Окончание. Начало в № 6`2008

Что такое «не везет» и как с ним бороться, или Как обеспечить надежность РЭС при разработке

Владимир КОВАЛЕВ
notes@novus-lab.ru

Конструкционные аспекты

Спектр конструкционных вопросов очень широк. Полностью ответить на них в рамках журнальной статьи практически невозможно. Затронем лишь наиболее «популярные».

«Мостиком» от электрической схемы к конструктиву можно считать печатную плату (ПП), которая являет собой некий «сплав» электрической схемы и механической конструкции. Именно печатная плата определяет механическую прочность электронного узла, смонтированного на ней. В ряде случаев она служит одним из несущих элементов конструкции всего устройства. Во многом именно печатная плата ответственна за ЭМС, терморежимы РЭ. В общем, ПП является крайне важным звеном разрабатываемого изделия, и многие конструкционные вопросы надежности прямо или косвенно с ней связаны.

Одним из определяющих элементов надежности смонтированной печатной платы служат посадочные места РЭ. Правильно разработанное (с учетом технологии пайки и монтажа, разброса размеров РЭ) посадочное место обеспечит минимум брака при сборке изделия. В своей практике, при проектировании посадочных мест, автор предпочитает руководствоваться стандартом IPC-7351. Неплохим подспорьем в этом является "freeware"-программа "PCB Matrix LP Viewer" (**http://www.pcblibraries.com**). В любом случае, этот вопрос желательно согласовывать с конструкторами и технологами сборочного производства, которые дадут необходимые рекомендации по допускам и зазорам (например, для операции автоматического монтажа). Внимательнее нужно относиться и к правильной маркировке посадочного места. Необходимая информация в слое маркировки значительно упростит процесс монтажа, ремонта и обслуживания. Например: информация о контуре РЭ, цоколевка (маркировка ключевых выводов), позиционное обозначение, поясняющие надписи у разъемов, контрольных точек и подстроечных элементов.

При разработке посадочного места массивных компонентов не стоит забывать о необходимости их дополнительного крепления. Это значительно повышает надежность печатного узла при механических перегрузках (удары и вибрации). Нередко разработчики считают дополнительные крепления крупногабаритных деталей необязательными, несмотря на рекомендации производителя РЭ. Автору не раз приходилось видеть буквально высыпавшиеся из плат РЭ в устройствах, работающих в условиях сильных вибрационных нагрузок. Поэтому выбор точек закрепления печатной платы также важен. Редко какой инженер делает прочностной (в том числе и модальный) анализ, несмотря на то, что современные CAD- и FEM-пакеты позволяют проводить такой анализ (хотя бы оценочный) даже инженеру средней квалификации. Во многих случаях достаточно даже провести компоновку с учетом возможных ударных и вибронагрузок руководствуясь опытом, прикидочными расчетами и здравым смыслом, хотя, конечно, субъективные критерии иногда бывают далеки от объективных. Кстати, для аппаратуры, работающей в условиях значительных механических перегрузок, возможно, стоит пересмотреть элементную базу, отдав предпочтение, например, РЭ с лучшими массо-габаритными показателями (обладающими, естественно, и меньшими моментами инерции, что благоприятно скажется на вибростойкости электронного узла).

С процессом проектирования печатной платы неразрывно связан и вопрос компоновки, связанный также с дизайном изделия. Часто можно встретить такие огрехи компоновки, как неудачное расположение РЭ, теплоотводов, подсоединительных терминалов и интерфейсных разъемов, что затрудняет монтажные или ремонтные работы, нарушает технологические зазоры (например, для высоковольтных цепей), создает неблагоприятный терморежим уже собранного в корпусе устройства и т. д.

Особое внимание следует уделить вопросам обеспечения нормальных терморежимов РЭ в смонтированном устройстве. Оптимальное размещение элементов на печатной плате позволяет избежать многих неприятных эффектов, например градиента температур (что особенно важно для прецизионных узлов). В приведенном на рис. 9 примере показано, как градиент температур, вызванный неудачной компоновкой, может привести к ухудшению характеристик схемы. Разница между элементами R3 и R4 достигает 20 °С, что для резисторов общего назначения значит уход сопротивлений от номинала на единицы процентов.

На рис. 10 показан эффект «локального перегрева», когда близко расположенные на плате тепловыделяющие элементы, «подогревая

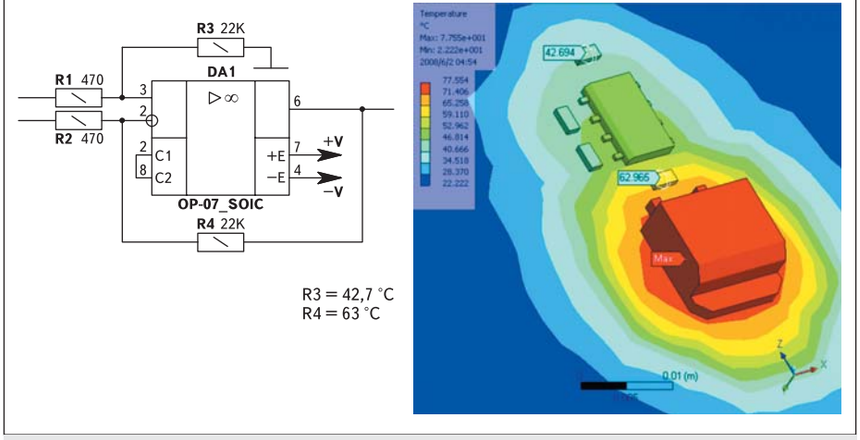

Рис. 9. Компоновка, приводящая к нарушению работы прецизионной схемы из-за температурного градиента на печатной плате

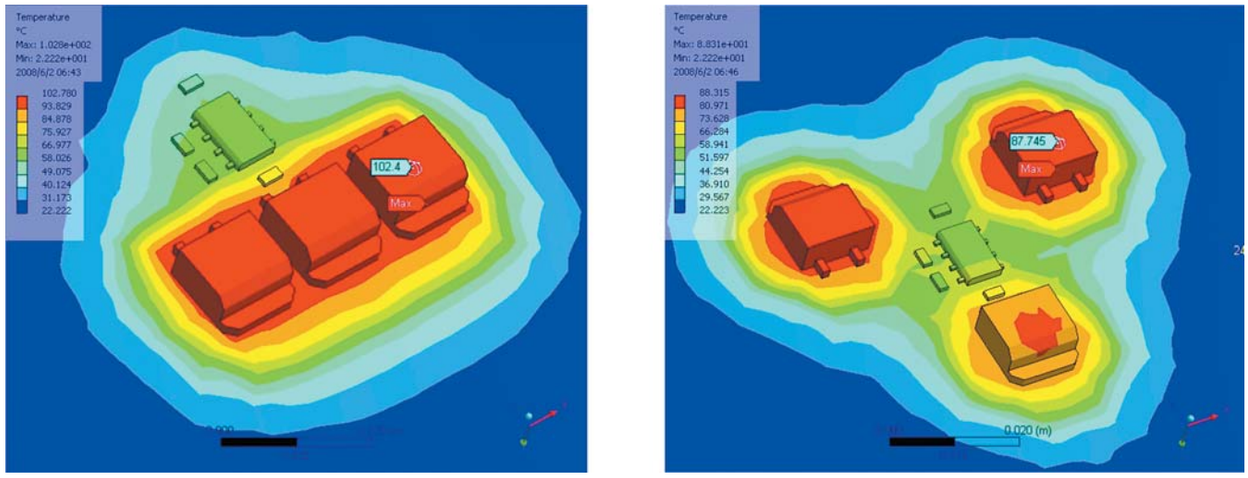

Рис. 10. Пример компоновки, приводящей к локальному перегреву печатной платы и компонентов, и пример компоновки для обеспечения более легкого температурного режима

друг друга», обеспечивают локальную концентрацию большой тепловой мощности при неудовлетворительном ее отводе. Причем для выхода из данной ситуации достаточно обеспечить более равномерное распределение тепла, например, «растащив» на бóльшие расстояния тепловыделяющие элементы.

Показанный на рис. 11 «тепличный» эффект гарантирует перегрев зоны «внутри» радиатора за счет поглощения теплового излучения расположенными там РЭ, да еще и за счет изоляции конвекционных потоков.

Можно привести десятки неудачных, с точки зрения обеспечения терморежимов, примеров компоновок. Это показывает лишь, насколько индивидуальным может быть каждое конструкторское решение, и что универсальных рекомендаций тут не существует. Нужно лишь помнить о возможных проблемах и внимательно относиться к вопросу обеспечения терморежимов конструкции.

Не менее важен, чем обеспечение терморежимов, и вопрос электромагнитной совместимости (ЭМС). Здесь как нельзя лучше подходит выражение «Профилактика лучше лечения». Лучше не бороться с помехами, а не создавать условий для их возникновения. Один из методов — бороться с импульсными и ВЧ-помехами путем локализации цепей, генерирующих помеху. Как правило, это цепи, где протекают импульсные и ВЧ-токи значительной величины («значительной» — понятие сугубо субъективное и индивидуальное для каждой конкретной разработки). Хороший метод — минимизация трасс на печатной плате и соединительных проводов, для цепей с протекающими импульсными и ВЧ-токами, а также шунтирование емкостями по питанию тех узлов схемы, которые потребляют большие импульсные мощности. Если же источник помех локализовать не удалось, то нужно защищать подверженные помехам цепи и узлы схемы. Что касается слаботочных и прецизионных цепей, есть ряд методов, описанных в литературе. Еще одним действенным методом защиты от помех является экранирование. Экранировать можно как источник помех, так и подверженные влиянию помех чувствительные узлы схемы. Экранирование может осуществляться как экранами (электромагнитными и электростатическими), так и элементами конструкции, а также полигонами и специальными слоями на печатной плате.

Следующий важный момент, особенно для сильноточных, импульсных, прецизионных схем и в самом тяжелом случае — их сочетанием, — это разводка «питания» и «земли». Часто разводке этих цепей не уделяется должного внимания, в результате резко снижается надежность схем, повышается вероятность сбоев в работе, ухудшаются характеристики, вплоть до полной неработоспособности удачно работающего «в макете» изделия.

Для обеспечения разводки цепей «земли» и «питания» необходимо выделить функциональные блоки схемы, потребляющие большие токи, а также блоки, чувствительные к помехам по питанию. Выделив такие участки схемы, необходимо развести питающие и «земляные» цепи таким образом, чтобы получились независимые контуры протекания питающих токов для каждого блока. При этом «земли» всех блоков, а также общего источника питания должны быть эквипотенциальны, причем как для постоянного тока, так и для ВЧ (для импульсных схем можно ориентироваться на частоту 3-й, а лучше 5-й гармоники самого короткого импульсного сигнала). Конструктивно, эквипотенци-

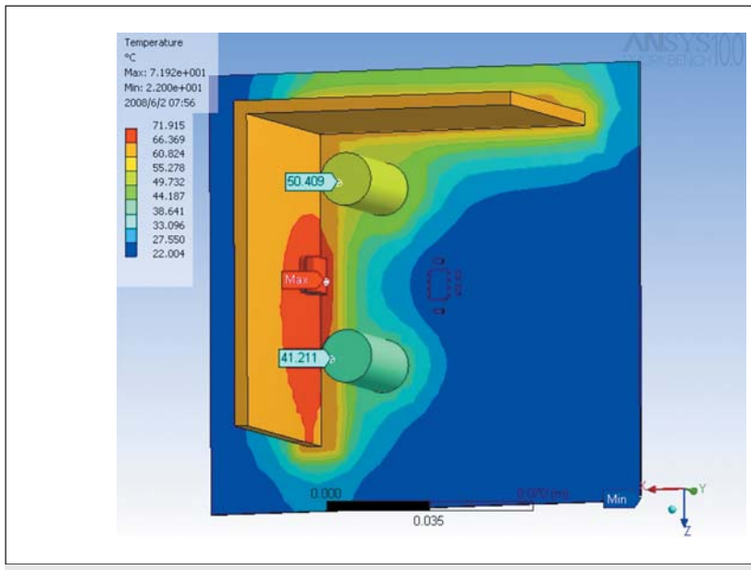

Рис. 11. Пример неудачной с точки зрения отвода тепла компоновки (тепличный эффект)

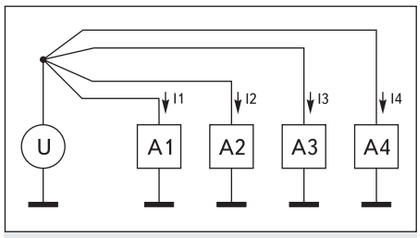

Рис. 12. Теоретически правильная схема разделения контуров питающих токов

альность обычно можно достичь за счет разводки «земли» трассами наименьшей длины и наибольшего возможного сечения. Для примера рассмотрим топологию, показанную на рис. 12.

Если «земли» всех блоков A1…A4 и источника питания U, с пренебрежимо малым внутренним сопротивлением, эквипотенциальны, то токи I1...I4 в контурах питания никоим образом не будут влиять друг на друга. На практике реализация данной топологии может быть затруднительна, в частности, проблематично обеспечить эквипотенциальность «земель» всех блоков. Как правило, реальная схема выглядит примерно так, как показано на рис. 13, то есть присутствует некая «земляная шина» с ненулевым (возможно, даже довольно большим) импедансом. Задача разработчика — свести ее импеданс к минимуму.

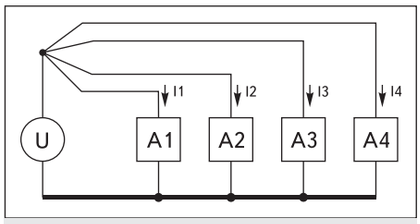

Рис. 13. Схема разделения контуров питающих токов с учетом реального ненулевого импеданса «земляной» шины

Кроме снижения индуктивности и омического сопротивления проводников и печатных трасс посредством вариации их «геометрии», довольно эффективным известным средством снижения импеданса по ВЧ является применение шунтирующих конденсаторов по питанию, которые позволяют нивелировать эти паразитные параметры. Исключив протекание импульсных токов по трассам питания и «земли», мы тем самым не допустим работы этих проводников в качестве «антенн», передающих помехи. Достигается это, как уже сказано, с помощью грамотной топологии и применения шунтирующих (в данном случае можно сказать «накопительных») емкостей по цепям питания. Термин «накопительный» уместнее применять для емкостей, стоящих в непосредственной близости от узлов, потребляющих большие импульсные токи, при этом питание узла происходит именно за счет энергии, накопленной в конденсаторе, исключая, таким образом, протекание импульсных токов по длинным подводящим трассам питания и «земли».

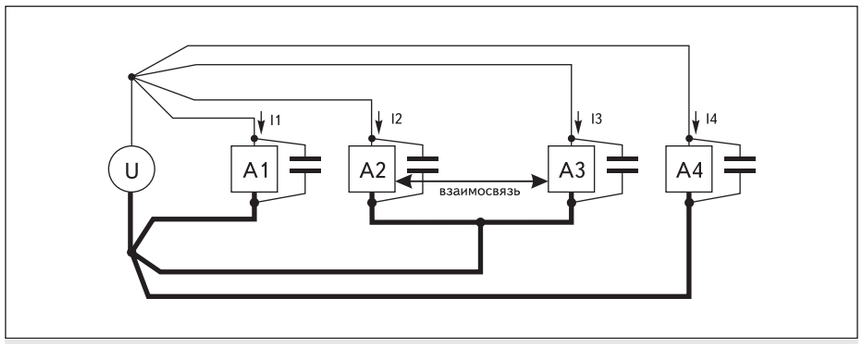

Рис. 14. Пример комбинированной топологии разводки «земли» (эквипотенциальные и «независимые» «земляные» узлы)

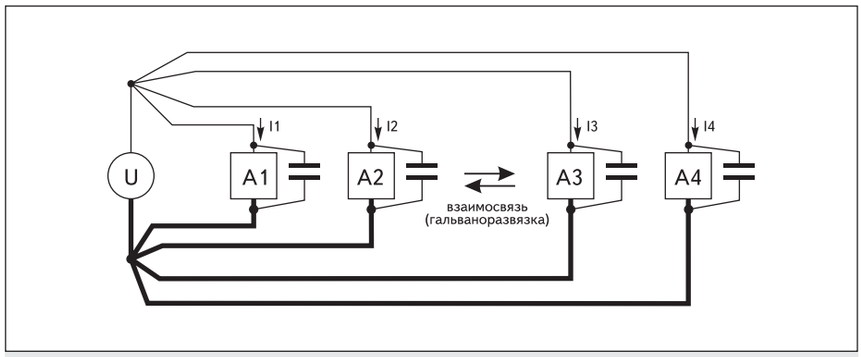

Рис. 15. Пример топологии разводки с «независимыми» «землями» и гальваноразвязанным каналом взаимосвязи блоков

Если блоки связаны аналоговыми сигналами относительно общей «земли», то вариант с эквипотенциальными «землями» из возможных в реализации является единственным. Как альтернативный вариант — возможно построение схемы с гальванической развязкой по управляющим сигналам, но для аналоговых систем это ведет к серьезному усложнению схемотехники. Для цифровых систем, в большинстве случаев, гальваноразвязку можно сделать гораздо проще.

Если блоки независимы, или зависимы некоторые из них, то задача значительно упрощается. В этом случае можно, как показано на рис. 14, сделать эквипотенциальными только «земли» связанных (зависимых) блоков A2 и A3, либо разделить контуры питания, снабдив каждый блок шунтирующей (накопительной) емкостью, а зависимые блоки соединить гальванически развязанным каналом, как показано на рис. 15.

Другой путь решения проблемы развязки по питанию — применение ФНЧ (рис. 16) — более сложен в реализации, но иногда и более эффективен, так как позволяет упростить разводку питания (чаще применяется в ВЧ-схемах), правда, в этом случае также встает проблема обеспечения эквипотенциальности «земель». На практике часто приходится применять оба метода (истина, как обычно, находится где-то посередине).

Вообще говоря, практически все меры по улучшению ЭМС направлены на минимизацию длин трасс протекания импульсных и ВЧ-токов. Можно также сказать, что оптимальным решением будет локализация этих токов в контурах минимальных размеров, образованных компонентами блока и элементами монтажа.

Резюмируя вышесказанное, можно посоветовать делать хотя бы приблизительные инженерные расчеты. Посчитайте максимальные возможные перегрузки элементов, проанализируйте компоновку печатной платы с точки зрения ЭМС и обеспечения терморежимов силовых РЭ и прецизионных узлов.

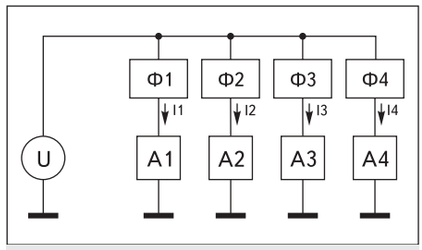

Рис. 16. Пример топологии разводки цепей питания с фильтрами

Проверьте разводку «земли», силовых, прецизионных и высокоимпедансных цепей. Необходимо оценить подверженность прецизионных и высокоимпедансных цепей наведенной помехе. Прецизионными считаем цепи, величина напряжений или токов в которых на шесть и более порядков меньше их абсолютных величин в данной схеме. Высокоимпедансными считаем цепи с модулем полного сопротивления десятки килоом и выше (как правило, это входы аналоговых каскадов, ОУ, АЦП и т. п.). Нужно проанализировать схему на стойкость к кондуктивным помехам по входам/выходам и другим цепям внешних связей. Также нелишним будет проверить стойкость изделия к ударам и вибрациям, при сомнении — провести хотя бы прикидочный статический и модальный анализ, естественно, если есть соответствующие требования ТЗ. Необходимо внимательнее отнестись к монтажу (верификации посадочных мест, маркировке печатной платы, закреплению массивных РЭ), а также к связи компоновки изделия и его дизайна.

Программные проблемы

Ввиду того, что программируемых компонентов превеликое множество, невозможно дать универсальные рекомендации, подходящие во всех случаях и ко всем программируемым компонентам. Зато можно организовать свою работу таким образом, чтобы максимально упростить отладку, поиск ошибок и модификацию программ в будущем. Автор не помнит ни одного случая, чтобы написанная однажды программа никогда бы не улучшалась и не модифицировалась, ведь как известно (из тех же незабвенных «законов Мерфи»): «Любая полезная программа содержит ошибки».

Не претендуя на истину в последней инстанции, «как нужно работать программисту», приведем правила, не раз помогавшие автору эффективно решать и предотвращать проблемы, возникающие при разработке, отладке и модификации ПО.

В программировании существует такое мнение: «Лучший комментарий к программе на малоизвестном языке — программа на известном языке». Так вот — язык, которым в совершенстве владеет каждый из нас, — это естественный язык человеческого общения. Поэтому первое и наиглавнейшее правило при программировании — всегда не лениться комментировать исходные тексты. Причем никогда не откладывать это «на потом», ибо многие идеи, воплощенные в коде, быстро забываются, остается голый код, и понять потом суть программы бывает очень непросто. Таким образом, комментарий должен быть неотъемлемой частью программы, и писать его необходимо одновременно с ней. Чтобы написание комментария не воспринималось как ненужная рутина, приведем несколько правил.

1) Всегда необходимо в заголовке программы (процедуры или функции) поставить блочный комментарий и детально описать суть ее работы, синтаксис (входные/выходные параметры), правила применения. Словесно описать общий алгоритм работы, именно общий, не вдаваясь в подробности, разве что указав некоторые ключевые моменты. А также всегда указывать версию и дату последней модификации.
2) Снабжать исходный текст строчными и блочными комментариями. Причем комментарий не должен повторять исходный текст: он должен пояснять его. Рассмотрим примитивный пример:

```
…
;
mov     A,B     ; копируем регистр B в A
rol     A       ; сдвигаем A влево на 1 разряд
add     A,B     ; складываем A и B
```

Такой комментарий абсолютно бесполезен, так как его информационная ценность равна нулю, он просто дублирует машинную программу, причем на самом низком уровне. Можно даже сказать, что комментарий — это программа на идеальном языке программирования высокого уровня — естественном. Он должен пояснять суть программы, ее алгоритм, дополнять исходный текст.

```
; -- Увеличим время работы XXX втрое.
mov     A,B     ; В регистре «B» исходное время
rol     A       ;
add     B,A     ; Теперь в регистре «B» утроенное время.
;---------------
```

Согласитесь, что такой комментарий (даже вырванный из общего контекста программы) кое-что значит. Он совсем не дублирует математические действия, определяемые операторами машинного языка, он поясняет программу на более высоком, алгоритмическом уровне. Не всегда необходимо писать подробный строчный комментарий, иногда можно пояснить целый блок кода, но краткие строчные комментарии, как правило, бывают полезны.

Очень хороший способ проверить, правильно ли все прокомментировано, — это попробовать «ухватить» суть программы только по одним комментариям.

Следующий «кит», на которого опирается «искусство программирования», это, несомненно, блок-схема. Можно долго спорить: нужна ли она или нет. Твердое убеждение автора: блок-схема абсолютно необходима. Бытует мнение, что блок-схемы, называемые также структурными и функциональными (см. аналогию в разделе «Схемотехнические аспекты»), бесполезны для современных систем и кросс-средств разработки, где используются объектно-ориентированные языки программирования, всевозможные шаблоны и «визарды». Сами же встроенные системы имеют многозадачные операционные среды «на борту», самые модные приложения, вплоть до Java-машин, TCP/IP и прочих отличий ПК. И что-де блок-схема — жуткий атавизм времен Фортрана IV.

Автор не согласен с таким мнением. С помощью блок-схемы возможно описание практически любой программы, в том числе и многопоточных программ и программ для систем с прерываниями. Да — алгоритмические описания стали более сложными (даже визуально они все больше отходят от классических алгоритмов), сложнее стало и взаимодействие с операционной средой, и это тем более веский довод в их пользу. К тому же работа на уровне алгоритма — ключ к созданию эффективного и надежного ПО. До сих пор не угасают споры (уже не один десяток лет!) о преимуществах и недостатках разных языков и методов программирования, но все равно неизменной остается одна истина: лучшая оптимизация — алгоритмическая. То есть косвенно подтверждается почти забытый факт, что любая программа, на любом языке, начинается не с объявления переменных, функций и определений, а именно с алгоритма. По нашему глубокому убеждению, настоящий программист (не путать с «кодером»), реализуя свои идеи, оперирует именно алгоритмами, и среда, в которой он сегодня работает, — это всего лишь инструмент (именно на данный момент). Если разработчик серьезно собирается заниматься «soft-поддержкой» своего изделия, то наличие блок-схемы значительно упростит процесс модификации ПО, не говоря уже о поиске и исправлении ошибок. К тому же, вопрос трансляции встроенного ПО на другие платформы не такой простой, как может показаться. Большинство встроенных систем по-своему уникальны, и зачастую их ПО жестко привязано к аппаратным ресурсам. Действительно сложную программу просто перенести на другую платформу, только если она функционирует под управлением какой-нибудь известной стандартной ОС. А если это «вещь в себе», то неизбежно возникают проблемы переносимости, какими бы высокоуровневыми языками не пользовался программист. Наличие в этом случае грамотной блок-схемы значительно упрощает ситуацию.

Очередной непростой этап разработки надежного встроенного программного обеспечения — тестирование. Это очень ответственный и зачастую долгий процесс. И тут как нельзя кстати будет наличие под рукой блок-схемы. По своему опыту можем сказать, что подавляющее большинство ошибок находится именно при анализе алгоритма, то есть блок-схемы. Методики тестирования ПО у каждого программиста свои. Кому-то нравится многократно прогонять программу в симуляторе, кто-то пользуется аппаратными интерфейсами отладки, кто-то, вооружившись приборами, отлаживает программу в реальном времени. Все эти методы по-своему хороши и обладают каждый своими плюсами и минусами, но главное, чтобы тестированию было уделено самое пристальное

внимание. Лучше всего начинать его в процессе написания программы, что при применении подхода «структурного программирования» не составляет большого труда, так как любой программный «кирпичик», будь то функция, подпрограмма или даже целая ветвь алгоритма, может быть автономно проверен и «доведен до ума».

Ну и, пожалуй, последняя рекомендация — общая культура программирования. Повторимся, что изложенные здесь принципы — личное мнение автора, но он неоднократно убеждался на практике, насколько сильно они влияют на написание качественного и действительно надежного ПО.

Итак, культура программирования. Принципов здесь не так много, и важнейшие из них следующие:

1) При проектировании ПО с успехом можно применять подход к проектированию сложных систем, когда на уровне более общей структурной схемы определяются принципиальные моменты взаимодействия программных и аппаратных ресурсов.
2) Необходимо стремиться, по возможности, к структурному программированию (если нет крайне жестких ограничений по аппаратно-программным ресурсам). Каждая подпрограмма и функция в идеале должна быть автономной законченной и отлаженной единицей, этаким «черным ящиком», дали задание — получили результат.
3) Следствие п. 2 — разделение функциональности по специализированным программным и аппаратным модулям. Как правило, опытный программист встроенных систем эффективно распределяет аппаратные ресурсы программируемого компонента в соответствии с общим алгоритмом работы всего устройства (опять блок схема).
4) В свою очередь, следствие п. 3 — оформление исходных текстов в виде относительно небольших и относительно самостоятельных программ, подпрограмм и процедур, хранящихся в отдельных файлах. Написание всей программы в виде одной большой «простыни» — признак «плохого тона». К тому же с такой программой, как правило, крайне неудобно работать.
5) Комментарии!!! Как общие алгоритмические, так и применительно к входным/выходным параметрам подпрограмм и функций (см. выше).
6) Все глобальные константы и переменные, необходимые для работы программы, должны быть описаны в одном месте (файле). Как правило, большинство систем программирования поддерживают многофайловые проекты (когда исходный тест программы — совокупность файлов: программных модулей и процедур). Так вот именно в одном файле должны быть декларированы и подробно описаны глобальные константы и переменные, и, соответственно, изменяться они должны только в этом одном файле.
7) Никогда не использовать в исходных текстах числовые константы. Все числовые константы должны быть именованы и описаны в соответствии с п. 6.
8) Использовать только подходящие по смыслу имена подпрограмм, переменных и адресные метки. «Абракадабра» в метках и именах или безликие идентификаторы (типа label1, label2, file1 и т. п.) крайне затрудняют понимание исходного текста программы.
9) Головная программа проекта или файл описания переменных должны содержать краткое функциональное описание всех подпрограмм, процедур и функций.
10) Снабдить головную программу или файл описания пояснениями принципиальных особенностей проекта (например, какие должны быть ключи компиляции и наименование среды разработки, какова должна быть конфигурация программируемого компонента, если она задается дополнительными средствами, и т. п.). А также описать в общих словах суть и алгоритм работы всей программы.
11) Визуально структурировать исходный текст, пользуясь отступами, выравниванием и т. п. Искусно форматированный исходный текст естественным образом позволяет «читать» программу функциональными блоками.
12) Всегда делать внятные и в меру подробные описания блок-схем, переменных и констант, программных текстов и т. п. «Исходник» должен быть понятен абсолютно стороннему программисту. Такой подход позволит в дальнейшем, даже по прошествии некоторого времени, легко вспомнить суть программы и подправить ее в случае необходимости. При корпоративной работе (когда проект может «кочевать» от разработчика к разработчику) вообще умолчим, здесь это условие просто обязательно.
13) Имена переменных, констант, подпрограмм и меток в блок-схеме и исходном программном тексте должны совпадать.
14) Блок-схема должна быть актуальной, то есть всегда должно быть соответствие между блок-схемой и программным текстом. Сначала необходимо вносить изменения в блок-схему и только потом — в программный текст. Если действовать наоборот, то есть большой риск внести в программу ошибки (иначе получится, как в шутке: «В новой версии программы исправлены старые ошибки и добавлены новые»).

Проблемы дизайна

Одной из основных точек соприкосновения проблем дизайна и обеспечения надежности является эргономика. Помимо внешнего вида изделия как воплощения полета мысли дизайнера, существует еще и эргономика — как направление создания удобных для человека вещей. Частенько «отстает» именно эргономика. За множеством модных интерфейсов и «лампочек» часто не найти нужных органов управления, которые, казалось бы, должны быть на первом месте. Излишне броская или не в меру яркая индикация (как, впрочем, и излишне блеклая и незаметная) часто мешает восприятию необходимых данных. Как важнейший элемент, ответственный за эффективность взаимодействия человека с техникой, именно эргономика влияет на надежность этого взаимодействия. Не нужно забывать, что в соответствии с теми же «законами Мерфи»: «Любая система, зависящая от человеческой надежности, — ненадежна», и задача разработчика свести влияние «человеческого фактора» к минимуму.

На втором месте стоит сопряжение дизайнерских решений с конструкционными и технологическими решениями проектируемого изделия. Особенности используемых материалов, результаты конструкционных расчетов (тепловых, прочностных, ЭМС и т. п.) и технологии производства и сборки необходимо учитывать при создании дизайн-концепции изделия.

Задачи тестирования

На самом деле тестирование технических решений идет всегда параллельно с разработкой (аналогично рекомендации начинать тестирование ПО уже в процессе его написания). В принципе, разработка устройства в некотором роде и есть мысленное тестирование на возможность работы в разных условиях. Помимо традиционного макетирования и программного моделирования, всегда полезно проверять работу узлов схемы в условиях, приближенным к рабочим. Гораздо меньше сюрпризов преподносит система, построенная из отлаженных и испытанных ранее узлов. Это, конечно, не отменяет финальных испытаний устройства на соответствие требуемым в ТЗ характеристикам. Для облегчения процесса тестирования, как в последующем и процесса обслуживания или ремонта, необходимо предусмотреть на схемах и печатных платах необходимый набор контрольных точек, диагностических разъемов и т. п. При массовом производстве устройства необходимо обеспечить процедуру выходного контроля изделий по ряду важнейших параметров и, естественно, разработать грамотную методику тестирования. Схемотехника и конструктив должны быть выполнены с учетом возможности подключения изделия к тестовым стендам. Как правило, аппаратно-конструктивные и программные решения самих стендов исключительно индивидуальны для каждого нового изделия. Нужно ли говорить о том, что при создании средств контроля качества необходимо придерживаться правил обеспечения надежного проектирования?

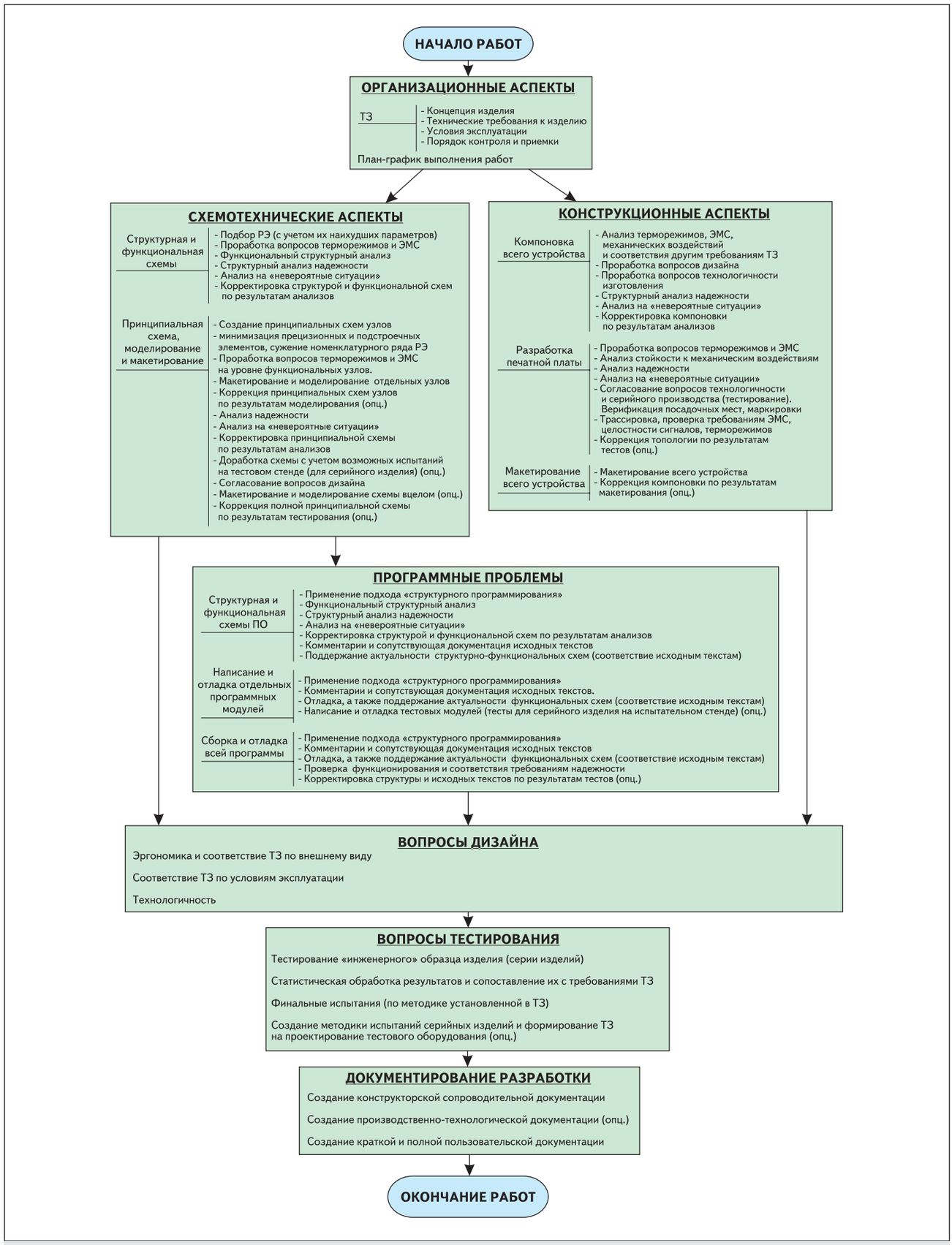

Рис. 17. Ключевые этапы обеспечения надежности РЭС при разработке

Особенно строго следует подходить к разработке методики испытаний. Этот вопрос не так прост, как может показаться на первый взгляд, и при его решении нужно учитывать требования ТЗ. По крайней мере, в соответствии с ними должен быть выполнен контрольный стенд для выборочного полного тестирования изделий. В некоторых случаях бывает нелишне подумать и о входном контроле качества компонентов. Качественная проверка, как и разработка вообще, немыслима без парка хороших измерительных приборов, которые далеко не всегда есть в наличии. Видимо ввиду их дороговизны разработчики часто пользуются весьма ограниченным набором приборов. По правде сказать, многие задачи действительно решаются методами косвенных измерений. Тут главное точно знать: что хотим измерить, с какой точностью, и какими способами этого можно достичь. Контролировать с помощью приборов результат разработки необходимо всегда, не надеясь на характеристики комплектующих (так как кроме характеристик есть еще схемотехника, физика, банальный брак РЭ и многое другое).

Ну и напоследок одно замечание: удачно изготовленный один образец еще ничего не значит. Статистика и еще раз статистика. Чтобы дать заключение о работоспособности изделия (а в особо ответственных случаях — даже отдельного узла), необходимы испытания ряда аналогичных изделий. Часто, к сожалению, таким испытанием служит выход устройства «в серию». Чтобы избежать проблем с «серийным» изделием, процесс тестирования нужно проводить тщательно и ответственно.

Оформление КД и ПД

К сожалению, созданием качественной конструкторской и пользовательской документации сегодня пренебрегает подавляющее большинство разработчиков. А ведь хорошая конструкторская документация — это и дальнейшая техподдержка изделия, и его модификация, и анализ отказов, и многое, многое другое. Создание грамотной конструкторской документации сопряжено с трудностями использования САПР и библиотеками УГО в отечественных стандартах, но эти трудности преодолимы (есть и коммерческие библиотеки, да и самому можно определить необходимый минимум в соответствии с ГОСТ). Игнорирование ГОСТ при создании конечной КД, на наш взгляд, крайне нежелательно, так как, во-первых, вносит путаницу (и ошибки) в документацию, а во-вторых, значительно затрудняет техническое сопровождение изделия (особенно разными специалистами).

Что касается пользовательской документации, то она совершенно необходима во избежание проблем, связанных с неправильной эксплуатацией изделия. Практически же пользовательскую документацию почти всегда читают только при возникновении каких-либо проблем. Как всегда, подтверждается один из «законов Мерфи»: «Если ничто другое не помогает — прочтите, наконец, инструкцию». В свете вышесказанного можно рекомендовать, во-первых, разработку устройства с дружественным пользователю интерфейсом, всевозможными «защитами от дурака», а во-вторых, создание краткой памятки пользователя (наряду с полным руководством, которое также необходимо). Написание приличного руководства пользователя доступно далеко не всем «технарям», тут можно рекомендовать руководствоваться примерами, помощью «гуманитариев» и здравым смыслом. Как бы то ни было, написание ПД — абсолютно необходимая часть работы.

Напоследок можем лишь добавить, что разработка не считается законченной, если она не документирована.

Заключение

Поднятые проблемы проектирования хоть и затрагивают вопрос надежности разрабатываемой аппаратуры, но больше имеют отношение к квалификации инженера-разработчика. В этой статье автор поделился некоторыми методами повышения качества работы инженера-разработчика РЭС. Ключевые моменты, многие из которых ускользают от внимания разработчика и на которые стоит обратить внимание, отображены на схеме (рис. 17).

Статья, конечно, не охватывает всех проблем, возникающих при разработке РЭС и влияющих на ее надежность. Не были, например, затронуты специальные конструкционные и схемотехнические методы обеспечения надежности, такие как различные способы резервирования, проектирование по результатам анализа критичности отказов, разработка устойчивых к сбоям программных решений и т. д. «За кадром» (или затронутыми вскользь) остались также многие конструкционные методы повышения надежности, такие как компоновка изделия, вопросы защиты от ударов и вибраций, экранирование, проектирование печатных плат, пыле- и влагозащита и много другое. Это специальные вопросы, о которых при проектировании РЭС не стоит забывать, и ответы на которые заинтересованный читатель может найти в специальной литературе. Но, тем не менее, наиболее «популярные» общие проблемы обеспечения надежности РЭС при разработке автор постарался осветить. В некоторых случаях даны конкретные рекомендации, в иных просто указано на проблему, и тут уже задача разработчика не упустить ее из виду и проработать методы ее решения. Приведенный в конце статьи список литературы поможет более полно разобраться со многими, затронутыми здесь вскользь, проблемами. Пересказывать изложенный в этих книгах фундаментальный материал — не имеет смысла. Заинтересовавшийся читатель найдет в приведенных трудах исчерпывающее объяснение многих вопросов обеспечения надежности РЭС при разработке. В любом случае данная статья задумывалась не как набор инструкций «как нужно делать», а как некий «сборник рецептов» или даже «памятка». Воспринимать эти советы или нет — личное дело каждого. Скажем лишь, что эти рекомендации — результат работы над многими проектами. Их соблюдение не раз облегчало автору его инженерную деятельность. Предвидим возражения: «Но ведь разработка будет длиться долго и постоянно будет «спотыкаться» об эти правила!» Да, не всегда все просто, но когда соблюдение подобных правил войдет в систему, вы сами уже не сможете работать по-другому, потому что вознаграждением будет — безотказная работа спроектированной аппаратуры и минимум проблем с заказчиком, ведь, как известно, лучше «медленно запрягать, но быстро ехать». К тому же такой подход обеспечивает регламент ведения работ и четкое понимание того, что, когда, зачем и как нужно делать. Согласитесь, никому не захотелось бы покупать автомобиль, изготовленный на заводе, где правит принцип «сделаем как-нибудь». Так почему же при проектировании РЭС мы, разработчики, такое себе частенько позволяем?

В заключение пожелаем коллегам-инженерам побольше успешных проектов, полезных и надежных устройств. Все-таки хочется, чтобы техника работала на нас, а не мы на нее.

Литература

1. Уразаев В. Г. Влагозащита печатных узлов. М.: Техносфера, 2006.
2. Усатенко С., Каченюк Т. К., Терехова М. В. Выполнение электрических схем по ЕСКД. Справочник. М.: Издательство стандартов, 1989.
3. Хоровиц П., Хилл У. Искусство схемотехники / Перевод с англ. Том 1–3. М.: Мир, 1993.
4. Воробьев Е. Как читать data sheet на микросхемы // Электронные компоненты. 2008. № 3.
5. Маквецов Е. Н., Тартаковский А. М. Механические воздействия и защита радиоэлектронной аппаратуры. Учебник для вузов. М.: Радио и связь, 1993.
6. Тугов Н. М., Глебов Б. А., Чарыков Н. А. Полупроводниковые приборы. Учебное пособие для вузов. М.: Энергоатомиздат, 1990.
7. Воронин П. А. Силовые полупроводниковые ключи. М.: Издательский дом «Додэка-XXI», 2001.
8. Справочник конструктора РЭА: Общие принципы конструирования / Под ред. Р. Г. Варламова М.: Советское радио, 1980.
9. Павлов В. Н., Ногин В. Н. Схемотехника аналоговых электронных устройств. Учебное пособие для вузов. М.: Радио и связь, 1997.
10. Левин Б. Р. Теория надежности радиотехнических систем. М.: Советское радио, 1978.