**5 Особенности контроля запоминающих устройств**

Задача обеспечения высокой надежности записи и хранения информации в настоящее время является одной из наиболее актуальных задач в сфере IT технологий. Степень интеграции СБИС типа СнК достигает 10 млрд. транзисторов на кристалле. Большую часть проектируемых СнК занимает память. Для обеспечения надежного функционирования цифровых систем большое внимание уделяется своевременному обнаружению неисправностей, которые могут появляться в результате ошибок проектирования, неточности технологии производства, а также при влиянии дестабилизирующего воздействия внешней среды. Результаты исследований свидетельствуют о том, что до 70% отказов относится к отказам оперативных запоминающих устройств (ОЗУ).

Методы контроля запоминающих устройств подразделяются на методы встроенного функционального самотестирования и методы оперативного контроля. Методы оперативного контроля основаны на применении кодирования информации. Методы встроенного самотестирования ориентированы на обнаружение широкого класса неисправностей:

1. Неисправности константного типа (Stuck-At Fault‒SAF), при появлении которых логическое значение ячейки памяти всегда равно 0 (SA0) или 1 (SA1) независимо от операции, производимой с этой или другими ячейками памяти;
2. Переходные неисправности (Transition Fault - TF), при появлении которых ячейка памяти не способна осуществить переход из состояния логического 0 в состояние логической 1 (TF) либо наоборот (TF);
3. Неисправности взаимного влияния (Coupling Fault - CF), когда изменение логического значения одной (влияющей) ячейки отражается (различным образом) на значении второй (зависимой) ячейки.

Современные запоминающие устройства зачастую проектируются как встроенные модули, доступ к которым практически невозможен извне. Поэтому проблему тестирования встроенных модулей решают с помощью встроенной в объект аппаратуры самотестирования BIST (Built-In Self-Test).

Аппаратура самотестирования проектируется параллельно с проектированием функциональных блоков и содержит средства формирования тестовых последовательностей и средства анализа реакций объекта при тестировании. Для обеспечения условий автономного тестирования обеспечивается изоляция функционального ядра от внешних связей.

Основные достоинства технологии BIST заключаются в следующем:

1. Тестирование объекта возможно на всех стадиях жизненного цикла, начиная от проектирования и заканчивая этапом эксплуатации;
2. При тестировании могут использоваться более высокие частоты, что позволяет сократить время тестирования;
3. Нет необходимости во внешнем дорогостоящем оборудовании контроля;
4. Одновременно можно тестировать все IP блоки СнК. На этапе производства можно одновременно тестировать все компоненты на кремниевых пластинах.

**5.1 Методы функционального самотестирования**

Методы функционального самотестирования подразделяются на методы разрушающего самотестирования (off-line BIST) и методы неразрушающего (текущего) самотестирования (on-line BIST).

Особенностью оперативных запоминающих устройств (ОЗУ) является высокая интенсивность изменения хранящихся данных. Основной задачей самотестирования ОЗУ является обнаружение факта наличия неисправности. При этом задача определения типа неисправности и места ее появления не ставится.

На рисунке 1 приведена упрощенная функциональная модель ОЗУ: DB –внешняя двунаправленная m-разрядная шина данных, AB – входная n- разрядная адресная шина, CB – входная шина управления.

Механизм самотестирования ОЗУ основан на использовании функциональных алгоритмических тестов, которые представляют собой последовательность операций записи и чтения, затрагивающих все запоминающие элементы. Широкое применение

**…**

**…**

**Дешифратор**

**адреса**

Логика

чт/зп

Массив

запоминающих

элементов

АВ

СВ

n

m

DВ

Рисунок 5.1 - Функциональная модель ОЗУ

нашли маршевые тесты и язык MTL (Memory Test Language) для их описания. Достоинством маршевых тестов является относительно высокая скорость выполнения и покрывающая способность. Одним из первых маршевых тестов ОЗУ является тест MATS (Modified Algorithmic Test Sequence) для обнаружения неисправностей константного типа. MATS имеет следующую структуру:

(w0); (r0, w1); (r1),

где символ является символом адресации, определяющим полный обход всего адресного пространства ОЗУ. Данный тест является разрушающим и не требует сжатия выходных реакций. Анализ реакций очевиден.

Дальнейшее развитие маршевых тестов привело к увеличению числа операций и к применению различных подходов к адресации памяти ( - перебор адресов по убыванию,  - перебор адресов по возрастанию). Для информационных емкостей до 1 Mb данный подход требует для организации контроля до 5% от общей площади кристалла ОЗУ.

Для контроля слово ориентированных ОЗУ емкостью более 1 Mb нашли применение маршевые тесты, разработанные на основе применения разнообразных кодов. При этом применяется простое решение, например, замена операций w0, w1 операциями записи кодов w [000…0], w [111…1], а операций r0, r1 – операциями чтения r [000…0], r [111…1]. На практике проверяющая способность данной группы тестов оказалась мало эффективной, что привело к расширению набора кодов. Набор применяемых на практике кодов разбивается на две группы G0 и G1, которые при контроле используются последовательно.

Рассмотрим пример маршевых тестов для случая M = 4 (четырехбитовые слова). Пусть G0  содержит наборы [0000], [0011], [0101], [1010], а G1 содержит наборы [1111], [1100], [0101], [0101], формируемые на инверсных входах. Тогда полный маршевый тест имеет следующий вид:

(w [0000]); (r [0000], w [1111]); (r [1111]);

(w [0011]); (r [0011], w [1100]); (r [1100]);

(w [0101]); (r [0101], w [1010]); (r [1010]);

(w [1010]); (r [1010], w [0101]); (r [0101]).

Приведенный разрушающий маршевый тест может быть использован для различных типов ОЗУ. В то же время разным технологиям свойственны разные в определенной мере типы неисправностей. И не все типы неисправностей (к примеру, кодочувствительные неисправности) могут обнаруживаться данным тестом.

Появилась необходимость в настраиваемой аппаратуре самотестирования ОЗУ, которая способна обеспечить подачу произвольных последовательностей маршевых элементов, тестовых наборов и направлений обхода элементов памяти. Для решения задачи в такой постановке разработаны маршевые примитивы, пригодные для создания произвольного маршевого теста.

Ниже приведен один из предложенных наборов маршевых примитивов для создания маршевых тестов контроля ОЗУ с учетом особенностей технологии их изготовления и контроля:

SM0 = <(wx)>;

SM1 = <()>;

SM2 = <()>;

SM3 = <()>;

SM4 = <()>;

SM5 = <()>;

SM6 = <()>;

SM7 = <()>.

Для каждого примитива необходимо задать значение x и направление обхода адресного пространства. С учетом программных реализаций рассмотренных примитивов строятся микропрограммы, которые способны воспроизводить требуемый маршевый тест для аппаратуры встроенного самотестирования ОЗУ. Перенос одного проектного решения самотестируемого ОЗУ на другую технологию изготовления требует изменения только программного модуля теста в памяти микропрограмм.

**5.2 Неразрушающие маршевые тесты**

Разрушающее самотестирование может применяться в случаях, когда тестируемая компонента ОЗУ не используется по прямому назначению POST (Power- On- Self- Test). При разрушающем самотестировании применяются детерминированные последовательности операций записи и чтения в элементах памяти и сравнение получаемых реакций с ожидаемыми.

Постоянные запоминающие устройства (ПЗУ) отличаются тем, что функционируют только в режиме чтения. Контроль ПЗУ осуществляется на основе методов неразрушающего самотестирования. Неразрушающее самотестирование запоминающих устройств требует специализированных алгоритмических тестов, которые «восстанавливают» хранящиеся данные после завершения процесса диагностирования. Неразрушающее самотестирование может быть реализовано при переходе системы в режим ожидания. При неразрушающем самотестировании применяется микрооперация инвертирования данных. Анализ данных, получаемых при неразрушающем самотестировании, может происходить на основе вычисления сигнатур до и после тестирования. Несовпадение сигнатур свидетельствует о наличии ошибки.

Для построения аппаратуры встроенного самотестирования ПЗУ необходимо наличие генератора тестовой последовательности (ГТП), М – канальный анализатор сигнатур, устройство управления самотестированием и набор мультиплексоров для изоляции ПЗУ от внешних устройств.

ГТП служит для формирования псевдослучайной последовательности адресов ПЗУ.

**5.3 Методы оперативного контроля**

В основе методов оперативного контроля лежит применение кодов, обнаруживающих и исправляющих ошибки. Код, в котором все разряды слова используются для представления информации, называется простым кодом. Здесь любая возникшая ошибка превращает одну информационную комбинацию в другую.

Код, содержащий в себе кроме информационных контрольные разряды, называется систематическим кодом. Такой код обладает избыточностью. Проверяющая и корректирующая способности кода связаны с понятием кодового расстояния.

Кодовое расстояние d (A,B) для кодовых комбинаций А и B определяется как вес такой третьей кодовой комбинации, которая получается сложением по *mod 2* исходных комбинаций. Это определение совпадает с понятием кодового расстояния по Хэммингу. Поэтому в теории кодирования оно называется Хэмминговым расстоянием.

Вес кодовой комбинации V(A) – это количество единиц в кодовой комбинации. Покажем на примере определение расстояния Хэмминга для кодовых комбинаций А и B:

А = 011011100 V(A) = 5

В = 100111001 V(В) = 5

111100101 V(С = А + В) = 6

Простой код имеет минимальное кодовое расстояние d = 1. Для избыточных кодов dmin > 1. Если, к примеру, d = 2, то любые две комбинации данного кода различаются не менее чем в двух разрядах. Любая одиночная ошибка не сможет превратить данную комбинацию в какую либо разрешенную комбинацию, а приведет к появлению запрещенной. Следовательно, такая ошибка будет обнаружена.

В теории кодирования показано, что систематический код обладает способностью обнаруживать ошибки только тогда, если минимальное кодовое расстояние для него d ≥ t, где t – кратность обнаруживаемых ошибок. Чтобы не только обнаруживать, но и исправлять ошибку, необходимо иметь минимальное кодовое расстояние d ≥ 2t+1 (для одиночных ошибок d ≥3).

Абсолютная избыточность выражается количеством контрольных разрядов k, а относительная избыточность – отношением k/n, где n = k+m (m- количество информационных разрядов).

На практике наибольшей вероятностью искажения при передаче информации является искажение одного символа. Поэтому основные усилия направлены на обнаружение и исправление одиночной ошибки.

**5.3.1 Кодирование по методу чет - нечет**

Одним из первых методов кодирования информации является метод контроля четности или нечетности информации. Метод заключается в дополнении к коду байта лишнего контрольного символа, который в зависимости от количества единиц в коде устанавливается в состояние 1(0) для того, чтобы общее количество единиц было равно нечетному количеству при контроле или четному в зависимости от принятого метода. В данном случае при контроле прочитанной информации простым образом выявляется возникшая ошибка.

**5.3.2 Равновесные коды**

Наибольшее практическое применение получил код «2 из 5». Особенность равновесного кода заключается в том, что каждая информационная комбинация содержит фиксированное число единиц и нулей. Комбинации различаются только позициями единиц. В таблице 1 приведен код «2 из 5» для кодирования десятичных чисел.Код «2 из 5» обнаруживает одиночные и групповые ассиметричные ошибки.

Схема контроля приведена на рисунке 2. Схема включает десять элементов И и детектор, который обнаруживает ошибку, если число единиц n во входном векторе n >1 или n <1.

Таблица 5.1 – Пример кодирования десятичных

чисел по коду «2 из 5»

|  |  |  |  |
| --- | --- | --- | --- |
| Кодируемое число | Код  «2 из 5» | Кодируемое число | Код  «2 из 5» |
| 0 | 01100 | 5 | 00101 |
| 1 | 11000 | 6 | 10001 |
| 2 | 10100 | 7 | 01001 |
| 3 | 10010 | 8 | 00101 |
| 4 | 01010 | 9 | 00011 |

0

1

0

0

1

0

0

1

1

1

0

1

0

0

1

0

0

0

1

0

&

1

2

1

3

1

4

1

5

2

3

2

4

2

5

3

4

3

5

4

5

**Детектор «1 из 10»**

Сигнал ошибки

1

&

&

&

&

&

&

&

&

&

Рисунок 5.2 – Схема контроля кода «2 из 5»; приведен код числа 4 – «01010»

**5.3.3 Код Хэмминга**

Рассмотрим код Хэмминга, применяемый для исправления одиночных ошибок (при d = 3) или для исправления одиночных ошибок и для обнаружения двойных (при d = 4). Обозначим n – общее число разрядов кода Хэмминга, m – число информационных разрядов, k – число контрольных разрядов; n = m+k. Количество контрольных разрядов зависит от длины информационного слова. Каждый контрольный разряд обеспечивает проверку соответствующих групп разрядов из общего числа n. При декодировании производится k групповых проверок на четность информации. В результате каждой проверки в соответствующий разряд регистра ошибки записывается 0, если проверка была успешной, или 1, если обнаружена нечетность числа единичных разрядов.

Группы для проверки образуются таким образом, что в регистре ошибки после окончания проверки получается k – разрядное двоичное число, показывающее номер позиции ошибочного двоичного разряда. Изменение этого разряда и есть исправление ошибки.

Рассмотрим код Хэмминга для исправления одиночных ошибок, то есть код с минимальным кодовым расстоянием d = 3. Ошибка возможна в одной из n позиций. Следовательно, число контрольных разрядов регистра ошибок должно удовлетворять условию k ≥ log2 (n+1). Отсюда m ≤ n - log2 (n+1). В таблице 2 приведены значения параметров m и k для некоторых коротких кодов.

Таблица 5.2 – Значения параметров для кода Хэмминга

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| m | 1 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 11 |
| k | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 |

Для того, чтобы число в регистре ошибок (РОШ) указывало номер позиции ошибочного разряда, группы для контроля выбираются по следующему правилу:

1-я группа – все нечетные позиции (включая и позиции контрольных разрядов) то есть позиции, в первом младшем разряде которых стоит 1 (первый разряд справа);

2-я группа – все позиции, номера которых в двоичном представлении имеют 1 во втором разряде справа (2, 3, 6, 7, 10 и т.д.);

3-я группа – позиции с наличием 1 в третьем разряде справа;

4-я группа – позиции с наличием 1 в четвертом разряде справа и т. д.

При кодировании каждый контрольный разряд входит только в одну проверяемую группу. В таблице 3 приведены проверяемые позиции в первых пяти группах проверяемых позиций.

Таблица 5.3 – Проверяемые позиции кода Хэмминга по группам

|  |  |  |
| --- | --- | --- |
| N группы | Проверяемые позиции кода | Позиции контрольного разряда |
| 1 | 1,3,5,7,9,11,13,15,17,19… | 1 |
| 2 | 2,3,6,7,10,11,14,15, 18,19... | 2 |
| 3 | 4,5,6,7,12,13,14,15… | 4 |
| 4 | 8,9,10,11,12,13,14,15 … | 8 |
| 5 | 16,17,18,19,20… | 16 |

Рассмотрим пример кодирования информации при m = 5. Очевидно, что количество контрольных разрядов равно 4 (см. таблицу 2). В таблице 4 приведены 7-значные коды Хэмминга для кодирования десятичных чисел.

Таблица 5.4 – Коды Хэмминга для кодирования десятичных чисел

|  |  |  |
| --- | --- | --- |
| Десятичное число  Позиции  контрольных  разрядов | Простой  двоичный код | Код  Хэмминга |
|  | **К3 К2К1** |
| 0 | 0000 | 0 0 0 **0** 0 **0 0** |
| 1 | 0001 | 0 0 0 **0** 1 **1 1** |
| 2 | 0010 | 0 0 1 **1** 0 **0 1** |
| 3 | 0011 | 0 0 1 **1** 1 **1 0** |
| 4 | 0100 | 0 1 0 **1** 0 **1 0** |
| 5 | 0101 | 0 1 0 **1** 1 **0 1** |
| 6 | 0110 | 0 1 **1** **0** 0 **1 1** |
| 7 | 0111 | 0 1 1 **0** 1 **0 0** |
| 8 | 1000 | 1 0 0 **1** 0 **1 1** |
| 9 | 1001 | 1 0 0 **1** 1 **0 0** |

Пусть передан код десятичного числа 6 в виде кода Хэмминга «0110011», а принят с искажением код «0100011». Ошибка произошла в разряде пятой позиции. Очевидно, что пятая позиция проверяется в первой и третьей группе разрядов (таблица 5.3). При проверке позиций этих групп определяется нарушение четности, и устанавливаются единицы в первый и третий разряды счетчика ошибок, что и указывает на номер ошибочного разряда.

**5.4 Аппаратура оперативного контроля запоминающих устройств**

Реализация методов оперативного контроля требует применения дополнительного оборудования. На рисунке 3 приведена обобщенная структура встроенной аппаратуры

оперативного контроля ЗУ:

AB – N – разрядная шина адреса;

DI – входная шина данных разрядности M;

Synd – синдром помехоустойчивого кода;

Перед выполнением записи данных в запоминающее устройство формируются пакеты (слова), которые подаются на устройство кодирования информации. Формируется кодовое слово, содержащее кроме информационной части дополнительные контрольные биты, вырабатываемые в соответствии с применяемым алгоритмом кодирования.

Сформированное кодовое слово во время записи (W = 1) записывается в массив запоминающего устройства. При чтении информации по адресу на адресной шине AB (R = 1) кодовое слово контролируется и декодируется. Информационная часть кодового слова используется в вычислительном процессе.

Очевидно, что применение методов оперативного контроля информации связано с аппаратной избыточностью и снижением быстродействия.

M

Synd

Логика

чтения

M

S

R

N

Логика

записи

Схема коррекции

Схема декодера

Контрольные

разряды

Массив

запомин.

элементов

Схема кодера

W

R

DO

Err

AB

DI

R

Рисунок 5.3- Обобщенная структура встроенной аппаратуры оперативного контроля ЗУ