ПРАКТИЧНА РОБОТА

ДОСЛІДЖЕННЯ ПРОГРАМНОЇ МОДЕЛІ І СИСТЕМИ КОМАНД МІКРОКОНТРОЛЕРА

**Мета роботи:** оволодіння методикою відпрацювання програм симулятивним методом у середовищі AVR Studio та ознайомлення з основними прийомами написання програм із використанням системи команд мікроконтролерів AVR.

**Програма роботи:**

Розробити пристрій управління одним світлодіодним індикатором за допомогою однієї кнопки. При натисканні кнопки світлодіод повинний засвітитись, при відпусканні – згаснути.

Повна схема пристрою, що дозволяє вирішити поставлену задачу, наведена на рисунку 1.

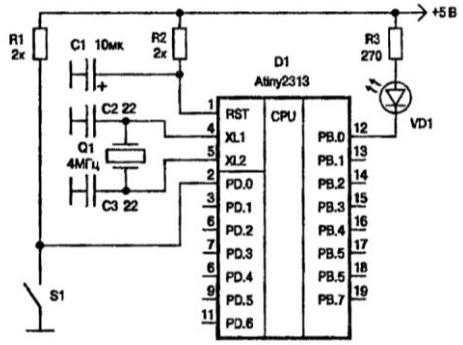


Рисунок 1 – Принципова схема з одним світлодіодом і однією кнопкою

Вдосконалена схема зображена на рисунку 2.

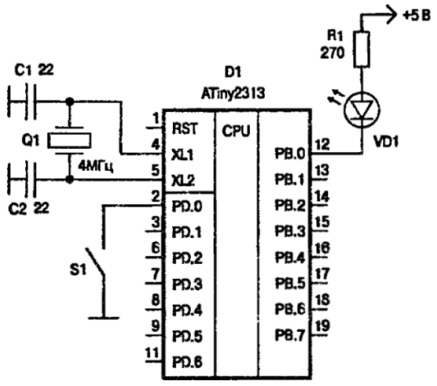


Рисунок 2 – Вдосконалена схема для першого завдання

Текст можливого варіанту програми мовою Асемблер, що реалізує поставлену вище задачу, наведений у лістингу 1:

```
;------------------------ Псевдокоманды управления

.include "tn2313def.inc"                ; Присоединение файла описаний
.list                                   ; Включение листинга

;------------------------ Начало программного кода

.cseg                                   ; Выбор сегмента программного кода
.org    0                               ; Устанеовка текущего адреса на ноль

.def    temp = r16                      ; Определение главного рабочего регистра


;------------------------ Инициализация стека

        ldi             temp, 0x7F      ; Выбор адреса вершины стека
        out             SPL, temp       ; Запись его в регистр стека
        ldi             temp,0



;------------------------ Инициализация портов ВВ

        out             DDRD, temp      ; Записываем ноль в DDRD (порт PD на ввод)

        ldi             temp, 0xFF      ; Записываем число $FF в регистр temp
        out             DDRB, temp      ; Записываем это число в DDRB (порт PB на вывод)
        out             PORTB, temp     ; Записываем то же число в PORTB (потушить светодиод)
        out             PORTD, temp     ; Записываем его же в PORTD (включаем внутр. резисторы)



;------------------------ Основной цикл
main:
        in              temp, PIND      ; Читаем содержимое порта PD
        out             PORTB, temp     ; Пересылаем в порт PB
        rjmp    main                    ; К началу цикла
```

Виконати відпрацювання і трансляцію програми у середовищі AVR Studio, створити проект. Виконати програму. Прослідкувати за зміною даних в регістрах та ОЗП мікроконтролера. Пояснити, які складові програмної моделі змінили свій стан.

**Теоретичні відомості**

***Загальні відомості***

*Мова програмування* – це спеціально розроблена мова, що служить посередником між машиною і людиною.

*Головна задача мови* – точно описати послідовність дій, яку повинний виконати мікроконтролер.

У процесі створення програми програміст просто пише її текст на комп'ютері точно так, як він пише будь-який інший текст. Потім програміст запускає спеціальну програму – транслятор.

*Транслятор* – це спеціальна програма, яка переводить текст, написаний програмістом, у машинні коди, тобто у форму, зрозумілу для мікроконтролера.

Написаний програмістом текст програми називається *початковим* або *об'єктним кодом*. Код, отриманий у результаті трансляції, називається *результуючим* або *машинним кодом.* Саме цей код записується у пам'ять програм мікроконтролера. Для запису машинного коду в програмну пам'ять застосовуються спеціальні пристрої – *програматори.*

Усі мови програмування поділяються на дві групи: *мови низького рівня* і *мови високого рівня.*

Типовим прикладом мови програмування низького рівня є *мова Асемблер*. Ця мова максимально наближена до системи команд мікроконтролера.

У процесі трансляції кожна команда просто *заміняється кодом операції*. Складаючи програму мовою Асемблер, програміст повинний оперувати тими ж видами даних, що і сам процесор, тобто байтами і бітами.

*Специфіка* мови Асемблер полягає ще і в тому, що набір операторів для цієї мови напряму залежить від системи команд конкретного мікроконтролера. Ми будемо вивчати одну конкретну версію мови Асемблер. А саме *Асемблер для мікроконтролерів серії AVR.*

У недалекому минулому мова Асемблер була єдиною мовою програмування для мікроконтролерів. Але зараз, коли можливості сучасних мікроконтролерів значно виросли, для складання програм усе частіше використовують мови високого рівня, такі як Бейсик, СІ і т.п.

Мови високого рівня відрізняються тим, що вони набагато більше орієнтовані на людину. Такі мови оперують вже не з байтами, а з десятковими

числами, а також зі змінними, константами та іншими елементами, знайомими нам з математики.

Транслятор з мови високого рівня проводить більш складні перетворення, ніж транслятор з Асемблера. Але в результаті також маємо *програму у машинних кодах*. При цьому транслятор використовує усі ресурси мікроконтролера на свій погляд.

Програма-транслятор обирає сама, у яких саме регістрах або комірках пам'яті вона буде зберігати значення описаних програмістом змінних, за якими алгоритмами вона буде обчислювати математичні функції. Тому задача ефективності алгоритму отриманої у результаті трансляції програми цілком лягає на програму-транслятор.

*Алгоритм* – це послідовність дій, яку повинний виконати мікроконтролер, щоб досягнути потрібного результату.

Для простих задач алгоритм можна просто описати словами. Для більш складних задач алгоритм креслиться у графічному вигляді.

У цілому, програми, написані на мовах високого рівня, займають у пам'яті мікроконтролера об'єм на 30-40% більший, ніж аналогічні програми, написані на мові Асемблер.

Але якщо мікроконтролер має достатньо пам'яті і запас по швидкодії, то це збільшення програми не є проблемою. *Первагою* мов високого рівня є істотне прискорення процесу розробки програми. З усіх мов високого рівня *самою ефективною* є мова СІ.

Вивчення прийомів програмування ми будемо здійснювати на ряді конкретних прикладів:

- кожний приклад буде починатись з постановки задачі;
- потім ми навчимось обирати схемне рішення;
- тільки після цього будуть розібрані приклади програм.

### ***Послідовність дій для створення проекту, трансляції і відпрацювання програми***

*Створення проекту*

Для розробки проекту можна використовувати наступну послідовність операцій.

Виберіть у верхній строчці меню розділ Project і, відповідно, опцію New Project (створення нового проекту). На екрані монітору з'явиться вікно Create new project (рис.3), що використовується для встановлення атрибутів проекту. У вікні Project type позначте мову програмування - Atmel AVR Assembler. У вікні Project name визначте ім'я Вашого проекту (наприклад, Lab1). Введіть позначки у вікна створення початкового файлу (Create initial file) та створення папки проекту (Create folder). Визначте розміщення (Location) папки проекту, натиснувши кнопку «…»..

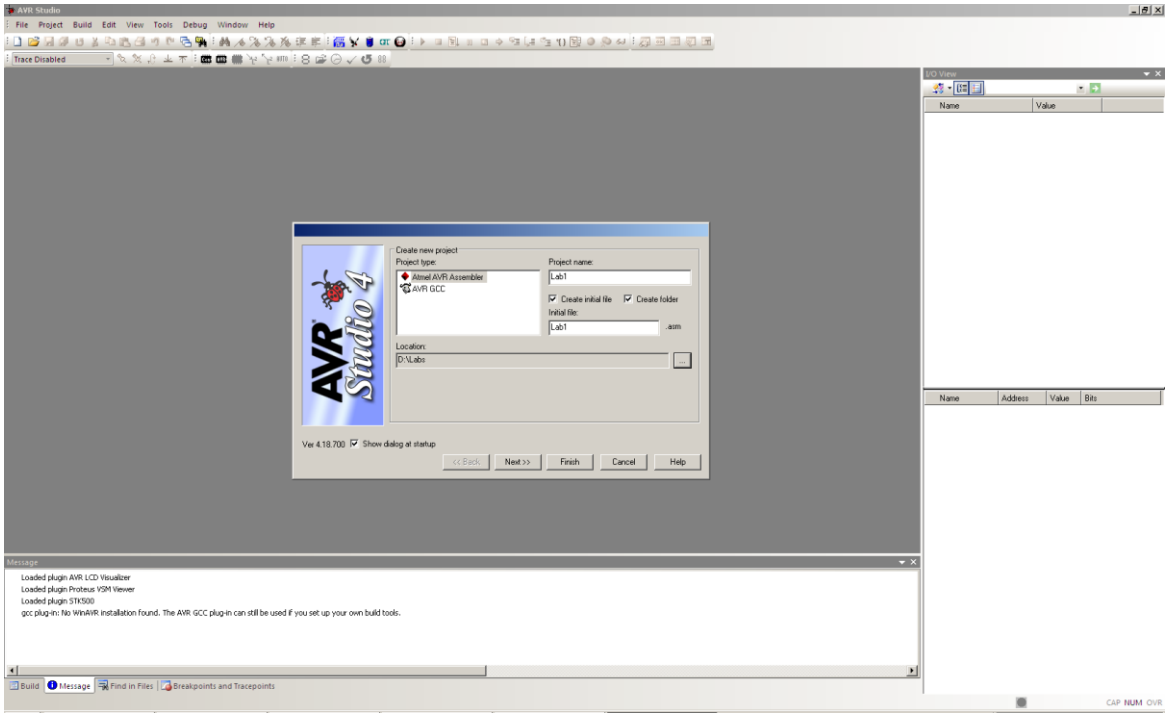


Рисунок 3

Для переходу до наступного етапу побудови проекту натисніть кнопку «Next» Після цього з'являється наступне вікно (рис.4), в якому розташовані два поля під загальною назвою «Select debug platform and device» (вибір платформи відпрацювання і пристрою).

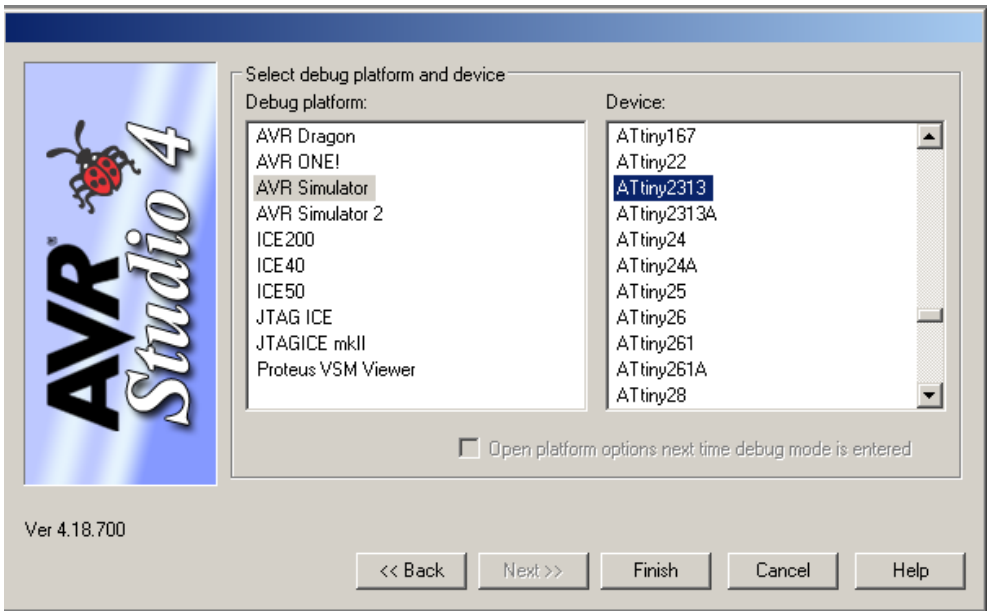


Рисунок 4

Оберіть в якості платформи «AVR Simulator» (Програмний імітатор AVR). В полі «Device» обираємо мікроконтролер ATtiny2313. Для завершення процесу встановлення атрибутів проекту натискаємо кнопку «Finish». Після натиснення цієї кнопки програма створює проект і записує його в обрану Вами директорію: D:\Labs\Lab1\. Зараз він складається з двох файлів: Lab1.aps (файл проекту) і Lab1.asm (файл, де ми розмістимо текст програми на Асемблері).

Тепер у центральному вікні AVR Studio набираємо текст програми і зберігаємо його на жорсткому диску (див. рисунок 5).

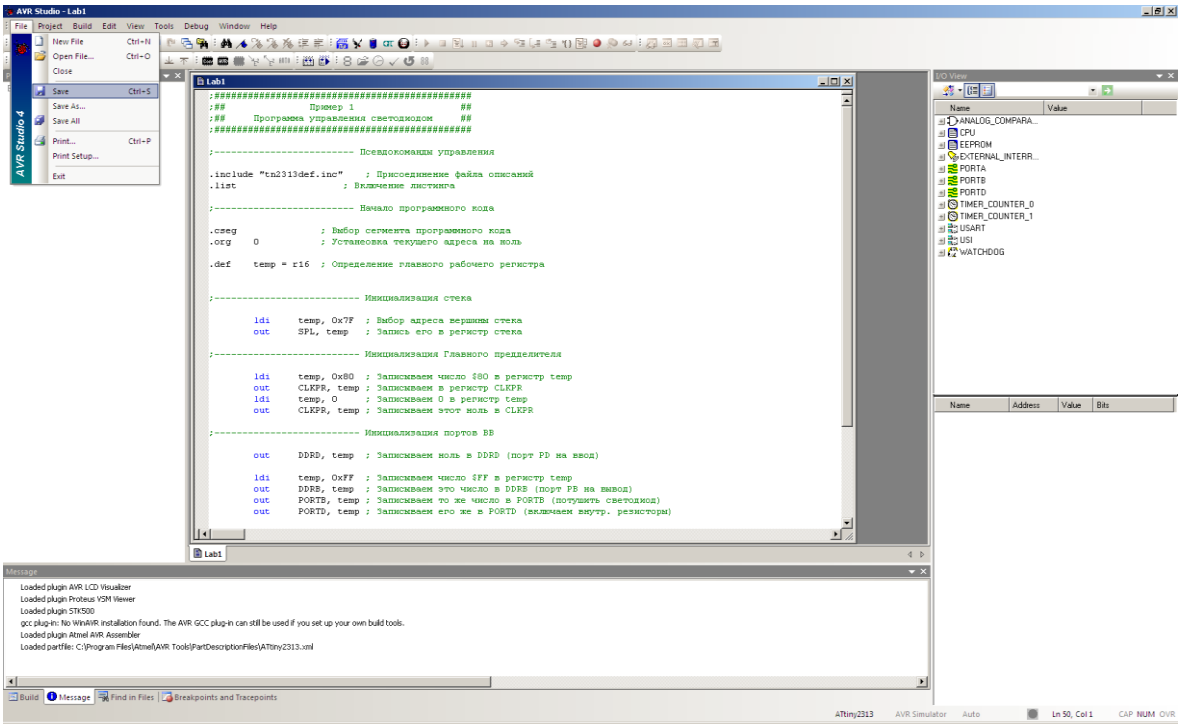

Рисунок 5

*Трансляція програми*

Для запуску процесу трансляції поточного проекту у меню «Build» обираємо пункт «Build»

У нижній частині монітору, у випадку відсутності помилок, з`являється повідомлення про використання різних сегментів пам`яті мікроконтролера (див. рисунок 6).

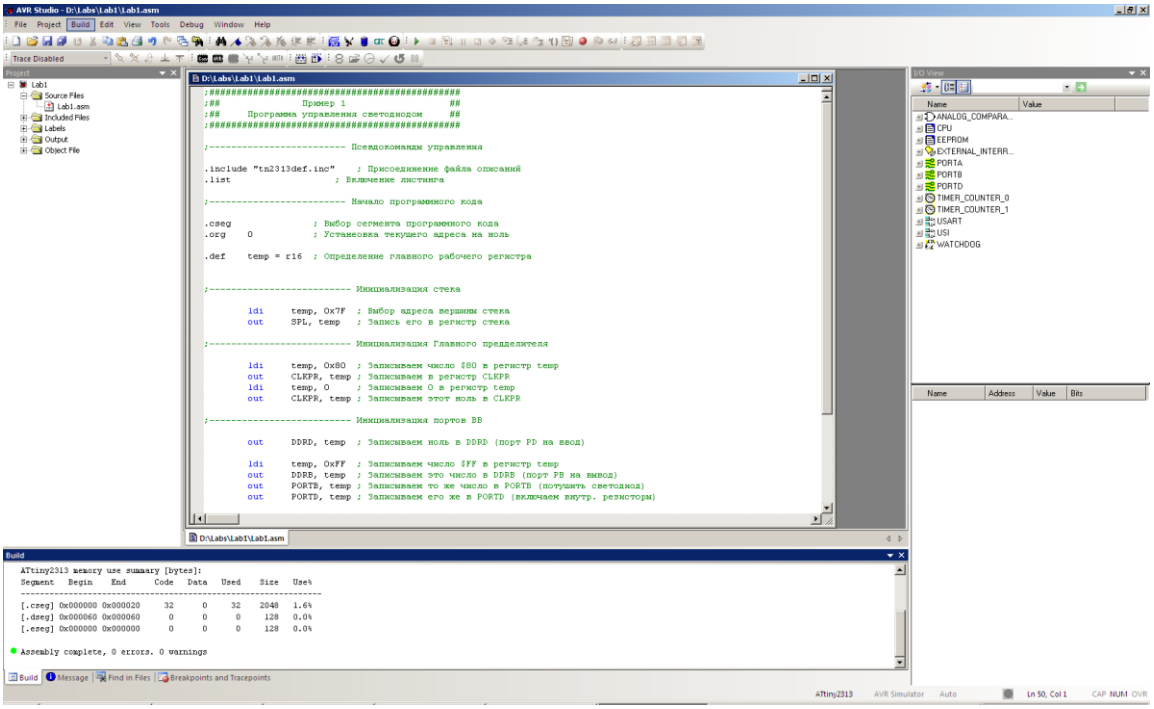


Рисунок 6

Після закінчення роботи проект необхідно записати. Для цього у розділі меню Project викликають опцію Save Project.

*Відпрацювання програми*

Для переходу у режим відпрацювання програми у розділі меню «Debug» обираємо пункт «Start Debugging». Після закінчення процесу підготовки програма перейде у новий режим роботи (див. рисунок 7).

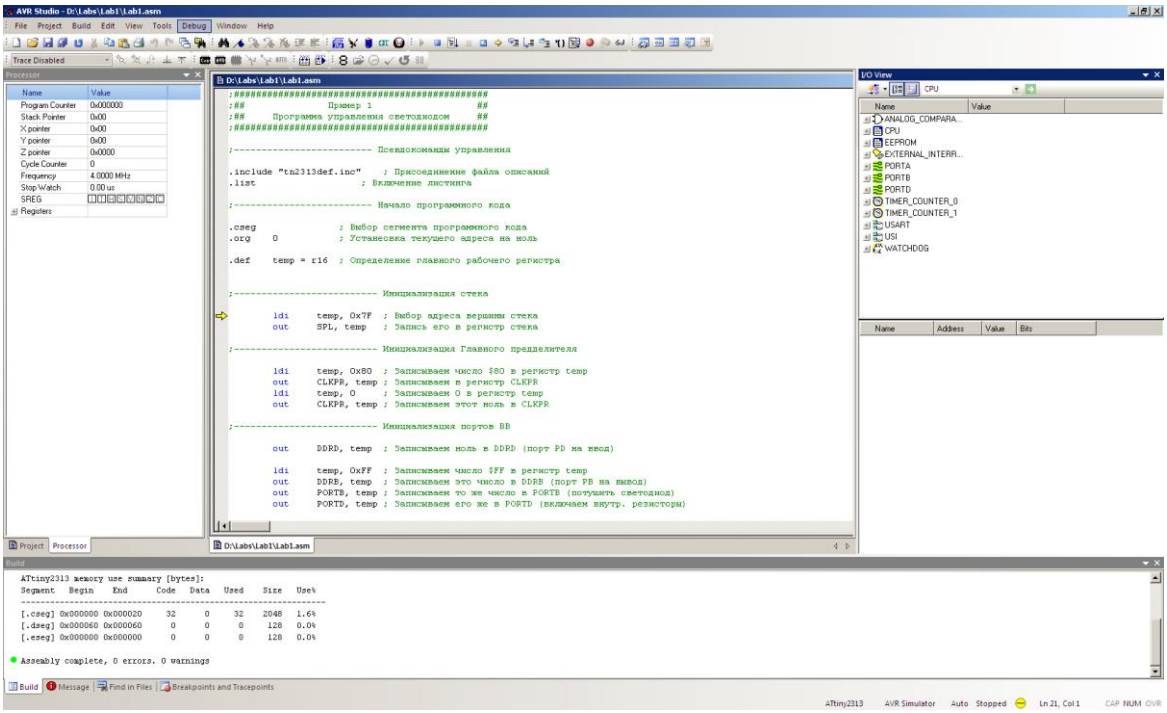


Рисунок 7

Виконаємо відпрацювання програми покроковим методом. Для того, щоб виконати один крок, потрібно у меню «Debug» вибрати пункт «Step into». Після закінчення відпрацювання програми у меню «Debug» вибираємо пункт «Stop Debugging».

***Система команд мікропроцесорів сімейства AVR містить п'ять груп команд:*** умовного розгалуження, безумовного розгалуження, арифметичні і логічні операції, команди пересилки даних, команди роботи з бітами. У останніх версіях AVR сімейства Mega реалізована функція апаратного множення.

***Програмна модель AVR, що зображена на рис. 8, являє собою сукупність програмно доступних ресурсів AVR.***

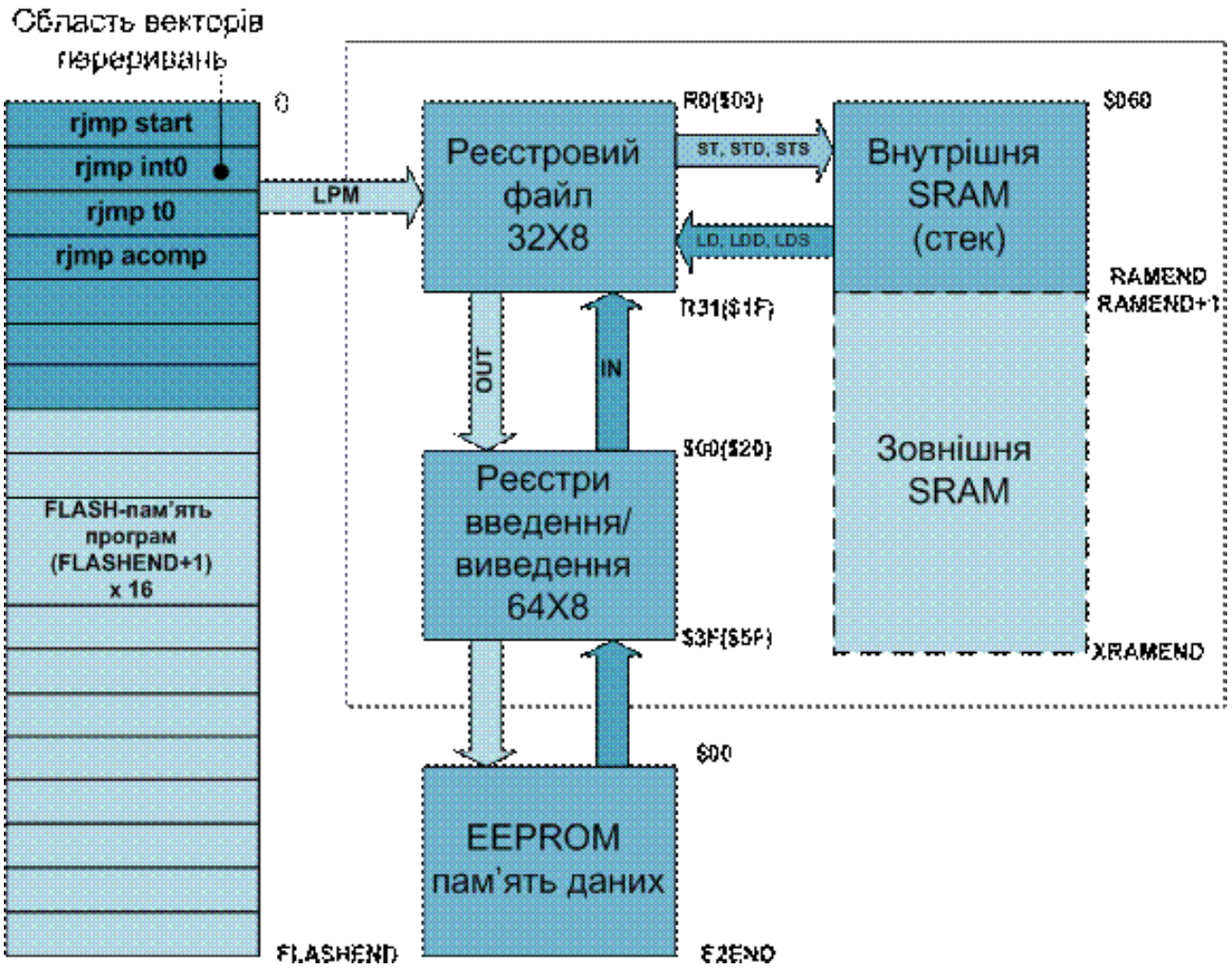


Рисунок 8 - Програмна модель AVR-мікроконтролерів

**Зміст звіту:**

1. Тема і мета роботи.
2. Блок-схема алгоритму роботи програми мікроконтролера.
3. Текст програми з коментарями, згідно з програмою роботи.
4. Дані регістрового файлу, регістрів введення/виведення, комірок ОЗП.

**Контрольні запитання:**

1. Що називається мовою  програмування? Головна задача мови.
2. Що називається транслятором?
3. Що називається початковим або об'єктним кодом?
4. Що називається результуючим або машинним кодом?
5. Для чого призначений програматор?

6. Поясніть різницю між мовами низького і високого рівня.
7. Поясніть, у чому полягає специфіка мови Асемблер.
8. Поясніть, що називається алгоритмом?
9. Яка мова високого рівня є найбільш ефективною для програмування мікроконтролерів?
10. Поясніть, що називається директивою. Які директиви Асемблера ви знаєте?
11. Яка послідовність дій по створенню проекту та відпрацюванню помилок у середовищі AVR Studio.
12. Дайте визначення програмної симуляції прикладних програм.
13. Поясніть використання AVR Studio для програмної симуляції створеного проекту.
14. Як відбувається запуск програми в середовищі AVR Studio?
15. Як виправити помилки в тексті програми?
16. Визначити класифікацію системи команд мікропроцесорів AVR.
17. Наведіть програмну модель AVR-мікроконтролерів.