1. *Bramfitt B. L.* Metallographer's Guide – Practices and Procedures for Iron and Steels / B. L. Bramfitt, Arlan O. Benscoter. – ASM International, 2002. – 354 p.
2. *Hosseini H.* Characterization of microstructures and mechanical properties of Inconel 617/310 stainless steel dissimilar welds./ H. Hosseini, M. Shamaniana, A. Kermanpura // Materials Characterization. – Vol. 62, Issue 4. – 2011. – P. 425 – 431.
3. Cast-iron metallographic structure by computer picture processing system / Wang Zhiping, Lu Yang, Wu Chenwed, Xu Jianlin, Yang Xinzhuang // Journal of Cansu University of Technology, – Vol.E-1, No.1. – 1997. – p. 29 – 32.
4. VideoTesT Ltd "Application of Image analysis software "VideoTesT-Morphology" in mycology, phytopathology and industrial microbiology" [Электронный ресурс] – Режим доступа: http://www.videotest.ru/en/article/ view/48/category/11 (26.08.2007)
5. VisionPE Metlab Corporation. Image Analysis, [Электронный ресурс] – Режим доступа: http://www.metlabcorp.com/image_analysis.html (31.01.2009)
6. SIAMS: Index of/products/siams700 [Электронный ресурс] – Режим доступа: http://www.siams.com/products/siams700/ (14.01.2013)
7. *Журавлев Ю. И.* Распознавание. Математические методы. Программная система. Практические применения / Ю. И. Журавлев, В. В. Рязанов, О. В. Сенько. – М. : Фазис, – 2005. – 159 с.
8. *Gonzalez R.S.* Digital image processing. – 2nd ed. / R.S. Gonzalez, R.E. Woods. – USA: Prentice, 2002. – 703 p.
9. *Емельянов В. А.* Нейросетевой метод определения количественных характеристик металлов / В.А. Емельянов // Радіоелектронні і комп'ютерні системи. – 2010. – № 4(45). – С.169 - 173.
10. *Suzuki Kenji* Artificial Neural Networks: Architectures and Applications [Text] / Kenji Suzuki. – InTech, 2013. – 256 p.
11. Сайт: База данных микроструктур металлов и сплавов. [Электронный ресурс] – Режим доступа: http://www.microstructure.ru/rudbview

Поступила 3.02.2014р.

УДК 004.382.76

Д.В. Стась, Київ

ПРОГРАМНІ ЗАСОБИ ОЦІНКИ ЕНЕРГОЗБЕРЕЖЕННЯ МОБІЛЬНИХ ПРИСТРОЇВ

Abstract. In this paper, the theoretical information about mobile device energy consumption was presented. The methods of energy saving implemented in modern mobile operating systems were reviewed. Programmatic methods for measuring energy consumption of mobile devices were investigated and compared based on iOS and Android mobile operating systems.

Постійно зростаючі можливості смартфонів, планшетів, компактних ПК вимагають від виробників мобільних пристроїв використання спеціальних

методів збереження енергії. Використання високотехнологічних апаратних компонентів у поєднанні з оптимізованими операційними системами та передвстановленим програмним забезпеченням дозволяє ефективно використовувати енергоресурси мобільних пристроїв.

На сьогоднішній день оптимізувати програмне забезпечення набагато дешевше, ніж вкладати мільярди доларів на розробку нових аккумуляторних батарей або процесорів з використанням новітніх технологій виробництва або інших компонентів.

Одна з перших моделей вимірювання енергоспоживання була запропонована індійським дослідником Авіралом Кумаром Тіварі[1]. Ця модель враховувала базові витрати та колові накладні витрати. Базові витрати – це середній струм, що споживає процесор під час безперервного виконання інструкцій. Накладні витрати – це витрати на зміну стану електричного кола під час виконання послідовності інструкцій. Як тільки струм стає відомим, можна визначити енергію за наступною формулою:

$$E = (I * Vdd) * (n * \tau)$$

де E – енергія, I – струм, Vdd – вольтаж джерела енергії, n – число циклів, τ - період циклу.

Для вимірювання величини струму використовують звичайний осцилограф. Інструкція, для якої потрібно виміряти струм, ставиться у нескінченний цикл, після чого знімається показник струму з осцилографу. Проте, якщо в циклі буде 2 або більше інструкцій, сумарний струм буде більшим ніж сума струмів цих операцій, виміряних окремо. Виходячи з цього, загальна спожита програмою енергія – це сума базових витрат та накладних витрат енергій, виміряних для всіх інструкцій.

Цю методику можна застосовувати для побудови таблиць, за допомогою яких можна оцінювати енергоспоживання. Ці таблиці наповнюються емпіричними значеннями, здобутими в ході експериментів. Проте, через велику кількість інструкцій у сучасних процесорах побудова таких таблиць може ускладнитись.

Модель дослідження енергоспоживання, запропонована Расселом[1], використовує процесори на базі архітектури i960 (80960JF та 80960HD). Метою цієї моделі є знаходження середньої потужності процесора за цикл разом з часом виконання програми (у циклах), добуток цих двох величин і буде загальною енергією, що споживає програма. Модель енергії представляється наступною формулою:

$$E = \int_{t_0}^{t_0+T} P(t)dt$$

де T – час виконання програми, P(t) – миттєва потужність, t_0 – час початку виконання програми.

Середню потужність можна визначити за формулою:

$$P_{\text{ср}} = \frac{1}{T}\int_{t_0}^{t_0+T} P(t)dt$$

Виходячи з цього, формулу енергії можна записати так:

$$E = T * P_{\text{ср}}.$$

Ця модель є більш перспективною ніж та, що була запропонована дослідником Тіварі, бо вона враховує величину середньої потужності процесора на цикл. Так як інструкція може зайняти більш ніж 1 цикл процесора, більш доцільно спиратись на середню потужність на цикл. Модель Рассела є набагато простішою у реалізації та не потребує довготривалих експериментів, як модель Тіварі. Результати вимірювань дають максимальну похибку 8%.

Методи зниження енергоспоживання поділяються на апаратні, програмні та комбіновані. Апаратні методи зазвичай використовуються виробниками процесорів та інших компонентів мобільних пристроїв. Оптимізовані компоненти сприяють кращому енергозбереженню та надають базовий виграш у їх використанні в мобільних платформах.

Існує багато розробок комбінованих методів зниження енергоспоживання, які використовують як апаратні, так і програмні компоненти для досягнення своєї мети. Більшість цих методів орієнтовані на оптимізацію бездротових комунікацій, які зазвичай споживають значну кількість енергії мобільного пристрою.

Один з таких методів – CoolSpots – дозволяє пристрою переключатись на більш економний вид зв'язку у разі його доступності [2]. CoolSpots може бути використаний з Wi-Fi мережами у комбінації з технологією Bluetooth. Наприклад, у приміщенні встановлена Wi-Fi мережа та декілька Bluetooth-точок. При потраплянні мобільного пристрою у зону дії Bluetooth-точки (так звана CoolSpot – "холодна точка"), цей пристрій автоматично переключається з мережі Wi-Fi на Bluetooth. При цьому зберігаються всі встановлені з'єднання, але знижується швидкість обміну даними. Використання технології Bluetooth дозволяє знизити енергоспоживання за рахунок зменшення відстані передачі даних та швидкості. З появою окремої специфікації Bluetooth 4 Low Energy, орієнтованої на підвищене енергозбереження, витрати енергії можна скоротити до 4 разів. При цьому, коли мобільний пристрій залишає зону дії Bluetooth-точки, програмне забезпечення автоматично під'єднує пристрій до Wi-Fi мережі.

Інший метод підвищення енергозбереження передбачає використання спеціальних сигналів для увімкнення бездротового зв'язку у разі необхідності (наприклад, при отриманні вхідного VoIP-дзвінка)[3]. Введення такого сигналу пов'язане з надзвичайно інтенсивним енергоспоживанням навіть у режимі очікування та без передачі даних. Згідно цього методу, спеціальний сервер, який отримує VoIP-двінок, виконує виклик абонента за допомогою

GSM/3G мережі. Програмне забезпечення мобільного пристрою розпізнає цей дзвінок як системний та підключається до вільної Wi-Fi мережі без попередження користувача. Як тільки виконується з'єднання, мобільний пристрій отримує VoIP-виклик через Wi-Fi мережу. Таким чином, у період очікування мобільний пристрій не витрачає енергію на зв'язок з Wi-Fi мережею, і використовує її лише у разі отримання вхідного VoIP дзвінка чи схожої події.

Програмні методи енергозбереження зазвичай реалізовуються як частина операційної системи мобільного пристрою. Ці методи направлені на оптимізацію всіх модулів системи, що включають найголовніші компоненти: комунікаційні компоненти, компоненти відтворення інформації, компоненти обробки графічних сигналів та компоненти сторонніх розробників. Оптимізація кожного з цих компонентів дозволяє ефективно використовувати енергоресурси та збільшувати час автономної роботи мобільного пристрою. Кожна з сучасних операційних систем має свої методи, у яких є свої переваги та недоліки.

Згідно глобальної статистики, найбільш популярними мобільними операційними системами є iOS та Android. Ці операційні системи використовують різні підходи до керування енергозбереженням, проте в них також присутні й схожі методи.

Головна проблема енергозбереження у мобільних пристроях – це багатозадачність. Це, з точки зору користувача, можливість користуватись декількома додатками одночасно та швидко переходити від одного додатку до іншого (наприклад, музикальний плеєр та веб-браузер). З точки зору операційної системи важливо, щоб кожен з додатків, знаходячись у фоновому режимі, споживав якнайменше ресурсів, а також відгукувався на повторний запуск з фонового режиму якнайшвидше. Такий підхід, у поєднанні з жорсткими правилами для розробників, дозволяє створити найкращу комбінацію програмної та апаратної конфігурації для збереження енергії на якомога довший час.

Саме такого принципу притримуються розробники операційної системи iOS – компанія Apple. У кожного додатку є свій життєвий цикл, що являє собою лінію часу, протягом якої додаток приймає різні стани, серед яких основні: активне, неактивне, призупинене, вимкнене. Якщо додаток не вимагає особливих привілегій (наприклад, програвання аудіо-файлів у фоновому режимі, VoIP-телефонія або геолокація), то після закриття він переходить у неактивний режим з можливістю виконання невеликої кількості операцій протягом малого проміжку часу (до 1 хвилини). Якщо додаток не потребує виконання цих операцій, то система одразу переводить програму у призупинений стан. У такому стані додаток не може виконувати будь-які операції та отримувати повідомлення про події. Згодом, система звільняє пам'ять від призупинених додатків, щоб надати ресурси тим програмам, що недавно увійшли в фоновий режим.

Особливі привілегії для аудіо-додатків дозволяють їм перебувати у

фоновому режимі допоки відтворюється звук на пристрої. VoIP-додатки можуть періодично встановлювати зв'язок з сервером для синхронізації, а геолокаційні додатки отримують актуальні дані про місцерозташування пристрою, як тільки вони змінюються. Тобто, додатки виконують деяку роботу у фоновому режимі тільки тоді, коли це справді потрібно.

В операційній системі Android використовується своя модель багатозадачності. Її суть полягає в тому, що система в будь-який момент може запустити або завершити роботу будь-якого додатку, що знаходиться у фоновому режимі. Операційна система намагається визначити найменш "потрібні користувачу" додатки, які в даний момент запущені в фоновому режимі, після чого відбувається збереження актуального стану додатку з подальшим його завершенням. Сам процес завершення відбувається дуже жорстко, без попередження, тому додаток може втратити деякі дані, які в цей час оброблялись. Згодом, система може запустити вже завершений додаток, у останньому успішно збереженому стані (при цьому, збереження стану не гарантується і залежить в більшості випадків від ОС Android, а не розробника додатку, як в iOS).

Ця модель поведінки не гарантує високої енергоефективності через те, що у кожний момент часу в системі присутні запущені додатки або процеси, які виконують роботу в фоновому режимі, споживаючи системні ресурси.

До речі, в ОС Android немає чіткого зв'язку додатку та процесу: на два додатки може бути створений один спільний процес та навпаки. Це негативно впливає на керування енергозбереженням через відсутність чіткої логіки роботи з такими незвичними ситуаціями. Подібні концептуальні та архітектурні проблеми призводять до того, що розробник ОС змушений вимагати від виробників конкретну апаратну конфігурацію для пристроїв, що будуть випускатись. Для розробників програмного забезпечення під платформи iOS та Android існують спеціальні інструменти для відстежування навантаження на систему та споживання ресурсів.

Продуктивність Android-додатків можна відстежувати за допомогою інструменту Systool. Він відображає на графіках у вигляді ліній часу споживання ресурсів процесора, ядра та деяких інших компонентів системи. Серед недоліків цього інструменту можна зазначити те, що він не здатний розділяти на графіку споживання ресурсів саме кодом додатку та системними викликами. Тому, якщо деяка частина системних функцій ОС недостатньо оптимізована, розробник не одразу зможе визначити джерело проблем енергоспоживання.

Інструмент для відстежування продуктивності додатків на платформі

iOS під назвою Instruments містить в собі багато компонентів для різних типів задач – профілювання часу (скільки часу займає виклик тієї чи іншої функції), моніторинг системних ресурсів (процесор, пам'ять чи батарея), моніторинг витіків у пам'ять, діагностика використання мережі та багато інших. Розробник може використати кожен з цих інструментів для виявлення проблем перед публікуванням додатку. Серед модулів Instruments також є модуль діагностики енергоспоживання. Розробнику надається графік у вигляді лінії часу, а також докладний звіт про енергоспоживання конкретним додатком. На цей графік наносяться значення у межах від 0 до 20 з обраною періодичністю. Це абстрактні значення енергоспоживання у конкретний проміжок часу. Значення 20 є максимально допустимим для додатку і при наближенні до цього значення, операційна система може прийняти рішення про закриття додатку для економії енергії. Тому надзвичайно важливо відстежити проблеми з енергозбереженням ще на етапі розробки, бо це дозволить уникнути непорозумінь з користувачами додатку.

Також на обох платформах доступні системні функції, за допомогою яких можна відстежувати енергозбереження у самому додатку. Цю інформацію можна виводити на екран пристрою або у консоль розробника, проте бажано позбавитись такого відстежування перед публікуванням додатку, оскільки ця інформація може бути цікава лише розробнику.

Висновки. Використовуючи всі вищезгадані інструменти та методи, розробник додатків для мобільних платформ має можливість знайти джерело проблем енергоспоживання та оптимізувати код додатку для підтримання стандартів операційної системи. У період, коли потужніші (але такі ж компактні) джерела енергії ще не винайдено, кожен розробник несе відповідальність за роботу операційної системи в цілому, якщо в ній встановлено його додаток.

1. *Nadeem Ahmad Khan, Jahangir Ikram.* Power Consumption Analysis for Mobile Devices and Power Consumption Mathematical Models — Lahore University of Management Sciences — 10, 11 cc.
2. *Trevor Pering et al.* CoolSpots: Reducing the Power Consumption of Wireless Mobile Devices with Multiple Radio Interfaces — ACM, 2006.— 222 c.
3. *Kravets, R. and Krishnan P.*, "Application-driven power management for mobile communication," — *Wireless Networks*, том 6, с. 4

Поступила 26.02.2014р.