УДК 519.713:681.3

Л.В. ДЕРБУНОВИЧ, док. техн. наук, проф.,
И.Г. ЛИБЕРГ, к.т.н., доц.,
Д.В. ЯКУБОВСКИЙ, НТУ «ХПИ» (г. Харьков)

МЕТОД ФУНКЦИОНАЛЬНОГО ДИАГНОСТИРОВАНИЯ НЕИСПРАВНОСТЕЙ МИКРОКОНТРОЛЛЕРНЫХ УСТРОЙСТВ УПРАВЛЕНИЯ

Пропонується підхід для розрахунку вертикальної сигнатури, розмір якої є змінюваним, що дозволяє поєднати вертикальну, горизонтальну сигнатури а також розмір програмного сегменту у комбіновану сигнатуру. Це дає змогу покращити головні параметри функціонального контролю мікроконтролерної системи .

A scheme for calculation of vertical signature with variable length is proposed. It makes possible to combine vertical and horizontal signatures with length of data block into one hybrid signature. This approach allows to improve main parameters of concurrent monitoring.

Широкое использование микроконтроллерных устройств управления (МКУ) позволило определить, что неисправности неустойчивого типа являются основной причиной отказов, возникающих в процессе управления и обработки данных, и, в меньшей степени, причиной ошибок при передаче данных. Наибольшее распространение в дискретных системах управления получили методы тестового и функционального диагностирования. Тестовое диагностирование производится подачей тестовых воздействий на диагностируемый объект, при функциональном диагностировании на входы диагностирующего устройства поступают только рабочие воздействия. Наименьший латентный период обнаружения ошибки, максимальное количество ошибок, которые могут быть обнаружены диагностирующим устройством – факторы, определяющие отказоустойчивость системы управления. Таким образом, стоит задача улучшения этих параметров с сохранением простоты диагностического оборудования и минимизацией временной избыточности и избыточности памяти.

В [1] предложено использование схемы получения гибридной сигнатуры (объединяющей вертикальную и горизонтальную). В [2] описаны методы обеспечения контроля функционирования микроконтроллера с использованием сигнатурного мониторинга. В [3] предложено использование сигнатурного мониторинга для диагностирования правильности функционирования программы МКУ.

Цель статьи – описание метода сигнатурного мониторинга, максимально отвечающего требованиям быстродействия и минимальности аппаратурных затрат.

Гибридный сигнатурный мониторинг. Большинство предыдущих работ по сигнатурному мониторингу МКУ посвящено методам, где используются только вертикальные сигнатуры. Главный недостаток таких методов – большой латентный период обнаружения ошибки. Концепция горизонтальных сигнатур, комбинированных с вертикальными, впервые была представлена в [1] с целью сократить латентный период обнаружения ошибки. Этот метод должен был быть совмещён с подсистемой памяти, оснащённой проверкой чётности или коррекцией одиночной и обнаружением двойной ошибки. С целью уменьшения латентного периода обнаружения ошибки, как правило, надо увеличивать количество сигнатур и программных сегментов, однако это приведёт к возрастанию избыточности памяти и замедлит выполнение основной программы. В этой работе предлагается гибридная схема сигнатурного мониторинга, которая объединяет вертикальную и горизонтальную сигнатуры с длиной программного сегмента в одно сигнатурное слово. Гибридная сигнатура не похожа на обычно используемые вертикальные сигнатуры, генерируемые функцией сложения по модулю 2, которые требуют целое слово такой же разрядности, что и команда. Свойство переменной длины, являющееся главной особенностью предлагаемого подхода, позволяет совместить вертикальную, горизонтальную сигнатуры и длину программного сегмента в одном слове.

Ниже описывается процесс генерации вертикальной сигнатуры.

В двукоординатной системе (X, Y) пусть (x_{vs}, y_{vs}) будет вертикальной сигнатурой программного сегмента.

При генерации сигнатуры в каждый момент времени рассматриваются две строки кода программы, следующие друг за другом.

Имеется такой набор возможных состояний для каждой пары бит, расположенных в столбце:

$$S = (s_1, s_2, s_3, s_4) = \left(\left(\frac{0}{0}\right), \left(\frac{0}{1}\right), \left(\frac{1}{0}\right), \left(\frac{1}{1}\right)\right).$$

В системе двух координат (X,Y) вводится такое символьное кодирование состояний S:

$$(s_1, s_2, s_3, s_4) = (\rightarrow, \leftarrow, \downarrow, \uparrow),$$

где стрелка соответствует единичному перемещению по одной из координат.

В начале вычисления сигнатуры её значение составляет (x_{vs}, y_{vs}) = (0,0). Далее происходит изменение сигнатуры в соответствии с состоянием пар бит в текущих строках кода:

в случае s_1: (x_{vs}, y_{vs}) ← (x_{vs}+1, y_{vs});
в случае s_2: (x_{vs}, y_{vs}) ← (x_{vs}–1, y_{vs});

в случае s_3: (x_{vs}, y_{vs}) ← (x_{vs}, y_{vs}–1);
в случае s_4: (x_{vs}, y_{vs}) ← (x_{vs}, y_{vs}+1);

Пример. Имеется две строки кода программы:

S_1 : 1 0 1 0 1 0 1 1
S_2 : 0 1 0 1 0 0 0 0

Этот сегмент кодируется следующим образом:

$$\left(\frac{1}{0}\frac{0}{1}\frac{1}{0}\frac{0}{1}\frac{1}{0}\frac{0}{0}\frac{1}{0}\frac{1}{0}\right) = (\downarrow\leftarrow\downarrow\leftarrow\downarrow\rightarrow\downarrow\downarrow).$$

В двукоординатной плоскости это можно проиллюстрировать графиком, который показан на рис. 1. Для данного программного сегмента вертикальная сигнатура составляет (-1,-5).

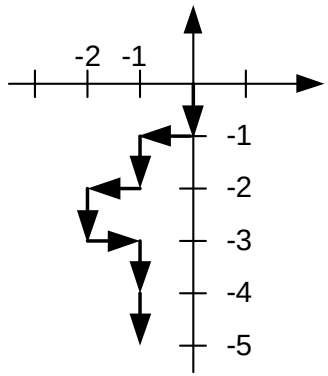

Рис. 1. Пример получения сигнатуры

Каждое состояние может быть закодировано любым из набора элементов (→,←,↓,↑). Следовательно, может быть получено 24 возможных кодировки состояний. Далее приводится обоснование выбора кодировки.

Главная задача при выборе кодировки – это увеличение покрытия ошибок. 24 варианта назначения соответствий можно разбить на 2 группы, А и Б:

А. Пусть комбинации $\left(\frac{0}{1}\right)$ и $\left(\frac{1}{0}\right)$ определяют движения по одной и той же оси, но в разных направлениях. Всего возможны 8 кодировок, которые принадлежат к этой группе. Когда одиночный столбец в программном сегменте имеет количество n состояний $\left(\frac{0}{1}\right)$, где n>=1, этот

столбец должен также иметь как минимум (n–1) и как максимум (n+1) состояний $\binom{1}{0}$. Однако $\binom{0}{1}$ и $\binom{1}{0}$ соответствуют перемещению по одной и той же оси, но в противоположных направлениях, и эти перемещения в двумерном пространстве компенсирует друг друга. Следовательно, в сигнатуре от этого столбца останется либо $\binom{0}{1}$, либо $\binom{1}{0}$, либо два эти движения будут полностью компенсированы. Например, если определить соответствие $\left(\binom{0}{1}\binom{1}{0}\right)$=(→,←), сигнатура по оси X для одного столбца ограничена значениями (-1,0,1). Учитывая, что каждый столбец может быть обработан независимо, такое кодирование приведёт к увеличению вероятности того, что или сигнатура по оси X или сигнатура по оси Y будет ограничена значениями (-1,0,1). Вследствие этого ограничения, нельзя гарантировать достижения хорошего баланса распределения сигнатур в двумерных координатах. Также увеличится вероятность маскируемых ошибок. Итак, возрастает возможность не обнаружить ошибку при проверке сигнатуры, и, значит, уменьшается покрытие ошибок.

Б. Пусть комбинации $\binom{0}{1}$ и $\binom{1}{0}$ определяют движения по разным осям. Остальные 16 кодировок относятся к этой группе. Из вышеизложенных соображений распределение сигнатур при этих 16 кодировках более сбалансированы в двухкоординатном пространстве, чем для класса А. Таким образом, класс Б имеет меньшую вероятность того, что ошибка будет не обнаружена, чем класс А. Очевидной разницы в воздействии на покрытие ошибок среди кодировок внутри этой группы нет. Поэтому используется кодировка состояний, которая описана в примере генерации сигнатуры.

Количество бит, описывающие каждую из координат сигнатуры, определяет размер пространства сигнатуры. Большее сигнатурное пространство может уменьшить вероятность того, что ошибка не будет обнаружена, но при этом требуется больше бит для сохранения сигнатуры.

Таким образом, появляется возможность выбора длины вертикальной сигнатуры, чего не позволяет классическая схема генерации сигнатур. Ограничив длину вертикальной сигнатуры, можно соединить вертикальную и горизонтальную сигнатуры, а также длину программного сегмента в одно сигнатурное слово. Горизонтальная сигнатура восполнит уменьшение покрытия ошибок, вызванное уменьшением длины вертикальной сигнатуры, и уменьшит задержку обнаружения ошибки. Роль

элемента “длина программного сегмента” – дать указание диагностирующему процессору, когда необходимо провести сравнение эталонной сигнатуры с вычисленной в процессе функционирования. Указание длины программного сегмента в сигнатурном слове позволяет устранить биты меток из слов команд, упростить систему памяти и уменьшить избыточность её использования.

Сигнатурное слово разделено на три части: Длина программного сегмента, Горизонтальная сигнатура и Вертикальная сигнатура, как показано на рис. 2.

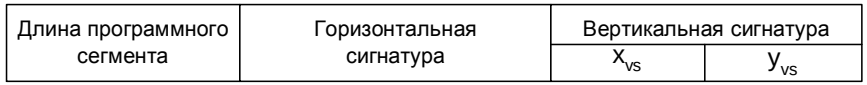

Рис. 2. Поля гибридной сигнатуры

Таким образом, предлагаемый метод вычисления вертикальной сигнатуры с использованием двухкоординатного пространства даёт возможность выбирать размер сигнатуры, что, в свою очередь, позволяет заключить в сигнатурное слово дополнительную информацию, например, о длине программного сегмента или горизонтальную сигнатуру. Такая (гибридная) сигнатура даёт возможность получить более высокие показатели обнаружения ошибок, чем традиционные.

Список литературы: 1. *K.Wilken and J. Shen* “Continuous Signature Monitoring: Low-Cost Concurrent Detection of Processor Control Errors” // IEEE Trans. Computer-Aided Design. – 1990. – Vol.9, N 6. – P. 629-641. **2**. *И.Н. Миков, Л.В. Дербунович, В.В. Нешвеев* Программируемые контроллеры повышенной надежности для управления автоматическими линиями. Обзор.-М.: НИИмаш. – 1984. – С. 52. **3.** *Kent D. Wilken, Timothy Kong* “Concurrent Detection of Software and Hardware Data-Access Faults” // IEEE Trans. Computer-Aided Design. - 1997. - Vol.46, N 4. - P. 412-424.

Поступила в редколлегию 10.07.07