

Жаднов В.В., Тихменев А.Н.

МОДЕЛИРОВАНИЕ КОМПОНЕНТОВ ЭЛЕКТРОННЫХ СРЕДСТВ С РЕКОНФИГУРИРУЕМОЙ СТРУКТУРОЙ

Развитие вычислительных способностей контроллеров и их применение в построении электронных средств (ЭС) привело к все более широкому распространению реконфигурируемых ЭС. Особенно часто реконфигурацию используют при проектировании аппаратуры ответственного применения, к которой предъявляются повышенные требования по надежности.

Для реализации алгоритмов реконфигурации в ЭС вводят дополнительные элементы, которые осуществляют контроль и управление функциональными компонентами. Это позволяет отключать неиспользуемые компоненты, изменять рабочие режимы в зависимости от решаемых задач и т.д. Так же появляется необходимость осуществить реконфигурацию ЭС в случае отказа элементов, тем самым сохраняя систему в работоспособном состоянии. Такая (переменная) структура имеет преимущества по показателям потребляемой энергии, эффективности функционирования, износу элементов, надежности и др.

При разработке ЭС ответственного применения необходимо провести расчет показателей надежности и, тем самым обосновать, что проект соответствует требованиям, предъявленным в техническом задании. Для реконфигурируемых электронных средств расчет надежности сложен [1] из-за необходимости учета алгоритмов переключения компонентов, приводящих к изменению их режимов работы.

Для расчета могут применяться (и часто используются) аналитические методы. Однако их недостатками являются трудоемкость проведения расчетов, необходимость повторного проведения вывода расчетных формул при корректировке алгоритмов резервирования [1]. Так же зачастую приходится использовать упрощения в расчетах, что приводит к получению «нижней» оценке значений показателей надежности. Это является существенным недостатком аналитических методов, так как может привести к излишнему резервированию и, следовательно, к неоправданному удорожанию ЭС и снижению его конкурентоспособности.

Для расчета показателей надежности ЭС со сложной структурой, таких как реконфигурируемые ЭС, возможно применение методологии имитационного моделирования. Такой подход предоставляет ряд преимуществ над аналитическими методами [2]. При имитационном моделировании строится модель ЭС, над которой проводятся эксперименты с целью определить нагрузку на отказ. Модель позволяет более точно описать происходящие процессы и алгоритмы реконфигурации, поэтому рассчитывается не нижняя оценка показателей надежности, а их среднее значение. Точность расчета зависит от количества экспериментов и растет при увеличении их числа. Основным недостатком является необходимость проведения большого числа экспериментов и, следовательно, и большие требования к вычислительным ресурсам. Так же к недостаткам можно отнести сложность построения моделей.

В рамках данной задачи необходимо было разработать средства для создания модели ЭС и проведения на ней имитационных экспериментов. В качестве инструмента для создания моделей был предложен специализированный язык описания отказов реконфигурируемых ЭС.

В [3] приведен алгоритм выполнения имитационного эксперимента, логические и алгоритмические возможности разрабатываемого языка в обобщенном виде, но не описывалась модель ЭС и не рассматривались примеры, которые могут дать представления о возможности использования языка для конкретных задач.

Текстовая модель, составленная на нем, преобразовывается в программную. Элементарным объектом моделирования является узел (*knot* в рамках синтаксиса языка). *Knot* описывает один компонент структурной схемы надежности, это может

быть отдельное электрорадиоизделие (ЭРИ) или электронный модуль, блок и т.д.

Основными параметрами узла являются режим (*mode*) и состояние (*state*). Список состояний позволяет моделировать «сеансный» режим работы и (или) восстановления элемента. Режимы предназначены для случая, если в одном состоянии наработка узла может описываться разными законами распределения.

Описание узла, представленное на рис. 1, моделирует компонент с именем *BLOCK1*, жизненный цикл которого протекает по следующему алгоритму.

```
knot BLOCK1
{
  state: Fail, WaitRecovery, Recovery, Work;
  mode: Normal,Hard;
  startState: Work;
  startMode: Normal;

  contriMode: unDistribution;

  tableDistribution:
  | Normal | Hard |
  |-----|-----|
  WaitRecovery | cell 1 Dis_WRTIME | cell 1 Dis_WRTIME |
  Recovery      | cell 2 Dis_RecoveryTime | cell 2 Dis_RecoveryTime |
  Work          | Dis_Work_Normal | Dis_Work_Hard;

  tableStateChange:
  | Normal | Hard |
  |-----|-----|
  WaitRecovery | Recovery | Recovery |
  Recovery      | Work | Work |
  Work          | WaitRecovery | WaitRecovery;

};
```

Рис.1. Модель компонента *BLOCK1*

В начальный момент времени узел исправен и работает с нормальной интенсивностью отказов (*startState* и *startMode*). По истечению времени, разыгранному по закону *Dis_Work_Normal* узел отказывает и начинается восстановление. Оно проходит в два этапа, на первом *BLOCK1* ожидает, к примеру, обнаружения отказа, если контроль не полный, или доставки запасных частей. Время ожидания распределено по закону *Dis_WRTIME* (*tableDistribution*). В данный момент не будем рассматривать конкретное распределение, оно может быть практически любым так, как в языке предусмотрен широкий инструментарий для задания распределений. После ожидания начинается непосредственно восстановление, так же со своим распределением необходимого времени. Далее узел возвращается в рабочее состояние.

При старте эксперимента узлу присваиваются стартовые значения (*startState* и *startMode*). Разыгрывается время по закону, соответствующему паре состояние-режим. После розыгрыша модель узла просто ожидает наступления этого времени по часам эксперимента. По истечении этого времени узлу присваивается новое состояние по таблице *tableStateChange*. При каждом переходе из состояния в состояние или изменении режима разыгрывается время пребывания в новом состоянии, по закону, взятому из *tableDistribution*.

Этот процесс останавливается только по достижении нулевого состояния, которое указано в списке состояний первым (в данном случае *Fail*). Это окончательный отказ, после его наступления не разыгрываются случайные величины для данного узла и невозможно сменить состояние извне до окончания эксперимента. В представленном примере это состояние не используется, так как *BLOCK1* восстанавливаемым.

Два режима введено для моделирования изменения нагрузки на узел, но изменение нагрузки может происходить из-за отказов других узлов и в явном виде его закон распределения не известен. Предположим, что *BLOCK1* работает в нагруженном резерве с идентичным узлом *BLOCK2*. Режим работы первого зависит от работоспособности

второго блока. Такой алгоритм моделируется при помощи событий `switch_event` (рис 2).

```
switch_event BLOCK2_FAIL
(BLOCK2:Work->)
{
  set_mode (BLOCK1:Hard);
};

switch_event BLOCK2_WORK
(->BLOCK2:Work)
{
  set_mode (BLOCK1:Normal);
};
```

Рис. 2. Зависимость режима работы `BLOCK1` от компонента `BLOCK2`

Первое событие (`BLOCK2_FAIL`) возникает, когда `BLOCK2` покидает работоспособное состояние, результатом является переход `BLOCK1` в более нагруженный режим работы (`Hard`). Это осуществляется через вызов специализированного оператора `set_mode(knot_name:mode_name)`. При изменении режима так же будет проведен новый розыгрыш значения наработки на отказ по новому закону распределения. Но есть вероятность, что переход состоится в момент, когда узел находится на восстановлении, и заново разыгрывать время в этом случае не нужно. Именно для такой ситуации при описании узла в таблице распределений (`tableDistribution`) используется ключевое слово `cell`. Этим словом резервируются состояния, при переходе между которыми не нужно заново разыгрывать случайную величину. Каждый такой набор должен обладать идентификатором (число после `cell`, в данном случае 1 и 2) и в одной ячейке у каждой пары состояний должны быть одинаковые распределения, иначе компилятор выдаст ошибку.

Неверные результаты могут быть получены и при изменении режима во время работы узла. Алгоритм с повторным розыгрышем реализации наработки на отказ по другому закону адекватен только при условии, что оба закона экспоненциальные, с неизменной интенсивностью отказов. В случае не экспоненциальных законов существуют два пути решения проблемы. Наиболее универсальный и всеобъемлющий заключается в самостоятельном создании расчетной процедуры (в рамках описания модели, то есть на том же языке, с теми же глобальными переменными, но с дополнительными входными параметрами, которые могут быть использованы при расчетах), которая выполнялась бы вместо заложенной в программе. Такой метод является наиболее гибким, так как при расчете можно использовать практически любые данные, поставлять наработку в зависимости от состояний других узлов, более гибко, нежели при простой смене законов распределения, использовать при расчетах историю текущего эксперимента (теоретически можно использовать и данные, собранные по серии экспериментов, но практическая необходимость этого сомнительна). Но этот метод зачастую будет излишен, так как «не экспоненциальные» законы распределения используются редко. Поэтому здесь этот метод рассматриваться не будет, так как находится в стадии разработки, еще нет сложившегося синтаксиса вызова такой процедуры и не определены дополнительные параметры для нее.

Вторым вариантом решения задачи являются модифицированные ячейки (`lscell`). Этот метод применим, когда физический процесс, вызывающий отказ, схож с износом. Синтаксис использования модифицированных ячеек аналогичен рассмотренному ранее синтаксису `cell`.

Рассмотрим, что происходит при использовании модифицированных ячеек на примере узла с двумя режимами работы и разными распределениями, объединенными в ячейку. При входе в ячейку разыгрывается случайная величина ξ , принадлежащая интервалу $[0, 1]$. Эта величина определяет реализацию наработки на отказ (1) и хранится до выхода из ячейки.

$$T_1 = F_1^{-1}(\xi), \quad (1)$$

где T_1 – наработка на отказ, $F_1^{-1}(\xi)$ – функция, обратная первому закону распределения, ξ – случайная величина.

Если по прошествии времени τ изменяется закон распределения, то новая реализация наработки на отказ T_2 рассчитывается по формулам (2) и (3):

$$T_n = F_2^{-1}(\xi), \quad (2) \quad T_2 = T_n - T_n \cdot \frac{\tau}{T_1} \quad (3)$$

где T_n – промежуточная реализация наработки на отказ, $F_2^{-1}(\xi)$ – функция, обратная второму закону распределения, ξ – та же реализация случайной величины, что и в (1).

При следующем изменении закона распределения (по прошествии времени τ_2)? при перерасчете наработки на отказ τ в формуле (3) принимается равным $T_{nold} - (T_{2old} + \tau_2)$, где T_{nold} – это значение T_n из (2) на предыдущем шаге, аналогично T_{2old} и T_2 .

Таким образом ξ выступает как реализация ресурса, который расходуется с разными скоростями при различных режимах работы. Это позволяет адекватно моделировать ситуацию переключения режимов работа/хранение или изменение нагрузок на узел, если наработка на отказ распределена не по экспоненциальному закону, а, к примеру, по нормальному.

В большинстве случаев на практике используются независимые от времени интенсивности отказов, не требующие таких перерасчетов. То есть, разработанные модели в рамках синтаксиса языка предоставляют возможности, хотя и редко требуемые на практике, но обеспечивают расширение средств моделирования разнообразных компонентов ЭС и их режимов работы.

Применимость различных подходов к перерасчету наработки на отказ должен определять инженер, проводящий расчет, на основе анализа используемых моделей отказа и физических процессов, их вызывающих. Поэтому разрабатываемое программное обеспечение направленно на использование специалистами, хорошо знающими особенности функционирования моделируемого объекта и физические процессы, протекающие в компонентах.

ЛИТЕРАТУРА

1. Жаднов, В.В. Автоматизация проектных исследований надежности радиоэлектронной аппаратуры: научное издание. / В.В. Жаднов, Ю.Н. Кофанов, Н.В. Малютин и др. – М.: Радио и связь, 2003. – 156 с.
2. Строгалев, В.П. Имитационное моделирование. / В.П. Строгалев, И.О. Толкачева. – М.: Изд-во МГТУ им Н.Э. Баумана, 2008. – 280 с.
3. Тихменев, А.Н. Язык описания отказов электронных средств с реконфигурируемой структурой. / А.Н. Тихменев. // Научно-техническая конференция студентов, аспирантов и молодых специалистов МИЭМ: Тез. докладов. – М.: МИЭМ, 2010. – с. 137.

Тройков С.М., Палагута К.А.

ПОЛУНАТУРНОЕ МОДЕЛИРОВАНИЕ ЭЛЕКТРОННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ДВИГАТЕЛЕМ ВНУТРЕННЕГО СГОРАНИЯ (ЭСУ ДВС)

Существование многочисленных диагностических стендов и стендов имитации отдельных режимов работы ДВС привело к идее разработки системы,

позволяющей комплексно и согласованно выполнять моделирование всех функций реальной электронной системы управления (ЭСУ) двигателем внутреннего