УДК 681.3:004.383.4

К.т.н., доцент Плахотний М.В., студент Черкас С.С.

**Національний технічний університет України
«Київський політехнічний інститут»**

АРХІТЕКТУРА ТА ОСОБЛИВОСТІ ПРОГРАМУВАННЯ СИСТЕМ УПРАВЛІННЯ НА БАЗІ КОНТРОЛЛЕРА ТИПУ ARM AT91M42800A

Abstract

Mykola V. Plahotnyi, assoc. prof., PhD; Sergiy Cherkas, student

Architecture and features of programming of control systems on the base of ARM controller AT91M42800A

*The **ARM** is a 32-bit reduced instruction set computer (RISC) developed by ARM Holdings. It was known as the **Advanced RISC Machine**, and before that as the **Acorn RISC Machine**. The ARM architecture is the most widely used 32-bit ISA in terms of numbers produced. The relative simplicity of ARM processors made them suitable for low power applications.*

Вступ

Архітектура ARM (спочатку Advanced RISC Machine — покращена RISC машина, попередник Acorn RISC Machine) — це 32-бітна RISC архітектура процесорів, яку розробила компанія ARM Limited. Широко застосовується у розробці портативних пристроїв. Головною причиною цього є використання енергозберігаючих технологій. Саме тому ця архітектура домінує у пристроях, головною метою яких є енергозбереження. На сьогоднішній день сімейство ARM займає приблизно 75% всіх портативних 32-бітних RISC процесорів, що робить її найбільш використовуваною серед всіх 32-бітних архітектур. Процесори ARM знайшли своє застосування у багатьох пристроях (КПК, цифрові аудіоплеєри, мобільні телефони, калькулятори, ігрові консолі, тощо), комп'ютерній периферії (маршрутизатори). Найпопулярніші пристрої, що працюють на цій архітектурі - плеєри iPod та смартфони Nokia, iPhone [1].

Постановка задачі

Задача полягає в виявленні особливостей програмування систем управління на базі контроллера AT91M42800A, використовуючи плату компанії Atmel AT91EB42.

В плату компанії Atmel AT91EB42 (Рис. 1) входить мікроконтроллер AT91M42800A на базі ядра ARM7TDMI та наступні периферійні пристрої:

- Два серійних порта
- Кнопка перезапуску
- Чотири кнопки для програмування
- Вісім LED індикаторів
- 256 Kb SRAM
- 2 Mb флеш пам’яті
- 4 Mb Serial DataFlash
- 64 Kb EEPROM
- АЦП з SPI доступом

Засоби і методи вирішення

AT91M42800A – представник сімейства 16/32 - розрядних мікроконтроллерів Atmel AT91, який виконаний на базі ядра процесора ARM7TDMI. Даний процесор має 32-розрядну RISC-архітектуру з великим набором 16-розрядних інструкцій та дуже малим енергоспоживанням. Крім того, велика кількість регістрів дозволяє значно підвищити швидкодію обробки даних, що робить цей пристрій зручним для вирішення задач у реальному часі. AT91M42800A можна безпосередньо підключити до зовнішньої пам'яті, у тому числі флеш - пам'ять, через інтерфейс зовнішньої шини. Контроллер управління енергоспоживанням дозволяє користувачу керувати активністю мікроконтроллера залежно від поточних вимог, а малопотужний генератор частотою 32,768 кГц дозволяє максимально знизити енергоспоживання [2].

Цікавим додатком до дизайну ARM ядра є використання 4-бітного коду обставини на початку кожної інструкції, а це означає, що виконання кожної інструкції умовно не обов'язкове. Інші процесорні архітектури, як правило, мають код обставини як розгалуження інструкції. Це значно скорочує кодування біту, придатного для переміщення інструкції в доступну пам'ять; але з іншого боку, це дозволяє уникнути розгалуження інструкцій при генерації коду для малих операторів if. Стандартним прикладом є Алгоритм Евкліда: На мові програмування С виглядає так:

```
while (i != j)
{
  if (i > j)
     i -= j;
  else
     j -= i;
}
```

Для ARM процесорів асемблерний код виглядає так:

```
loop    CMP    Ri, Rj      ; set condition "NE" if (i != j)
                           ;              "GT" if (i > j),
                           ;           or "LT" if (i < j)
        SUBGT  Ri, Ri, Rj   ; if "GT", i = i-j;
        SUBLT  Rj, Rj, Ri   ; if "LT", j = j-i;
        BNE    loop         ; if "NE", then loop
```

що дозволяє уникнути розгалуження коду навколо операторів then та else. Ще однією унікальною особливістю набору є можливість згортання зсувів при "обробці даних" (арифметичних, логічних та регістрових переміщень) інструкції, для прикладу як у С-операторі *a += (j <<);* окремий цикл інструкції для ARM може бути оформлено як одне слово *ADD Ra, Ra, Rj, LSL #2.* Це призводить до того, що типові ARM програми щільніші, ніж очікувалося, і з меншою кількістю звертань до пам'яті, тому конвеєр використовується більш ефективно. Навіть незважаючи на те, що ARM працює із нижчою швидкістю, але він в цілому успішно конкурує з більш складнішими процесорними архітектурами [1].

Характерною задачею систем управління є маніпулювання бітами на портах введення виведення. Процедура зміни біта звичайно має на увазі читання всього слова, накладення бітової маски і запис нового значення слова назад в порт. В RISC процесорах така операція проводиться однією командою, що виконує нерозривний цикл «зчитування-модифікація-запис». Оскільки в архітектурі ARM немає команд, що забезпечують такий цикл при звертанні до пам'яті, маніпуляція з бітами може стати проблемою. За наявності багатозадачності між моментами первинного значення порту і записом його модифікованого значення може відбутися переривання, в результаті якого зміниться поточний стан порту. Для виключення такої ситуації необхідно забороняти переривання перед кожною бітовою процедурою [3]. У цьому випадку буде потрібно виконати наступну послідовність інструкцій:

```
        MRS    r0,CPSR                  ; Заборона переривань на рівні ядра
        ORR    r0,r0,#(I_BIT OR _F_BIT)
        MSR    CPSR,r0
        LDR    r0,=PORT_ADDR    ; Завантаження покажчика адресою порту PORT_ADDR
        LDR    r1,[r0]                  ; Читання значення порту
        ORR    r1,r1,#80                ; Накладення маски
        STR    r1,[r0]                  ; Запис результату в порт
        MRS    r0,CPSR                  ; Дозвіл переривань
        BIC    r0,r0,#(I_BIT OR _F_BIT)
        MSR    CPSR,r0
```

Очевидно, що така процедура надзвичайно громіздка і повільна. Тому для підвищення ефективності роботи з портами в мікроконтролерах з ядром ARM застосовується незвичайна для 8-розрядних мікроконтролерів структура порту. Кожний порт представлений у вигляді двох віртуальних регістрів, один з яких служить тільки для установки бітів, а інший - тільки для їх скидання. В результаті відпадає необхідність в попередньому читанні порту, забороні переривань і вся процедура модифікації біта виконується однією інструкцією [3]. Послідовність інструкцій для цього випадку виглядає таким чином:

LDR r0,=PORT_ADDR ; Завантаження покажчика адресою порту PORT_ADDR

MOV r1,#80 ; Завантаження маски біта в регістр

STR r1,[r0,#PORT_SET_offset] ; Установка біта в регістрі «установки», де PORT_SET_offset - зсув адреси регістра щодо PORT_ADDR

Висновки

Перспективність застосування мікроконтроллерів з ядром ARM обумовлена, насамперед, використання енергозберігаючих технологій. Низьке енергоспоживання та малі габарити дають можливість використання їх в системах управління, головною ціллю яких є мобільність. Потенційні проблеми ефективності виконання програм, пов'язані з особливостями RISC архітектури, успішно розв'язуються за рахунок використовування оригінальних апаратних рішень на рівні периферійних модулів.

Процесори ARM знайшли своє застосування у багатьох пристроях та комп'ютерній периферії. Найпопулярніші пристрої, що працюють на цій архітектурі — плеєри iPod та смартфони Nokia, iPhone.

Література

1. Архітектура ARM // http://uk.wikipedia.org/wiki/Архітектура_ARM
2. AT91M42800A Мікроконтроллер сімейства AT91 ARM Thumb // http://www.gaw.ru/html.cgi/txt/ic/Atmel/micros/arm/AT91M42800A.htm
3. Микроконтроллеры ARM (ARM7 и ARM9) обзор // http://www.phyton.ru/pages/page41.html