Научная статья
Original article
УДК 004.031.6

ЭНЕРГОСБЕРЕЖЕНИЕ ВСТРОЕННЫХ СИСТЕМ

ENERGY SAVING IN EMBEDDED SYSTEMS

Гисматов Азат Рустемович, магистрант 2 курс, кафедра промышленной информатики, МИРЭА - Российский технологический университет, Россия, г. Москва
Научный руководитель: Ревякин Вадим Александрович

Gismatov Azat Rustemovich, Graduate Student 2nd year, Department of Industrial Informatics, MIREA - RUSSIAN TECHNOLOGICAL UNIVERSITY, Russia, Moscow
Scientific adviser: Revyakin Vadim Aleksandrovich

Аннотация: Энергопотребление встроенных систем является серьезной проблемой. Всегда существует потребность в продлении срока службы батареи и снижении воздействия системы на окружающую среду. Исторически это была только аппаратная проблема, но те времена прошли. В современных встроенных системах программное обеспечение берет на себя все большую ответственность за управление питанием. В данной статье рассматривается, как достигается управление питанием во время работы устройства, и рассматриваются методы, используемые для минимизации энергопотребления, когда устройство неактивно.

В целом существует два контекста, в которых может рассматриваться энергопотребление устройства: когда оно используется и когда оно находится

в режиме ожидания. В первом случае ключевым требованием является активное управление питанием; в последнем случае развертывание режимов центрального процессора (далее - ЦП) с низким энергопотреблением может быть выгодным.

Annotation: Power consumption by embedded devices is a critical issue. There is always a need to extend battery life and reduce the environmental impact of a system. Historically, this was purely a hardware issue, but those days are past. In modern embedded systems software takes an increasing responsibility for power management. This article reviews how power management is achieved while a device is operating and looks at the techniques employed to minimize power consumption when a device is inactive.

There are broadly two contexts in which a device's power consumption may be considered: when it is in use and when it is idle. In the former, active power management is the key requirement; in the latter, the deployment of low power central processing unit modes may be advantageous.

Ключевые слова: управление питанием, встроенные системы, энергопотребление, режим энергосбережения, центральный процессор.

Key words: power management, embedded systems, power consumption, power saving mode.

Программное обеспечение может предпринять несколько мер, чтобы свести энергопотребление к минимуму:

- выключение периферийного устройства, когда они не используются;
- динамическое масштабирование напряжения и частоты ЦП в соответствии с текущими требованиями к производительности.

Отключение периферийных устройств. Совершенно очевидно, что лучший способ сэкономить энергию при использовании любого электрического или электронного устройства - это просто выключить его. Таким образом, логично проектировать электронные системы так, чтобы

периферийные устройства и подсистемы могли включаться и выключаться программным обеспечением по мере необходимости [1, с. 11].

Это средство не так просто, как кажется, поскольку некоторые типы периферийных устройств, например, сетевой интерфейс, требуют определенного времени для настройки при включении. Эта задержка может быть неприемлемой, если периферийное устройство постоянно включается и выключается. Также бывают ситуации, когда периферийное устройство может продолжать передавать данные после того, как ЦП закончило их адресацию; преждевременное отключение питания приведет к потере данных.

Динамическое масштабирование напряжения и частоты. Инженеру-программисту не сразу очевидно, как напряжение ЦП и тактовая частота влияют на энергопотребление. Вообще говоря, чем ниже частота работы, тем ниже энергопотребление. Это можно рассматривать с точки зрения того, сколько работы может выполнить данная часть программного обеспечения. Например, представьте, что ЦП необходимо выполнить 100000 инструкций некоторого программного обеспечения, чтобы выполнить задание, и это нужно выполнять каждую секунду. Если бы ЦП работал с тактовой частотой, которая позволяла бы ему выполнять миллион инструкций в секунду, он был бы способен выполнять в 10 раз больше требуемой работы [3]. Таким образом, снижение тактовой частоты на эту величину приводит производительность в соответствие с требованиями и оптимизирует энергопотребление.

Режимы низкого энергопотребления. Когда устройство не используется, оно может быть полностью отключено. Для этого требуется небольшая поддержка программного обеспечения, хотя некоторым устройствам может потребоваться некоторая информация о состоянии, сохраняемая при отключении питания [2, с. 121]. Единственная проблема заключается в том, что запуск полностью выключенного устройства может занять некоторое время. Даже с облегченной операционной системой реального времени загрузка современных крупных приложений может занять несколько секунд.

Альтернативой отключению питания является своего рода спящий режим. Спящие режимы, используемые в большинстве ноутбуков:

- режим ожидания - ЦП и периферийные устройства отключены, но питание подается на оперативную память. Преимущество этого режима в том, что пробуждение происходит очень быстро, но недостатком является то, что питание продолжает потребляться, поэтому существует ограничение на то, как долго устройство может находиться в режиме ожидания [4];
- спящий режим - данные записываются на диск, и система выключается. Преимущество этого режима в том, что в нем нет постоянной утечки энергии, поэтому состояние гибернации может поддерживаться неограниченное время. Однако пробуждение занимает больше времени, так как данные необходимо копировать обратно в оперативную память, но это все же намного быстрее, чем холодная загрузка [4].

Все чаще встроенные процессоры имеют встроенные спящие режимы.

Внедрение управления питанием. Со всеми аспектами разработки программного и аппаратного обеспечения неразумно изобретать уже существующие решения. Если у кого-то есть эффективное решение проблемы или реализация алгоритма, в большинстве случаев не имеет смысла писать свой код программы. Повторное использование существующей интеллектуальной собственности намного более рентабельно. Управление питанием хорошо изучено, поэтому реализации легко доступны.

Операционные системы реального времени. Функциональность может быть реализована в коде приложения, это громоздко и не очень логично. Гораздо разумнее, чтобы операционная система включала структуру управления питанием, поскольку, в частности, корректная работа драйверов может сильно зависеть от мер по энергосбережению, и операционная система может легко справиться с этим. На рисунке 1 реализовано управление питанием в современной операционной системе реального времени с поддержкой питания.

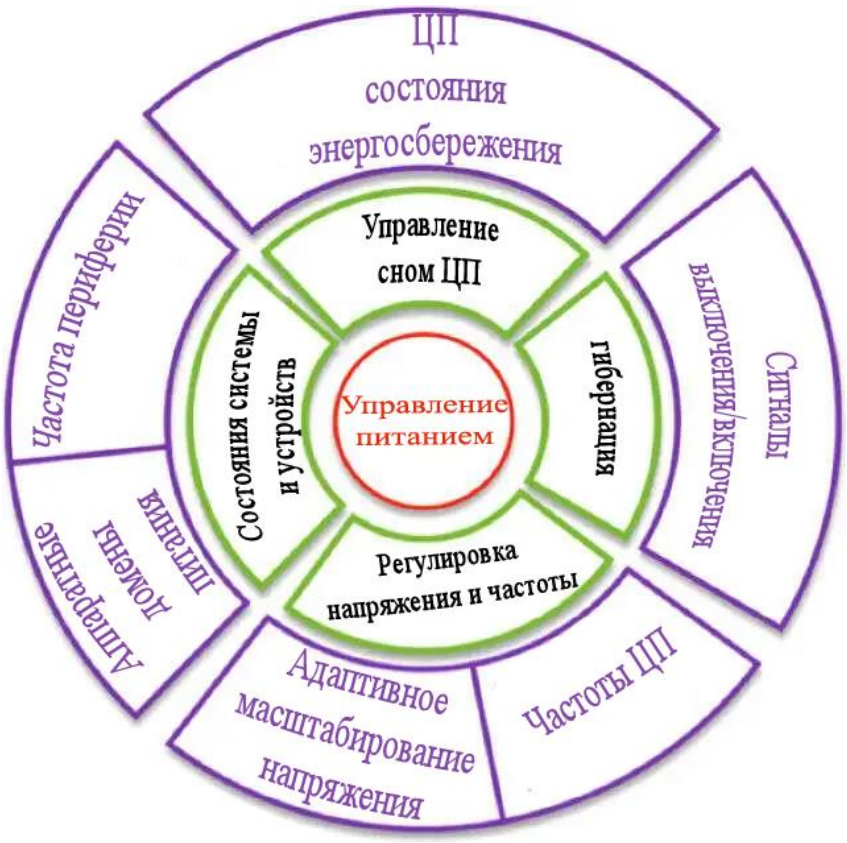

Рисунок 1. Управление питанием операционной системе реального времени

Операционная система реального времени не может просто оптимизировать мощность некоторого кода приложения, поскольку она не знает о требованиях к производительности кода, а производительность выполнения в первую очередь влияет на энергопотребление [5].

Обычный подход заключается в том, что разработчик анализирует приложение и определяет ряд вариантов использования - конкретных ситуаций, в которых используется устройство, которые можно считать состояниями. Каждый вариант использования требует определенного уровня производительности и определенного сочетания доступных ресурсов. Производительность может определяться конкретной комбинацией напряжения, частоты, и они называются рабочими точками. Каждому варианту использования также назначается рабочая точка. Платформа управления питанием принимает информацию о вариантах использования в

качестве входных данных и, следовательно, может определить, как регулировать питание в любой момент времени.

Существует множество причин, по которым необходимо оптимизировать энергопотребление устройства. Но среди самых распространенных стимулов - требование соответствовать ожиданиям пользователей. Зачастую пользователям хотелось бы, чтобы время автономной работы их повседневных устройств было весомым.

Использованные источники:

1. Perez F. M. Embedded Energy Management System for the ICT Saving Energy Consumption. — 2014. P. 10-12.
2. Schmitz M.T. System-Level Design Techniques for Energy-Efficient Embedded Systems. — 2016. P. 119-123.
3. Комплексный подход управления питанием. [Электронный ресурс]. URL:https://journals.sagepub.com/doi/10.1155/2011/807091 (дата обращения: 26.03.2022).
4. Политики динамического управления питанием для встроенных систем [Электронный ресурс]. URL: https://cseweb.ucsd.edu/~gdhiman/Gaurav_files/CSE-237A/TopicResearch/DPMPolicies.htm (дата обращения: 26.03.2022).
5. Управление питанием встроенных систем. [Электронный ресурс]. URL: https://www.embedded.com/power-management-in-embedded-software/ (дата обращения: 26.03.2022).

References:

1. Perez F. M. Embedded Energy Management System for the ICT Saving Energy Consumption. — 2014. P. 10-12.
2. Schmitz M.T. System-Level Design Techniques for Energy-Efficient Embedded Systems. — 2016. P. 119-123.
3. A Comprehensive Approach to Power Management. [Electronic resource]. URL:https://journals.sagepub.com/doi/10.1155/2011/807091 (accessed: 26.03.2022).

4. Dynamic Power Management Policies for Embedded Systems [Electronic resource]. URL: https://cseweb.ucsd.edu/~gdhiman/Gaurav_files/CSE-237A/TopicResearch/DPMPolicies.htm (accessed: 26.03.2022).
5. Power management in embedded systems. [Electronic resource]. URL: https://www.embedded.com/power-management-in-embedded-software/ (accessed: 26.03.2022).

Для цитирования: Гисматов А.Р. ЭНЕРГОСБЕРЕЖЕНИЕ ВСТРОЕННЫХ СИСТЕМ // Научно-образовательный журнал для студентов и преподавателей «StudNet» №4/2022.