КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (16), 2022
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2022.16.142158
УДК 621.317.7: 621.382.2:004.021

Бушма Олександр Володимирович
доктор технічних наук, професор, професор кафедри комп'ютерних наук і математики,
Київський університет ім. Бориса Грінченка, м. Київ, Україна
ORCID ID: 0000-0003-1604-6129
o.bushma@kubg.edu.ua

Турукало Андрій Валерійович
аспірант кафедри комп'ютерних наук,
Національний університет біоресурсів і природокористування, м. Київ, Україна
ORCID ID: 0000-0003-2944-1806
tyrykalo@gmail.com

ОЦІНКА ПАРАМЕТРІВ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ШКАЛЬНОГО ВІДОБРАЖЕННЯ ДАНИХ

Анотація. Робота присвячена оцінці споживання ресурсів мікроконтролера для синтезу шкального дискретно-аналогового відображення даних на світлодіодному інформаційному полі в двотактному режимі. У роботі порівняно програми багататактного та двотактного шкального виводу інформації для системи відображення інформації. Показано значущість впливу двотактних інформаційних моделей на мінімізацію витрат машинного часу однокристального мікроконтролера. Визначено, що для зниження споживання ресурсів в розроблюваних рішеннях необхідно зосередити увагу на максимально можливій оптимізації блоків програм, які виконуються під час переривань, та обслуговують підсистему введення-виведення мікроконтролера. При цьому найкращі результати має шкальна індикація на основі двотактної адитивної інформаційної моделі. З'ясовано, що традиційний підхід до оцінки ефективності програм з використанням спеціальних benchmark програм, з подальшим виміром обсягу коду та часу виконання всієї програми не дозволяє коректно оцінити ефективність програми та роботу мікроконтролера на етапі проектування пристроїв. В якості альтернативи запропоновано використовувати для оцінки ефективності програми розмір байт-коду програми та швидкість виконання основного циклу – процедури виводу інформації. З'ясовано, що за швидкістю виконання та споживання ресурсів багататактний варіант суттєво програє двотактній реалізації. Визначено, що зменшення кількості тактів формування зображення на інформаційному полі є одним із найефективніших шляхів мінімізації споживання ресурсів мікроконтролера для обслуговування підсистеми індикації.

Ключові слова: споживання ресурсів мікроконтролера; шкала; шкальний індикатор; двотактна інформаційна модель; мікроконтролер; дискретно-аналогова індикація; програмна реалізація; світлодіод; динамічний режим.

ВСТУП

Дискретні шкальні індикатори знайшли широке застосування завдяки унікальному комплексу їх технічних та ергономічних характеристик. Переважна більшість сучасних рішень базується на програмному синтезі зображень на світлодіодному (СД) інформаційному полі засобів відображення даних [1], [2].

Дослідження програмного підходу до збудження шкальних індикаторів (ШІ) [3], [4], [5] показало, що для зниження споживання ресурсів в розроблюваних рішеннях необхідно зосередити увагу на максимально можливій оптимізації блоків програм, які виконуються під час переривань, та обслуговують підсистему введення-виведення мікроконтролера (МК). Механізм переривань найкраще підходить для обробки подій, які відбуваються асинхронно при виконанні програми. Одночасно одним з основних завдань оптимізації програми є зниження відсотку активного процесорного часу на потреби підсистеми відображення інформації. Одним із рішень є використання інформаційної

моделі (ІМ), яка застосовується на шкальному індикаторі з мінімізованою кількістю електричних з'єднань завдяки його побудові на основі двовимірної матриці електрооптичних елементів [6], [7]. В той же час оцінка ресурсів, які потрібні для реалізації підсистеми шкального виводу інформації, не проводилася, що ускладнює оптимізацію інтерактивних пристроїв та вбудованих систем.

Метою роботи є оцінка споживання ресурсів МК для синтезу шкального дискретно-аналогового відображення даних на СД інформаційному полі в двотактному режимі.

ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

Узагальнений підхід до побудови двотактної програмної підтримки роботи ШІ передбачає загальну ініціалізацію цієї підсистеми одноразово при старті пристрою на МК [8]. Така функція обов'язково включає загальне налаштування обробника переривання індикації та відповідних змінних. Для підвищення ефективності програми та зниження відсотка активного процесорного часу на її виконання одним з основних завдань є вибір ІМ та алгоритму, який дозволить контролювати весь ланцюжок процесів від отримання даних до їх відображення на шкалі з метою мінімізації кількості виконуваних блоків коду [9].

Проведене дослідження програмної реалізації двотактних ІМ показало, що основна концентрація зусиль, спрямованих на зниження ресурсоємності розроблених рішень, повинна концентруватися на оптимізації блоків програм, які виконуються під час переривань, що обслуговують підсистему індикації [10], [11]. При цьому доцільно розглянути обидва можливі варіанти ІМ, які формують зображення на інформаційному полі в двотактному режимі [12].Алгоритм узагальненого обробника переривання ШІ, інваріантний відносно виду двотактної ІМ, поданий на рис.1.

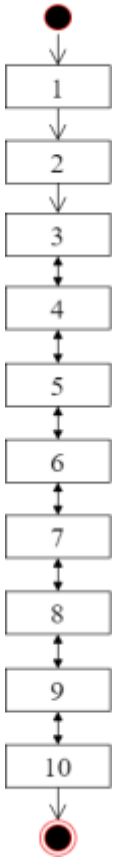

Рис. 1 Узагальнений алгоритм програми шкальної індикації

ISSN 2663 - 4023

Перший блок забезпечує ініціалізацію поточних змінних конкретного моменту часу. Блок 2 – це селектор тактів, який відповідає за зв'язок поточного переривання з відповідними функціями тактів. Блок 3 приймає та зберігає в ОЗП поточні значення даних, які підлягають візуалізації. Блок 4 формує коди керування (КК) для старших розрядів матриці СД у першому такті для першої та для другої ІМ, відповідно:

$$A_{1,1} = \left\{ \bigcup_{x=1}^{q} a_{xy} \left| \begin{matrix} t=t_s+\tau_g-0 \\ t=t_s+0 \end{matrix} \right. \right\} \quad ;$$

$$B_{1,1} = \left\{ \bigcup_{y=1}^{v-mq} a_{xy} \left| \begin{matrix} t=t_s+\tau_g-0 \\ t=t_s+0 \end{matrix} \right. \right\} \quad .$$

В блоці 4 також генерується КК старших розрядів матриці СД у другому такті для першої та другої ІМ, відповідно:

$$A_{2,1} = \left\{ \bigcup_{x=q+1}^{m} a_{xy} \left| \begin{matrix} t=t_s+2\tau_g-0 \\ t=t_s+\tau_g+0 \end{matrix} \right. \right\} \quad ;$$

$$B_{2,1} = \left\{ \bigcup_{y=v-mq+1}^{m} a_{xy} \left| \begin{matrix} t=t_s+2\tau_g-0 \\ t=t_s+\tau_g+0 \end{matrix} \right. \right\} \quad .$$

Блок 5 формує КК для молодших розрядів матриці СД у першому такті для першої та для другої ІМ, відповідно:

$$A_{1,2} = \left\{ \bigcup_{y=1}^{m} a_{xy} \left| \begin{matrix} t=t_s+\tau_g-0 \\ t=t_s+0 \end{matrix} \right. \right\} \quad ;$$

$$B_{1,2} = \left\{ \bigcup_{x=1}^{q+1} a_{xy} \left| \begin{matrix} t=t_s+\tau_g-0 \\ t=t_s+0 \end{matrix} \right. \right\} \quad .$$

В блоці 5 також генерується КК молодших розрядів матриці СД другого такту першої та для другої ІМ, відповідно:

$$A_{2,2} = \left\{ \bigcup_{y=1}^{v-mq} a_{xy} \left| \begin{matrix} t=t_s+2\tau_g-0 \\ t=t_s+\tau_g+0 \end{matrix} \right. \right\} \quad ;$$

$$B_{2,2} = \left\{ \bigcup_{x=1}^{q} a_{xy} \left| \begin{matrix} t=t_s+2\tau_g-0 \\ t=t_s+\tau_g+0 \end{matrix} \right. \right\} \quad .$$

ISSN 2663 - 4023

В матричному вигляді це можна представити як дві частини символу $S_{\nu \mathrm{BG}}$ в якому необхідно утворити пов'язаний динамічний набір $\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{D}}$, або такий самий у матричній формі $\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{M}}$. Ці дві еквівалентні множини розділені на дві підмножини $\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM11}}$, $\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM21}}$, що не перетинаються, та збуджуються в різні періоди часу

$$\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{M}} = \tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{D}} = \left\{ \tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM11}}, \tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM21}} \right\}$$

Така дія подається як

$$\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{M}} = \tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM}} =$$

$$= \begin{Vmatrix} \tilde{a}_{11} & \tilde{a}_{12} & \dots & \tilde{a}_{1(y_\nu-1)} & \tilde{a}_{1y_\nu} & \tilde{a}_{1(y_\nu+1)} & \dots & \tilde{a}_{1(m-1)} & \tilde{a}_{1m} \\ \tilde{a}_{21} & \tilde{a}_{22} & \dots & \tilde{a}_{2(y_\nu-1)} & \tilde{a}_{2y_\nu} & \tilde{a}_{2(y_\nu+1)} & \dots & \tilde{a}_{2(m-1)} & \tilde{a}_{2m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{a}_{(x_\nu-1)1} & \tilde{a}_{(x_\nu-1)2} & \dots & \tilde{a}_{(x_\nu-1)(y_\nu-1)} & \tilde{a}_{(x_\nu-1)y_\nu} & \tilde{a}_{(x_\nu-1)(y_\nu+1)} & \dots & \tilde{a}_{(x_\nu-1)(m-1)} & \tilde{a}_{(x_\nu-1)m} \\ \tilde{a}_{x_\nu 1} & \tilde{a}_{x_\nu 2} & \dots & \tilde{a}_{x_\nu(y_\nu-1)} & \tilde{a}_{x_\nu y_\nu} & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \end{Vmatrix};$$

і далі

$$\tilde{\mathbf{A}}_{\nu \mathrm{BG}}^{\mathrm{DM11}} =$$

$$= \begin{Vmatrix} \tilde{a}_{11} & \tilde{a}_{12} & \dots & \tilde{a}_{1(y_\nu-1)} & \tilde{a}_{1y_\nu} & \tilde{a}_{1(y_\nu+1)} & \dots & \tilde{a}_{1(m-1)} & \tilde{a}_{1m} \\ \tilde{a}_{21} & \tilde{a}_{22} & \dots & \tilde{a}_{2(y_\nu-1)} & \tilde{a}_{2y_\nu} & \tilde{a}_{2(y_\nu+1)} & \dots & \tilde{a}_{2(m-1)} & \tilde{a}_{2m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{a}_{(x_\nu-1)1} & \tilde{a}_{(x_\nu-1)2} & \dots & \tilde{a}_{(x_\nu-1)(y_\nu-1)} & \tilde{a}_{(x_\nu-1)y_\nu} & \tilde{a}_{(x_\nu-1)(y_\nu+1)} & \dots & \tilde{a}_{(x_\nu-1)(m-1)} & \tilde{a}_{(x_\nu-1)m} \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \end{Vmatrix},$$

$$\tilde{\mathbf{A}}_{\nu\mathrm{BG}}^{\mathrm{DM21}} = \left\| \begin{matrix} 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ \tilde{a}_{x_\nu 1} & \tilde{a}_{x_\nu 2} & \dots & \tilde{a}_{x_\nu (y_\nu - 1)} & \tilde{a}_{x_\nu y_\nu} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \end{matrix} \right\|.$$

Ці матриці описують дві підмножини елементів, які беруть участь у динамічному формуванні повного зображення символу $S_{\nu\mathrm{BG}}$ за два такти [13].

Блок 6 відповідає за блокування індикації пристрою відображення інформації. Це необхідно для коректного формування зображення на матриці СД при зміні тактів та даних, що відображаються на індикаторі. Блок 7 передає КК з ОЗП у порти МК для збудження відповідних множин СД у перший і другий такти виводу даних на шкалу.

Сформовані КК фіксуються у портах МК та забезпечують стале збудження елементів матриці СД до виникнення наступного переривання. Використовуючи інерційність людського зору та циклічно повторюючи збудження цих двох груп елементів з частотою більше 50 Гц, ми можемо сформувати цілісний візуальний образ, який відповідає отриманому символу.

Блок 8 відповідальний за розблокування індикації та відображення нового зображення на інформаційному полі у відповідності до наявних КК у портах МК. Блок 9 виконує модифікацію та збереження в ОЗП змінних для подальшого формування нових КК. Блок 10 виконує вихід з процедури обробника переривання індикації.

Багататактна ІМ, або динамічна індикація, є загально відомою і реалізує послідовне циклічне збудження електрооптичних елементів інформаційного поля по стовпцям або рядкам двовимірної матриці [14]**.** В нашому випадку ми будемо порівнювати програми, які створені на мові Assembler для матричного ШІ размірністю 10х10. Для формування зображення при використанні двотактної ІМ використовуються два такти, а для багататактної − десять.

МЕТОДИКА ДОСЛІДЖЕННЯ

На першому етапі формується технічне завдання на розробку програми, яке визначає параметри апаратних засобів та відповідні обмеження й умови, що визначають характеристики середовища, де буде працювати кінцевий продукт. Далі створюється загальна блок-схема алгоритму роботи програми разом із деталізованими блок-схемами окремих процедур. Після цього формується вихідний текст програми, який інтегрує всі розроблені компоненти.

Мови програмування МК за своєю структурою мало відрізняються від класичних мов для персональних комп'ютерів. Основною відмінністю є орієнтація на роботу з вбудованими периферійними пристроями. Окрім цього, програмна модель МК має, як правило, команди для роботи з окремими бітами. Останні дозволяють виконувати роботу з окремими лініями портів введення / виведення або прапорцями регістрів тощо, суттєво підвищуючи гнучкість програмного забезпечення та мінімізуючи його обсяг. Подібні команди відсутні в більшості інших комп’ютерних засобах [15].

У якості мови програмування був обраний асемблер, який дозволяє найбільш повно розкрити всі можливості МК, і отримати максимальну швидкодію та мінімізований код. Асемблер відмінно підходить для програмування МК, які мають обмежені ресурси, наприклад, 8-ми бітні моделі з малим об'ємом пам'яті.

Традиційний підхід до оцінки ефективності програми полягає у використанні спеціальних benchmark програм з подальшим виміром обсягу коду та часу виконання всієї програми [16]. Цей підхід не дозволяє коректно оцінювати ефективність програми та роботу МК на етапі проектування пристроїв. Крім цього, на результати оцінки benchmark програм впливає ефективність реалізації самого компілятора, що знижує об'єктивність оцінки характеристик кінцевого продукту.

В якості альтернативи пропонується використовувати для оцінки ефективності програми розмір байт-коду програми та швидкість виконання основного циклу – процедури виводу інформації [17]. Оцінити швидкодію можна шляхом підрахунку загальної кількості машинних тактів МК, які витрачаються на виконання байт-коду програми.

Довжина команд МК сімейства 8051, який застосований в розробці, – один, два або три байти, причому більшість з них (94%) – одно- або двобайтні [18]. Всі команди виконуються за один або два машинних цикли (відповідно 1 або 2 мкс при типовій тактовій частоті 12 МГц). Виняток – команди множення та ділення, які виконуються за чотири машинних цикли (4 мкс) [19]. МК сімейства 8051 використовують пряму, безпосередню, непряму та неявну адресацію даних. В якості операндів команди ці МК можуть використовувати окремі біти, чотирибітні цифри, байти та двобайтні слова [20].

Всі ці риси звичайні для набору команд будь-якого SISC-процесора, і, в порівнянні з RISC набором команд, забезпечують більшу компактність програмного коду та підвищення швидкодії при виконанні складних операцій [21]. Всього МК сімейства 8051 виконує 13 типів команд. При цьому перший байт команди завжди містить код операції, а другий і третій (якщо вони присутні в команді) – адреси операндів або їх безпосередні значення [22]. Склад операндів включає в себе операнди чотирьох типів: біти, нібли (4 розряди), байти та 16-бітні слова. Час виконання команд складає 1, 2 або 4 машинних цикли [23]. При тактовій частоті 12 мГц тривалість машинного циклу складає 1 мкс, при цьому 64 команди виконуються за 1 мкс, 45 команд – за 2 мкс і 2 команди (множення і ділення) – за 4 мкс [24].

Для оцінки споживання ресурсів МК будемо враховувати тільки ту частину програми, яка безпосередньо відповідає за виведення інформації. Для порівняння швидкодії саме ця частина є важливою. В неї входить підпрограма переривання, підпрограми виводу десятків і одиниць та підпрограма перезапуску таймера МК, які можуть бути враховані як

$$C^M = \sum_{i=1}^{4} c^M{}_i = c^M{}_1 + c^M{}_2 + c^M{}_3 + c^M{}_4 \quad , \qquad (1)$$

де C^M - загальна кількість циклів підпрограми для синтезу інформації на інформаційному полі,

$c^M{}_1$ - кількість циклів підпрограми переривання,

$c^M{}_2$ - кількість циклів підпрограми виводу десятків,

$c^M{}_3$ - кількість циклів підпрограми виводу одиниць,

$c^M{}_4$ - кількість циклів підпрограми перезапуску таймера.

ISSN 2663 - 4023

РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Оцінка споживання ресурсів МК для синтезу шкального дискретно-аналогового відображення даних на СД інформаційному полі була виконана для багатотактної та двотактної ІМ [26].

Оцінка швидкодії програми багатотактної ІМ

Підпрограма переривання, яка обслуговує шкальне відображення даних в багатоактному режимі, подається алгоритмом, наведеним на рис. 2.

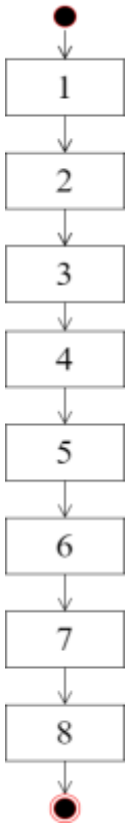

Рис.2 Алгоритм підпрограми переривання для реалізації багатотактної ІМ

Блок 1 виконує початкові маніпуляції для виводу числа на матрицю електрооптичних елементів інформаційного поля. В процесі його виконання відбувається скидання відповідного керуючого біту та ініціалізація двох змінних *Cycle* та *Tact* з нульовими значеннями. Вони вказують на кількість відпрацьованих циклів та тактів, відповідно. Кількість циклів необхідно запам’ятовувати для відслідковування того, як довго знак відображається на шкалі. Змінна *Tact* використовується для визначення такту, який відображає частину символу: десяток, число (остача) або порожній такт. Блок 2 для всіх керуючих ліній портів P0, P2 та P3 формує низький рівень, щоб вимкнути СД матрицю. В блоці 3 перевіряється нульовий цикл виконання програми. Якщо це ненульовий цикл – переходимо до блоку 4. В ньому збільшується лічильник циклів на одиницю та виконується декілька перевірок. В блоці 5 перевіряється кількість циклів: якщо вона більша за 60 – переходимо до підпрограми генерації наступного числа, якщо ні – виведення поточного числа продовжується. В блоці 6 перевіряється кількість

ISSN 2663 - 4023

тактів: якщо вона більша 10 – лічильник тактів скидається в 0, і починається новий цикл виведення числа на інформаційне поле.

В блоці 7 проводяться наступні перевірки:

• якщо кількість тактів менша за кількість десятків – переходимо до підпрограми виводу десятків;

• якщо кількість тактів дорівнює кількості десятків – переходимо до підпрограми виводу одиниць числа;

• в іншому випадку – переходимо до блоку 8 ; це формує «порожні» такти.

Блок 8 збільшує змінну *Tact* на 1, викликає підпрограму запуску таймеру для його повторного старту та завершує обробку переривання, тобто повертається в основний цикл. Аналіз підпрограми переривання показав, що вона складається з 31 команди та виконується за $c^{M}{}_{1} = 48$ циклів.

Підпрограма виводу десятків реалізована у вигляді алгоритму, блок-схема якого наведена на рис. 3

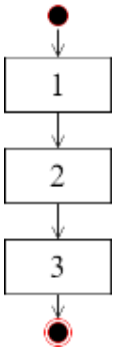

Рис.3 Алгоритм підпрограми виводу десятків

Блок 1 завантажує з пам’яті МК бітову маску для контролюючого порту МК. В блоці 2 перевіряється кількість десятків числа: якщо величина більша 8, тоді маска використовується для порту P2, якщо менша – для P3. Така перевірка необхідна, тому що один порт має 8 ліній і керується за допомогою коду в 1 байт. У випадку, коли кількість десятків більша 8, – потрібно керувати матрицею СД через порт P2 і використовувати додаткову бітову маску для активації ліній цього порту, так як саме до них приєднані елементи керування розрядами матриці, починаючи з дев’ятого. Для керування розрядами з першого по восьмий бітова маска застосовується для ліній порту P3. Блок 3 закінчує виконання підпрограми та повертає керування до основної програми. Підпрограма виводу десятків має 11 команд та виконується за $c^{M}{}_{2} = 18$ циклів.

Підпрограма виводу одиниць реалізована у вигляді алгоритму, блок-схема якого наведена на рис. 4.

ISSN 2663 - 4023

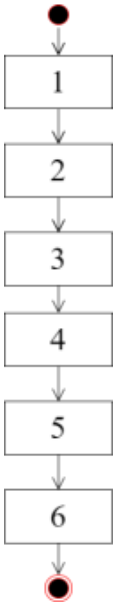

Рис.4 Алгоритм підпрограми виводу одиниць

Блок 1 завантажує бітову маску для виводу одиниць і записує її в регістр R0 МК. Далі виконується перевірка (блок 2): якщо кількість одиниць числа, що виводиться на матрицю СД, менша або дорівнює восьми – тоді активується увесь порт МК P0, що вимкне перших вісім СД. В протилежному випадку – для порту P0 застосовується маска з регістра R0, а регістр переводиться в нульовий стан. Наступний блок 3 завантажує бітову маску для активації першої одиниці.

Блок 4 виконує перевірку чи більша кількість десятків за 8 і завантажує маску до відповідного порту P2, якщо кількість одиниць більша за 8, або до P3 якщо кількість одиниць менша, або дорівнює 8. Блок 5, в залежності від кількості одиниць, застосовує маску до порту, якщо це порт P2 – виконується операція логічного «АБО» з умістом регістра R0 для збереження кількості одиниць для об'єднання з маскою для активації СД. Блок 6 збільшує змінну *Tact* на одиницю та здійснює повернення до основної програми. Підпрограма виводу одиниць складається з 25 команд та виконується за $c^M{}_3 = 28$ циклів.

Підпрограма перезапуску таймера, реалізована у вигляді алгоритму, блок-схема якого подана на рис. 5.

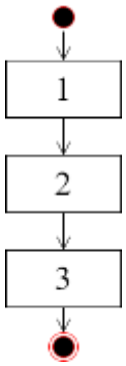

Рис.5 Алгоритм підпрограми перезапуску таймера

Блок 1 налаштовує засоби введення / виведення інформації до / з МК та запускає його внутрішній таймер у режимі 1, а також налаштовує переривання від таймеру і переходить до підпрограми його ініціалізації. Блок 2 налаштовує частоту таймера та запускає його. Регістр таймера 1 в режимі роботи 1 має розрядність 16 біт, що дозволяє встановити максимальний інтервал – 65 535 мс. Встановлюючи початкове значення молодшого та старшого байтів регістру таймера, задаємо частоту його оновлення в 16,6 мс, так як цієї частоти достатньо для формування цілісного зображення на ІП використовуючи інерційність людського зору.

Основний цикл програми (блок 3) очікує переповнення регістру таймера 1, що ініціалізує виклик переривання. Підпрограма перезапуску таймера має 4 команди та виконується за $c^M{}_4 = 7$ циклів.

Загальна кількість циклів МК, які потрібно виконати процесору для обслуговування одного з десяти тактів програми синтезу багатотакної шкальної індикації, у відповідності до (1)

$$C^M = \sum_{i=1}^{4} c^M{}_i = 48 + 18 + 28 + 7 = 101 .$$

Таким чином, витрати часу на формування повного зображення на матриці елементів шкали 10х10 при типовій тактовій частоті МК 12 мГц будуть складати 1010 мкс.

Оцінка швидкодії програми двотактної ІМ

Аналогічним способом підрахуємо кількість машинних циклів для двотактної програми. Алгоритми підпрограм можуть бути представлені як

$$C^B = \sum_{i=1}^{4} c^B{}_i = c^B{}_1 + c^B{}_2 + c^B{}_3 + c^B{}_4 , \qquad (2)$$

де C^B - загальна кількість циклів підпрограми для синтезу інформації на інформаційному полі,

$c^B{}_1$ - кількість циклів підпрограми переривання,

$c^B{}_2$ - кількість циклів підпрограми першого такту,

$c^B{}_3$ - кількість циклів підпрограми другого такту,

$c^B{}_4$ - кількість циклів підпрограми перезапуску таймера.

Підпрограму переривання для формування зображення на індикаторі на основі двотактної ІМ реалізує алгоритм, наведений на рис. 6.

ISSN 2663 - 4023

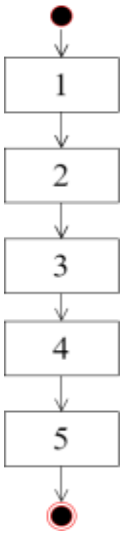

Рис.6 Алгоритм підпрограми переривання для реалізації двотактної ІМ

Блок 1 виконує підготовку для виводу числа на матрицю електрооптичних елементів і починається зі скидання таймеру та відповідного керуючого біту, а також ініціалізує змінні *Tact1* та *Tact2* з нульовими значеннями. Останні використовуються для розділення зображення на дві частини, що синтезуються відповідно до правил двотактної ІМ. Блок 2 для всіх ліній портів P0, P2 та P3 формує низький рівень, щоб вимкнути СД шкали. Блок 3 отримує і зберігає в пам'яті МК поточне значення даних, які потрібно візуалізувати. Блок 4 інкрементує лічильник циклів на 1 і виконує перевірку – якщо кількість циклів більша за 60, тоді переходимо до підпрограми генерації наступного числа. Блок 5 зберігає значення змінних *Tact1* та *Tact2*, викликає підпрограму запуску таймера для його повторного старту і завершує обробку переривання. Відбувається повернення в основну програму. Підпрограма переривання складається з 23 команд і виконується за $c^{B}_{1} = 29$ циклів.

Поділ на 2 такти дозволяє сформувати довільне зображення в динамічному режимі з двох непересічних множин збуджених елементів матриці шкали. Цикли формуються шляхом обчислення поточних значень старших і молодших розрядів матриці.

ISSN 2663 - 4023

Підпрограму першого такту на основі двотактної ІМ подає алгоритм, наведений на рис. 7.

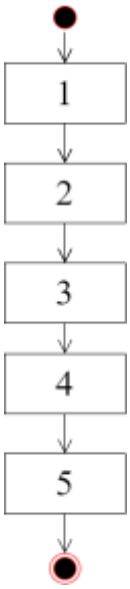

Рис.7 Алгоритм підпрограми першого такту

Блок 1 завантажує з пам'яті МК бітову маску для контролюючого порту та записує її в регістр R0. Блок 2 перевіряє кількість десятків в числі, що відображається: якщо воно більше 8-ми, тоді маска застосовується до порту P2, якщо менша – до P3. Така перевірка необхідна, так як один порт має 8 ліній і керується за допомогою восьми бітів. В разі, якщо кількість десятків більше за 8, тоді потрібно задіяти порт P2 і використати бітову маску для активації ліній цього порту, так як саме до них приєднані ключі, які вмикають потрібні лінії. Якщо кількість десятків більше за 8 застосовується бітова маска для порту P3. Блок 3 виконує перевірку кількості одиниць числа, яке виводиться на шкалу: якщо воно менше або дорівнює 8 – активується порт P0. Це вимкне перших вісім СД. Якщо число буде більшим – застосовується маска з регістра R0 для порту P2, а регістр завантажується нульове значення. Блок 4 завантажує бітову маску для активації необхідних сегментів інформаційного поля. Блок 5 закінчує виконання підпрограми і здійснює повернення до основної програми. Підпрограма першого такту має 14 команд та виконується за $c^B{}_2 = 21$ цикл.

Підпрограму другого такту на основі двотактної ІМ реалізує алгоритм, наведений на рис. 8.

ISSN 2663 - 4023

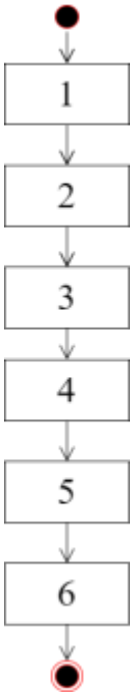

Рис.8 Алгоритм підпрограми другого такту

Блок 1 завантажує з пам'яті МК бітову маску для збудження елементів шкали у другому такті та записує її в регістр R1. Блок 2 виконує перевірку: якщо кількість одиниць менша або дорівнює 8 – активується увесь порт P0. Це вимкне перших вісім СД. В протилежному випадку – для порта P1 застосовується маска з регістра R1, а в регістр завантажується нульове значення. Блок 3 перевіряє чи кількість десятків перевищує 8 і завантажує бітову маску до відповідного порту P2 або P3. Блок 4 перевіряє, чи більша кількість одиниць за 8 і завантажує маску до порту P2 або P3. Блок 5 застосовує маску до обраного порту: якщо це порт P2 – виконується операція логічного «АБО» з регістром R1 для поєднання з маскою першого такту. Блок 6 активує необхідні елементи шкали, та повертає МК до основної програми. Підпрограма другого такту складається з 24 команд і виконується за $c^B_3 = 28$ циклів.

Підпрограма перезапуску таймера не відрізняється від багатотактного аналога, отже складається з 4 команд і теж виконується за $c^B_4 = 7$ циклів.

Таким чином, відповідно (2) загальна кількість циклів двотактної програми дорівнює

$$C^B = \sum_{i=1}^{4} c^B_i = 29 + 21 + 28 + 7 = 85 \ .$$

При типовій тактовій частоті МК 12 мГц цей код виконується за 85 мкс. Час формування повного зображення буде складати 170 мкс.

Другим важливим критерієм для порівняння програм є їх розмір, а саме обсяг байт-коду, який буде завантажено в пам'ять контролера.

Байт-код – машино-незалежний код низького рівня, що генерується транслятором і виконується процесором.

Після компіляції програм в однакових умовах отримано наступні результати:

- багатотактний варіант займає обсяг пам’яті 748 байт,
- двотактний варіант – 814 байт.

Порівняння варіантів реалізації програм шкальної індикації

Результати співставлення двотактного та багатотактного формування зображення на шкалі наведено в табл. 1.

Таблиця 1

Порівняння двотактної програми з багатотактною

Параметр	Двотактна програма C^B	Багатотактна програма C^M
Розмір, байт	814	748
Час формування повного зображення, мкс	170	1010

Багатотактна програма має невеликі переваги в розмірі, тобто в пам’яті МК вона буде займати менше місця. Слід зауважити, що враховуючи невеликий обсяг пам’яті програм МК, навіть невелика перевага може бути суттєвою. За швидкістю виконання переривання індикації багатотактний варіант значно програє двотактній програмі. Зменшення кількості тактів формування ІМ на інформаційному полі є одним із найефективніших шляхів підвищення надійності динамічних пристроїв відображення інформації. Це функціональне рішення одночасно покращує ергономічні характеристики, спрощує конструкцію та знижує струмові навантаження на СД і вихідні ланцюги системи індикації, підвищуючи надійність технічної реалізації [22].

ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У роботі були розроблені та порівняні програми традиційного багатотактного виводу інформації шляхом сканування матриці елементів індикатора по одній з координат та його двотактного функціонального аналога для шкальних засобів виводу даних у вбудованих системах на основі МК. Проведений аналіз показав що:

- багатотактна програма має невеликі переваги в розмірі перед двотактною;
- витрати машинного часу однокристального МК на підтримку підсистеми індикації двотактного варіанту є в 6 разів меншими в порівнянні з багатотактним, що вивільняє обчислювальні ресурси для актуальних задач вбудованих технічних засобів.

Отримані результати можуть слугувати основою для ефективного вирішення завдань щодо підвищення рівня техніко-економічних показників серійних і спеціалізованих пристроїв, а також для спрощення їх інтеграції та впровадження в актуальні автоматизовані засоби керування складними процесами та об'єктами різного призначення, де суттєву роль відіграє людина-оператор.

Подальші розвідки доцільно зосередити на розробці та дослідженні інформаційних моделей з мінімізованою кількістю тактів формування зображення на індикаторі та їх програмних реалізаціях. Такі технічні рішення мають бути спрямовані на покращення техніко-економічних та ергономічних параметрів засобів відображення інформації для

відповідальних застосувань на пультах операторів автоматизованих комплексів та мобільних об'єктів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Miller M. (2019). *Color in Electronic Display Systems*. Springer. 248 p.
2 Linliu K. (2018). *Micro-LED Display*. KDP Print US. 140 p.
3 Bushma, A. V. (2008). Matrix models of bar graph data display for bicyclic excitation of the optoelectronic scale. *Semiconductor physics, quantum electronics and optoelectronics*, *11*(2), 188–195. https://doi.org/10.15407/spqeo11.02.188
4 Bushma, A. V., Turukalo, A. V. (2020). Software controlling the LED bar graph displays. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *23*(3), 329–335. https://doi.org/10.15407/spqeo23.03.329
5 Бушма А.В., Ярцев В.П. (2014). Многотактное формирование дискретно-аналоговых форм представления сообщений на светодиодной шкале. *Сучасний захист інформації*, 4–9.
6 Beaty, H. W., Santoso, S. (2018). *Standard Handbook for Electrical Engineers, Seventeenth Edition*. McGraw-Hill Education.
7 Sanchez, J., Canton, M. P. (2018). The Microchip PIC. У Microcontroller Programming (p.p. 129–140). CRC Press.
8 Canton, M. P., Sanchez, J. (2013). Microcontrollers: High-Performance Systems and Programming. Taylor & Francis Group.
9 Steiner, C. (2005). The 8051/8052 Microcontroller: Architecture, Assembly Language, And Hardware Interfacing. Universal Publishers.
10 Bushma, O., Turukalo, A. (2021). Multi-element scale indicator devices in built-in systems. *Cybersecurity: Education, Science, Technique*, *3*(11), 43–60. https://doi.org/10.28925/2663-4023.2021.11.4360
11 Sagar, D. K. (2011). Microcontroller 8051. Alpha Science International, Limited.
12 Бушма, А. В., Сукач, Г. А., Ярцев, В. П. (2006). Многотактное формирование дискретно-аналоговых форм представления сообщений на светодиодной шкале. *Приборы и системы. Управление, контроль, диагностика*, (9), 16–21.
13 Bushma, A. V. (2008). Matrix models of bar graph data display for bicyclic excitation of the optoelectronic scale. *Semiconductor physics, quantum electronics and optoelectronics*, *11*(2), 188–195. https://doi.org/10.15407/spqeo11.02.188
14 Bushma, A. V. (2002). Control circuits for LED positional indicator. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *5*(4), 442–448. https://doi.org/10.15407/spqeo5.04.442
15 Gimenez, S. P. (2019). *8051 Microcontrollers*. Springer International Publishing. https://doi.org/10.1007/978-3-319-76439-9
16 Gupta, G. S., Mukhopadhyay, S. C. (2010). *Embedded Microcontroller Interfacing*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-13636-8
17 Gimenez, S. P. (2018). Input/Output Ports of 8051 Core Microcontrollers. У *8051 Microcontrollers* (с. 191–224). Springer International Publishing. https://doi.org/10.1007/978-3-319-76439-9_6
18 Mazidi, M. A. (2013). *The 8051 microprocessor: A systems approach*. Pearson.
19 MacKenzie, I. S., Phan, R. C.-W. (2006). *8051 Microcontroller*. Pearson Education, Limited.
20 *Embedded Systems Robots Projects Using The 8051 Microcontroller*. (2009). Cengage Learning.
21 Kapadia, R. (2005). *8051 Microcontroller and Embedded Systems*. Jaico Publishing House.
22 Mayes, K., Markantonakis, K. (2013). *Secure Smart Embedded Devices, Platforms and Applications*. Springer.
23 Rafiquzzaman, M. (2014). *Fundamentals of Digital Logic and Microcontrollers*. Wiley & Sons, Incorporated, John.
24 Parab, J. S., Shinde, S. A., Shelake, V. G., Kamat, R. K., Naik, G. M. (2008). *Practical Aspects of Embedded System Design using Microcontrollers*. Springer Netherlands. https://doi.org/10.1007/978-1-4020-8393-8
25 Mishra, J. (2006). *Embedded Systems.* Alpha Science International, Ltd.
26 Bushma, A. V. (2002). Model of dynamic indication in the bar graph form. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *5*(2), 193–196. https://doi.org/10.15407/spqeo5.02.193

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 4 (16), 2022
ISSN 2663 - 4023

Bushma Oleksandr Volodymyrovych
Doctor of Technical Sciences, Professor,
Professor of the Department of Computer Science and Mathematics,
Borys Grinchenko Kyiv University, Kyiv, Ukraine
ORCID ID 0000-0003-1604-6129
o.bushma@kubg.edu.ua

Turukalo Andrii Valeriiovych
Postgraduate student of the Department of Computer Science,
National University of Life and Environmental Sciences of Ukraine,
Kyiv, Ukraine
ORCID ID 0000-0003-2944-1806
tyrykalo@gmail.com

EVALUATION OF PARAMETERS IN SOFTWARE IMPLEMENTATION BAR GRAPH DISPLAY DEVICES

Abstract. The work is devoted to the estimation of resource consumption of the microcontroller for the synthesis of bar graph discrete-analog data display on the LED information field in bicyclic mode. The paper compares the programs of multicycle information output and its bicyclic analogue for discrete-analog means for the information display systems. The significance of the influence of bicyclic information models on the minimization of machine time resources of a single-chip microcontroller is shown. It is determined that in order to reduce resource consumption in the developed solutions it is necessary to focus on the maximum possible optimization of program blocks that are executed during interrupts and serve the I / O subsystem of the microcontroller. In this sense the bar graph displays based on the bicyclic additive information model has the best results. It was found that the traditional approach to assessing the effectiveness of programs using special benchmark programs, with subsequent measurement of code and execution time of the entire program does not allow to correctly assess the effectiveness of the program and the work of microcontroller at the device design stage. Therefore, as an alternative, it was proposed to use the size of the bytecode of the program and the speed of the main cycle - the procedure of information output to assess the effectiveness of the program. It was found that in terms of speed of execution and consumption of resources, the multicycle version significantly loses to the bicyclic program. Also, reducing the number of image formation cycles in the information field is one of the most effective way to minimize the consumption of microcontroller resources for display services.

Keywords: consumption of microcontroller resources; scale; bar graph display; bicyclic information model; microcontroller; discrete-analog indication; software implementation; LED; dynamic mode.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1 Miller M. (2019). *Color in Electronic Display Systems*. Springer. 248 p.
2 Linliu K. (2018). *Micro-LED Display*. KDP Print US. 140 p.
3 Bushma, A. V. (2008). Matrix models of bar graph data display for bicyclic excitation of the optoelectronic scale. *Semiconductor physics, quantum electronics and optoelectronics*, *11*(2), 188–195. https://doi.org/10.15407/spqeo11.02.188
4 Bushma, A. V., Turukalo, A. V. (2020). Software controlling the LED bar graph displays. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *23*(3), 329–335. https://doi.org/10.15407/spqeo23.03.329
5 A.V. Bushma, V.P. Yartsev (2014). “Multi-cyclic formation of discrete-analog forms of message representation at the LED scale”. *Suchasnyi zakhyst informatsii*. (1), p.4–9.
6 Beaty, H. W., Santoso, S. (2018). *Standard Handbook for Electrical Engineers, Seventeenth Edition*. McGraw-Hill Education.
7 Sanchez, J., Canton, M. P. (2018). The Microchip PIC. У Microcontroller Programming (с. 129–140). CRC Press.

8 Canton, M. P., Sanchez, J. (2013). Microcontrollers: High-Performance Systems and Programming. Taylor & Francis Group.
9 Steiner, C. (2005). The 8051/8052 Microcontroller: Architecture, Assembly Language, And Hardware Interfacing. Universal Publishers.
10 Bushma, O., Turukalo, A. (2021). Multi-element scale indicator devices in built-in systems. *Cybersecurity: Education, Science, Technique*, *3*(11), 43–60. https://doi.org/10.28925/2663-4023.2021.11.4360
11 Sagar, D. K. (2011). Microcontroller 8051. Alpha Science International, Limited.
12 A. V. Bushma, G. A. Sukach, V.P. Yartsev (2006). "Multi-cycle formation of discrete-analog forms of message representation on the LED scale". *Pribory i Sistemy. Upravlenie, kontrol, diagnostika*. (9), 16–21.
13 Bushma, A. V. (2008). Matrix models of bar graph data display for bicyclic excitation of the optoelectronic scale. *Semiconductor physics, quantum electronics and optoelectronics*, *11*(2), 188–195. https://doi.org/10.15407/spqeo11.02.188
14 Bushma, A. V. (2002). Control circuits for LED positional indicator. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *5*(4), 442–448. https://doi.org/10.15407/spqeo5.04.442
15 Gimenez, S. P. (2019). *8051 Microcontrollers*. Springer International Publishing. https://doi.org/10.1007/978-3-319-76439-9
16 Gupta, G. S., Mukhopadhyay, S. C. (2010). *Embedded Microcontroller Interfacing*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-13636-8
17 Gimenez, S. P. (2018). Input/Output Ports of 8051 Core Microcontrollers. У *8051 Microcontrollers* (с. 191–224). Springer International Publishing. https://doi.org/10.1007/978-3-319-76439-9_6
18 Mazidi, M. A. (2013). *The 8051 microprocessor: A systems approach*. Pearson.
19 MacKenzie, I. S., Phan, R. C.-W. (2006). *8051 Microcontroller*. Pearson Education, Limited.
20 *Embedded Systems Robots Projects Using The 8051 Microcontroller*. (2009). Cengage Learning.
21 Kapadia, R. (2005). *8051 Microcontroller and Embedded Systems*. Jaico Publishing House.
22 Mayes, K., Markantonakis, K. (2013). *Secure Smart Embedded Devices, Platforms and Applications*. Springer.
23 Rafiquzzaman, M. (2014). *Fundamentals of Digital Logic and Microcontrollers*. Wiley & Sons, Incorporated, John.
24 Parab, J. S., Shinde, S. A., Shelake, V. G., Kamat, R. K., Naik, G. M. (2008). *Practical Aspects of Embedded System Design using Microcontrollers*. Springer Netherlands. https://doi.org/10.1007/978-1-4020-8393-8
25 Mishra, J. (2006). *Embedded Systems.* Alpha Science International, Ltd.
26 Bushma, A. V. (2002). Model of dynamic indication in the bar graph form. *Semiconductor Physics, Quantum Electronics and Optoelectronics*, *5*(2), 193–196. https://doi.org/10.15407/spqeo5.02.193