающих устройств: справочник/ В.В.Баранов, М.В.Бекин, А.Ю.Гордонов и др.; Под ред. А.Ю.Гордонова и Ю.Н.Дьякова. – М.: Радио и связь, 1986;

2. Горохов Е.В. и др. Построение ПЗУ и ППЗУ большой информационной емкости на серийных БИС памяти. – М.: МГТУ им. А.Н.Косыгина, 1990:

3. Авт. свид. СССР №1365130.

ABOUT STRUCTURE PRINCIPLES OF BUILDING READ-ONLY MEMORY OF INFORMATIONAL SYSTEM

Block schemes of the read-only memory, allow increasing count of bits in scanned words, count of scanned words, read and write words from memory are considered.

МАТЕМАТИЧЕСКИЕ МОДЕЛИ ОЦЕНКИ ВЛИЯНИЯ ТЕХНИЧЕСКОГО ОБСЛУЖИВАНИЯ НА НАДЕЖНОСТЬ ПОДСИСТЕМЫ ПАМЯТИ

Лужецкая В.Г., Радов А.В., Иванов А.М., Севостьянов П.А.

Московский государственный текстильный университет им. А.Н. Косыгина

Целью технического обслуживания подсистемы памяти является предупреждение отказов и повышение ее надежности. Перед проведением технического обслуживания подсистемы необходимо обосновать стратегию технического обслуживания, выполнить оценку объема работ, оценить экономические затраты, при построении моделей принять обоснованные допущения, позволяющие изучить влияние технического обслуживания на надежность подсистемы[1-3].

Разработку математических моделей оценки влияния технического обслуживания на надежность подсистемы памяти необходимо предварить следующими допущениями:

- время восстановления подсистемы не всегда распределяется по экспоненциальному закону, так как предположение о постоянстве времени восстановления будет противоестественным, что подтверждено практикой;
- однако данные практики свидетельствуют о том, что среднее время безотказной работы БИС значительно превышает время восстановления работоспособности изделия. В связи с этим многие показатели надежности не зависят от вида закона распределения времени восстановления и может быть принято допущение об экспоненциальном законе распределения времени восстановления подсистемы;
- отказы каждой БИС памяти и общего оборудования подсистемы взаимно независимы;
- техническое обслуживание подсистемы производится через определенный фиксированный интервал времени. Для упрощения математических моделей оценки влияния технического обслуживания на надежность подсистемы памяти можно принять допущение о том, что из каждого работоспособного состояния подсистема в среднем через $1/\lambda_{то}$ единиц времени переходит в состояние технического обслуживания. При этом можно также считать, что время до начала проведения технического обслуживания распределено по экспоненциальному закону с параметром $\lambda_{то}$, где $\lambda_{то}$ – интенсивность технического обслуживания;
- допущение об экспоненциальном законе распределения времени до начала проведения технического обслуживания является, однако, довольно грубым и позволяет лишь качественно изучить влияние технического обслуживания на надежность подсистемы памяти. Для того, чтобы это допущение не сказывалось на результатах моделирования, достаточно принять, что $\lambda_{то} = 0$;
- функционирование подсистемы памяти описывается марковским процессом с непрерывным временем.

Одной из моделей оценки влияния технического обслуживания на надежность подсистемы памяти является математическая модель оценки влияния числа отказавших БИС на время восстановления работоспособности подсистемы памяти. При этом рассмотрено множество состояний подсистемы, в число которых входят как работоспособные, так и отказовые состояния с учетом параметров подсистемы: L – число линеек; n – число разрядов кодового слова; k – число разрядов информационного слова; λ – интенсивность отказов БИС памяти; λ_o – интенсивность отказов общего оборудования подсистемы; μ_o – интенсивность восстановления общего оборудования подсистемы памяти; $\lambda_{то}$ – интенсивность перехода подсистемы в состояние технического обслуживания; μ_1 – интенсивность перехода из состояния технического обслуживания в работоспособное состояние; ρ – некоторое целое пороговое число, принимающее значение от 0 до L. При составлении графа состояний подсистемы памяти для этого случая учитывается множество отказовых состояний подсистемы в отличие от [3], из которых под влиянием восстановления ее работоспособности производится переход в исходное работоспособное (0-е) состояние, характеризующееся полным отсутствием отказов. Следует, однако, заметить, что восстановление подсистемы из каждого отказового состояния будет выполняться с разной интенсивностью $\mu_{oj,}$ где μ_{oj} – интенсивность восстановления из j-го отказового состояния в исходное работоспособное состояние. На основании полученного графа состояний подсистемы представлены система дифференциальных уравнений Колмогорова и система линейных алгебраических уравнений для

определения вероятностей нахождения системы в различных состояниях для стационарного режима. Эти вероятности определяются с помощью выражения

$$P_j = P_O \theta_j , \quad (1)$$

где P_j – вероятность нахождения подсистемы в j-м состоянии, P_0 – вероятность нахождения подсистемы в 0-м состоянии;

$$\theta_j = n^j \lambda^j \prod_{i=1}^{j} (L - i + 1) / ((nL - i)\lambda + \lambda_O + \lambda_{TO}) \quad (2)$$

Стационарный коэффициент готовности подсистемы с учетом зависимости времени восстановления от числа отказавших БИС памяти и общего оборудования определен с помощью выражения

$$K_\Gamma = \sum_{j=0}^{\rho} P_j / \sum_{j=0}^{2\rho+1} P_j , \quad (3)$$

где (2ρ+1)-е состояние – это состояние, в котором в подсистеме отказали две БИС памяти в одной из линеек и по одной БИС памяти в (ρ-1)-й линейках или отказало общее оборудование и по одной БИС памяти в ρ линейках. (2ρ+2)-е состояние – это состояние технического обслуживания.

Определены среднее время восстановления подсистемы, ее среднее время наработки с учетом числа отказавших БИС. Выполнено компьютерное моделирование влияния принятой стратегии восстановления подсистемы памяти с заменой только той БИС, отказ которой вызвал отказ подсистемы, а также выполнена оценка принятой стратегии полного восстановления подсистемы памяти после ее попадания в критическое состояние (ρ-е состояние). Показано, что с увеличением интенсивности восстановления коэффициент стационарной готовности увеличивается, а с увеличением интенсивности отказов БИС памяти – уменьшается.

Литература

1. Барзилович Е.Ю., Каштанов В.А. Некоторые математические вопросы теории обслуживания сложных систем.- М.: Сов. Радио, 1971

2. Горшков В.Н. Надежность оперативных запоминающих устройств ЭВМ. – Л.: Энергоатомиздат, 1987

3. Мысина Г.В., Лужецкая В.Г., Радов А.В. и др. Компьютерное моделирование влияния параметров подсистемы памяти на ее надежность. Всероссийская НТК "Современные технологии и оборудование текстильной промышленности" (Текстиль – 2004),тезисы докладов. – М.: МГТУ им. А.Н. Косыгина, 2004

THE MATHEMATICAL MODELS OF ESTIMATION OF INFLUENCES OF THE TECHNICAL SERVICE ON THE CHARACTERISTICS OF RELIABILITY OF THE SUBSYSTEM OF MEMORY

The influence of the technical service on the characteristics of reliability of the subsystem of memory was considered. The mathematical models of estimation of influence of the number of refused BIS on time of restoration of the subsystem of memory, the estimations of accepted strategy of restoration with substitution only BIS which had brought to refused subsystem, the estimations of accepted strategy of full restoration after getting in the critical state were described. The models were used under computer model of influence of the technical service on reliability of the subsystem of memory.

———————♦———————

ПРИМЕНЕНИЕ КОДОВ ХЕММИНГА В ПОСТОЯННЫХ ЗАПОМИНАЮЩИХ УСТРОЙСТВАХ

Крепкая Е.В., Терехова А.И., Иванов А.М.

Московский Государственный текстильный Университет им. А.Н. Косыгина

Модифицированный код Хемминга обнаруживает некоторые кратные ошибки, но среди ошибок, кратность которых больше двух, могут быть такие, которые этот код не обнаруживает, пропуская их или выполняя неправильную коррекцию ошибок. В связи с этим коды Хемминга применяются в тех устройствах, к достоверности информации которых предъявляются высокие требования и в которых с учетом их схемотехнической особенности наиболее вероятны одиночные ошибки и совсем редко ошибки большей кратности. Примером таких устройств является полупроводниковая оперативная память, построенная на одноразрядных БИС памяти, вероятность сбоев в которых очень высока. В ряде работ также утверждается, что если используется 4-хразрядные БИС памяти, для которых достаточно вероятен сразу 4-хразрядный пакет ошибок, то применение кода Хемминга становится неэффективным [1].

Построение восьмипортового ПЗУ

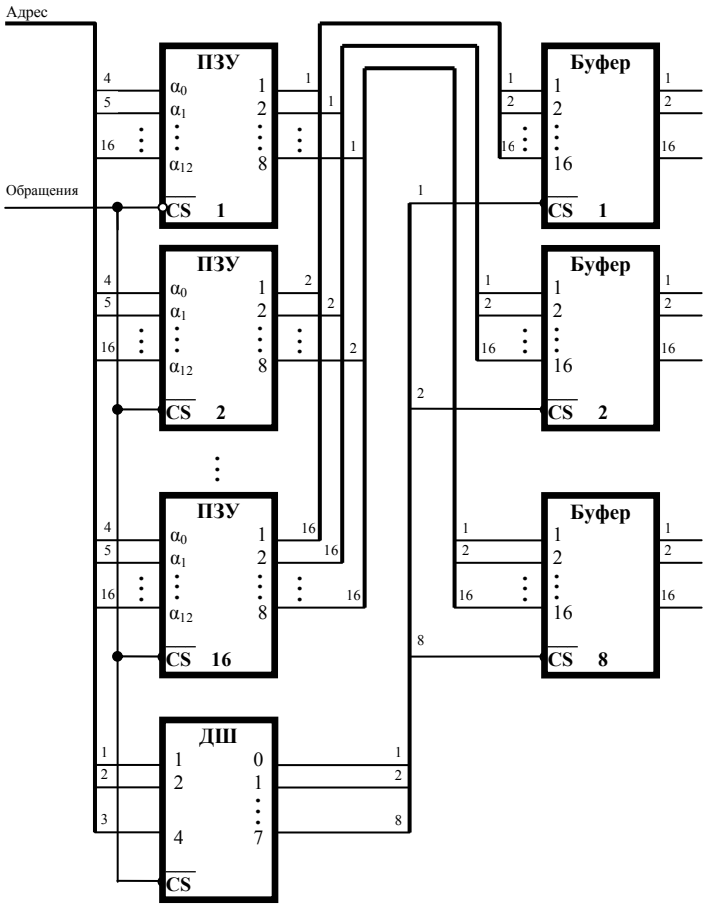


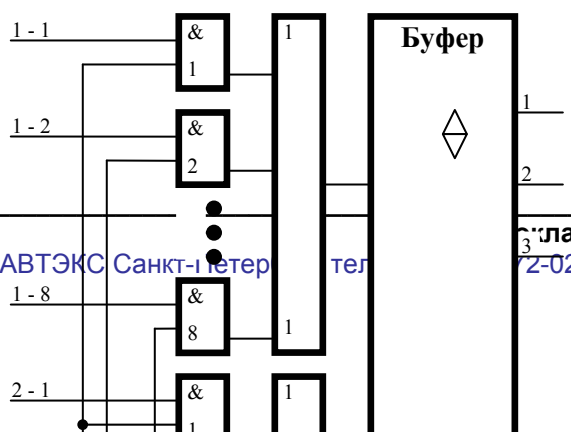
Рис. 1

Однако при использовании нетрадиционного подхода к построению устройств памяти на БИС с большим числом информационных выходов и входов от этого недостатка кода Хемминга можно избавиться. Примером могут служить две схемы ПЗУ, приведенные ниже.

Структурная схема восьмипортового ПЗУ с организацией $8К^{x}8^{x}16$, в котором предполагается использовать модифицированный код Хемминга, приведена на рис.1.

В устройстве использованы 16 БИС ПЗУ, дешифратор на 3 входа и 8 выходов, 8 буферов на 16 информационных входов и 16 информационных выходов. Информационные выходы БИС ПЗУ и информационные входы буферов так соединены, что каждый разряд БИС ПЗУ можно рассматривать как отдельную виртуальную одноразрядную по выходу БИС-аналог одноразрядной БИС памяти. Один и тот же адрес (его старшие разряды) поступают на все БИС ПЗУ. Одноименные разрядные выходы каждой БИС ПЗУ подключены к информационным входам соответствующего буфера (16 БИС ПЗУ и 16 информационных входов буфера). Выбор соответствующего буфера происходит с помощью дешифратора при поступление на его входы младших разрядов адреса. При обращении к ПЗУ на входы $\overline{CS}$ каждой БИС памяти и дешифратора поступает сигнал ОБРАЩЕНИЕ, и восемь слов по 16 разрядов сразу приходят на информационные входы соответствующих буферов – каждое слово на входы своего буфера. Однако слово вводится только в тот буфер, который выбран по входу $\overline{CS}$. Таким образом, эта структура представляет собой восемь виртуальных блоков ПЗУ, каждый из которых хранит свой массив слов.

Структурная схема ПЗУ с одним портом
(поле памяти и дешифратор не показаны)



На рис.2 приведена структурная схема ПЗУ с одним портом. Особенностью этой структуры является использование между полем памяти на БИС ПЗУ и буфером специального коммутатора (поле памяти не показано). Слева на рис.2 связи, идущие от поля памяти к буферу, имеют обозначения, которые состоят из двух чисел, разделенных знаком “–”. Первое число – это номер

БИС ПЗУ, а второе – номер разрядного выхода БИС ПЗУ.

Специальный коммутатор состоит из группы элементов И, управляемых сигналами с выходов дешифратора, на которые поступают младшие разряды адреса.

Выполнено компьютерное моделирование процедур формирования исходного массива информации, записи кодовых слов в поле памяти, считывания с использованием языков программирования высокого уровня.

Литература

1. Потемкин И.С. Функциональные узлы цифровой автоматики. – М.: Энергоатом издат, 1988
2. Петросян О.А., Козырь И.Я., Коледов Л.А., Щетинин Ю.И. Схемотехника БИС постоянных запоминающих устройств. – М.: Радио и связь, 1987
3. Полупроводниковые БИС запоминающих устройств: Справочник / В.В. Баранов, Н.В. Белкин, А.Ю. Гордонов, Ю.А. Гордонов, А.В. Калинин, Е.П. Лепехин, Э.П. Севостьянов, В.П. Сидоренко, Ю.Н. Смирнов, В.В. Цыркин; под ред. А.Ю. Гордонов и Ю.Н. Дьякова. – М.: Радио и связь, 1986

USING CODES OF HEMMING IN READ-ONLY MEMORY

Block schemes of the read-only memory where field of the memory is constructed on serial BIS read-only memory are considered. These block schemes are received with the purpose to use modified code Hemming for correction single mistakes and detection double mistakes.

Рис. 2

ИСПОЛЬЗОВАНИЕ КОДОВ ЭЛАЙЕСА И ХЕММИНГА ДЛЯ ИСПРАВЛЕНИЯ ОШИБОК В ЗАПОМИНАЮЩИХ УСТРОЙСТВАХ

Матвеева С.М., Чистова Т.Н., Иванов А.М.

Московский государственный текстильный университет им. А.Н. Косыгина

Метод коррекции ошибок с помощью двумерного кода Элайеса заключается в том, что, если не выполняется проверка на четность для i – ой строки и j – го столбца, то символ b_{ij}, лежащий на их пересечении, инвертируется. Таким образом исправляются все одиночные, часть двойных, тройных и других ошибок более высокой кратности. Обнаруживаются также ошибки, при которых только строка или столбец содержат нечетное число искаженных символов.

Коды Элайеса обладают высокой помехозащищенностью как по отношению к независимым искажениям информации, так и по отношению к пакетам ошибок.

Совместное применение кодов Элайеса и Хемминга позволяет получить новый способ исправления ошибок в запоминающих устройствах. Рассмотрим его на примере информационных слов с разрядностью k

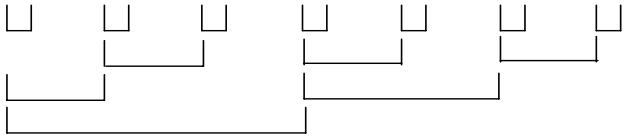
= 9. Тогда число возможных информационных слов равно $2^9 = 512$, каждое из которых имеет вид $B = [b_9\ b_8\ b_7 \ldots b_2\ b_1]'$, где b_i - информационный символ.

Информационное слово вида $b_9\ b_8\ b_7 \ldots b_2\ b_1$ с конкретными значениями информационных разрядов 110010110 может быть представлено в виде матрицы:

B_9	b_8	b_7		1	1	0
B_6	b_5	b_4	или конкретно	0	1	0
B_3	b_2	b_1		1	1	0

Формирование кодового слова с помощью классического кода Хемминга производится по следующему алгоритму, условно показанному ниже. Над кодовым словом указаны номера разрядов:

13	12	11	10	9	8	7	6	5	4	3	2	1
1	1	0	0	1	1	0	1	1	0	0	1	1



Внизу, под символами кодового слова скобками указаны группы разрядов, участвующих в формировании контрольных разрядов кодового слова, которые занимают позиции 2^j, где j = 0, 1, 2, 3, ..., т.е. позиции 1, 2, 4, 8, … Суммирование по mod 2 информационных символов каждой группы обеспечивает получение контрольных разрядов кодового слова, занимающих позиции 1, 2, 4, 8. При таком формировании кодового слова обеспечивается исправление одиночных ошибок и обнаружение двойных, но не более того.

	6	5	4	3	2	1
6	1	1		0		
5	0	1		0		
4						
3	1	1		0		
2						
1						

Однако, если применить код Хемминга к каждой строке и каждому столбцу, то матричное представление кодового слова будет иметь вид, показанный справа.

Получим контрольные разряды для строк этой матрицы, заполненных информационными символами: 6-я строка 110011, 5-я строка 011001, 3-я строка 110011. Заполним матрицу – шаблон кодового слова, полученными для строк контрольными разрядами:

	6	5	4	3	2	1
6	1	1	0	0	1	1
5	0	1	1	0	0	1
4						
3	1	1	0	0	1	1
2						
1						

Применим теперь код Хемминга к каждому столбцу с учетом контрольных разрядов, полученных для каждой строки: 6-й столбец 101101, 5-й столбец 110100, 4-й столбец 011001, 3-й столбец 000000, 2-й столбец 101101, 1-й столбец 110100. Теперь полностью заполним матрицу кодового слова контрольными разрядами, полученными применением кода Хемминга к каждому столбцу (вид справа). Над матрицей указаны номера столбцов, а слева – номера строк.

	6	5	4	3	2	1
6	1	1	0	0	1	1
5	0	1	1	0	0	1
4	1	0	1	0	1	0
3	1	1	0	0	1	1
2	0	0	0	0	0	0
1	1	0	1	0	1	0

Произведем проверку исходного кодового слова, используя классический код Хемминга, 1100110110011. С помощью этой проверки получим синдром ошибки S = 0000, свидетельствующий об отсутствии одиночной ошибки.

Теперь произведем проверку кодового слова, сформированного в виде последней матрицы, сначала по строкам, а потом по столбцам. При проверке по строкам номера позиций располагаются справа налево, начиная с первой:

6-я строка 110011. Синдром ошибки $S'_6 = 000$; 5-я строка 011001. Синдром ошибки $S'_5 = 000$;

4-я строка 101010. Синдром ошибки $S'_4 = 000$; 3-я строка 110011. Синдром ошибки $S'_3 = 000$;

2-я строка 000000. Синдром ошибки $S'_2 = 000$; 1-я строка 101010. Синдром ошибки $S'_1 = 000$.

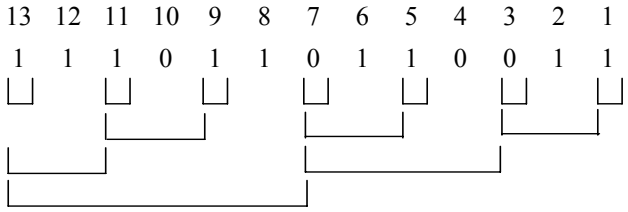
Проверка по строкам показала, что все построчные синдромы равны 000.

Проведем проверку по столбцам. При проверке по столбцам номера позиций располагаются снизу вверх, начиная с первой:

6-й столбец 101101. Синдром ошибки $S''_6 = 000$; 5-й столбец 110100. Синдром ошибки $S''_5 = 000$; 4-й столбец 011001. Синдром ошибки $S''_4 = 000$; 3-й столбец 000000. Синдром ошибки $S''_3 = 000$; 2-й столбец 101101. Синдром ошибки $S''_2 = 000$; 1-й столбец 110100. Синдром ошибки $S''_1 = 000$.

Проведенные проверки свидетельствуют об отсутствии ошибок.

Внесем одну ошибку в исходное кодовое слово, представленное с помощью классического кода Хемминга и определим синдром ошибки:



Синдром ошибки равен 1011. Следовательно, в исходном кодовом слове ошибка содержится в 11-м разряде, который следует исправить инвертированием. Кодовое слово с ошибкой, представленное в виде матрицы, имеет вид, приведенный на рис.1. Проверка, проведенная по 6-й строке и 3-м столбце, дает следующий результат: 6-я строка 110111. Синдром ошибки $S'_6 = 011$,

	6	5	4	3	2	1
6	1	1	0	1	1	1
5	0	1	1	0	0	1
4	1	0	1	0	1	0
3	1	1	0	0	1	1
2	0	0	0	0	0	0
1	1	0	1	0	1	0

Рис.1

3-й столбец 100000. Синдром ошибки $S''_3 = 110$. Таким образом, ошибка присутствует в 3-й позиции 6-й строки и 6-ой позиции 3-го столбца. Как и раньше в коде Элайеса, ошибка находится на пересечении i-й строки и j-го столбца (i = 6, j = 3).

Рассмотрим случай 3-х однократных ошибок, расположенных в разных строках и столбцах. Позиции ошибок кодового слова в матричном представлении заштрихованы (рис.2).

	6	5	4	3	2	1
6	1	0	0	0	1	1
5	1	1	1	0	0	1
4	1	0	1	0	1	0
3	1	1	0	1	1	1
2	0	0	0	0	0	0
1	1	0	1	0	1	0

Рис.2

Выполним проверку кодового слова в этом представлении сначала по строкам, а потом по столбцам. При проверке по строкам номера позиций располагаются справа налево, начиная с первой:

6-я строка 100011. Синдром ошибки $S'_6 = 101$; 5-я строка 111001. Синдром ошибки $S'_5 = 110$;

4-я строка 101010. Синдром ошибки $S'_4 = 000$; 3-я строка 110111. Синдром ошибки $S'_3 = 011$;

2-я строка 000000. Синдром ошибки $S'_2 = 000$; 1-я строка 101010. Синдром ошибки $S'_1 = 000$.

Результаты проверки по строкам свидетельствуют о наличии одиночных ошибок в 6-й (позиция 5), 5-й (позиция 6) и 3-й строках (позиция 3).

При проверке по столбцам номера позиций располагаются снизу вверх, начиная с первой:

6-й столбец 111101. Синдром ошибки $S''_6 = 101$; 5-й столбец 010100. Синдром ошибки $S''_5 = 110$;

4-й столбец 011001. Синдром ошибки $S''_4 = 000$; 3-й столбец 000100. Синдром ошибки $S''_3 = 011$;

2-й столбец 101101. Синдром ошибки $S''_2 = 000$; 1-й столбец 110100. Синдром ошибки $S''_1 = 000$.

Результаты проверки по столбцам свидетельствуют о наличии одиночных ошибок в 6-м (позиция 5), 5-м (позиция 6) и 3-м столбцах (позиция 3).

Результаты проверок по столбцам и строкам подтверждают друг друга и, следовательно, могут быть использованы для исправления трех одиночных ошибок в кодовом слове.

Рассмотренные примеры явились основанием для проведения компьютерного моделирования на языках высокого уровня процедур обнаружения и исправления ошибок различной кратности в запоминающих устройствах.

Литература

1. Хетагуров Я.А., Руднев Ю.П. Повышение надежности цифровых устройств методами избыточного кодирования. – М.: Энергия, 1974

2. Дмитриев В.И. Прикладная теория информации. – М.: Высш. шк., 1989

3. Борисенок М.А., Лукьянов С.Е., Иванов А.М. Применение специального кодирования информации при ее передаче и хранении. Труды LVII Научной сессии, посвященной Дню Радио, том 2. – М.: РНТОРЭС им. А.С. Попова, 2002

4. Матвеева С.М. Итеративные коды и их использование для обнаружения ошибок при передаче и хранении информации. – М.: МГТУ им. А.Н. Косыгина, 2004

USAGE OF ELIESE'S AND HEMMING'S CODES FOR ERROR CORRECTION IN CODE STORAGE

Join usage of Eliese's and Hemming's codes for error correction in code storage have been studied. Join usage of Eliese's and Hemming's codes allows us to get another one way of error correction in code storage. Report is illustrated with case study. Computer simulation of procedures of information's encoding and decoding whith different order error detection have been executed.

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ОЦЕНКИ НАДЕЖНОСТИ ПОДСИСТЕМЫ ПАМЯТИ С УЧЕТОМ СБОЕВ

Лужецкая В.Г., Радов А.В., Иванов А.М., Стрельников Б.А., Севостьянов П.А.

Московский государственный текстильный университет им. А.Н. Косыгина

Статистические данные свидетельствуют о том, что сбои возникают в памяти в 5…10 раз чаще, чем отказы, для вычислительных систем среднее время работы между сбоями в 20 раз меньше среднего времени между устойчивыми отказами. Опасность сбоев в том, что они при работе подсистемы памяти приводят к искажению хранимой и считываемой информации – данных или программ.

Современные вычислительные системы, как правило, снабжены достаточно эффективными аппаратными и программными средствами исправления последствий сбоев, обеспечивающими автоматическое исправление 80…90% всех отказов и сбоев. Одним из аппаратных способов обнаружения и исправления ошибок в подсистемах памяти является использование помехоустойчивого кодирования.

Структурная схема подсистемы памяти с помехоустойчивым кодированием информации приведена на рис.1.

Структурная схема подсистемы памяти с помехоустойчивым кодированием информации.

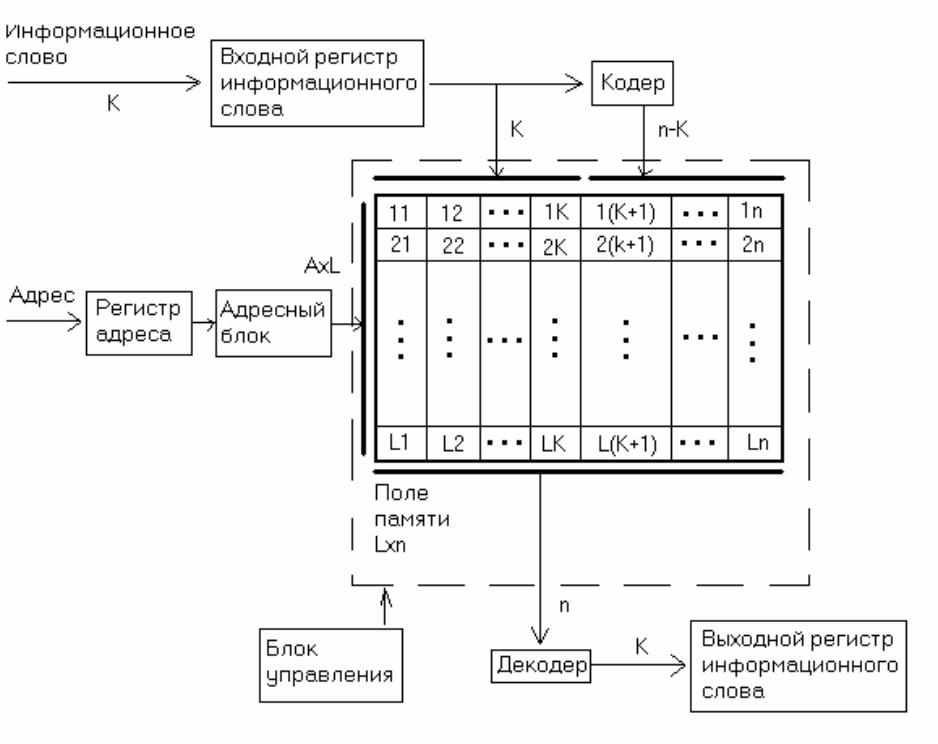


Рис.1

В структурной схеме приняты следующие обозначения:

k – число разрядов информационного слова;

n – число разрядов кодового слова;

n-k – число контрольных разрядов;

L – число линеек БИС памяти. Каждая линейка содержит А кодовых слов.

В [1 - 3] рассматривалась математическая модель оценки надежности подсистемы памяти с учетом отказов. При разработке математической модели подсистемы памяти с учетом сбоев примем некоторые допущения. Время до появления сбоя в одной БИС памяти и общем оборудовании с интенсивностями λ_c и λ_{oc} соответственно подчинено экспоненциальному закону. По характеру влияния на работу подсистемы имеют место защищенные сбои и ошибки из-за сбоев. Первые обнаруживаются системой контроля и автоматически устраняются с помощью информационной избыточности. Ошибки из-за сбоев являются результатом деструктивного воздействия незащищенных сбоев. Ошибки из-за сбоев с большей вероятностью возникают в тех линейках, в которых уже есть отказавшие БИС памяти, чем в тех, в которых отказавшие БИС памяти отсутствуют.

Граф состояний подсистемы памяти с учетом сбоев.

В связи с этим при оценке эффективности использования того или иного способа помехоустойчивого кодирования по отношению к сбоям целесообразно учитывать стационарные вероятности пребывания подсистемы памяти в работоспособных состояниях, выявленных по отношению к отказам. Для этого выделим j-тое работоспособное состояние подсистемы памяти(j=0, 1,… , ρ, где ρ - некоторое пороговое целое число, принимающее значения от 0 до L). Тогда процесс появления сбоев в БИС памяти может быть описан с помощью графа состояний (рис.2).

Итак, по отношению к отказам рассмотрено j-е работоспособное состояние. Тогда, как следует из рис.2, подсистема может находиться в следующих состояниях:

0 – нет сбоев в БИС памяти;

1 – произошел сбой в одной из БИС памяти из числа линеек, в которых отсутствуют БИС памяти с отказами;

.
.
.

h – произошел сбой в h БИС памяти, которые распределены по линейкам, не имеющим отказавших БИС памяти;

.
.
.

(L-j) – произошел сбой в (L-j) БИС памяти, которые распределены по всем линейкам, не имеющим отказавших БИС памяти;

(L-j+1) – возникла ошибка, то есть информация считана и выдана на выход подсистемы искаженной, а это как раз опасное событие.

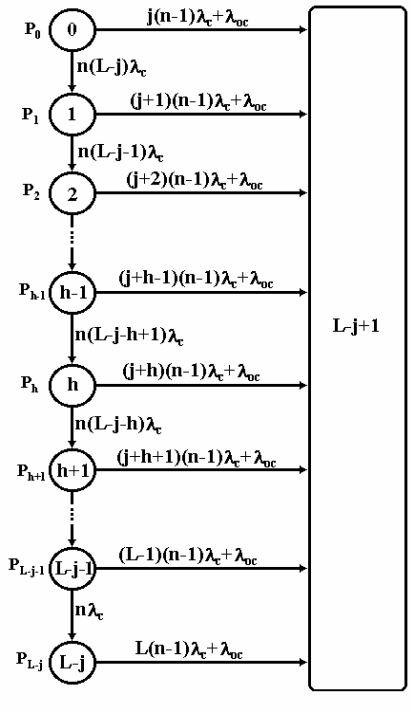


Рис.2

На основании этого графа состояний относительно сбоев составлены система дифференциальных уравнений и после преобразований Лапласа получена система алгебраических уравнений, из которых определены стационарные вероятности состояний подсистемы относительно сбоев.

Тогда средняя наработка подсистемы на ошибку из-за сбоев T_C определена с помощью соотношения

$$T_C = \sum_{j=0}^{\rho} P_j T_{jC}$$

где P_j – стационарная вероятность нахождения подсистемы в j-м работоспособном состоянии;

T_{Cj} – стационарное время нахождения подсистемы в j-м состоянии по отношению к сбоям.

Отсюда следует, что средняя наработка подсистемы памяти с учетом сбоев T_C является функцией параметров ρ, λ_0, λ, λ_{oc}, λ_c, μ_1 и зависит от выбранной стратегии восстановления, где

λ_0 – интенсивность отказов общего оборудования;

μ_1 – интенсивность восстановления БИС памяти;

μ_0 – интенсивность восстановления общего оборудования.

Выполнено компьютерное моделирование влияния параметров подсистемы памяти на ее надежность с учетом сбоев.

Литература

1. Горшков В.Н. Надежность оперативных запоминающих устройств ЭВМ. – Л.: Энергоатомиздат,1987,168с.

2. Карабанова М.В., Мысина Г.В., Иванов А.М. Применение корректирующих кодов для защиты информации в ЗУ. Труды РНТОРЭС им. А.С. Попова. Серия: Научная сессия, посвященная Дню Радио, вып. LIX-2. – М.: РНТОРЭС им. А.С. Попова, 2004

3. Мысина Г.В. Разработка АРМ для компьютерного моделирования характеристик надежности технических систем(на примере оперативной памяти). – М.: МГТУ им. А.Н. Косыгина, 2004

THE MATHEMATICAL MODEL OF ESTIMATION OF RELIABILITY OF THE SUBSYSTEM OF MEMORY ON IT RELIABILITY TAKING INTO ACCOUNT THE MALFUNCTIONS

The danger of the malfunctions was noted. The structured scheme of the subsystem of memory with noise-immunity coding was presented. The mathematical model of estimation of reliability of the subsystem of memory taking into account the malfunctions was described. The computer model of influence of parameters of the subsystem of memory on it reliability taking into account the malfunctions was done.

ПРИМЕНЕНИЕ ГРУППОВЫХ КОДОВ В МАТРИЧНОМ ПРЕДСТАВЛЕНИИ ДЛЯ ИСПРАВЛЕНИЯ ОШИБОК В ЗАПОМИНАЮЩИХ УСТРОЙСТВАХ

Чистова Т.Н., Матвеева С.М., Иванов А.М.

Московский государственный текстильный университет им. А.Н. Косыгина

В настоящее время разработаны многие коды, способные обнаруживать и исправлять произвольное число ошибок. При многообразии корректирующих кодов затруднительно дать их законченную и точную классификацию. Однако, некоторые группы кодов можно строго классифицировать. Например, систематические коды являются кодами, в которых информационные и проверочные символы всегда расположены в строго определенных местах кодовых слов. Систематические коды являются равномерными кодами, так как все кодовые комбинации кода с заданными корректирующими способностями имеют одинаковую разрядность.

В систематических кодах проверочные символы образуются с помощью различных линейных комбинаций информационных слов. Декодирование систематических кодов так же основано на проверке линейных соотношений между символами, расположенными на определенных позициях, образующих группу проверяемых информационных символов и соответствующим проверочным символом. В случае двоичных кодов процесс декодирования кодовых слов сводится к проверке на четность, т.е. к выполнению операций суммирования по mod 2.

Систематические коды формируются при помощи производящей (образующей, порождающей) и проверочной матриц. Линейные групповые коды являются систематическими кодами. Проверочные символы линейных кодов представляют собой линейные комбинации информационных символов – для двоичных кодов эти комбинации образуются сложением по mod 2. В свою очередь, линейные коды образуют алгебраическую группу по отношению к операции сложения по mod 2 и по этой причине их называют групповыми кодами.

Одним из свойств группового кода является выполнение условия: минимальное кодовое расстояние d_0 между кодовыми словами равно минимальному весу ненулевых кодовых слов W_{min}. В свою очередь, вес кодового слова равен числу его ненулевых символов. Например, вес W кодового слова 1011 равен 3.

Кодовое расстояние между, например, двумя кодовыми словами равно весу кодовой комбинации, получаемой в результате сложения исходных кодовых слов по mod 2. Например, кодовое расстояние между кодовыми словами 1100011 и 1001111 равно:

$1100011 \oplus 1001111 = 0101100$, где $d_0=3$ и $W_{min}=d_0=3$.

Групповые коды можно задавать с помощью порождающих матриц, размерность которых определяется параметрами кода k и n-k, где k- число символов информационного слова, а (n-k)- число проверочных символов порождающей матрицы, n- число символов кодовых слов. Число строк порождающей матрицы равно k, число ее столбцов равно n.

Порождающая матрица имеет вид:

$$P=\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} & p_{11} & p_{12} & \dots & p_{1(n-k)} \\ a_{21} & a_{22} & \dots & a_{2k} & p_{21} & p_{22} & \dots & p_{2(n-k)} \\ \dots & & & & & & & \\ a_{k1} & a_{k2} & \dots & a_{kk} & p_{k1} & p_{k2} & \dots & p_{k(n-k)} \end{bmatrix},$$

Групповые коды, порождаемые такими матрицами, называются в теории кодирования (n,k)- кодами, а соответствующие им матрицы называются образующими, производящими, порождающими. Здесь P- обозначение порождающей матрицы размерности k×n, где k- число строк, n- число столбцов матрицы; P_{ij}- проверочный символ i-ой строки, j-го столбца порождающей матрицы; a_{ij}- элементы информационной части порождающей матрицы.

Производящая матрица P может быть представлена с помощью двух матриц I и H (информационной и проверочной). Число столбцов и строк матрицы I равно k, а число столбцов матрицы H равно n-k:

$$P=\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1k} \\ a_{21} & a_2 & \dots & a_{2k} \\ \dots & & & \\ a_{k1} & a_{k2} & \dots & a_{kk} \end{bmatrix} * \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1k} \\ p_{21} & p_{22} & \dots & p_{2k} \\ \dots & & & \\ p_{k1} & p_{k2} & \dots & p_{kk} \end{bmatrix} = I*H .$$

В качестве информационной матрицы I удобно использовать единичную матрицу.

Выбор проверочной матрицы производят из следующих соображений: чем больше единиц в разрядах проверочной матрицы, тем ближе порождающий групповой код к оптимальному. Критерием оптимальности таких кодов является выполнение условия:

$$2^{n-k}-1 \geq \sum_{i=1}^{3} c_n^i,$$

где n- число символов кодового слова; k- число символов информационного слова; n-k – число проверочных (контрольных) символов; η- число ошибок, исправляемых кодом.

При η=1 это условие принимает вид $2^{n-k}-1=n$.

Отсюда $2^{n-k}=n+1$ и $n-k=\log_2(n+1)$.

Например, если n=15 и k=11, то n-k=4. Итак, чтобы исправить одну ошибку в кодовых словах разрядности n, нужно использовать 4 проверочных символа.

Однако, чем больше единиц в проверочной матрице H, тем больше нужно использовать сумматоров по mod 2 в кодере и декодере, т.е. чем больше единиц в этой матрице, тем сложнее аппаратура. С другой стороны, если основным требованием к схемотехнической аппаратуре является простота, то необходимо, чтобы вес каждой строки проверочной матрицы должен быть не менее $W_H \geq d_0 - W_I$, где W_H- вес соответствующей строки проверочной матрицы H; W_I- вес соответствующей строки информационной матрицы.

И если информационная матрица является единичной, то ее вес $W_I=1$.

Нетрудно видеть, что при $W_I>1$ усложняется построение групповых кодов и их схемотехническое решение. Отсюда следует очевидность удобства использования единичной матрицы в качестве информационной матрицы I.

При выполнении этих требований любая производящая матрица может быть представлена в виде

$$P=\begin{bmatrix} b_1 & b_2 & b_3 & \dots & b_k & c_1 & c_2 & \dots & c_{(n-k)} \\ 1 & 0 & 0 & \dots & 0 & p_{11} & p_{12} & \dots & p_{1(n-k)} \\ 0 & 1 & 0 & \dots & 0 & p_{21} & p_{22} & \dots & p_{2(n-k)} \\ \dots & & & & & & & & \\ 0 & 0 & 0 & \dots & 1 & p_{k1} & p_{k2} & \dots & p_{k(n-k)} \end{bmatrix} = [IH],$$

где b_i- символы информационного слова, c_j- контрольные символы кодового слова.

Такая форма представления производящей матрицы называется левой канонической формой производящей матрицы.

Построим производящую матрицу группового кода, оптимального с точки зрения минимума контрольных разрядов при максимуме информационных разрядов, для использования этого кода в системе хранения информации с информационной емкостью 2048 информационных слов.

Решение:

1. $2^k \geq 2048$; k=11.

Для определения числа c_J контрольных разрядов с минимальным кодовым расстоянием $d_0=3$ и известным числом разрядов информационных слов k=11 используем выражение

n-k=] $\log_2$[(k+1)+] $\log_2$(k+1)[ ] [,

тогда получим, что

n-k=] $\log_2$[(11+1)+] $\log_2$(11+1)[ ] [= 4.

Обратные квадратные скобки означают, что при округлении нужно использовать ближайшее целое число.

2. Проверим условие оптимального кода для η=1: $2^{n-k}-1=n$, $2^{15-11}-1=15$.

3. Вес каждой кодовой комбинации проверочной матрицы равен H $W_H \geq d_0 - 1$; $W_H \geq 2$.

4. Число строк производящей матрицы P равно k=11. Поэтому в качестве проверочных используются все четырехразрядные двоичные кодовые комбинации с весом $W_H \geq 2$.

5. Производящая матрица группового кода, исправляющего одиночную ошибку, имеет вид:

$$P=\left[\begin{array}{cccccccccccccccc|l}
1&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1 - \text{номер строки}\\
0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&2\\
0&0&1&0&0&0&0&0&0&0&0&0&1&1&0&3\\
0&0&0&1&0&0&0&0&0&0&0&0&1&1&1&4\\
0&0&0&0&1&0&0&0&0&0&0&1&0&0&1&5\\
0&0&0&0&0&1&0&0&0&0&0&1&0&1&0&6\\
0&0&0&0&0&0&1&0&0&0&0&1&0&1&1&7\\
0&0&0&0&0&0&0&1&0&0&0&1&1&0&0&8\\
0&0&0&0&0&0&0&0&1&0&0&1&1&0&1&9\\
0&0&0&0&0&0&0&0&0&1&0&1&1&1&0&10\\
0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&11
\end{array}\right]$$

Информационная матрица — Проверочная матрица

6. Используем эту производящую матрицу для формирования кодовых слов, состоящих из информационных слов и контрольных разрядов.

Определение контрольных разрядов для конкретного информационного слова осуществляется суммирование по mod 2 тех строк проверочной матрицы, номера которых соответствуют номерам разрядов информационного слова, содержащих единицу.

Пусть информационное слово 10000000000, ему соответствуют контрольные разряды 0011 ,и тогда кодовое слово имеет вид 100000000000011. Если информационное слово 11011011101, то ему соответствуют контрольные разряды, получаемые суммированием по mod 2 тех строк проверочной матрицы, номера которых соответствуют номерам разрядов информационного слова, содержащего единицы, т.е. (0011)⊕(0101)⊕(0111)⊕(1001)⊕(1011)⊕(1100)⊕(1101)⊕(1111)=(1101)

Итак, контрольные разряды для информационного слова 11011011101 равны 1101; тогда кодовое слово имеет вид 110110111011101.Определим по очереди для каждого из этих двух кодовых слов синдром ошибки с помощью приведенной выше производящей матрицы.

Для кодового слова 100000000000011:

S_1=0⊕0*1⊕0*0⊕0*0⊕0*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0=0;

S_2=0⊕0*1⊕1*0⊕1*0⊕1*0⊕0*0⊕0*0⊕0*0⊕0*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0=0;

S_3=1⊕1*1⊕0*0⊕1*0⊕1*0⊕0*0⊕1*0⊕1*0⊕0*0⊕0*0⊕0*0⊕1*0⊕1*0⊕1*0=0;

S_4=1⊕1*1⊕1*0⊕0*0⊕1*0⊕1*0⊕0*0⊕1*0⊕0*0⊕1*0⊕0*0⊕1*0⊕1*0=0.

Следовательно, синдром ошибки этого кодового слова равен 0000. что свидетельствует об отсутствии ошибки.

Внесем ошибку в 3-й разряд, считая слева. Тогда получим кодовое слово

101000000000011.

Определим синдром ошибки для этого кодового слова:

S_1=0⊕0*1⊕0*0⊕0*1⊕0*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0=0;

S_2=0⊕0*1⊕1*0⊕1*1⊕1*0⊕0*0⊕0*0⊕0*0⊕1*0⊕1*0⊕1*0⊕1*0⊕1*0=1;

S_3=1⊕1*1⊕0*0⊕1*1⊕1*0⊕0*0⊕1*0⊕1*0⊕0*0⊕0*0⊕1*0⊕1*0⊕1*0=1;

S_4=1⊕1*1⊕1*0⊕0*1⊕1*0⊕1*0⊕0*0⊕1*0⊕0*0⊕1*0⊕0*0⊕1*0⊕1*0=0.

Итак, синдром ошибки этого слова равен 0110. Синдром соответствует третьей строке проверочной матрицы и расположенная напротив строка информационной матрицы указывает номер позиции ошибочного разряда 00100000000. Выполним аналогичные вычисления для кодового слова 1101101110101101.

S_1=1⊕0*1⊕0*1⊕0*0⊕0*1⊕1*1⊕1*0⊕1*1⊕1*1⊕1*1⊕1*0⊕1*1=0;

S_2=1⊕0*1⊕1*1⊕1*0⊕1*1⊕0*1⊕0*0⊕0*1⊕1*1⊕1*1⊕1*0⊕1*1=0;

S_3=0⊕1*1⊕0*1⊕1*0⊕1*1⊕0*1⊕1*0⊕1*1⊕0*1⊕0*1⊕1*0⊕1*1=0;

S_4=1⊕1*1⊕1*1⊕0*0⊕1*1⊕1*1⊕0*0⊕1*1⊕0*1⊕1*1⊕0*0⊕1*1=0.

Синдром ошибки для этого кодового слова равен 0000, что свидетельствует об отсутствии ошибки в кодовом слове.

Внесем ошибку в 6-й разряд кодового слова, считая слева. Тогда получим кодовое слово с ошибкой 110111111011101. Определим для него синдром ошибки:

S_1=1⊕0*1⊕0*1⊕0*0⊕0*1⊕1*1⊕1*1⊕1*1⊕1*1⊕1*1⊕1*0⊕1*1=1;
S_2=1⊕0*1⊕1*1⊕1*0⊕1*1⊕0*1⊕0*1⊕0*1⊕1*1⊕1*1⊕1*0⊕1*1=0;
S_3=0⊕1*1⊕0*1⊕1*0⊕1*1⊕0*1⊕1*1⊕1*1⊕0*1⊕0*1⊕1*0⊕1*1=1;
S_4=1⊕1*1⊕1*1⊕0*0⊕1*1⊕1*1⊕0*1⊕1*1⊕0*1⊕1*1⊕0*0⊕1*1=0.

Итак, синдром ошибки этого кодового слова равен 1010, что соответствует 6-й строке проверочной матрицы, расположенная напротив строка информационной матрицы указывает место ошибочного разряда кодового слова: 00000100000.

Аналогичным образом определяется место разряда при возникновении одиночной ошибки в других случаях.

При определении контрольных символов c_j кодовых слов использовалось соотношение

$$c_j=\sum_{i=1}^{k} P_{ij}b_i \,(\mathrm{mod}2),\ \text{при } i=1,...,k;\ j=1,...,(n-k),$$

а при вычислении символов S_j синдрома ошибки при декодировании кодовых слов применялось выражение

$$S_j=c_j \oplus \sum_{i=1}^{k} P_{ij}b_i \,(\mathrm{mod}2),$$

где i-текущий номер строки порождающей матрицы; j- текущий номер столбца проверочной матрицы H и текущий номер контрольного символа кодового слова;

b_i- i-й символ информационного слова; P_{ij}- символ проверочной матрицы H.

Произведено компьютерное моделирование процедур кодирования и декодирования информации с использованием производящей матрицы и применением языков высокого уровня.

Литература

1. Цымбал В.П. Теория информации и кодирования. –Киев: Вища школа,1977

2. Скляр Б. Цифровая связь. Теоретические основы и практическое применение. 2-е издание. –М.: Издательский дом «Вильямс», 2003

3. Чистова Т.Н., Матвеева С.М. Проектирование элементов автоматизированной системы для исследования влияния информационной избыточности на надежность передачи и хранения информации в АСОИУ. –М.: МГТУ им. А.Н. Косыгина, 2004

APPLICATION OF GROUP CODES IN MARIX FORMULATION FOR ERROR CORRECTION IN CORE STORAGE

Application of group codes in matrix formulation for error correction in code storage have been treated. Examples of its using have been given. Computer simulation of procedures of information's encoding and decoding with usage of producing matrix and higher-level languages have been executed.

———————♦———————

ЗАЩИТА ИНФОРМАЦИИ В ЗАПОМИНАЮЩИХ УСТРОЙСТВАХ С ПОМОЩЬЮ КОДОВ ЭЛАЙЕСА

Матвеева С.М., Чистова Т.Н., Иванов А.М.

Московский государственный текстильный университет им. А.Н. Косыгина

Операции кодирования в кодах Элайеса выполняются над совокупностью информационных символов, размещаемым по нескольким координатам q – мерного пространства. Часто в качестве такого пространства использовалось двумерное, в котором информационные символы располагались по строкам и столбцам матрицы. Таким образом, можно представить коды Элайеса как многомерные или многостепенные коды. Тогда число информационных символов в кодовом слове q – степенного кода определяется выражением

$$K = \sum_{i=1}^{q} ч_i, \qquad (1)$$

где $ч_i$ - число информационных символов по координате i. Последовательности информационных символов по каждой из координат кодируются каким-либо линейным кодом, например, кодом Хемминга.

Простейший код Элайеса с проверкой на четность по строкам и столбцам широко применяется на практике для обнаружения ошибок на магнитных носителях информации, причем код Элайеса может приме-

няться как для контроля отдельных информационных слов, так и для страниц информационных слов. В общем виде расположение информационных и проверочных символов изображено на рис. 1.

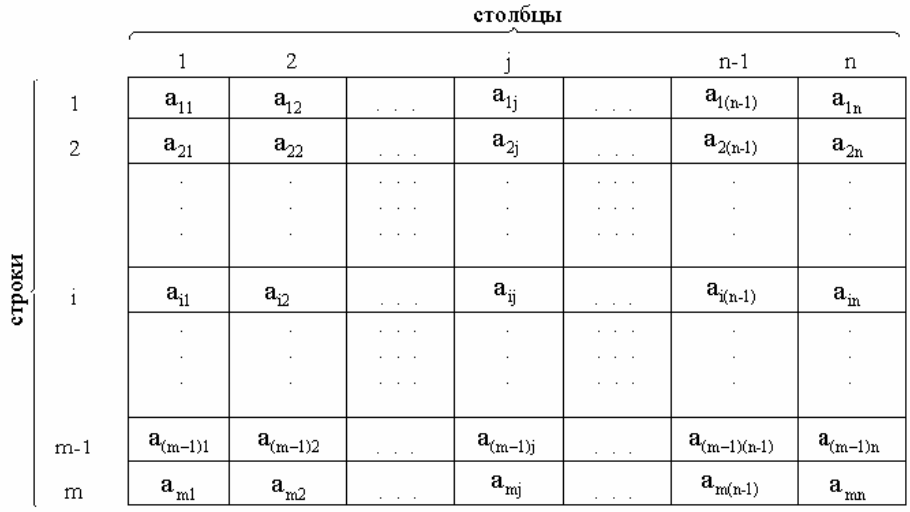


Рис.1 Расположение информационных и проверочных символов двумерного кода Элайеса.

Значения проверочных символов, расположенных в крайнем правом (или в любом другом) столбце и нижней (или в любой другой) строке определяются с помощью выражений:

$$a_{in} = \sum_{j=1}^{n-1} a_{ij} (\mathrm{mod} 2);\ \ a_{mj} = \sum_{i=1}^{m-1} a_{ij} (\mathrm{mod} 2);\ \ a_{mn} = \sum_{i=1}^{m-1} a_{in} (\mathrm{mod} 2) = \sum_{j=1}^{n-1} a_{mj} (\mathrm{mod} 2), \qquad (2)$$

где a_{ij} - символы кода Элайеса; i – текущий номер (индекс) строки; j – текущий номер (индекс) столбца; n – число столбцов матрицы; m – число строк матрицы.

Передача символов такого кода производится последовательно символ за символом от одной строки к другой, либо параллельно целыми строками. Декодирование кодового слова – блока информации, состоящего из m строк и n столбцов, начинается сразу, без ожидания поступления всего блока.

Проверка справедливости соотношений (2) при декодировании позволяет исправить любое нечетное число искаженных символов, размещенных в одной строке (или столбце). Строка (столбец) с нечетным числом ошибочных символов будет выявлена тем, что проверка соотношений (2) на четность даст отрицательный результат, а искаженные символы будут локализованы проверкой соотношений (2) по столбцам (строкам).

Необнаруживаемыми оказываются ошибки при четных искажениях символов как по строкам, так и столбцам, расположение которых образует прямоугольник.

Число четырехкратных ошибок такого вида для кодового слова – блока из $m \times n$ символов определяется с помощью выражения 3.

$$E'_{RR_4} = C_m^2 * C_n^2 = (mn(m-1)(n-1))/4 \qquad (3)$$

Примем m = 4, n = 4, тогда $E'_{RR_4} = 36$. Однако, общее число четырехкратных ошибок может быть определено следующим образом (4)

$$E_{RR_4} = C_{mn}^4 = (mn(mn-1)(mn-2)(mn-3))/4 \qquad (4)$$

Отношение числа необнаруживаемых четырехкратных ошибок к общему числу таких ошибок определяется с помощью выражения

$$E'_{RR_4} / E_{RR_4} = 6(m-1)(n-1)/((mn-1)(mn-2)(mn-3)) \qquad (5)$$

Оценим это отношение при m = 4, n = 4.

$$E'_{RR_4} / E_{RR_4} \approx 0.0197$$

Ненулевой вектор двумерного кода Элайеса с минимальным весом должен содержать только одну ненулевую вектор-строку, которая имеет минимальный вес, равный 2. Проверка на четность каждого из ненулевых столбцов двумерного кода Элайеса также дает только один ненулевой вектор-столбец с минимальным весом, равным 2. В связи с этим, минимальный вес ненулевого вектора кода Элайеса с двойной проверкой на четность будет равен $2 \times 2 = 4$.

В общем случае минимальный вес вектора кода Элайеса равен произведению минимальных весов векторов обрабатываемых кодов. Отсюда следует, что так как минимальное кодовое расстояние для линейного кода равно минимальному весу его ненулевых векторов, то минимальное кодовое расстояние кода Элайеса равно

$$d_{min} = \prod_{i=1}^{q} d_i ,$$

где d_i - минимальное кодовое расстояние линейного кода по координате i; q – число координат.

Исправление ошибок кодового блока с помощью кода Элайеса осуществляется последовательно, но сначала корректируются ошибки по одной координате, затем исправляют оставшиеся ошибки по другой координате и т.д. Такая процедура декодирования достаточно проста как при аппаратной, так и программной реализации. Однако, при этом оказывается невозможным исправить часть ошибок кратности $(d_{min} - 1)/2$.

Далее рассмотрен принцип совместного использования кодов Элайеса и Хемминга для исправления ошибок в запоминающих устройствах.

Литература

1. Хетагуров Я.А., Руднев Ю.П. Повышение надежности цифровых устройств методами избыточного кодирования. – М.: Энергия, 1974
2. Дмитриев В.И. Прикладная теория информации. – М.: Высш. шк., 1989
3. Скляр Б. Цифровая связь. Теоретические основы и практическое применение. 2 - е издание. – М.: Издательский дом «Вильямс», 2003
4. Матвеева С.М. Итеративные коды и их использование для обнаружения ошибок при передаче и хранении информации. – М.: МГТУ им. А.Н. Косыгина, 2004

INFORMATION GUARDING WITH USING OF ELIESE'S CODES IN CODE STORAGE

Information guarding with using of Eliese's codes in code storage have been studied. The encryption operations in Eliese's codes are executed by collection of information symbols, which are arranged on several coordinates of q – dimensional space. The easiest Eliese's code with lines and columns even parity is widely used on practice for error detection in matrix carrier.

ИСПОЛЬЗОВАНИЕ КОРРЕКТИРУЮЩИХ КОДОВ ДЛЯ ПОВЫШЕНИЯ НАДЕЖНОСТИ ОПЕРАТИВНЫХ ЗАПОМИНАЮЩИХ УСТРОЙСВТВ

Карабанова М.В.

Московский Государственный Текстильный Университет имени А.Н. Косыгина

Надежность ОЗУ на динамических БИС ЗУ (ДБИС ЗУ) в значительной степени определяется надежностью микросхем памяти. Это объясняется тем, что число ДБИС ЗУ составляет большую часть (55 – 80%) в общем оборудовании ОЗУ и их надежность может быть ниже надежности остальных комплектующих изделий, входящих в состав ОЗУ.

Нарушение функционирования БИС ЗУ имеет постоянный или кратковременный характер. Причиной постоянной неисправности, является отказ – необратимый физический дефект БИС ЗУ. Отказ приводит к возникновению неисправности в ОЗУ, которая устраняется только при ремонте запоминающего устройства. Кратковременное нарушение функционирования БИС ЗУ возникает вследствие сбоев. Сбой ДБИС ЗУ, в отличие от отказа, вызывает временное изменение логического состояния отдельных бит, правильное состояние которых восстанавливается при записи истинных данных. Причиной сбоев ДБИС *ЗУ* являются помехи в цепях электропитания, деградация параметров микросхем памяти и альфа-частицы. Специфическая особенность динамических БИС ЗУ – их повышенная чувствительность к воздействию альфа-частиц, образующихся при распаде радиоактивных материалов, которые содержатся в составе корпусов интегральных схем. Альфа-частицы, проникающие в кристалл, генерируют электронно-дырочные пары, которые при определенных условиях могут разрушить заряд, хранящийся на запоминающей емкости элемента памяти, и соответственно явиться причиной возникновения случайного сбоя ДБИС ЗУ. При сбое из-за альфа-частиц нарушается, как правило, функционирование отдельного бита ДБИС ЗУ. Чтобы уменьшить влияние альфа-частиц, разрабатываются новые схемотехнические решения и технологические процессы.

Исследования показали, что значение интенсивности отказов ДБИС ЗУ различной информационной емкости составляет на этапе эксплуатации $2 \cdot 10^{-5} – 2 \cdot 10^{-7}$ $ч^{-1}$. Интенсивность сбоев из-за альфа-частиц может быть в 10 – 1000 раз выше этого значения. Интенсивность отказов ДБИС ЗУ при поставке изготовителем обычно выше эксплуатационного значения интенсивности отказов микросхем памяти. Для снижения начальной интенсивности отказов ДБИС ЗУ, а также устранения неисправностей, обусловленных скрытыми дефектами других комплектующих изделий, входящих в состав ЗУ, при изготовлении ОЗУ производится термотренировка ОЗУ. Это требует, однако, разработки специального технологического оборудования и выбора оптимальных режимов приработки устройства.

Для ДБИС ЗУ характерно наличие относительно большой доли отказов отдельных бит. Менее вероятны отказы отдельных строк и столбцов элементов памяти, полные и смешанные отказы ДБИС ЗУ. Для ОЗУ большой емкости, содержащих до $10^3 – 10^4$ микросхем памяти и более, среднее время возникновения ошибки может иметь недопустимо малое значение. Это вызывает необходимость применения в таких устройствах, различных средств повышения надежности.

Повышение уровня интеграции БИС ЗУ обеспечивается в основном за счет сокращения размеров ЭП, повышения чувствительности усилителей. Так, в динамических ЗУПВ в качестве ЭП используется все меньшая емкость, а в статических ЗУПВ уменьшаются перепады напряжения. Дальнейшее повышение плотности упаковки приводит к росту числа несистематических, произвольных сбоев в работе БИС ЗУ, вызванных не физическими дефектами в схеме. Предполагалось, что причиной этих сбоев являются помехи в системах, работа при предельных допусках напряжений, недостаточная чувствительность усилителей или зависимость от типа тестовых последовательностей. Затем было установлено, что основную роль играет повышенная чувствительность БИС к облучению светом или радиацией. Это особенно характерно для динамических ЗУ, принцип хранения информации в которых основан на накоплении заряда на запоминающей емкости.

Обнаружение ошибок – это процесс выявления сигналов, отличающихся от тех, которые должны иметь место при нормальной работе цифровой вычислительной машины. Неисправностью считается искажение или отклонение от нормы одного или нескольких параметров сигнала. Ошибка – это выходной сигнал какой-либо схемы, логически отличающийся от нормального. Не всякая неисправность приводит к ошибке: некоторые неисправности вызовут ошибки только при определенных последовательностях входных сигналов, подаваемых на блок ЦВМ, содержащий неисправный элемент.

Ошибки в оперативной памяти возникают из-за возникновения разных неисправностей в различных функциональных частях запоминающего устройства: отказы и сбои запоминающих элементов, неисправности разрядных цепей (обрыв или замыкание). Отказы и сбои в подсистемах оперативной памяти в происходят в связи с наличием дефектов в БИС ЗУ, которые были допущены в процессе изготовления БИС (врожденные дефекты) или появились в ходе эксплуатации схем (приобретенные дефекты) под воздействием агрессивной окружающей среды (действие высоких температур, радиоактивное излучение) или неправильной эксплуатации.

Система обнаружения и исправления ошибок в оперативной памяти позволяет избежать потери данных при передаче и хранении информации.

Решить проблему обеспечения достоверности воспроизведения информации за счет улучшения свойств носителя, условий хранения и эксплуатации, создания более совершенных конструкций ЗУ достаточно сложно из-за больших экономических затрат. Реальный способ обеспечения достоверности – применение корректирующих кодов, служащих для обнаружения и коррекции ошибок, возникающих при работе ЗУ.

Обеспечение требуемой достоверности при передаче цифровой информации по каналам связи невозможно без применения избыточного кодирования. Для защиты запоминающих устройств от ошибок используют различные корректирующие коды. Эти коды предназначены для надежного хранения информации в памяти с дефектными элементами памяти.

В настоящее время разработано сравнительно большое количество корректирующих кодов, основанных на различных разделах математического аппарата: теории групп, коммутативной алгебры, теории конечных полей. Выбор кода и способа его реализации зависит от назначения и типа ЗУ. Один и тот же код, исправляющий ошибки или дефекты, может применяться как для повышения надежности БИС ЗУ, так и систем памяти на БИС ЗУ.

Циклические коды предназначены для обеспечения надежного хранения информации в памяти с дефектными ЭП. Считается, что местоположение, таких ЭП известно или может быть определено с помощью специальных измерений.

В настоящее время существует много типов запоминающих устройств, которые используются в цифровых системах, включая запоминающие устройства на магнитных сердечниках, тонких пленках и магнитных лентах; дисковые запоминающие устройства; устройства на магнитных барабанах и линиях задержки; фотографические запоминающие устройства и постоянные запоминающие устройства.

Обнаружение ошибок в запоминающем устройстве включает в себя обнаружение того, что

1) данные, записываемые в ЗУ, правильны;
2) данные, взятые из памяти, не имеют ошибок;
3) обращение к памяти делается по правильному адресу;
4) вся аппаратура запоминающего устройства работает правильно.

В одних запоминающих устройствах вся проверка может осуществляться одновременно, в других необходимы различные схемы обнаружения для разных классов ошибок.

Когда в считываемой информации обнаруживают ошибку, то необходима какая-либо программа восстановления или способ исправления информации, полученной из памяти, с ошибкой. Возможно, иногда выгодно применять схемы автоматического исправления ошибок, которые обычно основываются на кодах с исправлением ошибок.

Многообразие существующих методов борьбы с ошибками можно разделить на два класса:
1) синтез избыточных устройств, нечувствительных к определенному количеству неисправностей;
2) синтез системы обнаружения ошибок (системы контроля).

Система обнаружения ошибок позволяет определить тип ошибки (систематическая или случайная). В этом случае исправление систематической ошибки производится с помощью реконфигурации системы, т.е. подключением резерва, и устранением неисправности, порождающей ошибку. Исправление случайных ошибок обычно производится программными методами (повторением участка программы или отдельных операций, при выполнении которых произошла ошибка).

Методы борьбы с ошибками в цифровых системах (ЦС) оцениваются по степени их влияния на следующие характеристики системы: показатели надежности, производительность или быстродействие, количество аппаратуры.

Количественные характеристики надежности системы определяются ее назначением и спецификой эксплуатации. Из характеристик надежности для ЦС наиболее часто используются следующие: коэффициент готовности, вероятность безотказной работы за заданное время, наработка на отказ, также оценивается достоверность информации, получаемой на выходе системы.

Основным средством обеспечения высокой помехоустойчивости сложной системы является введение избыточности, необходимой для обнаружения и исправления ошибок, возникающих при работе системы и ее элементов. Теоретической базой эффективного использования вводимой избыточности является теория помехоустойчивого кодирования.

Недопустимость ошибок в вычислительных системах, а в некоторых случаях критическая природа самих данных требуют использования оборудования, исключающего ошибки, или какого-либо рода кодов, исправляющих или обнаруживающих ошибки на выходе конечных устройств. Создано несколько классов длинных мощных кодов. Для некоторых из этих классов кодов разработаны процедуры декодирования, которые не требуют большого количества оборудования.

Одним из способов повышения надежности изделий является информационное резервирование (избыточность). Обеспечение требуемой достоверности при передаче цифровой информации по каналам связи невозможно без применения избыточного кодирования. Для этой цели в основном применяется два типа корректирующих кодов – циклические и итеративные с проверкой количества единиц на четность (нечетность).

При построении корректирующих (помехоустойчивых) кодов к информационному слову добавляется контрольные (проверочные) разряды. Эти разряды не несут информацию, но они могут сказать, произошла ошибка или нет. Существует множество разновидностей корректирующих кодов, например: коды с повторением, код с общей проверкой на четность, циклические коды, линейные коды.

К циклическим кодам относятся коды: Рида – Маллера, Боуза – Чоудхури – Хонквингема (коды БЧХ), Рида – Соломона, Файра, Питерсона – Горенстейна – Циллера и другие.

Циклические коды являются важнейшим частным видом линейных групповых кодов, которые допускают сравнительно простую техническую реализацию с помощью регистров сдвига с обратными связями. В этих кодах часть комбинаций кода или все комбинации могут быть получены путем циклического сдвига одной или нескольких комбинаций кода. Циклический сдвиг осуществляется справа налево, причем крайний левый символ переносится каждый раз в конец комбинации.

Циклический код – циклическим называется такой линейный код, в котором любой циклический сдвиг разрешенного кодового слова также является кодовым словом. Циклические коды бывают всевозможных алфавитов, но чаще рассматриваются двоичные коды.

В циклических кодах кодовые комбинации представляются в виде многочленов, что позволяет свести действия над кодовыми комбинациями к действиям над многочленами (используя аппарат полиномиальной алгебры). Циклические коды все относятся к систематическим кодам и поэтому обладают всеми их свойствами. В них контрольные и информационные разряды расположены на строго определенных местах. Кроме того, они относятся к числу блочных кодов, каждый блок (одна буква является частным случаем блока) кодируется самостоятельно, линейным, корректирующим и равномерным кодам. Циклические коды ис-

пользуются в ЭВМ при последовательной передаче данных. Первоначально они были созданы для упрощения схем кодирования и декодирования. Их эффективность при обнаружении и исправлении ошибок обеспечила им широкое применение на практике.

Идея построения циклических кодов базируется на использовании неприводимых многочленов. Неприводимым называется многочлен, который не может быть представлен в виде произведения многочленов низших степеней, т.е. такой многочлен делиться только на самого себя или на единицу и не делиться ни на какой другой многочлен. На такой многочлен делиться без остатка двучлен $x^n + 1$. Идея коррекции ошибок в циклических кодах базируется на том, что разрешенные комбинации кода делятся без остатка на некоторый образующий многочлен, который выбирается из числа неприводимых многочленов. Чтобы обнаружить ошибку при делении на выбранный многочлен, надо чтобы все комбинации кода не делились ни на какой другой многочлен, а для этого необходимо, чтобы выбранный многочлен не разлагался на другие многочлены, т.е. был простым неприводимым многочленом. Неприводимые многочлены в теории циклических кодов играют роль образующих полиномов.

Чтобы понять принцип построения циклического кода, умножаем комбинацию простого *k*-значного кода *Q(x)* на одночлен x^r, а затем делим на образующий полином *P(x)*, степень которого равна *r*. В результате умножения *Q(x)* на x^r степень каждого одночлена, входящего в *Q(x)*, повышается на *r*. При делении произведения $x^rQ(x)$ на образующий полином получается частное *C(x)* такой же степени, как и *Q(x)*.Результат можно представить в виде:

$$\frac{Q(x)x^r}{P(x)} = C(x) + \frac{R(x)}{P(x)}, \qquad (1)$$

где *R(x)* - остаток от деления $Q(x)x^r$ на *P(x)*.

Частное *C(x)* имеет такую же степень, как и кодовая комбинация *Q(x)* простого кода, поэтому *C(x)* является кодовой комбинацией этого же простого *k*-значного кода. Следует заметить, что степень остатка не может быть больше степени образующего полинома, т.е. его наивысшая степень может быть равна *(r-1)*. Следовательно, наибольшее число разрядов остатка *R(x)* не превышает числа *r*.

Умножая обе части равенства (1) на *P(x)* и, произведя некоторые перестановки (так как $1 \oplus 1=0$, а значит $-1=1$), получаем:

$$F(x) = C(x)P(x) = Q(x)x^r + R(x) \qquad (2)$$

Таким образом, кодовая комбинация циклического *n*-значного кода может быть получена двумя способами:

1) умножение кодовой комбинации *Q(x)* простого кода на одночлен x^r и добавление к этому произведению остатка *R(x)*, полученного в результате деления произведения $Q(x)\,x^r$ на образующий полином *P(x)*;

2) умножения кодовой комбинации *C(x)* простого *k*-значного кода на образующий полином *P(x)*.

При построении циклических кодов первым способом расположение информационных символов во всех комбинациях строго упорядочено – они занимают *k* старших разрядов комбинации, а остальные *(n-k)* разрядов отводятся под контрольные.

При втором способе образования циклических кодов информационные и контрольные символы в комбинациях циклического кода не отделены друг от друга, что затрудняет процесс декодирования.

Циклические коды, как и любые другие корректирующие коды, применяются для устранения случайных ошибок, возникающих из-за сбоя в оборудовании. В то время как систематические ошибки исправляются другими способами (не программно). Для оптимизации использования кодов требуется определенный выбор кода, от чего будут зависеть характеристики системы, в которой используется данный код. Например, если известно, что точность системы очень высока, то вряд ли стоит использовать код, исправляющий более 2000 ошибок. Для обнаружения единичных и немногочисленных ошибок используется наиболее часто код Хэмминга, для исправления же вспышек ошибок используется, например, код Файра. Специализированные корректирующие циклические коды используются в основном на точной технике и в системах, где очень важна точность и минимальное содержание ошибок.

Разнообразие подходов создания циклических кодов позволяет охватить наибольший спектр узких проблем, где использование других кодов либо невозможно, либо связано со слишком большими потерями в качестве работы системы и в скорости обработки информации.

Литература

1. Цымбал В.П. Теория информации и кодирование – Киев: Вища школа, 1977
2. Питерсон, Уэлдон Э. Коды исправляющие ошибки – М.: Мир, 1976
3. Конопелько В.К., Лосев В.В. Надёжное хранение информации в полупроводниковых ЗУ – М.: Радио и связь, 1986
4. Хетагуров Я.А., Руднев Ю.П. Повышение надёжности цифровых устройств методами избыточного кодирования – М.: Энергия,1974

The reasons of occurrence of mistakes in operative memory. Refusals and failures in subsystems of operative memory. The system of detection and correction of mistakes in the operative memory, allowing to avoid loss of the data by transfer and storage of the information. Maintenance of required reliability by transfer of the digital information on liaison channels with application of superfluous coding. The adjusting codes used for reliable storage of the information in memory with defective units of memory.

БАЗОВЫЙ АППАРАТНЫЙ И БАЗОВЫЙ ФУНКЦИОНАЛЬНЫЙ ПРОФИЛИ КАК СЕТЕВЫЕ РЕШЕНИЯ ТЕХНОЛОГИЙ ОТКРЫТЫХ ВЫЧИСЛЕНИЙ ПРИ ПРОЕКТИРОВАНИИ ОТКРЫТОЙ ПАМЯТИ

Бадекин Б.И.

Московский инженерно-физический институт /государственный университет/

Ставится вопрос о месте технологий открытых вычислений в структуре сетевых приложений, построенных по технологии открытых систем. Вопрос не праздный, т.к. точность позиционирования этого профиля вычислений будет определять развитие и физической платформы форм носителей информации, и прикладной платформы в специфике собственного операционного и функционального профиля. Поскольку открытые вычисления всегда не точны за счет существующей постоянной возможности избыточности, связанной с добавлением лишних символов в сообщение или в блок данных, то о полноте функционального профиля сетевых приложений можно говорить лишь с позиций коллапсного функционального профиля. Полный функциональный профиль DNA был разработан корпорацией DEC в 1974 – 1975 годах. Она предложила множество стандартов и сетевых служб для всех семи уровней среды сетевого проектирования. Коллапсный функциональный профиль имеет набор задач достаточный для того, чтобы отсутствующие несколько сетевых уровней полностью определялись упрощенными функциями оставшихся уровней. Наблюдается явное упрощение технологий, увеличивающих быстродействие и простоту исполнения в данном случае локальных сетей за счет упрощения ряда функций области взаимодействия. Обычно процессу упрощения подвергаются возможности представления и передачи данных /сетевой и транспортный уровни/ и организация сеансов соединений /сеансовый и представительский уровни/. По аналогии с процессами, действующими в пространственно-распределенных сетях на принципе коллапсирующего функционального профиля, открытые вычисления точно также проявляют себя во времени на устройствах, которые служат источниками открытых вычислений. Случайные воздействия извне или изнутри системы, оказывающие влияния в сетях через считывающие устройства, через коммуникационную сеть, обеспечивающую передачу данных, изменяют содержание и формат блока или пакета данных в каналах связи в процессе передачи. Для сетевых приложений – это борьба с шумами, исправления нарушений и сбоев в работе устройств, проверка кодов на соответствие; для открытой вычислимости – это не полное описание функционирования способа открытых вычислений. Открытую передачу данных в общем звене прорабатывает транспортная платформа и вопрос об открытости вычислений для этой платформы также проблематичен. Говоря об архитектуре абонентского звена как поставщика открытых вычислений, выделяют известные модели по виду взаимосвязи компонентов: терминал – главный компьютер, клиент – сервер и одноранговая архитектура. Можно говорить о логической, функциональной и физической организации технических средств сети, которые могут быть задействованы в среде открытых вычислений. Погружение в архитектуру терминал – главный компьютер и др. выделяет компьютер как производящий открытый вычислитель, который для информационной сети представляет неопределенность в силу специфики способа вычислений и открытых определяющих методов производства данных. В этой ситуации нельзя говорить о техническом и программном обеспечении методов кодирования, архитектуре безопасности данных, архитектуре взаимодействия в информационной сети в схеме полного функционального профиля, т.к. любая концепция управления данными определяется категорией открытости. Можно говорить о базовой эталонной модели взаимодействия открытых систем, но при этом иметь в виду, что область взаимодействия, располагаемая между прикладными процессами и физическими средствами соединений, подвергается во времени произвольному проникновению открытых сигналов по всем семи уровням. Разрабатываемые решения в структуре полного функционального профиля для дискретной информационной сети поддерживают решения полностью устраняющие возникающие ошибки в способе передачи данных от уровня шумов до отдельных сбоев между соседними системами. Для открытых вычислений наличие встраиваемого сигнала в систему /шума, сбоя при передачи сообщения, открытого извне сигнала влияния и др./ является способом существования открытых вычислений, а не явлением, с которым надо бороться. Этот сигнал согласно архитектуры дискретной сети DNA может проявлять себя на любом сетевом уровне и нарушать работу известных способов передачи данных. При этом протоколы сетевого сер-

виса формируют логические каналы через коммуникационную сеть, управляющие потоками данных, выполняющих функции сегментирования и сборки блоков данных и сквозной контроль ошибок. Далее, протокол сетевого и транспортного уровня согласно Международной организации по стандартизации /ISO, 1977 год/ и, известный протокол Протокол управления передачей/Межсетевой протокол /TCP/IP/ или сетевой пакетный аналог IPX/SPX поддерживают передачу данных транспортной платформы с учетом исправления ошибок в широком списке нарушений и сбоев. Также и для других сетевых служб уровня абонентских и административных систем сразу отметим, что в известных архитектурах сетей всякое проникновение в сеть является нарушением ее функциональности. Более того, и базовая эталонная модель взаимодействия открытых систем /ISO, 1979 год/, и среда взаимодействия открытых систем в период разработки международных стандартов, раскрывающих понятие "открытая система", ни Международной организацией по стандартизации /ISO/, ни Международным союзом электросвязи /ITU/ не предусмотрели принципы организации открытых вычислений в формате открытого проникновения сигнала в открытую архитектуру как для аппаратной платформы соединений, так и для операционной, и прикладной платформ. Открытая сетевая архитектура ONA поддерживает средства тестирования на уровне физических средств соединений /это сигналы конформности изделий, сети коммутации пакетов, дискретной сети с интегральным сервисом /ISDN/, локальной сети и др./, а на верхнем уровне /прикладном/ ONA задействует три области передачи данных: передачу сообщений, диалог, обработка данных /в формате кадра, ячейки, пакета, блока данных и отдельных символов/. Промежуточные сетевые уровни /транспортный, сеансовый и представительский/ организованы по типу общего звена в межсоединениях. Полный функциональный профиль ONA не предусматривает обработку вспомогательной среды средствами выделенных частей сетевой архитектуры на принципе открытого ввода: открытый ввод понимается как шум, сбой в работе устройств и устраняется по типу фильтрации. В этом смысле открытая сетевая обработка данных ONC, предложенная как базовый функциональный профиль, до сих пор не имеет коллапсного обоснования не только для прикладной платформы, но и для аппаратной платформы не только по типам соединений, но и по типам профилей базовой аппаратной платформы. /У нас это принято понимать как профили общего или отраслевого назначения./ Протоколы ONC как сетевые службы гетерогенных сетей созданы для прикладной платформы, включающей три последних уровня: сеансовый, представительский и прикладной. Независимость от типов операционных систем определяет универсальный характер интерфейса транспортного уровня при передаче пакетов и блоков данных, а принципы, заложенные в ONC, создают информационные сети с распределенной обработкой данных. Мы наблюдаем широкое распространение баз данных в архитектуре открытых сетей, но без свойства открытых вычислений.

Перестройка архитектуры по типу открытых продуктов /продукты, производимые различными фирмами/, применяемых в абонентских системах, предлагает различные модели вычислений в среде открытой системы OSE - это модель взаимодействия различных операционных систем со стандартными прикладными программами. Новация в сторону открытых вычислений заключается в том, что в модели используются два типа элементов: платформа внешней сферы и прикладная платформа. Платформа внешней сферы моделируется элементами системы на уровне интерфейса внешней среды и определяет коммуникационный, информационный и сервис интерфейса пользователя, которые стыкуются с прикладными программами и внешними устройствами. Здесь также явление шума и сбоев как среда открытых вычислений рассматривается как негативное явление, т.к. протокол передачи данных, определяющий формат данных, не предусматривает на уровне интерфейса прикладных программ API, а также и на уровне интерфейса внешней среды /сферы/ при переносе прикладных программ из одной открытой системы в другую возможности параллельного рассмотрения и обработки этой информации как модели открытого вычислителя. Идея рассматривать систему реального времени как модель открытого вычислителя появится позже.

Период середины 80-х годов характеризуется широким уделением внимания, обращенным к понятию "открытая система". Появляется много аналогичных определений типа: архитектура открытых сценариев, открытый продукт, открытая обработка документов, открытая программная среда, открытый документ, открытая сетевая архитектура /ONA/, открытая сетевая обработка данных /ONC/, среда открытой системы /OSE/, среда открытых сообщений /OME – Open Messaging Environment/, технологии OpenDoc и др. [1] /некоторые были рассмотрены выше/, но среда открытых вычислений не определяет себя как способ вычислений в новой модальности и, соответственно, в собственном формате открытых приложений.

Для прикладной платформы базовая среда открытых вычислений должна обеспечивать совместимость сред:

1) внутренней вычислимости и внешних сфер приложения при условии, что среда параллельных вычислений должна одновременно прорабатываться в частотных диапазонах, превышающих спектр зашумления. Это объясняется тем, что совместимость относительно одной линейки включений содержит зависимые и независимые узлы недостижимости, размерность которых может быть неопределенной.

2) в условиях распараллеливания процессов вычисления осуществлять открытый ввод всего спектра информации в топологическом дизайне аппаратного парка, а быть работоспособными в этих условиях означа-

ет, что по окончании процесса распараллеливания система должна продолжать поддерживать основной тактовый режим работы сети, блока доступа, компьютера, приложений и т.д.

3) поддерживать в работоспособном состоянии режим релаксации данных в сети на принципах расчета резонансов в системе по типу расчета сети как grid-модели /grid-сети, метод расчета сетей на принципе бизнес-технологий, бизнес-приложений/, последней мили, включений опосредованных вычислений и др. Опосредованные вычисления прорабатываются на включения и определяются топологией дизайна устройств, материала изготовления и режимов тестирования оборудования.

4) нарабатывать статистики работоспособности и надежности открытых вычислений по включающему принципу типа "традиционный on-line".

Не рассматривая здесь специфические условия работы транспортной платформы, которая в условиях открытых вычислений должна быть прозрачной для способа передачи данных через коммуникационную сеть, остановимся на аппаратной платформе, определяющей возможности применения открытой информации по всему ассимилируемому диапазону частот на различных типах устройств. Особенности работы базового функционального профиля непосредственно в условиях открытых вычислений должны опираться на возможность записи всего спектра информации основной /внутри тактируемой/ и вспомогательной, внешней сфер в память, где функция защиты памяти носит условный характер и поэтому условие прозрачности для данной платформы не действует. Мы начинаем затрагивать собственно вопрос, указанный в названии доклада: вопрос взаимодействия при проектировании открытой памяти. При открытых вычислениях на аппаратной платформе в условиях непрозрачности измеримых свойств объекта, а именно, снятие возможностей по сохранению записи данных с учетом способа реализации этих возможностей, возникает ситуация снимающая приоритет значения записанного в область памяти в силу тех процессов, что происходят во внешней сфере и изменяют запись в некоторой части области памяти. Это принципиальные отличия от способа вычислений в условиях сохранения значений памяти, которые присущи известному способу ведения протокола записи/считывания в область памяти. Аппаратная платформа открытых вычислений строится с учетом этих требований: изменяться под действием внешнего потока информационных сфер /сред/ и фиксировать на себе /в себе/ эти изменения в условиях всех шкал изменения воздействия со стороны внешних сфер /сред/ или со стороны теста. Данная ситуация в открытых вычислениях понимается как непрозрачность по формату записи/считывания, ввода/вывода, что принципиально отличает ее от характера процедур известных функциональных и операционных приложений.

Возникает естественное требование ввести классифицирующий элемент в способы организации аппаратной платформы, ориентируемой на открытые вычисления. Издательство Wiley /США/ с конца 80-х годов проводит широкое обсуждение вопроса нейросетевых приложений в вычислительных сетях. Способы вычислений в этом случае, основанные на нейрокомпьютинге в режиме реального времени, понимаются как основа для создания машин 6-го поколения [2,3,4]. По типу порождающей модальности □ данный тип абстрактных машин отвечает подмножеству машин Тьюринга и известен как машина Колмогорова, всякий раз задействующая по позиционированию и сдвигу информации активно выделенные области носителей. Можно показать, что шестое поколение машин включает в аппаратную платформу свойства подструктур модальности ◊ /в частности, внешней по отношению к модальности □/ как гарантирующей признак прямых непрозрачных методов ввода и вывода информации как для базового функционального профиля, так и для аппаратного. В этом случае открытая память как элемент позиционирования информации имеет классификацию 6.0; аппаратные элементы транспортной платформы при той же внутренней тактируемости уже классифицируются по стандартам как 6.1. Полное отключение базового функционального профиля прикладной платформы в машинах 5-го поколения, в данном случае понимается как классификация 6.6, поддерживающая только протокол приложения внешних сфер /вирус – одно из этих явлений/. Этим объясняется, что полное совпадение двух машин вида 5-го поколения и такого же вида 6-го поколения ЭВМ по типу внутренней конфигурации /активно выделенной области во внутренней памяти как в машине Колмогорова/ может приводить к несовпадению этих конфигураций при выполнении одной и той же программы по истечению некоторого малого количества тактов.

Литература

1. Архитектура, протоколы и тестирование открытых информационных сетей, Толковый словарь. /В.Ф. Баумгарт, С.П. Волкова, А.В. Гнедовский и др.: Под ред. Э.А. Якубайтиса. – М.: Финансы и статистика, 1989, – 192 с.

2. Бадекин Б.И. Развитие и применение открытых систем. III Международная конференция , Москва, 22-26 апреля 1996 г. "Открытая память и открытый процессор на примере машины Тьюринга в открытых системах", с. 221-225.

3. Бадекин Б.И. "Способ записи/считывания информации в полупроводниковых устройствах с использованием /блока/ открытого сигнала". Описание изобретения к заявке. Комитет Российской Федерации по патентам и товарным знакам. RU 94009040 A1. Дата подачи 15.03. 1994. Дата публикации БИ №6 от 27.02.1996.

4. Soucek B. "Neural and concurrent real-time systems: the sixth generation". N.J., Wiley, 1989, XVIII, p. 387.

СТОХАСТИЧЕСКИЕ МЕТОДЫ ЗАЩИТЫ ПАМЯТИ

Прилуцкий С.О., Иванов М.А

Московский инженерно-физический институт (государственный университет)

Стохастическими методами принято называть методы защиты, основанные на использовании генераторов псевдослучайных последовательностей (ПСП) [1]. Можно упомянуть, например, стохастические коды С.А. Осмоловского [2], позволяющие решать все задачи обеспечения безопасности, возникающие при передаче информации по каналу связи: защиты от помех; обеспечения секретности информации; обеспечения аутентичности (целостности, подлинности) информации.

Стохастический метод внесения неопределенности (рандомизации) в работу объектов и средств защиты является одним из немногих методов защиты от разрушающих программных воздействий (РПВ) (компьютерных вирусов (КВ), сетевых червей, троянских программ, а также эксплойтов, использующих уязвимости ПО), при реализации которых защищающаяся сторона имеет преимущество перед нападающей, так как последняя «не понимает» поведения атакуемого объекта. Уже существуют и используются методы внесения неопределенности в последовательность выполнения отдельных шагов алгоритма (пермутация); методы внесения неопределенности в длительность выполнения отдельных шагов алгоритма (для защиты от так называемых временных атак); методы внесения неопределенности в результат преобразования или выполнения программы (вероятностное шифрование); методы внесения неопределенности в механизм функционирования (полиморфизм и метаморфизм) [3, 4].

Стохастические методы активно используются разработчиками РПВ для затруднения их обнаружения и нейтрализации. Можно упомянуть пермутирующие, полиморфные и метаморфные КВ, полиморфные эксплойты [3]. Технологии создания РПВ и средств защиты от них развиваются параллельно. По мере появления новых типов РПВ с некоторым опозданием (к сожалению) совершенствуются и методы защиты от них.

В настоящей работе рассматриваются стохастические методы защиты памяти.

1. Рандомизация последовательности адресов

Внесение неопределенности в последовательность извлечения слов команд из памяти требует замены n-разрядного счетчика команд на генератор ПСП с числом состояний 2^n. В обычной системе инструкции расположены последовательно, за исключением ситуаций, связанных с ветвлениями в программе (которые как раз и являются самыми тяжелыми для анализа объектами в исполняемом коде), счетчик команд также меняется вполне предсказуемо. Анализируя код, последовательно расположенный в памяти, атакующий может легко получить любую часть кода, так многие эксплойты построены на основе предсказания местоположения нужного кода или данных в памяти.

На рис. 1 показан ход выполнения программы без использования рандомизации (а) и при ее использовании (б). На рис. 1, в показан пример 8-разрядного стохастического счетчика команд.

Защищенный таким образом буфер с кодом можно отлаживать, трассировать, но автоматический пассивный анализ без применения этих методов, т.е. без конкретно исполнения, будет либо невозможен, либо требовать криптографического исследования. При использовании генератора ПСП, последовательность переключения которого зависит от секретного ключа, без знания ключа буфер с кодом будет совершенно бесполезен для анализа, и, соответственно, атака станет невозможной.

Дальнейшее развитие рассмотренного метода приводит к идеям стеганографической защиты исполняемого кода и к идее стохастической вычислительной машины.

2. Стеганографическая защита исполняемого кода

В теле программы-контейнера, содержащей достаточно большое число кодов различных команд, можно спрятать практически произвольное количество кодов других программ (рис. 1, г). При этом для извлечения соответствующей скрытой программы требуется лишь задать нужную последовательность адресов, по которым находятся коды команд этой программы.

3. Идея стохастической вычислительной машины

В памяти такой машины хранятся коды всех команд процессора. Конкретная исполняемая программа появляется лишь в момент ее запуска на выполнение с ключом, задающим логику работы генератора ПСП, последовательные состояния которого определяют порядок выполнения команд.

4. Рандомизация среды исполнения программы

На рис. 2 показан метод рандомизации тракта передачи данных «память команд - процессор», направленный против удаленных атак на программные системы, основанных на вставке кода (code injection). Метод основан на использовании обратимого преобразования XOR при передаче данных. Таким образом, для

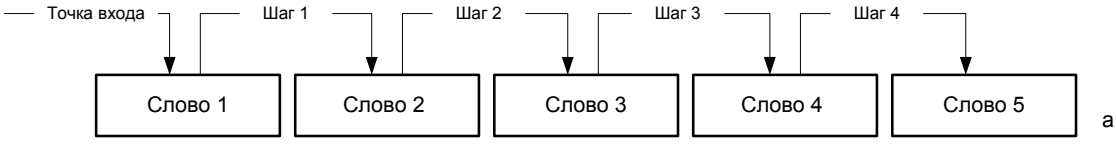
каждого процесса, выполняющегося в рамках одной системы, создается уникальная среда исполнения. В результате атакующий не понимает используемый «язык» и не может «разговаривать» с машиной.

5. Рандомизация доступа к данным

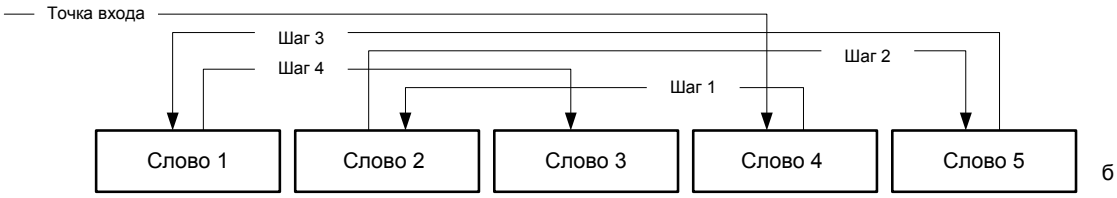
Логично попытаться рандомизировать доступ к данным и их расположение в памяти. Эта идея уже была озвучена в некоторых статьях, посвященных созданию КВ. Заключалась она в том, что для обхода эвристических алгоритмов обнаружения КВ расшифровка декриптором тела вируса проходит не последовательно, а с имитацией обычной работы приложения путем. обращения к блокам по псевдослучайным адресам. Описанная схема доступа к буферу данных позволяет обходить эвристические методы, основанные на анализе циклов расшифровки.

Таким же образом можно переставлять не только адреса, но и блоки данных (рис. 3), что позволит внести неопределенность в расположения данных в памяти, что также очень сильно затруднит атакующего, особенно это касается атак, связанных с переполнением буфера, где схема расположения данных в памяти сильно влияет на возможность атаки.

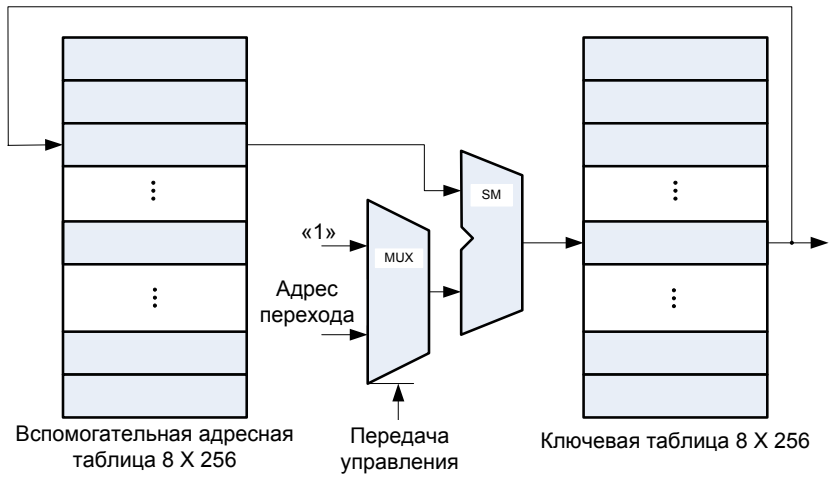
В качестве основного рабочего объекта при разработке рассмотренной защиты выступают таблицы перестановок с числом элементов, равным числу переставляемых элементов буфера или адресов. Такие таблицы активно используются в криптографии и носят названия S- и R-блоков [5, 6].



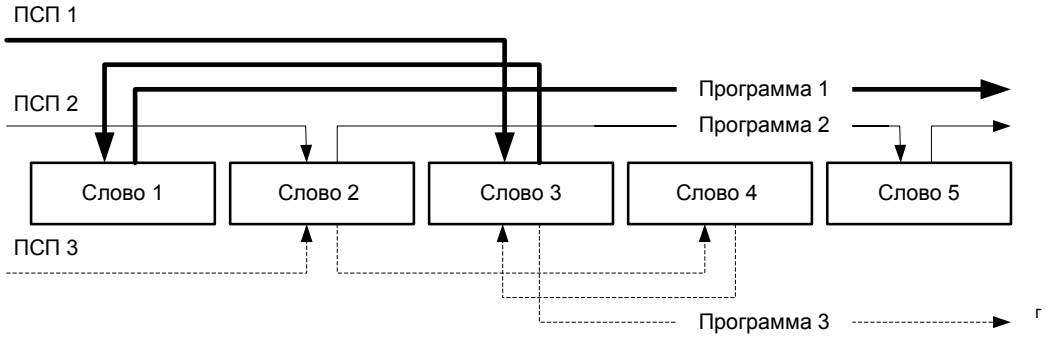
а



б



в



г

Рис. 1. Рандомизация последовательности извлечения слов команд из памяти

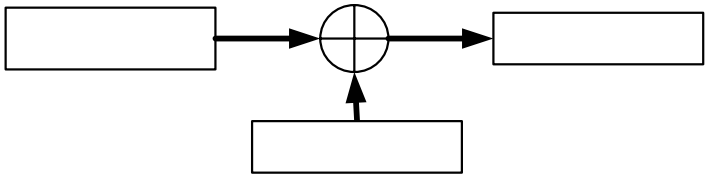


Рис. 2. Рандомизация среды исполнения

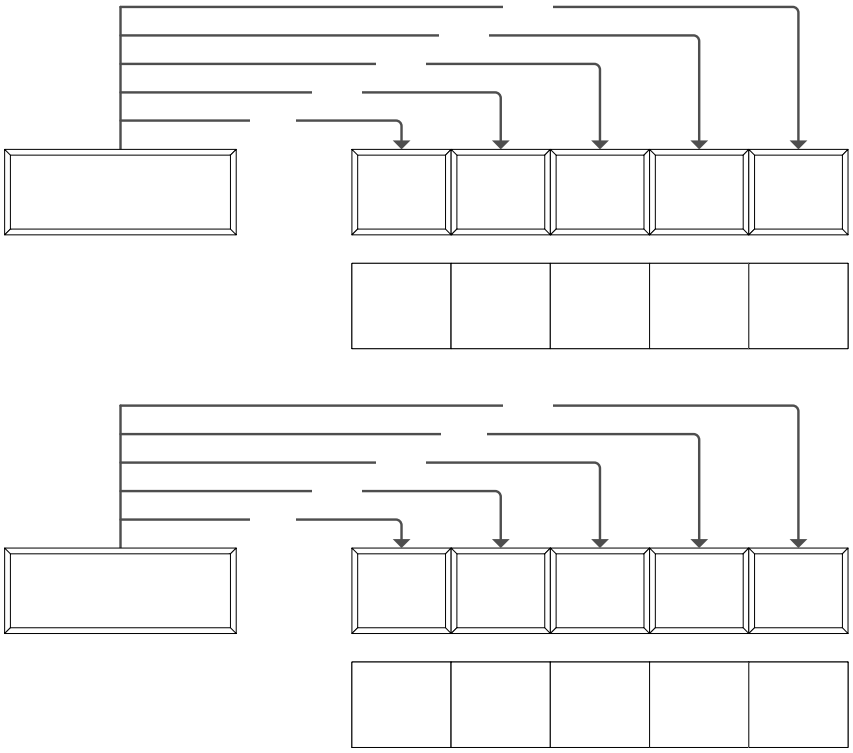


Рис 3. Схема обычной (а) и стохастической (б) расшифровки буфера

Литература

1. Иванов М.А., Чугунков И.В. Теория, применение и оценка качества генераторов псевдослучайных последовательностей. Серия СКБ (специалисту по компьютерной безопасности). Книга 2. - М.: КУДИЦ-ОБРАЗ, 2003.

2. Осмоловский С.А. Стохастические методы защиты информации. М.: Радио и связь, 2003.

3. Прилуцкий С.О. Исследование и разработка методов защиты от разрушающих программных воздействий (рукопись). 2004.

4. Ассемблер в задачах защиты информации / Абашев А.А., Жуков И.Ю., Иванов М.А. и др. – М.: КУДИЦ-ОБРАЗ, 2004.

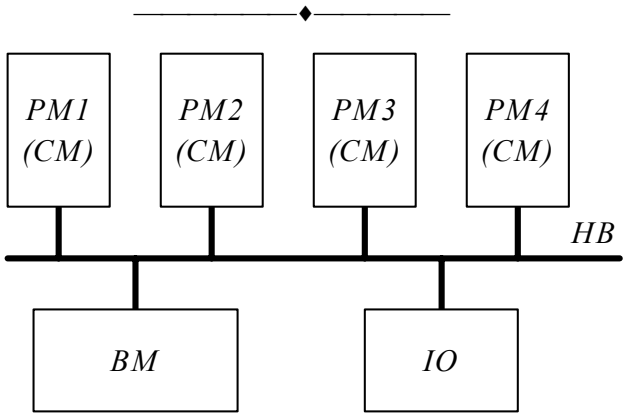
5. Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях. М.: КУДИЦ-ОБРАЗ, 2001.

6. Поточные шифры / А.А.Асосков, М.А.Иванов, А.Н.Тютвин и др. Серия СКБ (специалисту по компьютерной безопасности). Книга 3. - М.: КУДИЦ-ОБРАЗ, 2003.

RANDOMIZATION TECHNIQUES FOR MEMORY SECURITY

Prilutsky S., Ivanov M.

New general approaches for safeguarding computer systems against any type of malware are presented.



КОГЕРЕНТНОСТЬ ПАМЯТИ МНОГОПРОЦЕССОРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Дерюгин А.А.
Московский энергетический институт (технический университет)

В многопроцессорных вычислительных системах (МПВС) с общей памятью (рис. 1) в целях повышения быстродействия в состав каждого из вычислительных модулей (РМ) включается кэш-память (СМ). При этом в процессе работы МПВС обрабатываемые данные из модуля общей памяти (ВМ) передаются в СМ (и обратно) блоками, обычно содержащими 4 слова. Адреса или теги блоков, находящихся в данный момент в каждой из СМ, фиксируются в таблице блоков, расположенной в контроллере этой СМ.

В СМ разных РМ в одно и то же время может находиться один и тот же блок данных. Когерентность – это согласованность данных, хранящихся в ВМ и в кэш-памятях разных процессорных модулей, под которой понимается тот факт, что считываться из каждой ячейки блока должно последнее значение данных, записанных в эту ячейку [1, 2].

Рис. 1. Структура МПВС с общей памятью
РМ – процессорный модуль, содержащий кэш-память (СМ); ВМ – модуль общей памяти; IO – устройства ввода/вывода и внешней памяти; НВ – системная шина

Если из ячеек памяти, входящих в состав какого-либо блока, производится только считывание, то когерентность не нарушается, поскольку данные, хранящиеся в ячейках этого блока, и в СМ и в ВМ остаются неизменными.

Если же в одну из ячеек блока в одном из РМ произошла запись, то когерентность нарушается. Для ее поддержания можно, например, выполнить запись не только в СМ данного РМ , но и в соответствующую ячейку памяти ВМ (сквозную запись), а одноименные блоки в СМ других РМ объявить недостоверными, так что при необходимости этот блок будет передан из ВМ в другие РМ уже в измененном виде.

Очевидно, что для поддержания когерентности может потребоваться большое время, связанное с выполнением сквозной записи и с повторной пересылкой блока в СМ других РМ.

Для уменьшения этого времени разработано много различных протоколов поддержания когерентности [2, 3]. Все они основываются на сокращении числа сквозных записей и числа широковещательных опросов всех контроллеров СМ на предмет определения есть ли в других СМ блок, передаваемый в данный момент из ВМ в одну из СМ. Первое действие вызывает блокировку работы одного РМ, а второе – блокировку всей МПВС.

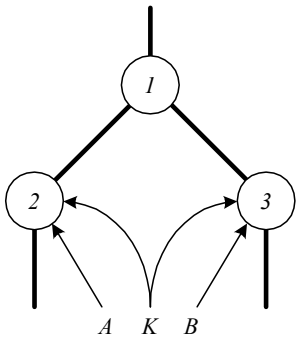


Рис. 2. Фрагмент графа параллельного выполнения частей программы

Число опубликованных работ, в которых рассматриваются различные протоколы поддержания когерентности, составляет несколько десятков. Упомянем здесь только самый известный протокол MESI (Modified/Exclusive/Shared/Invalid), используемый в МВС, построенных на базе микропроцессоров Pentium и др. По этому протоколу предусмотрено 4 возможных состояния блоков, находящихся кэш-памятях (модифицированный, эксклюзивный, общий и недостоверный) и их переходы из одного состояния в другое для всех возможных режимов работы МПВС.

Однако во всех опубликованных работах речь идет о следствии нарушения когерентности и не вскрывается причина ее появления.

Для выявления причины нарушения когерентности рассмотрим фрагмент графа, отражающего параллельное выполнение частей программы, представленных в виде узлов (рис. 2). После выполнения узла 1 могут одновременно (параллельно) выполняться узлы 2 и 3, например, в РМ1 и РМ2.

Данные, используемые при выполнении узлов 2 и 3, могут быть представлены в виде трех групп:

А – данные, которые используются только в узле 2 и могут изменяться (в дальнейшем – изменяемые данные),

В – изменяемые данные, используемые только в узле 3,

К – данные, используемые в узлах 2 и 3. Эти данные не могут изменяться в принципе, поскольку в случае изменения этих данных параллельное выполнение узлов 2 и 3 невозможно.

Если данные из групп А, В и К разместить в разные блоки, то узлы 2 и 3 могут выполняться параллельно без какого-либо взаимного влияния и без нарушения когерентности. Для пересылок блоков из ВМ в СМ (и обратно) может быть применен весьма эффективный алгоритм обратной записи. При этом изменяемые данные должны передаваться из кэш-памятей РМ1 и РМ2 в ВМ только один раз после выполнения узлов 2 и 3.

Однако при компиляции программ узлов 2 и 3 может оказаться, что в один блок попали данные из групп А и В. Именно это и является причиной нарушения когерентности и такое размещение данных не следует допускать. Не целесообразно допускать размещение в одном блоке данных из групп А и К, а также из групп В и К, поскольку при таком размещении кэш-память будет использоваться неэффективно. Например, блок с данными из групп А и К может находиться в СМ процессорного модуля, в котором выполняется узел 3, а в этой СМ к данным из группы А не может быть обращений.

Очевидно, что при разумном размещении данных по блокам проблема поддержания когерентности решается сама собой и не нужны никакие специальные средства для ее решения.

Одной из причин возникновения проблемы поддержания когерентности, по мнению автора, является то, что разработчики аппаратуры "засекретили" от программистов суть работы кэш-памяти. Пока использовались однопроцессорные вычислительные системы проблем не было. Их не было бы и при использовании МПВС, если бы программисты могли оперировать таким параметром как "размер блока" так же, как они оперируют параметром "разрядность слова".

Что же конкретно надо сделать, чтобы устранить причину появления некогерентности? Нужны совместные усилия программистов и разработчиков аппаратуры. А именно:

1. Программистам прикладных программ надо разделять специальным символом данные, относящиеся к разным параллельно работающим программам, например, двумя запятыми вместо одной.

2. Системным программистам надо строить компиляторы, позволяющие после специального символа размещать данные, начиная с нового блока.

3. Разработчикам аппаратуры надо:

- ввести в один из системных регистров РМ параметр "размер блока" для передачи его компилятору на этапе конфигурирования МПВС,

- убрать все средства, обеспечивающие поддержание когерентности: сквозную запись, протоколы наблюдения, широковещательные аннулирования и обновления и т.п. [1].

Автор отлично понимает, что предлагаемые им выше меры опоздали лет на 15 – 20, "поезд уже ушел"... Но "лучше поздно, чем никогда", поскольку поддержание когерентности сейчас обходится слишком дорого как в отношении аппаратурных, так и временных затрат, особенно в МПВС с матричной структурой связей между РМ и модулями общей памяти и в вычислительных системах с распределенной памятью.

Все это позволяет надеяться, что предложения автора будут востребованы – "игра стоит свеч".

Литература

1. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: Учебник для вузов. – СПб.: Питер, 2004. – 668 с.
2. Katz R. H., Eggers S.J., Wood D.A., Perkins C.L., Sheldon R.G. "Implementing a Cache Consistency Protocol". Proceedings of the12th International Symposium on Computer Architecture, 1985.
3. Tomasevic M., Milutinovic V. "The Cache Coherence Problem in Shared-Memory Multiprocessors", IEEE Computer Society Press, Los Alamitos, CA. 1993.

COGERENT MEMORY MALTYCOMPUTER SYSTEM

In the article the problem of the maltycomputer systems coherent memory support is analyzed. The reason of the appearance of non-coherent memory is found out. In the article is described, that the coherences of the memory can be easily presided and reached by fulfillment of some easy demands to placement in lines changeable data on the stage of program compilation.