509686-X (pbk).

19. Oswald U. Graphics in LATEX2", containing some Java source files for generating arbitrary circles and ellipses within the picture environment, and METAPOST / Urs Oswald [A Tutorial] . - Режим доступу до ресурсу.: http://www.ursoswald.ch.

20. Tantau T. TikZ&PGF Manual / Till Tantau. - Режим доступу до ресурсу.: CTAN://graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf.

21. Burt, John. Using poemscol for Critical Editions of Poetry / Burt, John [In:.e PracTEX Journal 3. 2005]. - Режим доступу до ресурсу.: http://www.tug.org/pracjourn/index.html.

22. Braams. LATEX2 / Braams, Johannes et al. - Режим доступу до ресурсу.: http://www.tug.org/texlive/Contents/live/texmf-dist/doc/latex/base/source2e.pdf.

23. Fenn, Jürgen. Managing Citations and Your Bibliography with BibTEX / Fenn, Jürgen [PracTEX Journal]. - Режим доступу до ресурсу.: http://www.tug.org/pracjourn/2006-4/fenn.

Надійшла 29.9.2012 р.
Рецензент: д.т.н. Поморова О.В,

УДК 681.3.04

І.А. ДИЧКА, М.В. ОНАЙ, О.В. ВАЩІЛІН
Національний технічний університет України «Київський політехнічний інститут»

АПАРАТНА РЕАЛІЗАЦІЯ ОПЕРАТОРІВ ТА ФУНКЦІЙ В ПОЛЯХ ГАЛУА

У статті обґрунтовано, що швидкодії при завадостійкому кодуванні даних та при виконанні криптографічних перетворень існує доцільність з метою побудови спеціалізованих апаратних засобів для виконання операцій в полях Галуа. Запропоновано два підходи для апаратної реалізації операторів та функцій в полях Галуа та докладно розглянуто побудову функціонального вузла для кожного оператора.

In paper it was grounded that to improve performance when noiseless coding of data and when performing cryptographic transformations there is a necessity of building specialized hardware for performing operations in Galois fields. Two approaches have been proposed for the hardware implementation of operators and functions in Galois fields and it has been considered in detail the construction of a functional node for each operator.

Ключові слова: скінченне поле, схема цілочислового ділення, примітивний елемент.

Вступ

Основні принципи теорії скінченних полів (які часто називають полями Галуа) як одного з розділів математики, розроблені в працях Ферма, Ейлера, Гауса, Галуа та інших видатних вчених [1– 3]. До недавнього часу теорія скінченних полів розвивалась як галузь класичної математики [3]. Але у зв'язку з розвитком завадостійкого кодування та криптографії активно розвиваються прикладні аспекти теорії [4– 7].

Обчислення у скінченних полях мають свою специфіку, і їх програмна реалізація з використанням універсальних комп'ютерів та мов програмування є не завжди ефективною з точки зору забезпечення потрібної швидкодії. Тому актуальною є проблема апаратної або апаратно-програмної реалізації обчислень у полях Галуа [8].

Постановка задачі

Аналіз класів задач, які мають місце в завадостійкому кодуванні та при криптографічному захисті інформації, показує, що можливі два підходи до створення апаратних засобів для реалізації обчислень в полях Галуа – побудова спеціалізованих комп'ютерних систем (СКС) та побудова проблемно-орієнтованих процесорів (ПОПр).

Для реалізації зазначених обчислювальних засобів, перш за все, необхідно розглянути питання створення спеціалізованих функціональних вузлів, на основі яких слід проектувати СКС та ПОПр.

Апаратний синтез операторів та функцій у скінченних полях

Розглянемо апаратну реалізацію операторів та функцій (табл. 1), які використовуються як функціонально завершені вузли для побудови функціональних блоків (спеціалізованих процесорів), орієнтованих на реалізацію арифметики полів Галуа.

Оператор обчислення лишку числа A ***за модулем*** N_{Ω} $(A \bmod N_{\Omega})$**.** Нехай $A \in GF(N_{\Omega})$, A – ціле число. У загальному випадку для обчислення $A \bmod N_{\Omega}$ (мнемонічне позначення – $\mathrm{RES}(A)$, RESidue – лишок) слід скористатися схемою цілочислового ділення (СЦД) A на N_{Ω} (рис. 1), результатом якого є частка і остача. Остачу позначимо $A \bmod N_{\Omega}$, вона є лишком числа за модулем N_{Ω} (часткою при цьому нехтують).

СЦД можна реалізувати мікропрограмно.

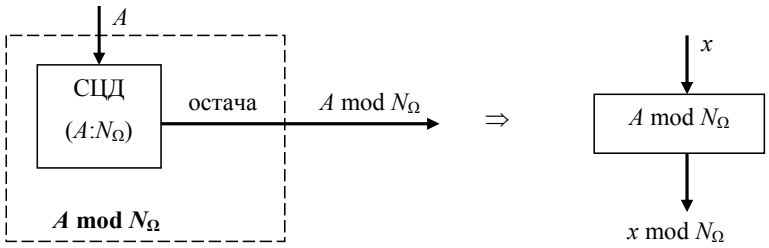

Рис. 1. Схемна реалізація оператора A mod N_Ω у загальному випадку

Таблиця 1

Оператори та функції в полях Галуа, які використовуються в завадостійкому кодуванні та криптографії

Назва оператора (функції)	Призначення оператора (функції)		Примітка
	Аналітичне	Мнемонічне	
Обчислення лишку числа за модулем N_Ω	$A \bmod N_\Omega$	$\mathrm{RES}(A)$	Реалізується мікропрограмо (рис. 1)
Обчислення протилежного елемента поля $GF(N_\Omega)$	$-A \bmod N_\Omega$	$\mathrm{CONTR}(A)$	Реалізується схемно (рис. 2)
Додавання елементів поля $GF(N_\Omega)$	$A + B$	ADD A, B	Реалізується схемно (рис. 3)
Віднімання елементів поля $GF(N_\Omega)$	$A - B$	SUB A, B	Реалізується схемно (рис. 4)
Множення елементів поля $GF(N_\Omega)$	$A \cdot B$	MUL A, B	Реалізується мікропограмно та схемно (рис. 5)
Піднесення примітивного елемента α поля $GF(N_\Omega)$ до степеня	$\alpha^z \bmod N_\Omega$	$\mathrm{EXP}(Z)$	Реалізується мікропограмно та схемно (рис. 6, 7)
Обчислення логарифма [від значення] елемента поля $GF(N_\Omega)$	$\log_\alpha A$	$\mathrm{LOG}(A)$	Реалізується схемно (рис. 8)
Обчислення оберненого елемента поля $GF(N_\Omega)$	A^{-1}	$\mathrm{INV}(A)$	Реалізується схемно (рис. 9)
Ділення елементів поля $GF(N_\Omega)$	A/B	DIV A, B	Реалізується схемно (рис. 10)
Обчислення значення многочлена в заданій точці	$c(W)$	$\mathrm{GRN}(c(x), W)$	Реалізується схемно (рис. 11)

Оператор знаходження протилежного елемента поля. Нехай $A \in GF(N_\Omega)$. Необхідно обчислити елемент $-A \in GF(N_\Omega)$, такий, що $A + (-A) = 0$.

У полі $GF(N_\Omega)$ є справедливою рівність $-A = N_\Omega - A$. Тому для знаходження протилежного елемента поля потрібно виконати операцію $N_\Omega - A$ (рис. 2), для реалізації якої необхідно використати комбінаційний суматор (См) та регістр (Рг) для зберігання значення модуля N_Ω.

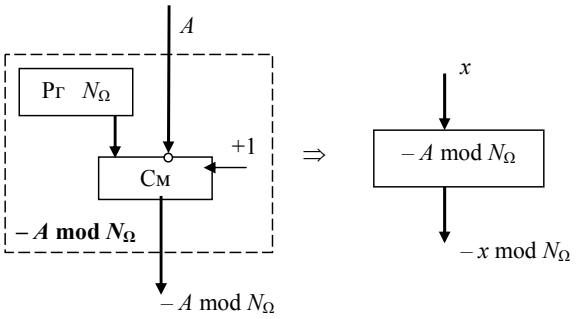

Рис. 2. Схемна реалізація оператора – *A* mod N_Ω

Даний оператор позначатимемо аналітично $-\boldsymbol{A} \bmod \boldsymbol{N_\Omega}$, мнемонічно $\mathrm{CONTR}(A)$; CONTRary – протилежний.

Додавання елементів поля за модулем N_Ω. Нехай A, B – елементи поля $GF(N_\Omega)$.

Для реалізації операції необхідні два комбінаційні суматори, регістр та мультиплексор (рис. 3).

Результат додавання $C = A + B$ тимчасово зберігається в Рг, а на См2 виконується операція $D = C - N_\Omega$. Якщо $D < 0$, то на вихід мультиплексора (МХ) надходить *C*, інакше – *D*.

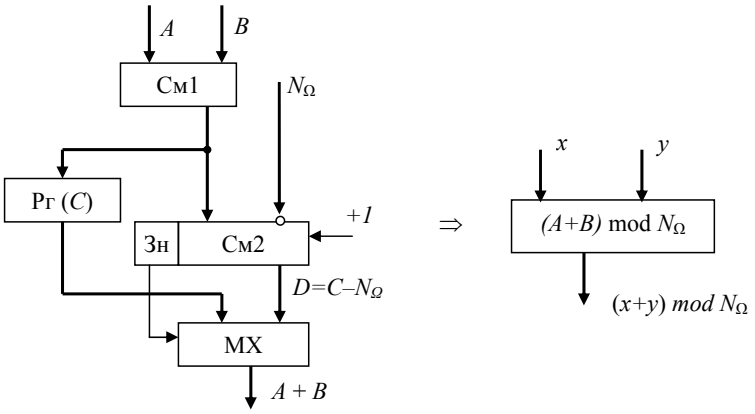

Рис. 3. Схемна реалізація оператора *A + B*

Даний оператор позначатимемо: аналітично $\boldsymbol{A + B}$, мнемонічно ADD *A*, *B*.

Віднімання елементів поля за модулем N_Ω. Нехай X, Y – елементи поля. Необхідно обчислити $X - Y$.

У полі $GF(N_\Omega)$ $X - Y = X + (-Y) = X + (N_\Omega - Y)$. Тобто операцію віднімання можна замінити операцією додавання елемента X та протилежного елемента $-Y$.

Таким чином, для виконання *X– Y* необхідно послідовно виконати два оператори: оператор знаходження протилежного елемента поля та оператор додавання елементів поля (рис. 4).

Мнемонічно даний оператор позначатимемо SUB *A*, *B*, а аналітично $\boldsymbol{A - B}$.

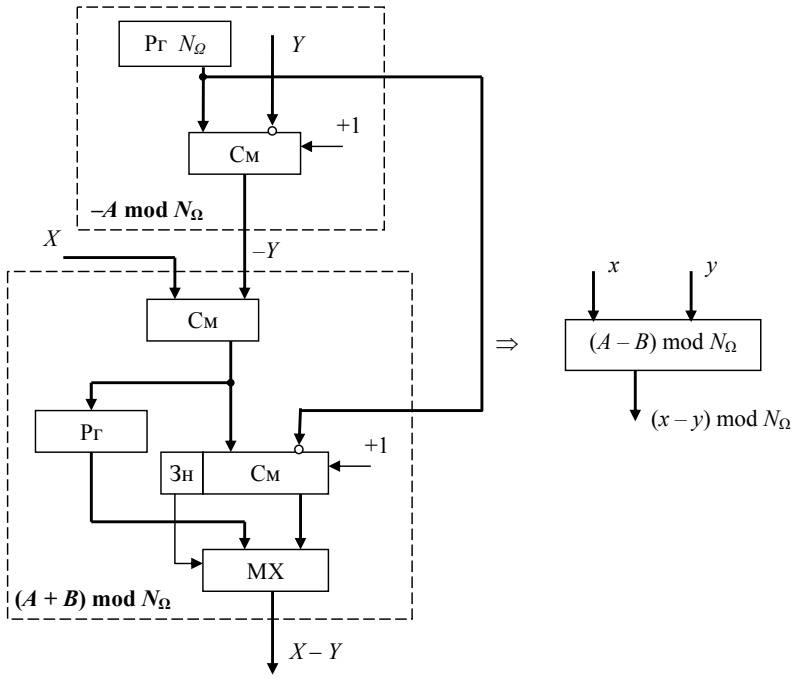

Рис. 4. Схемна реалізація оператора *A – B*

Множення елементів поля за модулем N_Ω. Нехай A, B – елементи поля, $A, B \in GF(N_\Omega)$. Необхідно обчислити $C = A \cdot B$, $C \in GF(N_\Omega)$. Операцію $C = A \cdot B$ можна реалізувати за два прийоми: спочатку виконати множення двійкових чисел без знаку A і B з використанням схеми множення (СхМн) двійкових чисел, а потім до результату C застосувати оператор $\boldsymbol{A \bmod N_\Omega}$ (оператор обчислення лишку числа (C) за модулем N_Ω) (рис. 5).

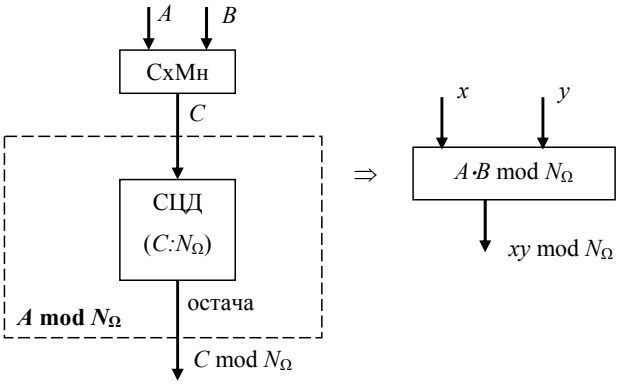

Рис. 5. Схемна реалізація оператора $A \cdot B$

Аналітично оператор позначатимемо $\boldsymbol{A \cdot B}$, а мнемонічно $\mathrm{MUL}\ A, B$.

Оператор піднесення примітивного елемента α поля GF ($N_Ω$) до цілого степеня. У загальному випадку для обчислення $\alpha^z \bmod N_\Omega$ необхідно мати блок піднесення значення α до степеня z $\left(\alpha^z\right)$ та СЦД (рис. 6).

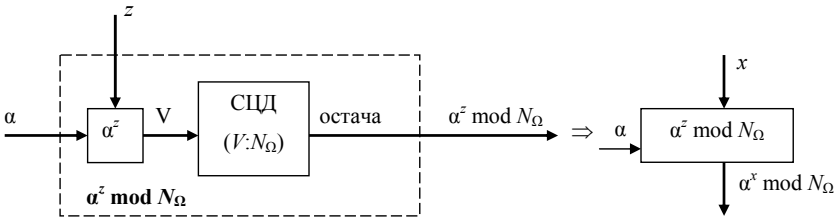

Рис. 6. Узагальнена схема реалізації оператора $\alpha^z \bmod N_\Omega$

Даний оператор позначатимемо: аналітично $\boldsymbol{\alpha^z \bmod N_\Omega}$, мнемонічно $\mathrm{EXP}(Z)$.

Але на практиці α зазвичай дорівнює 2, 3 або 5. Якщо α заздалегідь відоме, то схему обчислення $\alpha^z \bmod N_\Omega$ можна істотно спростити.

Нехай $\alpha = 2$. Побудуємо схему обчислення $2^z \bmod N_\Omega$ (рис. 7).

В лічильник (Лч), який працює в режимі декременту, записуємо величину z показника степеня, а в Рг1 значення 2 $(\alpha = 2)$.

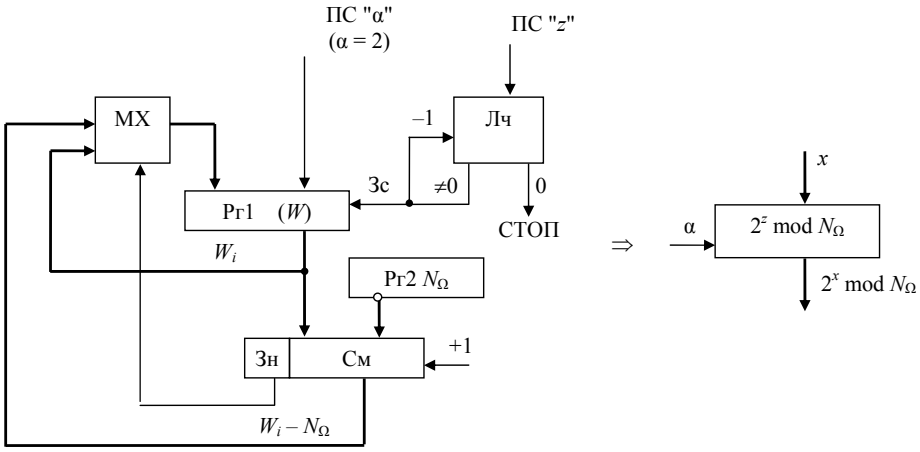

Рис. 7. Схемна реалізація оператора $\alpha^z \bmod N_\Omega$ при α = 2

Для отримання значення 2^z достатньо здійснити z зсувів Рг1 вліво. З метою зменшення розрядності доцільно в Рг1 зберігати не 2^i, а $2^i \bmod N_\Omega$. Для забезпечення цього у кожному такті роботи схеми обчислюємо значення $W_i = 2 \cdot \left(2^{i-1} \bmod N_\Omega\right)$ (в регістрі Рг1) та значення $W_i - N_\Omega$ (на См). Якщо знаковий розряд в См дорівнює 1, то це означає, що $W_i < N_\Omega$, і в Рг1 через МХ надходить значення W_i; якщо ж у знаковому розряді 0, то це означає, що $W_i \geq N_\Omega$, і в Рг1 записується значення $W_i - N_\Omega$. В такий спосіб

фактично здійснюється мікрооперація $W_i \bmod N_\Omega$, і, отже, перед черговим зсувом Рг1 завжди маємо $2^i \bmod N_\Omega$ (лишок числа 2^i за модулем N_Ω). Після z-го такту ((Лч) = 0) в Рг1 отримаємо $2^z \bmod N_\Omega$.

Особливістю схеми є те, що вона не потребує вузла цілочислового ділення (див. рис. 6).

Оператор знаходження оберненого елемента A^{-1} поля. Нехай $A \in GF\left(N_\Omega\right)$. Необхідно знайти $A^{-1} \in GF\left(N_\Omega\right)$ такий, що $A \cdot A^{-1} = 1$.

Кожний елемент поля можна подати у вигляді степеня α. Нехай $A = \alpha^a$. Тоді $A^{-1} = \alpha^{-a} = \alpha^{N_\Omega - a}$.

Даний оператор позначатимемо: $\boldsymbol{A^{-1}}$ – аналітично, $\mathrm{INV}(A)$ – мнемонічно (INVerse – обернений).

Оператор обчислення логарифма елемента поля за основою α. Для знаходження $a = \log_\alpha A$ у полі $GF(N_\Omega)$ необхідно послідовно починаючи з $a = 1$ (а потім $2, 3$, і т.д. можливо до $N_\Omega - 1$) обчислювати $\alpha^a \bmod N_\Omega$ та порівнювати його зі значенням A (рис. 8). При деякому a значення α^a збіжиться з A, тоді $a = \log_\alpha A$.

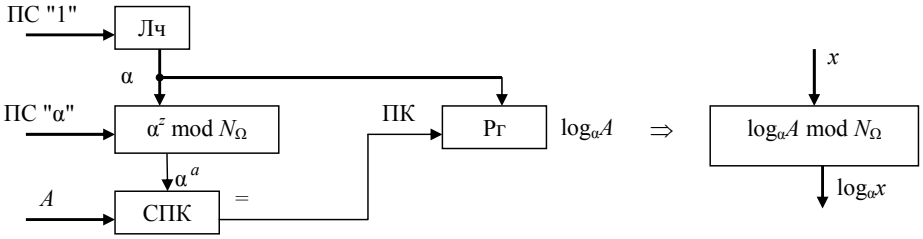

Рис. 8. Схемна реалізація оператора обчислення логарифма від елемента поля *GF* (N_Ω)

Максимальна кількість ітерацій при цьому може становити $N_\Omega - 1$.

Таким чином оператор $\log_\alpha A$ служить для знаходження показника степеня, до якого слід піднести примітивний елемент α поля, щоб отримати значення A.

Тоді для знаходження оберненого елемента $\left(A^{-1}\right)$ необхідно:

1) обчислити $\log_\alpha A$;

2) обчислити $\alpha^{N_\Omega - \log_\alpha A}$ (рис. 9).

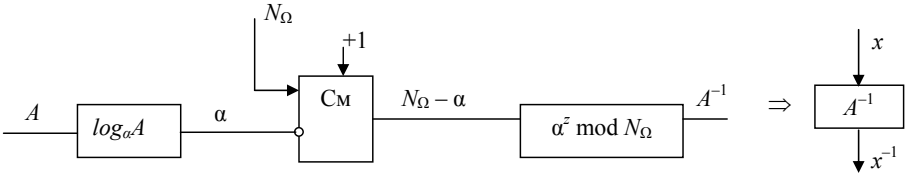

Рис. 9. Схема обчислення оберненого елемента (A^{-1}) у полі *GF* (N_Ω)

Оператор ділення елементів поля GF (N_Ω). Для реалізації оператора ділення A/B, де $A, B \in GF(N_\Omega)$ скористаємось поданням елементів поля у вигляді степеня α. Нехай $A = \alpha^a$, $B = \alpha^b$. Оскільки $A = \alpha^a = \alpha^{\log_\alpha A}$, a $B = \alpha^b = \alpha^{\log_\alpha B}$, то $A/B = \alpha^{\log_\alpha A - \log_\alpha B} = \alpha^{a-b}$. Тоді для реалізації ділення необхідно скористатися оператором обчислення логарифма елемента поля та оператором піднесення α до степеня (рис. 10).

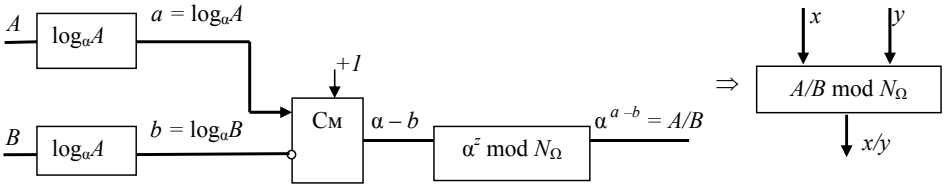

Рис. 10. Схемна реалізація оператора *A/B* ділення елементів поля *GF* (N_Ω)

Оператор обчислення значення многочлена в заданій точці. Нехай задано многочлен $c(x)=\sum_{i=0}^{n-1} c_i x^i$, коефіцієнти c_i якого є елементами поля $GF(N_\Omega)$, і необхідно обчислити його значення у точці W $(x=W)$.

Оскільки будь-який елемент поля можна подати у вигляді степеня α, вважатимемо, що $W=\alpha^w$.

Обчислимо значення многочлена в точці $x=W$ за схемою Горнера:

$$c(W)=(\ldots(((0+c_{n-1})W+c_{n-2})W+c_{n-3})W+\ldots+c_1)W+c_0$$

Спроектуємо схему для обчислення $c(W)$ *c (W)* (рис. 11). Вважатимемо, що коефіцієнти c_i многочлена розміщено в пам'яті.

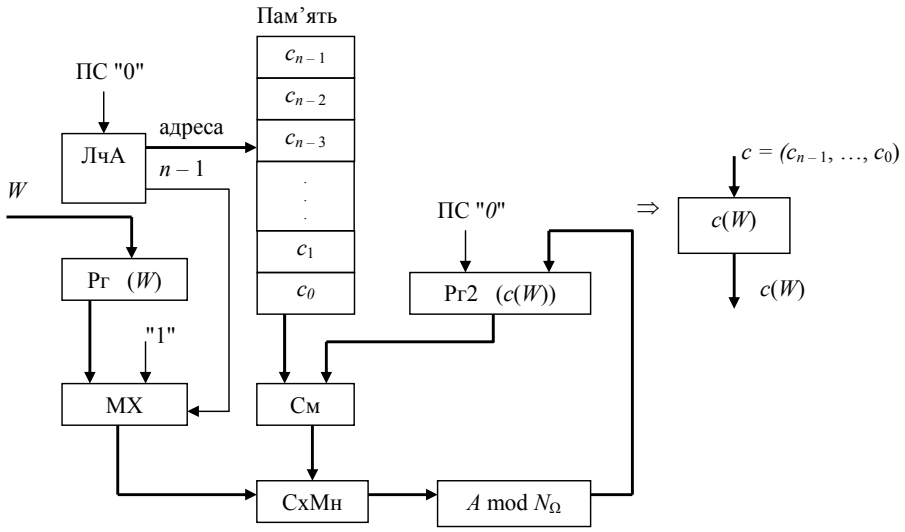

Рис. 11. Схемна реалізація оператора обчислення значення многочлена *c* (*x*) в точці *W* за схемою Горнера

У початковому стані $(\text{Рг1})=W=\alpha^1$; $(\text{Рг2})=0$, $(\text{ЛчА})=0$. З пам'яті зчитуємо c_{n-1} і на См додаємо з нулем (з Рг2). Схема множення виконує операцію $(0+c_{n-1})W$. Лишок результату запам'ятовується в Рг2.

У поточному такті виконується така послідовність операцій: читання c_i з пам'яті; підсумовування $c_i+(\text{Рг2})$; множення на W: $(c_i+(\text{Рг2}))\cdot W$; обчислення лишку результату (оператор $\boldsymbol{A \bmod N_\Omega}$); запис в Рг2.

В останньому ((*n* – 1)-у) такті у схему множення замість значення *W* подається одиниця.

Даний оператор позначатимемо: мнемонічно – $\text{GRN}(c(x), W)$, аналітично – $\boldsymbol{c(W)}$.

Особливості організації обчислень в полях Галуа

При завадостійкому кодуванні інформації особливістю обчислень в полях Галуа є мала розрядність даних та використання лише одного формату даних – цілих чисел без знаку, що дозволяє будувати спеціалізовані обчислювальні засоби з малою розрядністю даних, спрощеною архітектурою і системою команд. При цьому забезпечується обробка інформації в реальному часі.

Обчислення можуть здійснюватися за допомогою спеціалізованої системи, що складається з функціональних блоків, кожен з яких реалізує певну обчислювальну процедуру, з'єднаних між собою відповідно до алгоритму обробки. Її особливістю є апаратна реалізація переважної більшості обчислювальних процедур на основі операцій в полях Галуа, що завдяки високій швидкодії забезпечує обробку в реальному часі з більш високою продуктивністю порівняно з універсальними обчислювальними засобами.

Іншим варіантом організації обчислень є застосування структури проблемно-орієнтованого процесора (рис. 12), яка складається з універсального процесора (*M*-процесор) та спеціалізованого розширювача – співпроцесора Галуа (*G*-процесора), який реалізує операції у скінченних полях. За рахунок введення до системи команд універсального процесора спеціальних команд (табл. 1), які позначають процедури, що найчастіше зустрічаються в задачах обробки даних у скінченних полях і реалізуються *G*-процесором, підвищується продуктивність системи і забезпечується обробка в реальному часі.

Особливостями архітектури *G*-процесора є мала розрядність даних, один формат даних – цілі числа без знаку, спрощена система команд та мікропрограмний спосіб реалізації спецкоманд.

G-процесор з'єднується з *M*-процесором за допомогою шини даних (ШД) адреси (ША) та керування (ШК). Для з'єднання може використовуватися також контролер переривань (КП) (рис. 12).

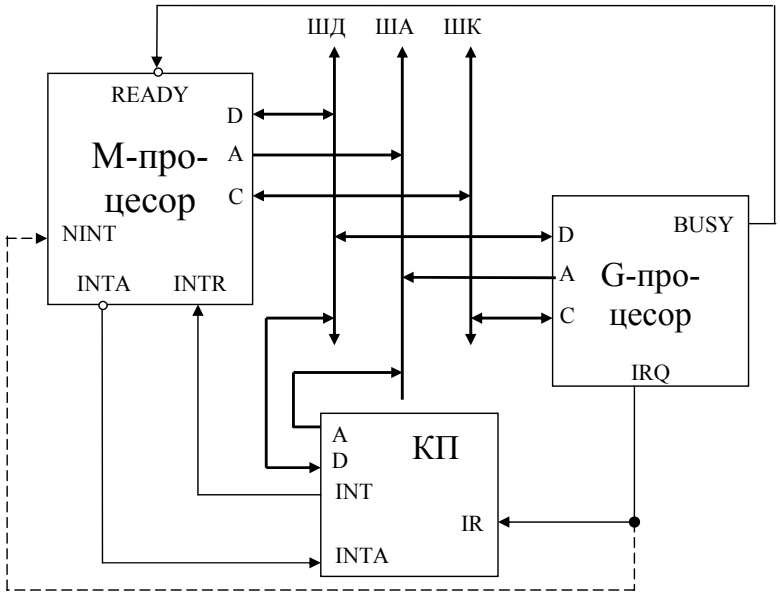

Рис. 12. Узагальнена структурна організація ПОПр для реалізації арифметики скінченних полів

Характерною особливістю обчислень у криптографії є велика довжина даних. У цьому випадку при виконанні операцій слова даних слід ділити на декілька частин так, щоб вони відповідали розрядності співпроцесора (наприклад, 32-розрядні слова), обробляти частини слів окремо та об'єднувати їх в єдине результуюче слово.

За таких обставин апаратна реалізація процедур та функцій є чи не єдиним способом досягнення потрібної швидкодії.

Висновки

Реалізація обчислень у полях Галуа може бути ефективною лише за умови створення спеціалізованих обчислювальних засобів. При цьому можливі два шляхи спеціалізації – створення спеціалізованої обчислювальної системи або проблемно-орієнтованого процесора. В обох випадках слід розробляти спеціалізовані функціональні вузли, на основі яких слід проектувати обчислювальні засоби. Такі вузли мають виконувати специфічні оператори та функції, що характерні для обчислень у скінченних полях при криптографічних застосуваннях та завадостійкому кодуванні інформації.

Це дозволить істотно підвищити ефективність обчислень порівняно з універсальними обчислювальними засобами.

Література

1. Lidl R. Finite fields / Lidl R, Niederreiter H. Addison Wesley. – 1983.

2. Lejla Batina Hardware architectures for public key cryptography / Lejla Batinaa, Sıddıka Berna Ors, Bart Preneel, Joos Vandewalle. – INTEGRATION, The VLSI journal 34, 2003. – P. 1– 64.

3. Miguel Morales-Sandoval An area/performance trade-off analysis of a *GF* (2^m) multiplier architecture for elliptic curve cryptography / Miguel Morales-Sandoval, Claudia Feregrino-Uribe, René Cumplido, Ignacio Algredo-Badillo. – Computers and Electrical Engineering 35, 2009, P. 54– 58.

4. Ernest M, Klupsch S, Hauck O, Huss SA. Rapid prototyping for hardware accelerated elliptic curve public key cryptosystems. In: Proceedings of 12th IEEE workshop on rapid system prototyping, RSP'2001, Monterey, CA; June 2001, p. 24– 31.

5. G. Orlando, C. Paar, A scalable *GF* (*p*) elliptic curve processor architecture for programmable hardware, in: C.K. Ko- c, D. Naccache, C. Paar (Eds.), Proceedings of Workshop on Cryptograpic Hardware and Embedded Systems (CHES 2001), Lecture Notes in Computer Science, Vol. 2162, Springer, Berlin, Paris, France, 2001, pp. 356– 371.

6. W. Stallings, Cryptography and network security, second ed, Prentice-Hall, Englewood cliffs, 1999.

7. Serdar S. Erdem Polynomial Basis Multiplication over *GF* (2^m) / Serdar S. Erdem, Tu grul Yanık, Çetin K. Koç. – Acta Appl Math, 2006. P. 33 – 55.

8. Wu, H.: Bit-parallel finite field multiplier and squarer using polynomial basis. IEEE Trans. Comput. 51 (7), 750– 758 (July 2002).

Надійшла 10.9.2012 р.
Рецензент: д.т.н. Поморова О.В.