

**ZESZYTY
NAUKOWE
POLITECHNIKI
ŚLĄSKIEJ**

ANDRZEJ HŁAWICZKA

**REJESTRY LINIOWE –
ANALIZA, SYNTEZA I ZASTOSOWANIA
W TESTOWANIU UKŁADÓW CYFROWYCH**

P. 4474/97

ELEKTRONIKA

z. 9

**GLIWICE
1997**

POLITECHNIKA ŚLĄSKA

ZESZYTY NAUKOWE

Nr 1370



P. 4474 / 97

ANDRZEJ HŁAWICZKA

REJESTRY LINIOWE –
ANALIZA, SYNTEZA I ZASTOSOWANIA
W TESTOWANIU UKŁADÓW CYFROWYCH

OPINIODAWCY
Prof. dr hab. inż. Henryk Krawczyk
Dr hab. Andrzej Kraśniewski

KOLEGIUM REDAKCYJNE

REDAKTOR NACZELNY — Prof. dr hab. Zygmunt Kleszczewski
REDAKTOR DZIAŁU — Doc. dr inż. Zdzisław Pogoda
SEKRETARZ REDAKCJI — Mgr Elżbieta Leško

REDAKCJA
Mgr Roma Łoś

REDAKCJA TECHNICZNA
Alicja Nowacka

Wydano za zgodą
Rektora Politechniki Śląskiej

PL ISSN 1231-1596

Wydawnictwo Politechniki Śląskiej
ul. Kujawska 3, 44-100 Gliwice

Nakł. 110+83 Ark. wyd. 16 Ark. druk. 16,125 Papier offset. kl. III 70x100, 80 g
Oddano do druku 17.11.97 Podpis. do druku 17.11.97 Druk ukończ. w grudniu 1997 r.
Zam. 381/97

Fotokopie, druk i oprawę
wykonano w Zakładzie Graficznym Politechniki Śląskiej w Gliwicach

SPIS TREŚCI

Wstęp	1
1. WSTĘP	1
1.1. Wstęp	1
1.2. Zakres i cel	1
1.3. Zakres i cel	1
1.4. Zakres i cel	1
1.5. Zakres i cel	1
2. WYKONANIE PRAC	1
2.1. WYKONANIE PRAC	1
2.2. WYKONANIE PRAC	1
2.3. WYKONANIE PRAC	1
2.4. WYKONANIE PRAC	1
2.5. WYKONANIE PRAC	1
2.6. WYKONANIE PRAC	1
2.7. WYKONANIE PRAC	1
2.8. WYKONANIE PRAC	1
2.9. WYKONANIE PRAC	1
2.10. WYKONANIE PRAC	1
2.11. WYKONANIE PRAC	1
2.12. WYKONANIE PRAC	1
2.13. WYKONANIE PRAC	1
2.14. WYKONANIE PRAC	1
2.15. WYKONANIE PRAC	1
2.16. WYKONANIE PRAC	1
2.17. WYKONANIE PRAC	1
2.18. WYKONANIE PRAC	1
2.19. WYKONANIE PRAC	1
2.20. WYKONANIE PRAC	1
2.21. WYKONANIE PRAC	1
2.22. WYKONANIE PRAC	1
2.23. WYKONANIE PRAC	1
2.24. WYKONANIE PRAC	1
2.25. WYKONANIE PRAC	1
2.26. WYKONANIE PRAC	1
2.27. WYKONANIE PRAC	1
2.28. WYKONANIE PRAC	1
2.29. WYKONANIE PRAC	1
2.30. WYKONANIE PRAC	1
2.31. WYKONANIE PRAC	1
2.32. WYKONANIE PRAC	1
2.33. WYKONANIE PRAC	1
2.34. WYKONANIE PRAC	1
2.35. WYKONANIE PRAC	1
2.36. WYKONANIE PRAC	1
2.37. WYKONANIE PRAC	1
2.38. WYKONANIE PRAC	1
2.39. WYKONANIE PRAC	1
2.40. WYKONANIE PRAC	1
2.41. WYKONANIE PRAC	1
2.42. WYKONANIE PRAC	1
2.43. WYKONANIE PRAC	1
2.44. WYKONANIE PRAC	1
2.45. WYKONANIE PRAC	1
2.46. WYKONANIE PRAC	1
2.47. WYKONANIE PRAC	1
2.48. WYKONANIE PRAC	1
2.49. WYKONANIE PRAC	1
2.50. WYKONANIE PRAC	1
2.51. WYKONANIE PRAC	1
2.52. WYKONANIE PRAC	1
2.53. WYKONANIE PRAC	1
2.54. WYKONANIE PRAC	1
2.55. WYKONANIE PRAC	1
2.56. WYKONANIE PRAC	1
2.57. WYKONANIE PRAC	1
2.58. WYKONANIE PRAC	1
2.59. WYKONANIE PRAC	1
2.60. WYKONANIE PRAC	1
2.61. WYKONANIE PRAC	1
2.62. WYKONANIE PRAC	1
2.63. WYKONANIE PRAC	1
2.64. WYKONANIE PRAC	1
2.65. WYKONANIE PRAC	1
2.66. WYKONANIE PRAC	1
2.67. WYKONANIE PRAC	1
2.68. WYKONANIE PRAC	1
2.69. WYKONANIE PRAC	1
2.70. WYKONANIE PRAC	1
2.71. WYKONANIE PRAC	1
2.72. WYKONANIE PRAC	1
2.73. WYKONANIE PRAC	1
2.74. WYKONANIE PRAC	1
2.75. WYKONANIE PRAC	1
2.76. WYKONANIE PRAC	1
2.77. WYKONANIE PRAC	1
2.78. WYKONANIE PRAC	1
2.79. WYKONANIE PRAC	1
2.80. WYKONANIE PRAC	1
2.81. WYKONANIE PRAC	1
2.82. WYKONANIE PRAC	1
2.83. WYKONANIE PRAC	1
2.84. WYKONANIE PRAC	1
2.85. WYKONANIE PRAC	1
2.86. WYKONANIE PRAC	1
2.87. WYKONANIE PRAC	1
2.88. WYKONANIE PRAC	1
2.89. WYKONANIE PRAC	1
2.90. WYKONANIE PRAC	1
2.91. WYKONANIE PRAC	1
2.92. WYKONANIE PRAC	1
2.93. WYKONANIE PRAC	1
2.94. WYKONANIE PRAC	1
2.95. WYKONANIE PRAC	1
2.96. WYKONANIE PRAC	1
2.97. WYKONANIE PRAC	1
2.98. WYKONANIE PRAC	1
2.99. WYKONANIE PRAC	1
2.100. WYKONANIE PRAC	1

Mojej Żonie poświęcam

SPIS TREŚCI

Ważniejsze skróty	7
Ważniejsze symbole	9
1. WPROWADZENIE	13
1.1. Rola testerów wewnątrzukładowych BIST w rozwiązywaniu problemów testowania systemów komputerowych	15
1.2. Znaczenie rejestrów liniowych w upraszczaniu testowania	21
1.3. Struktury testerów wewnątrzukładowych z rejestrami liniowymi	26
1.4. Niektóre zagadnienia związane z wyborem, projektowaniem i stosowaniem rejestrów liniowych w testowaniu układów cyfrowych	50
2. STRUKTURY I OPIS ALGEBRAICZNY REJESTRÓW LINIOWYCH	55
2.1. Różne struktury rejestrów liniowych MISR	56
2.2. Algebraiczne opisy pracy rejestrów liniowych i różnice w nich występujące	63
2.3. Czy można zastąpić graf przejść inną techniką określania stanów wewnętrznych rejestrów liniowych?	77
2.4. Rejestry liniowe przechowujące dowolne rozdzielne kody liniowe	93
3. PROJEKTOWANIE REJESTRÓW LINIOWYCH	101
3.1. Projektowanie rejestrów liniowych o różnych strukturach sprzężenia liniowego	101
3.2. Jak uzyskać minimalną liczbę bramek XOR w sprzężeniu liniowym?	118
3.3. Projektowanie rejestrów o strukturze CADT	136
3.4. Łatwo modyfikowalne rejestry liniowe	149
3.5. Projektowanie rejestrów COPMISR	156

4. KOMPAKCJA LINIOWA I JEJ PROBLEMY	162
4.1. Kompakcja liniowa oraz weryfikacja sygnatur	162
4.2. Skuteczność wykrywania błędów za pomocą analizy sygnaturowej	165
4.3. Zwiększanie skuteczności analizy sygnaturowej	191
5. PODSUMOWANIE	227
Bibliografia	237
Streszczenie	255

Contents

More important abbreviations	7
More important symbols	9
1. INTRODUCTION	13
1.1. The Role of Built-in Circuit Testers BIST in Resolving Computer Systems Testing Problems	15
1.2. The Meaning of Linear Registers in Testing Simplification	21
1.3. The Structures of Built-in Circuit Testers with Linear Registers	26
1.4. Some Problems Connected with Choice, Design and Application of Linear Registers in Digital Circuits Testing	50
2. THE STRUCTURES AND ALGEBRAIC DESCRIPTION OF LINEAR REGISTERS	55
2.1. Different Structures of Linear Registers Type MISR	56
2.2. Algebraic Description of the Operation of Linear Registers and their Differences	63
2.3. Can the State Transition Graph be Replaced by other Techniques for Linear Register Internal State Determination?	77
2.4. Linear Registers for Preserving Arbitrary Linear Codes	93
3. THE DESIGNING OF LINEAR REGISTERS	101
3.1. The Designing of Linear Registers with Different Structures of Linear Feedback	101
3.2. How to Obtain a Minimal Number of XOR Gates in Linear Feedback?	118
3.3. The Designing of Linear Registers with Structure Type CADT	136
3.4. Easy Modifiable Linear Registers	149
3.5. The Designing of Registers Type COPMISR	156

4. LINEAR COMPACTION AND ITS PROBLEMS	162
4.1. Linear Compaction and Signature Verification	162
4.2. The Effectiveness of Error Detecting by Means of Signature Analysis	165
4.3. Increasing Signature Analysis Effectiveness	191
5. SUMMARY	227
Bibliography	237
Abstract	257

Ważniejsze skróty

- BILBO - (Built-In Logic Block Observer) wbudowany w układ cyfrowy wielowejściowy rejestr przesuwający ze sprzężeniem liniowym
- BIMLON - (Binary Implemented Multi-Valued Logic Network) wielowartościowy układ logiczny zaimplementowany w technologii binarnej
- BIST - (Built-In Self-Test) tester wewnątrzukładowy
- BIKEV - (Built-In Evaluator Using Multiple (k) Compaction) wbudowany w układ cyfrowy weryfikator do analizy k-sygnaturowej
- BT - (Bottom-Top) struktura wewnętrzno-zewnętrzna sprzężenia liniowego
- CA - (Cellular Automata) struktura sprzężenia liniowego bazująca na jednowymiarowych hybrydowych automatach komórkowych
- CBILBO - (Concurrent BILBO) rejestr BILBO umożliwiający testowanie na bieżąco
- CLB - (Configurable Logic Block) konfigurowalny blok logiczny
- COPMISR - (COde Preserving MISR) wielowejściowy rejestr liniowy przechowujący kody liniowe
- CSTP - (Circular STP) pierścień testujący
- CCSTP - (Condensed CSTP) skondensowany pierścień testujący
- c-LFSR, c-SISR, c-MISR - rejestry liniowe o różnych strukturach sprzężeń liniowych $c \in \{IED, EED, TBD, BTD, IEDT, EEDT, BTDT, TBDT, DT, CADT, \dots\}$ (szczegółowe wyjaśnienia na str. 59 w rozdziale 2.1)
- DED-SEC - (Double Error Detection-Single Error Correction) wykrywanie błędów podwójnych oraz korekcja błędów pojedynczych
- DFT - (Design For Testability) projektowanie ułatwiające testowanie
- EE - (External Exclusive OR) struktura zewnętrzna sprzężenia liniowego
- FSR - (Feedback Shift Register) rejestr przesuwający ze sprzężeniem liniowym bez bramek XOR
- HP - wielomian charakterystyczny związany z rejestrami liniowymi używanymi przez firmę Hewlett Packard w analizatorach sygnatur
- IE - (Internal Exclusive OR) struktura wewnętrzna sprzężenia liniowego
- IOR - (Inclusive OR) bramka logiczna realizująca funkcję $a \odot b = ab + \overline{ab}$
- ISCAS - nazwa wzorcowych układów logicznych rozpowszechnionych na konferencjach ISCAS (IEEE Symp. on Circuits and Systems)

- KL - kompaktor liniowy
- k LFSR - rejestr LFSR do analizy k-sygnaturowej
- k MISR - rejestr MISR do analizy k-sygnaturowej
- kSA - (k-signature analysis) k-sygnaturowa analiza
- LFSR - (Linear Feedback Shift Register) autonomiczny rejestr liniowy
- LFSR-p(x), SISR-p(x), MISR-p(x) - rejestry liniowe związane tym samym wielomianem charakterystycznym p(x)
- LOCST - (Level Sensitive Scan Design On-Chip Self Test) tester wewnątrzukładowy typu LOCST
- MBILBO - (Modified BILBO) rejestr BILBO z modyfikacją
- MIShR - (Multi-Input Shift Register) wielowejsciowy rejestr przesuwający
- MISR - (Multi-Input Signature Register) wielowejsciowy rejestr liniowy
- MR - (Memory Register) blok pamięci
- MISR-MR - (MISR based Memory Register) blok pamięci zaprojektowany na bazie rejestru MISR
- PMISA - (Parity Code Preserving Multi-Input Signature Analyzer) wielowejsciowy rejestr liniowy przechowujący słowa z bitem parzystości
- TB - (Top-Bottom) struktura zewnętrzno-wewnętrzna sprzężenia liniowego
- TRC - (Two Rail Checker) dwutorowy układ kontrolny
- TUC - Testowany Układ Cyfrowy
- TZ - Tester Zewnętrzny
- TW - Tester Wewnętrzny
- SA - (Signature Analysis) analiza sygnaturowa
- SISR - (Single-Input Signature Register) jednowejsciowy rejestr liniowy
- SP - (Scan Path) ścieżka sterująco-obsługowa
- STP - (Self Test Path) ścieżka testująca
- STUMPS - (Self-Test Using MISR and Parallel SRSG) tester wewnątrzukładowy typu STUMPS
- TB - (Top-Bottom) struktura zewnętrzno-wewnętrzna sprzężenia liniowego
- TISR - (Two-Input Signature Register) dwuwejsciowy rejestr liniowy
- UK - układ kombinacyjny
- US - układ sekwencyjny
- WPU - współczynnik pokrycia uszkodzeń
- XOR - (Exclusive OR) bramka logiczna realizująca funkcję $a \oplus b = a\bar{b} + \bar{a}b$

Ważniejsze symbole

- + - symbol używany w następujących różnych znaczeniach: dodawanie w ciele GF(2), dodawanie macierzy w ciele GF(2), symbol formalny w budowie wielomianów, dodawanie wielomianów
- $\sum_{i=1}^w \oplus$ - symbol używany w powyższym znaczeniu w przypadku dodawania wielu (w) elementów
- + - symbol używany także w sumowaniu: w arytmetyce dziesiętnej we wszystkich indeksach dolnych i górnych, w określaniu wymiarów wektorów i macierzy oraz w określaniu czasu, używany ponadto do oznaczania przyłączania kolejnych komórek i modułów wielokomórkowych rejestrów liniowych typu CA
- \sum - symbol używany w wieloargumentowym sumowaniu w arytmetyce dziesiętnej
- symbol używany w następujących różnych znaczeniach: mnożenie w ciele GF(2), mnożenie wielomianów i mnożenie w arytmetyce dziesiętnej
- $\prod_{i=1}^w$ - symbol używany w przypadku mnożenia wielu (w) wielomianów
- \oplus - operator binarny realizowany przez bramkę XOR
- \odot - operator binarny realizowany przez bramkę IOR
- A, C, D - macierze $[a_{r,s}]$, $[c_{r,s}]$, $[d_{r,s}]$ o wymiarach (n,k), w których $a_{r,s}, c_{r,s}, d_{r,s} \in \{0,1\}$ są elementami ciała GF(2), a indeksy $r = 1,2,\dots,n$ oraz $s = 1,2,\dots,k$
- b - prawdopodobieństwo wylosowania jedynki na dowolnej pozycji ciągu błędów E
- $\beta_K(x)$ - charakterystyka plasterkowego modułu K_n
- $\oplus D, \oplus T, \oplus D, \oplus T, (\oplus D), 0, 1, \mathbf{D}, \mathbf{T}, \mathbf{0}, \mathbf{1}$ - symbole używane w uproszczonym zapisie struktur rejestrów liniowych (szczegółowe wyjaśnienia na str. 61 w rozdziale 2.1.1)

- E_i - ciąg błędów na i -tym wyjściu niesprawnego TUC
- $E(n,t)$ - paczka błędów, która zawiera $t \leq n$ błędów w odcinku ciągu błędów E nie dłuższym niż n
- $\mathbf{E}_f = \mathbf{U}_0 \oplus \mathbf{U}_f$ - macierz błędów na wyjściach uszkodzonego TUC
- $e_{cf}(x)$ - wielomian błędów ilustrujący macierz błędów \mathbf{E}_f skompresowaną w rejestrze c-MISR lub c-SISR
- $\mathbf{E}(n,t)$ - paczka błędnych wektorów w macierzy błędów \mathbf{E} , która zawiera $t \leq n$ identycznych wektorów błędów w odcinku macierzy \mathbf{E} nie dłuższym niż n
- $EF = 1 - P_{al}$ - teoretyczna skuteczność kompaktacji liniowej
- G - macierz generująca kod liniowy
- $GF(2)$ - dwuelementowe ciało Galois (Galois Field)
- k_r - współczynnik określający przerzutnik D ($k_r = 0$) lub T ($k_r = 1$)
- $K_n = k_0 k_1 k_2 \dots k_j \dots k_{n-1}$ - ciąg opisujący schemat rejestru liniowego typu CA lub n -bitowego plasterkowego modułu typu CA
- p_i - współczynnik określający obecność ($p_i = 1$) lub brak ($p_i = 0$) linii w sprzężeniu rejestru liniowego
- $p_c(x)$ - wielomian charakterystyczny związany z rejestrami liniowymi o strukturze c sprzężenia liniowego
- $p_{cr}(x)$ - wielomian związany z r -tym wejściem rejestru liniowego o strukturze c sprzężenia liniowego
- $p_{cw}(x)$ - strukturalna postać wielomianu charakterystycznego związanego z rejestrem liniowym o strukturze c sprzężenia liniowego
- $p^+(x)$ - wielomian odwrotny do wielomianu $p(x)$
- $\mathbf{P}_c(x)$ - macierz struktury c sprzężenia liniowego
- P_{al} - prawdopodobieństwo maskowania błędów \mathbf{E}
- $R[u(x), p(x)]$ - reszta $r(x)$ z dzielenia $u(x)$ przez $p(x)$
- r_i - i -ty bit reszty $r(x)$
- $s(x)$ - wielomian określający sygnaturę rejestru liniowego
- s_i - i -ty bit sygnatury $s(x)$

- S, N, U - cechy ciągu K_n
- s -z- x - uszkodzenie typu sklejenie z $x \in \{0,1\}$
- $T_K = (\beta_K(S), \beta_K(N), \beta_K(U))$ - typ modułu K
- U_i - ciąg danych diagnostycznych na i -tym wyjściu TUC
- \mathbf{U}_0 - macierz odpowiedzi (wyjściowa) sprawnego TUC
- \mathbf{U}_f - macierz odpowiedzi TUC z uszkodzeniem f
- $u_c(x), w_c(x), r_c(x), h_c(x), s_c(x)$ - wielomiany związane z algebraicznym opisem pracy rejestrów liniowych o strukturze c sprzężenia liniowego

1. WPROWADZENIE

Przemysł elektroniczny zmienił się zupełnie w ostatnim kwartale dwudziestego wieku. Produkty elektroniczne są znacznie bardziej złożone, o wiele mniejsze i nadzwyczaj wydajne, np. mikroprocesor Pentium Pro pracuje z szybkością 200 MHz i posiada 387 wyprowadzeń [Yu96]. Narzędzia do automatycznego projektowania przeniosły proces projektowania z deski kreślarskiej na ekran monitora, linie produkcyjne są wysoce zautomatyzowane, a klienci oczekują i praktycznie otrzymują produkty o najwyższej jakości.

Technologia testowania rozwinęła się gwałtownie, aby podtrzymać te zmiany. W 1969 r. testowanie było ręcznym procesem wykorzystującym warsztatowe urządzenia pomiarowe. Dzisiaj zarówno testowanie, jak również przygotowanie testowania są wysoce zautomatyzowane. Wyłonienie się projektowania ułatwiającego testowanie - DFT (Design For Testability) jako osobnej dyscypliny inżynierskiej [WillP82, Benne84, AbraB90, Hławi93b] stało się faktem o dużym znaczeniu.

DFT narodziło się na pierwszej konferencji ITC (International Test Conference) w 1969 r. Najpierw inżynierowie ds. testowania w czołowych firmach produkujących komputery zauważyli, że pewne właściwości projektu (np. brak globalnego ustawiania elementów pamięci lub obecność uniwibratorów, nie bramkowanych sygnałów zegarowych i długich łańcuchów liczników binarnych [GórkH77, Benne84]) permanentnie zwiększały koszt zarówno opracowywania testów, jak również ich stosowania. Później ci sami inżynierowie zaczęli uzyskiwać projekty urządzeń cyfrowych, dla których stało się niemożliwe generowanie wystarczająco dokładnych testów w ramach dostępnego budżetu lub w okresie poprzedzającym rozpoczęcie produkcji. Niewystarczalność takich testów oznaczała zwiększenie prawdopodobieństwa przekazania na rynek niesprawnych produktów.

Sytuacja, w której projektanci - nie zdając sobie sprawy z potrzeb testowania - przekazywali kompletne projekty do bezsilnego w takim przypadku zespołu odpowiedzialnego za testowanie, tworzyła w ramach jednej organizacji barierę pomiędzy projektowaniem a testowaniem.

waniem [Parke86, Hławi90b, Hławi90c, Hławi93b, Maund95, Hławi96a]. Sytuacja ta musiała być zmieniona. W naszym kraju sygnalizowano już ten problem w roku 1976 na międzynarodowej konferencji pt. "Diagnostyka i niezawodność systemów cyfrowych" [GórkH76, GórkH77, HławG77]. Opierając się na swoim zrozumieniu problemu testowania projektanci zaczęli starać się eliminować potencjalne trudności, natomiast inżynierowie ds. testowania tworzyli listę wskazówek do projektowania [GórkH77, Benne84], które przekazywali projektantom w celu uwzględnienia ich w swoich projektach. W ten sposób rozpoczął się proces przełamywania barier pomiędzy projektowaniem a testowaniem.

Techniki projektowania ułatwiającego testowanie przeszły długą drogę ewolucji od początkowych wskazówek. Tak zwane techniki strukturalnego projektowania ułatwiającego testowanie, takie jak ścieżka sterująco-obszernyjna (scan path) [WillP82, Benne84, Sapie87, AbraB90, Hławi93b], brzegowa ścieżka sterująco-obszernyjna (boundary scan path) [Beenk87, Goeri87, MaunB87, MaunT89, Hławi87b, HławB87, BaduH88a, IEEE90, Hławi89d, Hławi90c, HławK92b, Hławi93b,] oraz samotestowanie (self-test) [McClu85a, McClu85b, HławB87, HławN89, Kraśn89, AbraB90, Badur92, AgraK93a, AgraK93b, SaviB93, Hławi93b] są obecnie wykorzystywane przez czołowe firmy produkujące sprzęt elektroniczny. Większość najnowszych mikroprocesorów posiada właściwości samotestowania [Yu96], natomiast obecne narzędzia syntezy np. COMPASS mogą generować projekty układów scalonych ze ścieżką sterująco-obszernyjną już na poziomie funkcjonalnego opisu sprzętu. Środki sprzętowe i programowe do projektowania i produkowania łatwo testowalnych układów scalonych oraz pakietów są już dzisiaj dostępne w handlu. Szeroko je opisano w [Hławi93b]. Bazują one głównie na standardzie IEEE 1149.1. Są dowodem, że wiele firm zrozumiało już, że zyski płynące z ich zastosowania są znaczące w porównaniu ze stratami, jakie można ponieść wskutek ignorowania problemów ułatwiania testowania. Zatwierdzony w 1990 roku standard IEEE 1149.1 stał się dzisiaj wspólną furtką dla stosowania takich nadrzędnych procedur diagnostycznych, jak: testowanie wewnętrzne, emulacja mikroprocesorów, samotestowanie, rejestracja uszkodzeń (ang. fault logging), analiza układów za pomocą analizatorów logicznych, konfigurowanie systemów cyfrowych, uruchamianie programów, testowanie układów analogowych, diagnostyka serwisowa. Niedawno opracowano także projekt standardu P 1149.4 [P1149.4] umożliwiający ułatwienie testowania układów analogowych oraz układów analogowo-cyfrowych. Projekt ten rozszerza ideę magistrali IEEE 1149.1 o komórki ABC (ang. Analog Boundary

Cell) i dodatkowe linie związane z testowaniem układów analogowych. W efekcie powstała idea brzegowej ścieżki sterująco-obszernyjnej dla układów cyfrowo-analogowych i analogowych. Opracowano także standard IEEE 1149.5 pozwalający upraszczać testowanie urządzeń cyfrowych złożonych z wielu pakietów cyfrowych. Wkrótce pojawią się jeszcze inne standardy, np. związane z testowaniem na poziomie sieci łączącej wiele systemów cyfrowych [Hławi93b].

1.1. Rola testerów wewnętrznych BIST w rozwiązywaniu problemów testowania systemów komputerowych

Kompleksowe testowanie współczesnych mikroprocesorów stało się ogromnym przedsięwzięciem. Chociaż kapitał związany z testowaniem mikroprocesorów jest jeszcze mniejszy od kapitału związanego z testowaniem płytek krzemowych (wafer testing), to jego przyrost wyprzedza przewidywania [Yu96]. Dlaczego? Po pierwsze tester zewnętrzny jest droższy z powodu zwiększonej częstotliwości pracy dzisiejszych mikroprocesorów oraz dużej liczby ich wyprowadzeń. Po drugie testery zewnętrzne, które poprzednio kosztowały 50 tysięcy dolarów, dzisiaj kosztują dobrze ponad 5 milionów dolarów [Yu96]. Wreszcie z powodu złożoności struktur krzemowych oraz wzrastających wymagań jakościowych czas testowania coraz bardziej się wydłuża. W efekcie całkowity kapitał poświęcony testowaniu raptownie wzrasta [Yu96].

Obecnym trendem w projektowaniu ułatwiającym testowanie jest przenoszenie funkcji testowania z testerów zewnętrznych do wnętrza testowanych układów. Dotyczy to przede wszystkim układów cyfrowych, aczkolwiek ostatnio zaczęły się pojawiać próby wprowadzania takiego podejścia także dla układów analogowych i mieszanych [KhalK95]. Testowanie zewnętrzne, charakterystyczne dla lat 60 i 70, w którym układy były testowane wyłącznie przez zewnętrzne urządzenia testujące [Hławi93b], zmienia się na techniki wewnętrzne (typu "built-in") [Maund95, Hławi93b], w których w pełni zbudowany układ jest zdolny sam zweryfikować swoje własne funkcjonowanie. Przyjęło się nazywać testery wewnętrzne lub wewnątrzpakietowe jako BIST (Built-In Self-Test) [McClu85a, McClu85b, AbraB90, AgraK93a, AgraK93b, SaviB93, Hławi93b, MurrH96, KonBJ96, Yu96].

W przybliżeniu około 5% całkowitej liczby tranzystorów mikroprocesora Pentium Pro związanych jest z układami testera wewnątrzukładowego [Yu96]. Korporacja INTEL przewiduje, że w roku 2000 liczba tranzystorów przeznaczonych dla układów wspomagających samotestowanie wyniesie około 3 milionów [Yu96]. Jest to około 6% całkowitej liczby 50 milionów tranzystorów, które będą zawarte w jednej strukturze krzemowej. Ten procent - jak przewiduje Yu z korporacji INTEL - może wzrosnąć w roku 2006 [Yu96].

BIST posiada całkowitą przewagę nad testowaniem zewnętrznym, szczególnie wtedy gdy jest dostępny poprzez diagnostyczny interfejs np. IEEE 1149.1 na poziomie układu scalonego [IEEE90, Hławi89d, Hławi90c, HławiK92b, Hławi93b]. Po pierwsze tester wewnątrzukładowy jest w pełni kompatybilny z testowanym układem cyfrowym, tak więc dzisiejsze bardzo złożone pod względem funkcjonalnym i szybkie produkty nie muszą być testowane za pomocą testerów zbudowanych wcześniej przy użyciu wolniejszych technologii [Maund95, Hławi96a, KonBJ96, MurrH96]. Po drugie, podczas gdy liczba obwodów w typowym produkcie wzrastała wykładniczo, to liczba wyprowadzeń wejściowo/wyjściowych struktur (chipów) i pakietów nie wzrastała raptownie, ale liniowo [Hławi93b, Hławi96a, MurrH96, KonBJ96]. Ponieważ liczba wektorów testowych (testów) wymagana do efektywnego testowania wzrasta liniowo wraz ze złożonością testowanego układu, dlatego większe zbiory danych testowych muszą być przekazywane [Kraśn89, MurrH96, KonBJ96] przez zbyt mały interfejs (mała liczba we/wy). Ta "wąska szyjka" jest omijana przez umieszczenie funkcji testujących wewnątrz produktu. Zauważmy ponadto, że BIST umieszczone jest wewnątrz produktu na cały jego okres użytkowania (całe jego "życie"). Raz zweryfikowane przez projektanta funkcje BIST są dostępne w fazie testowania płytki krzemowej (wafer testing) jeszcze przed pocięciem jej na struktury (chip) oraz podczas testowania po zainstalowaniu struktury w obudowie [KonBJ96]. Są one dostępne nie tylko do przetestowania nowo wyprodukowanego układu scalonego, lecz także podczas sprawdzania produktu po jego zainstalowaniu, w czasie diagnostyki podczas normalnej pracy produktu (testowanie na bieżąco), w czasie serwisu, a także w ośrodku napraw [Maund95, Hławi96a, KonBJ96]. Tak więc różne rodzaje defektów wprowadzane na każdym z wymienionych etapów mogą być w prosty sposób wykrywane za pomocą tego samego testera wewnątrzukładowego [KonBJ96].

W przeszłości podejście do tych różnych faz testowania było niezależne [Hławi93b]. Różne strategie testowania były opracowywane - częściowo, aby sprostać różnym wyłania-

jącym się na każdym etapie życia produktu wyzwaniom, ale znacznie częściej dlatego, że za każdą fazę testowania były odpowiedzialne różne części tej samej organizacji lub nawet całkowicie inne organizacje [Parke86, Maund95, Hławi96a].

Rozpowszechnionym problemem jest liczba nieszkodzonych produktów wracających do naprawy jako rezultat rachitycznej diagnostyki, nieodpowiedniej dokumentacji i innych przyczyn, np. złych styków złącza TU/TZ (testowany układ - tester zewnętrzny). Często są cytowane wykresy ilustrujące, że 40% całej produkcji wraca do naprawy. Koszt niepotrzebnego krążenia tej części produkcji w pętli "produkcja - naprawa" jest poważny [Hławi93b, Maund95, Hławi96a].

Ponieważ produkty stają się coraz bardziej skomplikowane w użytkowaniu, zmienia się natura wezwań klientów o pomoc. Dla prostych produktów, takich jak np. telefon, użytkowanie jest proste i użytkownicy czynią niewiele pomyłek. Lecz dla złożonych produktów wiele zbędnych wezwań o pomoc będzie spowodowanych błędami użytkownika. W efekcie rezultatem będzie wielki nadmiar wizyt pracowników serwisu [Maund95, Hławi96a].

BIST w porównaniu z innymi strategiami testowania radzi sobie z tymi problemami. Pewna firma opublikowała informacje, że użycie BIST i brzegowej ścieżki sterująco-obszernyjnej jako bazy dla strategii testowania systemów cyfrowych może zredukować o 70 procent wielkość wymaganego na poziomie systemowym diagnostycznego oprogramowania i jeszcze zwiększyć diagnostyczną wydajność. Ponadto BIST może być łatwo wywoływany zdalnie, np. poprzez linię telefoniczną pozwalając na wyczerpującą diagnozę opisanych problemów zanim zostanie zaaranżowana wizyta inżyniera (nie można przesłać drogą telefoniczną urządzeń testujących) [Maund95, Hławi96a]. Nie jest ważne wtedy, że klient jest na jednej półkuli, a serwis na drugiej. Efektem takiej zdalnej diagnostyki jest szybkie uzyskanie przez serwis informacji o rodzaju uszkodzonego układu i możliwość przesłania pocztą sprawnego układu scalonego lub wskazanie najbliższego miejsca jego nabycia.

Jeżeli wszystko jest takie proste, to dlaczego nasze popularne komputery PC nie są łatwo testowalne. Przecież wiele z nich posiada już procesory wyposażone w brzegową ścieżkę sterująco-obszernyjną. Dlaczego ścieżki te nie są wykorzystywane przez służby serwisowe? Odpowiedź leży w rachunku ekonomicznym. Komponenty popularnego komputera PC są tak tanie, że nie opłaca się ich naprawiać. Pracownik serwisu przywozi walizkę pełną sprawnych kart i wymienia je w niesprawnym komputerze. Również przestój

takiego osobistego komputera nie spowoduje żadnej katastrofy ani upadku finansowego firmy. Inaczej będzie w przypadku firmy użytkującej sieć kilkudziesięciu komputerów PC. W celu zredukowania kosztów serwisu, które przekraczają obecnie wartość sprzętu, producenci będą musieli produkować łatwo testowalne komputery sieciowe, które aczkolwiek droższe już po kilku latach zwrócą z nadwyżką poniesione dodatkowe koszty zakupu. Tak więc niezwykle istotne są aspekty ekonomiczne testowania. Decydują one o tym, czy opłaca się myśleć o testowaniu już w trakcie projektowania struktury, pakietu cyfrowego (analogowego lub mieszanego) lub systemu cyfrowego (mieszanego) [Kraśn89, Hławi93b]. Projektowanie ułatwiające testowanie wymaga niewątpliwie znacznych nakładów finansowych. Jednakże odpowiednie jego wykorzystanie może zwrócić te koszty i przynieść dodatkowe korzyści we wszystkich fazach "życia" produktu.

Podobnie ma się rzecz z systemami komputerowymi stosowanymi od lat w wojsku, przemyśle, czy też lotnictwie, w których konsekwencje ewentualnej błędnej usługi komputera mogą mieć katastrofalny efekt ekonomiczny lub mogą spowodować utratę życia. W celu uniknięcia takich katastrofalnych skutków systemy te projektowane są tak, ażeby uzyskać ich wysoką wiarygodność (dependability) [Sapie87, Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c]. Oznacza to innymi słowy, że systemowi tak zaprojektowanemu będziemy mieli uzasadnione prawo zaufać, że będzie pełnił wymagane usługi, zawsze wtedy, gdy potrzeba. Są to systemy o wysokiej dyspozycyjności [Sapie87, Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c] (gotowości do użycia - ang. available) lub wysokiej niezawodności (reliable) inaczej ciągłości usług [Sapie87, Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c]. Są to systemy, które ze względu na uniknięcie katastrofalnych konsekwencji dla środowiska powinny gwarantować bezpieczne usługi (safe) [Sapie87, Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c] lub ze względu na zapobiegnięcie nieupoważnionemu dostępowi danych i/lub manipulowania danymi powinny gwarantować zabezpieczone usługi (secure) [Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c]. Wszystkie te wymagania można dzisiaj spełnić poprzez uwzględnienie ich w projekcie wiarygodnego systemu komputerowego. Oczywiście związane jest to z dodatkowymi nakładami, które ponoszone są z przyczyn czysto ekonomicznych w celu uniknięcia strat finansowych związanych z katastrofalnymi skutkami błędnych usług komputera w statkach kosmicznych (space shuttle), w systemach sterowania lotami samolotów, w szpitalnych systemach monitorowania pacjentów, w komputerowych systemach bankowych czy też wreszcie w systemach

sterowania sieciami energetycznymi. Już dzisiaj niektóre firmy, spośród tych, które świadczą usługi całą dobę, odwracają się od zawodnych stacji roboczych, czyli komputerów PC. Wracają do zlekceważonych dużych komputerów (mainframe'ów) obsługujących dawniej całe przedsiębiorstwo. Kupują wysoce wiarygodne i przez to kosztowne serwery pełniące najistotniejsze w firmie funkcje. Użytkownicy w dalszym ciągu w takich firmach pracują przy swoich tanich komputerach PC. Ich rola zostaje jednak ograniczona np. do wyświetlania danych pobieranych z serwera. Zepsucie się takiej stacji roboczej nie ma wpływu na żywotne interesy firmy, ponieważ są one zabezpieczane przez użyte w serwerze nadmiarowe środki sprzętowe i programowe do neutralizacji i tolerowania uszkodzeń. Polepszają one z jednej strony niezawodność serwera, a z drugiej strony ułatwiają proces testowania i lokalizacji ewentualnego uszkodzenia, umożliwiając tym samym jego szybkie usunięcie.

Istnieją dwie drogi polepszania wiarygodności sprzętu systemu komputerowego. Pierwsza z nich to wykorzystywanie wiarygodnych elementów najlepiej całkowicie sprawnych. Druga z nich to wprowadzanie do projektu tych elementów środków umożliwiających neutralizowanie i tolerowanie uszkodzeń [Sapie87, Lapri91, SiewS92, Pradh95, Krawc95, Hławi96c]. Poziom defektów (defect level) dzisiejszych złożonych układów VLSI mieści się w granicach 50-200 ppm (defective parts per million) [Kraśn96, Yu96]. Już wkrótce nie wystarczy on, aby zaspokoić wymagania niezawodnościowe projektowanych systemów komputerowych. Pod koniec lat 90 należy oczekiwać, że pożądaną poziom defektów układów scalonych będzie rzędu 1 ppm [WangM95, Kraśn96]. Ze względu na to, że nie należy się spodziewać w ciągu najbliższych kilku lat radykalnej poprawy jakości procesu produkcyjnego układów scalonych VLSI, pożądaną obniżenie poziomu defektów będzie możliwe jedynie na drodze wprowadzania testowania o wysokiej klasie [Kraśn96]. Obecnie taką techniką jest przede wszystkim BIST umożliwiający realizację samotestowalnych komponentów. Samotestowanie jest bardzo pożądaną cechą komponentów systemów komputerowych neutralizujących i tolerujących uszkodzenia. Umieszczenie w takich systemach samotestowalnych struktur krzemowych może znacznie ułatwić wprowadzanie cech samotestowalności na wyższych poziomach integracji, np. w wielostrukturalnych modułach, pakietach cyfrowych itp. Ma to istotne znaczenie dla testowania okresowego (periodic maintenance) całego systemu lub jego komponentów [SiewS92, Pradh95, Kraśn96]. Czyni to także bardziej efektywnym mechanizm uzdrawiania systemu, czyli

powrotu do normalnego stanu (recovery), np. przy identyfikacji, czy błąd był spowodowany przez uszkodzenie trwałe czy też uszkodzenie przejściowe albo też w przypadku diagnostyki, tzn. identyfikacji najmniejszego wymiennalnego uszkodzonego komponentu systemu [SiewS92, Pradh95, Kraśn96].

Techniki BIST są głównie wykorzystywane w tzw. testowaniu "off-line" wykonywanym wtedy, kiedy testowany system, pakiet czy też układ scalony nie realizuje swoich normalnych funkcji (np. testowanie produkcyjne, serwisowe, okresowe). W testowaniu takim błędy nie są wykrywane na bieżąco, ale na końcu procesu testowania. Natychmiastowe wykrywanie błędów w czasie normalnej pracy systemu (bez potrzeby jej przerywania w celu testowania systemu), inaczej testowanie "on-line" - tak istotne w przypadku neutralizacji błędów w wysoce wiarygodnych systemach komputerowych - realizowane jest za pomocą układów samokontrolujących się (self-checking circuits) [Sapie87, RaoF89, SiewS92, Pradh95, Piest95]. Układy te są normalnymi podzespołami systemu komputerowego zaprojektowanymi jednak specjalną metodą wykorzystującą kody detekcyjne (Hamminga, liniowe rozdzielne, podwójne, parzystości, z bitami przystości grupowej, arytmetyczne, Bergera itp. [PetW94, RaoF89, SiewS92, Pradh95, Piest95]) służące do kodowania sygnałów wyjściowych w słowa kodowe. Wykrywanie na wyjściu błędnych (niekodowych) słów - to zasada kontroli realizowanej przez takie podzespoły. Ostatnio układy samokontrolujące się stosowane są równolegle wraz z techniką BIST, co umożliwia jednocześnie wykorzystywanie zalet obu technik zarówno w testowaniu "on-line", jak i testowaniu "off-line" [GupP92, GupP96]. Można też z tych samych powodów projektować BIST wraz z układem samokontrolującym się jako jeden podzespół, co umożliwi dodatkowo znaczne zaoszczędzenie środków sprzętowych niezbędnych do ich realizacji [SogG96, GoesS96, HłaGS96a, HłaGS96b, HłaGS97]. Przy tolerowaniu uszkodzeń istotne są między innymi techniki sprzętowego maskowania błędów wykorzystujące kody korekcyjne (kod Hsiao - zmodyfikowany kod Hamminga [RaoF89]). Ma to szczególne znaczenie w przypadku maskowania błędów pamięci RAM, gdzie szeroko stosowane są między innymi kody DED-SEC (Double Error Detection - Single Error Correction [RaoF89]). Bazujące na takich kodach układy korygujące błędy można projektować i umieszczać w pamięci jako jeden podzespół razem z układem samokontrolującym się połączonym z testerem wewnętrznym BIST [HłaGS97]. Już obecnie wprowadzane są do wnętrza projektów układów ASIC środki sprzętowe neutralizujące i tolerujące uszkodzenia oraz ułatwiające testowanie (BIST)

[HunTZ96], a na konferencjach dyskutowane są problemy, gdzie i jak struktury krzemowe neutralizujące i tolerujące uszkodzenia mogą lepiej sprostać dzisiejszym jakościowym wyzwaniom [SavTI96].

1.2. Znaczenie rejestrów liniowych w upraszczaniu testowania

Każdy tester zawiera trzy następujące główne bloki: generator testów, weryfikator odpowiedzi testowanego układu cyfrowego oraz układ sterujący testowaniem. W testowaniu zewnętrznym te trzy funkcje pełni najczęściej komputer wyposażony w duże pamięci masowe do rejestracji testów i odpowiedzi na nie. Rejestry przesuwające ze sprzężeniem liniowym - w skrócie rejestry liniowe - spełniły istotną rolę w przechodzeniu od testowania zewnętrznego do testowania wewnętrznego [Hławi93b]. W pierwszej fazie tego procesu na przełomie lat 70 i 80 zrezygnowano z kontroli poprawności na bieżąco każdego bitu w punktach pomiarowych testowanego układu cyfrowego (TUC). W miejsce krokowej, czasochłonnej i realizowanej w każdym taktie zegarowym kontroli poprawności każdego bitu wprowadzono weryfikację jednorazową na końcu pomiaru całej odpowiedzi. Taki dynamiczny pomiar odpowiedzi umożliwił analizator sygnatur [FrohW77, Hewle97, HławiK81b, Kubiś84, HławiN85b, Hławi89g, Hławi93b], który komprimuje długie ciągi bitów w krótkie słowa zwane sygnaturami. Sygnatura, niczym "odcisk palca", stała się identyfikatorem testowanego układu cyfrowego porównywanym z sygnaturą wzorcową. Podstawowym blokiem funkcjonalnym analizatora sygnatur jest jednowejściowy rejestr przesuwający z liniowym sprzężeniem zwrotnym SISR (ang. Single Input Signature Register) tzw. szeregowy kompaktor liniowy [FrohW77, HławiN85b, Hławi93b]. W niektórych rozwiązaniach może nim być wielowejściowy rejestr o sprzężeniu liniowym MISR (ang. Multi Input Signature Register) realizujący równoległą kompaktację odpowiedzi TUC i nazywany potocznie równoległym kompaktorem liniowym [Hławi87c, Hławi88a].

Skrócenie czasu pomiaru oraz kompaktacja danych, to podstawowe cechy charakterystyczne dla analizatorów sygnatur wykorzystywanych do zewnętrznego pomiaru odpowiedzi testowanego urządzenia cyfrowego [Hławi93b].

Redukcja czasu pomiaru pozwoliła na realizację w technice testowania funkcjonalnego (Functional Testing) testowania dynamicznego, umożliwiającego wyłapywanie błędów mają-

cych swoje źródło w dynamice testowanego układu cyfrowego (np. wyścigi). Podczas testowania wewnątrzobwodowego (In-Circuit Testing) [Hławi89f, Hławi93b] zredukowano dzięki temu czas izolowania (tzw. elektrycznego "wycinania") z otoczenia złożonych układów scalonych VLSI, takich jak np. mikroprocesory, pamięci RAM oraz ROM, co chroniło je przed degradacją niezawodności lub uszkodzeniem [Hławi89f, Hławi93b]. Wreszcie skrócenie czasu umożliwiło wprowadzenie testowania cyklicznego polegającego na kilkakrotnym powtarzaniu testu. Dzięki cyklicznym testom analizator sygnatur może wykrywać również błędy przemijające, krótkotrwałe, występujące np. tylko w pewnych określonych warunkach temperaturowych, przy pewnych wibracjach itp.

Kompakcja liniowa danych diagnostycznych mierzonych w wewnętrznych punktach układu cyfrowego znacznie skróciła i uprościła procedurę lokalizowania uszkodzeń w testowaniu funkcjonalnym [Hławi89f, Hławi93b] zarówno w metodzie śledzenia wstecz (ręczne sondowanie lub przy użyciu matrycy igłowej) [Hławi89f, Hławi93b], jak również w metodzie wykorzystującej słownik uszkodzeń [Hławi89f, Hławi93b]. Kompakcja liniowa odpowiedzi testowanego układu cyfrowego pozwoliła równocześnie zminiaturyzować analizator sygnatur. Dzięki wyeliminowaniu z testera pamięci masowych analizator sygnatur stał się przenośnym i podręcznym testerem. Kompakcja liniowa odpowiedzi umożliwiła także dołączenie heksadecymalnych danych diagnostycznych bezpośrednio do dokumentacji ideowej testowanego układu cyfrowego. Umożliwiło to sprawdzanie poprawności działania układów cyfrowych za pomocą techniki podobnej do techniki testowania telewizorów. Trzeba tu jednak wyraźnie zaznaczyć, że zastosowany w analizatorze sygnatur proces kompaktacji liniowej wskutek zamaskowania błędów wywołanych uszkodzeniami może czasami prowadzić do niepożądanych rezultatów testowania, np. przekazania do użytkownika niesprawnego urządzenia (układu scalonego) [Hławi93b] lub zlokalizowania niewłaściwego miejsca uszkodzenia [Hławi93b].

Pomimo to wymienione wcześniej zalety tej nowej metody testowania spowodowały gwałtowny rozwój analizy sygnaturowej uszkodzeń. W konsekwencji oprócz pierwszego analizatora sygnatur 5004 [Froh77] pojawiło się jeszcze kilka innych znacznie ułatwiających i zwiększających skuteczność wykrywania i lokalizowania uszkodzeń. Na przykład analizator sygnatur 5006A [Hewle97] skraca czas testowania dzięki sumowaniu sygnatur w jedną sygnaturę złożoną i wprowadzeniu pamięci sygnatur (funkcja RECALL). Inny analizator sygnatur PAS 80, opracowany w Instytucie Elektroniki Politechniki Śląskiej

[HławiN85b], gwarantuje większą skuteczność analizy sygnaturowej dzięki zastosowaniu nowej techniki kompaktacji liniowej ciągów trójstanowych danych wykorzystującej dwuwejściowy rejestr liniowy TISR (Two Input Signature Register) [Hławi86b] i dzięki dwusygnaturowej analizie wykorzystującej dwa różne sprzężenia rejestru liniowego użytego do kompaktacji danych. Pomimo tych wielu zalet testowanie za pomocą analizatorów sygnatur pierwszej generacji posiadało jeszcze szereg następujących niedostatków: konieczność opracowania i stosowania przez techników drogiej i kłopotliwej dokumentacji diagnostycznej w postaci tabel sygnatur i schematów ideowych z naniesionymi sygnaturami, konieczność żmudnego wnoszenia poprawek do dokumentacji w sytuacji, gdy do projektu układu cyfrowego wniesiono poprawki, konieczność weryfikowania przez operatora kolejnych sygnatur heksadecymalnych poprzez ich odczytywanie z dokumentacji i porównywanie z sygnaturami odczytanymi na wyświetlaczu analizatorów sygnatur, konieczność korelowania sygnatur z punktami pomiarowymi testowanego układu cyfrowego.

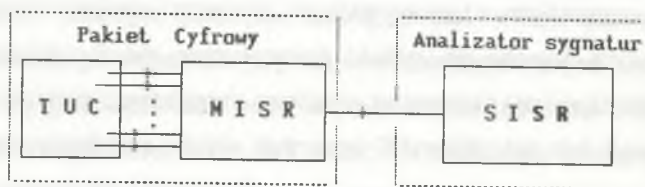
Analizator sygnatur drugiej generacji 1310A [Robin85] zwany weryfikatorem sygnatur oraz opracowany w Instytucie Elektroniki Politechniki Śląskiej weryfikator sygnatur WAS80 [Hławi88b] wyeliminował wszystkie ww. niedostatki. Weryfikator sygnatur wprowadził do analizy sygnaturowej, oprócz redukcji czasu i kompaktacji danych, także uproszczenie problemu przygotowania dokumentacji diagnostycznej [Hławi88b, Hławi93b] oraz uproszczenie procedury wykrywania i lokalizacji uszkodzeń [Hławi88b, Hławi93b].

W weryfikatorze sygnatur istnieje, oprócz kompaktacji ciągu w sygnaturę, również druga przyczyna maskowania błędów. Jest nią proces weryfikacji sygnatur. Właśnie ta druga przyczyna powoduje, że prawdopodobieństwo niewykrycia błędów uzyskiwane przy użyciu weryfikatora 1310A oraz WAS80 jest większe [HławiM86, Hławi88b] od prawdopodobieństwa uzyskanego dla analizatora HP 5004. Ten wzrost prawdopodobieństwa niewykrycia błędów został skompensowany za pomocą wydłużenia rejestru liniowego o dodatkowe przerzutniki [Robin85, HławiM86, Hławi88b].

Przenośne analizatory sygnatur oraz weryfikatory sygnatur dzięki swoim zaletom i możliwościom pozwoliły także na rozwiązanie szeregu problemów związanych z testowaniem serwisowym. Umożliwiły one wprowadzenie do serwisu nowej, prostej techniki lokalizacji uszkodzeń nie wymagającej żmudnej i czasochłonnej współpracy z komputerem. Pozwoliło to zdjęcie z najbardziej doświadczonych inżynierów na rzecz pracowników o niższym wykształceniu mniej ważne z żywotnego punktu widzenia fabryki działania, takie

jak serwisowa naprawa i diagnostyka [Hławi89g, Hławi93b]. To z kolei umożliwiło zlikwidowanie opóźnień związanych z transportem dobrych i uszkodzonych pakietów cyfrowych między użytkownikiem a służbami serwisowymi i stanowiskiem do testowania pakietów w fabryce [Hławi89g, Hławi93b]. W efekcie zlikwidowano także duże i kosztowne zapasy pakietów w fabryce [Hławi89g, Hławi93b].

Zaproponowano także możliwość realizacji tzw. równoległej analizy sygnaturowej, w której jednocześnie wiele punktów testowanego układu cyfrowego jest podłączonych do wielowejściowego rejestru liniowego MISR [Hławi93b]. Taka zilustrowana na rys. 1.1 technika analizy sygnaturowej wielokrotnie przyspiesza cały proces analizy sygnaturowej uszkodzeń, umożliwiając w ten sposób szybką detekcję niesprawnych pakietów cyfrowych. Do realizacji takiej równoległej analizy sygnaturowej można wykorzystać specjalne układy ASIC typu {45}, {ef} lub {45/ef} zaprojektowane w Instytucie Elektroniki Politechniki Śląskiej i Instytucie Technologii Elektronowej oraz wyprodukowane w Laboratorium INPG/TIM3 w Grenoble [Hławi93b]. Układy te wmontowane w pakiet cyfrowy i podłączone do TUC służą - podobnie jak komercyjne układy scalone firmy Logical Solution Technology (LST) [Turin 84, Turin 85] - głównie do realizacji sprzęgu P1149.3 [Hławi93b]. Ich dodatkową funkcją niewystępującą w układach firmy LST jest możliwość pracy jako rejestru MISR oraz możliwość łączenia ich ze sobą w szereg, co pozwala na skokowe wydłużanie rejestru MISR [Hławi93b] w zależności od liczby punktów obserwacyjnych TUC.



Rys. 1.1. Idea równoległej analizy sygnaturowej przy użyciu konwencjonalnego analizatora (weryfikatora) sygnatur

Fig. 1.1. The idea of parallel signature analysis using conventional signature analyzer

Wprowadzenie analizy sygnaturowej stało się wyzwaniem dla inżynierów do dalszego upraszczania testowania. Firma Hewlett-Packard wprowadziła na rynek stymulatory (exercises) typu 5001A, 5001B, 5001c i 5001D przeznaczone do analizy sygnaturowej

uszkodzeń w systemach z mikroprocesorami. Ich zadaniem było sterowanie pracą analizatorów sygnatur i generowanie z zewnątrz testów dla typowych makrobloków systemu mikroprocesorowego, takich jak pamięć ROM, pamięć RAM, czy też wreszcie sam mikroprocesor [Hławi82, Hławi83]. Dalszy rozwój techniki analizy sygnaturowej wprowadził na rynek emulatory sprzężone z analizatorami sygnatur [Hławi85]. W Instytucie Elektroniki powstał także tester układów scalonych SSI, MSI i LSI typu MASTER 86 [Mita88] z kompaktorem w postaci równoległego rejestru MISR. Wszystkie te metody testowania w znaczący sposób uprościły techniki testowania zewnętrznego, ale w dalszym ciągu nie przeniosły testowania z zewnątrz do wnętrza TUC.

Chcąc uchronić pracownika serwisu przed koniecznością zabierania ze sobą stymulatorów, zaczęto wbudowywać w pakiety systemów mikroprocesorowych cyfrowe środki sprzętowe ułatwiające generowanie testów (Built-In Test) i sterowanie analizatorami sygnatur [Kubiś84, Hławi93]. Przykładem krajowym takiego podejścia może być dostosowany do analizy sygnaturowej specjalizowany mikrokomputer do określania jakości żeliw CRISTALDIGRAF NC [Hławi81, Hławi83]. Mikrokomputer ten posiada dwa rodzaje wewnętrznej generacji testów tzw. swobodny obieg przestrzeni adresowej (free run), w którym rolę generatora testów pełni licznik adresów mikroprocesora (licznik binarny) oraz programowe generowanie testów zawartych w pamięci ROM. W ramach każdego z tych dwóch rodzajów testów generowane są równoległe sygnały sterujące pracą analizatora sygnatur. Całkowite przeniesienie funkcji testowania do wnętrza TUC stało się możliwe dzięki wbudowaniu uproszczonego analizatora sygnatur w pakiet cyfrowy mikrokomputera. W taki sposób została zaprojektowana zmodernizowana wersja mikrokomputera CRISTALDIGRAF NC-ST [Hławi89]. Niestety idea takiego samotestowania nie może być bezpośrednio przeniesiona do wnętrza każdego kombinacyjnego czy też sekwencyjnego układu cyfrowego. Powodem jest koszt sprzętu mikroprocesora, który w samotestowalnych mikrokomputerach jest jądrem sterującym testowaniem i wymuszającym sekwencje testów. Wprowadzenie do wnętrza TUC generatora testów w postaci licznika binarnego zmusza do realizacji tzw. testowania wyczerpującego (exhaustive), które dla liczby wejść pierwotnych $n > 30$ testowanego UK wydłuża czas testowania do nieakceptowalnej wartości. Skrócenie tego czasu drogą zmniejszenia liczby testów generowanych przez licznik binarny prowadzi z kolei do zmniejszenia współczynnika pokrycia uszkodzeń. Przyczyną są najstarsze bity licznika, które w znacznej części sekwencji testującej są stabilne. Rozwiązanie problemu tej

"krótkiej koldry" umożliwił autonomiczny rejestr liniowy LFSR generujący sekwencje pseudoprzypadkowych testów. Zaletą sekwencji testów pseudoprzypadkowych jest to, że są pewną "aproxymacją" sekwencji losowych [BardM87, AbraB90, SaviB93, Hławi93b] i generowane są przez rejestry LFSR w sposób cykliczny i bez powtarzania w cyklu identycznych wektorów testowych. Umożliwia to w sporej grupie zastosowań uzyskanie w rozsądnym czasie wysokiego współczynnika pokrycia uszkodzeń w TUC. Rejestr LFSR wbudowano więc w analizator sygnatur PAS 80 [HławiN85b], co pozwoliło na zewnętrzne testowanie wielu układów kombinacyjnych i niektórych układów sekwencyjnych. Nieco podobną ideę uproszczonego testowania zewnętrznego w postaci tzw. "random test socket" (RTS) wykorzystywaną dla testowania układów sekwencyjnych z blokiem pamięci ze ścieżką sterująco-obszerną zaproponowano wcześniej w [BardM82, BardM87, AbraB90]. Oba te urządzenia były jednym z ostatnich kroków w upraszczaniu testowania zewnętrznego. W następnym etapie upraszczania testowania układów kombinacyjnych oraz sekwencyjnych większość funkcji testera zewnętrznego typu RTS czy też PAS 80 zaczęto wprowadzać do wnętrza TUC.

1.3. Struktury testerów wewnątrzukładowych z rejestrami liniowymi

W testowaniu realizowanym przez tester wewnątrzukładowy (układ BIST) znaczna część funkcji testera zewnętrznego może być również wypełniana przez rejestry liniowe. Tak więc sekwencje testów pseudoprzypadkowych (Pseudorandom Pattern) mogą być generowane przez autonomiczne rejestry liniowe LFSR [PetW94, Golom82, BardM87, Hławi93b, Hławi96c, Hławi97a, Hławi97b]. Jednowejściowe rejestry liniowe SISR [PetW94, Frohw77, Hławi93b, Hławi97b] oraz wielowejściowe rejestry liniowe MISR [KoneM79, KoneM80, Hassa82, SridH82, HassL83, Hławi84c, Hławi85a, Hławi86c, Hławi93b, Hławi96b, Hławi96c, Hławi97a, Hławi97b] mogą pełnić rolę kompaktorów odpowiedzi testowanego przez BIST układu cyfrowego. Wreszcie zmodyfikowany n-bitowy rejestr liniowy umożliwiający zliczanie w zakresie do 2^n [SavaW95] i nazywany w [Hławi97a] licznikiem liniowym może pełnić istotną rolę w układzie sterującym testowaniem wewnątrzukładowym. Umożliwia on bowiem zliczanie liczby testów pseudoprzypadkowych genero-

wanych przez rejestr LFSR [SavaW95, Hławi97a]. Pozwala także odmierzać liczbę bitów stanów początkowych wprowadzanych szeregowo do generatora testów i kompaktora oraz umożliwia odmierzanie długości sygnatury wyprowadzanej z kompaktora [SavaW95, Hławi97a].

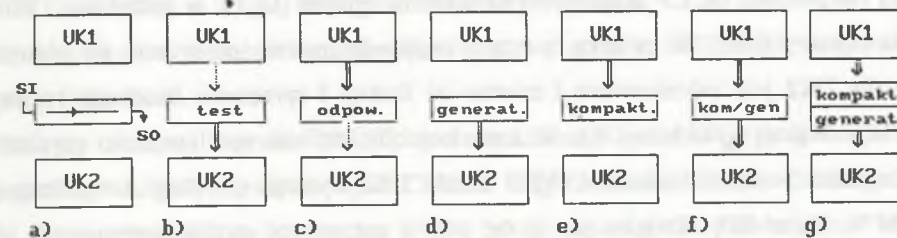
Do implementacji rejestrów liniowych w testerach wewnątrzukładowych potrzeba niedużych środków sprzętowych. W przypadku gdy w procesie projektowania DFT układ cyfrowy został wyposażony w ścieżkę sterująco-obszerną lub jej brzegowy odpowiednik, wówczas rejestry liniowe można wkomponować w takie ścieżki. Wtedy przyrost powierzchni struktury krzemowej związany jest właściwie jedynie z jedną lub kilkoma bramkami XOR niezbędnymi do realizacji sprzężeń liniowych rejestrów przesuwających. Wówczas też nie odczuwa się wpływu testera wewnątrzukładowego na degradację parametrów operacyjnych układu cyfrowego. W przypadkach wykluczających możliwość wykorzystania ścieżek sterująco-obszernych dochodzi tylko nadmiar układowy związany z przerzutnikami niezbędnymi do realizacji rejestrów liniowych. Ten nieduży nadmiar układowy powoduje, że również przyrost poboru mocy jest mały. Dzięki niemu możliwe jest także uzyskanie minimalnego odstępu czasu w cyklach zegarowych między podaniem dwóch kolejnych wektorów testowych, co zapewnia dużą szybkość testowania. Reasumując, należy podkreślić, że opisane koszty wprowadzania samotestowalności są bardzo małe w przypadku stosowania rejestrów liniowych. Stało się to jednym z głównych powodów ich chętnego wykorzystywania do realizacji podstawowych funkcji testerów wewnątrzukładowych.

1.3.1. Rejestry BILBO, MBILBO, CBILBO, MISR-MR oraz MIShR

Niektóre techniki wykorzystywania rejestrów liniowych w testowaniu wewnątrzukładowym można wyjaśnić za pomocą typowego dla strumieniowego przetwarzania danych układu dwóch układów kombinacyjnych UK1 oraz UK2 rozdzielonych wewnętrznym rejestrem równoległym MR (Memory Register), którego bieżące stany są niezależne od jego stanów poprzednich [Hławi96c]. Rejestr MR w czasie testowania można w tym przykładzie wykorzystywać zarówno do zapisywania odpowiedzi testowanego UK1, jak również do podawania testów na wejścia UK2. Można to zrealizować wzbogacając rejestr w dodatkowe środki sprzętowe umożliwiające jego następujące rodzaje pracy: normalny, w którym rejestr spełnia swoją systemową funkcję testowania, w którym rejestr wspomaga testowanie UK1

oraz UK2 oraz kilka funkcji pomocniczych, z których przesuwanie zawartości rejestru i jego zerowanie wydają się najistotniejsze. Na rys. 1.2. [Hławi96c] przedstawiono różne, związane z testowaniem, rodzaje pracy tego wielofunkcyjnego rejestru. Praca rejestru jako rejestru przesuwającego SR (Shift Register) umożliwia wprowadzanie z zewnątrz (np. z zewnętrznego rejestru LFSR) poprzez wejście SI (Scan In) nowego testu (rys. 1.2a) dla UK2 z jednoczesnym wyprowadzaniem na wyjściu SO (Scan Out) odpowiedzi UK1 na poprzedni test (np. do zewnętrznego kompaktora SISR). Wprowadzać i wyprowadzać należy także odpowiednio każdy stan początkowy oraz stan końcowy rejestru niezależnie od funkcji, jaką w danym momencie rejestr ten spełnia. Praca równoległa rejestru MR jest nie tylko jego systemową funkcją, ale także jedną z funkcji umożliwiających podawanie testów z zewnątrz. Wówczas rejestr ten najpierw zasila UK2 pamiętanym testem (rys. 1.2b), następnie z chwilą podania taktu zegarowego w miejsce testu zostaje wpisana odpowiedź testowanego UK1 (rys. 1.2c). W celu zlikwidowania konieczności podawania testów z zewnątrz rejestr można skonfigurować w autonomiczny rejestr liniowy LFSR. Umożliwia on generowanie (rys. 1.2d) w każdym taktie zegarowym nowego testu PR (Pseudo Random). Pozwala to na uzyskanie wysokiego współczynnika pokrycia uszkodzeń w UK2 po niedużej liczbie taktów zegarowych. Tam, gdzie jest to niemożliwe, do uzyskania w pożądanym czasie poszukuje się wśród różnych rejestrów liniowych LFSR takiego, który pozwala w najkrótszym możliwym do uzyskania czasie wygenerować wszystkie tzw. testy deterministyczne [BoubK93, LemGB94, LempG94, Helle95, Garbo96] wcześniej wyznaczone przez specjalne programy ATPG (Automatic Test Pattern Generator). Podobnie zamiast kolejno weryfikować na zewnątrz wszystkie odpowiedzi UK1, można realizować kompaktację na bieżąco wszystkich odpowiedzi UK1 w rejestrze MISR pełniącym rolę równoległego kompaktora liniowego (rys. 1.2e). Rejestr MISR przy odpowiednio dobranym sprzężeniu liniowym pozwala znacząco zmniejszyć problem maskowania błędów spowodowany wygenerowaniem tej samej sygnatury co w układzie nieuszkodzonym [Hławi93b]. Jeżeli bieżące stany (sygnatury) kompaktora MISR będą wykorzystywane takt po takcie jako testy PR dla UK2, wówczas proces kompaktacji wszystkich odpowiedzi UK1 oraz proces generacji testów dla UK2 może odbywać się jednocześnie w ramach jednej sesji (każda sesja testowania zaczyna się i kończy wprowadzeniem rejestru w stan przesuwania) testowania (rys. 1.2f). Wtedy jednak w celu uzyskania wysokiego współczynnika pokrycia uszkodzeń (WPU) w układzie UK2 należy wydłużyć czas trwania takiej sesji [Kim88]

(zwiększyć liczbę testów PR dla UK2). Ten sam efekt przy jednocześnie skróconym czasie trwania sesji testowania można uzyskać podwajając liczbę komórek pamięci rejestru (rys. 1.2g) i ustawiając jego pracę jako rejestru MISR (kompaktor dla UK1) i jednocześnie niezależnego rejestru LFSR (generator dla UK2).

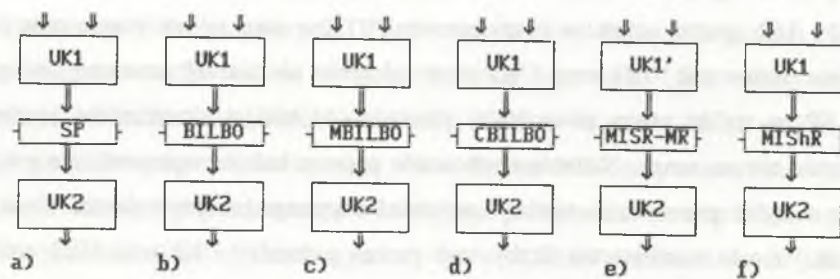


Rys. 1.2. Rodzaje pracy programowalnego rejestru: rejestr przesuwający (a), rejestr równoległy z patternem (b), rejestr równoległy z odpowiedzią (c), generator testów PR (d), równoległy kompaktor liniowy (e), równoległy kompaktor liniowy, którego bieżący stan wewnętrzny jest testem PR (f), równoległy kompaktor liniowy i niezależny generator testów (g)

Fig. 1.2. Programmable register operation types: shift register (a), parallel register with pattern (b), parallel register with response (c) PR test generator (d), parallel linear compactor (e), parallel linear compactor with current internal state which is PR test (f), parallel linear compactor and independent test generator (g)

Przykłady różnych wielofunkcyjnych rejestrów ułożonych pomiędzy UK1 oraz UK2 przedstawiono na rys. 1.3. Pierwszy z nich (rys. 1.3a) jest zwykłym rejestrem sterująco-obszaryjnym SP (Scan Path) [WillP82, Hławi96c] pracującym albo jako rejestr równoległy MR (praca normalna oraz testowanie), albo jako rejestr przesuwający SR. Jednoczesne testowanie UK1 oraz UK2 przy założeniu niezależnej autonomicznej pracy rejestru SP w trybie pracy równoległej prowadzi do niskiej skuteczności testowania i praktycznie nie ma sensu. Natomiast testowanie poprzez kolejne wprowadzanie i wyprowadzanie w trybie przesuwania testów i odpowiedzi wymaga przeprowadzenia wielu sesji testowania. W celu zmniejszenia liczby sesji można pomiędzy UK1 oraz UK2 umieścić rejestr BILBO (Built-In Logic Block Observer) opisany w [KoneM79, KoneM80, Hławi93b, Hławi96c]. Zilustrowano to nas rys. 1.3b. Przykładowe dwie komórki tego rejestru przedstawiono w [Hławi96c]. Ten wielofunkcyjny rejestr umożliwia ustawienie następujących rodzajów pracy: rejestr równoległy MR (praca normalna), rejestr przesuwający SR oraz kompaktor MISR, którego bieżące stany wewnętrzne są wykorzystywane jako testy PR dla UK2. Dodatkowym trybem pracy jest możliwość

równoległego zerowania komórek pamięci. Rejestr ten musi więc posiadać dwa wejścia sterujące. W procesie kompaktacji do rejestru MISR zapisywane są bieżące stany (sygnatury) będące wynikiem sumowania modulo dwa bieżącego wektora wyjściowego UK1 oraz stanu rejestru MISR z poprzedniego kroku. Testy dla UK2 są więc - modyfikowanymi przez wektory wyjściowe UK1 - stanami wewnętrznymi rejestru LFSR o sprzężeniu, które posiada rejestr MISR. W związku z tym prawdopodobieństwo pojawienia się różnych testów dla UK2 jest zróżnicowane i zależy od funkcji f sprzężenia liniowego rejestru MISR oraz funkcji f_M układu UK1. W konsekwencji określenie możliwego do uzyskania współczynnika pokrycia uszkodzeń WPU układu UK2 wymaga symulacji komputerowej [KoneM79, KoneM80]. Zwiększając liczbę taktów zegarowych można przypuszczać, że współczynnik pokrycia uszkodzeń WPU będzie wysoki [Kim88]. Kim i inni udowodnili, że stany (bieżące sygnatury) rejestru MISR mogą być traktowane jako testy PR dopiero po pewnej liczbie taktów zegarowych. Innymi słowy, bieżące sygnatury rejestru MISR będą wówczas zachowywały swój przypadkowy charakter niezależnie od tego, jakie wektory wyjściowe układu UK1 podawane są na wejścia rejestru MISR. W takim przypadku symulacja komputerowa przestaje być koniecznością. Poprzez dobór odpowiedniego sprzężenia liniowego rejestru MISR można próbować dążyć do uzyskania mniejszego zróżnicowania prawdopodobieństwa pojawienia się różnych testów. Pozwoliłoby to uzyskać wyższy współczynnik WPU układu UK2 przy niedużej liczbie taktów zegarowych.



Rys. 1.3. Testowanie przy użyciu programowalnego rejestru wewnątrzukładowego: SP (a), BILBO (b), MBILBO (c), CBILBO (d), MISR-MR (e), MIShR (f)

Fig. 1.3. Testing with the use of built-in circuit programmable register: SP (a), BILBO (b), MBILBO (c), CBILBO (d), MISR-MR (e), MIShR (f)

Pewną modyfikacją rejestru BILBO jest rejestr MBILBO (Modified BILBO) opisany w [McCIB81, WillP82, Wang87, Hławi96c]. Rejestr ten (rys. 1.3c) można w porównaniu

z rejestrem BILBO dodatkowo programować w rejestr LFSR. Wymaga to niestety dodania trzeciego wejścia sterującego pracą rejestru. W związku z dodaniem tego piątego trybu pracy testowanie UK1 oraz UK2 odbywa się w dwóch kolejnych sesjach [WillP82, Wang87, Hławi96c]. W pierwszej sesji odbywa się kompaktacja w rejestrze MISR odpowiedzi UK1, natomiast w drugiej sesji generowane są przez rejestr LFSR testy PR dla UK2. Dwie przykładowe komórki umożliwiające budowę rejestru MBILBO przedstawiono w pracy [Hławi96c].

Kolejną odmianą rejestru BILBO jest opracowany przez Wangę [WangM86d, WangM87b, Wang87, Hławi96c] rejestr CBILBO (ang. Concurrent BILBO). Dzięki podwojeniu liczby przerzutników rejestr ten (rys. 1.3d) można programować w rejestr MISR i jednocześnie w niezależny rejestr LFSR. Drugim trybem pracy odróżniającym go od rejestrów BILBO i MBILBO i wykorzystującym podwójną liczbę przerzutników jest normalna praca systemowa z testowaniem "on-line" na bieżąco (concurrent). Wówczas oprócz równoległego rejestru potrzebnego do normalnej pracy systemowej niezbędny jest drugi rejestr MISR pełniący rolę kompaktora. Pozostałe dwa rodzaje pracy to przesuwanie oraz praca normalna bez testowania. Do ustawiania rodzaju pracy w rejestrze CBILBO potrzebne są dwa sygnały sterujące. Przykładowe dwie komórki umożliwiające budowę rejestru CBILBO przedstawiono w pracy [Hławi96c]. Kolejnym przykładem wielofunkcyjnego rejestru jest rejestr MISR-MR (MISR based Memory Register) przedstawiony na rys. 1.3e. Rejestr ten umożliwia realizację dwóch rodzajów pracy: kompaktacji w rejestrze MISR oraz przesuwania. Dwie przykładowe komórki tego rejestru przedstawiono w pracy [Hławi96c]. Stosowanie rejestru MISR-MR wymaga modyfikacji funkcji f_K układu UK1 funkcją f_M związaną ze sprzężeniami liniowymi rejestru MISR. Tak zmodyfikowany układ UK1' realizuje funkcję $f_K \oplus f_M$. W konsekwencji w odróżnieniu od wszystkich poprzednich wielofunkcyjnych rejestrów, które w trybie MISR realizują kompaktację odpowiedzi układu kombinacyjnego o funkcji $f_K \oplus f_M$, rejestr MISR-MR realizuje w trybie MISR kompaktację odpowiedzi układu kombinacyjnego o funkcji $(f_K \oplus f_M) \oplus f_M = f_K$. W efekcie każdy bieżący stan wewnętrzny (bieżąca sygnatura) zawarty w i -tym kroku kompaktacji w rejestrze w trybie MISR jest identyczny do bieżącego stanu wyjścia UK1. Innymi słowy, rejestr MISR-MR w trybie MISR pełni jednocześnie funkcję rejestru równoległego MR. Nie trzeba ustawiać trybu testowanie, ponieważ każdy stan zawarty w rejestrze MISR-MR w trakcie normalnej systemowej pracy jest jednocześnie bieżącą sygnaturą. Umożliwia to

realizację testowania w trybie "on-line" w przeciwieństwie do rejestrów SP, BILBO oraz MBILBO, które realizują testowanie tylko w trybie "off-line". Dzięki takiej idei testowania można w rejestrze MISR-MR ograniczyć liczbę sygnałów sterujących do jednego sygnału, a także nieco uprościć komórki z bramkami XOR liniowego sprzężenia [Hławi96c]. Niestety taka idea testowania wpływa niekorzystnie na współczynnik pokrycia uszkodzeń UK2. Ze względu na to, że w rejestrze MISR-MR testy układu UK2 należą do zbioru wektorów wyjściowych UK1, prawdopodobieństwo pojawienia się różnych wektorów z tego zbioru jest zróżnicowane. Może się nawet zdarzyć, że w zbiorze tym nie będzie testów niezbędnych do uzyskania wysokiego współczynnika pokrycia uszkodzeń UK2. Współczynnik ten silnie więc będzie zależeć od funkcji f układu UK1 oraz testów podawanych na wejście UK1. W celu sprawdzenia WPU niezbędna staje się symulacja komputerowa.

W każdym z omówionych już rejestrów BILBO, MBILBO, CBILBO oraz MISR-MR generacja testów oraz kompaktacja odbywa się w sposób autonomiczny. Generacja testów zależy wyłącznie od struktury sprzężenia liniowego rejestru LFSR (MBILBO, CBILBO) albo struktury sprzężenia liniowego rejestru MISR i funkcji f_K (BILBO, MISR-MR). Podobnie jest z kompaktacją.

Przedstawiony na rys. 1.3f rejestr MIShR (Multi-Input Shift Register) [KraśP87b, KraśP88, Kraśn89, Badur89, Hławi96c] nie zawiera sprzężenia liniowego. Zawarte w każdym stopniu tego rejestru bramki XOR umożliwiają dodawanie modulo dwa poprzedniego stanu rejestru do aktualnego stanu wyjść UK1 i przesunięcie uzyskanego rezultatu o jeden stopień wzdłuż tego rejestru. Takie funkcjonowanie rejestru MIShR w przypadku jego autonomicznego zastosowania nie zapewnia generacji testów gwarantujących wysoki współczynnik pokrycia uszkodzeń UK2. Podobnie nieskuteczna jest w takim wypadku kompaktacja odpowiedzi UK1. W celu zwiększenia skuteczności testowania rejestr MIShR wpina się w pierścień testujący CSTP (ang. Circular Self Test Path) [KraśP88], w ścieżkę testującą STP (ang. Self Test Path) [Badur89] lub w skondensowany pierścień testujący CCSTP (Condensed CSTP) [BaduH96a, BaduH96b, BaduH96c, BaduH97]. Wówczas do wejścia SI rejestru MIShR podawana jest z zewnątrz quasi-przypadkowa sekwencja, która na bieżąco modyfikuje zawartość komórek pamięci tego rejestru. Sekwencja ta może być podawana albo z wyjścia SO innego rejestru MIShR, lub rejestru SR (ang. Shift Register) zawartych w pierścieniu CSTP, w ścieżce testującej STP, w skondensowanym pierścieniu testującym CCSTP, albo z wyjścia rejestru LFSR

wpiętego do pierścienia CSTP [Kopeć94]. Wpięcie rejestru MIShR w pierścień CSTP, CCSTP lub w ścieżkę STP umożliwia wyprowadzanie informacji o błędach w odpowiedziach UK1 poprzez wyjście SO do innych elementów pamięci, co zwiększa skuteczność kompaktacji. W każdym z tych przypadków niezbędna jest jednak symulacja komputerowa sprawdzająca skuteczność testowania [KraśP88, Badur89, BrynA90, BaduH96c]. Brak sprzężenia liniowego upraszcza nieco konstrukcję opisanych w [Hławi96c] komórek tego rejestru i zmniejsza nieznacznie czas testowania (mniejszy czas propagacji komórek). Dwa sygnały sterujące umożliwiają ustawianie następujących rodzajów pracy: normalna praca systemowa (rejestr MR), przesuwanie, zerowanie oraz testowanie (rejestr MIShR).

Najistotniejsze parametry związane z nadmiarem układowym, szybkością pracy i efektywnością testowania przy użyciu omówionych wielofunkcyjnych rejestrów przedstawiono w tabeli 1.1. Opóźnienie wnoszone do normalnej systemowej pracy przez komórki rejestrów BILBO, MBILBO, CBILBO, MISR-MR oraz MIShR założono, że jest jednakowe. Wynika to ze schematów komórek tych rejestrów przedstawionych w [Hławi96c]. W związku z tym w tabeli 1.1 nie uwzględniono wpływu sprzętu ułatwiającego testowanie na wydajność normalnej systemowej pracy testowanych układów.

1.3.2. Rejestry liniowe w testerach wewnątrzukładowych samotestowalnych układów sekwencyjnych

Schemat konwencjonalnego układu sekwencyjnego (US) przedstawiono na rys. 1.4a. Załóżmy, że układ ten posiada n wejść pierwotnych, m wyjść pierwotnych oraz blok pamięci w postaci rejestru MR złożonego z p przerzutników. W trakcie testowania stany S rejestru MR są jednocześnie testami, które poprzez pętlę sprzężenia zwrotnego wprowadzane są na wejścia US związane z p zmiennymi wewnętrznymi. W efekcie pojedynczym testem dla wybranego uszkodzenia testowanego układu sekwencyjnego będzie wektor $T = (I, S)$ złożony z wektora I podanego na wejścia pierwotne US i wektora stanu S. Niestety tester zewnętrzny TZ ma jedynie dostęp do n wejść pierwotnych, co utrudnia wprowadzanie bloku pamięci w pożądaną stan S. Na przykład ustawienie stanu S gwarantującego pobudzenie uszkodzenia (tzw. D-pobudzenie) [AbraB90, Hławi93b] lub też doprowadzenie do wyjść pierwotnych objawów uszkodzenia obserwowanych w bloku pamięci (tzw. D-sterowanie) może wymagać wielu dodatkowych taktów zegarowych [AbraB90, Hławi93b, Hławi97b]. W efekcie wyznaczanie testów dla US jest ogromnie pracochłonne

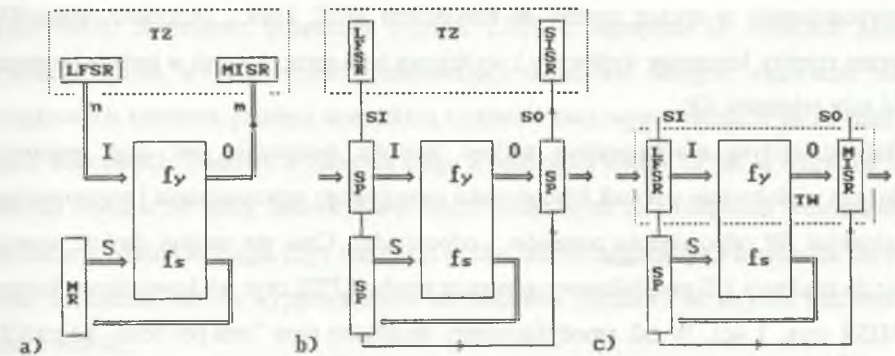
Tabela 1.1

REJESTR	SP	BILBO	MBILBO	CBILBO	MISR-MR	MIShR
Zależność od funkcji testowanego układu	nie	nie	nie	nie	nie	tak
Współczynnik pokrycia uszkodzeń UK2 przez testy	wys.	wys.	wys.	wys.	śr.	wys.
Symulacja komputerowa	nie	tak	nie	nie	tak	tak
Skuteczność kompaktacji odpowiedzi UK1	wys.	wys.	wys.	wys.	wys.	wys.
Jednoczesne testowanie UK1 oraz UK2	tak	tak	nie	tak	tak	tak
Liczba sesji testowania UK1 oraz UK2	$n \gg 2$	1	2	1	1	1
Czas testowania w ramach jednej sesji	min.	śred.	mały	mały	śred.	mały
Sumaryczny czas testowania	b.d.	śred.	duży	mały	śred.	mały
Testowanie "on-line"	nie	nie	nie	tak	tak	nie
Liczba sygnałów sterujących pracą rejestrów	1	2	3	2	1	2
Koszt sprzętu	b.m.	duży	duży	b.d.	śred.	mały
Zerowanie	szer.	równ.	równ.	szer.	szer.	równ.
Dodatkowy sprzęt zewnętrzny (LFSR, SISR, albo STP, CSTP)	tak	nie	nie	nie	nie	tak

i kosztowne. Bardzo trudnym zadaniem staje się wyznaczenie dla US takiego zbioru wektorów testowych I, który mógłby ustawiać kolejno blok pamięci w takie stany S, aby wygenerować w 2^{n+p} krokach cały zbiór testów wyczerpujących $T = (I, S)$ [AbraB90, Hławi93b] lub tym bardziej w dużo mniejszej liczbie kroków wygenerować pełny zestaw testów deterministycznych T (funkcjonalnych lub strukturalnych) [AbraB90, Hławi93b]. Do ich wyznaczenia niezbędne są drogie systemy automatycznej generacji testów ATPG (Automatic Test Pattern Generation). Poza tym testy te są skuteczne tylko dla założonego modelu uszkodzeń.

Metodą generacji testów, która radzi sobie częściowo z tymi problemami, jest generacja testów pseudolosowych na wejściach pierwotnych US. Ich "naturalność" [Kraśn89] umożliwia zwiększenie współczynnika pokrycia uszkodzeń związanych z dynamiką US,

np. opóźnień, hazardów itp. oraz uszkodzeń, których nie można opisać za pomocą założonego modelu uszkodzeń (non-target faults). Metoda testowania pseudolosowego (rys. 1.4a) podobna jest do testowania zastosowanego w schemacie BEST (Built-in Evaluation and Self-Test) podanym w [AbraB90]. Niestety podczas takiego testowania prawdopodobieństwo występowania różnych stanów S w bloku pamięci MR jest zróżnicowane i zależne od funkcji f_s układu sekwencyjnego i sekwencji testów I generowanych na wejściu pierwotnym przez tester zewnętrzny. W konsekwencji dla niektórych US można uzyskać bardzo duże zróżnicowanie wspomnianych prawdopodobieństw [Hławi97b]. Po za tym nie zawsze w sekwencji testów I generowanych losowo przez TZ znajdują się te ciągi testów, które w testowanym US są niezbędne do pobudzenia uszkodzeń lub do wyprowadzenia ich objawów (błędów) na wyjścia pierwotne do TZ. Problem ten można rozwiązać wprowadzając w miejsce bloku pamięci MR rejestr SP pracujący jako rejestr przesuwający albo rejestr równoległy. Zbędna staje się w takim rozwiązaniu konieczność doprowadzania do wejść pierwotnych specjalnego ciągu testów wprowadzających pamięć MR w pożądany stan. Wówczas też testowanie US zostaje sprowadzone do problemu testowania układu kombinacyjnego o $n+p$ wejściach pierwotnych [AbraB90, Hławi93b, Hławi97b].



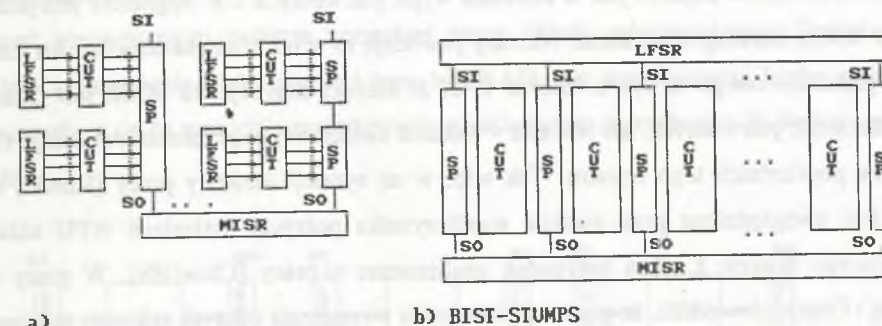
Rys. 1.4. Schemat układu sekwencyjnego: konwencjonalnego z testerem zewnętrznym TZ (a), ze ścieżką SP oraz z uproszczonym TZ (b), z testerem wewnątrzukładowym TW (c)

Fig. 1.4. Sequential circuit scheme: conventional with external tester TZ (a), with scan path SP and simplified TZ (b), with built-in circuit tester TW (c)

Podłączenie zewnętrznego testera do US związane jest z problemem połączenia $n+m$ linii testera z n wejściami i m wyjściami układu sekwencyjnego. Tę trudność usuwa się podłączając do wejść i wyjść pierwotnych US dwa dodatkowe rejestry SP. Ilustruje to

rysunek 1.4b [Hławi97b]. Oba rejestry SP w trakcie normalnej systemowej pracy US muszą być przezroczyste dla wektorów wejściowych I wprowadzanych z zewnątrz oraz dla wektorów wyjściowych O wyprowadzanych na zewnątrz. Rejestry te połączone szeregowo z rejestrem SP stanowiącym blok pamięci umożliwiają poprzez jedynie dwa wyprowadzenia (wejście SI i wyjście SO) wprowadzanie z zewnątrz testów i wyprowadzanie na zewnątrz odpowiedzi TUC. Tester zewnętrzny wyposażony między innymi np. w rejestr LFSR oraz kompaktor SISR realizuje proces testowania US, który podobnie jak w schemacie testowania "LSSD On-Chip Self-Test" (LOCST) [AbraB90] złożony jest z wielu sesji. Każda z nich polega na szeregowym wprowadzaniu testów do rejestrów SP podłączonych odpowiednio do n wejść pierwotnych oraz p wejść bloku pamięci, a następnie zapisie do rejestru równoległego w bloku pamięci i rejestru równoległego podłączonego do m wyjść pierwotnych odpowiedzi układów realizujących odpowiednio funkcje f_y i f_s . Szeregowo wyprowadzanie na zewnątrz tych odpowiedzi odbywa się jednocześnie z wprowadzaniem testów. Schematy testerów o takim scenariuszu testowania nazwano w pracy [AgraK93b] schematami typu "test-per-scan". Niestety rejestry SP w takich testerach wprowadzają dodatkowy nadmiar układowy, który należy uwzględnić w przypadku projektowania łatwo testowalnych US. Znaczące zmniejszenie tego kosztu następuje w przypadku projektowania US wyposażonego w sprzęg zgodny ze standardem IEEE 1149.1 [AbraB90, Hławi93b]. Wówczas rejestry brzegowe wejściowy i wyjściowy tego sprzęgu mogą w trakcie testowania pełnić rolę rejestrów SP.

Najistotniejszym mankamentem takiego sposobu testowania jest czas testowania, wydłużany wielokrotnie wskutek konieczności szeregowego wprowadzania i wyprowadzania do rejestrów SP odpowiednio paternów i odpowiedzi. Czas ten można skrócić wprowadzając do struktury US pseudolosowy generator testów LFSR oraz wielowejsiowy kompaktor MISR (rys. 1.4c). W tak zmodyfikowanej strukturze typu "test-per-scan" [AgraK93b] proces testowania na wejściach pierwotnych oraz wyjściach pierwotnych z powrotem z szeregowego staje się równoległy. Schematy testowania przedstawione na rys. 1.5 są innymi przykładami testowania typu "test-per-scan". Ich cechą charakterystyczną jest "scan-based compactor MISR" [PilaK95, Hławi97b]. Na rys. 1.5a przedstawiona jest struktura z kilkoma różnymi generatorami testów LFSR [PilaK95], natomiast na rys. 1.5b zilustrowano strukturę testera typu BIST-STUMPS (Self-Test Using MISR and Parallel SRSG) opisaną w [BardM82, BardM87, AbraB90, Hławi97b].



Rys. 1.5. Schematy testerów wewnątrzukładowych typu "test-per-scan": z wieloma generatorami testów LFSR (a), BIST-STUMPS (b)

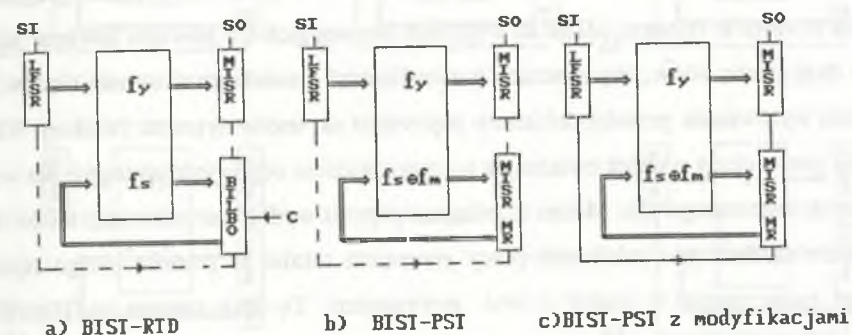
Fig. 1.5. Schemes of "test-per-scan" types built-in circuits testers: with a number of test generators LFSR (a), BIST-STUMPS (b)

Rolę rejestrów LFSR oraz MISR w testerze wewnątrzukładowym mogą pełnić rejestry brzegowe wejściowy i wyjściowy sprzęgu IEEE 1149.1 (rys. 1.4c.) uzupełnione dodatkowymi brankami XOR [Hławi87b, Hławi88c, GlosB89, Hławi89c, Hławi89e]. Wówczas schemat testowania przedstawiony na rys. 1.4c staje się podobny do schematu testowania typu "Centralized and Embedded architecture with Boundary Scan" (CEBS) opisanego w [AbraB90]. Szeregowo połączenie rejestru LFSR z rejestrem SP (blokiem pamięci) i rejestrem MISR w jeden rejestr przesuwający umożliwia wstępne ustawianie stanów początkowych komórek pamięci wszystkich rejestrów oraz wyprowadzanie na zewnątrz ich stanów końcowych. Niestety w dalszym ciągu z generatora testów LFSR są wsuwane szeregowo do rejestru SP testy losowe dla p wejść związanych ze zmiennymi wewnętrznymi. W efekcie w dalszym ciągu zbyt duża jest liczba taktów zegarowych niezbędna do pobudzenia uszkodzeń lub do wyprowadzenia ich objawów (błędów) na wyjścia pierwotne do rejestru MISR.

W celu znaczącego skrócenia czasu podawania testów losowych na p wejść związanych ze zmiennymi wewnętrznymi zastąpiono rejestr SP w bloku pamięci US pojedynczym wielofunkcyjnym rejestrem BILBO [BardM82, BardM83, BardM86]. Otrzymał strukturę BIST-RTD (BIST Random Test Data) [AbraB90, Hławi97b] zilustrowano na rys. 1.6a. Umożliwia ona realizację testowania w ramach jednej sesji, co znacząco upraszcza układ sterowania testowaniem. Dzięki temu, że sygnatura wytworzona w rejestrze MISR (BILBO) zawiera informację o błędach, nie muszą być one przesuwane (D-sterowanie) za pomocą

dodatkowych taktów zegarowych w kierunku wyjść pierwotnych US. Sygnatura jest jednocześnie testem dla magistrali stanu. Niestety powoduje to w odróżnieniu od schematu testowania przedstawionego na rys 1.4b oraz 1.4c, że bieżący stan rejestru MISR jest funkcją nie tylko wejść pierwotnych, ale również - wskutek nieliniowego sprzężenia zwrotnego (f_s) - stanów poprzednich tego rejestru. Tak więc w tej sytuacji rezultaty pracy [Kim88] nie mogą być uwzględnione przy analizie współczynnika pokrycia uszkodzeń WPU układu realizującego funkcję f_s . Ten przypadek analizowano w pracy [ChuaG89]. W pracy tej Chuang i Gupta udowodnili, że prawdopodobieństwo wystąpienia różnych sygnatur traktowanych jako testy może być do tego stopnia zróżnicowane, że niektóre z nich mogą w ogóle nie wystąpić. W związku z tym współczynnik WPU może się okazać zbyt niski. Problem ten jest spowodowany dwoma rodzajami pracy rejestru BILBO jako systemowego bloku pamięci MR oraz w trakcie testowania jako kompaktora MISR. W trybie testowania w linii wyjściowe testowanego układu sekwencyjnego wprowadzane są dodatkowe bramki XOR, podczas gdy w trybie normalnej pracy systemowej te bramki są wyłączone przez zewnętrzne sygnały sterujące. Wpływa to na zmianę funkcji wzbudzeń przerzutników bloku pamięci. W efekcie graf przejść dla normalnej pracy układu sekwencyjnego różni się od grafu jego pracy podczas samotestowania. W celu zwiększenia współczynnika WPU autorzy pracy [ChuaG89] zaproponowali specjalny algorytm kodowania tablicy przejść US uwzględniający funkcję f_m . Algorytm ten umożliwia zmniejszanie zróżnicowania prawdopodobieństwa wystąpienia różnych sygnatur, a przede wszystkim pozwala uzyskać wszystkie sygnatury. Złożoność tego algorytmu wpływa jednak na brak popularności jego stosowania. W związku z tym Escherman i Wunderlich w pracy [EschW91] zaproponowali zastąpienie w US rejestru BILBO rejestrem MISR-MR. Ilustruje to rys. 1.6b. Tę ideę samotestowania US jeszcze dwukrotnie opisywano jako tzw. strukturę BIST-PST (ang. BIST-Parallel Self-Test) w pracach [Esche92a, Esche92b], a następnie w pracach [Hławi97a, Hławi97b, Hławi97b]. Funkcja wzbudzeń bloku pamięci MISR-MR powstaje jako wynik sumy modulo dwa sygnałów reprezentujących funkcje wzbudzeń f_m przerzutników D dla zwykłego rejestru MISR oraz sygnałów reprezentujących funkcje wzbudzeń f_s przerzutników D dla układu sekwencyjnego będącego obiektem testowania. W efekcie nie funkcja wzbudzeń f_s , ale nowa funkcja wzbudzeń $f_s'' = f_s \oplus f_m$ jest realizowana dla tego układu sekwencyjnego. Tak więc w strukturze BIST-PST stan wewnętrzny rejestru MISR-MR jest jednocześnie stanem wewnętrznym układu sekwencyjnego. Samotestowanie i normalna praca w przy-

padku tej struktury BIST to jedno i to samo. Innymi słowami, sygnatury rejestru MISR-MR są stanami wewnętrznymi podczas normalnej pracy układu sekwencyjnego. Dodatkowym efektem zastosowania bloku pamięci typu MISR-MR jest zredukowanie liczby sygnałów sterujących, a co za tym idzie zredukowanie części sprzętu potrzebnego do budowy rejestru BILBO.



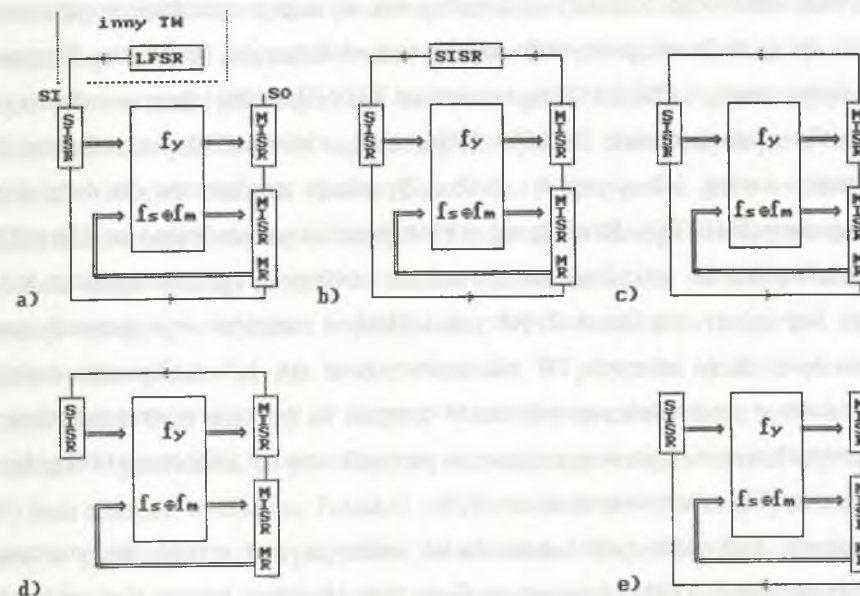
Rys. 1.6. Tester wewnętrzny dla US: z rejestrem BILBO (a), z blokiem pamięci w postaci rejestru MISR-MR (b) oraz (c)
 Fig. 1.6. A built-in circuit tester for sequential circuits US: with BILBO register (a), with memory in the form of MISR-MR register (b) and (c)

Ze względu na to, że graf związany z samotestowaniem nie różni się od grafu opisującego normalną pracę US, wykrywane są również te uszkodzenia, które w strukturze BIST-RTD z rejestrem BILBO były niewykrywalne. Efekt ten uzyskano dzięki możliwości wejścia w każdy stan (test-sygnatura) podczas samotestowania. Dzięki możliwości samotestowania podczas realizowania funkcji systemowych można także wykrywać uszkodzenia opóźnieniowe (delay faults) [Esche92a, Esche92b]. Wymaga to jedynie zwiększenia częstotliwości zegara systemowego do wartości granicznej i równie szybkiego podawania testów pseudoprzypadkowych na wejściach pierwotnych układu sekwencyjnego.

Niestety struktura BIST-PST nie likwiduje problemu zróżnicowania w trakcie samotestowania prawdopodobieństw występowania testów-sygnatur. Jest więc w tym podobna do schematu przedstawionego na rys. 1.4a. Pewną niedużą poprawę w tym zakresie można uzyskać za pomocą opisanego w [Hławi97b] schematu testera TW zilustrowanego na rys. 1.6c. Na rysunku tym wprowadzono do testera typu BIST-PST modyfikacje zwiększające współczynnik pokrycia uszkodzeń WPU. Pierwszą z nich jest połączenie wyjścia generatora testów LFSR z dodatkowym wejściem bloku pamięci MISR-MR, co w wielu przy-

padkach nieco zmniejsza różnice w prawdopodobieństwach pojawiania się testów-sygnatur. Drugą modyfikacją jest połączenie w szereg rejestru MISR-MR z kompaktorem MISR. Ta druga modyfikacja umożliwi wprowadzanie do kompaktora MISR błędów zamaskowanych w rejestrze MISR-MR. Wpływa to dodatkowo na zwiększenie efektywności testowania. Ponadto można wówczas po odpowiedniej liczbie dodatkowych taktów zegarowych zrezygnować z odczytywania sygnatury z bloku pamięci. Oczywiście w takim przypadku sygnatura zawarta w rejestrze MISR na wyjściach pierwotnych US powinna zawierać odpowiednio dużą liczbę bitów, aby znacząco minimalizować zjawisko maskowania błędów.

W celu wyrównania prawdopodobieństw pojawiania się testów-sygnatur (większy WPU) potrzebny jest większy wysiłek związany z wygenerowaniem odpowiednich testów dla wejść pierwotnych testowanego US. Można to osiągnąć poprzez wydłużanie sekwencji testów PR. Innym rozwiązaniem jest zakłócanie pracy generatora testów za pomocą innego rejestru liniowego połączonego w szereg z ww. generatorem. Tę ideę opisaną w [Hławi97b] przedstawiają schematy testerów wewnątrzukładowych pokazane na rys. 1.7. Umożliwiają one wyrównywanie prawdopodobieństw testów-sygnatur w rejestrze MISR-MR. Podejście to uzasadniają pozytywne wyniki opisane w pracy [Kopeć94]. Rejestr zakłócający nie musi być integralną częścią testera wewnątrzukładowego samotestowanego US. Gdy rejestr ten będzie częścią składową innego testera wewnątrzukładowego TW, tak jak to ilustruje rys. 1.7a, wówczas zwiększanie tą drogą współczynnika WPU nie wymaga nadmiaru układowego. Przedstawione na rys. 1.7b oraz rys. 1.7c modyfikacje polegające na łączeniu w pierścienie wszystkich rejestrów liniowych testera wewnątrzukładowego również umożliwiają - jak już wcześniej wspomniano - pożyteczne zakłócanie pracy generatora testów na wejściach pierwotnych testowanego US. Przypominają one pierścienie testujące CSTP (rys. 1.7c) opisany w pracach [KraśP87b, KraśP89, KraśN89, PilaK90, KraśP92] oraz technikę SEREG (ang. SERial REGISTER) [BrynA90] albo ich modyfikacje (rys. 1.7b) zaproponowane w [Kopeć94]. Rozwiązania przedstawione na rys. 1.7d oraz 1.7e podobne są do ścieżki samotestującej STP [Badur89, Badur90a, Badur90b, Badur92]. Ogólną cechą charakterystyczną testera TW zilustrowanego na rys. 1.7a jest to, że proces generacji testów na wejściach pierwotnych testowanego US jest niezależny od funkcji testowanego układu. W pozostałych strukturach testerów TW przedstawionych na rys. 1.7. ta właściwość nie została utrzymana, co jest źródłem dodatkowych trudności w analizie skuteczności testowania.



Rys. 1.7. Różne schematy testerów wewnątrzukładowych z blokiem pamięci MISR-MR oraz z zakłócaniem generacji testów na wejściach pierwotnych US
Fig. 1.7. Different schemes of built-in circuit testers with MISR-MR type memory and with test generation disturbance in the primary inputs of sequential circuit US

Opracowana w [HławiB94a, HławiB94b] metoda optymalizacji procesu syntezy funkcji $f_s'' = f_s \oplus f_m$ umożliwia kompensowanie nadmiaru układowego każdego testera TW przedstawionego na rys. 1.7 (także na rys. 1.6b i 1.6c) za pomocą zmniejszania kosztu realizacji sprzętowej układu testowanego US. Polega ona na takim doborze sprzężenia liniowego rejestru MISR i jego specjalnej struktury IEDT zawierającej przerzutniki T [Hławi92a, Hławi96b], aby funkcja f_m redukowałą koszt realizacji układowej funkcji $f_s'' = f_s \oplus f_m$.

W testerach wewnątrzukładowych przedstawionych na rys. 1.7, z wyjątkiem testera zilustrowanego na rys. 1.7a, sygnatura jest odczytywana we wszystkich rejestrach liniowych. Wpływa to niewątpliwie na znaczące zredukowanie zjawiska maskowania błędów w tych testerach TW i zwiększenie rozdzielczości uszkodzeń. Można w nich również zamienić kompaktory MISR na rejestry MISHR, co po pierwsze znacznie bardziej upodabnia te testery do ich wzorców (pierścienia testującego CSTP oraz ścieżki testującej STP), a po drugie pozwala dodatkowo nieznacznie zredukować nadmiar układowy.

Wszystkie omówione schematy samotestowania wymagają symulacji komputerowej niezbędnej do sprawdzenia praktycznie uzyskiwanej efektywności testowania. W ramach projektu badawczego nr PBZ 24/05 na zamówienie KBN [Hławi96a] przeprowadzono taką symulacją dla różnych modeli ISCAS89 (s386, s298, s1494, s820) oraz testerów TW zilustrowanych na rys. 1.7a,c oraz rys. 1.6b,c. Symulacje zrealizowano dla dużej liczby taktów zegarowych (10000). Rezultaty tej symulacji częściowo omówiono w [Hławi97b]. Współczynnik pokrycia uszkodzeń stabilizował się na długo przed zakończeniem każdej symulacji. Największy współczynnik pokrycia uszkodzeń osiągnięto w strukturach samotestowania opartych na testerach TW zilustrowanych na rys. 1.7a,c. Znacznie mniejszy WPU uzyskano w strukturach samotestowania opartych na testerach wewnątrzukładowych przedstawionych na rys. 1.6c, a najmniejszy w przypadku modeli s298 oraz s1494 połączonych z testerem TW zilustrowanym na rys. 1.6b.

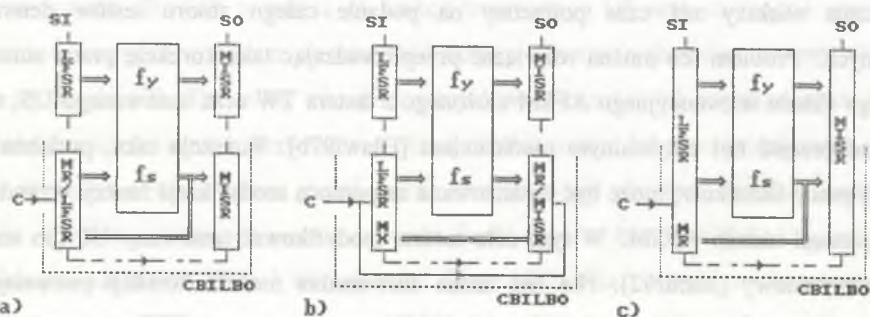
Skuteczność testowania można także badać analizując graf przejść autonomicznego układu sekwencyjnego AFSM (Autonomous Finite State Machine), którego blokiem pamięci są rejestry podłączone do $n+p$ wejść testowanego US [BrynA90, Badur89, Badur90a, Badur90b, Badur92, Hławi97b]. Funkcje wzbudzeń przerzutników zawartych w tym bloku pamięci decydują o formach, jakie może przyjąć graf przejść związany z danym schematem samotestowania. Funkcje te zależą od funkcji testowanego US oraz od funkcji wszystkich rejestrów zawartych w testerze wewnątrzukładowym również tych, które nie są podłączone do $n+p$ wejść US. Tak więc taki graf przejść w zależności od wspomnianych funkcji wzbudzeń może zawierać pierścienie o różnej liczbie jedno- lub dwuwęściowych stanów oraz związane z nimi pewne ilości gałęzi zawierających również niejednakową liczbę stanów. Graf ten może też zawierać pewną liczbę stanów bez jakichkolwiek wejść albo stanów posiadających kilka i więcej wejść [BrynA90]. Graf przejść z pierścieniem bez gałęzi o cyklu 2^{n+p} i zawierającym 2^{n+p} różnych jednowęściowych stanów ułatwia uzyskanie maksymalnej możliwej do osiągnięcia skuteczności testowania. Z punktu widzenia testowania jest to optymalny pierścień stanów. Przykład testera TW, który wraz z testowanym US tworzy autonomiczny układ sekwencyjny AFSM o grafie przejść z optymalnym pierścieniem, podano w pracy [Hławi97b]. W takim pierścieniu zawsze znajdują się sekwencje stanów, które umożliwiają przy minimalnej liczbie taktów zegarowych osiągnięcie bardzo wysokiego współczynnika pokrycia uszkodzeń WPU w testowanym US. Każda z nich rozpoczyna się w innym stanie początkowym. Wśród tych sekwencji można

wybrać taką, której końcowy stan gwarantuje uzyskanie minimalnego maskowania błędów lub wręcz jego brak. W przypadku struktur testerów TW przedstawionych na rys. 1.7 oraz na rys. 1.6 jest mało prawdopodobne uzyskanie grafu przejść w postaci optymalnego pierścienia. Realne jest natomiast uzyskanie grafów przejść zawierających jeden lub więcej pierścieni o różnych liczbach stanów [BrynA90]. W przypadku grafu z pierścieniem o liczbie stanów dużo mniejszej od liczby 2^{n+p} wskutek cyklicznego powtarzania się stanów pierścienia (limit cycling) [BrynA90, Badur92] może okazać się niemożliwe znalezienie sekwencji testującej o wysokim współczynniku WPU. Przykładem mogą być struktury samotestowania podane na rys. 1.6b,c. Natomiast w przypadku grafu z pierścieniem o liczbie stanów dużo większej od liczby 2^{n+p} niektóre ze stanów mogą być wielokrotnie powtarzane w pierścieniu, co powoduje, że sekwencje zapewniające wysoki współczynnik WPU będą znacznie wydłużone. Jednakże prawdopodobieństwo cyklicznego powtarzania się stanów w pierścieniu w tym przypadku jest minimalne [Kopeć96]. Takie pierścienie są cechą charakterystyczną grafów przejść autonomicznych układów sekwencyjnych AFSM zawierających testery wewnątrzukładowe zilustrowane na rys. 1.7a,b,c,d,e.

Po tej dyskusji nasuwa się następujące pytanie: Jak zaprojektować tester wewnątrzukładowy, aby WPU był maksymalny, a czas testowania najkrótszy z możliwych, czyli nieznacznie większy niż czas potrzebny na podanie całego zbioru testów deterministycznych? Problem ten można rozwiązać przeprowadzając taką korekcję pracy autonomicznego układu sekwencyjnego AFSM złożonego z testera TW oraz testowanego US, aby jego graf przejść był optymalnym pierścieniem [Hławi97b]. Korekcja taka, podobna do korekcji pracy liczników, może być zrealizowana za pomocą modyfikacji funkcji wzbudzeń bloku pamięci układu AFSM. W tym celu można modyfikować testowany US lub tester wewnątrzukładowy [Badur92]. Nie jest znana uniwersalna metoda korekcji pozwalająca w przypadku każdego testowanego US oraz każdej struktury testera TW otrzymać układ AFSM o grafie przejść z pierścieniem optymalnym. W każdym indywidualnym przypadku wprowadzenie korekcji gwarantującej uzyskanie takiego pierścienia związane jest z dużym kosztem na etapie projektowania, i najczęściej z nieopłacalnym dodatkowym nadmiarem układowym. Czy istnieje jakieś wyjście z tego impasu?

Pewne rozwiązanie opisanego problemu można uzyskać wprowadzając w miejsce bloku pamięci MR testowanego US rejestr CBILBO z podwojoną liczbą $2p$ przerzutników. Umożliwia to budowanie dwóch oddzielnych niezależnych rejestrów. Jeden z nich służy do

kompakcji stanów, a drugi służy do niezależnej generacji testów RP na magistrali stanu. W efekcie takie rozwiązanie pozwala na czas testowania rozcinać magistralę stanu i sprowadzać problem testowania US do problemu testowania UK. Struktury testerów wewnątrzukładowych dla US zawierających rejestr CBILBO opisano po raz pierwszy w pracach [WangM86d, WangM87b, Wang87]. Ich trzy rodzaje przedstawiono na rys. 1.8. W pracach [EschW91, EschW92], a potem w [Hławi97b] nazwano je strukturami typu BIST-DFF (D-Flip Flop). Uzasadniono to tym, że w rejestrze CBILBO, w odróżnieniu od rejestru MISR-MR, rejestr stanu układu sekwencyjnego wykorzystywany jest wyłącznie jako blok rozłącznych przerzutników D podczas normalnej pracy systemowej. W strukturze BIST-DFF (rys. 1.8a) p przerzutników rejestru stanu MR układu sekwencyjnego podczas testowania pracuje jako generator testów LFSR, natomiast pozostała nadmiarowa grupa p przerzutników pracuje jako kompaktor MISR. W przypadku struktury przedstawionej na rys. 1.8b rejestr stanu MR pracuje podczas testowania jako kompaktor MISR. Pozostała grupa p przerzutników pracuje wtedy jako generator LFSR (na rysunku rejestr LFSR MX). Posiada on na swoich wyjściach multiplexer (MX), który w zależności od wartości sygnału sterującego C podłącza do wejść UK realizującego funkcję f_s albo wyjścia bloku pamięci MR (normalna systemowa praca), albo wyjścia rejestru LFSR (podczas testowania).



Rys. 1.8. Tester wewnątrzukładowy z rejestrem CBILBO (a), w którym rejestr MR pracuje jako generator testów LFSR (b), w którym rejestr MR pracuje jako kompaktor MISR (c) połączony z resztą rejestrów w dwa rejestry liniowe

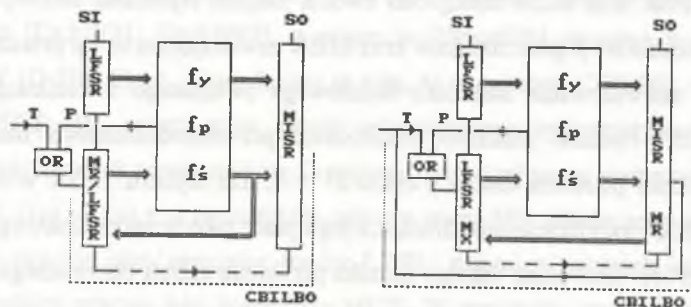
Fig. 1.8. A built-in circuit tester with CBILBO register: in which MR register operates as test generator LFSR (a), in which MR register operates as compactor MISR (b), connected with the other registers into two linear registers (c)

Jeżeli cykle pracy obu rejestrów LFSR podłączonych do $n+p$ wejść US będą wyrażone różnymi liczbami pierwszymi 2^n-1 oraz 2^p-1 , wówczas cykliczną sekwencję testów generowaną przez te rejestry można przedstawić w postaci pierścienia o cyklu $(2^n-1)(2^p-1)$. Taki

pierścień podobny do pierścienia optymalnego pozwala uzyskać wysoki współczynnik WPU i dobrać taki stan początkowy pracy obu rejestrów, aby czas testowania oraz maskowanie błędów były jak najmniejsze. Lepsze rezultaty można uzyskać w strukturze BIST-DFF przedstawionej na rys. 1.8c [Hławi97b]. W tym testerze wewnątrzukładowym cztery krótkie rejestry LFSR oraz MISR zastąpiono dwoma długimi rejestrami liniowymi odpowiednio LFSR o liczbie $n+p$ przerzutników oraz MISR zawierającym $m+p$ przerzutników. Pozwoliło to na zlikwidowanie nadmiaru układowego związanego z realizacją dwóch sprzężeń liniowych. Ponadto znacząco zmniejszono prawdopodobieństwo maskowania błędów oraz uzyskano pierścień testów o cyklu $2^{n+p}-1$. Do rejestru LFSR w schemacie c można wprowadzić modyfikacje umożliwiające jego pracę jako licznika liniowego (licznik de Bruijn) [Hławi97a]. Graf pracy takiego licznika przyjmuje kształt optymalnego pierścienia zawierającego 2^{n+p} różnych stanów. Znajdują się w nim zawsze sekwencje stanów, które umożliwiają przy minimalnej liczbie taktów zegarowych osiągnięcie bardzo wysokiego współczynnika WPU w testowanym US. Są wśród nich takie sekwencje, które pozwalają uzyskać minimalne maskowanie błędów lub umożliwiają jego likwidację. W ten sposób realna staje się możliwość uzyskania maksymalnej efektywności testowania w przypadku każdego układu sekwencyjnego.

Niestety podstawową wadą opisanych struktur BIST-DFF jest dodatkowy nadmiar układowy związany z koniecznością podwojenia liczby przerzutników w bloku pamięci. Zagadnieniem zmniejszenia nadmiaru układowego struktury BIST-DFF zajmowali się Escherman i Wunderlich w pracach [EschW91, EschW92]. Zauważyli oni, że część przejść funkcji f_s , pokrywa się z przejściami funkcji rejestru LFSR włączonego w magistralę stanu. Wykorzystali więc ten rejestr do generacji tych przejść również w normalnej pracy systemowej. Ilustruje to rys. 1.9 [Hławi97b]. Dzięki temu zmniejszyli koszt realizacji zmodyfikowanej funkcji wzbudzeń f_s' bloku pamięci układu sekwencyjnego. Sygnał przełączania P (funkcja f_p) powoduje, że rejestr LFSR w miejsce układu realizującego funkcje wzbudzeń f_s' przejmuje odpowiedzialność za tworzenie stanów następnych ($P=1$) podczas normalnej systemowej pracy US. Zagadnieniem doboru takiej struktury sprzężenia liniowego rejestru LFSR, aby jak największa część przejść funkcji f_s , pokrywała się z przejściami funkcji rejestru LFSR, zajmowano się w [Garbo95]. W pracy tej wybrano do tego celu specjalną strukturę sprzężenia liniowego rejestru LFSR zawierającą także przerzutniki T [Hławi92a, Hławi96b].

Ze względu na wykorzystanie rejestru LFSR jako generatora paternów (PATern) traktowanych jako następne stany układu sekwencyjnego obie przedstawione na rys. 1.9 struktury samotestowania nazwane zostały w pracach [EschW91, EschW92], a potem w [Hławi97b] strukturami BIST-PAT.



Rys. 1.9. Dwie struktury testera wewnątrzukładowego typu BIST-PAT
Fig. 1.9. Two schemes of a BIST-PAT type built-in circuit tester

Struktury BIST-DFF oraz struktura BIST-PAT posiadają również pewne wady, na przykład mogą wystąpić sytuacje, w których niewykrywalne będą niektóre uszkodzenia opóźnieniowe (delay faults). Spowodowane jest to tym, że stan następny jest monitorowany w oddzielnym rejestrze MISR, który nie jest używany w pracy systemowej.

Żadna z przedstawionych struktur wbudowanego samotestowania nie może być preferowana dla uniwersalnego stosowania. Podobnie nie można wydzielić klasy rejestrów liniowych jako zalecanego standardu dla samotestowalnych układów sekwencyjnych i systemów cyfrowych. Najlepszym tego potwierdzeniem są prace Lempela i Gupty poświęcone algorytmom określania sprzężeń liniowych dla generacji zadanych testów deterministycznych [LempG94] oraz kompaktacji liniowej bez maskowania założonych błędów [LempG95]. Dla każdego indywidualnego układu sekwencyjnego należy wybrać tę strukturę BIST, która umożliwi uzyskanie możliwie wysokiego współczynnika pokrycia uszkodzeń przy niedużych dodatkowych kosztach sprzętu i stosunkowo krótkim czasie poświęconym na testowanie. Jest to problem tzw. krótkiej koldry, który dla różnych układów sekwencyjnych może być rozwiązany w sposób optymalny przy użyciu różnych schematów testerów wewnątrzukładowych wykorzystujących rejestry liniowe LFSR, SISR oraz MISR o odmiennych długościach i rozmaitych sprzężeniach liniowych. Skuteczną metodą umożliwiającą wybór struktury samotestowania jest symulacja komputerowa przy użyciu pakietów programów do symulacji

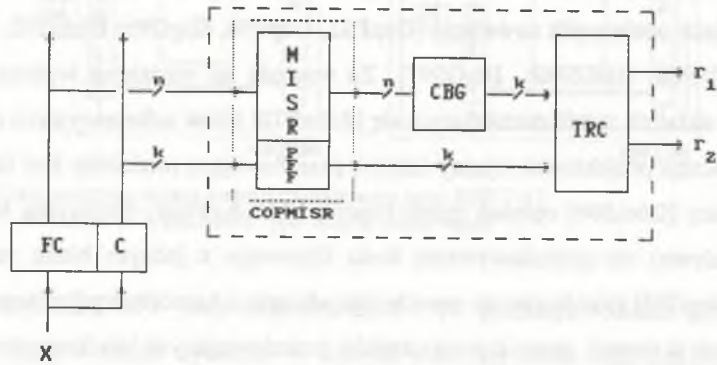
uszkodzeń, generacji testów oraz analizy odpowiedzi symulowanych testowanych układów cyfrowych. Rozwiązania optymalne wymagają specjalistycznych narzędzi wspomagających projektowanie rejestrów liniowych zintegrowane z narzędziem wspomagającym projektowanie zadanych układów scalonych.

1.3.3. Rejestry liniowe do współpracy z układami samokontrolującymi się

Idea projektowania rejestrów liniowych wykorzystywanych do współpracy z układami samokontrolującymi się [AshaR78] od kilku lat jest tzw. "gorącym problemem" w niektórych laboratoriach naukowych na świecie [GupP92, GupP96, SogG95, HłGS95, GoesS96, SogG96, HłGS96a, HłGS96b, HłGS97]. Ze względu na rozdzielne wykorzystywanie w klasycznym układzie samokontrolującym się [AshaR78] bitów informacyjnych oraz bitów kontrolnych zaczęto projektować rejestry liniowe przechowujące rozdzielny kod liniowy. Na przykład w pracy [GoesS96] opisano rejestr liniowy PMISA (Parity-Preserving Multi-Input Signature Analyser) do przechowywania kodu liniowego z jednym bitem parzystości, natomiast w [SogG96] przedstawiono metodę uzupełniania n-komórkowych rejestrów MISR tzw. korektorem w postaci grupy k przerzutników przechowujących bity kontrolne rozdzielonego liniowego kodu podwojonego. Rejestry liniowe przechowujące dowolny kod Hamminga należący do grupy rozdzielnych kodów liniowych opisano po raz pierwszy w [HłGS95]. Ideę tę rozszerzono w [HłGS96a, HłGS96b], a następnie w [HłGS97] na dowolne rozdzielne kody liniowe, np. kod SEC-DED Hsiao (zmodyfikowany kod Hamminga), kod podwojony czy też kod z bitami parzystości grupowej, a rejestry liniowe przechowujące te kody nazwano rejestrami COPMISR (COde Preserving MISR). Rejestr ten złożony jest z n-bitowego rejestru MISR oraz bloku PFF (Parity Flip Flops) zawierającego k przerzutników D pamiętających bity kontrolne.

Technikę wykorzystania rejestru COPMISR do budowy samotestowalnego układu samokontrolującego się przedstawia rys. 1.10. Na rysunku tym do wejść konwencjonalnego układu samokontrolującego się złożonego z generatora bitów kontrolnych CBG (Check-Bit Generator) i dwutorowego układu kontrolującego TRC (Two-Rail Checker) dołączono rejestr COPMISR. System cyfrowy z układem FC (Functional Circuit) oraz z układem bitów kontrolnych C w pewnych sytuacjach [SogG96] zachowuje się tak, że tylko nieduży podzbiór słów kodu liniowego pojawia się na wyjściach tych układów. Wówczas schemat konwencjonalnego układu samokontrolującego się (bez rejestru

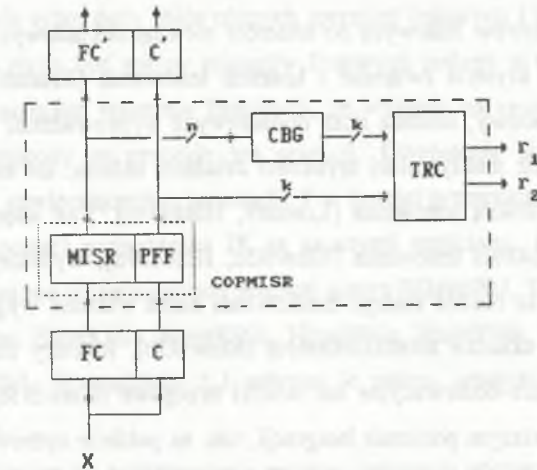
COPMISR) nie zostanie całkowicie przetestowany, a więc nie będzie on posiadał cech samotestowalności. Jeżeli w takiej sytuacji byłby podłączony do wyjść FC oraz C układ przedstawiony na rys. 1.10, wtedy nieuszkodzony rejestr COPMISR, zasilany na swoich wejściach co najmniej jednym wektorem należącym do kodu liniowego, wygeneruje na swoich wyjściach po pewnej liczbie taktów zegarowych wszystkie słowa kodowe. Wystarczy to, aby układ samokontrolujący się stał się w ten sposób układem samotestownym w czasie normalnej pracy systemu zawierającego układ FC.



Rys. 1.10. Samotestowalny układ samokontrolujący się do testowania na bieżąco (concurrent testing)
Fig. 1.10. A self-testing checker for concurrent testing

Zastosowanie samotestowalnego układu samokontrolującego się do natychmiastowego wykrywania błędów w trakcie normalnej pracy systemu z układem FC również przedstawia rys. 1.10. Jeśli podczas takiej pracy w układzie przedstawionym na tym rysunku sygnalizowany jest błąd na wyjściach r_1, r_2 , wówczas układ FC wraz z układem C, rejestr COPMISR, układ CBG lub dwutorowy układ kontrolujący TRC mogą być uszkodzone. Przy założeniu że tylko jeden z wymienionych zespołów może być uszkodzony, można je wszystkie wykorzystać do lokalizacji tego uszkodzonego bloku. Wówczas n-stopniowy rejestr MISR zostaje najpierw użyty jako analizator sygnatur. Jeżeli sygnatura jest poprawna, należy przyjąć, że rejestr MISR oraz układ FC są sprawne. Wtedy w dalszej normalnej pracy ignoruje się wskazania układu samokontrolującego się TRC. Jeżeli jednak sygnatura jest niezgodna z sygnaturą wzorcową, wówczas w kolejnych $2^n - 1$ taktach zegarowych wymusza się na wejściach układów FC oraz C stały wektor X. Wtedy też w $2^n - 1$ kolejnych taktach zegarowych wymuszany jest na wejściach rejestru COPMISR stały

wektor. Jeżeli po tym czasie sygnatura w rejestrze MISR równa jest stanowi początkowemu tego rejestru, to oznacza to, że uszkodzony jest układ FC. Jeżeli sygnatura jest niewłaściwa, to rejestr MISR jest niesprawny. Również w takiej sytuacji w dalszej normalnej pracy układu FC układ TRC nie jest wykorzystywany.



Rys. 1.11. Samotestowalny układ samokontrolujący się do testowania wewnątrzukładowego (off line testing)
Fig. 1.11. A self-testing checker for off-line testing

Rejestr COPMISR można wykorzystywać do realizacji testowania wewnątrzukładowego. Dodatkowe zastosowanie w takim testowaniu układu samokontrolującego się pozwala na jednoczesne wykorzystywanie zalet obu technik. Tę ideę przedstawia rys. 1.11. Błędy w trakcie testowania sygnalizują wyjścia r_1 oraz r_2 lub sygnatura rejestru COPMISR. Układ samokontrolujący się w trakcie testowania "off-line" umożliwia zarówno skracanie czasu testowania, jak i zwiększanie skuteczności tego testowania. W trybie normalnej pracy systemu FC rejestr COPMISR pracuje wyłącznie jako rejestr równoległy. Tylko w trybie testowania pełni on swoje funkcje kompaktora. Tak więc samotestowanie układu samokontrolującego się odbywa się tylko w trybie testowania "off-line". Rejestr COPMISR przechodzący zmodyfikowany kod Hamminga (kod SEC-DED) można zastosować wraz z dwutorowym układem kontrolującym się TRC do budowy układu przeznaczonego do detekcji i korekcji błędów w pamięciach. To zastosowanie rejestru COPMISR szeroko opisano w [HłaGS97].

1.4. Niektóre zagadnienia związane z wyborem, projektowaniem i stosowaniem rejestrów liniowych w testowaniu układów cyfrowych

Wprowadzenie rejestrów liniowych do testerów wewnątrzukładowych pozwala spełnić nie tylko takie istotne kryteria związane z kosztem testowania [Kraśn89, Hławi93b], jak niewielki nadmiar układowy, nieduża liczba dodatkowych wyprowadzeń, mały pobór mocy czy też niewielki spadek maksymalnej szybkości działania układu, ale także najważniejsze kryteria związane z jakością testowania [Kraśn89, Hławi93b]. Tak więc umożliwiają one uzyskiwanie dużej szybkości testowania [Hławi93a, Hławi97a], wysokiego współczynnika pokrycia uszkodzeń oraz bardzo małego maskowania błędów wskutek wygenerowania takiej samej sygnatury co w układzie nieuszkodzonym [Hławi93b]. Rejestry liniowe wkomponowane w ścieżki sterująco-obszaryjne lub ścieżki brzegowe [Hławi93b] mogą być także wykorzystywane na wyższym poziomie integracji, tzn. na pakiecie cyfrowym do testowania połączeń między układami zamocowanymi na pakiecie. Pozwalają także uzyskiwać wysoką rozdzielczość uszkodzeń [Hławi87b, Hławi89e].

Można w łatwy sposób modyfikować istniejące już rejestry liniowe, dostosowując je do wprowadzanych w trakcie prototypowania zmian logiki funkcjonalnej. Dzięki temu ich wrażliwość na przeróbki logiki funkcjonalnej jest bardzo mała. Ze względu na małą liczbę różnych komórek występujących w strukturze rejestrów liniowych ich projektowanie jest podatne na automatyzację [Hławi93a, Hławi97a]. Również złożoność procesu projektowania rejestrów liniowych jest nieduża [Hławi93b]. Mogą być one efektywnie stosowane w wielu klasach układów cyfrowych [Hławi93b], a ich wykorzystywanie w testowaniu wewnątrzukładowym, z wyjątkiem struktur BIST-PST [Hławi97b] oraz BIST-PAT [Hławi97b], nie wymaga modyfikacji testowanego układu cyfrowego.

Celem optymalizacji syntezy testerów wewnątrzukładowych i testowanych układów cyfrowych jest znaczące zredukowanie kosztów samotestowalności i uzyskanie wysokiej jakości testowania. Najlepsze dopasowanie rejestru liniowego do potrzeb związanych z taką optymalizacją wymaga właściwego doboru sprzężenia liniowego oraz jego struktury. Efektywne wykorzystywanie do realizacji wskazanego celu takich zasobów współczesnych programowalnych układów FPGA, jak konfigurowalne moduły logiczne CLB (Configurable Logic Block) i uniwersalne sieci połączeń pomiędzy nimi [Hławi97a] również wymaga

należytej selekcji sprzężeń liniowych i ich struktur [HławiB94a, HławiB94b, Hławi97a]. Jak najlepsze ich dopasowania do reguł projektowania obowiązujących w wybranym układzie FPGA pozwala uzyskać wysoki współczynnik wydajności (performance) układu [Hławi97a].

Swobodę w optymalnym wyborze, pozwalającym spełnić większość z wymienionych kryteriów, umożliwia tylko duży zbiór różnych sprzężeń liniowych i ich struktur. Przykładem takiego zbioru może być zestaw rejestrów liniowych podany w tabeli 3.2. Zbiór klasycznych struktur sprzężeń rejestrów liniowych, ze względu na swoją małą liczbę, nie zapewnia takiej swobody w procesie ich selekcji. Gwarantują ją natomiast struktury sprzężeń liniowych zawierających przerzutniki T w postaci przerzutnika D z bramką XOR na wejściu lub w postaci przerzutnika JK ze zwartymi wejściami. Opisy takich struktur liniowych zawarte są we wcześniejszych pracach autora [Hławi90d, Hławi91b, HławiK92a, Hławi92a, Hławi93a, HławiK93a, HławiK93b, HławiB94a, HławiB94b, HławiKC96, Hławi96b, Hławi97a, Hławi97b]. W rozdziale 2.1 zebrano je razem, uzupełniono i przedstawiono w jednolitej formie.

Przed przystąpieniem do projektowania rejestru liniowego istotne jest określenie wielomianu charakterystycznego, który jest wyznacznikiem macierzy równań [Hławi96b] opisujących pracę wszystkich stopni tego rejestru. Wielomian ten jest wspólną cechą charakterystyczną wielu różnych struktur sprzężenia rejestru liniowego. Struktury wynikające z tego samego wielomianu charakterystycznego mogą różnić się między sobą liczbą bramek XOR, czasem propagacji sygnału pomiędzy sąsiednimi przerzutnikami, liczbą połączeń w sprzężeniu zwrotnym, obciążalnością wyjść przerzutników, rozkładem przerzutników T wzdłuż rejestru liniowego itp. Określenie tych różnic oraz innych właściwości przydatnych w samotestowaniu wymaga znajomości algebraicznego opisu pracy rejestrów liniowych. Opis taki opracowany na bazie pierścienia wielomianów nad ciałem $GF(2)$ i na podstawie poprzednich prac autora [Hławi84c, Hławi85a, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi90a, Hławi92a, Hławi92b, Hławi93b, Hławi96b] przedstawiono w postaci dzielenia wielomianów w rozdziale 2.2. Umożliwił on opracowanie jednego wspólnego schematu zastępczego dla rejestrów liniowych o różnych sprzężeniach i różnych ich strukturach. Schemat, ten również opisany w rozdziale 2.2, umożliwia zastąpienie grafu przejść w procesie określania stanu końcowego (sygnatury) kompaktorów liniowych SISR, TISR oraz MISR przy znanym ich stanie początkowym i znanej sekwencji wektorów wejściowych. Pozwala także określać stan początkowy, który należy wprowadzić do rejestru

liniowego, aby po podaniu znanego ciągu wektorów wejściowych uzyskać pożądaną sygnaturę. Umożliwia ponadto określanie sekwencji wejściowej, która ustawia w rejestrze liniowym pożądaną sygnaturę przy założonym dowolnym stanie początkowym. Wszystkie te zagadnienia, bardzo przydatne w procesie projektowania kompaktorów liniowych o różnych właściwościach, opisano w rozdziale 2.3. Rozdział ten stanowi powiązanie, rozwinięcie i uściślenie odpowiednich fragmentów poprzednich prac autora [Hławi84c, Hławi86c, Hławi87a, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi89e, Hławi90a, Hławi92b]. W rozdziale tym podano także technikę konwersji sygnatur w przypadku kompaktorów MISR o różnych strukturach sprzężeń liniowych, ale opisanych tym samym wielomianem charakterystycznym.

Strukturę oraz opis algebraiczny rejestrów liniowych COPMISR (COde Preserving MISR) przechowujących wyłącznie słowa kodowe należące do rozdzielnego kodu liniowego C (kodu Hamminga, kodu SEC-DED Hsiao, kodu podwojonego, kodu z bitami parzystości grupowej itp.) opisano w rozdziale 2.4. Przydatność tych rejestrów liniowych do równoległego stosowania wraz z układami samokontrolującymi się do testowania na bieżąco (concurrent testing) lub testowania wewnątrzukładowego (BIST) została opisana we wcześniejszych pracach autora [HłGS96a, HłGS96b, HłGS97].

Cały rozdział 3 poświęcono omówieniu różnych technik projektowania rejestrów liniowych o strukturach zawierających przerzutniki T. W szczególności w rozdziale 3.1 przedstawiono uniwersalną technikę projektowania rejestrów LFSR, SISR i MISR o różnych strukturach sprzężeń liniowych wynikających z założonego źródłowego wielomianu charakterystycznego. Technika ta jest pewnym rozwinięciem fragmentów uprzednio publikowanych prac autora [Hławi89a, Hławi89b, Hławi92a, Hławi92b]. Pozwala ona zrezygnować z korzystania z niewygodnych i często niedostępnych tabel zawierających wyłącznie klasyczne konstrukcje rejestrów liniowych o sprzężeniach opisanych różnymi wielomianami charakterystycznymi.

Metody projektowania rejestrów LFSR, SISR oraz MISR zawierających mniejszą liczbę bramek XOR w sprzężeniu, niż wymagają tego rejestry z tradycyjnie stosowanymi strukturami sprzężeń liniowych, przedstawiono w rozdziale 3.2. Tylko pewna część rozdziału opiera się na wynikach z wcześniejszych prac autora [Hławi89a, Hławi89b, Hławi92a, Hławi92b, Hławi97a]. Wszystkie idee zawarte w podrozdziałach 3.2.1 oraz 3.2.4 nigdzie dotąd nie były publikowane.

Projektowanie na podstawie założonego wielomianu charakterystycznego rejestrów liniowych o strukturze komórkowej CA (Cellular Automata based register) jest znacznie trudniejszym zadaniem niż projektowanie rejestrów o pozostałych strukturach sprzężeń liniowych zawierających przerzutniki T. Dlatego też autor w pracach [Hławi90d, Hławi91b, HławiK92a, Hławi93a, Hławi93b, HławiK93b, HławiKC96] zajmował się utworzeniem różnych bibliotek kilkukomórkowych modułów plasterkowych umożliwiających ich proste łączenie w jeden długi komórkowy rejestr CA o założonych cechach wielomianu charakterystycznego opisującego jego sprzężenie liniowe. W rozdziale 3.3 niektóre rezultaty tych prac ujednolicono, uzupełniono i powiązano w jedną całość.

W rozdziale 3.4 omówiono zagadnienie wyboru łatwo modyfikowalnych rejestrów liniowych, minimalnie wrażliwych na wprowadzane w trakcie prototypowania przeróbki logiki funkcjonalnej testowanego układu cyfrowego i pozwalających w związku z tym na ich proste, a więc tanie dostosowanie się do tych zmian. Problem ten tylko częściowo autor omawiał wcześniej w pracach [Hławi90d, HławiK92a].

Technikę projektowania rejestrów liniowych COPMISR dla kodu z bitami parzystości grupowej oraz zmodyfikowanego kodu Hamminga przedstawiono w rozdziale 3.5. Technika ta w porównaniu z poprzednimi pracami autora [HłGS96a, HłGS96b, HłGS97] umożliwia również projektowanie rejestrów COPMISR o strukturach zawierających przerzutniki T.

Rozdział 4 omawia problemy związane z wykorzystaniem rejestrów liniowych do kompaktacji odpowiedzi testowanych układów cyfrowych zarówno w testerach wewnątrzukładowych, jak również przy użyciu analizatorów i weryfikatorów sygnatur. Rozdział ten jest usystematyzowanym połączeniem fragmentów poprzednich prac autora [HławiK81b, Hławi84a, Hławi84b, Hławi84c, HławiN85b, Hławi86a, Hławi86b, Hławi86c, Hławi86d, Hławi86f, Hławi87b, Hławi87c, Hławi88a, Hławi88b, Hławi88c, Hławi88b, Hławi89a, Hławi89b, HławiM89, Hławi89e, Hławi89g, Hławi90a, Hławi90d, Hławi91a, Hławi91b, HławiK92a, Hławi92a, Hławi92b, Hławi93b, HławiK93a, Hławi96b], a także ich uzupełnieniem i rozwinięciem. Algebraiczny opis kompaktacji liniowej oraz zasady analizy (weryfikacji) sygnaturowej przedstawia podrozdział 4.1. Określeniem kryteriów umożliwiających wybór sprzężeń gwarantujących minimalne maskowanie błędów w procesie kompaktacji zajmuje się podrozdział 4.2. Opisano w nim różne przyczyny maskowania błędów, następnie zdefiniowano miary tego maskowania, a w końcu wykorzystano je do wskazania

sprzężeń liniowych, które najlepiej nadają się do zastosowania w kompakcji. Podrozdział 4.3 poświęcony jest opisowi różnych technik zmniejszania maskowania błędów w procesie kompakcji liniowej, co w efekcie wpływa na zwiększenie skuteczności analizy sygnaturowej. W rozdziale tym przedstawiono także metodę zupełnej likwidacji maskowania błędów.

W rozdziale 5 znajduje się podsumowanie istotnych rezultatów pracy.

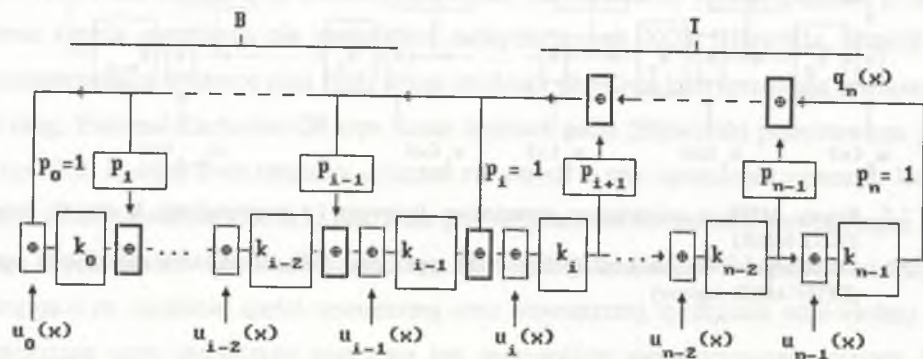
2. STRUKTURY I OPIS ALGEBRAICZNY REJESTRÓW LINIOWYCH

Rejestry liniowe są sekwencyjnymi układami liniowymi [Elsa59, Kautz65, Gill67, Golom82]. W tej pracy rejestrami liniowymi nazywać będziemy każdy z niżej podanych typów rejestrów przesuujących o sprzężeniu liniowym zrealizowanym przy użyciu bramek XOR i/lub przerzutników T (przerzutnik D z bramką XOR) i opisanym za pomocą charakterystycznego wielomianu $p(x)$ [PetW94, Golom82, RaoF89, Hławi96b]. Będziemy rozróżniać nie posiadające żadnych wejść n -bitowe rejestry liniowe LFSR (ang. Linear Feedback Shift Register) [PetW94, Golom82], a także jednowejściowe n -bitowe szeregowo rejestry liniowe SISR (ang. Single Input Signature Register) [PetW94, Frohw77, Hławi93b] oraz wielowejściowe n -bitowe rejestry liniowe MISR (ang. Multi-Input Signature Register) [KoneM79, KoneM80, Hassa82, SridH82, HassL83, Hławi84c, Hławi85a, Hławi86c, Hławi93b, Hławi96b] nazywane potocznie równoległymi rejestrami liniowymi. Wśród tych ostatnich został wyodrębniony przez autora dwuwiejściowy rejestr TISR (ang. Two-Input Signature Register) [Hławi84c, HławiN84, HławiN85b, Hławi86c, BaduH88a], rejestr liniowy z wieloma parami wejść MTISR (ang. Multi-Two-Input Signature Register) [Hławi84a, Hławi85b, Hławi86f, MitaH88], a także pochodne wielowejściowe rejestry MISR [Hławi87c, Hławi88a] umożliwiające przyłączanie do nich układów cyfrowych z liczbą wyjść znacznie przekraczającą liczbę n komórek rejestru MISR. Rejestr LFSR oraz rejestr SISR są szczególnymi przypadkami rejestru MISR.

Dotąd przyzwyczajeni byliśmy do tego, że założony wielomian charakterystyczny $p(x)$ w jednoznaczny sposób opisywał strukturę sprzężenia liniowego rejestru. Tak jest zawsze w przypadku rejestru z powszechnie znaną wewnętrzną strukturą sprzężenia liniowego [PetW94, BhavH81, SridH82, Golom82, Hassa82, HassL83, HassM83] lub strukturą typu FSR (ang. Feedback Shift Register) w postaci pętli sprzężenia nie zawierającej żadnych

przedstawionych na omówionych poprzednio rysunkach. Pozostawiając w rejestrze MISR tylko dwa wejścia określone na zasadach podanych w [Hławi84c, HławiN84, HławiN85, Hławi86c, BaduH88a], otrzymujemy rejestry liniowe typu TISR. Grupując wejścia rejestru MISR parami według określonych w [Hławi84a, Hławi85b, Hławi85f, MitaH88] zasad, uzyskamy rejestry MTISR. Jeżeli przedstawione struktury sprzężeń nie będą zawierały bramek XOR utajonych w przerzutnikach T, wówczas będziemy mieli do czynienia ze znanymi z literatury lat wcześniejszych rejestrami zawierającymi wyłącznie przerzutniki D i z następującymi sprzężeniami:

- wewnętrznym [PetW94, Golom82, Hassa82, HassL83, HassM83, Hławi87c, Hławi88a, Hławi96b],
- zewnętrznym [Froh77, KoneM79, KoneM80, Hławi84c, Hławi85a, Hławi86c, Hławi87c, Hławi88a, Hławi96b],
- mieszanym [WangM86b, WangM86c, WangM88, Hławi89a, Hławi89b, Hławi90a, Hławi92b, Hławi96b].

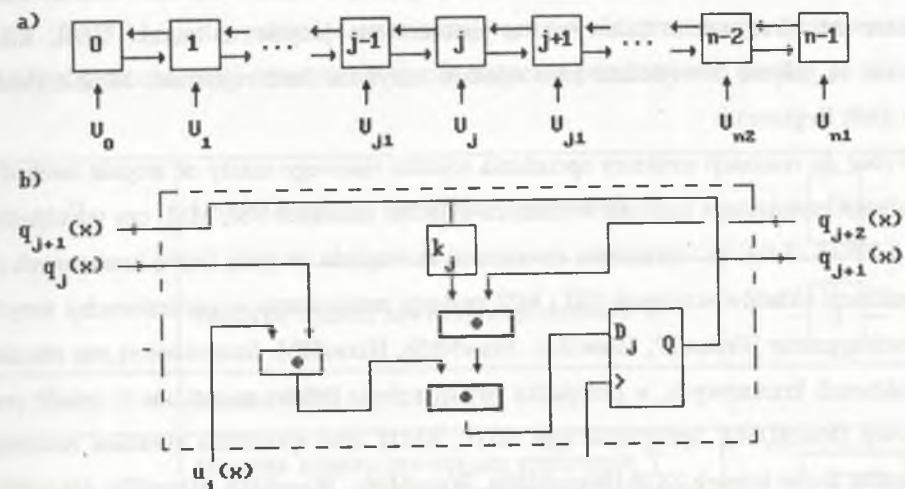


Rys. 2.4. Rejestr MISR z wewnętrznym-zewnętrznym sprzężeniem liniowym oraz z przerzutnikami D oraz T (rejestr BTDT-MISR)

Fig. 2.4. MISR register with Bottom-Top Exclusive OR type linear feedback and with flip-flops D and T (BTDT-MISR register)

W pracach [HortM89, HortM90, SerrS90, Hławi90d, HławiK92a, Hławi96b] przedstawiono także strukturę liniowego sprzężenia zwrotnego bazującą na jednowymiarowych hybrydowych automatach komórkowych CA (ang. Cellular Automata). Na rys. 2.5. przedstawiono schemat blokowy n -komórkowego rejestru MISR typu CA oraz schemat ideowy jego i -tej komórki, w której $k_i=1$ oznacza obecność sprzężenia, natomiast $k_i=0$ oznacza brak sprzężenia. Komórki te są symbolicznie oznaczane przez 1 i 0. Znane są one

także jako komórki realizujące prawa odpowiednio 150 i 90 [HortM89, HortM90, SerrS90]. Wprowadzając do realizacji zarówno przerzutniki D, jak i T można komórki 1 realizować wprost za pomocą przerzutników T, natomiast komórki 0 za pomocą przerzutników D. Rejestry LFSR typu CA oraz SISR typu CA są konstruowane podobnie, jak wyjaśniono to już wcześniej.



Rys. 2.5. Schemat blokowy rejestru komórkowego MISR typu CA (rejestr CADT-MISR) (a) oraz schemat ideowy j -tej komórki tego rejestru (b)

Fig. 2.5. Scheme of Cellular Automata based - MISR register (CADT-MISR register) (a), circuit diagram of j -th cell (b)

W celu odróżnienia od siebie rejestrów liniowych o różnych strukturach sprzężeń liniowych, ale tym samym wielomianie charakterystycznym $p(x)$, wprowadzono następujące oznaczenia c-LFSR, c-SISR, c-TISR, c-MISR, gdzie: c jest symbolem oznaczającym parę oznaczeń dwuliterowych $\{xy/ab\}$. Para $xy \in \{IE, EE, TB, BT, CA, -\}$ określa wcześniej wyjaśniony rodzaj podstawowej struktury sprzężenia liniowego. Element pusty "-" oznacza brak sprzężenia liniowego. Para $ab \in \{DT, D-, -T\}$ określa natomiast rodzaj użytych w rejestrze przerzutników D i/lub T. W przypadku symbolu T jest to informacja o wprowadzeniu do struktury przerzutnika T (np. przerzutnika JK ze zwartymi wejściami albo przerzutnika D z bramką XOR na wejściu). W efekcie $c \in \{IEDT, IED, IET, EEDT, EED, EET, DT, D, T, BTDT, LTD, TBT, TBD, CADT\}$. Tak więc rejestry LFSR, SISR oraz MISR w zależności od rodzaju zastosowanej struktury liniowego sprzężenia zwrotnego




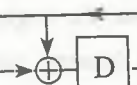

można oznaczać w prosty następujący sposób: np. IEDT-LFSR, IEDT-MISR, EED-LFSR, EED-TISR, EEDT-MISR, TBD-LFSR. Rejestry liniowe ze strukturą typu FSR oznaczane będą przez DT-MISR (DT-SISR, DT-LFSR), jeśli złożone są z przerzutników D oraz T lub przez D-MISR (D-SISR, D-LFSR) oraz T-MISR (T-SISR, T-LFSR), jeśli zawierają odpowiednio tylko przerzutniki D lub tylko przerzutniki T. Rejestry MISR, z których usunięto całkowicie sprzężenie liniowe (pętlę sprzężenia oraz wszystkie bramki XOR związane ze sprzężeniem liniowym), a pozostawiono jedynie te bramki XOR, które związane są jedynie z wejściami tego rejestru, nazywane będą rejestrami MIShR (Multi Input Shift Register).

Wybór do realizacji struktury sprzężenia rejestru liniowego zależy od stopnia swobody, jaki oferuje architektura logiczna wybranych układów scalonych SSI, MSI, czy też układów PLD i FPGA. I tak np. sprzężenia zewnętrzne ze względu na małą liczbę koniecznych do ich realizacji układów scalonych SSI i MSI znalazły zastosowanie w analizatorach i weryfikatorach sygnałów [Froh77, Hławi84c, HławiN85b, Hławi88b]. Stosowane są one również w strukturach krzemowych, w przypadku gdy sprzężenia liniowe są ustalane w sposób programowy [MuchD86]. Sprzężenia typu IEDT, EEDT oraz sprzężenia mieszane zawierają minimalną liczbę bramek XOR [WangM86b, WangM86c, WangM88, Hławi89a, Hławi89b, Hławi90a, Hławi92b, Hławi97a], co pozwala zwiększać uzysk przy ich implementacji w strukturze krzemowej układów ASIC projektowanych metodą "semi custom", a w szczególności projektowanych metodą "full custom". Rejestry LFSR, SISR i MISR typu CA ze względu na swoją regularną i topologicznie podobną komórkową architekturę oraz bezpętlową realizację najlepiej nadają się do automatyzacji projektowania układów ASIC wyposażonych w brzegową ścieżkę sterująco-obszewacyjną [GlosB89]. W szczególności nadają się one do projektowania układów FPGA firmy XILINX. Wreszcie rejestry zbudowane z przerzutników D oraz T doskonale się nadają do stosowania w niektórych układach FPGA (np. ACTEL) [Hławi97a] oraz szczególnie w tych układach PLD oraz FPGA, w których jest możliwy równoprawny dostęp do przerzutników D oraz T (np. EXEL czy też ALTERA) [HławiB94a, HławiB94b].

Różne konfiguracje sprzężeń rejestrów liniowych mogą być też stosowane w zależności od wymagań, jakie stawia projektowany układ cyfrowy lub projektowany wraz z nim układ BIST. Ten i poprzedni problem szerzej omówiono w rozdziałach 3,4 i 5.

2.1.1. Uproszczony sposób zapisu struktur rejestrów liniowych

Próbie uproszczenia zapisu schematów ideowych rejestrów TBD oraz BTD przedstawiono w pracy [WangM88]. Ten sposób zapisu w odczuciu autora jest mało czytelny i jest ograniczony jedynie do rejestrów o hybrydowych sprzężeniach liniowych. Inny sposób uproszczonego zapisu rejestrów liniowych pozbawiony tych wad przedstawiono w [Hławi92a, Hławi93a]. Ten sposób po jego rozszerzeniu na rejestry liniowe EEDT, TBTD, BTDT, CADT i rejestry pochodne wykorzystuje autor w niniejszej pracy. Związany z nim jednolity, jednoznaczny i uproszczony system oznaczeń przedstawiono poniżej.

D	komórka rejestru zawierająca przerzutnik D	
T	komórka rejestru zawierająca przerzutnik T	
\oplus	bramka XOR	
$(\oplus'D)$	komórka rejestru zawierająca przerzutnik T zrealizowany na bazie przerzutnika D z bramką XOR na wejściu	
$D^k, T^k, (\oplus'D)^k$	ciąg k komórek D, T oraz $(\oplus'D)$ nie przedzielonych bramką XOR oraz inną komórką	$D^k = DDD\dots D\dots D;$ $T^k = TTT\dots T\dots T;$
$\oplus D, \oplus T$	komórka D, T zawierająca bramkę XOR sprzężenia liniowego umieszczoną wewnątrz rejestrów o podstawowej konfiguracji sprzężenia liniowego typu IE, TB oraz BT	
\oplus_D, \oplus_T	komórka D, T zawierająca bramkę XOR sprzężenia liniowego umieszczoną na zewnątrz rejestrów o podstawowej konfiguracji sprzężenia liniowego typu EE, TB oraz BT	

1	komórka wewnętrzna oraz komórka lewo- (pravo-) stronna typu $(\oplus'D)$ lub T w rejestrze CADT	
0	komórka wewnętrzna oraz komórka lewo- (pravo-) stronna typu D w rejestrze CADT	
D,T,0,1	komórki odpowiednio D,T,0 oraz 1 zawierające dodatkową bramkę XOR umożliwiającą przyłączenie do rejestru c-SISR lub c-MISR linii wejściowych	

Sposób wykorzystania powyższego systemu oznaczeń przedstawia przykład:

Przykład 2.1

rejestr IEDT-LFSR;	n = 9:	DDDDTT \oplus DDD, $D^4(\oplus'D)^2\oplus D^3$, $D^4T(\oplus'D)\oplus D^3$,
rejestr EEDT-LFSR;	n = 9:	DDD \oplus TTDDDD, $D^3\oplus(\oplus'D)^2D^4$, $D^3\oplus(\oplus'D)TD^4$,
rejestr EED-TISR;	n = 6:	DD\oplusD\oplusDD\oplusD
rejestr DT-LFSR;	n = 8:	DDDDTTT, $D^5T(\oplus'D)^2$, $D^5(\oplus'D)T^2$
rejestr BT-D-SISR;	n = 7:	D\oplusDD\oplusDDD\oplusD ,
rejestr TB-D-SISR;	n = 9:	D\oplusDDDDDD\oplusD\oplusDD ,
rejestr CADT-MISR;	n = 5:	01101

Kolejne stopnie występujące w rejestrach liniowych narastają od lewej do prawej strony zgodnie z przedstawionymi w tym rozdziale schematami i uproszczonymi sposobami ich zapisu. Ze względu jednak na odwrotny do tego kierunku przyjęty w następnych rozdziałach binarny zapis stanów wewnętrznych i odpowiadających im wielomianów należy zwrócić uwagę na konieczność właściwej korelacji kolejnych bitów stanów wewnętrznych z kolejnymi komórkami rejestrów liniowych. Prawidłowy sposób polega na tym, że najmłodszy (najstarszy) bit słowa binarnego znajdujący się w związku z przyjętym zapisem po prawej (lewej) stronie tego słowa należy korelować z komórką schematu rejestru liniowego znajdującą się po lewej (prawej) stronie tego schematu.

Umówmy się, że ciągi znaków opisujących schemat ideowy rejestrów liniowych oznaczać będziemy przez R_n , gdzie n oznacza liczbę stopni danego rejestru.

Definicja 2.1

Rejestry liniowe o różnych strukturach sprzężeń liniowych c, ale takim samym wynikającym z tych sprzężeń wielomianem charakterystycznym $p(x)$ nazywać będziemy rejestrami c-LFSR, c-SISR, c-MISR związanymi wielomianem charakterystycznym $p(x)$, w skrócie rejestrami związanymi c-LFSR- $p(x)$, c-SISR- $p(x)$ oraz c-MISR- $p(x)$.

Rejestry liniowe związane z wielomianami pierwotnymi nazywać będziemy rejestrami pierwotnymi.

Rejestry c-LFSR, c-SISR lub c-MISR określone ciągiem znaków R_n i związane wielomianem charakterystycznym $p(x)$ zapisywać będziemy jako $R_n \cong p(x)$. W przypadku zbioru $\{R_n\}$ będziemy zapisywali $\{R_n\} \cong p(x)$.

Przykład 2.2

Zbiór rejestrów c-LFSR- $p(x)$:

$$\{D\oplus DD, DD\oplus D, DTT, TDT, TTD, 011, 110\} \cong p(x) = 1 + x + x^3$$

Zbiór rejestrów c-MISR- $p(x)$:

$$\{D\oplus DD, DD\oplus D, DTT, TDT, TTD, 110, 011\} \cong p(x) = 1 + x + x^3$$

2.2. Algebraiczne opisy pracy rejestrów liniowych i różnice w nich występujące

Algebraiczny opis pracy rejestrów liniowych umożliwia dokładną analizę pracy tych rejestrów i wnikliwe zbadanie różnic w nich występujących. Wymaga to jednak wprowadzenia odpowiedniej algebry.

2.2.1. Zastosowana algebra i niektóre pojęcia

Dwuelementowym ciałem Galois (Galois Field) $GF(2)$ [RaoF89, PeteW94] nazywany jest zbiór dwóch elementów $\{0,1\}$ wraz z sumą modulo dwa "+" oraz iloczynem modulo dwa "." zdefiniowanymi następująco:

+	0	1	•	0	1
0	0	1	0	0	0
1	1	0	1	0	1

Wielomianem nad ciałem GF(2) [RaoF89, PeteW94] nazywane jest wyrażenie w postaci $f(x) = f_0x^0 + f_1x^1 + f_2x^2 + \dots + f_ix^i + \dots + f_{m-1}x^{m-1} + f_mx^m$, gdzie $f_i \in \{0,1\}$ są elementami ciała GF(2), x jest nieokreśloną zmienną, a indeksy i i wskaźniki są liczbami całkowitymi nieujemnymi.

Pierścieniem wielomianów nad ciałem GF(2) [RaoF89, PeteW94] nazywany jest zbiór

$$\text{wielomianów } f(x) \text{ wraz z sumą } f(x) + g(x) = \sum_{i=0}^m \oplus (f_i + g_i) x^i$$

$$\text{i iloczynem } f(x) g(x) = \sum_{i=0}^{2m} \oplus \left(\sum_{j=0}^i \oplus f_j g_{i-j} \right) x^i.$$

Znak "+" w definicjach ww. używany jest w trzech różnych znaczeniach: dodawanie wielomianów, dodawanie w ciele GF(2), symbol formalny w budowie wielomianu. Podobnie znak "." w definicji drugiej użyty jest w dwóch różnych znaczeniach: mnożenie wielomianów i mnożenie w ciele GF(2). Ponieważ wielomian $f(x)$ jest sumą wielomianów $f_0x^0, f_1x^1, f_2x^2, \dots, f_ix^i, \dots, f_mx^m$, dlatego symbol formalny w budowie wielomianów tożsamy jest dodawaniu wielomianów. Ponieważ jednocześnie dodawanie (mnożenie) wielomianów sprowadza się do dodawania (mnożenia) elementów ciała GF(2), dlatego używanie symboli "+" oraz "." w wielu różnych znaczeniach nie prowadzi do nieporozumień. Jednak w celu odróżnienia znaku \sum przypisywanego sumowaniu w arytmetyce dziesiętnej od dodawania opisanego wyżej wszędzie w pracy wprowadzono symbol $\sum \oplus$.

Jeżeli w wielomianie $f(x)$ współczynnik $f_m \neq 0$, to liczbę m nazywamy stopniem wielomianu. Stopień wielomianu $f(x)$ oznaczamy przez $\deg f(x)$.

W pierścieniu wielomianów nad ciałem GF(2) niemożliwe jest dzielenie w ogólnym przypadku. Jednakże dla wielomianów $f(x)$ w tym pierścieniu możliwe jest ich skracanie (np. jeżeli $f(x) = u(x)w(x)$, to $f(x)/w(x) = u(x)$) [RaoF89, PeteW94], a także dzielenie z resztą $r(x)$ (np. $f(x)/p(x) = q(x) + r(x)/p(x)$) [RaoF89, PeteW94].

Wielomianem pierwszym nazywa się taki wielomian $p(x)$ nad ciałem GF(2), który jest podzielny tylko przez wielomian $p(x)$ lub 1 będące elementem ciała GF(2).

Rzędem wielomianu pierwszego $p(x)$ nazywa się najmniejszą liczbę całkowitą k taką, że dwumian $x^k + 1$ jest podzielny przez $p(x)$.

Wielomianem pierwotnym nazywa się taki wielomian pierwszy $p(x)$ stopnia n , którego rząd $k = 2^n - 1$.

2.2.2. Algebraiczny opis pracy rejestrów c-MISR

W teorii kodowania ciągi mogą być traktowane jako wielomiany z fikcyjną zmienną x oraz współczynnikami należącymi do ciała GF(2) [PetW94, Golom82, RaoF89]. Za pomocą takich wielomianów opisywane są także wielomiany charakterystyczne sprzężeń rejestrów liniowych [PetW94, Golom82, RaoF89]. Przy użyciu pierścienia wielomianów nad ciałem GF(2) [RaoF89, Hławi87c, Hławi88a] przedstawiono algebraiczny opis pracy rejestrów IED-LFSR w [PetW94, Golom82], rejestrów IED-SISR oraz EED-SISR w [Smith80] oraz rejestrów IED-MISR w [Hassa82, HassL83]. Na tej podstawie stał się możliwy do określenia algebraiczny opis pracy wszystkich rejestrów c-MISR o dowolnych przedstawionych poprzednio strukturach liniowych sprzężeń zwrotnych c .

Poniższe dzielenie jest algebraicznym opisem pracy rejestrów c-MISR. Zostało ono określone przy użyciu pierścienia wielomianów nad ciałem GF(2) [Hławi84c, Hławi85a, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi90a, Hławi92a, Hławi92b, Hławi93b, Hławi96b]:

$$[w_c(x) + x^m h_c(x)] / p_c(x) = q_c(x) + r_c(x) / p_c(x) \quad (2.1)$$

gdzie:

$$w_c(x) = \sum_{r=0}^{n-1} \oplus u_r(x) p_{cr}(x) \quad \text{odzwierciedla ciąg wejściowy rejestru c-MISR;}$$

$\deg w_c(x) = m + n - 2$; $u_r(x)$ jest wielomianem określającym ciąg wejściowy na r -tym wejściu rejestru c-MISR; $\deg u_r(x) = m - 1$,

$$h_c(x) = \sum_{r=0}^{n-1} \oplus g_r p_{cr}(x) \quad \text{jest wielomianem zawierającym bity } g_r \text{ stanu początkowego}$$

rejestru c-MISR; $\deg h_c(x) = n - 1$,

$$q_c(x) \quad \text{jest wielomianem określającym iloraz dzielenia, który}$$

w postaci ciągu binarnego pojawia się na wyjściu ostatniego $(n-1)$ przerzutnika rejestru c-MISR; $\deg q_c(x) = m - 1$,

$$r_c(x) = \sum_{r=0}^{n-1} \oplus s_r p_{cr}(x) \quad \text{jest wielomianem określającym resztę z dzielenia}$$

i zawierającym bity s sygnatury (stanu końcowego rejestru c-MISR); $\deg r_c(x) = n - 1$,

$p_c(x)$ jest wielomianem charakterystycznym (dzielnikiem) rejestru c-MISR; $\deg p_c(x) = n$.

Wielomian $p_c(x)$ dla każdej struktury sprzężenia liniowego c jest taki sam i równy $p(x)$. Tak więc wielomian $p_c(x)$ można w dzieleniu (2.1) zastąpić dzielnikiem $p(x)$ niezależnym od struktury sprzężenia liniowego c. Jednakże ze względu na konieczność wykorzystywania różnych postaci wielomianu $p_c(x)$ [Hławi84c, Hławi85a, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi90a, Hławi92a, Hławi92b, Hławi93b, Hławi96b] do projektowania rejestrów o odmiennych strukturach sprzężeń liniowych c uporządkowano rozmaite postaci tych wielomianów i zebrano w tabeli 2.1.

Tabela 2.1

c	Postaci wielomianów $p_c(x)$ oraz $p_{cr}(x)$, $p_{cj}(x)$
IEDT	$p_{IEDT}(x) = p_0 + \sum_{r=1}^n \oplus p_r \prod_{j=0}^{r-1} (k_j + x)$ $p_{IEDTr}(x) = \prod_{j=0}^{r-1} (k_j + x)$ $p_{IEDTr} = 1 \quad \text{dla } r = 0$
IED	$p_{IED}(x) = \sum_{r=0}^n \oplus p_r x^r$ $p_{IEDr}(x) = x^r$
IET	$p_{IET}(x) = \sum_{r=0}^n \oplus p_r (1 + x)^r$ $p_{IETr}(x) = (1 + x)^r$
EEDT	$p_{EEDT}(x) = \sum_{i=0}^{n-1} \oplus p_i \prod_{r=i}^{n-1} (k_r + x) + p_n$ $p_{EEDTj}(x) = \sum_{i=0}^{j-1} \oplus p_i \prod_{r=i}^{j-1} (k_r + x) + p_j$
EED	$p_{EED}(x) = \sum_{i=0}^n \oplus p_i x^{n-i}$ $p_{EEDj}(x) = \sum_{i=0}^j \oplus p_i x^{j-i}$
EET	$p_{EET}(x) = \sum_{i=0}^n \oplus p_i (1 + x)^{n-i}$ $p_{EETj}(x) = \sum_{i=0}^j \oplus p_i (1 + x)^{j-i}$
BTD	$p_{BTD}(x) = x^n + \left[\sum_{r=0}^i \oplus p_r x^r \right] \left[\sum_{r=i+1}^n \oplus p_r x^{n-r} \right]$ $p_{BTDj}(x) = x^j + G(j) \sum_{r=0}^i \oplus p_r x^r \sum_{r=i+1}^j \oplus p_r x^{j-r}$ $G(j) = \begin{cases} 0 & \text{dla } j < i \\ 1 & \text{dla } j > i \end{cases}$

cd. tab. 2.1

BTD	$p_{BTD}(x) = x^n + \left[\sum_{r=0}^i \oplus p_r x^r \right] \left[\sum_{r=i+1}^n \oplus p_r x^{n-r} \right]$ $p_{BTDj}(x) = x^j + G(j) \sum_{r=0}^i \oplus p_r x^r \sum_{r=i+1}^j \oplus p_r x^{j-r}$ $G(j) = \begin{cases} 0 & \text{dla } j < i \\ 1 & \text{dla } j > i \end{cases}$
TBDT	$p_{TBDT}(x) = 1 + \left[\sum_{r=0}^{i-1} \oplus p_r \prod_{j=r}^{i-1} (k_j + x) \right] \left[p_i + \sum_{r=i+1}^n \oplus p_r \prod_{j=i}^{r-1} (k_j + x) \right]$
TBD	$p_{TBD}(x) = 1 + \left[\sum_{r=0}^{i-1} \oplus p_r x^{i-r} \right] \left[\sum_{r=i}^n \oplus p_r x^{r-i} \right]$ $p_{TBDj}(x) = x^{G(j)[j-(i-1)]} \sum_{r=0}^{H(j)} \oplus p_r x^{H(j)-r}$ $G(j) = \begin{cases} 0 & \text{dla } j < i-1 \\ 1 & \text{dla } j \geq i-1 \end{cases} \quad H(j) = \begin{cases} j & \text{dla } j < i-1 \\ i-1 & \text{dla } j \geq i-1 \end{cases}$
CADT	$p_{CADT}(x) = p_n(x) = p_{n-2}(x) + (k_{n-1} + x)p_{n-1}(x);$ $p_0(x) = 1, \quad p_1(x) = k_0 + x$ $p_{CADTr}(x) = p_r(x) = p_{r-2}(x) + (k_{r-1} + x)p_{r-1}(x);$ $p_0(x) = 1, \quad p_1(x) = k_0 + x$

Wielomiany $p_{cr}(x)$ ilustrują wpływ wyboru r-tego wejścia rejestru c-MISR na przedstawione dzielenie. Dla wybranego r-tego (j-tego) wejścia wielomiany te różnią się również między sobą w zależności od struktury sprzężenia liniowego c. Ogólne wyrażenia opisujące te wielomiany znajdują się także w tab. 2.1. Przykłady wielomianów $p_{cr}(x)$ związanych z r-tym wejściem rejestrów c-MISR przedstawiono w [Hassa82, HassL83, HassM83, Hławi87c, Hławi88a, Hławi90a] dla rejestrów IED-MISR, w [Hławi84c, Hławi86c, Maxwe88, Hławi90a, Hławi96a] dla rejestrów EED-MISR, w [Hławi89a, Hławi90a, Hławi92b, Hławi96b] dla rejestrów TBD-MISR, w [Hławi89b, Hławi90a, Hławi92b, Hławi96b] dla rejestrów BTD-MISR, w [Hławi93b, Hławi96b] dla rejestrów MISR typu CA oraz w [Hławi92a] dla rejestrów IEDT-MISR, a także IET-MISR.

charakterystycznego $p_{CADT_2}(x) = (1 + x + x^2)(1 + x^2 + x^5)$, co ma kolosalny wpływ na skuteczność testowania. Takiej właściwości nie posiadają w ogóle rejestry c-SISR, w których bez względu na rodzaj struktury sprzężenia liniowego wejście 0 związane jest zawsze z wielomianem $p_{c0}(x) = 1$. Algebraiczne opisy pracy innych przykładów rejestrów c-MISR można znaleźć w [Hławi84c, Hławi86c] dla rejestru EED-MISR, w [Hławi89a, Hławi89b, Hławi92b, Hławi93b, Hławi96b] dla rejestrów BTM-MISR oraz TBD-MISR lub w [Hławi92a] dla rejestrów IET-MISR.

2.2.3. Algebraiczny opis pracy rejestrów c-SISR, c-TISR, c-LFSR oraz MISHR

Wyrażenia opisujące pracę rejestrów liniowych c-SISR oraz c-LFSR można uzyskać wprost z wyrażenia 2.1.

Przyjmując dla 2.1, że $u_0(x) = u(x) \neq 0$ i $u_r(x) = 0$ dla $r = 1, 2, \dots, n-1$ otrzymamy $w_c(x) = u(x)$. W efekcie uzyskujemy nowe następujące wyrażenie, które ilustruje pracę rejestrów c-SISR:

$$\left[u(x) + x^m h_c(x) \right] / p_c(x) = q_c(x) + r_c(x) / p_c(x) \quad (2.5)$$

Zakładając, że $u_k(x) \neq 0$, $u_r(x) \neq 0$ oraz $u_j(x) = 0$ dla każdego $j \neq i \in \{k, r\}$ otrzymamy $w_c(x) = u_k(x) p_{ck}(x) + u_r(x) p_{cr}(x)$. Tak więc z 2.1 otrzymujemy dzielenie

$$\left[u_k(x) p_{ck}(x) + u_r(x) p_{cr}(x) + x^m h_c(x) \right] / p_c(x) = q_c(x) + r_c(x) / p_c(x) \quad (2.6)$$

ilustrujące pracę rejestrów c-TISR. Dla rejestrów EED-TISR taki algebraiczny model pracy przedstawiono po raz pierwszy w pracach [Hławi84c, Hławi86c].

Przyjmując w wyrażeniu 2.5 $u(x) = 0$ otrzymujemy równanie

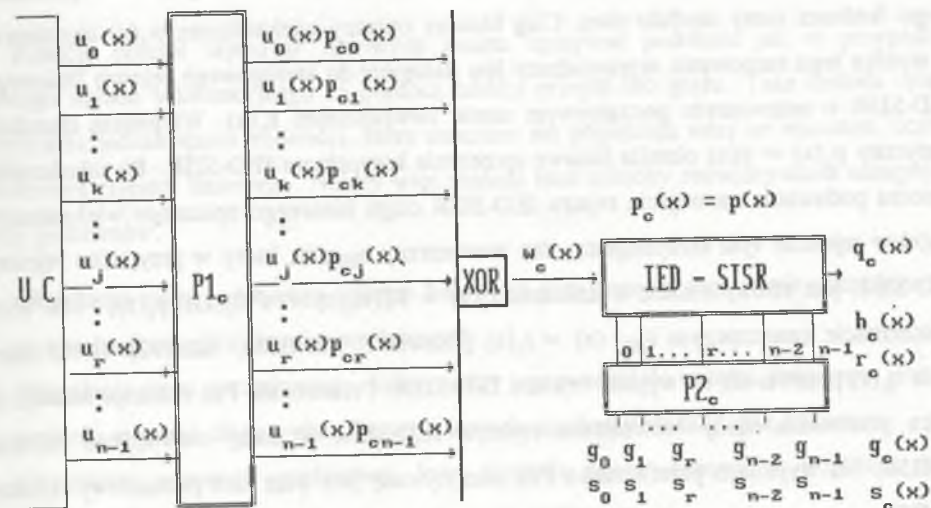
$$x^m h_c(x) / p_c(x) = q_c(x) + r_c(x) / p_c(x) \quad (2.7)$$

ilustrujące pracę rejestru c-LFSR, w którym ustawiono wstępnie stan zawierający bity g_r .

Podstawiając w wyrażeniu 2.1 wielomian $p_c(x) = x^n$ oraz przyjmując, że $p_{cr}(x) = x^r$ otrzymujemy wyrażenie opisujące pracę wielowejsściowego rejestru MISHR. Wyrażenie to, przy założeniu że $u_0(x) = u(x) \neq 0$ i $u_r(x) = 0$ dla $r = 1, 2, \dots, n-1$, przyjmuje postać opisującą zwykły n-komórkowy rejestr przesuwający SR (Shift Register).

2.2.4. Schemat zastępczy rejestrów liniowych

Każdy z opisanych poprzednio rejestrów liniowych c-MISR można przedstawić w postaci schematu zastępczego podanego na rys. 2.7 [Hławi87c, Hławi88a]. Pewną uproszczoną próbę ilustrowania pracy równoległego rejestru liniowego ograniczoną jednak do rejestru IED-MISR przedstawiono wcześniej w [SridH82]. Schemat przedstawiony na rys. 2.7 ułatwia wyjaśnianie szeregu zagadnień, między innymi związanych z testowaniem, [Hławi86b, Hławi86a] za pomocą dzielenia wielomianów w klasycznym rejestrze liniowym IED-SISR. Jest on związany z ogólnym algebraicznym opisem pracy rejestrów c-MISR podanym w wyrażeniu 2.1.



Rys. 2.7. Schemat zastępczy wielowejsściowych rejestrów liniowych c-MISR
Fig. 2.7. Substitute scheme of multi-input signature registers c-MISR

Założmy, że sekwencja m wektorów pobudza UC, na którego n wyjściach w odpowiedzi pojawia się $(m \times n)$ -bitowa macierz odpowiedzi \mathbf{U} . Oznaczając przez U_r ciąg danych o długości m na r-tym wyjściu UC można zilustrować wspomnianą macierz odpowiedzi w następujący sposób:

$$\mathbf{U} = \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_r \\ \vdots \\ U_{n-1} \end{bmatrix}$$

W schemacie zastępczym wyjścia układu cyfrowego UC połączone są z przetwornikiem liniowym P1c przekształcającym binarne ciągi U_r opisane wielomianem $u_r(x)$ w ciągi ilustrowane wielomianem $u_r(x) p_{cr}(x)$. Te ostatnie są podawane na n wejść wieloargumentowego funktora sumy modulo dwa. Ciąg binarny opisany wielomianem $w_c(x)$ uzyskany w wyniku tego sumowania wprowadzany jest następnie do szeregowego rejestru liniowego IED-SISR o ustawionym początkowym stanie wewnętrznym $h_c(x)$. Wielomian charakterystyczny $p_c(x) = p(x)$ określa liniowe sprzężenie kompaktora IED-SISR. Po zakończeniu procesu podawania na wejście rejestru IED-SISR ciągu binarnego opisanego wielomianem $w_c(x)$ w rejestrze tym otrzymujemy stan wewnętrzny $s_{IED}(x)$, który w przypadku rejestru IED-SISR jest równy reszcie z dzielenia $r_c(x) = R[(w_c(x) + x^m h_c(x)), p_c(x)]$. Tak więc w schemacie zastępczym $s_{IED}(x) = r_c(x)$ [Hławi87c, Hławi88a, SaluS92]. Iloraz dzielenia $q_c(x)$ pojawia się na wyjściu rejestru IED-SISR. Przetwornik P2c realizuje funkcję δ , która przetwarza stany wewnętrzne rejestru IED-SISR w stany wewnętrzne rejestru c-MISR. Na wyjściach przetwornika P2c odczytywany jest więc stan początkowy rejestru c-MISR

$$g_c(x) = \sum_{j=0}^{n-1} \oplus g_r = \delta(h_c(x))$$

oraz zawarta w nim po zakończeniu procesu kompaktacji końcowa sygnatura

$$s_c(x) = \sum_{j=0}^{n-1} \oplus s_r = \delta(r_c(x))$$

Związane z r -tymi wejściami rejestru c-MISR wielomiany $p_{cr}(x)$ opisują sposób, w jaki przetwarzane są poszczególne wielomiany $u_r(x)$ w przetworniku P1c. Są one też niezbędne do określenia funkcji δ przetwornika P2c.

Schemat zastępczy rejestru szeregowego c-SISR zawiera jednowejściowy - połączony z UC - rejestr IED-SISR oraz przetwornik P2c. Schemat zastępczy dwuwiejściowego

rejestru c-TISR jest podobny do schematu przedstawionego na rys. 2.7. Jedyna różnica tkwi w przetworniku P1c, który zamiast n wejść posiada ich w tym schemacie tylko dwa. Schemat zastępczy rejestru c-LFSR jest podobny do schematu kompaktora c-SISR. Różnicą jest brak wejścia w rejestrze IED-SISR, co w konsekwencji powoduje, że w schemacie tym zamiast IED-SISR należy umieścić rejestr IED-LFSR.

2.3. Czy można zastąpić graf przejść inną techniką określania stanów wewnętrznych rejestrów liniowych ?

Funkcję przejść rejestrów liniowych można opisywać podobnie jak w przypadku każdego układu sekwencyjnego za pomocą tablicy przejść lub grafu. Taka metoda opisu przysparza jednak sporo trudności, które znacząco się pogłębiają wraz ze wzrostem liczby n komórek rejestru liniowego. Należy więc znaleźć inne sposoby rozwiązywania następujących problemów.

1. Określanie stanu końcowego rejestru liniowego przy znanym jego stanie początkowym i znanej sekwencji wektorów wejściowych.
2. Określanie stanu początkowego, który należy wprowadzić do rejestru liniowego, aby po podaniu znanego ciągu wektorów wejściowych uzyskać pożądany stan końcowy.
3. Określanie sekwencji wejściowej, która ustawia rejestr liniowy w pożądany stan końcowy przy założonym dowolnym stanie początkowym.

Okazuje się, że powyższe zagadnienia można rozwiązać po przekształceniu 2.1 w następujące równości:

$$\text{dla problemu 1: } r_c(x) = w_c(x) + x^m h_c(x) + q_c(x) p(x) \quad (2.8)$$

$$\text{dla problemu 2: } h_c^*(x) = w_c^*(x) x + x^m r_c^*(x) + q_c^*(x) p^*(x) \quad (2.9)$$

$$\text{dla problemu 3: } \begin{cases} w_c(x) = r_w(x) + q_w(x) / p(x) \\ x^m h_c(x) = r_h(x) + q_h(x) / p(x) \end{cases} \quad (2.10)$$

gdzie $q_c(x) = q_w(x) + q_h(x)$ oraz $r_c(x) = r_w(x) + r_h(x)$.

W wyrażeniu 2.8 ustawiony stan początkowy uwikłany w wielomianie $h_c(x)$ oraz podana sekwencja wektorów wejściowych przetransponowana w $w_c(x)$ umożliwia określenie stanu końcowego rejestru liniowego ukrytego w reszcie $r_c(x)$. Dzięki zastosowanej w 2.1 operacji odwracania wielomianów [Hławi87c, Hławi88a] uzyskano wyrażenie 2.9. W wyrażeniu tym ustawiony stan początkowy ukryty w odwróconym wielomianie $r_c^+(x)$ oraz podana odwrócona sekwencja wektorów wejściowych przetransponowana w wielomian $w_c^+(x)$ pozwalają na określenie stanu końcowego rejestru liniowego uwikłanego w odwróconej reszcie $h_c^+(x)$. Wyrażenie 2.9 uzyskano dla rejestrów EED-MISR po raz pierwszy w [Hławi87a], a następnie w [Hławi87c, Hławi88a]. Nieco inną formę tego wyrażenia przedstawiono także w [Hławi89e, Hławi90a]. W układzie równań 2.10 dzięki znanemu stanowi początkowemu (wielomian $h_c(x)$) oraz założonemu stanowi końcowemu (wielomian $r_c(x)$) staje się możliwe ustalenie wielomianu $w_c(x)$, a w efekcie tych wszystkich macierzy wyjściowych UC, które można przetransponować w $w_c(x)$.

Ponieważ wielomiany $a(x)$ oraz $a^+(x)$ można wzajemnie odwracać, dlatego zagadnienie określania stanów (początkowego lub końcowego) w każdym z powyższych trzech problemów można sprowadzić do techniki ich wyznaczania w oparciu o następujące, określone w 2.1, wyrażenia:

$$h_c(x) = \sum_{r=0}^{n-1} \oplus g_r p_{cr}(x) \quad r_c(x) = \sum_{r=0}^{n-1} \oplus s_r p_{cr}(x) \quad (2.11)$$

Zawarte w 2.11 zbiory bitów $\{g_r\}$ oraz $\{s_r\}$ określają stan początkowy $g_c(x)$ oraz stan końcowy $s_c(x)$ rejestru c-MISR lub c-SISR. Ustalenie któregośkolwiek z nich wymaga jednak znajomości wielomianu $h_c(x)$ lub reszty $r_c(x)$, które w związku z tym muszą być określone w pierwszej kolejności.

2.3.1. Techniki określania reszty $r_c(x)$ oraz wielomianu $h_c(x)$

Sposób określania reszty $r_c(x)$ oraz wielomianu $h_c(x)$ zostanie wyjaśniony na przykładzie reszty. W oparciu o 2.8 można napisać, że:

$$r_c(x) = R [w_c(x) + x^m h_c(x), p(x)]$$

Żałujemy, że został ustawiony zerowy stan początkowy w rejestrze liniowym. Oznacza to równocześnie, że $h_c(x) = 0$. Przy założeniu że $p(x)$ jest stałe dla każdej struktury

sprzężenia liniowego rejestru c-MISR (c-SISR) o końcowej postaci reszty $r_c(x)$ będzie decydować wyłącznie określony w 2.1. wielomian:

$$w_c(x) = \sum_{r=0}^{n-1} \oplus u_r(x) p_{cr}(x)$$

Tak więc w pierwszej kolejności zostanie wyjaśniony sposób obliczania wielomianu $w_c(x)$. Przedstawiono go w przykładzie 2.3. W kolejnych dwóch przykładach 2.4 oraz 2.5 wyjaśniono dwie techniki określania reszty $r_c(x)$.

Przykład 2.3

Żałujemy wielomian $p(x) = x^5 + x^4 + x^3 + x^2 + 1$. Słowo $\langle 111101 \rangle$ jest jego postacią binarną. Następujący zbiór rejestrów c-MISR jest związany tym wielomianem:

IED-MISR; $DD \oplus D \oplus D \oplus D$ EET-MISR; $(\oplus' D)(\oplus' D)^\oplus(\oplus' D)(\oplus' D)(\oplus' D)$
 IEDT-MISR; $DD(\oplus' D) \oplus DD$ TBD-MISR; $D^\oplus DD \oplus DD$
 TD-MISR; $DTTDT$ BTM-MISR; $D \oplus DD \oplus D^\oplus D$
 EED-MISR; $D^\oplus D^\oplus D^\oplus DD$ MISR typu CA; 10011

W oparciu o wyrażenia podane w tab. 2.1. można obliczyć wielomiany $p_{c_j}(x)$ charakteryzujące j-te wejścia tych rejestrów liniowych. Wyniki obliczeń przedstawiono w tab. 2.2.

Tabela 2.2

c	$P_{c0}(x)$	$P_{c1}(x)$	$P_{c2}(x)$	$P_{c3}(x)$	$P_{c4}(x)$
IED	1	x	x^2	x^3	x^4
IEDT	1	x	x^2	$x^2 + x^3$	$x^3 + x^4$
TD	1	x	$x + x^2$	$x + x^3$	$x^2 + x^4$
EED	1	$1 + x$	$1 + x + x^2$	$1 + x + x^2 + x^3$	$x + x^2 + x^3 + x^4$
EET	1	$1 + x$	x^2	$x^2 + x^3$	$x^2 + x^4$
TBD	1	$1 + x$	$x + x^2$	$x^2 + x^3$	$x^3 + x^4$
BTD	1	x	x^2	x^3	$1 + x + x^3 + x^4$
CADT	1	$1 + x$	$1 + x + x^2$	$1 + x^2 + x^3$	x^4

Żałujemy następujący zbiór wielomianów $u_j(x)$ i odpowiadających im ciągów binarnych opisujących macierz wyjściową UC podawaną na wejścia przykładowych rejestrów c-MISR:

$$\begin{aligned} u_0(x) &= x^4 + x^2 + x; & \langle 10110 \rangle; & & u_3(x) &= x^3 + x^2 + 1; & \langle 01101 \rangle \\ u_1(x) &= x^2 + 1; & \langle 00101 \rangle; & & u_4(x) &= x^4 + x^3 + 1; & \langle 11001 \rangle \\ u_2(x) &= x^4 + x^3 + x; & \langle 11010 \rangle & & & & \end{aligned}$$

Wybierając w pierwszej kolejności rejestr **DTTDT**, otrzymujemy w oparciu o tab. 2.2. oraz podane wielomiany $u_j(x)$ następujący wielomian:

$$\begin{aligned} w_{TD}(x) &= u_0(x) + u_1(x)x + u_2(x)(x+x^2) + u_3(x)(x+x^3) + u_4(x)(x^2+x^4) = \\ &= x^8 + x^7 + x^6 + x^2 + x; & \langle 111000110 \rangle \end{aligned}$$

W podobny sposób otrzymujemy dla pozostałych rejestrów wielomiany $w_c(x)$ i odpowiadające im różniące się między sobą ciągi binarne

$$\begin{array}{lll} w_{IED}(x); & \langle 110011100 \rangle & w_{EED}(x); \langle 100011010 \rangle & w_{BTD}(x); \langle 101101111 \rangle \\ w_{IEDT}(x); & \langle 101110000 \rangle & w_{EET}(x); \langle 111011001 \rangle & w_{CA}(x); \langle 110011110 \rangle \\ w_{TD}(x); & \langle 111000110 \rangle & w_{TBD}(x); \langle 101000001 \rangle & \end{array}$$

Zauważamy jednak, że $w_{IED}(x) = w_{IEDT}(x)$, jeżeli w macierzy wyjściowej UC wielomiany $u_3(x) = u_4(x) = 0$. Podobnie przy założeniu $u_1(x) = u_2(x) = 0$ stwierdzamy, że $w_{IEDT}(x) = w_{TBD}(x)$. Tak więc dla każdego rejestru c-MISR-p(x) o innej strukturze sprzężenia liniowego niż d-MISR-p(x) wielomiany

$$w_c(x) \neq w_d(x)$$

W przypadku rejestrów c-SISR-p(x) wielomian $w_c(x) = u_0(x)$, co oznacza, że dla różniących się strukturą sprzężenia liniowego rejestrów c-SISR-p(x) oraz d-SISR-p(x) zawsze wielomian $w_c(x) = w_d(x)$.

Różnice w wielomianach $w_c(x)$ oraz $w_d(x)$ wpływają na to, że również różnią się reszty $R[w_c(x), p(x)]$, $R[w_d(x), p(x)]$. Tak więc w przypadku rejestrów c-MISR i d-MISR związanych wielomianem charakterystycznym $p(x)$ mamy $r_c(x) \neq r_d(x)$. Oznacza to, że wynik dzielenia przez wielomian $p(x)$ zależy od struktury sprzężeń liniowych w rejestrach c-MISR oraz d-MISR związanych wielomianem $p(x)$.

W przypadku rejestrów c-SISR-p(x) oraz d-SISR-p(x) prawdziwa jest równość $r_c(x) = r_d(x)$. Oznacza ona, że reszty z dzielenia w tych rejestrach są niezależne od ich struktury sprzężenia liniowego.

Innym wspomnianym poprzednio istotnym problemem związanym z resztą jest znalezienie prostego sposobu jej określania. Zastosowanie znanej techniki dzielenia ciągu binar-

nego (wielomianu $w_c(x)$) w rejestrze IED-SISR jest jednym ze sposobów rozwiązania tego problemu. Poniższy przykład służy przypomnieniu tej techniki.

Przykład 2.4

Należy określić resztę $r_{TD}(x) = R[w_{TD}(x), p(x)]$ dla rejestru **DTTDT**. W oparciu o 2.8 można zrealizować następujące dzielenie

$$\begin{array}{r} w_{TD}(x) \quad 111000110 \\ \underline{111101} \\ 101110 \\ \underline{111101} \\ r_{TD}(x) \quad 10011 \end{array} \quad : \quad 111101 \quad p(x)$$

Reszta $r_{TD}(x) = x^4 + x + 1$.

W przypadku konieczności określenia reszt dla zbioru rejestrów c-MISR-p(x) metoda dzielenia osobno każdego wielomianu $w_c(x)$ staje się nieefektywna. Potrzebny jest więc inny sposób.

Zauważmy, że resztę $r_c(x)$ można także obliczyć korzystając z poniższego wyrażenia

$$r_c(x) = \sum_{j=0}^{n-1} \oplus R[u_j(x)p_{c_j}(x), p(x)] = \sum_{j=0}^{n-1} \oplus r_{c_j}(x) \quad (2.12)$$

Zauważmy ponadto, że wielomian związany z j-tym wejściem rejestru c-MISR można przedstawić w takiej postaci

$$p_{c_j}(x) = \sum_{i=0}^j \oplus a_{c_{ji}} x^i \quad \text{gdzie} \quad a_{c_{ji}} \in \{0, 1\}$$

Podstawiając to wyrażenie do 2.12 otrzymujemy

$$r_c(x) = \sum_{j=0}^{n-1} \oplus r_{c_j}(x) = \sum_{j=0}^{n-1} \oplus \sum_{i=0}^j \oplus R[u_j(x)a_{c_{ji}}x^i, p(x)] \quad (2.13)$$

które jest szczególnie przydatne do określania reszt w przypadku licznego zbioru rejestrów c-MISR-p(x). Tworząc następujący katalog reszt

$$\{r_{ij}(x)\} = \{R[u_j(x)x^i, p(x)]\} \quad j = 0, 1, \dots, n-1; \quad i = 0, 1, \dots, j$$

można przy użyciu 2.13 znacznie szybciej obliczyć każdą resztę $r_c(x)$.

Przykład 2.5

Dla rejestrów c-MISR-p(x) podanych w przykładzie 2.3. oraz macierzy wyjściowej UC określonej przez zbiór wielomianów $\{u_j(x)\}$ ($j = 0, 1, 2, 3, 4$) tego przykładu należy określić

katalog reszt $\{r_{ji}(x)\}$. Stosując technikę dzielenia ciągów binarnych podaną w przykładzie 2.4, otrzymujemy następujący zbiór reszt:

$$\begin{aligned} r_{00}(x) &= R[u_0(x), p(x)] = \langle 10110 \rangle; & r_{32}(x) &= R[u_3(x)x^2, p(x)] = \langle 01001 \rangle \\ r_{10}(x) &= R[u_1(x), p(x)] = \langle 00101 \rangle; & r_{33}(x) &= R[u_3(x)x^3, p(x)] = \langle 10010 \rangle \\ r_{11}(x) &= R[u_1(x)x, p(x)] = \langle 01010 \rangle; & r_{40}(x) &= R[u_4(x), p(x)] = \langle 11001 \rangle \\ r_{20}(x) &= R[u_2(x), p(x)] = \langle 11010 \rangle; & r_{41}(x) &= R[u_4(x)x, p(x)] = \langle 01111 \rangle \\ r_{21}(x) &= R[u_2(x)x, p(x)] = \langle 01001 \rangle; & r_{42}(x) &= R[u_4(x)x^2, p(x)] = \langle 11110 \rangle \\ r_{22}(x) &= R[u_2(x)x^2, p(x)] = \langle 10010 \rangle; & r_{43}(x) &= R[u_4(x)x^3, p(x)] = \langle 00001 \rangle \\ r_{30}(x) &= R[u_3(x), p(x)] = \langle 01101 \rangle; & r_{44}(x) &= R[u_4(x)x^4, p(x)] = \langle 00010 \rangle \\ r_{31}(x) &= R[u_3(x)x, p(x)] = \langle 11010 \rangle \end{aligned}$$

Na podstawie tego katalogu, a także wielomianów charakteryzujących j-te wejścia rejestrów c-MISR (tab. 2.2) oraz wyrażenia 2.13 można określić każdą resztę $r_c(x)$. Wyjaśnia to związany z rejestrem **DTDT** następujący wielomian

$$r_{TD}(x) = r_{00}(x) + r_{11}(x) + [r_{21}(x) + r_{22}(x)] + [r_{31}(x) + r_{33}(x)] + [r_{42}(x) + r_{44}(x)]$$

Po podstawieniu słów binarnych z katalogu reszt otrzymujemy $r_{TD}(x) = \langle 10011 \rangle$. Reszty obliczone w ten sposób dla wszystkich rejestrów c-MISR przedstawiono poniżej

$$\begin{aligned} r_{IED}(x) &= \langle 11110 \rangle; & r_{EED}(x) &= \langle 00110 \rangle; & r_{BTD}(x) &= \langle 01001 \rangle \\ r_{IEDT}(x) &= \langle 10110 \rangle; & r_{EET}(x) &= \langle 01100 \rangle; & r_{CA}(x) &= \langle 01100 \rangle. \\ r_{TD}(x) &= \langle 10011 \rangle; & r_{TBD}(x) &= \langle 11010 \rangle; \end{aligned}$$

Zauważmy również, że reszta $r_{00}(x) = R[u_0(x), p(x)] = \langle 10110 \rangle$ jest resztą uzyskiwaną w każdym z rejestrów c-LFSR-p(x).

Przydatność podanego sposobu określania reszt dla zbioru rejestrów c-MISR-p(x) w szczególny sposób ujawnia się, gdy wymiary macierzy wyjściowej UC są bardzo duże. Korzystając przy obliczaniu katalogu reszt $\{r_{ji}(x)\}$ z szybkich procedur dla określania reszt w rejestrach IED-SISR i podanych w [SaluS92] lub w [Tan87, LambI91, SeeS92], można w znaczny sposób zwiększyć efektywność procesu obliczania reszt dla grupy rejestrów c-MISR-p(x).

Skuteczność tych procedur zilustrowano poniżej na przykładzie techniki szybkiego określania reszty podanej w [SaluS92]. W celu uproszczenia opisu ograniczono uwagę do dzielenia w rejestrze IED-MISR-p(x) ciągu podawanego na jego r-te wejście. Stopień wielo-

mianu $p(x)$, $\deg p(x) = 16$. Niech $u_r(x) x^r = u(x) = \sum_{i=0}^{k-1} \oplus u_i(x) x^{8i}$, gdzie $\deg u_i(x) = 7$.

Oznacza to, że ciąg komprimowany $u(x)$ jest podzielony na k fragmentów, każdy o długości ośmiu bitów. W każdym i -tym kroku procedury Saluji określa się resztę z dzielenia ośmiobitowego ciągu $u_i(x)$ przez wielomian $p(x)$ związany z 16-bitowym rejestrem IED-SISR. Rejestr ten przed każdym takim dzieleniem ustawiony jest w stan początkowy $h_i(x) = x^8 h_{i1}(x) + h_{i2}(x)$, który jest resztą z dzielenia $r_{i-1}(x)$ uzyskaną w poprzednim $i-1$ kroku. W procedurze tej stopnie wielomianów określających stan początkowy są następujące: $\deg h_i(x) = 15$, $\deg h_{i1}(x) = \deg h_{i2}(x) = 7$. Na tej podstawie proces kompaktacji w i -tym kroku tej procedury można opisać za pomocą następującego dzielenia:

$$[u_i(x) + x^8(x^8 h_{i1}(x) + h_{i2}(x))]/p(x) = q_i(x) + r_i(x)/p(x)$$

Tak więc reszta

$$r_i(x) = R[u_i(x), p(x)] + R[x^8(x^8 h_{i1}(x), p(x))] + R[x^8 h_{i2}(x), p(x)]$$

Biorąc pod uwagę, że

$$R[u_i(x), p(x)] = u_i(x), R[x^8(x^8 h_{i1}(x), p(x))] = r_{ih1}(x) \text{ oraz}$$

$$R[x^8 h_{i2}(x), p(x)] = x^8 h_{i2}(x) \text{ otrzymujemy ostatecznie}$$

$$r_i(x) = r_{ih1}(x) + [x^8 h_{i2}(x) + u_i(x)]$$

Ze względu na to, że $r_{IED}(x) = s_{IED}(x)$, resztę $r_{ih1}(x)$ określa się w procedurze Saluji w oparciu o tablicę "lookup" zawierającą stany rejestru IED-LFSR-p(x) uzyskane po ośmiu taktach zegarowych podanych na wejście zegarowe tego rejestru, w momencie w którym zawierał on stan początkowy $x^8 h_{i2}(x)$. Ze względu na to, że istnieje tylko 256 różnych stanów $x^8 h_{i2}(x)$, tablica typu "lookup" zawierać będzie 2^8 wierszy zamiast 2^{16} . Ostatecznie określenie reszty $r_i(x) = r_{ih1}(x) + [x^8 h_{i2}(x) + u_i(x)]$ w każdym i -tym kroku procedury złożone jest z trzech następujących operacji:

- operacji czytania stanu $r_{ih1}(x)$ z 256-wierszowej tablicy typu "lookup",
- operacji konkatencji dwóch bajtów $x^8 h_{i2}(x)$ i $u_i(x)$ oraz,
- operacji sumowania modulo dwa $r_{ih1}(x) + [x^8 h_{i2}(x) + u_i(x)]$.

Określenie reszty $r(x) = R[u(x), p(x)]$ wymaga więc jedynie k -krotnego powtórzenia tych trzech prostych operacji.

2.3.2. Określanie stanów rejestrów liniowych oraz ich konwersja

Sygnatura jest stanem wewnętrznym szeregowego lub równoległego rejestru liniowego uzyskiwanym na końcu procesu podawania na wejścia tych rejestrów wektorów z macierzy wyjściowej UC. W schemacie zastępczym (rys. 2.7) rejestrów liniowych c-MISR (c-SISR) przetwornik P2c generuje na swoich wyjściach wewnętrzne stany tych rejestrów. Obecnie określimy realizowaną przez ten przetwornik funkcję δ .

Ogólna formuła określająca w wyrażeniu 2.1. resztę $r_c(x)$ jest niezwykle trudna do przekształcenia w ogólne wyrażenie przedstawiające sygnaturę $s_c(x)$ jako funkcję bitów reszty $r_c(x)$. Proces ten jest natomiast bardzo prosty w przypadku założenia konkretnej struktury sprzężenia liniowego pojedynczego rejestru c-MISR. Wówczas w pierwszym kroku należy określić formuły opisujące resztę jako funkcję bitów sygnatury. Wyjaśnia to przykład 2.6. Następnie w drugim kroku w oparciu o uzyskane wyrażenia należy określić formuły opisujące sygnaturę jako funkcje bitów reszty. Ten krok wyjaśnia przykład 2.7.

Przykład 2.6

Weźmy zbiór rejestrów c-MISR (przykład 2.3). W pierwszej kolejności skoncentrujemy się na rejestrze **DTTDT**. Podstawiając do określonego w 2.1 wyrażenia $r_c(x)$ w miejsce $P_{Cr}(x)$ wzięte z tab. 2.1 (w. 1) wyrażenie $P_{IEDT_r}(x)$ i przyjmując $n = 5$, otrzymujemy

$$r_{TD}(x) = s_0 + \sum_{r=1}^4 \oplus s_r \prod_{j=0}^{r-1} (k_j + x)$$

Współczynniki k_j odpowiadające przerzutnikom T oraz D rejestru **DTTDT** przyjmują następujące wartości $k_0 = 0$; $k_1 = 1$; $k_2 = 1$; $k_3 = 0$; $k_4 = 1$. Podstawiając je do powyższego wyrażenia, otrzymujemy

$$r_{TD}(x) = s_0 + s_1 x + s_2 (x + x^2) + s_3 (x + x^3) + s_4 (x^2 + x^4)$$

Po przekształceniach wyrażenie to przyjmuje postać

$$r_{TD}(x) = s_0 + (s_1 + s_2 + s_3)x + (s_2 + s_4)x^2 + s_3 x^3 + s_4 x^4$$

z której wynikają następujące proste formuły

$$r_0 = s_0 \quad r_1 = s_1 + s_2 + s_3 \quad r_2 = s_2 + s_4 \quad r_3 = s_3 \quad r_4 = s_4$$

W podobny sposób można uzyskać formuły umożliwiające obliczanie reszt dla pozostałych rejestrów c-MISR z przykładu 2.3. Formuły te zilustrowano w tab. 2.3.

Tabela 2.3

	IED-MISR	IE-MISR	TD-MISR	EED-MISR
$r_0 =$	s_0	s_0	s_0	$s_0 + s_1 + s_2 + s_3$
$r_1 =$	s_1	s_1	$s_1 + s_2 + s_3$	$s_1 + s_2 + s_3 + s_4$
$r_2 =$	s_2	$s_2 + s_3$	$s_2 + s_4$	$s_2 + s_3 + s_4$
$r_3 =$	s_3	$s_3 + s_4$	s_3	$s_3 + s_4$
$r_4 =$	s_4	s_4	s_4	s_4
	EET-MISR	TBD-MISR	BTD-MISR	MISR typu CA
$r_0 =$	$s_0 + s_1$	$s_0 + s_1$	$s_0 + s_4$	$s_0 + s_1 + s_2 + s_3$
$r_1 =$	s_1	$s_1 + s_2$	$s_1 + s_4$	$s_1 + s_2$
$r_2 =$	$s_2 + s_3 + s_4$	$s_2 + s_3$	s_2	$s_2 + s_3$
$r_3 =$	s_3	$s_3 + s_4$	$s_3 + s_4$	s_3
$r_4 =$	s_4	s_4	s_4	s_4

Przykład 2.7

Przekształcamy wzięte z przykładu 2.6. i związane z rejestrem **DTTDT** formuły

$$r_0 = s_0 \quad r_1 = s_1 + s_2 + s_3 \quad r_2 = s_2 + s_4 \quad r_3 = s_3 \quad r_4 = s_4$$

do następującej postaci:

$$s_0 = r_0 \quad s_1 = r_1 + r_2 + r_3 + r_4 \quad s_2 = r_2 + r_4 \quad s_3 = r_3 \quad s_4 = r_4$$

W podobny sposób można przekształcić pozostałe przedstawione w tab. 2.3 formuły. Efekt tych przekształceń ujęto w tab. 2.4.

Podstawiając do przedstawionych w tab. 2.4. formuł wartości bitów różnych reszt rejestrów c-MISR uzyskanych wcześniej w przykładzie 2.5, otrzymujemy następujące odpowiadające tym resztom sygnatury:

$$\begin{aligned} s_{IED}(x) &= \langle 11110 \rangle; & s_{EED}(x) &= \langle 00101 \rangle; & s_{BTD}(x) &= \langle 01001 \rangle \\ s_{IEDT}(x) &= \langle 11010 \rangle; & s_{EET}(x) &= \langle 01000 \rangle; & s_{CA}(x) &= \langle 01001 \rangle \\ s_{TD}(x) &= \langle 10101 \rangle; & s_{TBD}(x) &= \langle 10011 \rangle; & & \end{aligned}$$

Tabela 2.4

	IED-MISR	IE-MISR	TD-MISR	EED-MISR
$s_0 =$	r_0	r_0	r_0	$r_0 + r_1 + r_4$
$s_1 =$	r_1	r_1	$r_1 + r_2 + r_3 + r_4$	$r_1 + r_2$
$s_2 =$	r_2	$r_2 + r_3 + r_4$	$r_2 + r_4$	$r_2 + r_3$
$s_3 =$	r_3	$r_3 + r_4$	r_3	$r_3 + r_4$
$s_4 =$	r_4	r_4	r_4	r_4
	EET-MISR	TBD-MISR	BTD-MISR	MISR typu CA
$s_0 =$	$r_0 + r_1$	$r_0 + r_1 + r_2 + r_3 + r_4$	$r_0 + r_4$	$r_0 + r_1 + r_3$
$s_1 =$	r_1	$r_1 + r_2 + r_3 + r_4$	$r_1 + r_4$	$r_1 + r_2 + r_3$
$s_2 =$	$r_2 + r_3 + r_4$	$r_2 + r_3 + r_4$	r_2	$r_2 + r_3$
$s_3 =$	r_3	$r_3 + r_4$	$r_3 + r_4$	r_3
$s_4 =$	r_4	r_4	r_4	r_4

Dokonując dzielenia wielomianu $u_0(x)$ za pomocą rejestrów c-SISR $p(x)$, otrzymuje się jednakową resztę $\langle 10110 \rangle$ dla wszystkich tych rejestrów. Wszystkie bity tej reszty podstawione do formuł zawartych w tabelicy 2.4 dają po zsumowaniu zbiór następujących różniących się między sobą sygnatur w rejestrach c-SISR:

$$\begin{aligned} s'_{IED}(x) &= \langle 10110 \rangle; & s'_{EED}(x) &= \langle 11100 \rangle; & s'_{BTD}(x) &= \langle 11101 \rangle \\ s'_{IEDT}(x) &= \langle 11010 \rangle; & s'_{EET}(x) &= \langle 10011 \rangle; & s'_{CA}(x) &= \langle 10101 \rangle. \\ s'_{TD}(x) &= \langle 10010 \rangle; & s'_{TBD}(x) &= \langle 11011 \rangle. \end{aligned}$$

Podane przykłady pozwalają zaakceptować następujące wnioski związane z rejestrami [Hławi84c, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi92b]:

$$r_c(x) \neq 0 \Leftrightarrow s_c(x) \neq 0 \quad \text{oraz} \quad r_c(x) = 0 \Leftrightarrow s_c(x) = 0$$

Istotne jest także przypomnienie, że w przypadku rejestrów IED-SISR ważna jest następująca równość $r_{IED}(x) = s_{IED}(x)$.

Dla rejestrów c-MISR (c-LFSR) oraz d-MISR (d-LFSR) związanych wielomianem $p(x)$ ważny jest także wniosek, że $s_c(x) \neq s_d(x)$. Wniosek ostatni w związku z tym, że $r_c(x) = r_d(x)$ jest w przypadku rejestrów c-LFSR- $p(x)$ oraz d-LFSR- $p(x)$ pewnym zaskoczeniem.

Zauważmy, że na podstawie znanego stanu wewnętrznego np. sygnatury $s_c(x)$ rejestru liniowego c-SISR- $p(x)$ można określić sygnaturę $s_d(x)$, czyli stan wewnętrzny rejestru d-SISR związanego tym samym wielomianem charakterystycznym $p(x)$, ale o innej strukturze sprzężenia liniowego. Taki proces nazywany jest konwersją sygnatury $s_c(x)$ w sygnaturę $s_d(x)$ [SaluC92, Frank95]. Może być on stosowany w następujących przypadkach [SaluC92]:

- * W celu uniknięcia czasochłonnego procesu odczytywania ciągu wyjściowego UC i wprowadzania go do rejestru d-SISR.
- * Wtedy kiedy określono sygnaturę $s_c(x)$ dla rejestru c-SISR tuż po podjęciu decyzji o wyborze wielomianu charakterystycznego $p(x)$, ale jeszcze przed podjęciem decyzji o rodzaju struktury sprzężenia liniowego c-SISR, czy też d-SISR.
- * Gdy konieczna jest znajomość obu sygnatur $s_c(x)$ oraz $s_d(x)$ jeszcze przed ostateczną decyzją o wyborze typu rejestru c-SISR czy też d-SISR
- * Wtedy kiedy proces symulacji pozwalający określić sygnaturę $s_c(x)$ może być szybszy od procesu symulacji służącego do określenia sygnatury $s_d(x)$. W takim przypadku określamy tę sygnaturę, którą można szybciej określić, czyli $s_c(x)$, a następnie przeprowadzamy jej konwersję w sygnaturę $s_d(x)$.

Proces konwersji w przypadku rejestrów c-SISR- $p(x)$ oraz d-SISR- $p(x)$ realizowany jest w dwóch krokach. W pierwszym określa się resztę $r_c(x)$ w oparciu o znaną sygnaturę $s_c(x)$. W drugim ze względu na to, że $r_d(x) = r_c(x)$ ustala się $s_d(x)$ w oparciu o $r_c(x)$.

Jedną z technik konwersji polegającą na korzystaniu z tablic "lookup" przedstawiono w [SaluS92]. Opis jej ograniczono jednak tylko do dwóch typów konwersji związanych z sygnaturami rejestrów IED-SISR i EED-SISR.

Przedstawione w przykładach 2.6 oraz 2.7 techniki umożliwiają realizację obu kroków konwersji dla wszystkich opisanych w rozdziale 2.1 typów sprzężeń liniowych rejestrów SISR. W pierwszym kroku można korzystać z tabeli podobnej do tab. 2.3, natomiast w drugim można zastosować tabelę podobną do tab. 2.4. Nieco podobną technikę konwersji, ale ograniczoną do konwersji sygnatury rejestru IED-SISR w sygnaturę rejestru SISR typu CA przedstawiono w pracy [Frank95].

Proces konwersji sygnatur w przypadku dwóch różnych rejestrów c-MISR- $p(x)$ oraz d-MISR- $p(x)$ można zrealizować za pomocą dwóch opisanych poprzednio kroków tylko w sytuacji, gdy $w_c(x) = w_d(x)$ lub w przypadku bardziej ogólnym, gdy $r_c(x) = r_d(x)$.

W pozostałych przypadkach z powodu różnic istniejących pomiędzy $r_c(x)$ oraz $r_d(x)$ nie można stosować podanej metody. Należy więc uzupełnić ją dodatkowym pośrednim krokiem polegającym na określaniu reszty $r_d(x)$ w oparciu o określoną w pierwszym kroku resztę $r_c(x)$ i niektóre wybrane reszty z katalogu reszt $\{r_{ji}(x)\}$. Najogólniej rzecz biorąc

$$r_d(x) = r_c(x) + \sum \oplus r_{ji}(x) \quad (2.14)$$

Różnica wielomianów $p_{cj}(x) + p_{dj}(x)$ charakteryzujących j -te wejścia rejestrów c-MISR- $p(x)$ oraz d-MISR- $p(x)$ wskazuje za pomocą wykładników "i" zmiennych x te reszty $r_{ji}(x)$ z katalogu reszt, które powinny być odjęte (dodane modulo dwa) od reszty $r_c(x)$ w wyrażeniu 2.14. Cały ten proces wraz z pozostałymi znanymi już krokami procesu konwersji wyjaśnia poniższy przykład.

Przykład 2.8

Należy określić sygnaturę rejestru $\mathbf{D} \oplus \mathbf{DD} \oplus \mathbf{D}^{\oplus} \mathbf{D}$ (przykład 2.3) w oparciu o sygnaturę $s_{IEDT}(x) = \langle 11010 \rangle$ rejestru $\mathbf{DD}(\oplus' \mathbf{D}) \oplus \mathbf{DD}$.

W oparciu o tab. 2.3. oraz sygnaturę $s_{IEDT}(x)$ ustalamy resztę $r_{IEDT}(x) = \langle 10110 \rangle$. Przy użyciu tab. 2.2 określamy wszystkie różnice wielomianów charakteryzujących kolejne wejścia obu przykładowych rejestrów, a następnie wybieramy za pomocą wykładników "i" zmiennych x te reszty $r_{ji}(x)$ z katalogu reszt, które powinny być odjęte od reszty $r_c(x)$ w wyrażeniu 2.14. Oto rezultaty tych działań

$$\begin{aligned} j = 0; & P_{BTD0}(x) + P_{IEDT0}(x) = 0 \\ j = 1; & P_{BTD1}(x) + P_{IEDT1}(x) = 0 \\ j = 2; & P_{BTD2}(x) + P_{IEDT2}(x) = 0 \\ j = 3; & P_{BTD3}(x) + P_{IEDT3}(x) = x^2; \quad r_{32}(x) = R[u_3(x)x^2, p(x)] = \langle 01001 \rangle \\ j = 4; & P_{BTD4}(x) + P_{IEDT4}(x) = 1 + x; \quad r_{40}(x) = R[u_4(x), p(x)] = \langle 11001 \rangle, \\ & r_{41}(x) = R[u_4(x)x, p(x)] = \langle 01111 \rangle \end{aligned}$$

W efekcie $r_{BTD}(x) = r_{IEDT}(x) + r_{32}(x) + r_{40}(x) + r_{41}(x) = \langle 01001 \rangle$. Korzystając z tabeli 2.4. otrzymujemy ostatecznie $s_{BTD}(x) = \langle 01001 \rangle$.

Zauważmy, że przy założeniu braku katalogu reszt należy obliczyć jedynie niewielki zbiór reszt $r_{ji}(x)$. Podany przykład to potwierdza.

Przy określaniu stanu początkowego $g_c(x)$ można założyć, że wielomian $h_c(x)$ określający bity g_r wielomianu $g_c(x)$ jest resztą z poprzedniego dzielenia zrealizowanego w danym rejestrze c-MISR (c-SISR). Wówczas wszelkie wyrażenia określające poszczególne bity

sygnatury w funkcji wartości bitów reszty obowiązują równocześnie dla relacji pomiędzy bitami stanu początkowego a wartościami współczynników wielomianu $h_c(x)$ traktowanego jako reszta z poprzedniego dzielenia. Oznacza to, że w miejsce zmiennych s_r (np. tab. 2.3) można podstawić zmienne g_r , natomiast w miejsce zmiennych r_i (tab. 2.3) można podstawić zmienne h_i .

2.3.3. Określanie stanu początkowego dla danej sygnatury i sekwencji do niej doprowadzającej

Zagadnienie zasygnalizowane w nagłówku - to problem 2 rozwiązywany przy użyciu wyrażenia 2.9. Technikę wynikającą z tego wyrażenia wyjaśnia przykład.

Przykład 2.9

Zakładamy rejestr $\mathbf{D} \oplus \mathbf{DD} \oplus \mathbf{D}^{\oplus} \mathbf{D}$ związany z wielomianem charakterystycznym $p(x) = \langle 111101 \rangle$ (przykład 2.3). Zakładamy także następującą sygnaturę $s_{BTD}(x) = \langle 10111 \rangle$ oraz macierz (6 x 5) wyjściową UC w postaci następujących wielomianów:

$$\begin{aligned} u_0(x) &= \langle 001011 \rangle; & u_2(x) &= \langle 110111 \rangle; & u_4(x) &= \langle 110011 \rangle \\ u_1(x) &= \langle 111100 \rangle; & u_3(x) &= \langle 010111 \rangle; \end{aligned}$$

Należy określić stan początkowy $g_{BTD}(x)$ zawarty w rejestrze $\mathbf{D} \oplus \mathbf{DD} \oplus \mathbf{D}^{\oplus} \mathbf{D}$, który umożliwi ustawienie w tym rejestrze stanu $s_{BTD}(x) = \langle 10111 \rangle$ po podaniu na jego wejścia opisanej wyżej macierzy wyjściowej UC.

W związku z tym, że $h_{BTD}^+(x) = R[w_{BTD}^+(x)x + x^6 r_{BTD}^+(x), p^+(x)]$, należy najpierw ustalić następujące wielomiany biorące udział w dzieleniu:

$$p^+(x), w_{BTD}^+(x)x \text{ oraz } x^6 r_{BTD}^+(x).$$

Poniżej przedstawiono sposób ich obliczenia

$$\begin{aligned} p^+(x) &= [p(x)]^+ = [\langle 111101 \rangle]^+ = \langle 101111 \rangle, \\ w_{BTD}(x) &= u_0(x) + u_1(x)x + u_2(x)x^2 + u_3(x)x^3 + u_4(x)(1 + x + x^3 + x^4) = \\ &= \langle 1011101010 \rangle; \quad w_{BTD}^+(x) = \langle 0101011101 \rangle; \quad w_{BTD}^+(x)x = \langle 01010111010 \rangle; \\ r_{BTD}(x) &= \langle 11100 \rangle \text{ w oparciu o tab. 2.3 i } s_{BTD}(x) = \langle 10111 \rangle; \quad r_{BTD}^+(x) = \\ &= \langle 00111 \rangle; \quad x^6 r_{BTD}^+(x) = \langle 00111000000 \rangle. \end{aligned}$$

W efekcie otrzymujemy

$$[w_{BTD}^+(x)x + x^6 r_{BTD}^+(x)] = z(x) = \langle 01101111010 \rangle$$

Realizujemy dzielenie $z(x)/p^+(x)$

$$\begin{array}{r}
 z(x) \quad 01101111010 : 101111 \quad p^+(x) \\
 \underline{101111} \\
 110001 \\
 \underline{101111} \\
 111100 \\
 \underline{101111} \\
 100111 \\
 \underline{101111} \\
 10000 \quad h_{\text{BTD}}^+(x) = \langle 10000 \rangle
 \end{array}$$

Po odwróceniu $h_{\text{BTD}}^+(x) = \langle 10000 \rangle$ otrzymujemy $h_{\text{BTD}}(x) = \langle 00001 \rangle$. Podstawiając do tab. 2.4. (przykład 2.7) w miejsce zmiennych s_i zmienne h_i ze stanu $h_{\text{BTD}}(x) = \langle 00001 \rangle$ oraz w miejsce zmiennych r_i zmienne g_i otrzymujemy poszukiwany stan początkowy $g_{\text{BTD}}(x) = \langle 00001 \rangle$.

2.3.4. Określanie sekwencji wejściowej ustawiającej pożądaną sygnaturę w rejestrze liniowym z ustalonym stanem początkowym

W równaniach 2.10, które są podstawą rozwiązania postawionego problemu, znane są wielomiany $h_c(x)$ oraz $r_c(x)$. Należy określić wielomian $w_c(x)$. Zakładamy, że interesować nas będą najkrótsze sekwencje wejściowe o właściwościach podanych w nagłówku. Dla ułatwienia zadania zakładamy także dwa następujące rodzaje takich sekwencji:

- sekwencja szeregową podawana na zerowe wejście rejestru c-MISR (c-SISR),
- sekwencja równoległą podawana na wszystkie wejścia rejestru c-MISR.

W przypadku pierwszym sekwencję taką można opisać wielomianem $u_0(x)$ stopnia $t \leq n-1$ ($m \leq n$), natomiast w drugim przypadku będzie to pojedynczy wektor $w_0(x) = u_{00} + u_{10}x + u_{20}x^2 + \dots + u_{r0}x^r + \dots + u_{n-2,0}x^{n-2} + u_{n-1,0}x^{n-1}$ wpisany równoległe do n-wejściowego rejestru c-MISR jednym taktom zegarowym ($m = 1$).

Podstawiając do 2.10 w miejsce $w_c(x)$ założone sekwencje $u_0(x)$ (przykład a) oraz $w_0(x)$ (przykład b), otrzymujemy ostatecznie dwa zestawy następujących par równań:

$$\text{ad a) } u_0(x)/p(x) = q_w(x) + r_w(x)/p(x); \quad x^t h_c(x)/p(x) = q_h(x) + r_h(x)/p(x)$$

$$\text{ad b) } w_0(x)/p(x) = q'_w(x) + r'_w(x)/p(x); \quad x h_c(x)/p(x) = q'_h(x) + r'_h(x)/p(x)$$

W związku z tym, że $\deg u_0(x) = \deg w_0(x) = n-1$ ilorazy $q_w(x) = q'_w(x) = 0$.

Na tej podstawie otrzymujemy $u_0(x) = r_w(x)$ oraz $w_0(x) = r'_w(x)$.

W efekcie podstawiając $r_w(x) = u_0(x)$ do równania $r_c(x) = r_w(x) + r_h(x)$, otrzymujemy $u_0(x) = r_c(x) + r_h(x)$ (przykład a). Podobnie podstawiając $r'_w(x) = w_0(x)$ do równania $r_c(x) = r'_w(x) + r'_h(x)$, otrzymujemy $w_0(x) = r_c(x) + r'_h(x)$ (przykład b). Po obliczeniu reszty $r_h(x)$ lub reszty $r'_h(x)$ można określić wielomian $u_0(x)$ lub $w_0(x)$. Poniższy przykład wyjaśnia podaną metodę.

Przykład 2.10

Zakładamy następujący rejestr $\mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D}$ związany z wielomianem charakterystycznym $p(x) = \langle 111101 \rangle$ (przykład 2.3) oraz jego początkowy stan $g_{\text{BTD}}(x) = \langle 11000 \rangle$ i sygnaturę $s_{\text{BTD}}(x) = \langle 10010 \rangle$. Należy określić sekwencję $u_0(x)$ o długości $m = 4$, która umożliwi przejście ze stanu pierwszego do stanu drugiego.

W oparciu o tab. 2.3. i podane stany określamy następujące wielomiany:

$$h_{\text{BTD}}(x) = \langle 10011 \rangle \text{ i resztę } r_{\text{BTD}}(x) = \langle 11001 \rangle.$$

Należy zrealizować dzielenie $x^4 h_{\text{BTD}}(x)/p(x) = q_h(x) + r_h(x)/p(x)$

$$\begin{array}{r}
 x^4 h_{\text{BTD}}(x) \quad 10011 \mid 0000 \quad 111101 \quad p(x) \\
 \underline{11110} \quad 1 \\
 1111 \quad 01 \\
 \underline{10} \quad 1100 \\
 11 \quad 1101 \\
 \underline{1} \quad 0001
 \end{array}$$

Podstawiając $r_h(x) = \langle 10001 \rangle$ oraz $r_{\text{BTD}}(x) = \langle 11001 \rangle$ do równania $u_{\text{BTD}}(x) = r_w(x) + r_h(x)$, otrzymujemy $u_0(x) = r_w(x) + r_{\text{BTD}}(x) + r_h(x) = \langle 1000 \rangle$.

2.3.5. Określanie struktury rejestru liniowego c-LFSR, która przy założonym stanie poprzednim rejestru umożliwia przejście w pożądaną stan następnym

Jak określić strukturę rejestru c-LFSR, który umożliwia przejście z określonego stanu początkowego do pożądanego stanu końcowego po minimalnej liczbie m taktów zegarowych? Załóżmy wstępnie, że $m = 1$. Wówczas to zagadnienie można rozwiązać po przekształceniu 2.7 do postaci $xh_c(x) + r_c(x) = q_i p_c(x)$ ilustrującej pracę rejestru c-LFSR w jednym i -tym taktie zegarowym, po którym na wyjściu rejestru pojawia się jeden bit q_i ilorazu $q_c(x)$. Ze względu na to, że $q_i \in \{0,1\}$, należy rozpatrzeć dwa następujące przypadki powyższego równania:

$$xh_c(x) + r_c(x) = 0 \text{ (dla } q_i = 0) \text{ oraz } xh_c(x) + r_c(x) = p_c(x) \text{ (dla } q_i = 1).$$

Równania te po podstawieniu wyrażeń określających $h_c(x)$ oraz $r_c(x)$ i po przekształceniach przyjmują odpowiednio następujące postaci:

$$\sum_{r=0}^{n-1} \oplus (g_r x + s_r) p_{cr}(x) = 0 \quad \text{oraz} \quad \sum_{r=0}^{n-1} \oplus (g_r x + s_r) p_{cr}(x) = p_c(x)$$

które umożliwiają rozwiązanie problemu wymienionego w nagłówku tego podrozdziału. Sposób rozwiązania wyjaśnia przykład.

Przykład 2.11

Należy znaleźć strukturę rejestru liniowego o długości $n = 7$, który umożliwia przejście ze stanu $g(x) = 1101100$ w stan $s(x) = 1010001$ po jednym taktie zegarowym.

Zakładając, że poszukujemy rejestru IED-LFSR, otrzymujemy dwa równania

$$\sum_{r=0}^6 \oplus (g_r x + s_r) x^r = 0 \quad \text{oraz} \quad \sum_{r=0}^6 \oplus (g_r x + s_r) x^r = p_c(x)$$

z których pierwsze odrzucamy, ponieważ w założonym przejściu $q_1 = 1$. Podstawiamy więc do równania drugiego odpowiednie bity g_r oraz s_r z założonych stanów $g(x)$ oraz $s(x)$. W efekcie otrzymujemy wielomian

$$p_{IED}(x) = 1 + x^3 [1 + x^4] \text{ związany z rejestrem } D^3 \oplus D^4.$$

Zakładając poszukiwanie rejestru IEDT-LFSR należy wziąć pod uwagę, że

$$p_{cr}(x) = p_{IEDT}(x) = \prod_{j=0}^{r-1} (k_j + x)$$

Podstawiając wielomian $p_{cr}(x)$ do drugiego poprzednio wskazanego równania, otrzymujemy

$$\sum_{r=0}^6 \oplus (g_r x + s_r) \prod_{j=0}^{r-1} (k_j + x) = p_c(x)$$

Jeżeli przyjmiemy, że $k_j = 0$ dla wszystkich $j = 0, 1, 2, 4, 5, 6$ oraz że $k_3 = 1$, wówczas równanie to dla założonego przejścia daje w wyniku wielomian $p_{IEDT}(x) = 1 + x^6(1 + x) = 1 + x^3(1 + x)x^3$. W efekcie otrzymujemy rejestr $DDD(\oplus D)DDD$.

W przypadku braku możliwości znalezienia struktury rejestru liniowego c-LFSR umożliwiającego przejście po jednym taktie zegarowym ze stanu $g(x)$ w stan $s(x)$ można szukać takiej struktury rejestru c-LFSR, która umożliwi to przejście po minimalnej liczbie $m > 1$ taktów zegarowych. Poszukiwania te można prowadzić metodą prób i błędów. Wyjaśnia to przykład.

Przykład 2.12

Należy znaleźć strukturę rejestru liniowego o długości $n = 7$, który umożliwia przejście ze stanu $g(x) = 0101110$ w stan $s(x) = 0010011$ po minimalnej liczbie m taktów zegarowych.

Zakładamy rejestr $D^3 \oplus D^4$, w którym $r(x) = s(x)$ oraz $h(x) = g(x)$ oraz $m \leq 5$.

Realizujemy następujące dzielenie:

$$x^5 g(x) = x^5 h(x) \quad \begin{array}{r|l} 0101110 & 00000 \\ \hline 100010 & 01 \\ \hline 1100 & 0100 \\ \hline 1000 & 1001 \\ \hline 100 & 11010 \\ \hline 100 & 01001 \\ \hline 000 & 10011 \end{array} \quad \begin{array}{l} : 10001001 \\ p(x) \\ r(x) = s(x) \end{array}$$

Jego efektem jest liczba $m = 5$ taktów zegarowych.

W powyższym przykładzie można by kontynuować podobne dzielenia dla kolejnych założonych rejestrów c-LFSR i zakończyć ten proces albo po wyczerpaniu wszystkich możliwych struktur sprzężeń liniowych, albo po uzyskaniu zadowalającej liczby m . W pierwszym przypadku taki proces byłby zbyt pracochłonny. W drugim przypadku nie znana byłaby odległość uzyskanego rezultatu od rzeczywistego rozwiązania minimalnego. W efekcie rozwiązanie zagadnienia określania struktury rejestru c-LFSR, który po minimalnej liczbie m taktów zegarowych umożliwia przejście z określonego stanu początkowego do pożądanego stanu końcowego, jest problemem w dalszym ciągu oczekującym na rozwiązanie.

2.4. Rejestry liniowe przechowujące dowolne rozdzielne kody liniowe

Problem projektowania rejestrów liniowych przechowujących wyłącznie słowa kodowe należące do kodu C przewija się już w literaturze prawie od dwudziestu lat. Przykładem są prace [PradH78, Nebus83]. Ostatnio w [GupP92, GupP96] przedstawiono rejestr liniowy CMISR przechowujący słowa należące do nierozdzielonego kodu cyklicznego, natomiast w [SogG95, SogG96] opisano rejestr liniowy PMISA umożliwiający przechowywanie słów kodowych należących do rozdzielonego kodu liniowego zawierającego pojedynczy bit parzystości. W pracy [GoesS96] przedstawiono także metodę uzupełniania n -komórkowych

rejestrów MISR tzw. korektorem w postaci grupy k przerzutników przechowujących bity kontrolne rozdzielnego liniowego kodu podwojonego oraz rozdzielnych kodów nieliniowych arytmetycznego i Bergera. Rejestry liniowe HPMISA przechowujące dowolny kod Hamminga i należące do grupy rozdzielnych kodów liniowych opisano po raz pierwszy w [HłGS95]. Ideę tę rozszerzono na dowolne rozdzielne kody liniowe w [HłGS96a, HłGS96b], a następnie w [HłGS97]. Przedstawione w nich rejestry liniowe nazwano rejestrami COPMISR (ang. COde Preserving MISR). W pracach tych podano przykłady struktur rejestrów COPMISR przechowujących słowa z kodu Hamminga, kodu SEC-DED Hsiao (zmodyfikowany kod Hamminga), kodu podwojonego oraz kodu z bitami parzystości grupowej. Obecnie przedstawimy strukturę oraz algebraiczny opis rejestru liniowego COPMISR.

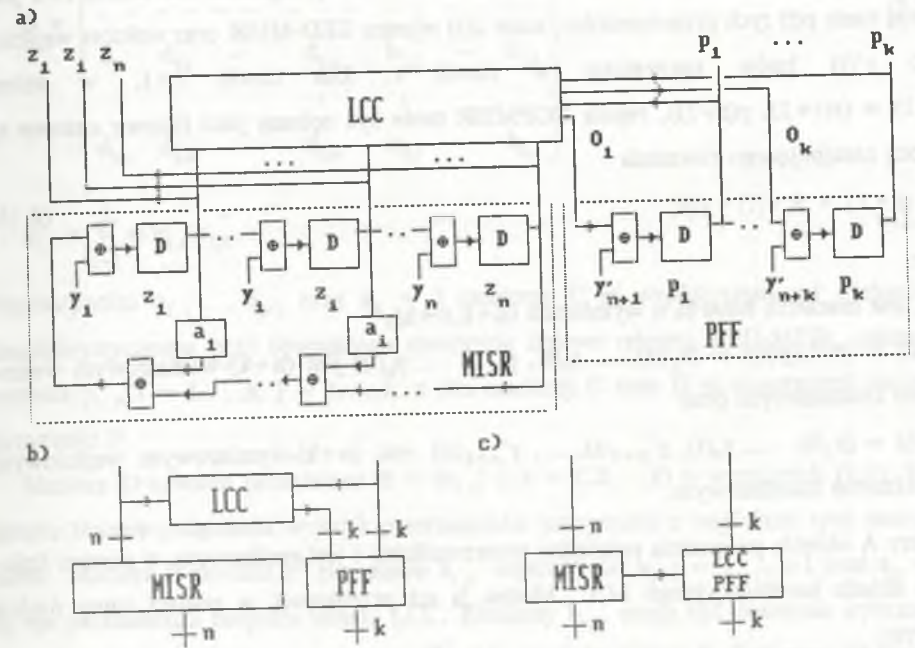
2.4.1. Rejestry liniowe COPMISR

Na wstępie kilka zdań przypominających wybrane pojęcia z teorii kodów liniowych [RaoF89]. Liniowy $(n+k, n)$ kod C nad ciałem $GF(2)$ może być zdefiniowany jako przestrzeń wierszowa macierzy generującej G o wymiarach $(n, n+k)$ lub jako $(k, n+k)$ -wymiarowa macierz H przestrzeni zerowej kodu [RaoF89]. Jeśli macierz generująca G jest przedstawiona w postaci systematycznej $G = (I_n, P)$, gdzie I_n jest (n, n) -wymiarową macierzą identycznościową oraz $P = [p_{ij}]$ jest (n, k) -macierzą parzystości, wówczas $H = [P^T, I_k]$. I_k jest (k, k) -wymiarową macierzą identycznościową, natomiast P^T jest transpozycją macierzy P . Jeżeli n informacyjnych bitów występuje w niezmielonej postaci w odpowiadającym jej słowie kodowym, wówczas taki kod nazywany jest kodem systematycznym (rozdzielnym) [PetW94]. W rozdzielnym kodzie liniowym $(n+k, n)$ słowo kodowe posiada $n+k$ bitów, w tym n bitów informacyjnych oraz k bitów kontrolnych. Słowo takiego kodu $y = (y_1, \dots, y_n, y'_{n+1}, \dots, y'_{n+k})$ otrzymuje się ze zbioru bitów informacyjnych y_1, \dots, y_n za pomocą następującego mnożenia $y = (y_1, \dots, y_n) G$, gdzie:

$$G = [I_n, P_{n,k}] = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 & p_{11} & p_{12} & \dots & p_{1k} \\ 0 & 1 & \dots & 0 & 0 & p_{21} & p_{22} & \dots & p_{2k} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 & 0 & p_{n-1,1} & p_{n-1,2} & \dots & p_{n-1,k} \\ 0 & 0 & \dots & 0 & 1 & p_{n,1} & p_{n,2} & \dots & p_{n,k} \end{pmatrix} \quad (2.15)$$

Bity parzystości $y'_{n+1}, \dots, y'_{n+k}$ określone są w sposób następujący:

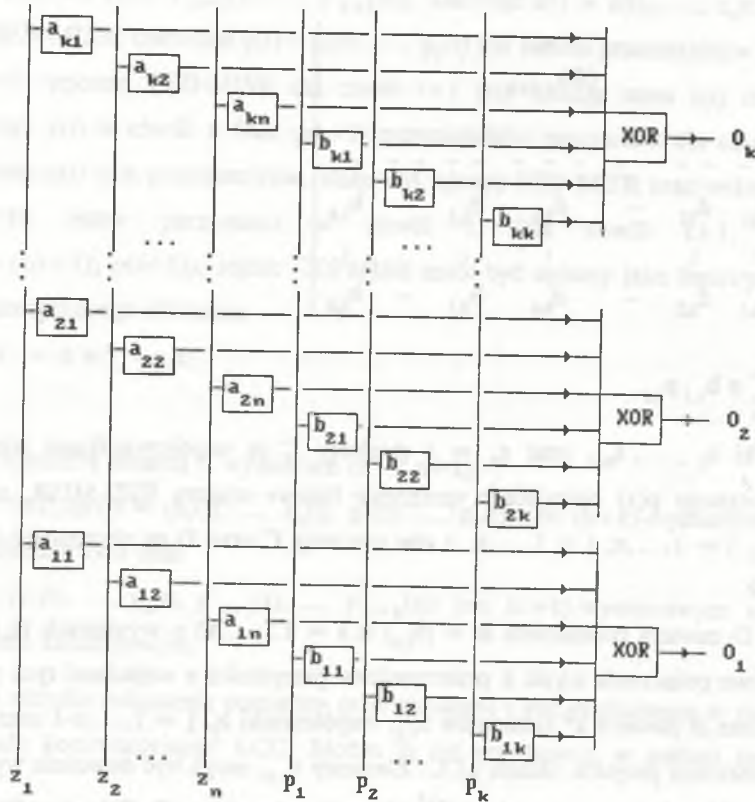
$$y'_{n+i} = \sum_{j=1}^n \oplus p_{ji} y_j \quad \text{dla } i = 1, \dots, k$$



Rys. 2.8. Ogólna struktura rejestru liniowego COPMISR (a) oraz jego dwa uproszczone schematy blokowe (b) i (c)

Fig. 2.8. General scheme of linear register COPMISR (a) and its two simplified schemes (b) and (c)

Ogólna struktura rejestru liniowego COPMISR przechowującego rozdzielny kod liniowy $(n+k, n)$ przedstawiona jest na rys. 2.8. Rejestr COPMISR jest liniowym układem sekwencyjnym [Elsa59, Kautz65, Gill67, Golom82]. Składa się ze zwyczajnego n -bitowego rejestru EED-MISR, k dodatkowych przerzutników parzystości PFF oraz dodatkowego liniowego układu kombinacyjnego LCC. Przerzutniki rejestru EED-MISR służą do przechowywania n bitów informacyjnych, natomiast k przerzutników PFF służy do przechowywania k bitów kontrolnych. Liniowy układ kombinacyjny LCC określa liniowe połączenia wyjść $n+k$ przerzutników rejestru COPMISR z wejściami k przerzutników parzystości. Wektor stanu oraz wektor wejściowy rejestru COPMISR oznaczone są dla chwili t przez $v(t) = (z(t), p(t)) = (z(t)_1, \dots, z_n(t), p_1(t), \dots, p_k(t))$ oraz odpowiednio przez



Rys. 2.9. Ogólna struktura liniowego układu kombinacyjnego LCC
 Fig. 2.9. General scheme of linear combinational circuit LCC

$$C = \begin{pmatrix} k_0 & 0 & 0 & \dots & 0 & a_0 \\ 1 & k_1 & 0 & \dots & 0 & a_1 \\ 0 & 1 & k_2 & \dots & 0 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & k_{n-2} & a_{n-2} \\ 0 & 0 & 0 & \dots & 1 & k_{n-1} + a_{n-1} \\ \hline c_{11} & c_{12} & c_{13} & \dots & c_{1,n-1} & c_{1,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{k,1} & c_{k,2} & c_{k,3} & \dots & c_{k,n-1} & c_{k,n} \end{pmatrix} \quad (2.22)$$

$$c_{i,j} = p_{j,i} k_{j-1} + p_{j+1,i} \quad \text{dla } i = 1, \dots, k; \quad j = 1, \dots, n-1 \quad (2.23)$$

$$c_{i,n} = \sum_{j=1}^n \oplus p_{j,i} a_{j-1} + k_{n-1} p_{n,i}$$

Dla każdego $(n+k)$ -wymiarowego wektora $v(t)$ wektor $C v(t)$ jest słowem kodowym z rozpatrywanego liniowego kodu $(n+k, n)$. Zauważmy także, że $D v(t) = 0$ dla każdego $v(t)$ będącego słowem kodowym.

Założmy, że wektor stanu $v(t)$ oraz wektor wejściowy $y(t)$ rejestru COPMISR są słowami kodowymi z rozpatrywanego rozdzielnego kodu liniowego $(n+k, n)$. Przy tym założeniu

$$v(t+1) = A v(t) + y(t) = C v(t) + D v(t) + y(t) = C v(t) + y(t)$$

co oznacza, że suma dwóch słów kodowych $C v(t)$ oraz $y(t)$ jest także słowem kodowym $v(t+1)$. Powyższe rozumowanie, przeprowadzone już wcześniej w [HłaGS96a, HłaGS96b, HłaGS97], jest dowodem na to, że przedstawiony rejestr COPMISR posiada zdolność przechowywania rozdzielnego kodu liniowego.

2.4.2. Rejestry liniowe PMISA

Jeśli ograniczymy liczbę bitów kontrolnych słowa kodowego do $k = 1$, to COPMISR staje się rejestrem PMISA [SogG95, SogG96]. Wówczas macierz generująca G przyjmuje następującą postać

$$G = [I_n, P] = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 & p_{11} \\ 0 & 1 & 0 & \dots & 0 & 0 & p_{21} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 & p_{n-1,1} \\ 0 & 0 & 0 & \dots & 0 & 1 & p_{n,1} \end{pmatrix} \quad (2.24)$$

gdzie $p_{j,1} = 1; j = 1, 2, \dots, n$.

Na tej podstawie macierz A staje się następującą $(n+1, n+1)$ -wymiarową macierzą [HłaGS95, HłaGS96b]:

$$A = \begin{bmatrix} a_1 & a_2 & a_3 & \dots & a_{n-1} & 1 & 0 \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \bar{a}_1 + b_{11} & \bar{a}_2 + b_{11} & \bar{a}_3 + b_{11} & \dots & \bar{a}_{n-1} + b_{11} & \bar{b}_{11} & b_{11} \end{bmatrix} \quad (2.25)$$

Macierz ta pozwala określić tylko dwa różne schematy rejestrów PMISA. Pierwszy z nich określony dla $b_{11} = 0$ okazuje się identyczny z rejestrem PMISA podanym w [SogG95, SogG96]. Natomiast drugi schemat określony dla $b_{11} = 1$ i różny od poprzedniego został przedstawiony w [HłaGS95, HłaGS96b].

3. PROJEKTOWANIE REJESTRÓW LINIOWYCH

Struktury sprzężeń rejestrów liniowych podawane są najczęściej w postaci tabel [Kalis77 (tab. 4.6), PieńT80 (tab. 4.7), Golom82, WangM88, XILINX94 (str. 9-24), ACTEL (str. 9-60)]. Zawierają one informacje określające, które komórki rejestru powinny posiadać wyprowadzenia połączone z bramkami XOR, aby opisywany rejestr liniowy związany był z pierwotnym wielomianem charakterystycznym [PetW94, Golom82, Hławi87c, Hławi88a]. Zamiast niezbyt wygodnych i często niedostępnych tabel lepszym rozwiązaniem byłaby prosta technika projektowania rejestrów liniowych. Pewnym przykładem takiej techniki są rejestry IED-LFSR (IED-MISR), których schemat można właściwie odczytywać wprost z postaci źródłowego wielomianu charakterystycznego [Golom82, Hławi87c, Hławi88a]. Brakuje uniwersalnej i nieskomplikowanej zarazem techniki projektowania rejestrów liniowych o różnych strukturach sprzężeń liniowych związanych z założonym źródłowym wielomianem charakterystycznym $p(x)$. Rozwiązanie tego problemu przedstawia niniejszy rozdział.

3.1. Projektowanie rejestrów liniowych o różnych strukturach sprzężenia liniowego

Projektowanie rejestrów LFSR, SISR oraz MISR przy użyciu przerzutników D, bramek XOR oraz w szczególnych przypadkach także przy użyciu przerzutników T wymaga określenia źródłowego wielomianu $p(x)$, jaki powinien charakteryzować sprzężenie liniowe projektowanego rejestru. Będzie nim najczęściej jeden z wielomianów pierwotnych wybrany np. z katalogu podanego w [PetW94]. Następnie należy podjąć decyzję, która ze struktur

sprzężenia liniowego $c \in \{IEDT, IED, EEDT, EED, DT, BTD, TBD, CADT\}$ ma być wprowadzona do projektowanego rejestru liniowego. Ta decyzja umożliwi wybór tej postaci wielomianu charakterystycznego $p_c(x)$ (tab. 2.1), która jest związana z wybraną konfiguracją sprzężenia liniowego.

Projektowanie w pierwszej swojej fazie polegać więc będzie na takim przekształceniu założonego źródłowego wielomianu $p(x)$, aby uzyskać tę postać wielomianu charakterystycznego $p_c(x)$, która związana jest z założoną strukturą projektowanego rejestru. Podobną ideę projektowania ograniczoną jednak wyłącznie do rejestrów TBD-LFSR oraz BTD-LFSR związanych z niektórymi pierwotnymi wielomianami charakterystycznymi przedstawiono w pracy [WangM88]. Ideę tę rozszerzono na wszystkie wielomiany charakterystyczne i na rejestry TBD-SISR i TBD-MISR w [Hławi89a, Hławi92b] oraz na rejestry BTD-SISR i BTD-MISR w [Hławi89b, Hławi92b]. Nie pozwala ona jednak odczytywać schematu połączeń rejestru liniowego wprost z postaci wielomianu $p_c(x)$. Dobrym tego przykładem jest wielomian $p_{TBD}(x) = 1 + (x^3 + x^4 + x^7 + x^8)(1 + x^2)$ związany z rejestrem $D^{\oplus}DDD^{\oplus}D^{\oplus}DDD^{\oplus}DD$. W celu podania schematu połączeń struktury sprzężenia tego rejestru liniowego autorzy pracy [WangM88] proponują korzystanie ze specjalnego wielomianu $1 \uparrow x \uparrow x^4 \uparrow x^5 \uparrow x^8 \uparrow x^{10}$. Cały katalog tak opisanych struktur sprzężeń liniowych rejestrów TBD-LFSR o różnej długości znajduje się w [WangM88]. Tej wady nie posiada technika projektowania rejestrów IEDT podana w [Hławi92a]. Niestety jest ona ograniczona tylko do grupy rejestrów liniowych IETD, IED, IET, DT. W pracy [Hławi92a] źródłowy wielomian charakterystyczny $p(x)$ przekształca się do postaci $p_c(x)$, a następnie do tzw. strukturalnej postaci $p_{cw}(x)$, która jest ściśle skorelowana ze schematem ideowym projektowanego rejestru liniowego i z której wprost można odczytać schemat ideowy tego rejestru. Tę technikę projektowania wykorzystano w niniejszej pracy, rozszerzając jej zastosowanie na konfiguracje sprzężeń liniowych EEDT, BTD, TBD i sprzężenia pochodne.

Iteracyjny sposób tworzenia wielomianu $p_{CA}(x)$ znacznie utrudnia znalezienie prostych związków pomiędzy niezerowymi współczynnikami a_i wielomianu $p(x)$ a współczynnikami k_j związanego z tym wielomianem rejestru CADT [Hławi90d, Hławi93a]. Częściowo z tym problemem uporano się w [SerrS90, CattM96], gdzie opisano algorytm projektowania rejestrów CADT na podstawie zadanego charakterystycznego wielomianu. Metoda ta jest jednak bardzo skomplikowana, co w efekcie spowodowało, że do projektowania rejestrów CADT autor zastosował zupełnie inną technikę [Hławi91b, Hławi92a, Hławi93a].

3.1.1. Projektowanie rejestrów o strukturze IEDT, IED oraz IET

Wielomian charakterystyczny o postaci (tab. 2.1)

$$P_{IEDT}(x) = P_0 + \sum_{r=1}^n \oplus P_r \prod_{j=0}^{r-1} (k_j + x)$$

opisuje strukturę IEDT [Hławi92a] rejestru liniowego. Założmy, że współczynniki tego wielomianu P_0, P_r, P_s, P_t, P_n są różne od zera, przy czym $n > t > s > r > 0$. Przy założeniu konkretnych wartości 0 oraz 1 dla współczynników k_j oraz zakładając $b_1 + c_1 = r, b_2 + c_2 = s - r, b_3 + c_3 = t - s$ oraz $b_4 + c_4 = n - t$ podany wielomian można przekształcić do następującej postaci:

$$P_{IEDT}(x) = P_0 + x^{b_1}(1+x)^{c_1} \left[P_r + x^{b_2}(1+x)^{c_2} \left[P_s + x^{b_3}(1+x)^{c_3} \left[P_t + x^{b_4}(1+x)^{c_4} P_n \right] \right] \right],$$

w której $\sum_{j=1}^4 (b_j + c_j) = n$. Wprowadzając w miejsce współczynników P_0, P_r, P_s, P_t, P_n jedynki, otrzymujemy ostatecznie następującą formę wielomianu

$$P_{IEDT}(x) = 1 + x^{b_1}(1+x)^{c_1} \left[1 + x^{b_2}(1+x)^{c_2} \left[1 + x^{b_3}(1+x)^{c_3} \left[1 + x^{b_4}(1+x)^{c_4} \right] \right] \right] \quad (3.1)$$

Postać ta uzyskana po raz pierwszy w [Hławi92a] nazywana będzie w dalszej części pracy postacią strukturalną wielomianu charakterystycznego związaną z rejestrem liniowym o strukturze IEDT. Jednoznacznie skorelowany z tym wielomianem rejestr liniowy np. IEDT-LFSR można zapisać w następujący sposób $D^{b_1}(\oplus'D)^{c_1} \oplus D^{b_2}(\oplus'D)^{c_2} \oplus D^{b_3}(\oplus'D)^{c_3} \oplus D^{b_4}(\oplus'D)^{c_4}$. Wprowadzając w miejsce komórek $(\oplus'D)$ komórki T otrzymujemy rejestr $D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}T^{c_4}$ [Hławi92a]. Zakładając, że tylko część komórek $(\oplus'D)$ zostaje zastąpiona komórkami T, można otrzymać np. taki rejestr $D^{b_1}T^{c_1} \oplus D^{b_2}(\oplus'D)^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}(\oplus'D)^{c_4}$. Wszystkie te trzy rejestry posiadają bramki XOR wewnętrznego sprzężenia liniowego na wyjściu r-tej, s-tej i t-tej komórki D, T lub $(\oplus'D)$, co ściśle skorelowane jest z pozycjami kolejnych lewych nawiasów kwadratowych w wielomianie 3.1. Im mniej tych nawiasów kwadratowych, tym mniejszej liczby bramek XOR należy użyć do realizacji liniowego sprzężenia wewnętrznego. Zmniejszając liczbę komórek $(\oplus'D)$ lub zastępując je przerzutnikami T, również wpływamy na zmniejszanie liczby bramek XOR w tym sprzężeniu liniowym. Wyjaśnia to poniższy przykład.

Przykład 3.1

Zakładamy źródłowy pierwotny wielomian charakterystyczny $p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$. Zaprojektować rejestr IETD-LFSR- $p(x)$. W tym celu należy przekształcić wielomian $p(x)$ do postaci $P_{IETD}(x)$, a następnie do postaci strukturalnej $P_{IETDw}(x)$ (3.1). Rezultat jednego z przekształceń:

$$\begin{aligned} \text{a) } P_{IETD}(x) &= 1 + x^3(1+x)^4(1+x+x^2+x^3) \\ P_{IETDw}(x) &= 1 + x^3(1+x)^4[1+x[1+x[1+x]]] \cong \\ &\cong \text{DDD}(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D) \oplus D \oplus D \oplus D \cong D^3T^4 \oplus D \oplus D \oplus D \cong \\ &\cong T^2D^3T^2 \oplus D \oplus D \oplus D \end{aligned}$$

Stosując niektóre z innych przekształceń tego wielomianu, otrzymujemy odpowiednio

$$\begin{aligned} \text{b) } P_{IETD}(x) &= 1 + x^3(1+x)(1+x^2+x^4+x^6) \\ P_{IETDw}(x) &= 1 + x^3(1+x)[1+x^2[1+x^2[1+x^2]]] \cong \\ &\cong \text{DDD}(\oplus'D) \oplus \text{DD} \oplus \text{DD} \oplus \text{DD} \cong D^3T \oplus D^2 \oplus D^2 \oplus D^2, \end{aligned}$$

$$\begin{aligned} \text{c) } P_{IETD}(x) &= 1 + x^3(1+x)^3(1+x^4) \\ P_{IETDw}(x) &= 1 + x^3(1+x)^3[1+x^4] \cong \text{DDD}(\oplus'D)(\oplus'D)(\oplus'D) \oplus D^4 \cong D^3T^3 \oplus D^4. \end{aligned}$$

Zakładając $k_j = 0$, otrzymujemy z wielomianu $P_{IETD}(x)$ następującą postać strukturalną [Hławi92a]

$$P_{IEDw}(x) = 1 + x^r[1 + x^{s-r}[1 + x^{t-s}[1 + x^{n-t}]]] \quad (3.2)$$

natomiast zakładając $k_j = 1$ otrzymujemy [Hławi92a]

$$P_{IETw}(x) = 1 + (1+x)^r[1 + (1+x)^{s-r}[1 + (1+x)^{t-s}[1 + (1+x)^{n-t}]]] \quad (3.3)$$

Wielomianom tym odpowiadają następujące schematy ideowe rejestrów odpowiednio IED-MISR oraz IET-MISR:

$$D^r \oplus D^{s-r} \oplus \dots \oplus D^{n-t} \cong P_{IEDw}(x) \text{ oraz } T^r \oplus T^{s-r} \oplus \dots \oplus T^{n-t} \cong P_{IETw}(x).$$

Przykład 3.2

Zaprojektować rejestr IED-LFSR- $p(x)$, a następnie IET-LFSR- $p(x)$, gdzie

$$p(x) = 1 + x^2 + x^3 + x^4 + x^5.$$

$$P_{IEDw}(x) = 1 + x^2[1 + x[1 + x[1 + x]]] \cong \text{DD} \oplus D \oplus D \oplus D$$

$$p(x) = 1 + x^2(1+x) + x^4(1+x) = 1 + (1+x)[x^2(1+x)^2]$$

$$P_{IETw}(x) = 1 + (1+x)^3[1 + (1+x)^2] \cong \text{TTT} \oplus \text{TT}$$

W pracy [Hławi92a] udowodniono, że dla wielomianów o parzystej liczbie niezerowych współczynników nie można zaprojektować rejestrów o strukturach typu IET.

3.1.2. Projektowanie rejestrów o strukturze EEDT, EED oraz EET

Rejestrowi liniowemu o strukturze EEDT odpowiada poniższy wielomian charakterystyczny (tab. 2.1)

$$P_{EEDT}(x) = \sum_{i=0}^{n-1} \oplus P_i \prod_{r=1}^{n-1} (k_r + x) + P_n$$

Przyjmujemy, że współczynniki p_0, p_r, p_s, p_t, p_n w tym wielomianie są różne od zera, przy czym $n > t > s > r > 0$. Przy założeniu konkretnych wartości 0 oraz 1 dla współczynników k oraz zakładając $b_1 + c_1 = r, b_2 + c_2 = s - r, b_3 + c_3 = t - s$ oraz $b_4 + c_4 = n - t$, otrzymujemy wielomian

$$P_{EEDT}(x) = P_n + x^{b_4}(1+x)^{c_4} [p_t + x^{b_3}(1+x)^{c_3} [p_s + x^{b_2}(1+x)^{c_2} [p_r + x^{b_1}(1+x)^{c_1} p_0]]]$$

w którym $n = \sum_{j=1}^4 (b_j + c_j)$. Podstawiając jedynki w miejsce współczynników p_0, p_r, p_s, p_t, p_n otrzymujemy następującą postać strukturalną

$$P_{EEDTw}(x) = 1 + x^{b_4}(1+x)^{c_4} [1 + x^{b_3}(1+x)^{c_3} [1 + x^{b_2}(1+x)^{c_2} [1 + x^{b_1}(1+x)^{c_1}]]] \quad (3.4)$$

Skorelowany z $P_{EEDT}(x)$ rejestr liniowy na przykład EEDT-MISR przedstawia następujący ciąg znaków $D^{b_1}(\oplus'D)^{c_1} \oplus D^{b_2}(\oplus'D)^{c_2} \oplus D^{b_3}(\oplus'D)^{c_3} \oplus D^{b_4}(\oplus'D)^{c_4}$. Podstawiając w miejsce komórek $(\oplus'D)$ komórki T , otrzymujemy rejestr $D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}T^{c_4}$. Oba te rejestry posiadają bramki XOR zewnętrznego sprzężenia liniowego na wyjściu r -tej, s -tej i t -tej komórki D, T lub $(\oplus'D)$, co ściśle skorelowane jest z pozycjami kolejnych, liczonych od prawej strony, lewych nawiasów kwadratowych w wielomianie 3.4.

Zwróćmy uwagę, że jeżeli $P_{EEDTw}(x) = P_{IETTw}(x) =$

$$= 1 + x^{b_1}(1+x)^{c_1} [1 + x^{b_2}(1+x)^{c_2} [1 + x^{b_3}(1+x)^{c_3} [1 + x^{b_4}(1+x)^{c_4}]]]$$

to ciąg komórek $T^{c_4}D^{b_4} \oplus T^{c_3}D^{b_3} \oplus T^{c_2}D^{b_2} \oplus T^{c_1}D^{b_1}$ rejestru o strukturze EEDT można określić za pomocą odwróconego ciągu komórek $D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}T^{c_4}$ rejestru o strukturze IEDT.

Przykład 3.3

Zaprojektować trzy różne rejestry EETD-LFSR- $p(x)$, gdzie $p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$.

- a) $p(x) = 1 + x^3(1+x)^4 [1 + x[1 + x[1 + x]]] \cong [D^{\oplus}D^{\oplus}D^{\oplus}D^{\oplus}(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D)DDD] \cong [D^{\oplus}D^{\oplus}D^{\oplus}D^{\oplus}T^4D^3]$,
- b) $p(x) = 1 + x^3(1+x)[1 + x^2 [1 + x^2 [1 + x^2]]] \cong [DD^{\oplus}DD^{\oplus}DD^{\oplus}(\oplus'D)DDD] \cong [D^2\oplus D^2\oplus D^2\oplus TD^3]$,
- c) $p(x) = 1 + x^3(1+x)^3[1 + x^4] \cong [DDDD^{\oplus}(\oplus'D)(\oplus'D)(\oplus'D)DDD]$

Zauważmy, że w każdym z trzech przypadków a), b), c) przykładu 3.3 wielomian $P_{EEDT_w}(x) = P_{IEDT_w}(x)$. W efekcie ciągi komórek rejestrów EEDT-LFSR z przykładu 3.3 są budowane z odwróconych ciągów komórek rejestrów IEDT-LFSR podanych w przykładzie 3.1. Przykłady rejestrów EEDT ilustrujących odwrócone ciągi komórek podano po raz pierwszy w [Hławi92a].

Przy założeniu $k_j = 0$ wielomian $p_{EEDT}(x)$ przyjmuje następującą postać strukturalną

$$P_{EEDT_w}(x) = 1 + x^{n-t} [1 + x^{t-s} [1 + x^{s-r} [1 + x^r]]] \quad (3.5)$$

Zakładając $k_j = 1$, otrzymujemy inną postać strukturalną

$$P_{EET_w}(x) = 1 + (1+x)^{n-t} [1 + (1+x)^{t-s} [1 + (1+x)^{s-r} [1 + (1+x)^r]]] \quad (3.6)$$

Wielomianom tym odpowiadają następujące schematy ideowe rejestrów

$D^r \oplus D^{s-r} \oplus D^{t-s} \oplus D^{n-t}$ (EED-MISR) oraz $T^r \oplus T^{s-r} \oplus T^{t-s} \oplus T^{n-t}$ (EET-MISR).

Przykład 3.4

Zaprojektować rejestr EED-LFSR- $p(x)$ oraz EET-LFSR- $p(x)$, gdzie

$$p(x) = 1 + x^2 + x^3 + x^4 + x^5$$

$$p(x) = 1 + x^2 [1 + x [1 + x [1 + x]]] \cong D^{\oplus}D^{\oplus}D^{\oplus}DD$$

$$p(x) = 1 + (1+x)^3 [1 + (1+x)^2] \cong TT^{\oplus}TTT.$$

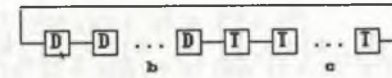
3.1.3. Projektowanie rejestrów o strukturze FSR

Źródłowy wielomian charakterystyczny $p(x)$ można w wielu przypadkach przekształcić do postaci

$$P_{DT_w}(x) = 1 + x^b (1+x)^c \quad (3.7)$$

w której $b + c = n$ oraz w której nie ma żadnego nawiasu kwadratowego. Jest to najprostsza postać wielomianu 3.1 lub 3.4, którą w wielu przypadkach można uzyskać. Po raz pierwszy została ona opisana w pracy [Hławi92a]. Na rys. 3.1 przedstawiono schemat ideowy rejestru DT-LFSR ze sprzężeniem liniowym opisanym za pomocą wielomianu 3.7.

Przy zastosowaniu uproszczonej symboliki schemat ten można przedstawić za pomocą następujących ciągów znaków D^bT^c , $D^b(\oplus'D)^c$ lub $D^bT^{c-a}(\oplus'D)^a$.



Rys. 3.1. Schemat ideowy rejestru liniowego DT-LFSR
Fig. 3.1. Scheme of linear register DT-LFSR

Przykład 3.5

Zaprojektować rejestr DT-LFSR- $p(x)$, gdzie

$$p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}.$$

$$P_{DT_w}(x) = 1 + x^3 (1+x)^7 \cong DDDTTTTTTT \cong DTTDTTDTTT.$$

Schemat ideowy zaprojektowanego rejestru jest ciągiem znaków D^3T^7 nie przedzielonych żadną bramką XOR. Jeżeli zastąpimy komórki T komórkami $(\oplus'D)$, wówczas otrzymamy rejestr $D^3(\oplus'D)^7 \cong P_{DT_w}(x)$ zawierający 7 bramek XOR.

Jeżeli rejestr liniowy będzie złożony wyłącznie z komórek T nie przedzielonych żadną bramką XOR, wówczas jego wielomian charakterystyczny przyjmuje postać

$$P_{T_w}(x) = 1 + (1+x)^n \cong TT^{n-1} \text{ lub } T^n \quad (3.8)$$

Są to rejestry odpowiednio typu T-SISR i T-MISR opisane po raz pierwszy w [Hławi92a].

Jeżeli zamiast komórek T, T w tym rejestrze stosujemy komórki D, D , wówczas otrzymujemy konwencjonalny opisany w [David78, David80] rejestr D-SISR lub przedstawiony w [David85, David86] rejestr D-MISR - oba o wielomianie charakterystycznym

$$P_{D_w}(x) = 1 + x^n \cong DD^{n-1} \text{ lub } D^n \quad (3.9)$$

3.1.4. Projektowanie rejestrów o strukturze TBDT oraz TBD

Zakładamy rejestr o strukturze TBDT, któremu odpowiada następująca postać wielomianu charakterystycznego (tab. 2.1)

$$P_{TBDT}(x) = 1 + \left[\sum_{r=0}^{i-1} \oplus p_r \prod_{j=r}^{i-1} (k_j + x) \right] \left[p_i + \sum_{r=i+1}^n \oplus p_r \prod_{j=i}^{r-1} (k_j + x) \right]$$

Zakładamy jednocześnie, że nie tylko współczynniki p_0 , p_1 oraz p_n , ale także współczynniki p_r , p_s , p_t oraz p_u tego wielomianu są różne od zera.

Przekształcając wielomian $p_{\text{TBDT}}(x)$, otrzymujemy:

$$p_{\text{TBDT}}(x) = 1 + \left[p_0 \prod_{j=0}^{i-1} (k_j + x) + p_r \prod_{j=r}^{i-1} (k_j + x) + p_s \prod_{j=s}^{i-1} (k_j + x) \right] \\ \left[p_i + p_t \prod_{j=i}^{t-1} (k_j + x) + p_u \prod_{j=i}^{u-1} (k_j + x) + p_n \prod_{j=i}^{n-1} (k_j + x) \right]$$

gdzie: $t = i + t'$; $u = i + u'$; $n = i + n'$ oraz $n > u > t > i > s > r > 0$.

Wielomian ten można także zapisać w następującej postaci:

$$p_{\text{TBDT}}(x) = 1 + \left[\prod_{j=s}^{i-1} (k_j + x) \left[p_s \prod_{j=r}^{s-1} (k_j + x) \left[p_r \prod_{j=0}^{r-1} (k_j + x) p_0 \right] \right] \right] \\ \left[p_i + \prod_{j=i}^{t-1} (k_j + x) \left[p_t + \prod_{j=i}^{u-1} (k_j + x) \left[p_u + \prod_{j=u}^{n-1} (k_j + x) p_n \right] \right] \right]$$

Przy założeniu konkretnych wartości 0 oraz 1 dla współczynników k_j oraz zakładając $b_1 + c_1 = r$, $b_2 + c_2 = s - r$, $b_3 + c_3 = i - s$, $b_4 + c_4 = t - i$, $b_5 + c_5 = u - t$, $b_6 + c_6 = n - u$ oraz $\sum_{j=1}^6 (b_j + c_j) = n$, otrzymujemy wielomian

$$p_{\text{TBDT}}(x) = 1 + \left[x^{b_3} (1+x)^{c_3} \left[p_s + x^{b_2} (1+x)^{c_2} \left[p_r + x^{b_1} (1+x)^{c_1} \right] \right] \right] \\ \left[p_i + x^{b_4} (1+x)^{c_4} \left[p_t + x^{b_5} (1+x)^{c_5} \left[p_u + x^{b_6} (1+x)^{c_6} \right] \right] \right]$$

Wpisując jedynki w miejsce współczynników p_0 , p_r , p_s , p_i , p_t , p_u , p_w , oraz p_n , otrzymujemy ostatecznie następującą postać strukturalną

$$p_{\text{TBDTw}}(x) = 1 + \left[x^{b_3} (1+x)^{c_3} \left[1 + x^{b_2} (1+x)^{c_2} \left[1 + x^{b_1} (1+x)^{c_1} \right] \right] \right] \\ \left[1 + x^{b_4} (1+x)^{c_4} \left[1 + x^{b_5} (1+x)^{c_5} \left[1 + x^{b_6} (1+x)^{c_6} \right] \right] \right] \quad (3.10)$$

Skorelowany z tym wielomianem strukturalnym ciąg znaków

$D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}T^{c_4} \oplus D^{b_5}T^{c_5} \oplus D^{b_6}T^{c_6}$ przedstawia rejestr TBDT-LFSR.

Zastępując niektóre przerzutniki T strukturą $(\oplus'D)$, można uzyskać inny rejestr

$D^{b_1}T^{c_1} \oplus D^{b_2}(\oplus'D)T^{c_2-1} \oplus D^{b_3}T^{c_3} \oplus D^{b_4}(\oplus'D)^{c_4} \oplus D^{b_5}T^{c_5} \oplus D^{b_6}(\oplus'D)^{c_6}$.

Przykład 3.6

Zaprojektować trzy różne rejestry TBDT-LFSR- $p(x)$, gdzie

$$p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}.$$

Należy przekształcić wielomian $p(x)$ do postaci $p_{\text{TBDTw}}(x)$.

$$a) p_{\text{TBDT}}(x) = 1 + x^3 (1+x^2)(1+x+x^4+x^5)$$

$$p_{\text{TBDTw}}(x) = 1 + [x^3 [1+x^2]][1+x[1+(1+x)x^3]] \cong DD^{\oplus}DDD_{\oplus}D_{\oplus}TDDD \cong \\ \cong DD^{\oplus}DDD_{\oplus}D_{\oplus}DDTD \cong DD^{\oplus}DDD_{\oplus}D_{\oplus}DD(\oplus'D)D$$

$$b) p_{\text{TBDT}}(x) = 1 + x^3 (1+x)(1+x^2+x^4+x^6)$$

$$p_{\text{TBDTw}}(x) = 1 + [x^3 [1+x]][1+x^2[1+(1+x)^2x^2]] \cong D^{\oplus}DDD_{\oplus}DD_{\oplus}TTDD \cong \\ \cong D^{\oplus}DDD_{\oplus}DD_{\oplus}TDTD \cong D^{\oplus}DDD_{\oplus}DD_{\oplus}TD(\oplus'D)D$$

$$c) p_{\text{TBDT}}(x) = 1 + x^3 (1+x^2+x^4+x^6)(1+x)$$

$$p_{\text{TBDTw}}(x) = 1 + [x^3 [1+x^2[1+(1+x)^2x^2]]][1+x] \cong DDTT^{\oplus}DD^{\oplus}DDD_{\oplus}D \cong \\ \cong DTD^{\oplus}DD^{\oplus}DDD_{\oplus}D \cong D(\oplus'D)DT^{\oplus}DD^{\oplus}DDD_{\oplus}D.$$

Zakładając $k_j = 0$, otrzymujemy z 3.10 następującą postać strukturalną

$$p_{\text{TBDw}}(x) = 1 + x^{i-s} [1 + x^{s-r} [1 + x^r]] [1 + x^{t-i} [1 + x^{u-t} [1 + x^{n-u}]]] \quad (3.11)$$

Uproszczony zapis schematu rejestru TBD-LFSR związanego z tym wielomianem przedstawia następujący ciąg znaków $D^{r\oplus} D^{s-r\oplus} D^{i-s} \oplus D^{t-i} \oplus D^{u-t} \oplus D^{n-u}$.

Przykład 3.7

Zaprojektować trzy różne rejestry TBD-LFSR- $p(x)$, gdzie

$$p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}.$$

Zauważmy, że $p_{\text{TBDT}}(x) = p_{\text{TBD}}(x)$ w każdym z trzech przypadków a), b), c) przykładu 3.6. W efekcie otrzymujemy

$$a) p_{\text{TBDw}}(x) = 1 + [x^3 [1+x^2]][1+x[1+x^3[1+x]]] \cong DD^{\oplus}DDD_{\oplus}D_{\oplus}DDD_{\oplus}D$$

$$b) p_{\text{TBDw}}(x) = 1 + [x^3 [1+x]][1+x^2[1+x^2[1+x^2]]] \cong D^{\oplus}DDD_{\oplus}DD_{\oplus}DD_{\oplus}DD$$

$$c) p_{\text{TBDw}}(x) = 1 + [x^3 [1+x^2[1+x^2[1+x^2]]]][1+x] \cong DD^{\oplus}DD^{\oplus}DD^{\oplus}DDD_{\oplus}D.$$

3.1.5. Projektowanie rejestrów o strukturze BTDT oraz BTDD

Zakładamy rejestr o strukturze BTDT, któremu odpowiada następująca postać wielomianu charakterystycznego (tab. 2.1)

$$p_{\text{BTDT}}(x) = \prod_{j=0}^{n-1} (k_j + x) + \left[p_0 + \sum_{r=1}^i \oplus p_r \prod_{j=0}^{r-1} (k_j + x) \right] \left[p_n + \sum_{r=i+1}^{n-1} \oplus p_r \prod_{j=r}^{n-1} (k_j + x) \right]$$

Zakładamy także, że oprócz współczynników p_0 , p_i oraz p_n również współczynniki p_r , p_s , p_t , p_u oraz p_w tego wielomianu są różne od zera.

Przekształcając wielomian $p_{\text{BTDT}}(x)$, otrzymujemy:

$$p_{\text{BTDT}}(x) = \prod_{j=0}^{n-1} (k_j + x) + \left[p_0 + p_r \prod_{j=0}^{r-1} (k_j + x) + p_s \prod_{j=0}^{s-1} (k_j + x) + p_i \prod_{j=0}^{i-1} (k_j + x) \right] \\ \left[p_n + p_w \prod_{j=w}^{n-1} (k_j + x) + p_u \prod_{j=u}^{n-1} (k_j + x) + p_t \prod_{j=t}^{n-1} (k_j + x) \right]$$

gdzie: $n > w > u > t > i > s > r > 0$. Wielomian ten można także zapisać w następującej postaci:

$$p_{\text{BTDT}}(x) = \prod_{j=0}^{n-1} (k_j + x) + \left[p_0 + \prod_{j=0}^{r-1} (k_j + x) \right] \left[p_r + \prod_{j=r}^{s-1} (k_j + x) \right] \left[p_s + \prod_{j=s}^{i-1} (k_j + x) p_i \right] \\ \left[p_n + \prod_{j=w}^{n-1} (k_j + x) \right] \left[p_w + \prod_{j=u}^{w-1} (k_j + x) \right] \left[p_u + \prod_{j=t}^{u-1} (k_j + x) p_t \right]$$

Przy założeniu konkretnych wartości 0 oraz 1 dla współczynników k_j oraz zakładając $b_1+c_1 = r$, $b_2+c_2 = s-r$, $b_3+c_3 = i-s$, $b_4+c_4 = u-t$, $b_5+c_5 = w-u$, $b_6+c_6 = n-w$ oraz $c = \sum_{j=1}^6 c_j$, otrzymujemy wielomian

$$p_{\text{BTDT}}(x) = x^{n-c} (1+x)^c + \left[p_0 + x^{b_1} (1+x)^{c_1} \left[p_r + x^{b_2} (1+x)^{c_2} \left[p_s + x^{b_3} (1+x)^{c_3} p_i \right] \right] \right] \\ \left[p_n + x^{b_6} (1+x)^{c_6} \left[p_w + x^{b_5} (1+x)^{c_5} \left[p_u + x^{b_4} (1+x)^{c_4} p_t \right] \right] \right]$$

W efekcie postać strukturalna

$$p_{\text{BTDT}_w}(x) = x^{n-c} (1+x)^c + \left[1 + x^{b_1} (1+x)^{c_1} \left[1 + x^{b_2} (1+x)^{c_2} \left[1 + x^{b_3} (1+x)^{c_3} \right] \right] \right] \\ \left[1 + x^{b_6} (1+x)^{c_6} \left[1 + x^{b_5} (1+x)^{c_5} \left[1 + x^{b_4} (1+x)^{c_4} \right] \right] \right] \quad (3.12)$$

Zauważmy, że $n = \sum_{j=1}^6 (b_j + c_j) + (t-i)$.

Skorelowany z wielomianem 3.12 ciąg komórek

$D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{t-i} \oplus D^{b_4}T^{c_4} \oplus D^{b_5}T^{c_5} \oplus D^{b_6}T^{c_6}$ ilustruje rejestr liniowy BTDT-LFSR. Zastępując niektóre przerzutniki T strukturą $(\oplus'D)$, można uzyskać inny rejestr $D^{b_1}T^{c_1} \oplus D^{b_2}(\oplus'D)^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{t-i} \oplus D^{b_4}(\oplus'D)^{c_4} \oplus D^{b_5}T^{c_5} \oplus D^{b_6}T^{c_6}$.

Jeżeli założymy, że $c = \sum_{j=1}^6 c_j + d$, wówczas brakująca liczba d komórek $(\oplus'D)$ lub T powinna się znaleźć w ciągu $t-i$ komórek dodawanych do rejestru. Wówczas też skorelowany z wielomianem 3.12 rejestr BTDT-LFSR przyjmuje postać następującą $D^{b_1}T^{c_1} \oplus D^{b_2}T^{c_2} \oplus D^{b_3}T^{c_3} \oplus D^{t-i-d} \oplus D^{b_4}T^{c_4} \oplus D^{b_5}T^{c_5} \oplus D^{b_6}T^{c_6}$.

Ten przypadek rejestru o strukturze BTDT ilustruje przykład 3.8.

Przykład 3.8

Dany jest źródłowy wielomian pierwotny

$$p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}.$$

Zaprojektować rejestr BTDT-LFSR- $p(x)$

$$p_{\text{BTDT}}(x) = x^7 (1 + x + x^2 + x^3) + (1 + x + x^2)(1 + x + x^4)$$

$$p_{\text{BTDT}_w}(x) = x^7 (1 + x)^3 + [1 + x(1 + x)][1 + x(1 + x^3)] \cong \\ \cong DT_{\oplus} DDTT^{\oplus} DDD^{\oplus} D \cong D(\oplus'D)_{\oplus} DD(\oplus'D)(\oplus'D)^{\oplus} DDD^{\oplus} D \cong \\ \cong (\oplus'D)D_{\oplus} D(\oplus'D)(\oplus'D)D^{\oplus} DDD^{\oplus} D, \text{ gdzie } t = 6, i = 2, d = 2$$

Zakładając $k_j = 0$, otrzymujemy z 3.12 następujące wyrażenie

$$p_{\text{BTDT}_w}(x) = x^n + \left[p_0 + x^r \left[p_r + x^{s-r} \left[p_s + x^{i-s} \right] \right] \right] \\ \left[p_n + x^{n-w} \left[p_w + x^{w-u} \left[p_u + x^{u-t} p_t \right] \right] \right] \quad (3.13)$$

Uproszczony zapis schematu rejestru TBD-LFSR związanego z tym wielomianem przedstawia następujący ciąg znaków $D^r_{\oplus} D^{s-r}_{\oplus} D^{i-s}_{\oplus} D^{t-i}_{\oplus} D^{u-t}_{\oplus} D^{w-u}_{\oplus} D^{n-w}$.

Przykład 3.9

Weźmy źródłowy wielomian pierwotny

$$p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}.$$

Zaprojektować dwa różne rejestry BTDT-LFSR- $p(x)$.

$$p(x) = x^{10} + (1 + x^4 + x^5 + x^6) + x^3 (1 + x^4 + x^5 + x^6)$$

$$a) p_{\text{BTDT}}(x) = x^{10} + (1 + x^3)(1 + x^4 + x^5 + x^6)$$

$$p_{\text{BTDT}_w}(x) = x^{10} + [1 + x^3][1 + x^4 [1 + x[1 + x]]] \cong DDD_{\oplus} D^{\oplus} D^{\oplus} D^{\oplus} DDD$$

$$b) p_{\text{BTDT}}(x) = x^{10} + (1 + x^4 + x^5 + x^6)(1 + x^3)$$

$$p_{\text{BTDT}_w}(x) = x^{10} + [1 + x^4 [1 + x[1 + x]]][1 + x^3] \cong DDDD_{\oplus} D_{\oplus} D_{\oplus} D^{\oplus} DDD$$

3.1.6. Określanie wielomianu źródłowego $p(x)$ na podstawie schematu rejestru liniowego

Przedstawiona metoda projektowania na podstawie zadanego źródłowego wielomianu $p(x)$ rejestrów liniowych o strukturach IEDT, IED, IET, EEDT, EED, EET, BTDT, TBD, BTDT, TBD, DT umożliwia także proste odwrócenie tego procesu, tzn. określenie wielomianu charakterystycznego źródłowego na podstawie schematu rejestru liniowego. Jest to

możliwe dzięki jednoznacznej zgodności zapisu schematu rejestru liniowego z postacią związanego z nim strukturalnego wielomianu $p_{cw}(x)$ (tab. 3.1).

Tabela 3.1

cw	Strukturalna postać wielomianu charakterystycznego $p_{cw}(x)$
$P_{IEDT_w}(x)$	$1 + x^{b^1}(1+x)^{c^1} [1 + x^{b^2}(1+x)^{c^2} [1 + \dots [1 + x^{b^i}(1+x)^{c^i}] \dots]]$
$P_{IED_w}(x)$	$1 + x^r [1 + x^{s-r} [1 + \dots [1 + x^{n-t}] \dots]]$
$P_{IET_w}(x)$	$1 + (1+x)^r [1 + (1+x)^{s-r} [1 + \dots [1 + (1+x)^{n-t}] \dots]]$
$P_{EEDT_w}(x)$	$1 + x^{b^4}(1+x)^{c^4} [1 + x^{b^3}(1+x)^{c^3} [1 + x^{b^2}(1+x)^{c^2} [1 + x^{b^1}(1+x)^{c^1}]]]$
$P_{EED_w}(x)$	$1 + x^{n-t} [1 + x^{t-s} [1 + x^{s-r} [1 + x^r]]]$
$P_{EET_w}(x)$	$1 + (1+x)^{n-t} [1 + (1+x)^{t-s} [1 + (1+x)^{s-r} [1 + (1+x)^r]]]$
$P_{DT_w}(x)$	$1 + x^b(1+x)^c$
$P_{T_w}(x)$	$1 + (1+x)^n$
$P_{D_w}(x)$	$1 + x^n$
$P_{TBDT_w}(x)$	$1 + [x^{b^3}(1+x)^{c^3} [1 + x^{b^2}(1+x)^{c^2} [1 + x^{b^1}(1+x)^{c^1}]]]$ $[1 + x^{b^4}(1+x)^{c^4} [1 + x^{b^5}(1+x)^{c^5} [1 + x^{b^6}(1+x)^{c^6}]]]$
$P_{TBD_w}(x)$	$1 + x^{i-s} [1 + x^{s-r} [1 + x^r]] [1 + x^{t-i} [1 + x^{u-t} [1 + x^{n-u}]]]$
$P_{BTDT_w}(x)$	$x^{n-c}(1+x)^c + [1 + x^{b^1}(1+x)^{c^1} [1 + x^{b^2}(1+x)^{c^2} [1 + x^{b^3}(1+x)^{c^3}]]]$ $[1 + x^{b^6}(1+x)^{c^6} [1 + x^{b^5}(1+x)^{c^5} [1 + x^{b^4}(1+x)^{c^4}]]]$
$P_{BTD_w}(x)$	$x^n + [1 + x^r [1 + x^{s-r} [1 + x^{i-s}]]] [1 + x^{n-w} [1 + x^{w-u} [1 + x^{u-t}]]]$

Algorytm postępowania jest następujący:

Na podstawie schematu rejestru liniowego odtwarza się wielomian strukturalny $p_{cw}(x)$, który po zlikwidowaniu wszelkich nawiasów i uporządkowaniu zostaje przekształcony w źródłowy wielomian $p(x)$.

Przedstawioną technikę odtwarzania wielomianu charakterystycznego $p(x)$ na podstawie schematów rejestrów IEDT zastosowano do optymalizacji syntezy samotestowalnych układów sekwencyjnych w [HławB94a, HławB94b].

Przykład 3.10

Mamy następujące trzy rejestry liniowe

- $DD \oplus DD (\oplus 'D) D \oplus DDD \oplus D \oplus DDDD \oplus DD$;
- $D (\oplus 'D) \oplus DDDD \oplus DDTD \oplus DDD \oplus DD \oplus D$;
- $DDDD \oplus D \oplus D \oplus DTTT \oplus D \oplus DDDTT$;

Należy sprawdzić, czy wielomiany charakterystyczne tych rejestrów są identyczne.

ad a) rejestr BTDT; $n = 16$, jedna komórka typu $(\oplus 'D)$,

ad b) rejestr TBDT; $n = 16$, dwie komórki typu $(\oplus 'D)$ i T,

ad c) rejestr IEDT; $n = 16$, pięć komórek typu T,

$$P_{BTDT_w}(x) = x^{15}(1+x) + [1 + x^2 [1 + x^3(1+x)[1 + x^3[1+x]]][1 + x^2]$$

$$P_{TBDT_w}(x) = 1 + x^4 [1 + x(1+x)][1 + x^3(1+x)[1 + x^3[1 + x^2[1+x]]]$$

$$P_{IEDT_w}(x) = 1 + x^4 [1 + x[1 + x[1 + x(1+x)^3 [1 + x[1 + x^3(1+x)^2]]]]]$$

Po wymnożeniu, uproszczeniu i uporządkowaniu w każdym z trzech powyższych przypadków uzyskujemy wielomian charakterystyczny

$p(x) = 1 + x^4 + x^5 + x^6 + x^7 + x^{12} + x^{15} + x^{16}$, który znany jest jako wielomian BCH.

3.1.7. Niektóre właściwości rejestrów liniowych

Na podstawie zadanego źródłowego wielomianu charakterystycznego $p(x)$ można zaprojektować tylko jeden rejestr o sprzężeniu IED oraz jeden rejestr o strukturze EED. Ten wielomian umożliwia także zaprojektowanie dwóch lub nieco więcej rejestrów o strukturze TBD [WangM88, Hławi89a, Hławi92b] oraz strukturze BTD [WangM88, Hławi89b, Hławi92b]. Na podstawie tego wielomianu można zaprojektować również dwa, a czasami więcej rejestrów o strukturze CADT [SerrS90, HławK93a]. Razem liczba możliwych do zaprojektowania rejestrów o strukturach IED, EED, TBD, BTD oraz CADT związanych tym samym wielomianem charakterystycznym $p(x)$ na ogół jest mniejsza od 10. Jest to zbyt mała liczba, aby można było w każdym przypadku znaleźć rejestr dobrze przystający do potrzeb wynikających z zastosowanej technologii jego implementacji [Hławi92a], czy też wynikających z posiadanej biblioteki komórek standardowych [Hławi93a] lub też wpływających z potrzeb procesu optymalizacji syntezy samotestowalnego układu sekwencyjnego [HławB94a, HławB94b], czy wreszcie ze względu na konieczność przeprojektowania rejestru w trakcie prototypowania samotestowalnego układu ASIC [Hławi90d]. Czasami ze

względem na pewne ograniczenia narzucane przez projekt UC również istotne jest miejsce lokowania w rejestrze bramek XOR sprzężenia liniowego. Większość tych potrzeb mogłaby być zaspokojona, gdyby liczba różnych realizacji rejestrów liniowych związanych tym samym wielomianem charakterystycznym $p(x)$ znacząco wzrosła. Okazuje się, że jest to możliwe w przypadku projektowania rejestrów IEDT, EEDT, TBDT oraz BTDT.

W każdym rejestrze IEDT, EEDT, TBDT lub BTDT znajduje się co najmniej jeden ciąg komórek $\dots D^k(\oplus'D)^f \dots D^k T^g \dots$ lub $\dots D^k(\oplus'D)^{f-g} T^g \dots$, który skorelowany jest z następującym fragmentem wielomianu strukturalnego $\dots x^k(1+x)^f \dots$. Ciągi takie można ustawiać w dowolnym porządku tworząc różne permutacje. Wynika to z możliwości dowolnego zapisu powyższego wielomianu. Na przykład wielomian $\dots x^{k-2}(1+x)^f x^2 \dots$ skorelowany jest z ciągiem komórek $\dots D^{k-2}(\oplus'D)^f D^2 \dots$ lub $\dots D^{k-2} T^f D^2 \dots$.

Przykład 3.11

Załóżmy wielomian strukturalny $p_{DTw}(x) = 1 + x^2(1+x)^3$ skorelowany z rejestrzem $DD(\oplus'D)(\oplus'D)(\oplus'D)$. Przesuwając komórki D oraz $(\oplus'D)$, można uzyskać jeszcze następujące permutacje związane wielomianem $p(x)$: $D(\oplus'D)D(\oplus'D)(\oplus'D)$, $D(\oplus'D)(\oplus'D)D(\oplus'D)$, $D(\oplus'D)(\oplus'D)(\oplus'D)D$, $(\oplus'D)DD(\oplus'D)(\oplus'D)$, $(\oplus'D)D(\oplus'D)D(\oplus'D)$, $(\oplus'D)D(\oplus'D)(\oplus'D)D$, $(\oplus'D)(\oplus'D)DD(\oplus'D)$, $(\oplus'D)(\oplus'D)D(\oplus'D)D$, $(\oplus'D)(\oplus'D)(\oplus'D)DD$.

Wyrażenie określające liczbę takich permutacji ilustruje dwumian Newtona

$$\binom{r}{k} = \frac{r!}{k!(r-k)!}$$

w którym k określa liczbę komórek $(\oplus'D)$ lub T, natomiast r określa liczbę wszystkich komórek D, T oraz $(\oplus'D)$.

Przykład 3.12

Liczbę permutacji powstałych przez przestawianie przerzutników rejestru $D^{14}T^3$ określa dwumian Newtona $\binom{14}{5} = 680$, natomiast liczbę permutacji dla rejestru $DT^{\oplus}DDDD_{\oplus}DDDTTTT_{\oplus}DTT$ określa wyrażenie $\binom{2}{1}\binom{7}{4}\binom{3}{2} = 2 \times 35 \times 3 = 210$.

Oba przykłady ilustrują znaczący wzrost liczby różnych rejestrów związanych tym samym wielomianem charakterystycznym. Liczby te można dalej znacznie powiększać

przez tworzenie dalszych zbiorów rejestrów liniowych o innych strukturach związanych tym samym wielomianem charakterystycznym. Liczności niektórych zbiorów rejestrów liniowych związanych tym samym wielomianem charakterystycznym są ogromne wręcz zaskakujące. Na przykład na bazie rejestru $DD^{41}T^5$ można stworzyć 153393 różnych permutacji, natomiast w oparciu o rejestr $D^{32}T^{33}$ można ich otrzymać aż 36097×10^{14} . Pozwala to zdać sobie sprawę z ogromnych możliwości projektowych, jakie tkwią w rejestrach liniowych o nowych strukturach sprzężeń IEDT, EEDT, TBDT oraz BTDT.

Struktury sprzężeń liniowych różnią się między sobą przede wszystkim liczbą bramek XOR (LX), czasem propagacji sygnału pomiędzy sąsiednimi przerzutnikami (CZ), liczbą połączeń w sprzężeniu zwrotnym (LP), liczbą krótkich połączeń w tym sprzężeniu (KP), obciążalnością wyjść przerzutników, czyli współczynnikiem ich rozptywu (WR) oraz rozkładem komórek $(\oplus'D)$ lub T wzdłuż rejestru liniowego. Niektóre różnice wymienionych parametrów wynikają z zastosowania w projekcie rejestru liniowego przerzutników T zamiast D. Przed przystąpieniem do projektowania rejestru liniowego istotne jest więc nie tylko określenie wielomianu charakterystycznego, ale także ważne jest określenie ww. parametrów. Wyjaśnia to przykład podany w tab. 3.2, w której zebrano kilkanaście najbardziej charakterystycznych spośród wielu różnych realizacji rejestrów liniowych związanych tym samym pierwotnym wielomianem $p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$. Czas propagacji sygnału CZ określany jest w tej tablicy za pomocą liczby bramek XOR występujących w kaskadzie tych bramek w ścieżce pomiędzy wyjściem i-tego przerzutnika a wejściem j-tego przerzutnika. Do liczby połączeń LP wliczana jest pętla łącząca wyjście ostatniej komórki z wejściem komórki pierwszej, wliczane są wszystkie linie sprzężenia zwrotnego doprowadzone do zewnętrznych bramek XOR, i wliczana jest linia łącząca wyjście kaskady zewnętrznych bramek XOR z wejściem pierwszej komórki rejestru. Liczba ta nie uwzględnia linii, które są niezbędne do realizacji komórek $(\oplus'D)$. Liczbę KP określają te pętle sprzężenia, które łączą komórki sąsiednie. Współczynnik rozptywu WR określa liczbę wejść różnych komórek połączonych z wyjściem jednej komórki. W przypadku parametrów CZ oraz WR podano ich największe wartości występujące w rozpatrywanym rejestrze. Bramka XOR należąca do komórki $(\oplus'D)$ nie jest doliczana do liczby określającej parametr LX oraz CZ, jeżeli zamiast komórki $(\oplus'D)$ zostanie zastosowana komórka T. Ten ostatni przypadek może wystąpić w niektórych układach PLD (np. XL78C800 [EXEL]) oraz FPGA

(np. MAX 7000E [ALTERA, HławB94a, HławB94b]). W przypadku układów ACT-3 firmy ACTEL również bramki XOR zawarte w strukturach IED, IEDT nie są brane pod uwagę przy określaniu CZ [Hławi97a].

Tabela 3.2

c	Rejestr liniowy	LX	CZ	LP	KP	WR
IED	$DDD \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D$	7	1	1	0	8
IEDT	$DDD(\oplus'D)(\oplus'D) \oplus D \oplus DDD \oplus D$	5	1	1	0	4
IEDT	$DDD(\oplus'D) \oplus DD \oplus DD \oplus DD$	4	1	1	0	4
IEDT	$DDD(\oplus'D)(\oplus'D)(\oplus'D) \oplus DDDD$	4	1	1	0	2
EED	$D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D$	7	7	9	0	2
EEDT	$D \oplus DDD \oplus D \oplus (\oplus'D)(\oplus'D) DDD$	5	3	5	0	2
EEDT	$DD \oplus DD \oplus DD \oplus (\oplus'D) DDD$	4	3	5	0	2
EEDT	$DDDD \oplus (\oplus'D)(\oplus'D)(\oplus'D) DDD$	4	1	3	0	2
DT	$DDD(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D)(\oplus'D)$	7	1	1	0	2
DT	$DDDTTTTTT$	0	0	1	0	1
TBDT	$DD \oplus DDD \oplus D \oplus DD(\oplus'D) D$	4	1	3	0	3
TBD	$DD \oplus DDD \oplus D \oplus DDD \oplus D$	4	1	3	0	4
TBD	$DD \oplus DD \oplus DD \oplus DDD \oplus D$	4	3	5	0	2
BTDT	$D(\oplus'D) DD(\oplus'D)(\oplus'D) \oplus DDD \oplus D$	6	3	4	0	2
BTD	$DDD \oplus D \oplus D \oplus D \oplus DDD$	4	3	5	0	2
BTD	$DDDD \oplus D \oplus D \oplus D \oplus DDD$	4	2	3	0	4
CADT	0110100110	14	2	0	9	3
CADT	0110010110	14	2	0	9	3

Przy założeniu braku dostępu do przerzutników T oraz zakładając, że potrzebna jest minimalna liczba bramek XOR, minimalny czas propagacji, tylko jedno długie połączenie i najmniejszy współczynnik rozprywu, wówczas najlepszy do implementacji wydaje się być rejestr $DDD(\oplus'D)(\oplus'D)(\oplus'D) \oplus DDDD$. Przy założeniu że rejestr nie powinien posiadać pętli łączącej ostatni przerzutnik z przerzutnikiem pierwszym ($LP = 0$), rozwiązaniem mogą być rejestry typu CA, np. rejestr 0110100110. Wydaje się jednak, że duża liczba

bramek XOR dyskwalifikuje taki wybór. Na przykład w przypadku wielomianu pierwotnego $p(x) = 1 + x + x^{60}$ związany z nim rejestr typu CA (111001111010010111010000101111001101000010111010010111001110 [CattM96]) wymaga użycia aż 93 bramek XOR. Natomiast rejestr $D \oplus D^{59}$ związany tym samym wielomianem $p(x)$ wymaga użycia wprawdzie jednego długiego połączenia, ale za to pozwala zaoszczędzić 92 bramki XOR (technikę eliminowania długich połączeń w projekcie rejestrów IED, IEDT, EED i EEDT omówiono w rozdziale 3.2.6). Parametry LX, CZ, LP, KP oraz WR, a także obecność i rozkład komórek $(\oplus'D)$ lub T w rejestrze liniowym decydują o wyborze najlepiej przystającej struktury sprzężenia liniowego do architektury logicznej oraz sieci połączeń oferowanej przez układ FPGA czy też PLD, w którym implementowany jest rejestr liniowy. W pracy [Hławi97a] uzasadniono wielką przydatność rejestrów liniowych o strukturach IEDT w optymalnym dopasowywaniu ich do zasad projektowania obowiązujących w układach FPGA (ACT-3) firmy ACTEL.

Dobrym tego przykładem mogą być rejestry MISR. W zaproponowanej w [Hławi97a] technice projektowania n-stopniowych rejestrów IEDT-MISR (IED-MISR) bramki XOR w kombinacyjnych modułach AX1C służą wyłącznie do realizacji struktury sprzężenia liniowego tych rejestrów, natomiast bramki XOR modułów sekwencyjnych "s" (typu D-XA w [Hławi97a]) służą wyłącznie do podłączania wyjść TUC. Taka struktura, aczkolwiek zawierająca minimalną liczbę modułów CLB, nie pozwala jeszcze na wyegzekwowanie maksymalnej szybkości pracy. Szybkość tę można osiągnąć zastępując d kombinacyjnych modułów AX1C modułami sekwencyjnymi typu D-XA. Uzyskana w ten sposób $(n+d)$ -bitowa struktura rejestru MISR zawiera wyłącznie moduły sekwencyjne D-XA. Dołączone moduły sekwencyjne nie są w ogóle podłączone do wyjść TUC i służą wyłącznie do realizacji struktury sprzężenia liniowego takiego specjalnego rejestru MISR. W ten sposób rejestr MISR wydłużony o d takich dodatkowych komórek posiada $n + d$ przerzutników D i tylko po jednej bramce XOR pomiędzy każdym z nich, co pozwala na pracę rejestru z maksymalną szybkością. Wartość liczby d jest minimalizowana za pomocą podanych w następnych rozdziałach technik projektowania struktur sprzężeń liniowych o zredukowanej liczbie bramek XOR. Ta specjalna struktura rejestru IEDT-MISR (IED-MISR), zrealizowana przy użyciu tej samej liczby $n + d$ modułów CLB co struktura tradycyjna, pozwala także zmniejszyć 2^d razy statyczne prawdopodobieństwo maskowania błędów w procesie kompaktacji (rozdział 4).

3.2. Jak uzyskać minimalną liczbę bramek XOR w sprzężeniu liniowym?

Problem projektowania rejestrów liniowych o mniejszej liczbie bramek XOR w sprzężeniu, niż wymagają tego rejestry konwencjonalne, jest istotnym zagadnieniem praktycznym. W tym rozdziale przedstawiono metodę określania takich wielomianów charakterystycznych, które umożliwiają projektowanie rejestrów liniowych o strukturach sprzężeń liniowych zawierających mniej bramek XOR, niż potrzebują tego struktury IED oraz EED.

3.2.1. Właściwości wielomianów gwarantujące minimalne realizacje rejestrów o strukturach IEDT oraz EEDT

Wielomian $p(x)$ nad ciałem $GF(2)$, dzielący się tylko przez wielomian $p(x)$ lub 1, będące elementem ciała $GF(2)$, nazywa się wielomianem pierwszym [PetW94]. Każdy wielomian pierwszy $p(x)$ posiada nieparzystą liczbę $r+2$ niezerowych współczynników p_i . W związku z tym można go przedstawić w postaci

$$p(x) = 1 + x^k [1 + x^{c_1} [1 + x^{c_2} [1 + \dots + x^{c_i} [1 + \dots + x^{c_{r-1}} [1 + x^{c_r} \dots] \dots]]]]$$

kąta umożliwia zaprojektowanie rejestrów liniowych

$D^k \oplus D^{c_1} \oplus D^{c_2} \oplus \dots \oplus D^{c_i} \oplus \dots \oplus D^{c_{r-1}} \oplus D^{c_r}$ o strukturze IEDT oraz

rejestrów $D^{cr} \oplus D^{c_{r-1}} \oplus \dots \oplus D^{c_i} \oplus \dots \oplus D^{c_2} \oplus D^{c_1} \oplus D^k$ o strukturze EED. Rejestry te zawierają po r bramek XOR w swoich sprzężeniach liniowych.

Każdy wielomian charakterystyczny $p(x)$ o nieparzystej liczbie $r+2$ niezerowych współczynników można przedstawić w postaci $1 + x^k h(x) = 1 + x^k (1+x)^f g(x)$, w której wielomian $(1+x)^f$ posiada s niezerowych współczynników, natomiast wielomian $g(x)$ posiada m niezerowych współczynników. Wielomian ten można także przedstawić w formie

$$p(x) = 1 + \sum_{i=0}^{m-1} \oplus \sum_{j=0}^{s-1} \oplus x^{(k+a_i)+v_j} \quad (3.14)$$

w której $a_0 = 0, v_0 = 0$. Niech $p/2$ oznacza liczbę sum identycznych elementów $x^q + x^q$, które wzajemnie się zerują po wymnożeniu wielomianów $x^k (1+x)^f g(x)$. Wówczas $ms - p = r + 1$. Z równości tej wynika, że $r = ms - p - 1$.

Wielomian $p(x)$ można także przekształcić w postać

$$p(x) = 1 + x^k (1+x)^f g(x) = \\ = 1 + x^k (1+x)^f [1 + x^{b_1} [1 + x^{b_2} [1 + \dots + x^{b_i} [1 + \dots + x^{b_{m-2}} [1 + x^{b_{m-1}} \dots] \dots] \dots]]]$$

kąta umożliwia zaprojektowanie rejestrów

$D^k (\oplus D)^f \oplus D^{b_1} \oplus D^{b_2} \oplus \dots \oplus D^{b_i} \oplus \dots \oplus D^{b_{m-2}} \oplus D^{b_{m-1}}$ o strukturze IEDT oraz rejestrów $D^{b_{m-1}} \oplus D^{b_{m-2}} \oplus \dots \oplus D^{b_i} \oplus \dots \oplus D^{b_2} \oplus D^{b_1} (\oplus D)^f D^k$ o strukturze EEDT.

Niech d oznacza liczbę bramek XOR w sprzężeniu liniowym ww. rejestrów. Zauważmy, że $d = f + m - 1$. Na podstawie tego wyrażenia i wcześniej uzyskanego wyrażenia określającego liczbę r otrzymujemy $d \leq r$, jeżeli $f + m - 1 \leq ms - p - 1$. Po przekształceniu tej nierówności otrzymujemy ostatecznie, że

$$d \leq r, \text{ jeżeli } m \geq \frac{f+p}{s-1} \quad (3.15)$$

Obecnie określimy konkretne postaci wielomianów charakterystycznych spełniających warunek 3.15. W celu uproszczenia rozważań ograniczymy naszą uwagę tylko do tych wielomianów $p(x) = 1 + x^k (1+x)^f g(x)$, w których $f < 15$. Taką liczbę f można zapisać jako

$$f = \beta_t 2^t + \beta_u 2^u + \beta_w 2^w, \text{ gdzie } w > u > t \geq 0 \text{ oraz } \beta_i \in \{0,1\}.$$

W związku z tym każdy wielomian

$$(1+x)^f = (1+x)^{\beta_t 2^t + \beta_u 2^u + \beta_w 2^w} = (1+\beta_t x)^{2^t} (1+\beta_u x)^{2^u} (1+\beta_w x)^{2^w} = \\ = 1 + \beta_t x^{2^t} + \beta_u x^{2^u} + \beta_t \beta_u x^{2^t+2^u} + \beta_w x^{2^w} + \beta_t \beta_w x^{2^t+2^w} + \\ + \beta_u \beta_w x^{2^u+2^w} + \beta_t \beta_u \beta_w x^{2^t+2^u+2^w}$$

Liczba s niezerowych współczynników tego wielomianu zawsze równa jest 2^b , gdzie $b \leq 3$. Na tej podstawie oraz w oparciu o 3.14 każdy wielomian $p(x) = 1 + x^k (1+x)^f g(x)$, w którym $f < 15$ można przedstawić jako

$$p_m(x) = 1 + \sum_{i=0}^{m-1} \oplus \left(x^{(k+a_i)} + \beta_t x^{(k+a_i)+2^t} + \beta_u x^{(k+a_i)+2^u} + \beta_t \beta_u x^{(k+a_i)+2^t+2^u} + \right. \\ \left. + \beta_w x^{(k+a_i)+2^w} + \beta_t \beta_w x^{(k+a_i)+2^t+2^w} + \beta_u \beta_w x^{(k+a_i)+2^u+2^w} + \beta_t \beta_u \beta_w x^{(k+a_i)+2^t+2^u+2^w} \right)$$

Spełnienie warunku $m \geq (\beta_t 2^t + \beta_u 2^u + \beta_w 2^w + p)/(2^b - 1)$ umożliwia zaprojektowanie rejestrów $D^k (\oplus D)^{(\beta_t 2^t + \beta_u 2^u + \beta_w 2^w)} \oplus D^{b_1} \oplus D^{b_2} \oplus \dots \oplus D^{b_i} \oplus \dots \oplus D^{b_{m-2}} \oplus D^{b_{m-1}}$ o strukturze

IEDT oraz rejestrów $D^{bm-1} \oplus D^{bm-2} \oplus \dots \oplus D^{bi} \oplus \dots \oplus D^{b2} \oplus D^{b1} \oplus (\oplus' D)^{(\beta_t 2^t + \beta_u 2^u + \beta_w 2^w)} D^k$ o strukturze EEDT wymagającej użycia $d \leq r$ bramek XOR. W powyższym warunku $p = 0$, jeżeli elementy wielomianu $p_m(x)$ posiadają takie wykładniki, że $a_{i+1} > a_i + f$, czyli $a_{i+1} > a_i + \beta_t 2^t + \beta_u 2^u + \beta_w 2^w$.

Dla $\beta_t = 1, \beta_u = \beta_w = 0$ wielomian

$$p_m(x) = 1 + \sum_{i=0}^{m-1} \oplus (x^{(k+a_i)} + x^{(k+a_i)+2^t}) \quad (3.16)$$

Tę postać wielomian ten przyjmuje dla $2^t = 1, 2, 4, 8, \dots$. Umożliwia ona projektowanie rejestrów IEDT oraz EEDT o sprzężeniach liniowych wymagających $d \leq r$ bramek XOR, gdy $m \geq 2^t$.

Przykład 3.13

Załóżmy $m > 2^t/1$ oraz $t = 0$. Otrzymujemy $m > 1$ oraz $f = 1$. Weźmy $m = 2$, dla którego $a_1 > a_0 + 1$. Przy tych założeniach ogólna postać wielomianu gwarantującego $d < r$ jest następująca

$$p_m(x) = 1 + \sum_{i=0}^1 \oplus (x^{(k+a_i)} + x^{(k+a_i)+1})$$

Na przykład dla $a_1 = 6$ otrzymujemy wielomian pierwotny $1 + (x^8 + x^9) + (x^{14} + x^{15})$. Umożliwia on między innymi realizację rejestru $D^8(\oplus' D) \oplus D^6$ oraz rejestru $D^6 \oplus (\oplus' D) D^8$ wymagających użycia tylko dwóch bramek XOR w przeciwieństwie do rejestru $D^8 \oplus D \oplus D^5 \oplus D$ lub rejestru $D \oplus D^5 \oplus D \oplus D^8$, które wymagają użycia trzech bramek XOR.

Dla $\beta_t = \beta_u = 1, \beta_w = 0$ wielomian

$$p_m(x) = 1 + \sum_{i=0}^{m-1} \oplus (x^{(k+a_i)} + x^{(k+a_i)+2^t} + x^{(k+a_i)+2^u} + x^{(k+a_i)+2^t+2^u}) \quad (3.17)$$

Tę postać wielomian ten przyjmuje dla $2^t + 2^u = 3, 5, 6, 9, 10, 12, \dots$. Umożliwia on projektowanie rejestrów o strukturach IEDT oraz EEDT wymagających użycia $d \leq r$ bramek XOR, gdy $m \geq (2^t + 2^u)/3$.

Przykład 3.14

Załóżmy $m > (2^t + 2^u)/3$ oraz $t = 1, u = 2$. Otrzymujemy $m > 2$ oraz $f = 6$. Weźmy $m = 3$, dla którego $a_{i+1} > a_i + 6$. Przy tych założeniach ogólna postać wielomianu gwarantującego $d < r$ jest następująca

$$p_m(x) = 1 + \sum_{i=0}^2 \oplus (x^{(k+a_i)} + x^{(k+a_i)+2} + x^{(k+a_i)+4} + x^{(k+a_i)+6})$$

Na przykład dla $a_2 = 14$ oraz $a_1 = 7$ otrzymujemy wielomian pierwotny

$$1 + (x + x^3 + x^5 + x^7) + (x^8 + x^{10} + x^{12} + x^{14}) + (x^{15} + x^{17} + x^{19} + x^{21}) = 1 + x(1 + x^2 + x^4 + x^6)(1 + x^7 + x^{14}) = 1 + x(1 + x)^6[1 + x^7[1 + x^7]]$$

umożliwiający zaprojektowanie rejestru $D(\oplus' D)^6 \oplus D^7 \oplus D^7$ zawierającego $d = 8$ bramek XOR zamiast $r = 11$ w przypadku struktury IED lub EED.

Dla $\beta_t = \beta_u = \beta_w = 1$ wielomian

$$p_m(x) = 1 + \sum_{i=0}^{m-1} \oplus (x^{(k+a_i)} + x^{(k+a_i)+2^t} + x^{(k+a_i)+2^u} + x^{(k+a_i)+2^t+2^u} + x^{(k+a_i)+2^w} + x^{(k+a_i)+2^t+2^w} + x^{(k+a_i)+2^u+2^w} + x^{(k+a_i)+2^t+2^u+2^w})$$

Tę postać wielomian ten przyjmuje dla $2^t + 2^u + 2^w = 7, 11, 13, 14, \dots$. Umożliwia on projektowanie rejestrów IEDT oraz EEDT o sprzężeniach liniowych wymagających $d \leq r$ bramek XOR, gdy $m \geq (2^t + 2^u + 2^w)/7$.

Przykład 3.15

Załóżmy $m > (2^t + 2^u + 2^w)/7$ oraz $t = 0, u = 1, w = 2$. Otrzymujemy $m > 1$ oraz $f = 7$. Weźmy $m = 2$, dla którego $a_1 > a_0 + 7$. Przy tych założeniach ogólna postać wielomianu gwarantującego $d < r$ jest następująca

$$p_m(x) = 1 + \sum_{i=0}^{m-1} \oplus (x^{(k+a_i)} + x^{(k+a_i)+1} + x^{(k+a_i)+2} + x^{(k+a_i)+3} + x^{(k+a_i)+4} + x^{(k+a_i)+5} + x^{(k+a_i)+6} + x^{(k+a_i)+7})$$

Na przykład dla $a_1 = 12$ otrzymujemy wielomian pierwotny

$$1 + (x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9) + (x^{14} + x^{15} + x^{16} + x^{17} + x^{18} + x^{19} + x^{20} + x^{21}) = 1 + x^2(1 + x)^7[1 + x^{12}]$$

Rejestr $D^2(\oplus' D)^7 \oplus D^{12}$ związany tym wielomianem wymaga użycia $d = 8$ bramek XOR zamiast $r = 15$ w przypadku struktury IED lub EED.

Załóżmy obecnie wielomian $p(x)$ o parzystej liczbie $r + 3$ niezerowych współczynników p_i . Rejestr IED lub EED o takim wielomianie charakterystycznym zawiera w swoim sprzężeniu liniowym $r + 1$ bramek XOR. Każdy taki wielomian można przedstawić w następującej postaci [Hławi92a]:

$$p(x) = 1 + x^p [1 + x^k h(x)] = 1 + x^p [1 + x^k (1 + x)^f [g(x)]] \text{ gdzie}$$

$$g(x) = 1 + x^{b_1} [1 + x^{b_2} [1 + \dots + x^{b_i} [1 + \dots + x^{b_{m-2}} [1 + x^{b_{m-1}}] \dots] \dots]]$$

umożliwiającej zaprojektowanie rejestrów

$D^p \oplus D^k (\oplus D)^f \oplus D^{b_1} \oplus D^{b_2} \oplus \dots \oplus D^{b_i} \oplus \dots \oplus D^{b_{m-2}} \oplus D^{b_{m-1}}$ o strukturze IEDT oraz rejestrów $D^{b_{m-1}} \oplus D^{b_{m-2}} \oplus \dots \oplus D^{b_i} \oplus \dots \oplus D^{b_2} \oplus D^{b_1} (\oplus D)^f D^k \oplus D^p$ o strukturze EEDT. Sprzężenia liniowe tych rejestrów wymagają użycia $m + f$ bramek XOR zamiast $r + 1$.

Przykład 3.16

Dla wielomianu CRC-16:

$$p(x) = 1 + x^2 + x^{15} + x^{16} = 1 + x^2 [1 + x^{13} (1 + x)] \cong [D^2 \oplus D^{13} (\oplus D)]$$

W tym przykładzie $m + f = r + 1 = 2$.

Dla wielomianu CRC-CCITT:

$$p(x) = 1 + x^5 + x^{12} + x^{16} = 1 + x^5 [1 + x^7 (1 + x)^4] \cong [D^5 \oplus D^7 (\oplus D)^4]$$

3.2.2. Tablice wielomianów pierwotnych gwarantujących najtańsze realizacje rejestrów o strukturach IEDT oraz EEDT

Rzędem wielomianu pierwszego $p(x)$ nazywamy najmniejszą liczbę całkowitą k taką, że dwumian $1 + x^k$ jest podzielny przez $p(x)$. Wielomianem pierwotnym nazywamy taki wielomian pierwszy $p(x)$ stopnia n , którego rząd $k = 2^n - 1$ [PetW94]. Wielomiany pierwotne o liczbie niezerowych współczynników $r + 2 \leq 5$ umożliwiają realizację klasycznych rejestrów liniowych IED oraz EED z jedną ($r = 1$) lub co najwyżej trzema ($r = 3$) bramkami XOR w sprzężeniu liniowym. Istnieją więc dwa następujące rodzaje takich wielomianów pierwotnych $p(x) = 1 + x^a + x^n$ oraz $p(x) = 1 + x^a + x^b + x^c + x^n$. Ich konkretne postaci można znaleźć w katalogach, np. w [PetW94]. Obecnie zajmiemy się odpowiedzią na pytanie, które z tych wielomianów umożliwiają projektowanie rejestrów o strukturach IEDT oraz EEDT zawierających $d \leq r$ bramek XOR.

W oparciu o 3.16 stwierdzamy, że każdy wielomian $p_m(x) = 1 + x^k + x^{k+1}$ umożliwia realizację sprzężenia liniowego rejestrów o strukturach IEDT, EEDT, IED, EED przy

użyciu tylko jednej bramki XOR. W tabeli 3.3 przedstawiono wszystkie wielomiany pierwotne $p(x) = 1 + x^k + x^{k+1}$ stopnia $k + 1 \leq 20$, które umożliwiają realizację rejestrów o strukturach IEDT (EEDT) przy użyciu jednej bramki XOR. W tabeli tej dla porównania umieszczono także rejestry o strukturach IED oraz EED związane z przedstawionymi wielomianami charakterystycznymi.

Tabela 3.3

n	$p(x)$	IEDT (EEDT)	IED	EED
3	$1 + x^2 + x^3$	$D^2 (\oplus D)$	$D^2 \oplus D$	$D \oplus D^2$
4	$1 + x^3 + x^4$	$D^3 (\oplus D)$	$D^3 \oplus D$	$D \oplus D^3$
5	nie istnieje dla IEDT (EEDT)			
6	$1 + x^5 + x^6$	$D^5 (\oplus D)$	$D^5 \oplus D$	$D \oplus D^5$
7	$1 + x^6 + x^7$	$D^6 (\oplus D)$	$D^6 \oplus D$	$D \oplus D^6$
dla $n = 8, 9, 10, 11, 12, 13, 14$ nie istnieją struktury IEDT (EEDT)				
15	$1 + x^{14} + x^{15}$	$D^{14} (\oplus D)$	$D^{14} \oplus D$	$D \oplus D^{14}$
dla $n = 16, 17, 18, 19, 20, 21$ nie istnieją struktury IEDT (EEDT)				
22	$1 + x^{21} + x^{22}$	$D^{21} (\oplus D)$	$D^{21} \oplus D$	$D \oplus D^{21}$

W oparciu o 3.17 stwierdzamy, że każdy wielomian

$p_m(x) = 1 + x^k + x^{k+2} + x^{k+a_1} + x^{k+a_1+2}$ umożliwia realizację rejestrów IEDT, EEDT oraz rejestrów IED, EED przy użyciu trzech bramek XOR. W tabeli 3.4. przedstawiono wszystkie wielomiany pierwotne $p(x) = 1 + (x^k + x^{k+2}) + (x^{k+a_1} + x^{k+a_1+2})$ stopnia $k + a_1 + 2 \leq 20$, które umożliwiają realizację rejestrów o strukturach IEDT oraz EEDT przy użyciu trzech bramek XOR, w tym dwóch, które znajdują się w komórkach $(\oplus D)$.

Tabela 3.4

n	p(x)	IEDT	EEDT
6	$1 + x + x^3 + x^4 + x^6$	$D(\oplus'D)^2 \oplus D^3$	$D^3 \oplus (\oplus'D)^2 D$
7	$1 + x + x^3 + x^5 + x^7$	$D(\oplus'D)^2 \oplus D^4$	$D^4 \oplus (\oplus'D)^2 D$
8	$1 + x^3 + x^5 + x^6 + x^8$	$D^3(\oplus'D)^2 \oplus D^3$	$D^3 \oplus (\oplus'D)^2 D^3$
9	$1 + x^4 + x^6 + x^7 + x^9$ $1 + x^2 + x^4 + x^7 + x^9$	$D^4(\oplus'D)^2 \oplus D^3$ $D^2(\oplus'D)^2 \oplus D^5$	$D^3 \oplus (\oplus'D)^2 D^4$ $D^5 \oplus (\oplus'D)^2 D^2$
10	$1 + x^5 + x^7 + x^8 + x^{10}$	$D^5(\oplus'D)^2 \oplus D^3$	$D^3 \oplus (\oplus'D)^2 D^5$
11	$1 + x^6 + x^8 + x^9 + x^{11}$ $1 + x^3 + x^5 + x^9 + x^{11}$ $1 + x^2 + x^4 + x^9 + x^{11}$	$D^6(\oplus'D)^2 \oplus D^3$ $D^3(\oplus'D)^2 \oplus D^6$ $D^2(\oplus'D)^2 \oplus D^7$	$D^3 \oplus (\oplus'D)^2 D^6$ $D^6 \oplus (\oplus'D)^2 D^3$ $D^7 \oplus (\oplus'D)^2 D^2$
12	nie istnieją dla IEDT oraz EEDT		
13	$1 + x^6 + x^8 + x^{11} + x^{13}$	$D^6(\oplus'D)^2 \oplus D^5$	$D^5 \oplus (\oplus'D)^2 D^6$
14	$1 + x^5 + x^7 + x^{12} + x^{14}$	$D^5(\oplus'D)^2 \oplus D^7$	$D^7 \oplus (\oplus'D)^2 D^5$
15	$1 + x^{10} + x^{12} + x^{13} + x^{15}$ $1 + x^8 + x^{10} + x^{13} + x^{15}$ $1 + x + x^3 + x^{13} + x^{15}$	$D^{10}(\oplus'D)^2 \oplus D^3$ $D^8(\oplus'D)^2 \oplus D^5$ $D(\oplus'D)^2 \oplus D^{12}$	$D^3 \oplus (\oplus'D)^2 D^{10}$ $D^5 \oplus (\oplus'D)^2 D^8$ $D^{12} \oplus (\oplus'D)^2 D$
16	$1 + x^{11} + x^{13} + x^{14} + x^{16}$ $1 + x^7 + x^9 + x^{14} + x^{16}$	$D^{11}(\oplus'D)^2 \oplus D^3$ $D^7(\oplus'D)^2 \oplus D^7$	$D^3 \oplus (\oplus'D)^2 D^{11}$ $D^7 \oplus (\oplus'D)^2 D^7$
17	$1 + x^{12} + x^{14} + x^{15} + x^{17}$ $1 + x^{11} + x^{13} + x^{15} + x^{17}$	$D^{12}(\oplus'D)^2 \oplus D^3$ $D^{11}(\oplus'D)^2 \oplus D^4$	$D^3 \oplus (\oplus'D)^2 D^{12}$ $D^4 \oplus (\oplus'D)^2 D^{11}$
18	nie istnieją dla IEDT oraz EEDT		
19	$1 + x^{10} + x^{12} + x^{17} + x^{19}$	$D^{10}(\oplus'D)^2 \oplus D^7$	$D^7 \oplus (\oplus'D)^2 D^{10}$
20	nie istnieją dla IEDT oraz EEDT		

Podobnie w oparciu o wyrażenie 3.17 stwierdzamy, że każdy wielomian $p(x) = 1 + x^k + x^{k+1} + x^{k+a_1} + x^{k+a_1+1}$ umożliwia realizację rejestrów o strukturach IEDT oraz EEDT przy użyciu dwóch bramek XOR zamiast trzech niezbędnych przy realizacji rejestrów o strukturach IED oraz EED. W tabeli 3.5 [Hławi97a] przedstawiono wszystkie takie wielomiany pierwotne $p_m(x)$ stopnia $k + a_1 + 1 \leq 20$, które umożliwiają realizację ww. rejestrów liniowych.

Tabela 3.5

n	p(x) *	IEDT	EEDT
5	$1 + x + x^2 + x^4 + x^5$ $1 + x^2 + x^3 + x^4 + x^5$	$D(\oplus'D) \oplus D^3$ $D^2(\oplus'D) \oplus D^2$	$D^3 \oplus (\oplus'D) D$ $D^2 \oplus (\oplus'D) D^2$
6	$1 + x + x^2 + x^5 + x^6$ $1 + x^2 + x^3 + x^5 + x^6$	$D(\oplus'D) \oplus D^4$ $D^2(\oplus'D) \oplus D^3$	$D^4 \oplus (\oplus'D) D$ $D^3 \oplus (\oplus'D) D^2$
7	$1 + x^4 + x^5 + x^6 + x^7$	$D^4(\oplus'D) \oplus D^2$	$D^2 \oplus (\oplus'D) D^4$
8	$1 + x + x^2 + x^7 + x^8$ $1 + x^2 + x^3 + x^7 + x^8$	$D(\oplus'D) \oplus D^6$ $D^2(\oplus'D) \oplus D^5$	$D^6 \oplus (\oplus'D) D$ $D^5 \oplus (\oplus'D) D^2$
9	$1 + x^4 + x^5 + x^8 + x^9$ $1 + x^5 + x^6 + x^8 + x^9$	$D^4(\oplus'D) \oplus D^4$ $D^5(\oplus'D) \oplus D^3$	$D^4 \oplus (\oplus'D) D^4$ $D^3 \oplus (\oplus'D) D^5$
10	$1 + x^6 + x^7 + x^9 + x^{10}$	$D^6(\oplus'D) \oplus D^3$	$D^3 \oplus (\oplus'D) D^6$
11	$1 + x^2 + x^3 + x^{10} + x^{11}$ $1 + x^3 + x^4 + x^{10} + x^{11}$ $1 + x^5 + x^6 + x^{10} + x^{11}$	$D^2(\oplus'D) \oplus D^8$ $D^3(\oplus'D) \oplus D^7$ $D^5(\oplus'D) \oplus D^5$	$D^8 \oplus (\oplus'D) D^2$ $D^7 \oplus (\oplus'D) D^3$ $D^5 \oplus (\oplus'D) D^5$
12	nie istnieją dla IEDT oraz EEDT		
13	$1 + x + x^2 + x^{12} + x^{13}$ $1 + x^3 + x^4 + x^{12} + x^{13}$ $1 + x^6 + x^7 + x^{12} + x^{13}$ $1 + x^9 + x^{10} + x^{12} + x^{13}$	$D(\oplus'D) \oplus D^{11}$ $D^3(\oplus'D) \oplus D^9$ $D^6(\oplus'D) \oplus D^6$ $D^9(\oplus'D) \oplus D^3$	$D^{11} \oplus (\oplus'D) D$ $D^9 \oplus (\oplus'D) D^3$ $D^6 \oplus (\oplus'D) D^6$ $D^3 \oplus (\oplus'D) D^9$
14	$1 + x^2 + x^3 + x^{13} + x^{14}$	$D^2(\oplus'D) \oplus D^{11}$	$D^{11} \oplus (\oplus'D) D^2$
15	$1 + x + x^2 + x^{14} + x^{15}$ $1 + x^8 + x^9 + x^{14} + x^{15}$	$D(\oplus'D) \oplus D^{13}$ $D^8(\oplus'D) \oplus D^6$	$D^{13} \oplus (\oplus'D) D$ $D^6 \oplus (\oplus'D) D^8$
16	nie istnieją dla IEDT oraz EEDT		
17	$1 + x^{14} + x^{15} + x^{16} + x^{17}$ $1 + x^9 + x^{10} + x^{16} + x^{17}$ $1 + x^4 + x^5 + x^{16} + x^{17}$ $1 + x^2 + x^3 + x^{16} + x^{17}$	$D^{14}(\oplus'D) \oplus D^2$ $D^9(\oplus'D) \oplus D^7$ $D^4(\oplus'D) \oplus D^{12}$ $D^2(\oplus'D) \oplus D^{14}$	$D^2 \oplus (\oplus'D) D^{14}$ $D^7 \oplus (\oplus'D) D^9$ $D^{12} \oplus (\oplus'D) D^4$ $D^{14} \oplus (\oplus'D) D^2$
18	nie istnieją dla IEDT oraz EEDT		
19	$1 + x^{13} + x^{14} + x^{18} + x^{19}$ $1 + x^8 + x^9 + x^{18} + x^{19}$	$D^{13}(\oplus'D) \oplus D^5$ $D^8(\oplus'D) \oplus D^{10}$	$D^5 \oplus (\oplus'D) D^{13}$ $D^{10} \oplus (\oplus'D) D^8$
20	$1 + x^6 + x^7 + x^{19} + x^{20}$ $1 + x^3 + x^4 + x^{19} + x^{20}$	$D^6(\oplus'D) \oplus D^{13}$ $D^3(\oplus'D) \oplus D^{16}$	$D^{13} \oplus (\oplus'D) D^6$ $D^{16} \oplus (\oplus'D) D^3$

3.2.3. Minimalne realizacje rejestrów o strukturach BTd, TBD

Wielomian $p_{\text{BTd}}(x) = 1 + p'_T(x) p'_B(x)$ zawiera iloczyn dwóch wielomianów $p'_T(x)$ oraz $p'_B(x)$. Podobnie wielomian $p_{\text{TBD}}(x) = x^n + p_B(x) p_T(x)$ zawiera iloczyn dwóch wielomianów $p_B(x)$ oraz $p_T(x)$. Załóżmy, że wielomiany $p'_T(x)$ oraz $p_B(x)$ zawierają s niezerowych współczynników, natomiast wielomiany $p'_B(x)$ oraz $p_T(x)$ zawierają m niezerowych współczynników. Przy tych założeniach każdy z rejestrów o strukturach BTd lub TBD wymaga $d = m + s - 2$ bramek XOR do realizacji sprzężenia liniowego. Poniższe dwa przykłady wyraźnie ilustrują, że nie zawsze liczba d jest mniejsza od liczby r bramek XOR niezbędnych do realizacji sprzężenia w strukturach IED lub EED.

Przykład 3.17

Projektując rejestr TBD-LFSR związany wielomianem pierwotnym

$$p(x) = 1 + x^{12} + x^{13} + x^{16} + x^{17}, \text{ otrzymujemy}$$

$$p_{\text{TBDT}}(x) = 1 + x^{12}(1 + x^2)(1 + x + x^2 + x^3), \text{ a następnie}$$

$$p_{\text{TBDTw}}(x) = 1 + x^{12}[1 + x^2][1 + x[1 + x[1 + x]]] \cong [DD^{\oplus}D^{12} \oplus D \oplus D \oplus D].$$

W efekcie $d = 4 > r = 3$.

Przykład 3.18

Projektując rejestr TBD-LFSR związany wielomianem pierwotnym

$$p(x) = 1 + x^{11} + x^{15}, \text{ otrzymujemy}$$

$$p_{\text{TBDT}}(x) = 1 + x^{11}(1 + x)(1 + x + x^2 + x^3), \text{ a następnie}$$

$$p_{\text{TBDTw}}(x) = 1 + x^{11}[1 + x][1 + x[1 + x[1 + x]]] \cong [D^{\oplus}D^{11} \oplus D \oplus D \oplus D].$$

W efekcie $d = 4 > r = 1$.

W obu przykładach liczba $p/2$ sum identycznych elementów $x^a + x^a$, które wzajemnie się zerują po wymnożeniu wielomianów $p_B(x)$ i $p_T(x)$, jest różna od zera. W efekcie $r = ms - p - 1$. Na podstawie tego wyrażenia i wcześniej uzyskanego wyrażenia określającego liczbę d otrzymujemy $d \leq r$, jeżeli $m + s - 2 \leq ms - p - 1$. Po przekształceniu tej nierówności otrzymujemy ostatecznie $d \leq r$, jeżeli $m \geq (s + p - 1)/(s - 1)$. W obu przykładach 3.17 i 3.18 ten warunek nie jest spełniony. W ogóle w przypadku wszystkich wielomianów typu $p(x) = 1 + x^a + x^b$ ten warunek jest niemożliwy do spełnienia. Jest on natomiast spełniony w przykładzie 3.7. Warunek ten spełniony jest także dla tych wielomianów $p(x) = 1 + x^a + x^b + x^c + x^n$, w których $a + n = b + c$ [WangM88,

Hławi89a]. Wielomiany takie można zawsze przekształcić w postać $p_{\text{TBDw}}(x) = 1 + [x^a [1 + x^{b-a}]] [1 + x^{c-a}]$ [WangM88, Hławi89a], dla której $d = 2$. Można to łatwo sprawdzić w przypadku wielomianu $p(x) = 1 + x^{12} + x^{13} + x^{16} + x^{17}$ z przykładu 3.17. Wspomniany wcześniej warunek spełniony jest także dla tych wielomianów $p(x) = 1 + x^a + x^b + x^c + x^n$, w których $c = a + b$ [WangM88, Hławi89b, Hławi92b]. Można je zawsze przekształcić w postać $p_{\text{TBDw}}(x) = x^n + [1 + x^a][1 + x^b]$ [WangM88, Hławi89a], dla której $d = 2$. Katalogi wielomianów pierwotnych $p(x) = 1 + x^a + x^b + x^c + x^n$, w których $a + n = b + c$ i/lub $c = a + b$, znajdują się w [WangM88].

3.2.4. Jak odzyskać do praktycznych realizacji wielomiany o dużej liczbie niezerowych współczynników?

Wielomiany $p(x)$ o dużej liczbie niezerowych współczynników p_r ze względu na duży koszt realizacji sprzężenia liniowego w strukturach IED oraz EED nie znalazły właściwie praktycznych zastosowań. Wprowadzenie nowych struktur sprzężeń liniowych, takich jak IEDT, EEDT, BTd, TBD, BTdT oraz TBdT umożliwia zmianę tej sytuacji. W przypadku niektórych wielomianów możliwe jest nawet kilkakrotne redukowanie liczby bramek XOR sprzężenia liniowego.

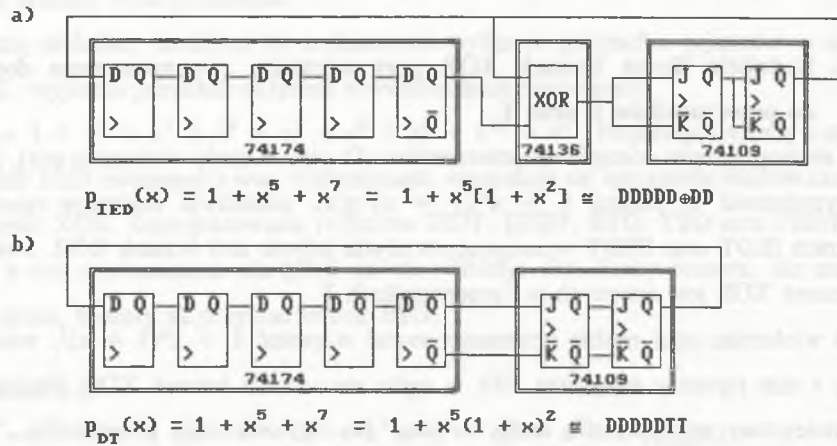
Weźmy np. pierwotny wielomian

$$p(x) = 1 + x^2 + x^3 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10} + x^{11} + x^{12} + x^{13} + x^{14} + x^{15} + x^{16} + x^{18} + x^{19} + x^{20} + x^{21}$$

o 19 niezerowych współczynnikach. Na jego podstawie można zaprojektować rejestr $D^2 \oplus D \oplus D^2 \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D \oplus D^2 \oplus D \oplus D \oplus D$ o strukturze IED, która wymaga zastosowania 17 bramek XOR. Po przekształceniu wielomianu $p(x)$ otrzymujemy jego inną postać, np. $p_{\text{IEDT}}(x) = 1 + x^2(1 + x)g(x)$, gdzie $g(x) = [1 + x^3 + x^5 + x^7 + x^9 + x^{11} + x^{13} + x^{16} + x^{18}]$. Pozwala ona zbudować rejestr $D^2(\oplus'D) \oplus D^3 \oplus D^2 \oplus D^2 \oplus D^2 \oplus D^2 \oplus D^2 \oplus D^3 \oplus D^2$ wymagający zastosowania $d = m + f - 1 = 9$ bramek XOR. Rejestr ten pozwala więc zaoszczędzić 8 bramek XOR. Identyfikacyjny rezultat umożliwia postać

$$p_{\text{BTd}}(x) = 1 + x^2(1 + x)(1 + x^3 + x^5 + x^7 + x^9 + x^{11} + x^{13} + x^{16} + x^{18}).$$

przez firmę INTEL. Ich konfigurowalne bloki logiczne pozwalają na równoprawne stosowanie przerzutników D oraz T. Jeszcze innym przykładem układów FPGA o takich możliwościach są układy z serii MAX 7000 firmy ALTERA [HławB94a, HławB94b]. Układ PLD typu XL78C800 firmy EXEL zawiera przerzutniki JK, które można zaprogramować jako przerzutniki typu T. Wreszcie układy FPGA typu ACT-3 firmy ACTEL umożliwiają wykorzystywanie modułów typu DFM8A jako przerzutników T lub D [Hławi97a].



Rys. 3.2. Realizacja opisanych tym samym wielomianem charakterystycznym rejestrów IED-LFSR (a) oraz DT-LFSR (b)

Fig. 3.2 Realization of linear registers with the same characteristic polynomial: register IED-LFSR (a), register DT-LFSR (b)

Rejestry $D^k T^f$ dla $f \leq 15$ można konstruować tylko w przypadku wykorzystania następującego wielomianu charakterystycznego (rozdział 3.2.1):

$$1 + x^k + \beta_t x^{k+2^t} + \beta_u x^{k+2^u} + \beta_t \beta_u x^{k+2^t+2^u} + \beta_w x^{k+2^w} + \beta_t \beta_w x^{k+2^t+2^w} + \beta_u \beta_w x^{k+2^u+2^w} + \beta_t \beta_u \beta_w x^{k+2^t+2^u+2^w}$$

Wszystkie związane z wielomianami pierwotnymi możliwe do zrealizowania rejestry $D^k T^f$ o długości $n = k + f \leq 63$ przedstawiono w tab. 3.6. [Hławi92a].

Analiza tej tabeli pozwala zauważyć, że nie istnieją rejestry $D^k T^f$ o następujących długościach n: 8, 12, 13, 14, 16, 19, 24, 26, 27, 30, 32, 34, 37, 38, 40, 42, 43, 44, 45, 46, 48, 50, 51, 53, 54, 56, 59, 61, 62. Dla wielomianów pierwotnych o takich stopniach

Tabela 3.6

n	Rejestry $D^k T^f$							
2	DT							
3	$D^2 T$	DT^2						
4	$D^3 T$							
5	$D^3 T^2$	$D^2 T^3$						
6	$D^5 T$	DT^5						
7	$D^6 T$	$D^4 T^3$	$D^3 T^4$	DT^6				
9	$D^5 T^4$	$D^4 T^5$						
10	$D^3 T^7$							
11	$D^9 T^2$	$D^2 T^9$						
15	$D^{14} T$	$D^{11} T^4$	$D^7 T^8$	$D^4 T^{11}$				
17	$D^{14} T^3$	$D^{12} T^5$	$D^{11} T^6$	$D^6 T^{11}$	$D^5 T^{12}$	$D^3 T^{14}$		
18	$D^7 T^{11}$							
20	$D^3 T^{17}$							
21	$D^{19} T^2$	$D^2 T^{19}$						
22	$D^{21} T$							
23	$D^{18} T^5$	$D^{14} T^9$	$D^9 T^{14}$	$D^5 T^{18}$				
25	$D^{22} T^3$	$D^{18} T^7$	$D^7 T^{18}$	$D^3 T^{22}$				
28	$D^{15} T^{13}$	$D^9 T^{19}$						
29	$D^2 T^{27}$	$D^{27} T^2$						
31	$D^{28} T^3$	$D^{25} T^6$	$D^{24} T^7$	$D^{18} T^{13}$	$D^{13} T^{18}$	$D^7 T^{24}$	$D^6 T^{25}$	$D^3 T^{28}$
33	$D^{20} T^{13}$	$D^{13} T^{20}$						
35	$D^{33} T^2$	$D^2 T^{33}$						
36	$D^{25} T^{11}$							
39	$D^{35} T^4$	$D^{31} T^8$	$D^{14} T^{25}$	$D^8 T^{31}$				
41	$D^{38} T^3$	$D^{21} T^{20}$	$D^{20} T^{21}$	$D^3 T^{38}$				
47	$D^{42} T^5$	$D^{33} T^{14}$	$D^{27} T^{20}$	$D^{26} T^{21}$	$D^{21} T^{26}$	$D^{20} T^{27}$	$D^{14} T^{33}$	$D^5 T^{42}$
49	$D^{40} T^9$	$D^{37} T^{12}$	$D^{34} T^{15}$	$D^{27} T^{22}$	$D^{22} T^{27}$	$D^{15} T^{34}$	$D^{12} T^{37}$	$D^9 T^{40}$
52	$D^{33} T^{19}$	$D^{21} T^{31}$	$D^3 T^{49}$					
55	$D^{32} T^{24}$							
57	$D^{35} T^{22}$	$D^7 T^{50}$						
58	$D^{39} T^{19}$							
60	$D^{59} T$	DT^{59}						
63	$D^6 T$	$D^{58} T^5$	$D^{32} T^{31}$	$D^{31} T^{32}$	$D^5 T^{58}$	DT^{62}		

n należy poszukiwać schematów rejestrów liniowych o strukturach IETD oraz EEDT zawierających tylko jedną bramkę XOR. Projektowanie takich rejestrów umożliwia np. wielomiany

$$p(x) = 1 + \sum_{i=0}^1 \oplus \left(x^{(k+a_i)} + \beta_t x^{(k+a_i)+2^i} + \beta_u x^{(k+a_i)+2^u} + \beta_t \beta_u x^{(k+a_i)+2^i+2^u} + \right. \\ \left. + \beta_w x^{(k+a_i)+2^w} + \beta_t \beta_w x^{(k+a_i)+2^i+2^w} + \beta_u \beta_w x^{(k+a_i)+2^u+2^w} + \beta_t \beta_u \beta_w x^{(k+a_i)+2^i+2^u+2^w} \right)$$

które można przekształcić w postać $1 + x^k(1 + x)^f[1 + x^{a^1}]$. Dobrze ilustruje to przykład 3.15. Poniżej przedstawiono przykład wielomianu o postaci wprawdzie różniące się od podanej, ale umożliwiającej również stosowanie tylko jednej bramki XOR w sprzężeniu liniowym.

Przykład 3.21

Wielomian HP stosowany w analizatorach sygnatur [Froh77, HławN84]:

$$p(x) = 1 + x^7 + x^9 + x^{12} + x^{16} = 1 + x^7(1 + x)^2[1 + x^5(1 + x)^2] \cong D^7T^2 \oplus D^5T^2.$$

Nie dla wszystkich wielomianów można znaleźć rejestry, które przy równoprawnym dostępie do przerzutników D oraz T umożliwiają realizację rejestrów wymagających użycia tylko jednej bramki XOR. Najlepiej ilustruje to tabela 3.7, w której przedstawiono najtańsze realizacje rejestrów o popularnych wielomianach charakterystycznych. Umieszczone w tej tabeli rejestry związane z wielomianami BCH oraz CRC-12 zawierają dwie i więcej bramek XOR w swoich sprzężeniach liniowych.

Tabela 3.7

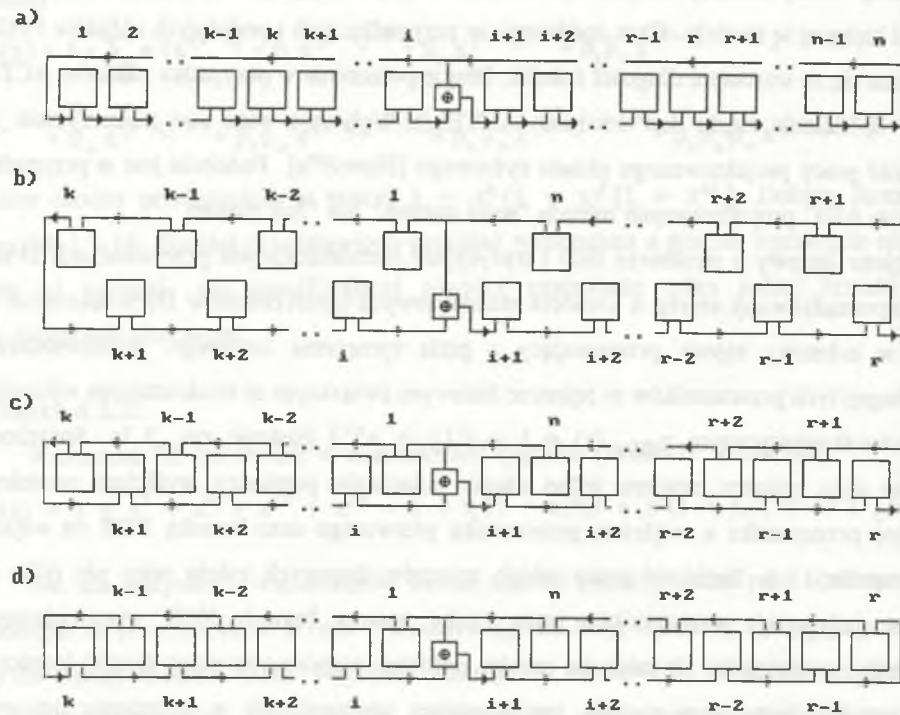
	IED	IETD
HP	$D \oplus D^2 \oplus D^3 \oplus D^4$	$D^7T^2 \oplus D^5T^2$
BCH	$D^4 \oplus D \oplus D \oplus D \oplus D^5 \oplus D^3 \oplus D$	$D^4 \oplus DT \oplus D^2T \oplus D^2 \oplus D^2T \oplus DT$
CRC-12	$D \oplus D \oplus D \oplus D^8 \oplus D$	$D \oplus DT \oplus D^9$
CRC-16	$D^2 \oplus D^{13} \oplus D$	$D^2 \oplus D^{13}T$
CRC-CCITT	$D^5 \oplus D^7 \oplus D^4$	$D^5 \oplus D^7T^4$

3.2.6. Sprzężenia liniowe bez długich połączeń

Opóźnienie pomiędzy dwoma logicznymi modułami w układach FPGA typu ACT-3 [ACTEL] zależy między innymi od obciążenia rezystancyjnego oraz pojemnościowego ścieżki łączącej te moduły. Czas opóźnienia w przypadku tych i podobnych układów FPGA zwiększa się ze wzrostem długości ścieżki. Długie połączenie w przypadku układów ACT-3 wnosi opóźnienie rzędu 4ns do 14ns [ACTEL]. Wpływają więc one niekorzystnie na szybkość pracy projektowanego układu cyfrowego [Hławi97a]. Podobnie jest w przypadku układów ASIC projektowanych metodą "semi custom" lub "full custom".

Rejestr liniowy o strukturze IED i tradycyjnie rozmieszczonych przerzutnikach D stanowi uporządkowany szereg n komórek standardowych (przerzutników D) połączonych ze sobą w n-bitowy rejestr przesuwający z pętlą sprzężenia liniowego. Rozmieszczenie (topologie) tych przerzutników w rejestrze liniowym związanym ze strukturalnym wielomianem charakterystycznym $p_{IEDw}(x) = 1 + x^1[1 + x^{n-1}]$ ilustruje rys. 3.3a. Sprzężenie liniowe tego rejestru zawiera jedno długie połączenie pomiędzy wyjściem ostatniego (n-tego) przerzutnika a wejściem przerzutnika pierwszego oraz bramką XOR na wejściu przerzutnika i+1. Szybkość pracy takich rejestrów liniowych zależy więc nie tylko od czasów propagacji przerzutników oraz liczby warstw bramek XOR wprowadzonych pomiędzy przerzutniki, ale także od czasów opóźnień wnoszonych przez ścieżki łączące te przerzutniki. Najprostszą metodą, umożliwiającą wprowadzenie w sprzężeniu liniowym krótkiego połączenia zamiast długiej ścieżki, jest technika projektowania rejestru liniowego w formie pierścienia z przerzutnikami rozmieszczonymi blisko siebie. Jeden z przykładów tak zaprojektowanego rejestru liniowego o sprzężeniu związanym z podanym poprzednio wielomianem charakterystycznym $p_{IEDw}(x)$ zilustrowano na rys. 3.3b. Przedstawiony pierścień zawiera dwa szeregi przerzutników. W górnym szeregu ścieżka łącząca n-ty przerzutnik z przerzutnikiem pierwszym jest krótkim połączeniem. Również krótkim połączeniem jest pionowa ścieżka łącząca wyjście n-tego przerzutnika z bramką XOR na wejściu przerzutnika i+1. Umieszczając pomiędzy przerzutnikami dolnego szeregu przerzutniki z górnego szeregu, otrzymuje się przeplatankę przerzutników zilustrowaną na rys. 3.3c (przeplatanka typu A). Inny sposób przeplatania przerzutników zilustrowano na rys. 3.3d (przeplatanka typu B). Można jeszcze tworzyć przeplatanki przerzutników, w których lewa strona jest przeplatanką typu A (B), a prawa strona jest przeplatanką typu

B (A). Ideę przeplatania przerzutników w jednym szeregu nie zawierającym długich połączeń opisano po raz pierwszy w pracy [BhavE97].



Rys. 3.3. Rozmieszczenie przerzutników rejestru liniowego: szereg uporządkowany (a), pierścieni (b), przeplatanka typu A (c), przeplatanka typu B (d)
 Fig. 3.3. Distribution of linear register flip-flops: row well-ordered (a), ring (b), an A type interlacing (c), a B type interlacing (d)

Autorzy tej pracy nie podali jednak reguł wskazujących sposób podziału uporządkowanego szeregu przerzutników (rys. 3.3a) na takie dwa rozłączne zbiory przerzutników, które, przeplatając się wzajemnie w jednym szeregu, nie wymagają długich połączeń. Zagadnienie to rozwiązano w tym rozdziale. Wybór k-tego przerzutnika dla lewej strony przeplatanki oraz r-tego przerzutnika dla prawej strony przeplatanki umożliwiają wyrażenia podane w tabeli 3.8. Pozwalają one określać liczby "k" oraz "r" na podstawie założonych wcześniej liczb "n" oraz "i".

Technikę tę można stosować również w przypadku rejestrów o strukturach IEDT związanych z następującymi strukturalnymi wielomianami charakterystycznymi: $P_{IEDT_w}(x) = 1 + x^b(1+x)^f[1+x^{n-1}]$ (rozdziały 3.2.1 oraz 3.2.2),

$P_{IEDT_w}(x) = 1 + x^i[1 + x^c(1+x)^g]$ (CRC-16 i CRC-CCITT w przykładzie 3.16), $P_{IEDT_w}(x) = 1 + x^b(1+x)^f[1 + x^c(1+x)^g]$ (HP w przykładzie 3.21), gdzie $b + f = i$ oraz $c + g = n - i$. Wówczas komórki standardowe przedstawione na rys. 3.3 pełnią funkcję przerzutników D lub T. Zauważmy także, że w przypadku takich rejestrów wartość współczynnika rozplywu WR nie jest większa od 2.

Tabela 3.8

Lp.	i+1	n	Lewa strona		Prawa strona	
			k	typ	r	typ
1	p	p	(i+1)/2	A	(n+i+1)/2	A
2	p	np	(i+1)/2	A	(n+i+2)/2	B
3	np	p	(i+2)/2	B	(n+i+2)/2	B
4	np	np	(i+2)/2	B	(n+i+1)/2	A

p - liczba parzysta; np - liczba nieparzysta

Przykład 3.22

Weźmy pierwotny wielomian charakterystyczny $p(x) = 1 + x^2 + x^4 + x^9 + x^{11}$. Odpowiadający mu wielomian strukturalny $P_{IEDT_w}(x) = 1 + x^2(1+x)^2[1+x^7]$ umożliwia zaprojektowanie następującego uporządkowanego szeregu przerzutników $D T D T \oplus D D D D D D D$, w którym $n = 11$ (liczba nieparzysta) oraz $i+1 = 5$ (liczba nieparzysta). W celu określenia liczb k oraz r wybieramy czwarty wiersz tab. 3.8, w którym $k = 3$ oraz $r = 8$. Na tej podstawie otrzymujemy następującą przeplatankę $D T T D \oplus D D D D D D D$.
 3 2 4 1 5 11 6 10 7 9 8

Również rejestry o strukturach EED oraz EEDT związanych z wielomianami $P_{EED_w}(x) = 1 + x^i[1 + x^{n-1}]$, $P_{EEDT_w}(x) = 1 + x^b(1+x)^f[1 + x^c(1+x)^g]$, $P_{EEDT_w}(x) = 1 + x^i[1 + x^c(1+x)^g]$ oraz $P_{EEDT_w}(x) = 1 + x^b(1+x)^f[1 + x^{n-1}]$ (rozdziały 3.2.1 oraz 3.2.2) można projektować bez długich połączeń stosując technikę przeplatania przerzutników. Technikę tę można także stosować w przypadku rejestrów o strukturach DT związanych ze strukturalnym wielomianem $P_{DT_w}(x) = 1 + x^b(1+x)^f$ (rozdział 3.2.5). Rejestry liniowe o strukturach CADT również nie wymagają długich połączeń. Sposoby ich projektowania przedstawia następny rozdział.

3.3. Projektowanie rejestrów o strukturze CADT

Ze względu na iteracyjny sposób określania wielomianu $p_{CADT}(x)$ niezwykle trudne jest projektowanie rejestrów o strukturach CADT w oparciu o założony wielomian charakterystyczny. Opisany ostatnio w [CattM96] algorytm określania ciągu komórek K_n w oparciu o zadany wielomian charakterystyczny jest złożony, co zniechęca do jego stosowania. Brak prostego i czytelnego związku pomiędzy niezerowymi współczynnikami wielomianu $p_{CADT}(x)$ a współczynnikami $k_0 k_1 k_2 k_3 \dots k_j \dots k_{n-1}$ zmusza do korzystania z katalogów rejestrów typu CA wcześniej utworzonych za pomocą specjalnych programów komputerowych. Część z nich to katalogi rejestrów zaprojektowanych w postaci łańcuchów sklejonych ze sobą plasterkowych pojedynczych komórek typu 1 oraz 0 [SerrM88, HortM89, GlosB89]. Inną część stanowią katalogi rejestrów zaprojektowanych jako łańcuchy sklejonych ze sobą kilkukomórkowych modułów plasterkowych [Hławi91b, HławK92a, Hławi93a, HławK93b]. Łańcuchy takie nazywane są dalej konkatencjami.

W niektórych programach komputerowych generowane są kolejno ciągi komórek K_n według z góry zadanego algorytmu. Programy takie określają metodą iteracyjną wielomiany charakterystyczne $p(x)$ związane z kolejno generowanymi ciągami komórek K_n . Proces ten zostaje zakończony w chwili, w której wielomian $p(x)$, związany z bieżąco sprawdzanym ciągiem K_n , jest identyczny z poszukiwanym wielomianem pierwotnym. Przykład 3.23 wyjaśnia tę ideę.

Przykład 3.23

Należy określić ciąg komórek $K_{10} = k_0 k_1 k_2 k_3 k_4 k_5 k_6 k_7 k_8 k_9$ rejestru CADT-LFSR związanego z wielomianem $p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$. Zakładamy hipotetyczny ciąg $K_{10} = 0110100111$. Następnie, stosując metodę iteracyjną wykorzystującą formułę $p_i(x) = p_{i-2}(x) + (k_{i-1} + x)p_{i-1}(x)$, sprawdzamy, jaki wielomian charakterystyczny związany jest z tym rejestrem CADT-LFSR.

Oto wyniki kolejnych obliczeń:

$$\begin{aligned} p_0(x) &= 1; p_1(x) = k_0 + x = x; p_2(x) = 1 + x + x^2; p_3(x) = 1 + x + x^3; \\ p_4(x) &= 1 + x^4; p_5(x) = x^3 + x^4 + x^5; p_6(x) = 1 + x^5 + x^6; \\ p_7(x) &= x + x^3 + x^4 + x^5 + x^6 + x^7; p_8(x) = 1 + x + x^2 + x^3 + x^5 + x^6 + x^8; \\ p_9(x) &= 1 + x + x^3 + x^6 + x^8 + x^9; p_{10}(x) = x + x^4 + x^5 + x^7 + x^{10} = p_{CA}(x) \neq p(x). \end{aligned}$$

Otrzymany wielomian nie jest zakładanym wielomianem charakterystycznym. Zmieniamy więc np. komórkę k_9 w ciągu K_{10} na $k_9' = 0$ i sprawdzamy ponownie.

$$\begin{aligned} p_{10}'(x) &= p_8(x) + (k_9' + x)p_9(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10} = \\ &= p_{CA}'(x) = p(x). \end{aligned}$$

W efekcie otrzymujemy rejestr 0110100110 o strukturze CADT.

W innych programach, między innymi zastosowanych przez autora do generacji rejestrów o strukturze CADT, wykorzystywane są plasterkowe moduły złożone z czterech i/lub pięciu komórek typu 0 oraz 1. Programy takie generują rejestry liniowe typu CA w postaci konkatencji wielokomórkowych modułów plasterkowych. Przypomina to ideę sklejania mikroprocesorów plasterkowych (ang. slice microprocessor). Takie moduły mogą tworzyć trzon biblioteki standardowych komórek [Hławi93a] służących do projektowania przy użyciu systemu CAD testerów wewnętrznych samotestowalnych układów ASIC.

3.3.1. Metoda selekcji ciągów K_n

Przeszukiwanie metodą kolejnych prób (przykład 3.23) zbioru ciągów K_n o liczności 2^n w przypadku dużych wartości n staje się niepraktycznym problemem typu NP o długim czasie realizacji. Czas ten mógłby zostać znacznie zredukowany, gdyby poszukiwania skoncentrować na zbiorze rejestrów typu CA związanych głównie z wielomianami pierwszymi. Jak więc uprościć określanie wielomianów niepierwszych związanych z odrzucanymi ciągami K_n ? Problem ten rozwiązano w pracach [Hławi90d, HławK92a, HławK93a]. W [Hławi90d] udowodniono, że znaczna grupa rejestrów typu CA związana jest z tzw. wielomianami charakterystycznymi ułomnymi typu $p(x) = x^k g(x)$. Rejestry 01101, 0110100 oraz 0110100111 w przykładzie 3.23 związane są odpowiednio z następującymi wielomianami ułomnymi:

$$\begin{aligned} p_5(x) &= x^3(1 + x + x^2), p_7(x) = x(1 + x^2 + x^3 + x^4 + x^5 + x^7) \\ \text{oraz } p_{10}(x) &= x(1 + x^3 + x^4 + x^6 + x^9). \end{aligned}$$

Takie rejestry nazwano w [Hławi90d] rejestrami ułomnymi. Opisany w [HławK93a] schemat zastępczy ułomnych rejestrów typu CA wskazuje na ich małą przydatność praktyczną. Liczba n -komórkowych rejestrów ułomnych stanowi 33.3% zbioru wszystkich rejestrów [HławK92a, HławK93a]. Podobny procent stanowią tzw. rejestry parzyste związane z wielomianami parzystymi typu $p(x) = (1 + x)^p h(x)$ [HławK92a, HławK93a]. Rejestry 0110, 0110100 oraz 011010011 podane w przykładzie 3.23 związane są odpowiednio z następującymi wielomianami

parzystymi $p_4(x) = (1 + x)^4$, $p_7(x) = (1 + x)(1 + x + x^2 + x^4 + x^6)$ oraz $p_9(x) = (1 + x)(1 + x^3 + x^4 + x^5 + x^8)$.

W pracy [HławiK93a] udowodniono, że dla dużych n liczność zbioru wszystkich n -komórkowych rejestrów ułomnych i/lub parzystych dąży do wartości $(5/9)2^n$, co stanowi 55.5% zbioru wszystkich n -komórkowych rejestrów typu CA. Poszukiwane przez nas rejestry związane z wielomianami pierwotnymi znajdują się więc w pozostałej grupie rejestrów stanowiącej niecałe 45% zbioru wszystkich rejestrów typu CA. Jak więc efektywnie odrzucać te ciągi komórek K_n , które są związane z wielomianami ułomnymi lub parzystymi i koncentrować poszukiwania wyłącznie w tym 45% fragmencie całego zbioru rejestrów? Prosta metoda takiej selekcji ciągów K_n opisano w pracach [HławiK92a, HławiK93a]. Oto jej ogólny zarys. Przyjmijmy:

$$K_{n-2} = k_0 k_1 k_2 \dots k_{n-3} \cong p_{n-2}(x) = x q_{n-2}(x) + a_{0,n-2}$$

$$K_{n-1} = K_{n-2} k_{n-2} \cong p_{n-1}(x) = x q_{n-1}(x) + a_{0,n-1}$$

$$K_n = K_{n-1} k_{n-1} \cong p_n(x) = x q_n(x) + a_{0,n}$$

gdzie ciąg K_{n-1} powstaje z ciągu K_{n-2} przez dodanie jednej komórki k_{n-2} , a ciąg K_n powstaje z ciągu K_{n-1} przez dodanie jednej komórki k_{n-1} . Wyrazami wolnymi wielomianów $p_{n-2}(x)$, $p_{n-1}(x)$, $p_n(x)$ są odpowiednio $a_{0,n-2}$, $a_{0,n-1}$ oraz $a_{0,n}$. Spełniają one następującą zależność [Hławi90d]:

$$a_{0,n} = a_{0,n-2} + k_{n-1} a_{0,n-1} \quad (3.18)$$

Cechą ciągu K_n nazwano w [HławiK92a] parę $\langle a_{0,n-1}, a_{0,n} \rangle = C(K_n)$. Cecha $C(K_n)$ nigdy nie jest równa $\langle 0,0 \rangle$ [HławiK92a], co oznacza, że wielomiany $p_{n-1}(x)$ oraz $p_n(x)$ nie mogą być jednocześnie ułomne. Pozostałe cechy $\langle 0,1 \rangle$, $\langle 1,1 \rangle$, $\langle 1,0 \rangle$ w celu uproszczenia zapisu oznaczać będziemy odpowiednio przez S, N, U. Na podstawie tych cech oraz 3.18 wyciągnięto następujące wnioski:

Wniosek 3.1 [Hławi90d]

Wielomian $p_n(x)$ jest ułomny wtedy, gdy:

- do ciągu komórek $K_{n-1} = K_{n-2} k_{n-2}$, w którym wielomian $p_{n-2}(x) \cong K_{n-2}$ jest ułomny, dołączono komórkę $k_{n-1} = 0$,
- do ciągu komórek $K_{n-1} = K_{n-2} k_{n-2}$, w którym wielomiany $p_{n-2}(x) \cong K_{n-2}$ oraz $p_{n-1}(x) \cong K_{n-1}$ są nieułomne, dołączono komórkę $k_{n-1} = 1$.

Wniosek 3.2 [Hławi90d]

Wielomian $p_n(x)$ będzie nieułomny wtedy, gdy:

- do ciągu komórek $K_{n-1} = K_{n-2} k_{n-2}$, w którym wielomian $p_{n-2}(x) \cong K_{n-2}$ jest ułomny, dołączona zostanie komórka $k_{n-1} = 1$,
- do ciągu komórek K_{n-1} , w którym wielomian $p_{n-1}(x) \cong K_{n-1}$ jest ułomny, dołączona zostanie dowolna komórka k_{n-1} ,
- do ciągu komórek $K_{n-1} = K_{n-2} k_{n-2}$, w którym wielomiany $p_{n-2}(x) \cong K_{n-2}$ oraz $p_{n-1}(x) \cong K_{n-1}$ są nieułomne, dołączona zostanie komórka $k_{n-1} = 0$.

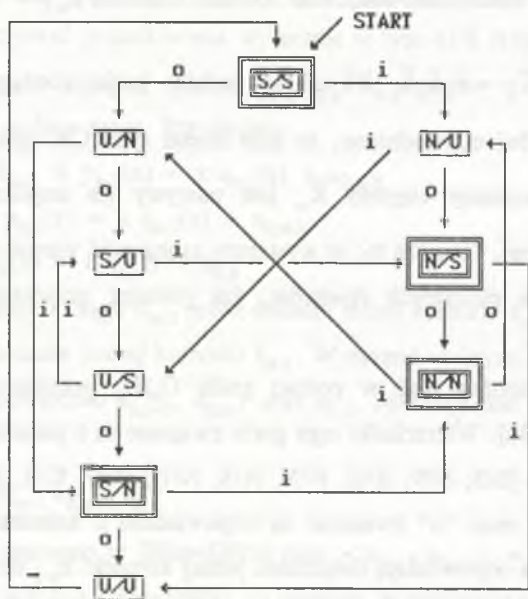
Załóżmy ciąg $\bar{K}_n = \bar{k}_0 \bar{k}_1 \bar{k}_2 \dots \bar{k}_{n-2} \bar{k}_{n-1}$ będący negacją ciągu K_n . W pracach [HławiK92a, HławiK93a] udowodniono, że jeśli rejestr typu CA opisany ciągiem K_n jest ułomny, to rejestr opisany ciągiem \bar{K}_n jest parzysty (ta implikacja zachodzi także w kierunku odwrotnym). Oznacza to, że wystarczy zanegować wartości komórek, aby mieć możliwość wykrywania parzystych rejestrów, jak również projektowania nieparzystych rejestrów.

Podane wnioski można ująć w postaci grafu G_u/G_p przedstawionego na rys. 3.4 [HławiK92a, HławiK93a]. Wierzchołki tego grafu związane są z parami cech należącymi do następującego zbioru $\{S/S, S/N, S/U, N/N, N/S, N/U, U/U, U/S, U/N\}$, natomiast łuki oznaczone literą "i" oraz "o" związane są odpowiednio z komórkami typu 1 oraz 0. Przejścia wzdłuż łuku odpowiadają dołączeniu jednej komórki k_{n-1} do ciągu K_{n-1} i uzyskaniu ciągu K_n . Wierzchołek z cechą U/x oznacza brak wolnego wyrazu w wielomianie $p_n(x) \cong K_n$ (rejestr ułomny), natomiast wierzchołki z cechami S/x i N/x oznaczają obecność wyrazu wolnego w wielomianie $p_n(x) \cong K_n$ (rejestr nieułomny). Podobnie wierzchołek z cechą x/U oznacza obecność dwumianu $(1 + x)$ w wielomianie p_n (rejestr parzysty), natomiast wierzchołki z cechami x/S i x/N oznaczają brak dwumianu $(1 + x)$ w wielomianie $p_n(x)$ (rejestr nieparzysty).

Graf G_u/G_p można wykorzystywać zarówno do wykrywania ułomnych/parzystych rejestrów, jak i do projektowania nieułomnych/nieparzystych rejestrów typu CA. Dołączanie do siebie kolejnych n komórek ciągu K_n jest równoważne przenoszeniu się wzdłuż n łuków grafu G_u związanych z komórkami $k_0 k_1 k_2 \dots k_{n-2} k_{n-1}$. Projektowanie ciągu K_n należy zaczynać od wierzchołka S/S grafu G_u/G_p [HławiK93a]. Następnie należy przejść wzdłuż łuków, projektowaną drogą, do wierzchołka związanego z jedną z następujących par

cech $\{S/S, N/N, S/N, N/S\}$. Przejście w stronę przeciwną, niż wskazuje strzałka w danym łuku, można traktować jak odłączenie pojedynczej komórki od projektowanego ciągu K_n . Tą metodą można również projektować rejestry typu CA. Wówczas projektowanie można zacząć w jednym z wierzchołków należących do zbioru $\{S/S, N/N, S/N, N/S\}$, a skończyć w wierzchołku S/S.

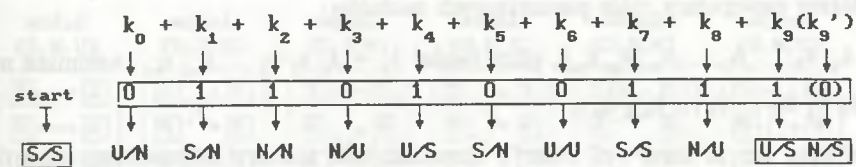
Zbadajmy, jak zmienia się cecha ciągu K_n w trakcie dołączania (odłączania) kolejnej komórki k_{n-1} .



Rys. 3.4. Graf G_u/G_p
Fig. 3.4. Graph G_u/G_p

Przykład 3.24

Weźmy ciąg $K_{10} = k_0k_1k_2k_3k_4k_5k_6k_7k_8k_9 = 0110100111$ z przykładu 3.23. Przyłączenie kolejnych komórek oznaczmy znakiem +. Zaczynamy w wierzchołku S/S grafu G_u/G_p . Dołączanie kolejnych komórek i uzyskiwane w efekcie cechy ilustruje rysunek 3.5. Dołączenie komórki $k_9 = 1$ powoduje, że osiągnięty zostaje wierzchołek U/S grafu, co świadczy, że uzyskany rejestr jest wprawdzie nieparzysty (S), ale zarazem niestety ułomny (U). Zamiana tej komórki na $k_9' = 0$ zamienia rejestr CADT-LFSR w rejestr jednocześnie nieparzysty i nieułomny, o czym świadczy para cech N/S ciągu K_{10} . Rezultaty uzyskane w tym przykładzie za pomocą grafu G_u/G_p całkowicie potwierdza przykład 3.23.



Rys. 3.5. Projektowanie rejestru CADT-LFSR nieułomnego/nieparzystego
Fig. 3.5. Designing of non-infirmutive/non-parity register CADT-LFSR

Wykorzystanie grafu G_u/G_p w programie określającym wielomiany charakterystyczne $p(x)$ na podstawie założonego rejestru typu CA ułatwia i przyspiesza przeszukiwanie zbioru ciągów komórek K_n oraz odrzucanie ciągów związanych z wielomianami ułomnymi i/lub parzystymi. Dodatkowym ułatwieniem mogą być rezultaty badań właściwości rejestrów typu CA zawarte w [HławK93a]. W pracy tej udowodniono, że każdy symetryczny ciąg komórek K_n jest związany z wielomianem ułomnym lub redukowalnym. Ponadto udowodniono, że zanegowany ciąg komórek \bar{K}_n jest ułomny i/lub parzysty, jeżeli ciąg komórek K_n jest odpowiednio parzysty i/lub ułomny. Nie bez znaczenia dla tej selekcji są także inne rezultaty badań. W pracy [Barde90] zaobserwowano, a w [HławK93a, CattM96] udokumentowano, że dla każdego wielomianu pierwszego, a więc i pierwotnego można znaleźć dwa związane z nimi rejestry typu CA opisywane ciągami komórek, które są wzajemnie odwrotne. Na przykład rejestr 0110010110 opisany odwrotnym ciągiem K_{10}^* do ciągu K_{10} podanego w przykładzie 3.23 związany jest również z wielomianem charakterystycznym $p(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10}$. Oznacza to, że w grupie rejestrów stanowiących niecałe 45% wszystkich rejestrów typu CA znajduje się zbiór rejestrów związanych ze wszystkimi wielomianami pierwszymi. Liczność tego zbioru jest dwa razy większa od licznosci zbioru rejestrów o strukturach IED lub EED i związanych z wszystkimi wielomianami pierwszymi. Tak więc również dwa razy większa jest szansa natrafienia na poszukiwany wielomian charakterystyczny.

3.3.2. Metoda wyboru wielokomórkowych modułów plasterkowych

Jakie wielokomórkowe moduły plasterkowe wybrać do projektowania nieułomnych i nieparzystych rejestrów typu CA? Podstawy teoretyczne rozwiązania tego problemu opracowano w [HławK92a]. Ich ogólny zarys przedstawiono poniżej.

Założmy następujący zbiór plasterkowych modułów:

$A = \{A_a, A_b, \dots, A_i, \dots, A_w, A_z, K_n\}$, gdzie moduł $A_i = a_0 a_1 a_2 \dots a_{i-2} a_{i-1}$, natomiast moduł $K_n = k_0 k_1 k_2 \dots k_i \dots k_{n-2} k_{n-1}$.

Umówmy się, że litery "o" oraz "i" oznaczać będą komórki odpowiednio typu 0 (90) oraz 1 (150). Umówmy się także, że znaki heksadecymalne służyć będą do oznaczania wszystkich czterokomórkowych modułów plasterkowych. Na przykład moduły "oooo" oraz "oooi" w skrócie zapisywane będą odpowiednio jako 0 oraz 1. Pięciokomórkowe moduły będą zapisywane za pomocą znaków heksadecymalnych i liter "o" oraz "i". Na przykład moduł "iiooi" oraz moduł "ooiio" oznaczać będziemy odpowiednio za pomocą "ci" oraz "3o".

Konkatenacją C plasterkowych modułów ze zbioru A jest więc dowolne połączenie tych modułów w jeden łańcuch, który można przedstawić w następujący sposób:

$C = M_1 + M_2 + \dots + M_i + \dots + M_{j-1} + M_j$, gdzie $M_i \in A$; $j \geq 2$

Na przykład dla $j = 2$ mamy $C = A_d + K_n = a_0 a_1 a_2 \dots a_{d-2} a_{d-1} k_0 k_1 k_2 \dots k_{n-2} k_{n-1}$.

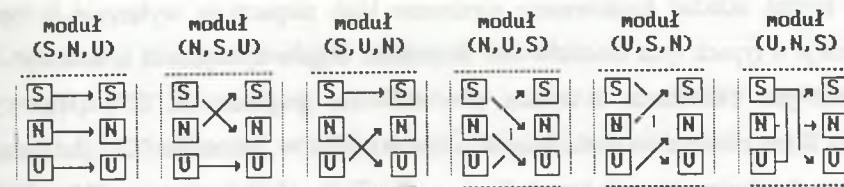
Jeżeli w grafie G_u/G_p oznaczymy przez x wierzchołek, od którego rozpoczynamy analizę (projektowanie), natomiast przez y oznaczymy wierzchołek, na którym kończy się ten proces, wówczas funkcja $y = \beta_K(x)$ określać będzie zależność y od wierzchołka x oraz od ciągu $k_0 k_1 k_2 \dots k_{n-2} k_{n-1}$.

Charakterystyką plasterkowego modułu K_n nazywamy funkcję $\beta_K(x)$ spełniającą następujące warunki [HławK92a]:

- 1) Dla wszystkich $x \in \{S, N, U\}$; $\beta_K(x) \in \{S, N, U\}$;
- 2) $\beta_K(x) = C(K_n)$;
- 3) K_n jest modułem pustym $\Rightarrow \beta_K(x) = x$;
- 4) $K_n = k_i = 0 \Rightarrow \beta_K(S) = U$; $\beta_K(N) = N$; $(U) = S$;
- 5) $K_n = k_i = 1 \Rightarrow \beta_K(S) = N$; $\beta_K(N) = U$; $(U) = S$;
- 6) Dla wszystkich $x \in \{S, N, U\}$; $\beta_{A+K}(x) = \beta_K(\beta_A(x))$ albo $\beta_{A+K} = \beta_A \square \beta_K$,

gdzie znak " \square " określa składanie funkcji β .

Natomiast typem modułu K_n nazywamy trójkę uporządkowaną [HławK92a]: $T_K = (\beta_K(S), \beta_K(N), \beta_K(U))$. Wszystkie typy modułów tworzą zbiór $Z = \{(S, N, U), (N, S, U), (S, U, N), (N, U, S), (U, S, N), (U, N, S)\}$. Charakterystyki tych modułów przedstawia rys. 3.6.



Rys. 3.6. Charakterystyki wszystkich typów plasterkowych modułów
Fig. 3.6. The characteristics of all types of slice modules

Zauważmy, że moduł jednokomórkowy $k_i = 1$ posiada charakterystykę modułu typu (N,U,S), zaś komórka $k_i = 0$ posiada charakterystykę ułomnego modułu typu (U,N,S). Natomiast plasterkowe moduły kilkukomórkowe należące np. do zbiorów $\{4, b, 3o, co\}$, $\{a, 5, 3i, 9i, ci, eo\}$ oraz $\{2, d, 1i, 6o\}$ posiadają charakterystyki odpowiednio modułów typu (S,U,N), (S,N,U), i (N,S,U).

Jeżeli zarówno moduł K_n , jak i jego negacja \bar{K}_n są typu $T_K \in \{(S, N, U), (S, U, N), (N, S, U), (N, U, S)\}$, to moduł K_n jest nieułomny i nieparzysty [HławK92a]. Przykładem modułów zarówno nieułomnych, jak i nieparzystych są moduły zawarte w zbiorach: $\{2, 4, 5, 1i, 3o, 3i, 9i\}$ oraz $\{d, b, a, eo, ci, co, 6o\} = \{\bar{2}, \bar{4}, \bar{5}, \bar{1i}, \bar{3o}, \bar{3i}, \bar{9i}\}$. Potwierdza to graf G_u/G_p .

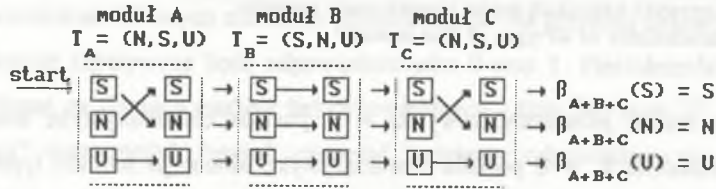
Ideę składania funkcji $\beta_K(x)$ związanych z różnymi typami plasterkowych modułów przedstawia następujący przykład:

Przykład 3.25

Weźmy trzy następujące typy modułów: moduł A typu $T_A = (N, S, U)$, moduł B typu $T_B = (S, N, U)$ oraz moduł C typu $T_C = (N, S, U)$. Złożmy je w jedną nieułomną konkatenację. Sposób składania funkcji $\beta_{A+B+C}(x) = \beta_A(x) \square \beta_B(x) \square \beta_C(x)$ przedstawiono na rys. 3.7.

Zauważmy, że w podanym przykładzie w efekcie składania funkcji $\beta_A(x) \square \beta_B(x)$ składane są także typy modułów. W rezultacie $T_{A+B} = (\beta_A(S) \square \beta_B(S), \beta_A(N) \square \beta_B(N), \beta_A(U) \square \beta_B(U)) = (\beta_{A+B}(S), \beta_{A+B}(N), \beta_{A+B}(U)) = T_A \square T_B$. W przykładzie tym nieznanne były ciągi komórek typu 0 oraz 1 z modułów A, B oraz C. Pomimo to można było określić, czy przykładowa potencjalna konkatenacja złożona z tych modułów była nieułomna. Nasuwa się w związku z tym następujące pytanie:

Czy można składać konkatencje nieułamne i/lub nieparzyste wyłącznie w oparciu o informacje o typach tych modułów bez znajomości ciągów opisujących te moduły? Odpowiedź jest twierdząca i wynika z właściwości grupoidu $(Z, "□")$ [HławK92a], w którym Z jest zbiorem wszystkich sześciu typów modułów, natomiast "□" jest dwuargumentowym działaniem na tych typach $T_{A+B} = T_A □ T_B$ zdefiniowanym w [HławK92a].



Rys. 3.7. Połączenie modułów dające konkatencję nieułamną typu $T_{A+B+C} = (\beta_{A+B+C}(S), \beta_{A+B+C}(N), \beta_{A+B+C}(U)) = (S, N, U)$
 Fig. 3.7. Connection of modules creating a type of non-infirmite concatenation of $T_{A+B+C} = (\beta_{A+B+C}(S), \beta_{A+B+C}(N), \beta_{A+B+C}(U)) = (S, N, U)$

Twierdzenie 3.1 [HławK92a]

Jeżeli zbiór A zawiera wyłącznie takie moduły, że zarówno one same, jak i ich negacje są typu $T_K \in \{(S, N, U), (S, U, N)\}$ albo typu $T_K \in \{(S, N, U), (N, S, U)\}$, to każda konkatencja złożona z modułów z tego zbioru jest nieułamna i nieparzysta.

Dowód:

Algebry $(\{(S, N, U), (S, U, N)\}, "□")$ oraz $(\{(S, N, U), (N, S, U)\}, "□")$ z dwuargumentowymi działaniami "□" zdefiniowanymi poniżej są grupoidami.

	T_A	
T_B	(S, N, U)	(S, U, N)
(S, N, U)	(S, N, U)	(S, U, N)
(S, U, N)	(S, U, N)	(S, N, U)

$T_{A+B} = T_A □ T_B$

	T_A	
T_B	(S, N, U)	(N, S, U)
(S, N, U)	(S, N, U)	(N, S, U)
(N, S, U)	(N, S, U)	(S, N, U)

$T_{A+B} = T_A □ T_B$

□

Twierdzenia tego nie spełniają plasterkowe jednokomórkowe moduły w postaci komórek typu 0 oraz 1. Również zaden zbiór dwukomórkowych oraz trzykomórkowych modułów plasterkowych nie spełnia tego twierdzenia. Natomiast spełniają to twierdzenie niektóre moduły cztero- i pięciokomórkowe. Potwierdza to następujący wniosek:

Wniosek 3.3

Konkatencje złożone z modułów należących do zbioru $A_{45} = \{5, a, 4, b, 3o, ci, 3i, co, \}$ albo do zbioru modułów odwrotnych $A_{45}^* = \{a, 5, 2, d, 6o, 9i, eo, li, \} = \{(5)^*, (a)^*, (4)^*, (b)^*, (3o)^*, (ci)^*, (3i)^*, (co)^*\}$ są zawsze nieułamne i nieparzyste.

Dowód:

Moduły należące do zbiorów A_{45} oraz A_{45}^* są modułami typu odpowiednio $T_K \in \{(S, N, U), (S, U, N)\}$ oraz $T_K \in \{(S, N, U), (N, S, U)\}$. □

Ze względu na to, że każdy moduł ze zbioru A_{45}^* jest odwróconym modułem ze zbioru A_{45} , konkatencje modułów ze zbioru A_{45}^* będą zawsze odwróconymi konkatencjami uzyskanymi z modułów zbioru A_{45} . Każdy wielomian pierwotny związany jest zawsze z dwoma rejestrami typu CA, które są opisane za pomocą dwóch wzajemnie odwrotnych ciągów komórek. Wystarczy więc tylko jeden z tych zbiorów potraktować jako podstawę poszukiwań konkatencji.

Niech λ oznacza liczbę wielomianów pierwotnych stopnia n . Liczba ta określa jednocześnie połowę rejestrów typu CA związanych z wielomianami pierwotnymi. Załóżmy, że generowanie konkatencji jest losowaniem rejestru typu CA. Załóżmy także, że zbiór wszystkich rejestrów realizowalnych na bazie modułów wziętych z A_{45} albo z A_{45}^* jest statystycznie reprezentatywną próbką zbioru wszystkich rejestrów nieułamnych i nieparzystych.

Przy takim założeniu prawdopodobieństwo $P_{4,5}$ wylosowania rejestru typu CA związanego z wielomianem pierwotnym podczas przeszukiwania konkatencji złożonych z modułów ze zbioru A_{45} albo zbioru A_{45}^* dąży dla $n \gg 1$ do wartości $2.25 \lambda / 2^n$ [HławK93a]. Prawdopodobieństwo to dla różnych $n \leq 24$ ilustruje tabela 3.9.

Kolumna z wartościami $1/P_{4,5}$ określa liczbę losowań gwarantującą znalezienie co najmniej jednej konkatencji związanej z wielomianem pierwotnym (konkatencji pierwotnej). Sprawdźmy więc, jakie są minimalne podzbiory modułów plasterkowych ze zbiorów A_{45} oraz A_{45}^* , które zapewniają pożądaną liczbę losowań. Załóżmy na początku, że tylko jeden czterokomórkowy albo tylko jeden pięciokomórkowy moduł z tych zbiorów będzie służył do generacji konkatencji. Takie założenie zapewnia tylko jedno losowanie, a więc stanowczo za mało, aby zagwarantować znalezienie konkatencji pierwotnej.

Tabela 3.9

n	λ	$P_{4,5}$	$1/P_{4,5}$	n	λ	$P_{4,5}$	$1/P_{4,5}$
8	16	0.1566	6.4	17	7110	0.1470	6.8
9	48	0.2343	4.2	18	8064	0.0768	13.0
10	60	0.1465	6.8	19	27594	0.1315	7.6
11	175	0.2135	4.6	20	24000	0.0572	17.4
12	144	0.0878	11.4	21	84672	0.1008	9.9
13	630	0.1922	5.2	22	120032	0.0715	13.9
14	756	0.1152	8.7	23	356960	0.1063	9.4
15	1800	0.1372	7.3	24	276480	0.0411	24.3
16	2048	0.0781	12.8				

Potwierdziły to praktyczne wyniki uzyskane za pomocą programu komputerowego. Okazuje się, że nie ma konkatenacji pierwotnych o długości $n \in \{10, 20\}$ złożonych z takich samych pięciokomórkowych modułów ze zbiorów A_{45} lub A_{45}^* . Podobnie nie można znaleźć konkatenacji pierwotnych o długości $n \in \{8, 16, 20, 24\}$ złożonych z identycznych modułów czterokomórkowych należących do tych samych zbiorów [Hławk93b]. Dla podzbiorów zawierających dwa różne moduły liczba możliwych losowań wzrasta. Określa ją wyrażenie 2^j , gdzie j określa liczbę modułów w konkatenacji. Dla konkatenacji budowanych z modułów czterokomórkowych o długości $n \in \{8, 12, 16, 20, 24\}$ liczba możliwych do wygenerowania konkatenacji wynosi odpowiednio $\{4, 8, 16, 32, 64\}$. Tak więc dla $n \in \{8, 12\}$ liczby te nieznacznie są mniejsze od minimalnej liczby losowań $1/P_{4,5}$ podanej w tabeli 3.9, natomiast już dla $n \geq 16$ liczby te są większe. W związku z tym podstawą do obliczeń konkatenacji pierwotnych może być każdy podzbiór zawierający co najmniej dwa różne moduły. Na przykład podzbiory $\{a, b\}$, $\{d, 5\}$ oraz $\{a, b, 4, 5\}$ były podstawą opracowania katalogów konkatenacji pierwotnych stopnia $n \times 4$ [Hławi91b, Hławk92a, Hławi93a, Hławk93b]. Dla podzbioru $\{d, 5\}$ opracowano w pracy [Hławk93b] katalog zawierający konkatenacje pierwotne od stopnia $n=4$ do stopnia $n = 320$. Jako przykład możliwości obliczeniowych specjalnego programu komputerowego podano w [Hławk93b] konkatenację pierwotną $C = d^{115} + 5dd5d5$ stopnia $n = 484$. Dla dwóch ośmiokomórkowych modułów $\{45, ef\}$ również opracowano katalog konkatenacji pierwotnych [Hławk93b]. Jego fragment ograniczony do konkatenacji stopnia $n \leq 160$ przedstawiono w tabeli 3.10.

Podzbiór modułów $\{4, 5, 3i, ci\}$ zawierający po dwa różne moduły cztero- i pięciokomórkowe był podstawą obliczeń konkatenacji pierwotnych dla każdego n zawartego w prze-

Tabela 3.10

n		n	
8	45	88	454545454545ef454545ef
16	45ef	96	45454545454545ef4545ef45
24	454545	104	454545454545454545ef45ef45
32	efef4545	112	4545454545454545ef4545ef4545
40	4545454545	120	4545454545454545efefefef4545ef45
48	4545ef454545	128	45454545454545454545efef4545ef45ef
56	4545efefef4545	136	4545454545454545454545efef45efef45ef
64	454545ef45454545	144	454545454545454545454545efef45ef45efef
72	4545ef454545efefef	152	454545454545454545454545454545ef45efef
80	45454545ef454545ef45	160	454545454545454545454545454545454545ef

dziale $12 \leq n \leq 22$. Konkatenacje te przedstawiono w tab. 3.11. Liczność zbiorów konkatenacji dla $n = 17$ i $n > 20$ jest dużo większa od liczby 16. Ilustruje to wyraźnie tabela 3.11, w której ze względu na brak miejsca dodano tylko jeden dodatkowy wiersz. Każda odwrócona konkatenacja z tej tabeli jest też związana z wielomianem pierwotnym. Konkatenacje te budowane są z modułów $\{2, a, eo, 9i\} = \{(4)^*, (5)^*, (3i)^*, (ci)^*\}$. Zwiększanie liczby modułów wpływa na wzrost liczby otrzymywanych konkatenacji pierwotnych. Bardzo dużą ich liczbę można uzyskać dla wszystkich plasterkowych modułów zbioru A_{45} albo zbioru A_{45}^* .

3.3.3. Efektywne projektowanie konkatenacji pierwotnych

Dotychczasowe rozważania dotyczyły projektowania pojedynczych rejestrów typu CA. Czasami istnieje jednak potrzeba zaprojektowania zbioru kilku pierwotnych konkatenacji różniących się między sobą nieznacznie liczbą co najwyżej kilkunastu komórek. W celu zwiększenia wydajności takiego projektowania można znaleźć taki zbiór tych konkatenacji, w którym rejestry posiadają bardzo długi wspólny r-komórkowy rdzeń. Rejestry z tego zbioru różnią się więc między sobą jedynie fragmentem dołączanym do rdzenia nazywanym dalej uzupełnieniem. W przypadku wykorzystywania modułów ze zbioru $\{4, 5, ci, 3i\}$ tym uzupełnieniem jest konkatenacja kilku modułów z tego zbioru o wspólnej długości $n-r$ komórek. Utworzone w ten sposób rejestry liniowe złożone z rdzenia i jego uzupełnienia stanowią katalog n -komórkowych konkatenacji pierwotnych. Przykład takiego katalogu zilustrowano w tab. 3.12. Przedstawiono w niej trzy przykłady podzbiorów rejestrów liniowych o wspólnym rdzeniu. Pierwszy z nich (a) z rdzeniem typu "cicici" = "ci³" umożliwia

Tabela 3.11

12	13	14	15	16	17	18	19	20	21	22
454	44ci	43i3i	cicici	4554	4ci55	4ci3i4	43i3i3i	45445	4ci455	453i4ci
545	45ci	ci5ci	cici3i	4445	4ci54	53i3i4	43icici	44545	4553i5	45ci3i5
555	4ci4	cici4	ci3ici	5445	4ci44	53i43i	53i3i3i	55454	45453i	45ci3i4
-	43i4	ci3i5	ci3i3i	5545	43i54	5ci4ci	53i3ici	3icicici	45443i	45cici5
-	53i5	3i53i	-	5554	443i4	543i3i	5ciicici	ci3icici	44ci45	45cici4
-	ci44	3i3i5	-	-	44ci5	55cici	ci43i3i	-	4453i4	4553ici
-	ci54	-	-	-	4443i	ci543i	ci5ci3i	-	445ci4	443i3i5
-	-	-	-	-	53i44	ci5ci5	ci3i5ci	-	44553i	444cici
-	-	-	-	-	53i54	cici44	ci3ici4	-	4443i5	53i3i54
-	-	-	-	-	53i55	cici54	3i4cici	-	444ci5	53i43i5
-	-	-	-	-	5ci45	3i443i	3i43ici	-	53i454	53i453i
-	-	-	-	-	5443i	3i4ci4	3i43i3i	-	543i55	53i53i4
-	-	-	-	-	544ci	3i543i	3ici5ci	-	5443i4	53i55ci
-	-	-	-	-	5453i	3ici44	3i3ici4	-	5444ci	54ci3i4
-	-	-	-	-	5553i	-	3i3ici5	-	5445ci	544ci3i
-	-	-	-	-	ci444	-	3i3i3i4	-	54453i	553i3i5
-	-	-	-	-	ci545	-	-	-	55ci45	5553i3i

projektowanie n-komórkowych rejestrów typu CA dla $n \geq 27$. Drugi z tych podzbiorów (b) posiada rdzeń "cicicicicici" = "ci⁶", który umożliwia projektowanie rejestrów dla $n \geq 44$. Wreszcie trzeci podzbiór (c) o rdzeniu "ci⁸" pozwala na zaprojektowanie rejestrów dla $n \geq 56$. Weźmy na przykład następujące pierwotne konkatenacje $\cdot ci^8 + 3i43i3i$, $ci^8 + 45554$, $ci^8 + 5ci545$ odpowiednio 59-, 60- i 61-komórkowe. Prawie cały wysiłek w tych przykładach wkładany jest jedynie w prace związane z utworzeniem trzech przedstawionych wyżej uzupełnień rdzenia "ci⁸". Wysiłek takiego projektowania porównać można do wysiłku związanego z projektowaniem konkatenacji zawierających co najwyżej 5 plasterkowych modułów o liczbie $n - r \equiv 20$ komórek. Ponieważ konkatenacja "cicicicicicici" jest konkatenacją pierwotną, można ją potraktować jako podstawę do zaprojektowania komórki standardowej, która wraz z komórkami standardowymi ze zbioru {4,5,ci,3i} może stanowić bibliotekę 5 standardowych komórek do efektywnego projektowania rejestrów o $n \geq 56$. Zauważmy, że przy użyciu tego rdzenia można także zaprojektować kilka rejestrów zawierających nieco mniej komórek. Są to rejestry o $n \in \{40,45,49, 52,53\}$.

Wydajność opisanego projektowania w przypadku każdego przykładowego rdzenia będzie rosła wraz ze wzrostem liczby n. Im dłuższy rdzeń, tym wydajniejsze projektowanie. Praca związana z takim projektowaniem nigdy nie jest większa niż praca związana

z projektowaniem konkatenacji złożonych z co najwyżej 5 plasterkowych modułów ze zbioru np. {4,5,ci,3i}.

Przedstawiona technika efektywnego projektowania nadaje się także do projektowania pojedynczych rejestrów typu CA.

Tabela 3.12

n	rdzeń i jego uzupełnienia	rdzeń i jego uzupełnienia
17	ci + 555	40 ci ⁸
18	ci ² + 44	41 ci ⁵ + 5555
19	ci + 3ici4	42 ci ⁴ + 3ici555
20	ci + 3icici	43 ci ⁵ + 53ici5
21	ci + 5445	44 ci ⁶ + 3i3i5 = ci ⁶ + 3i3i5
22	ci ² + 444	45 ci ⁶ = ci ⁶ + cici3i
23	ci ² + 553i	46 ci ⁶ + 5444 = ci ⁶ + 5444
24	ci ² + 3i4ci	47 ci ⁶ + 55ci4 = ci ⁶ + 55ci4
25	ci + 3i3icici	48 ci ⁶ + 5ci4ci = ci ⁶ + 5ci4ci
26	ci ² + 5555	49 ci ⁶ + 4 = ci ⁶ + cici3i4
27	ci ³ + 454 = cici3i4 + 454	50 ci ⁶ + 3icici3i = ci ⁶ + 3icici3i
28	ci ³ + 543i = cici3i + 543i	51 ci ⁶ + 5ci445 = ci ⁶ + 5ci445
29	ci ³ + 4cici = cici3i + 4cici	52 ci ⁸ + 455
30	ci ⁴ + 3ici = cici3i + ci3ici	53 ci ⁸ + 55ci
31	ci ³ + 4455 = cici3i + 4455	54 ci ⁷ + 3i43i3i
32	ci ³ + 553i4 = cici3i + 553i4	55 ci ⁷ + 3icicici
33	ci ⁴ + 3i54 = cici3i + ci3i54	56 ci ⁸ + 4455 = ci ⁸ + 4455
34	ci ⁴ + 3ici5 = cici3i + ci3ici5	57 ci ⁹ + 545 = ci ⁸ + ci545
35	ci ⁵ + 3i3i	58 ci ⁸ + 3i443i = ci ⁸ + 3i443i
36	ci ⁴ + 5445	59 ci ⁸ + 3i43i3i = ci ⁸ + 3i43i3i
37	ci ⁵ + 454	60 ci ⁸ + 45554 = ci ⁸ + 45554
38	ci ⁴ + 55cici	61 ci ⁸ + 5ci545 = ci ⁸ + 5ci545
39	ci ⁷ + 5	62 ci ⁹ + 54ci4 = ci ⁸ + ci54ci4
		63 ci ¹⁰ + 43i4 = ci ⁸ + cici43i4

3.4. Łatwo modyfikowalne rejestry liniowe

W trakcie tworzenia projektu samotestowalnego układu ASIC powstaje czasami konieczność przeprojektowania tego układu lub wyodrębnionego w nim modułu. W konsekwencji może to wpłynąć na zmianę liczby ich wejść i wyjść pierwotnych. To z kolei powoduje, że niezbędne staje się przeprojektowanie rejestrów LFSR podłączonych do ich wejść pierwotnych lub rejestrów MISR połączonych z ich wyjściami pierwotnymi. Takie przeprojektowywanie wymaga pewnego nakładu dodatkowych kosztów. W celu ich zmniejszenia należałoby rejestry liniowe tak projektować, aby były łatwo modyfikowalne.

Zagadnienie to dla rejestrów o strukturze CADT zasygnalizowano w [GlosB89], natomiast rozwiązano go częściowo w [Hławi90d, HławK92a].

Dodatkowe lub usunięte wejścia/wyjścia (we/wy) mogą zarówno przeplatać się z pozostawionymi bez zmian we/wy, jak również mogą być umieszczone z prawej lub lewej strony zbioru pozostawionych bez zmian wyprowadzeń we/wy. Zaproponowanie efektywnej techniki przeprojektowywania rejestrów liniowych we wszystkich podanych sytuacjach jest trudnym problemem. Zagadnienie to upraszcza się, jeśli ograniczy się uwagę tylko do tych przypadków, w których idea przeprojektowywania będzie polegała na dodawaniu (odejmowaniu) niezbędnych (zbędnych) komórek do (od) lewej lub prawej albo obu stron rejestru liniowego.

Umówmy się, że przeprojektowywanie n -komórkowych rejestrów liniowych będzie polegało na odjęciu c komórek, a następnie dodaniu $c + d$ albo $c - d$ nowych komórek. W związku z tym końcowa postać zmodyfikowanego rejestru będzie zawierała $n + d$ albo $n - d$ komórek. Fragment przeprojektowywanego rejestru, który nie ulega modyfikacjom, będzie zawierał $r = n - c$ komórek. Koszt przeprojektowywania będziemy określali liczbą c . Im koszt ten będzie mniejszy, tym bardziej rejestr będzie podatny na przeprojektowywanie. Jeżeli koszt $c = 0$, to o zmodyfikowanym rejestrze powiemy, że jest łatwo modyfikowalny. Takie modyfikowanie powoduje jednak, że nowy rejestr liniowy będzie związany z innym wielomianem charakterystycznym. Zarówno w przypadku generacji testów, jak również w przypadku kompaktacji rejestry liniowe związane z wielomianem pierwotnym dobrze spełniają wymagania związane z samotestowaniem [Hławi93b]. W związku z tym nowy wielomian charakterystyczny związany ze zmodyfikowanym rejestrem liniowym też powinien być pierwotny.

3.4.1. Łatwo modyfikowalne rejestry typu CA

W przypadku rejestrów o strukturze CADT metoda modyfikowania jest podobna do techniki projektowania tych rejestrów przy użyciu grafu G_u/G_p (rys. 3.4) lub techniki projektowania konkatenacji w oparciu o założony zbiór plasterkowych modułów. Dobrym przykładem rejestrów podatnych na przeprojektowywanie są rejestry o strukturze CADT w postaci konkatenacji złożonej z rdzenia i jego uzupełnienia (tab. 3.12). Załóżmy np., że wzięty z tej tabeli rejestr "ci⁸ + ci⁴" o 40-komórkowym rdzeniu i 9-komórkowym uzupełnieniu należy przeprojektować w rejestr 53-komórkowy. Wówczas wybieramy z tej

tabeli rejestr "ci⁸ + 55ci" złożony z identycznego rdzenia oraz z 13-komórkowego uzupełnienia. Przeprojektowanie polegać więc będzie na odjęciu uzupełnienia "ci⁴" zawierającego $c = 9$ komórek 1 oraz 0, pozostawieniu rdzenia "ci⁸" i dodaniu innego uzupełnienia "55ci" zawierającego $c + d = 13$ komórek. Koszt takiej modyfikacji wynosi $c = 9$.

Do dalszych rozważań zakładamy, że liczba d nie jest znana w momencie tworzenia projektu rejestru n -komórkowego. Jednak dla uproszczenia dyskusji teoretycznych przyjmujemy, że $d \leq 4$. Załóżmy także, że przeprojektowywanie, mające na celu wydłużenie rejestru, będzie polegać na tworzeniu konkatenacji złożonej z modyfikowanego rejestru opisanego ciągiem K_n (modułu K_n) oraz d modułów jednokomórkowych "o" oraz "i". Znając parę cech modułu K_n lub jego charakterystykę oraz charakterystyki komórek "o" oraz "i" można, za pomocą grafu G_u/G_p lub grupoidu $(Z, "□")$ [HławK92a], określać te zbiory d komórek, które dodane do prawej strony modułu K_n nigdy nie zwiążą rejestru, uzyskanego po modyfikacji, z wielomianem ułomnym i/lub parzystym. Niestety nie istnieje taki n -komórkowy rejestr opisany ciągiem K_n i związany z wielomianem pierwotnym, do którego można tak dodać kolejno dwie pojedyncze komórki k_n oraz k_{n+1} , że otrzymane po każdym kroku dwie następujące modyfikacje rejestru $K_n + k_n$ oraz $K_n + k_n + k_{n+1}$ też będą związane z wielomianami pierwotnymi [Hławi90d, HławK92a, HławK93a]. Wniosek ten pozwala stwierdzić, że podatność rejestrów typu CA na przeprojektowywanie nie odpowiada założonym oczekiwaniom.

W tab. 3.13 przedstawiono wszystkie możliwe ciągi A_d dla $d \leq 4$, które dodane z prawej strony pierwotnych rejestrów opisanych ciągiem K z parą cech należącą do zbioru $\{S/S, N/N, S/N, N/S\}$ pozwalają uzyskać konkatenację $K_n + A_d$ opisującą nieułomny i nieparzysty zmodyfikowany rejestr o strukturze CADT.

Tabela 3.13 powstała przy założeniu, że nienaruszalnym rdzeniem jest ciąg K_n . Prostopokątnymi nawiasami zaznaczono te grupy identycznych ciągów A_d , które wraz z ciągiem K_n mogą stanowić dłuższe rdzenie także nienaruszalne w trakcie przeprojektowywania. Tylko wśród rejestrów zmodyfikowanych przy użyciu przedstawionych w tab. 3.13 ciągów A_d można szukać rejestrów związanych z wielomianem pierwotnym. Zauważmy, że tylko pierwotne rejestry opisane ciągiem K_n z parą cech S/N lub N/S po dodaniu do nich z prawej strony jednej komórki 1 lub 0 mogą dać w efekcie pierwotną modyfikację tych rejestrów.

Tabela 3.13

	d	ciągi A_d dodawane z prawej strony rejestru typu CA										
S/S	2											
	3		$\overline{01}$		$\overline{10}$							
	4	$K_n +$	$\overline{01}$	i	$\overline{10}$	o	$\overline{001}$	$\overline{110}$	i			
			$\overline{01}$	o-	$\overline{10}$	i-	$\overline{001}$	o	$\overline{110}$	i	oooo	iiii
N/N	2		$\overline{00}$		$\overline{11}$							
	3	$K_n +$	$\overline{00}$	o	$\overline{11}$	i	$\overline{010}$	$\overline{101}$	o			
	4		$\overline{00}$	i-	$\overline{11}$	o-	$\overline{010}$	i	$\overline{101}$	o	oiii	i000
			$\overline{00}$	i-	$\overline{11}$	o-	$\overline{010}$	i	$\overline{101}$	o		
S/N	1		$\overline{1}$		$\overline{1}$							
	2		$\overline{1}$	$\overline{00}$	$\overline{1}$	$\overline{11}$			$\overline{0-}$	$\overline{0-}$		
	3	$K_n +$	$\overline{1}$	$\overline{00}$	$\overline{1}$	$\overline{11}$	i	$\overline{1}$	$\overline{101}$	$\overline{1}$	$\overline{0-}$	$\overline{0-}$
	4		$\overline{1}$	$\overline{00}$	o	$\overline{1}$	$\overline{11}$	i	$\overline{1}$	$\overline{101}$	$\overline{1}$	$\overline{0-}$
N/S	1		$\overline{0}$		$\overline{0}$							
	2		$\overline{0}$	$\overline{00}$	$\overline{0}$	$\overline{11}$			$\overline{1-}$	$\overline{1-}$		
	3	$K_n +$	$\overline{0}$	$\overline{00}$	$\overline{0}$	$\overline{11}$	i	$\overline{0}$	$\overline{101}$	$\overline{0}$	$\overline{1-}$	$\overline{1-}$
	4		$\overline{0}$	$\overline{00}$	o	$\overline{0}$	$\overline{11}$	i	$\overline{0}$	$\overline{101}$	$\overline{0}$	$\overline{1-}$

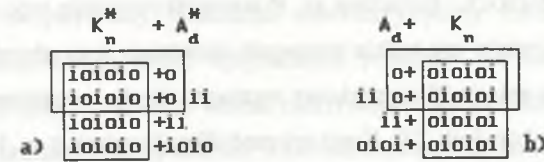
Przeprojektowanie na drodze odłączania ciągów d komórek wymaga zastosowania podobnej procedury postępowania. Sposób dodawania (odejmowanie) d-komórkowego modułu A_d reprezentującego d komórek 1 i 0 do (od) lewej strony modułu K_n wyjaśnia następujący oczywisty wniosek:

Wniosek 3.4 [Hławi90d]

Konkatenację $C = A_d + K_n$, związaną z pierwotnym wielomianem $p(x)$, można modelować za pomocą, związanej z identycznym wielomianem, odwrotnej konkatenacji $C^* = K_n^* + A_d^*$, w której ciąg A_d^* dodawany jest z prawej strony rejestru opisanego ciągiem K_n^* .

Wniosek ten wyjaśniono na rys. 3.8, na którym do lewej strony modułu oioioi dodawane są komórki 1 oraz 0.

Przeprojektowanie konkatenacji złożonych z wielu modułów, np. modułów ze zbiorów $A_{4,5}$ oraz $A_{4,5}^*$ również realizowane jest w podobny sposób. Należy jednak pamiętać, że każda pierwotna konkatenacja $C_{4,5}$ złożona z modułów ze zbiorów $A_{4,5}$ powiększona z prawej strony o jedną komórkę "o" lub "i" staje się konkatenacją niepierwotną. Natomiast pierwotna konkatenacja $C_{4,5}^*$ złożona z modułów ze zbiorów $A_{4,5}^*$ powiększona z prawej strony o jedną komórkę "o" lub "i" może okazać się konkatenacją pierwotną.



Rys. 3.8. Przeprojektowanie rejestru oioioi za pomocą dołączania komórek 1 lub 0 z jego lewej strony: określenie A_d^* (a), określenie A_d (b)
 Fig. 3.8. Redesigning of register oioioi by means of adding cells 1 and 0 to its left side: with the designation A_d^* (a), A_d (b)

Inne sposoby przeprojektowania konkatenacji złożonych z wielokomórkowych modułów wyjaśnia przykład.

Przykład 3.26

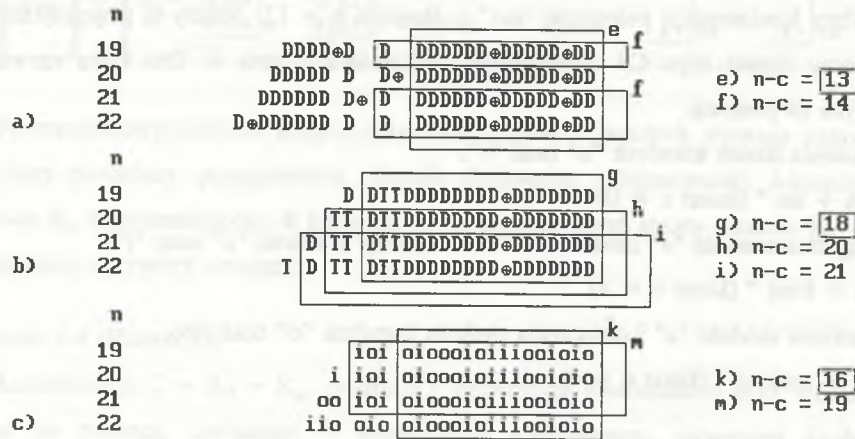
Załóżmy konkatenację pierwotną "aaa" o długości $n = 12$. Należy ją przeprojektować w pierwotny rejestr typu CA zawierający 15 komórek 1 oraz 0. Oto kilka rozwiązań uzyskanych za pomocą:

- a) dołączenia trzech komórek "o" oraz "i";
 "aaa + iio" (koszt $c = 0$)
- b) zastąpienia modułu "a" modułem "b" i dołączenia komórek "o" oraz "i";
 "aa + bioo" (koszt $c = 4$)
- c) odłączenia modułu "a" i dołączenia siedmiu komórek "o" oraz "i";
 "aa + ooiiooi" (koszt $c = 4$)

3.4.2. Łatwo modyfikowalne rejestry o strukturach IEDT, IED, EEDT oraz EED

Ze względu na kosztowny nadmiar bramek XOR w rejestrach typu CA interesujące jest zbadanie podatności na przeprojektowanie tańszych rejestrów o strukturach IED, IEDT, EED oraz EEDT. Przykład podany na rys. 3.9. pozwoli nam porównać podatność na przeprojektowanie trzech różnych zbiorów pierwotnych rejestrów o strukturach odpowiednio IED (rys. 3.9a), IEDT (rys. 3.9b) oraz CADT (rys. 3.9c). Każdy z tych zbiorów zawiera cztery rejestry odpowiednio 19-, 20-, 21- i 22-komórkowe. Litera T traktowana jest jako symbol zastępczy komórki ($\oplus'D$). Analizę podatności zaczniemy od rejestrów ze strukturą IEDT. Załóżmy, że pierwotny projekt zawierał 19-komórkowy rejestr liniowy DDTTDDDDDDDD \oplus DDDDDDDD. Pozostałe rejestry przedstawione na rys. 3.9b są tak dobrane, że mają wspólny z tym rejestrem rdzeń, którym jest ciąg 18 komórek

DTTDDDDDDDD \oplus DDDDDDDD (prostokąt g). Przeprojektowywanie tego rejestru w jakikolwiek z pozostałych rejestrów nie będzie wymagało naruszenia tego rdzenia. Na przykład jego przeprojektowanie w rejestr 20-komórkowy wymaga odjęcia lewostronnej komórki D, a następnie dodania ciągu komórek TT. Koszt tej modyfikacji wynosi $c = 1$. Jeżeli zmniejszymy licznosc zbioru rejestrów, które przewidujemy, że będą efektem końcowym przeprojektowywania, to wówczas można wydłużyć rdzeń tych rejestrów. Na przykład w przypadku zbioru rejestrów zawierających 20, 21 i 22 komórki ich rdzeniem (prostokąt h) jest ciąg TTTDDDDDDDDDD \oplus DDDDDDDD dłuższy od poprzedniego rdzenia o dwie komórki ($n - c = 20$). Koszt przeprojektowywania realizowanego w granicach tego zbioru rejestrów wynosi $c = 0$. Rejestry te są więc całkowicie podatne na przeprojektowywanie. Przykładem zbioru rejestrów o najdłuższym rdzeniu jest zbiór zawarty w prostokącie i.



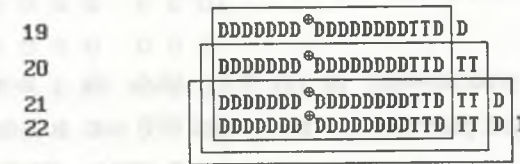
Rys. 3.9. Zbiory rejestrów o strukturach IED (a), IEDT (b) oraz CADT (c) umożliwiającym ich łatwe przeprojektowywanie
 Fig. 3.9. A set of registers with structures IED (a), IEDT (b) and CADT (c) enabling easy redesign

Cechą szczególną omawianego przykładu jest nienaruszalność nie tylko rdzenia, ale także prawej strony rejestrów liniowych przy ich przeprojektowywaniu. Wynika to z faktu zastosowania rejestrów o strukturach IEDT związanych z pierwotnym wielomianem charakterystycznym typu $p(x) = 1 + x^k(1 + x)^f[1 + x^a]$, w którym "a" jest stałe ($a = 7$). Taki wielomian charakterystyczny związany jest ze zbiorem rejestrów powstałych przez przestawianie pozycji komórek D oraz T w rejestrze $D^k T^f \oplus D^a$. Duża licznosc tego zbioru umożliwia ustawianie komórek ($\oplus'D$) (T) w dowolnym miejscu lewej strony tych rejestrów

bez wpływu na pierwotny wielomian charakterystyczny związany z tymi rejestrami. W efekcie umożliwia to dobór optymalnych rejestrów dla celów przeprojektowywania (prostokąty g,h oraz i). Niestety wzrost efektywności przeprojektowywania okupiony został pewnym zwiększeniem liczby bramek XOR niezbędnych do realizacji sprzężenia liniowego w komórkach typu $\oplus'D$.

Jeżeli w oparciu o wielomian $p(x) = 1 + x^k(1 + x)^f[1 + x^a]$ zostanie zaprojektowany zbiór rejestrów o strukturach EEDT, wówczas podczas ich modyfikowania nienaruszalny będzie rdzeń oraz ich lewa strona. Ten zbiór rejestrów przedstawiony jest na rys 3.10. Czas propagacji jest taki sam jak w przypadku zbioru rejestrów IEDT.

Zauważmy, że zmieniając wartość wykładnika "a", można uzyskać inne zbiory rejestrów o strukturach IEDT lub EEDT umożliwiające równie efektywne przeprojektowywanie.



Rys. 3.10. Zbiór rejestrów liniowych o strukturach EEDT umożliwiającym ich łatwe przeprojektowywanie
 Fig. 3.10. A set of registers with structures EEDT enabling easy redesign

Przykład zbioru rejestrów o strukturach IED przedstawiony na rys. 3.9a jest najmniej podatny na przeprojektowywanie. Najmniejszy koszt przeprojektowywania, jaki w ramach tego zbioru można uzyskać, wynosi $c = 6$. Jeżeli wielomiany charakterystyczne tych rejestrów będą podstawą zaprojektowania rejestrów o strukturach EED, wówczas podobnie jak w przypadku zbioru rejestrów o strukturach EEDT przeprojektowywanie można realizować za pomocą manipulowania prawostronnymi komórkami tych rejestrów. Również w tym przypadku najmniejszy koszt przeprojektowywania wyniesie $c = 6$. Dodatkową wadą zbioru rejestrów o strukturach EED jest zwiększony czas propogacji sygnału wywołany kaskadą trzech bramek XOR w zewnętrznym sprzężeniu liniowym.

Zastosowanie zbioru rejestrów typu CA przedstawionych na rys 3.9c umożliwia uzyskanie kosztu przeprojektowywania $c = 3$ (prostokąt k). Wyrzucając z tego zbioru rejestr 22-komórkowy, można koszt przeprojektowywania znacząco zredukować. Odwracając te rejestry, można uzyskać możliwość przeprojektowywania ich prawej strony.

Porównując koszt modyfikowania przykładowych grup rejestrów, najłatwiej modyfikowalnymi okazały się rejestry o strukturach IEDT oraz EEDT. Ich istotną zaletą w porównaniu z rejestrami typu CA jest możliwość znalezienia takiego n-komórkowego rejestru o strukturze IEDT (EEDT), do (od) którego można kolejno dołączać (odłączać) pojedyncze komórki D lub T i po każdym takim kroku otrzymywać modyfikację rejestru zawsze związaną z wielomianem pierwotnym.

Niestety przeprowadzona analiza porównawcza nie stanowi obiektywnego poglądu na sprawę dyskutowanej podatności. Przyczyną jest intuicyjny dobór analizowanych przykładów. Problem opracowania teorii umożliwiającej obiektywny wybór struktury rejestrów łatwo modyfikowalnych jest w dalszym ciągu problemem czekającym na rozwiązanie.

3.5. Projektowanie rejestrów COPMISR

Rejestr COPMISR, przedstawiony na rys. 2.8, składa się z n-bitowego rejestru EED-MISR, k dodatkowych przerzutników parzystości PFF oraz dodatkowego liniowego układu kombinacyjnego LCC. W oparciu o prace [HłaGS96a, HłaGS96b, Hławi96d, HłaGS97] przedstawiono w tym rozdziale technikę projektowania rejestru COPMISR dla dwóch wybranych rozdzielnych kodów liniowych.

3.5.1. Rejestr COPMISR dla kodu z bitami parzystości grupowej

Obecnie wyjaśnimy, jak może być zaprojektowany rejestr COPMISR dla kodu z bitami parzystości grupowej zawierającego sześć bitów informacyjnych y_1, \dots, y_6 oraz trzy bity parzystości y_7', y_8', y_9' . Kod ten zastosowano w [Toub93] do zaprojektowania układu samosprawdzalnego (ang. self-checking). Macierz generująca G dla tego kodu jest następująca:

$$G = [I_6, P] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Natomiast bity parzystości związane z przykładowym kodem są określone przez następujące wyrażenia:

$$\begin{aligned} y_7'(t) &= y_1(t) + y_3(t) \\ y_8'(t) &= y_2(t) + y_5(t) + y_6(t) \\ y_9'(t) &= y_4(t) \end{aligned}$$

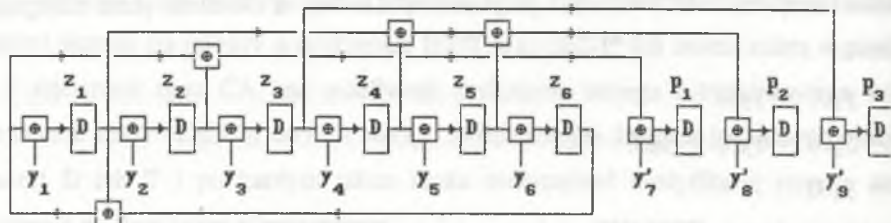
Do projektu rejestru COPMISR wybrano rejestr EED-MISR związany z wielomianem $p(x) = 1 + x^5 + x^6$. W związku z tym współczynniki macierzy C (2.18) związane z tym wielomianem będą następujące: $a_0 = a_1 = a_6 = 1$ oraz $a_2 = a_3 = a_4 = a_5 = 0$. Wstawiając je wraz z odpowiednimi bitami parzystości z macierzy G do równań 2.19, otrzymujemy wartości współczynników c_{ij} oraz $c_{i,n}$ dla $i = 1, \dots, k$; $j = 1, \dots, n-1$. Zakładając $b_{r,s} = 0$, otrzymujemy ostatecznie następującą macierz $A = C$:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Na podstawie tej macierzy otrzymujemy następujące wyrażenia określające funkcje wzbu-
dzeń przerzutników rejestru COPMISR:

$$\begin{aligned} z_1(t+1) &= [z_1(t) + z_6(t)] + y_1(t) \\ z_i(t+1) &= z_{i-1}(t) + y_i(t) \text{ dla } i = 2, \dots, 6 \\ p_1(t+1) &= [z_1(t) + z_6(t)] + z_2(t) + y_7'(t) \\ p_2(t+1) &= z_1(t) + z_4(t) + z_5(t) + y_8'(t) \\ p_3(t+1) &= z_3(t) + y_9'(t) \end{aligned}$$

Bramka XOR określona wyrażeniem $[z_1(t) + z_6(t)]$ może być równocześnie wykorzystana do realizacji sprzężenia liniowego rejestru EED-MISR oraz funkcji $p_1(t+1)$ układu LCC. Schemat przykładowego rejestru COPMISR przedstawia rys. 3.11.



Rys. 3.11. Rejestr EED-COPMISR dla kodu z bitami parzystości grupowej
Fig. 3.11. Register EED-COPMISR for parity group code

W kolejnym przykładzie do realizacji projektu rejestru COPMISR zostanie wykorzystany rejestr IEDT-MISR oraz struktura układu LCC opisana macierzą $A = C$ przedstawioną w 2.22 oraz 2.23. Przykład ten zostanie zrealizowany dla kodu z bitami parzystości grupowej zawierającego trzy bity informacyjne y_1, y_2, y_3 oraz trzy bity parzystości y_4', y_5', y_6' . Macierz generująca G dla tego kodu przedstawiona jest poniżej:

$$G = [I_3, P] = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Jej podmacierz P zawiera następujące współczynniki parzystości $p_{j,i}$ różne od zera $P_{11} = P_{31} = P_{22} = P_{32} = P_{23} = 1$. Zakładamy rejestr IEDT-MISR typu **DTD** związany z wielomianem pierwotnym $p(x) = x^3 + x^2 + 1 = 1 + x^2(1 + x)$. Rejestr ten posiada następujące współczynniki $a_0 = 1, a_1 = a_2 = 0, k_0 = k_2 = 0$ oraz $k_1 = 1$. Po podstawieniu ich wraz z podanymi powyżej współczynnikami $p_{j,i}$ do wyrażeń 2.23 otrzymujemy wartości współczynników $c_{i,j}$ oraz $c_{i,n}$ macierzy C (2.22) dla $i = 1, 2, 3$ oraz $j = 1, 2, 3$. Przy założeniu współczynników $b_{r,s} = 0$ otrzymujemy ostatecznie następującą macierz C

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

oraz następujące funkcje wzbudzeń projektowanego rejestru COPMISR:

$$z_1(t+1) = z_3(t) + y_1(t)$$

$$z_2(t+1) = [z_1(t) + z_2(t)] + y_2(t)$$

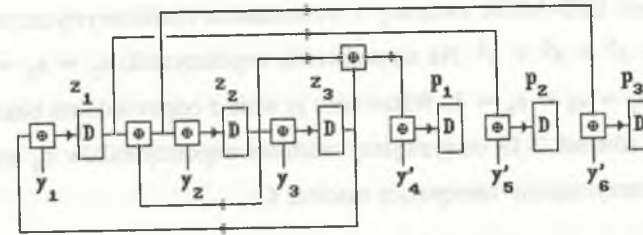
$$z_3(t+1) = z_2(t) + y_3(t)$$

$$p_1(t+1) = z_2(t) + z_3(t) + y_4'(t)$$

$$p_2(t+1) = z_1(t) + y_5'(t)$$

$$p_3(t+1) = [z_1(t) + z_2(t)] + y_6'(t)$$

Schemat rejestru COPMISR zrealizowany na podstawie podanych funkcji wzbudzeń przedstawia rys. 3.12.



Rys. 3.12. Rejestr IEDT-COPMISR dla kodu z bitami parzystości grupowej
Fig. 3.12. Register IEDT-COPMISR for parity group code

3.5.2. Rejestr COPMISR dla kodu SEC-DED Hsiao

Idea projektowania rejestru COPMISR dla zmodyfikowanego kodu Hamminga (kod Hsiao) została podana już wcześniej w [HłGS96a, HłGS96b, Hławi96d]. Tutaj zostanie wyjaśniona za pomocą przykładu kodu SEC-DED (13,5) podanego w [KatN91] dla układu scalonego SN54/74LS636. Słowo kodowe

$y = y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8 y_9' y_{10}' y_{11}' y_{12}' y_{13}'$ w tym kodzie (8+5,5) otrzymuje się ze zbioru bitów informacyjnych $y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8$ za pomocą mnożenia $y = (y_1 y_2 y_3 y_4 y_5 y_6 y_7 y_8) G$, gdzie:

$$G = [I_8, P] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Bity parzystości określają wyrażenia

$$y_9'(t) = y_1(t) + y_2(t) + y_4(t) + y_5(t)$$

$$y_{10}'(t) = y_1(t) + y_3(t) + y_4(t) + y_6(t) + y_7(t)$$

$$y_{11}'(t) = y_2(t) + y_3(t) + y_5(t) + y_6(t) + y_8(t)$$

$$y_{12}'(t) = y_1(t) + y_2(t) + y_3(t) + y_7(t) + y_8(t)$$

$$y_{13}'(t) = y_4(t) + y_5(t) + y_6(t) + y_7(t) + y_8(t)$$

Załóżmy rejestr EED-MISR związany z wielomianem charakterystycznym

$p(x) = 1 + x + x^5 + x^6 + x^8$. Na tej podstawie współczynniki $a_2 = a_3 = a_4 = a_7 = 0$ oraz $a_0 = a_1 = a_5 = a_6 = a_8 = 1$. Wstawiając je wraz z odpowiednimi bitami parzystości z macierzy G do równań 2.19 otrzymujemy wartości współczynników c_{ij} oraz $c_{1,n}$. Przyjmując $b_{r,s} = 0$, otrzymujemy następującą macierz C:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Macierz ta określa następujące funkcje wzbudzeń przerzutników przykładowego rejestru COPMISR.

$$z_1(t+1) = z_1(t) + [z_5(t) + z_6(t)] + z_8(t) + y_1(t)$$

$$z_i(t+1) = z_{i-1}(t) + y_i(t) \text{ dla } i = 2, \dots, 8$$

$$p_1(t+1) = z_3(t) + z_8(t) + [z_5(t) + z_6(t)] + z_4(t) + y_9'(t)$$

$$p_2(t+1) = z_1(t) + z_2(t) + z_3(t) + z_8(t) + y_{10}'(t)$$

$$p_3(t+1) = z_1(t) + z_2(t) + z_4(t) + z_7(t) + z_5(t) + y_{11}'(t)$$

$$p_4(t+1) = z_2(t) + z_5(t) + z_7(t) + z_8(t) + y_{12}'(t)$$

$$p_5(t+1) = z_4(t) + z_7(t) + [z_5(t) + z_6(t)] + z_3(t) + y_{13}'(t)$$

Bramkę XOR z funkcją $[z_5(t) \oplus z_6(t)]$ można wykorzystać do minimalizacji schematu rejestru COPMISR. Może ona realizować nie tylko funkcję sprzężenia liniowego rejestru EED-MISR, ale także funkcje $p_1(t+1)$ oraz $p_5(t+1)$ w układzie LCC.

4. KOMPAKCJA LINIOWA I JEJ PROBLEMY

Rejestr liniowy związany z wielomianem $p(x)$ jest od dawna stosowany w komunikacji [Socha66, Buckl75, Ralla78, Budko79, Seidl83] i sterownikach pamięci magnetycznych [Ralla78, Budko79, Gutma79] do detekcji błędów, jakie mogą powstać podczas przesyłania (zapisywania, przechowywania lub odczytu z pamięci) zakodowanej w postaci binarnego ciągu K binarnej informacji źródłowej Z . W zastosowaniach tych generowany jest systematyczny kod liniowy ilorazowy [Seidl83], dla którego każdy ciąg K ze zbioru ciągów kodowych jest określony na podstawie źródłowych ciągów binarnych Z w sposób, który przy wykorzystaniu działań na wielomianach $k(x)$, $z(x)$ ilustrujących ciągi K , Z przedstawia wyrażenie $k(x) = x^n z(x) \oplus R[x^n z(x), p(x)]$. W wyrażeniu tym $p(x)$ jest dowolnym wielomianem stopnia n , zawierającym wolny wyraz. Nazywany jest wielomianem generującym kod. Badanie, czy odebrany ciąg K' jest ciągiem kodowym K , polega na określeniu reszty z podzielenia $k'(x)$ przez $p(x)$ i sprawdzeniu, czy jest ona równa zero [Budko79]. Dzielenie to jest realizowane za pomocą jednowęściowego rejestru liniowego o sprzężeniu wewnętrznym również związanym z wielomianem charakterystycznym $p(x)$. Stan tego rejestru po podzieleniu wielomianu $k'(x)$ jest resztą $R[k'(x), p(x)]$. Wynik różny od zera sygnalizuje obecność ciągu błędów $E = K \oplus K'$. Innymi słowy $R[k'(x), p(x)] = R[\{k(x) \oplus e(x)\}, p(x)] = R[e(x), p(x)]$. Wynik równy zero oznacza brak błędów albo obecność błędów niewykrywalnych $e(x) = p(x)q(x)$ będących wielokrotnością wielomianu generującego kod.

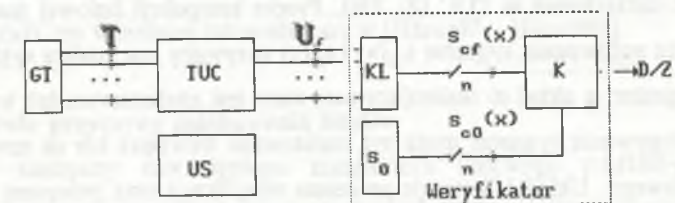
4.1. Kompakcja liniowa oraz weryfikacja sygnatur

Szeregowa kompakcja liniowa zastosowana w zewnętrznej analizie sygnowej uszkodzeń oraz równoległa lub szeregowo kompakcja liniowa stosowana w testowaniu wewnątrz-

układowym również polega na dzieleniu wielomianów [Smith80, Hassa82, HassL83, Hławi84c, Hławi86c, Hławi88a, Hławi92a, Hławi92b, Hławi93b, Hławi96b]. Istnieją jednak pewne istotne różnice. Wyjaśnimy to przy użyciu ogólnego schematu testowania przedstawionego na rys. 4.1. Schemat ten jednocześnie ilustruje w dużym uproszczeniu tester wewnątrzukładowy. Sekwencja testów \mathbf{T} o długości m generowana przez generator testów GT pobudza testowany układ cyfrowy TUC, na którego n wyjściach w odpowiedzi pojawia się $(m \times n)$ -bitowa macierz odpowiedzi \mathbf{U}_f , która dla $f = 0$ oznaczać będzie macierz odpowiedzi sprawnego UC, a dla $f \neq 0$, gdzie $f = 1, 2, \dots$, oznaczać będzie macierz uszkodzonego UC. Oznaczając przez U_r ciąg danych diagnostycznych o długości m na r -tym wyjściu sprawnego UC oraz przez U_{fr} ciąg na r -tym wyjściu testowanego UC, wspomniane macierze odpowiedzi można zilustrować w następujący sposób:

$$\mathbf{U}_0 = \begin{bmatrix} U_{00} \\ U_{01} \\ \vdots \\ U_{0r} \\ \vdots \\ U_{0n-1} \end{bmatrix} \quad \mathbf{U}_f = \begin{bmatrix} U_{f0} \\ U_{f1} \\ \vdots \\ U_{fr} \\ \vdots \\ U_{fn-1} \end{bmatrix} \quad \mathbf{U}_0 \oplus \mathbf{U}_f = \mathbf{E}_f = \begin{bmatrix} E_{f0} \\ E_{f1} \\ \vdots \\ E_{fr} \\ \vdots \\ E_{fn-1} \end{bmatrix}$$

Macierz $\mathbf{E}_f \neq 0$ jest macierzą błędów wymuszonych uszkodzeniem f . Macierz $\mathbf{E}_0 = 0$. Testowanie w klasycznym tego słowa pojęciu polega na porównywaniu obu macierzy \mathbf{U}_0 oraz \mathbf{U}_f i na tej podstawie stwierdzeniu, czy TUC jest uszkodzony, czy też nie.



Rys. 4.1. Ogólny schemat testowania układu cyfrowego
Fig. 4.1. General scheme of digital circuit testing

Przy użyciu równoległego kompaktora liniowego KL w postaci rejestru liniowego c -MISR- $p(x)$ macierz \mathbf{U}_0 sprawnego UC, odwzorowana w postaci reprezentowaną wielomianem $w_{c0}(x)$, jest dzielona przez wielomian charakterystyczny $p(x)$ w celu uzyskania wzor-

cowego n -bitowego stanu końcowego, czyli sygnatury $s_{c0}(x)$. Zakładamy, że rejestr c -MISR- $p(x)$ przed każdym procesem kompaktacji jest wstępnie ustawiony w ten sam stan $g_c(x) = \delta(h_c(x))$, który niekoniecznie musi być równy zero. Ilustruje to dzielenie (2.1) $[w_{c0}(x) + x^m h_c(x)]/p_c(x) = q_c(x) + r_{c0}(x)/p_c(x)$, w którym bity sygnatury $s_{c0}(x)$ są uwikłane w reszcie $r_{c0}(x)$. Sygnatura wzorcowa $s_{c0}(x)$ jest cechą charakterystyczną macierzy \mathbf{U}_0 (wielomian $w_{c0}(x)$) przy założonym stanie początkowym $g_c(x) = \delta(h_c(x))$ rejestru c -MISR. Przedstawiony na rys. 2.7 schemat zastępczy rejestru c -MISR- $p(x)$ umożliwi abstrakcyjne rozdzielenie procesu równoległej kompaktacji liniowej na dwa etapy. Pierwszy realizowany w przetworniku liniowym P1c ma charakter kompaktacji przestrzennej (space compaction), w której $(m \times n)$ -bitowa macierz \mathbf{U}_0 przekształcana jest w abstrakcyjny ciąg W_{c0} zawierający $m+n-1$ bitów. Drugi realizowany przez jednowyjściowy rejestr IED-SISR i przetwornik P2c ma charakter kompaktacji czasowej (time compaction), w której ciąg W_{c0} odwzorowywany jest w n -bitową sygnaturę.

Jeżeli TUC posiada uszkodzenie, wówczas na jego wyjściu może pojawić się macierz $\mathbf{U}_f = \mathbf{U}_0 \oplus \mathbf{E}_f$ przekształcona na wyjściu abstrakcyjnego przetwornika P1c w postać $W_{cf} = W_{c0} \oplus E_{cf}$ reprezentowaną wielomianem $w_{cf}(x) = w_{c0}(x) + e_{cf}(x)$, w którym $e_{cf}(x)$ ilustruje skompresowaną macierz błędów \mathbf{E}_f . Ostatecznym efektem kompaktacji macierzy \mathbf{U}_f w rejestrze c -MISR- $p(x)$ ze stanem początkowym $g_c(x)$ jest sygnatura $s_{cf}(x)$, której bity są uwikłane w reszcie $r_{cf}(x)$. Porównanie sygnatur $s_{c0}(x)$ i $s_{cf}(x)$ jest realizowane w komparatorze K. Efektem tego porównania może być sygnatura błędów $s_{ce}(x) = s_{c0}(x) + s_{cf}(x)$. Wynik $s_{ce}(x) = 0$ oznacza, że TUC jest sprawny (D; Dobry). Wynik różny od zera sygnalizuje obecność uszkodzenia w TUC (Z; Zły). Proces kompaktacji liniowej macierzy odpowiedzi TUC oraz porównania sygnatur $s_{c0}(x)$ i $s_{cf}(x)$ nazywany jest analizą sygnaturową lub weryfikacją sygnatur, a układ to realizujący nazywany jest analizatorem lub weryfikatorem sygnatur. Porównywanie sygnatur może być realizowane wewnątrz lub na zewnątrz testera wewnątrzukładowego. Układ US steruje procesem weryfikacji oraz procesem generowania testów.

Zakładając jednowyjściowy TUC ($n = 1$), można w miejsce równoległego kompaktora liniowego zastosować szeregowy rejestr liniowy c -SISR- $p(x)$. Wówczas proces kompaktacji liniowej można opisać za pomocą dzielenia (2.5)

$[u_0(x) + x^m h_c(x)]/p_c(x) = q_c(x) + r_c(x)/p_c(x)$, w którym wielomian $u_0(x)$, reprezentujący ciąg U_0 sprawnego UC, w odróżnieniu od wielomianu $k(x)$ opisującego ciąg kodowy K,

nie musi być kodem ilorazowym. Jego postać jest zależna od funkcji TUC, zbioru testów i kolejności, w jakiej te testy są podawane na wejścia TUC. Podobnie jest w przypadku wielomianu $w_{c0}(x)$ odwzorowującego macierz \mathbf{U}_0 . Proces weryfikacji odpowiedzi TUC w przypadku szeregowego kompaktora liniowego jest identyczny z opisanym poprzednio.

W celu skrócenia dalszych rozważań związanych z szeregowymi kompaktorami liniowymi większość dyskusji prowadzonych dla równoległych kompaktorów liniowych c -MISR- $p(x)$ będzie traktowana jako ważna także dla rejestrów c -SISR- $p(x)$.

4.2. Skuteczność wykrywania błędów za pomocą analizy sygnaturowej

Sygnatura $s_c(x)$ nie jest resztą $r_c(x)$ w wyrażeniu 2.1. Pomimo to warunki wykrycia za pomocą rejestru c -MISR- $p(x)$ macierzy błędów $\mathbf{E}_f \neq 0$ można określać w oparciu o resztę, którą na podstawie wniosków $r_c(x) \neq 0 \Leftrightarrow s_c(x) \neq 0$ oraz $r_c(x) = 0 \Leftrightarrow s_c(x) = 0$ (rozdział 2.3) można traktować jak sygnaturę, a więc także jako cechę charakterystyczną macierzy \mathbf{U}_0 . Na tej podstawie

$R\{w_{cf}(x) + x^m h_c(x)\}, p(x) = R\{w_{c0}(x) + x^m h_c(x) + e_{cf}(x)\}, p(x) =$
 $= R\{w_{c0}(x) + x^m h_c(x)\}, p(x) + R\{e_{cf}(x)\}, p(x)$. Wykrycie macierzy błędów \mathbf{E}_f nastąpi tylko wówczas, gdy $R\{e_{cf}(x)\}, p(x) \neq 0$. Wynik równy zero oznacza maskowanie błędów albo ich brak. Detekcja błędów jest niezależna od wartości stanu początkowego $g_c(x) = \delta(h_c(x))$, co wcześniej udowodniono w [Hławi87c, Hławi88a].

4.2.1. Dwie przyczyny maskowania błędów

Schemat zastępczy równoległego kompaktora liniowego c -MISR- $p(x)$ (rys. 2.7) umożliwia strukturalne wydorebnienie dwóch różnych przyczyn maskowania błędów [Hławi87c, Hławi88a]. W rejestrze IED-SISR tego schematu może wystąpić efekt podzielności wielomianu $e_{cf}(x)$ reprezentującego ciąg E_{cf} przez wielomian $p(x)$, natomiast w przetworniku liniowym P1c następuje zjawisko wzajemnego kasowania się błędów E_{fr} , które w takim przypadku odwzorowywane są w ciąg $E_{cf} = 0$ reprezentowany wielomianem $e_{cf}(x) = 0$.

Jeżeli wielomian $e_{cf}(x) = p(x)q_c(x)$, wówczas jest on niepożądanym kodem ilorazowym dzielonym bez reszty przez wielomian $p(x)$ w rejestrze c-MISR- $p(x)$. W takim przypadku $s_{cf}(x) = s_{c0}(x)$, co oznacza, że błędy są maskowane. Wyjaśnia to przykład.

Przykład 4.1

Założmy pięciowyjściowy uszkodzony TUC i wzięty z przykładu 2.3 równoległy kompaktor liniowy w postaci rejestru $\mathbf{DD}(\oplus' \mathbf{D})_{\oplus} \mathbf{DD} \equiv p(x) = x^5 + x^4 + x^3 + x^2 + 1$.

Założmy także następującą, wymuszoną testami i uszkodzeniem, macierz błędów

$$E_{f0} = \langle 0100000 \rangle, E_{f1} = \langle 1000000 \rangle, E_{f2} = \langle 1000000 \rangle, E_{f3} = \langle 0000000 \rangle,$$

$$E_{f4} = \langle 1000000 \rangle.$$

Wielomiany $p_{IEDTf}(x)$ charakteryzujące wejścia przykładowego rejestru IEDT-MISR zilustrowane są w drugim wierszu tab. 2.2. W oparciu o tę tabelę i podaną macierz błędów otrzymujemy następujący wielomian błędów

$$e_{IEDTf}(x) = e_{f0}(x) + e_{f1}(x)x + e_{f2}(x)x^2 + e_{f3}(x)(x^2 + x^3) + e_{f4}(x)(x^3 + x^4) = \\ = x^5 + x^6x + x^6x^2 + 0(x^2 + x^3) + x^6(x^3 + x^4) = x^5(x^5 + x^4 + x^3 + x^2 + 1).$$

Wielomian ten jest podzielny przez wielomian charakterystyczny rejestru $\mathbf{DD}(\oplus' \mathbf{D})_{\oplus} \mathbf{DD}$. W efekcie $R[e_{IEDTf}(x), p(x)] = 0$ jest powodem maskowania błędów na wyjściu TUC.

To charakterystyczne dla kompacji liniowej zjawisko podzielności wcześniej opisano dla rejestrów IED-SISR oraz EED-SISR w [Smith80], dla rejestru IED-MISR w [Hassa82, Hławi87c, Hławi88a], dla rejestru EED-MISR w [Hławi84c, Hławi86c, Hławi87c, Hławi88a], natomiast dla rejestrów BTM-MISR oraz TBM-MISR w [Hławi89a, Hławi89b, Hławi92b]. To zjawisko znane jest nie tylko w teorii kodów ilorazowych, ale także od dawna w teorii automatów liniowych.

Druga przyczyna maskowania błędów, nie opisana w teorii automatów liniowych i teorii kodów ilorazowych i nie występująca w szeregowych kompaktorach liniowych c-SISR- $p(x)$, wynika z równoczesnego wprowadzania różnych ciągów błędów na różne wejścia równoległego rejestru c-MISR- $p(x)$. Przy określonych wzorach błędów mogą się one wzajemnie skasować (mutual cancellation) [SaviB93]. Wynika to z wyrażenia $e_{cf}(x) = \sum_{r=0}^{n-1} e_{fr}(x)p_{cr}(x)$, które może być równe zero, pomimo że wielomiany $e_{fr}(x)$ reprezentujące ciągi błędów E_{fr} będą różne od zera.

Przykład 4.2

Założmy identyczny pięciowyjściowy uszkodzony TUC i równoległy kompaktor liniowy w postaci rejestru $\mathbf{DD}(\oplus' \mathbf{D})_{\oplus} \mathbf{DD} \equiv p(x) = x^5 + x^4 + x^3 + x^2 + 1$. Zakładamy jednakże inne uszkodzenie TUC, które pod wpływem testów wymusza następującą macierz błędów

$$E_{f0} = \langle 0000000 \rangle, E_{f1} = \langle 1000000 \rangle, E_{f2} = \langle 1000000 \rangle, E_{f3} = \langle 0000000 \rangle,$$

$$E_{f4} = \langle 0010000 \rangle.$$

W oparciu o tab. 2.2 i podaną macierz błędów otrzymujemy wielomian $e_{IEDTf}(x) = 0$. Wynika to z poniższych obliczeń

$$e_{IEDTf}(x) = e_{f0}(x) + e_{f1}(x)x + e_{f2}(x)x^2 + e_{f3}(x)(x^2 + x^3) + e_{f4}(x)(x^3 + x^4) = \\ = e_{f0}(x) + e_{f1}(x)x + [e_{f2}(x) + e_{f3}(x)]x^2 + [e_{f3}(x) + e_{f4}(x)]x^3 + e_{f4}(x)x^4 = \\ = 0 + x^6x + x^6x^2 + x^4x^3 + x^4x^4 = x^7 + x^8 + x^7 + x^8 = 0.$$

W efekcie również i w tym przypadku $R[e_{IEDTf}(x), p(x)] = 0$ sygnalizuje maskowanie błędów na wyjściu TUC.

Tę przyczynę maskowania po raz pierwszy zauważono w rejestrach IED-MISR w [Hassa82], a następnie zbadano dla rejestrów EED-MISR w [Hławi84c, Hławi86c]. Zauważmy, że rejestry c-MISR- $p(x)$ o różnych strukturach sprzężeń liniowych, chociaż związanych identycznym wielomianem $p(x)$, nie muszą maskować takiej samej macierzy błędów.

Przykład 4.3

Weźmy z przykładu 2.3 jeszcze dwa inne rejestry $\mathbf{10011}$ (MISR typu CA) oraz $\mathbf{D}^{\oplus} \mathbf{D}^{\oplus} \mathbf{D}^{\oplus} \mathbf{DD}$ (EED-MISR) związane takim samym wielomianem $p(x) = x^5 + x^4 + x^3 + x^2 + 1$, z jakim związany jest rejestr IEDT-MISR z przykładu 4.2. Sprawdźmy, czy macierz błędów podana w przykładzie 4.2 maskowana jest także przez rejestry $\mathbf{10011}$ oraz $\mathbf{D}^{\oplus} \mathbf{D}^{\oplus} \mathbf{D}^{\oplus} \mathbf{DD}$. W oparciu o tab. 2.2 otrzymujemy wielomiany $e_{CADTf}(x) \neq 0$ oraz $e_{EEDf}(x) \neq 0$. Wynika to z poniższych obliczeń.

$$e_{CADTf}(x) = e_{f0}(x) + e_{f1}(x)(1+x) + e_{f2}(x)(1+x+x^2) + e_{f3}(x)(1+x^2+x^3) + \\ + e_{f4}(x)x^4 = \\ = [e_{f0}(x) + e_{f1}(x) + e_{f2}(x) + e_{f3}(x)] + [e_{f1}(x) + e_{f2}(x)]x + [e_{f2}(x) + e_{f3}(x)]x^2 + \\ + e_{f3}(x)x^3 + e_{f4}(x)x^4 \neq 0$$

$$\begin{aligned}
e_{EEDf}(x) &= e_{f0}(x) + e_{f1}(x)(1+x) + e_{f2}(x)(1+x+x^2) + \\
&+ e_{f3}(x)(1+x+x^2+x^3) + e_{f4}(x)(x+x^2+x^3+x^4) = \\
&= [e_{f0}(x) + e_{f1}(x) + e_{f2}(x) + e_{f3}(x)] + [e_{f1}(x) + e_{f2}(x) + e_{f3}(x) + e_{f4}(x)]x + \\
&+ [e_{f2}(x) + e_{f3}(x) + e_{f4}(x)]x^2 + [e_{f3}(x) + e_{f4}(x)]x^3 + e_{f4}(x)x^4 \neq 0
\end{aligned}$$

Oba przykładowe rejestry nie maskują macierzy błędów maskowanej przez związany identycznym wielomianem $p(x)$ rejestr IED-MISR.

Na podstawie podanych przykładów oraz wyrażeń 2.1 i 2.5 oczywiste stają się następujące dwa wnioski [Hławi93b, Hławi96b]:

Wniosek 4.1

Zbiory macierzy błędów niewykrywanych przez rejestry c-MISR- $p(x)$ o różnych strukturach sprzężeń liniowych, chociaż związanych identycznym wielomianem $p(x)$, różnią się między sobą.

Wniosek 4.2

Zbiory sekwencji błędów niewykrywanych przez rejestry c-SISR- $p(x)$ o różnych strukturach sprzężeń liniowych związanych identycznym wielomianem $p(x)$ są identyczne.

Oba opisane w tym podrozdziale zjawiska maskowania błędów w literaturze anglosaskiej noszą nazwę "error aliasing" [CoIAR88, DamiO89a, GuptP88, Hławi89a, Hławi89b, IvanA87, IvanA88, WillD86, WillD89]. Ich efektem jest wymykanie się spod kontroli weryfikatora szeregu pobudzonych testem uszkodzeń. W konsekwencji prowadzi to do zmniejszenia się współczynnika pokrycia uszkodzeń testowanego układu cyfrowego. Oszacowanie skuteczności kompaktacji liniowej przy użyciu rejestrów c-SISR oraz c-MISR jest więc bardzo ważnym zagadnieniem.

4.2.2. Statyczne prawdopodobieństwo maskowania błędów

Teoretyczna skuteczność EF (ang. Effectiveness) kompaktacji jest zwykle definiowana jako prawdopodobieństwo P , że dwa różne ciągi wektorów odpowiedzi \mathbf{U} oraz \mathbf{U}' o długości m , uzyskane na wyjściu testowanego układu cyfrowego i zawierające wylosowane bity, są rozróżnialne w n -stopniowym rejestrze c-MISR- $p(x)$ przez ich sygnatury. Oznaczając przez f_k funkcję kompaktacji oraz przez s sygnaturę macierzy \mathbf{U} , mamy

$s = f_k(\mathbf{U})$. Wówczas teoretyczną skuteczność kompaktacji można wyrazić w następujący sposób:

$$EF = P [f_k(\mathbf{U}) \neq f_k(\mathbf{U}')] = 1 - P_{al}$$

gdzie: $P_{al} = P [f_k(\mathbf{U}) = f_k(\mathbf{U}')]$ jest prawdopodobieństwem maskowania różnicy $\mathbf{E} = \mathbf{U} \oplus \mathbf{U}'$ pomiędzy dwoma losowo wybranymi macierzami \mathbf{U} oraz \mathbf{U}' zawierającymi po $m \times n$ wylosowanych bitów. Macierz \mathbf{E} jest więc macierzą zawierającą $m \times n$ błędnych bitów, każdy o prawdopodobieństwie błędu $b = 0.5$. Jeżeli przez ε oznaczać się będzie zbiór wszystkich teoretycznie możliwych, niezależnych i jednakowo prawdopodobnych macierzy błędów \mathbf{E} , wówczas liczność tego zbioru $|\varepsilon|$ wraz z macierzą $\mathbf{E} = 0$ równa jest 2^{mn} . Ze względu na liniowy charakter kompaktacji macierze błędów \mathbf{E} ze zbioru ε są równomiernie odwzorowywane w 2^n różnych sygnatur. W efekcie zbiór ε można podzielić na 2^n równolicznych podzbiorów, w których różne macierze \mathbf{E} odwzorowywane są w identyczne sygnatury. Macierze błędów \mathbf{E} w każdym takim podzbiore są więc nierozróżnialne po kompaktacji przy porównaniu odpowiadających im sygnatur. Oznaczmy przez ε_m podzbiór macierzy $\mathbf{E} \neq 0$ odwzorowywanych w sygnaturę zerową charakterystyczną dla macierzy $\mathbf{E} = 0$. Wszystkie macierze podzbioru ε_m są maskowane przez kompaktor liniowy. Ponieważ założono, że każda macierz ze zbioru ε jest jednakowo prawdopodobna, dlatego teoretyczne prawdopodobieństwo maskowania P_{al} przez kompaktor liniowy macierzy błędów można określić jako

$$P_{al} = \frac{|\varepsilon_m|}{|\varepsilon|} \quad (4.1a)$$

Na tej podstawie teoretyczna skuteczność EF kompaktacji liniowej przy założeniu jednakowo prawdopodobnych macierzy błędów \mathbf{E} może być określona za pomocą wyrażenia:

$$EF = 1 - \frac{|\varepsilon_m|}{|\varepsilon|} \quad (4.1b)$$

Po podstawieniu odpowiednich wartości określających licznosci zbiorów ε_m oraz ε wyrażenie to dla m dążącego do nieskończoności zmierza do swojej granicznej postaci $1 - 2^{-n}$, co pozwoliło autorom pracy [Froh77] wyciągnąć generalny wniosek, że dla dużych wartości m teoretyczna skuteczność kompaktora EED-SISR w wykrywaniu błędów jest zależna jedynie od liczby stopni n szeregowego kompaktora. Wniosek ten potwierdzono dla rejestrów D-SISR w [David78, David80], dla rejestrów EED-SISR oraz IED-SISR w [Smith80] i uogólniono dla rejestrów IED-MISR w [Hassa82, HassL83], dla rejestrów

EED-MISR w [SridH82, Hławi84c, Hławi86c] i dla rejestrów D-MISR w [David85, David86]. W pracach [Hławi87c, Hławi88a], po zastosowaniu wyrażeń 4.1 w dwóch abstrakcyjnych krokach kompaktacji przestrzennej i czasowej schematu zastępczego (rys. 2.7) rejestrów IED-MISR oraz EED-MISR, wyprowadzono wzór określający teoretyczną skuteczność kompaktacji tych rejestrów i uwzględniający dwie różne przyczyny maskowania błędów. Wynikające z tego wzoru prawdopodobieństwo maskowania błędów również dąży do wartości 2^{-n} dla m dążącego do nieskończoności. Wniosek ten na podstawie schematu zastępczego (rys. 2.7) można rozszerzyć na pozostałe rejestry c-MISR [Hławi89a, Hławi89b, Hławi92b]. We wszystkich przytoczonych pracach nie określono zależności funkcji $P_{al}(m)$ od rodzaju charakterystycznego wielomianu $p(x)$ związane z kompaktorem liniowym.

Prawdopodobieństwo $P_{al} = 2^{-n}$ ze względu na swoją stałą wartość nazywane będzie dalej statycznym prawdopodobieństwem maskowania błędów. Nietrudno stwierdzić, że identyczne prawdopodobieństwo maskowania błędów będzie także posiadał SISR w postaci konwencjonalnego n -bitowego rejestru przesuwającego SR, nie zawierającego jakiegokolwiek sprzężenia [Smith80]. Podobnie identyczne prawdopodobieństwo maskowania błędów będzie charakteryzowało równoległy kompaktor w postaci rejestru MIShR. Jak udowodnić intuicyjnie oczywisty wniosek, że skuteczność analizy sygnaturowej w tych dwóch ostatnich przypadkach oraz w przypadku rejestrów o konfiguracji FSR jest mniejsza od rejestrów c-SISR- $p(x)$ oraz rejestrów c-MISR- $p(x)$ posiadających sprzężenia liniowe z bramkami XOR? Ten problem zostanie wyjaśniony w następnych podrozdziałach.

4.2.3. Czy wszystkie wielomiany charakterystyczne $p(x)$ są przydatne w analizie sygnaturowej?

Założenie występowania jednakowo prawdopodobnych błędów prowadzi do niedokładnej oceny skuteczności wykrywania uszkodzeń za pomocą analizy sygnaturowej. Realizowana jest ona w oderwaniu od struktury testowanego układu oraz w oderwaniu od modelu jego realnych uszkodzeń. Jej wynikiem jest jedynie statyczne prawdopodobieństwo niewykrycia błędów, które jest zupełnie niezależne od klasy uszkodzeń oraz identyczne dla każdego n -stopniowego rejestru c-SISR- $p(x)$ lub c-MISR- $p(x)$ niezależnie od rodzaju wielomianu charakterystycznego $p(x)$. W związku z tym model niezależnych od rzeczywistych uszkodzeń błędów zastąpiono w badaniach nad skutecznością modelem błędów zależnych od realnych uszkodzeń w skrócie nazywanych błędami zależnymi. Zastosowano dwa różne

podejścia: symulacyjne i analityczne. W pierwszym przypadku zastosowano komputer do symulacji analizy sygnaturowej przy założonej strukturze kompaktora liniowego, założonym modelu testowanego układu cyfrowego i jego uszkodzeń i symulowanym źródle testów [KoneM79, KoneM80, SridH82]. Taka symulacja komputerowa z jednej strony umożliwia wprawdzie dokładną ocenę skuteczności analizy sygnaturowej, ale z drugiej strony utrudnia wyciągnięcie ogólniejszych wniosków. Wady tej nie posiada podejście analityczne wykorzystujące pierścień wielomianów nad ciałem $GF(2)$. Pierwsze badania skuteczności analizy sygnaturowej taką metodą zrealizował Smith [Smith80]. W swojej pracy zajął się specyficznymi uszkodzeniami systemów mikroprocesorowych objawiających się paczkami błędów (ang. burst errors) w trakcie testowania typu "free run" lub "software driven". W celu określenia skuteczności analizy sygnaturowej w wykrywaniu takich błędów zależnych Smith badał podzielność przez wielomian charakterystyczny rejestru IED-SISR (EED-SISR) wielomianów opisujących hipotetyczne paczki błędów $E(b,t)$, występujących w wycinku m -bitowego ciągu złożonego z $b \leq n$ bitów i zawierających $t \leq b$ błędów. Załóżmy podobny model błędów, ale na r -tym wejściu rejestru c-MISR- $p(x)$. Tego typu błędy mogą wystąpić podczas testowania układu cyfrowego. Jednak najbardziej charakterystyczne są dla wielomodulowych układów scalonych ze strukturą testowania wewnątrzukładowego typu STUMPS (Self-Test Using MISR/Parallel SRSG) [BardM82, BardM83, BardM87, AbraB90, PilaK95], w których uszkodzenie jednego z modułów może wywołać ciąg błędów wprowadzanych do rejestru c-MISR na jednym tylko wejściu. Mamy więc wielomian błędów $e_r(x) = x^k b(x)$, gdzie $\deg b(x) < n$. Maskowanie takich paczek błędów zależy od wartości $R[x^k b(x) p_{cr}(x), p(x)]$. Zakładając równoległy kompaktor liniowy o charakterystycznym wielomianie ułomnym $p(x) = x^d g(x)$, gdzie $d \leq n$ oraz $\deg g(x) = n - d$ otrzymujemy $R[x^k b(x) p_{cr}(x), x^d g(x)] = 0$ wtedy, gdy $k \geq n$ oraz $g(x) = x^{n-d}$ lub $g(x) = b(x)p_{cr}(x)$.

Przykład 4.4

Weźmy rejestr MISR typu CA z następującym ciągiem komórek **01110101** i ułomnym wielomianem charakterystycznym $p_{CADT}(x) = x^6(x^2 + x + 1)$. Załóżmy na jego pierwszym wejściu, związanym z wielomianem $p_{CADT0}(x) = 1$, paczkę błędów $E(b,t) = \langle 0001110000000000000000 \rangle$, której odpowiada następujący wielomian

błędów $e_r(x) = x^{20} + x^{19} + x^{18} = x^{18}(x^2 + x + 1)$. Uszkodzenie objawiające się tym ciągiem błędów nie zostanie wykryte, ponieważ reszta

$$R[x^{18}(x^2 + x + 1), x^6(x^2 + x + 1)] = 0.$$

Na tej podstawie kompaktory równoległe typu MISHr oraz CADT-MISR- $x^d g(x)$ i IET-MISR- $x^d g(x)$ okazały się nieprzydatne w analizie sygnaturowej [Hławi92a, Hławi93a]. Z tego samego powodu nieprzydatne są także szeregowe kompaktory liniowe w postaci rejestru SR [Smith80], rejestrów CADT-SISR- $x^d g(x)$ [Hławi93a] i rejestrów IET-MISR- $x^d g(x)$ [Hławi92a]. Dla szeregowych kompaktorów liniowych można sformułować następujący wniosek [Hławi93b]:

Wniosek 4.3

Rejestry c-SISR- $p(x)$ związane ułomnym wielomianem $p(x) = x^d g(x)$ są nieprzydatne w analizie sygnaturowej pomimo, że statyczne prawdopodobieństwo maskowania błędów wprowadzanych na ich wejście równe jest 2^{-n} .

Oprócz tego wniosku można również w oparciu o przedstawione rozważania określić niektóre właściwości detekcyjne kompaktorów liniowych. Na przykład stwierdzono, że rejestry IED-SISR- $p(x)$ oraz EED-SISR- $p(x)$ o dowolnych wielomianach nieułomnych wykrywają wszystkie błędy grupowe o długości $b \leq n$ oraz wszystkie błędy pojedyncze [Smith80], natomiast rejestry o pierwotnych wielomianach $p(x)$ wykrywają ponadto wszystkie błędy podwójne w ciągu o długości $m \leq 2^n - 1$ [Smith80]. Wniosek ten przy wykorzystaniu pierścienia wielomianów nad ciałem GF(2) można uogólnić na każdy rejestr c-SISR- $p(x)$. W pracy [Hassa82] udowodniono, że wnioski te dotyczą także każdego r -tego wejścia rejestrów IED-MISR- $p(x)$. W pracach [Hławi84c, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi90a, Hławi92b] udowodniono ponadto, że wnioski te można przyjąć także dla rejestrów EED-MISR, TBD-MISR oraz BTM-MISR. Przy wykorzystaniu pierścienia wielomianów nad ciałem GF(2) można w prosty sposób udowodnić, że r -te wejście każdego rejestru c-MISR- $p(x)$ związanego z wielomianem $p(x)$ o podanych poprzednio cechach posiada takie właściwości detekcyjne.

Wartość reszty $R[x^k b(x) p_{cr}(x), p(x)] \neq 0$ w przypadku każdego nieułomnego wielomianu $p(x)$, nawet wtedy gdy są one redukowalne. Niestety nieułomne redukowalne wielomiany $p(x)$ powodują obniżenie teoretycznej skuteczności wykrywania błędów. Wyjaśnimy

to pozostając przy przykładzie testera wewnątrzukładowego typu STUMPS i przykładzie błędów zależnych występujących tylko na jednym wejściu rejestru c-MISR- $p(x)$. Innym przykładem może być też tester wewnątrzukładowy typu LOCST (LSSD On-Chip Self-Test) [LeBl84, AbraB90], w którym wszystkie wyjścia uszkodzonego TUC połączone są z wejściami ścieżki sterująco-obszernyjnej, której wyjście wprowadzono na jedno z wejść rejestru c-MISR (scan-based compaction [PilaK95]). W celu uogólnienia rozważań założmy jednak możliwość wystąpienia dowolnego błędu typu $e_r(x)$. Założmy także kompaktor c-MISR- $p(x)$ z nieułomnym, ale redukowalnym wielomianem $p(x) = d(x) g(x)$ oraz wielomianem $p_{cr}(x) = d(x)$, gdzie $\deg d(x) = d$. Wówczas na podstawie wyrażenia 2.1 mamy

$$e_r(x) p_{cr}(x) / p(x) = e_r(x) d(x) / d(x) g(x) = e_r(x) / g(x)$$

Efekt kompaktacji liniowej podobny jest w tym przypadku do efektu kompaktacji zrealizowanej przy użyciu rejestru związanego z wielomianem $p(x) = g(x)$ stopnia $n - d$, co zwiększa statyczne prawdopodobieństwo maskowania błędów do wartości $P_{al} = 2^{-(n-d)}$. W efekcie teoretyczna skuteczność analizy sygnaturowej takich zależnych błędów jest dużo mniejsza od wartości $EF = 1 - 2^{-n}$. Wpływ wielomianów $p_{cr}(x)$ na skuteczność analizy sygnaturowej wyjaśnia następujący przykład.

Przykład 4.5

Weźmy wielomian charakterystyczny:

$$p(x) = 1 + x + x^2 + x^3 + x^4 + x^6 + x^7 = (1 + x + x^2)(1 + x^3 + x^5),$$

który można przekształcić w następujące formy

$$1 + (x + x^3 + x^6)(1 + x) = p_{TBDa}(x) \text{ oraz } 1 + (x + x^2)(1 + x^2 + x^5) = p_{TBDb}(x).$$

Na podstawie tych wyrażeń zaprojektowano następujący zbiór rejestrów liniowych: $\mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D}$ (IED-MISR); $\mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D}$ (EED-MISR); $\mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D}$ (TBDa-MISR) oraz $\mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D} \oplus \mathbf{D}$ (TBDb-MISR). Dwa z przedstawionych rejestrów c-MISR posiadają wielomiany $p_{cr}(x)$, które są podzielne przez wielomian charakterystyczny $p(x)$. Są to rejestry EED-MISR oraz TBDa-MISR. Posiadają one następujące redukowalne wielomiany $p_{cr}(x)$:

$$p_{EED4}(x) = (1 + x)^2(1 + x + x^2) \text{ oraz } p_{TBDa4}(x) = x(1 + x)(1 + x + x^2).$$

W przypadku wystąpienia błędów zależnych $e_r(x)$ na wejściu 4 rejestru EED-MISR lub rejestru TBDa-MISR otrzymujemy

$$e_4(x)p_{EED_4}(x)/p(x) = e_4(x)(1+x)^2(1+x+x^2)/(1+x+x^2)(1+x^3+x^5) =$$

$$= e_4(x)(1+x)^2/(1+x^3+x^5)$$

$$e_4(x)p_{TBD_4}(x)/p(x) = e_4(x)x(1+x)(1+x+x^2)/(1+x+x^2)(1+x^3+x^5) =$$

$$= e_4(x)x(1+x)/(1+x^3+x^5)$$

W efekcie w przypadku rejestrów EED-MISR oraz TBDa-MISR statyczne prawdopodobieństwo maskowania błędów $e_4(x)$ zamiast 2^{-7} jest równe 2^{-5} . Prawdopodobieństwo to nie ulega powiększeniu w przypadku pozostałych dwóch rejestrów IED-MISR oraz TBDb-MISR.

Problem wpływu wielomianów $p_{cr}(x)$ na skuteczność analizy sygnaturowej został szczegółowo zbadany dla rejestrów TBD-MISR oraz BTd-MISR w pracach [Hławi89a, Hławi89b, Hławi92b]. Na podstawie rezultatów uzyskanych w tych pracach można sformułować następujący wniosek:

Wniosek 4.4

Statyczne prawdopodobieństwo maskowania błędów wprowadzonych na r -te wejście rejestrów c -MISR- $p_c(x)$, gdzie $c \in \{BTD, TBD\}$ równe jest 2^{-n} tylko wtedy, gdy każda składowa wielomianu $p_c(x)$ i wielomian $p_{cr}(x)$ są wzajemnie pierwsze lub gdy $p_{cr}(x) = x^r$ lub gdy wielomian $p_c(x)$ jest wielomianem pierwszym.

Wielomiany $p_c(x) = p_{cr}(x)g(x)$ podobnie jak wielomiany $p_c(x) = x^d g(x)$ nazwano w [Hławi89a, Hławi89b, Hławi90a, Hławi92b] wielomianami ułomnymi. Są one przyczyną zwiększenia do poziomu $2^{-(n-d)}$ statycznego prawdopodobieństwa niewykrycia błędów $e_r(x)$. Wniosek ten dla rejestrów EED-MISR chociaż wynikał w sposób niejawni z rezultatów prac [Hławi84c, Hławi86c], wyraźnie został dla tych rejestrów zilustrowany w pracy [Hławi90a]. Wniosek ten w oparciu o wyrażenie 2.1 można rozszerzyć na pozostałe rejestry c -MISR z wyjątkiem rejestrów IED-MISR [Hławi90a], które zawsze posiadają nieułamny wielomian $p_{IED}(x)$. Rezultaty uzyskane w pracach [Hławi89a, Hławi89b] i ujęte we wniosku 4.1 podważyły rezultaty zawarte w pracy [WillD89], z których wynikało, że statyczne prawdopodobieństwo maskowania dowolnych błędów przez n -stopniowe rejestry c -MISR jest równe 2^{-n} niezależnie od rodzaju wielomianu charakterystycznego tego rejestru. Ten błędny wniosek Williama podważa także przykład błędów identycznych na każdym wejściu rejestru c -MISR nazywanych czasem błędami skorelowanymi mającymi swoje źródło w tym samym uszkodzeniu.

Przykład 4.6

Weźmy rejestry $D \oplus D \oplus D \oplus D \oplus DD \oplus D$ (IED-MISR) oraz $D \oplus D \oplus DD \oplus DDD$ (TBDb-MISR) z przykładu 4.4. Ich wielomiany charakterystyczne spełniają warunki wniosku 4.4, a więc są nieułamne. Przy założeniu wystąpienia jednocześnie na trzech pierwszych wejściach rejestru IED-MISR identycznych błędów opisanych wielomianem $e(x)$ uzyskuje się następujący algebraiczny model pracy tego rejestru

$$[e(x) + e(x)x + e(x)x^2]/(1+x+x^2)(1+x^3+x^5) =$$

$$= e(x)(1+x+x^2)/(1+x+x^2)(1+x^3+x^5) = e(x)/(1+x^3+x^5).$$

Wynika z niego, że dla takiej klasy błędów statyczne prawdopodobieństwo ich niewykrycia jest powiększone do poziomu $2^{-(7-2)}$. Podobna sytuacja wystąpi w przypadku rejestru $D \oplus D \oplus DD \oplus DDD$ przy założeniu wystąpienia identycznych błędów $e(x)$ na jego wejściach 2, 3 oraz 4.

Tak więc jeżeli $p(x) = d(x)g(x)$ oraz jednocześnie $g(x) = \sum_{r=0}^{n-1} \oplus p_{cr}(x)$, wówczas może nastąpić przytoczona w powyższym przykładzie redukcja skuteczności analizy sygnaturowej identycznych błędów. Wynika z tego ostatecznie następujący wniosek:

Wniosek 4.5

Statyczne prawdopodobieństwo maskowania błędów przez rejestr c -MISR- $p(x)$ równe jest 2^{-n} , gdy wielomian $p(x)$ jest wielomianem pierwszym, a więc wielomianem nierozkładalnym.

W pracach [Hassa82, Hławi84c, Hławi86c, Hławi87c, Hławi88a, Hławi89a, Hławi89b, Hławi90a, Hławi92b] udowodniono, że rejestry c -MISR- $p(x)$ o dowolnych wielomianach nieułamnych wykrywają paczki błędnych wektorów $\mathbf{E}(n,t)$, tzn. takie błędne macierze \mathbf{E} , które zawierają $t \leq n$ identycznych wektorów błędów w odcinku macierzy \mathbf{E} nie dłuższym niż n . Oznacza to między innymi, że macierz \mathbf{E} złożona tylko z jednego błędnego wektora będzie wykryta przez każdy równoległy kompaktor liniowy c -MISR- $p(x)$ związany z nieułamnym wielomianem $p(x)$. Rejestr MIShR nie posiada tych właściwości detekcyjnych.

Wnioski uzyskane dotąd przy użyciu pierścienia wielomianów nad ciałem $GF(2)$ nie rozwiązują wszystkich problemów związanych z wyborem wielomianów $p(x)$ zapewniających najlepszą skuteczność kompaktacji liniowej.

4.2.4. Dynamiczne prawdopodobieństwo maskowania błędów

Wnioski 4.3, 4.4 oraz 4.5 umożliwiają ograniczenie zainteresowania projektanta wewnątrzukładowych kompaktorów liniowych do zbioru wielomianów $p(x)$ nieułamnych rozkładalnych i pierwszych w przypadku każdego z rejestrów c-SISR- $p(x)$ oraz zbioru wielomianów $p(x)$ nierozkładalnych w przypadku każdego z rejestrów c-MISR- $p(x)$. W dalszym ciągu otwarte pozostaje jednak pytanie zawarte w tytule rozdziału 4.2.3, ponieważ np. nie wiadomo, które wielomiany charakterystyczne są najlepsze czy pierwsze niepierwotne czy też pierwsze, które są jednocześnie pierwotne. Pełna odpowiedź wymaga wyjaśnienia jeszcze jednego następującego zagadnienia:

Jaki wpływ na przebieg funkcji $P_{al}(m)$ mają różne wielomiany charakterystyczne $p(x)$?

W pracy [HławM89] udowodniono, że użycie szeregowego kompaktora liniowego związanego z wielomianem $p(x)$ zapewniającym małe prawdopodobieństwo maskowania dla każdej z rzeczywistych klas uszkodzeń zapewnia również małe całkowite prawdopodobieństwo maskowania. To podstawowe kryterium optymalnego wyboru sprzężeń liniowych kompaktora c-SISR posłużyło w tej pracy za punkt wyjścia do opracowania teorii wpływu liniowych sprzężeń zwrotnych na skuteczność szeregowej kompaktacji liniowej. W celu zupełnego odejścia od błędów niezależnych i wielomianowego opisywania błędów zależnych wprowadzono w [HławM89] model uszkodzeń TUC w postaci tzw. standardowych klas uszkodzeń, które są przybliżeniami rzeczywistych klas uszkodzeń. Oto jego definicja:

Definicja 4.1

Model standardowej klasy uszkodzeń określony jest przez parametry b oraz m w następujący sposób:

- dla każdego ciągu błędów $\{E_i\}$ zdarzenia polegające na wylosowaniu jedynek na różnych pozycjach ciągu są zdarzeniami niezależnymi,
- dla każdego ciągu błędów $\{E_i\}$ prawdopodobieństwo wylosowania jedynki na jego dowolnej pozycji jest identyczne i wynosi b ,
- liczba bitów (długość) m każdego ciągu błędów $\{E_i\}$ jest jednakowa.

Podobny model błędów wykorzystywano także w pracach [David86, WillD86, WillD87a, IvanA87] związanych z badaniem wpływu sprzężenia liniowego kompaktorów liniowych na prawdopodobieństwo maskowania uszkodzeń. Wystąpienie każdego uszkodzenia interpreto-

wane jest w modelu standardowej klasy uszkodzeń jako wylosowanie ciągu błędów $\{E_i\}$. Pozwoliło to uniezależnić wyniki pracy [HławM89] od konieczności (najczęściej nierealnego) wyznaczenia wszystkich uszkodzeń TUC. Model ten pozwolił także na uniezależnienie wyników tej pracy od kolejności, w jakiej podawane są testy na wejścia TUC. Zamiast dobierać dla rzeczywistych klas uszkodzeń wielomian charakterystyczny $p(x)$ związany z kompaktorem liniowym, można go dobierać według podanego wcześniej kryterium dla standardowej klasy uszkodzeń określanej wartością prawdopodobieństwa b .

W zbiorze standardowych klas uszkodzeń występuje także ciąg złożony z samych zer. Jest on interpretowany jako brak uszkodzenia. Przy obliczaniu prawdopodobieństwa maskowania P dla danej standardowej klasy uszkodzeń można pominąć ten przypadek stosując następujący wzór [HławM89]:

$$P_{al} = [P_Q - P_Z] / [1 - P_Z] \quad (4.2)$$

w którym P_Q jest prawdopodobieństwem wylosowania z danej standardowej klasy uszkodzeń ciągu $\{E_i\}$ przetwarzanego w sygnaturę zerową, natomiast P_Z jest prawdopodobieństwem wylosowania z danej standardowej klasy uszkodzeń ciągu $\{E_i\}$ złożonego z samych zer. Zauważmy, że $P_Z = (1 - b)^m$. Wartość P_Q zarówno dla IED-SISR, jak i EED-SISR dla $m > n$ ograniczyć można w oparciu o [HławM89] poniższymi nierównościami

$$b \leq 0.5 \quad b^n \leq P_Q \leq (1 - b)^n \quad (4.3a)$$

$$b \geq 0.5 \quad (1 - b)^n \leq P_Q \leq b^n \quad (4.3b)$$

natomiast dla $m \leq n$ określić można za pomocą równości

$$P_Q = (1 - b)^m \quad (4.3c)$$

Wyrażenia 4.3 otrzymali wcześniej Williams i inni w pracach [WillD86, WillD87a], przy założeniu że $P_Q = P_{al}$. Nierówności 4.3 przedstawiono w [WillD86, WillD87a, HławM89] w postaci odpowiednich wykresów umożliwiających zilustrowanie obszaru dopuszczalnych wartości P_{al} w zależności od wartości b oraz przy $m = \text{const}$. Po raz pierwszy określono więc dolną i górną granicę wartości P_{al} , które mogą przyjmować dla założonej standardowej klasy uszkodzeń kompaktory liniowe związane z różnymi wielomianami $p(x)$. Założenie $P_Q = P_{al}$, ze względu na abstrakcyjne zerowe ciągi błędów zawarte w definicji standardowej klasy uszkodzeń, niedokładnie odwzorowuje rzeczywiste prawdopodobieństwo maskowania uszkodzeń. W efekcie dla $m \leq n$ prawdopodobieństwo P_{al} w oparciu o 4.3c różne

jest od zera dla każdego $0 < b < 1$, chociaż w rzeczywistości dla takich długości m maskowanie w ogóle nie występuje. Podobnie dla $m > n$ oraz $b = 0.5$ prawdopodobieństwo P_{al} w oparciu o nierówności 4.3 odbiega od rzeczywistości, ponieważ równe jest wartości 2^{-n} niezależnie od wartości m . W celu uniknięcia tych wad należy określić prawdopodobieństwo maskowania P_{al} w oparciu o wyrażenie 4.2. Zrealizowano to w [HławM89], gdzie nierówności 4.3 po wprowadzeniu do 4.2 umożliwiły dla $m > n$ ostatecznie ograniczyć wartość prawdopodobieństwa maskowania P_{al} dla rejestrów IED-SISR oraz EED-SISR nierównościami o następujących postaciach:

$$\frac{b^n - (1-b)^m}{1 - (1-b)^m} \leq P_{al} \leq \frac{[(1-b)^n - (1-b)^m]}{1 - (1-b)^m} \quad \text{dla } b \leq 0.5 \quad (4.4a)$$

$$\frac{[(1-b)^n - (1-b)^m]}{1 - (1-b)^m} \leq P_{al} \leq \frac{b^n - (1-b)^m}{1 - (1-b)^m} \quad \text{dla } b \geq 0.5 \quad (4.4b)$$

Z nierówności tych w [HławM89] otrzymano dla $m > n$ oraz $b = 0.5$ wyrażenie

$$P_{al} = \frac{[2^{-n} - 2^{-m}]}{[1 - 2^{-m}]} \quad (4.5)$$

które potwierdza wyniki teoretyczne uzyskane dla jednakowo prawdopodobnych błędów w [Froh77, David78, Smith80]. Po podstawieniu wartości P_Q (4.3c) do 4.2 w pracy [HławM89] otrzymano dla $m \leq n$ oraz $0 < b < 1$ wartość $P_{al} = 0$. W pracy [HławM89] ponadto udowodniono, że dla m dążącego do nieskończoności przebieg funkcji $P_{al}(m)$ dla $P_{al} = [P_Q - P_Z] / [1 - P_Z]$ dąży do wartości statycznego prawdopodobieństwa 2^{-n} nie tylko dla $b = 0.5$, ale również dla każdej innej standardowej klasy uszkodzeń. Podobny dowód dla $P_{al} = P_Q$ przeprowadzono wcześniej w pracach [WillD86, WillD87a]. Wyrażenie 4.2 wykorzystano także w pracy [HławM89] do określenia nierówności, które wykazały, że dla standardowych klas uszkodzeń zdeterminowanych przez $b < 0.5$, rejestry o strukturze FSR nie nadają się do analizy sygnaturowej. Rezultat ten podważył częściowo wartość prac Davida [David78, David80, David85, David86], który starał się wykazać, że rejestry ze sprzężeniem wyprowadzonym tylko z ostatniego przerzutnika nie są gorsze od rejestrów ze strukturami sprzężeń liniowych typu IED oraz EED. David udowodnił to tylko dla m dążącego do nieskończoności. Tym samym problemem zajmował się także Ivanov oraz Agarwal w [IvanA87]. W pracy tej, zakładając, że $P_{al} = P_Q - P_Z$, wyprowadzono wyrażenia udowadniające, że rejestry FSR mają gorsze właściwości detekcyjne dla $b < 0.5$ od rejestrów IED-SISR oraz EED-SISR.

Ostateczną odpowiedź na pytanie, jaki wpływ na przebieg funkcji $P_{al}(m)$ mają różne wielomiany charakterystyczne $p(x)$, umożliwiła symulacja procesu Markowa realizowana dla różnych standardowych klas uszkodzeń. Szczegóły dotyczące jego opisu w zastosowaniu do określania prawdopodobieństwa maskowania standardowych klas uszkodzeń w rejestrach c-SISR- $p(x)$ znaleźć można w pracy [HławM89]. Proces Markowa użyty został do obliczenia prawdopodobieństwa powrotu do stanu zerowego rejestru c-SISR- $p(x)$ zasilanego ciągiem błędów lub rejestru c-MISR- $p(x)$ zasilanego ciągiem błędnych wektorów. Jego efektem są krzywe ilustrujące przebieg prawdopodobieństwa P_{al} maskowania błędów w funkcji długości ciągu m dla założonej wartości b . W pracach [WillD86, WillD87a, WillD87b] do realizacji procesu Markowa wykorzystano wyrażenie $P_{al} = P_Q$, natomiast w pracach [IvanA87, IvanA89] użyto wyrażenia $P_{al} = P_Q - P_Z$. Krzywe $P_{al}(m)$ uzyskane w efekcie zastosowania w procesie Markowa tych wyrażeń obciążone są pewnym błędem wynikającym z braku w definicji P_{al} mianownika $1 - P_Z$. W celu uniknięcia tego błędu w pracy [HławM89] do realizacji procesu Markowa wykorzystano wyrażenie 4.2. W ww. pracach pokazano, że dla wszystkich wielomianów charakterystycznych związanych z rejestrami IED-SISR- $p(x)$ oraz EED-SISR i dla $0 < b < 1$ prawdopodobieństwo maskowania standardowych klas uszkodzeń zmierza do asymptoty 2^{-n} wykresu funkcji $P_{al}(m)$ przy m dążącym do nieskończoności. W pracach tych zauważono także, że o przydatności konkretnego szeregowego kompaktora liniowego związanego z wielomianem $p(x)$ decyduje szybkość osiągnięcia wartości granicznej. Zauważono, że w pierwszej fazie zbiegania do 2^{-n} krzywe te posiadają stany nieustalone, które dla różnych wielomianów charakterystycznych są różne. Dynamiczny charakter uzyskanych w efekcie symulacji procesu Markowa funkcji $P_{al}(m)$ stał się motywacją do nazywania tych funkcji dynamicznym prawdopodobieństwem maskowania błędów [Hławi93b].

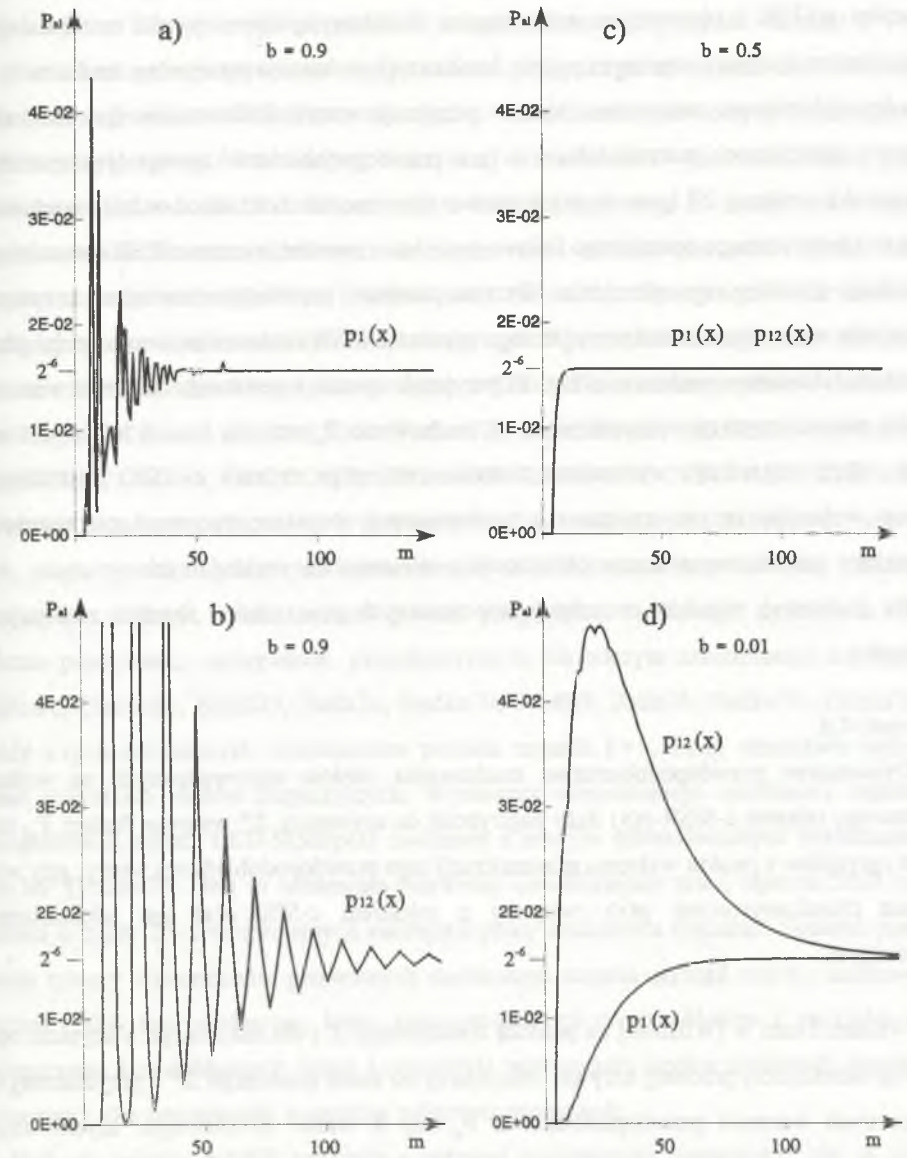
Różne dynamiczne właściwości procesu Markowa pozwoliły autorom prac [WillD86, WillD87a, WillD87b, IvanA87, HławM89, IvanA89] określić, które wielomiany charakterystyczne są najlepsze dla celów testowania. Krzywe $P_{al}(m)$ rejestrów c-SISR- $p(x)$ związanych z wielomianami pierwotnymi posiadają dla $b \rightarrow 0$ i $b \rightarrow 1$ unikalne właściwości, których nie mają krzywe $P_{al}(m)$ dla rejestrów c-SISR- $p(x)$ związanych z wielomianami rozkładalnymi. Dla $b > 0.5$ krzywe $P_{al}(m)$ w przypadku obu rodzajów rejestrów posiadają szereg pików. Górna granica ich wartości szczytowych znacznie przekracza poziom 2^{-n} . Granica ta jest jednak znacznie mniejsza w przypadku rejestrów z wielomianami pierwot-

nymi oraz znacząco większa w przypadku rejestrów z wielomianami rozkładalnymi. Piki związane z wielomianami pierwotnymi o wiele szybciej zanikają niż piki krzywej $P_{al}(m)$ związanej z wielomianami rozkładalnymi. W przypadku wielomianów pierwotnych zanik pików występuje już przy długości m zbliżonej do wartości 2^{n-1} charakterystycznej dla długości cyklu tych wielomianów. Dla obu rodzajów wielomianów funkcja $P_{al}(m)$ po zaniknięciu pików przyjmuje wartości 2^{-n} . Dla $b < 0.5$ krzywe $P_{al}(m)$ związane z wielomianami pierwotnymi przebiegają zawsze w stanie nieustalonym poniżej wartości 2^{-n} , podczas gdy krzywe $P_{al}(m)$ związane z wielomianami rozkładalnymi przebiegają w stanie nieustalonym zawsze powyżej tej wartości. Zauważono także, że dla jednakowo prawdopodobnych błędów, tzn. dla $b = 0.5$, krzywe $P_{al}(m)$ są identyczne dla obu rodzajów wielomianów.

Tabela 4.1

$p_1(x) = x^6 + x^5 + x^2 + x + 1$	pierwotny	cykl = 63
$p_2(x) = x^6 + x + 1$	pierwotny	cykl = 63
$p_3(x) = x^6 + x^5 + x^4 + x^2 + 1$	pierwszy	cykl = 21
$p_4(x) = x^6 + x^3 + 1$	pierwszy	cykl = 9
$p_5(x) = x^6 + x^5 + x^4 + x^3 + x + 1$	$= (x+1)(x^5 + x^3 + 1)$	cykl = 31
$p_6(x) = x^6 + x^4 + x^3 + 1$	$= (x+1)(x^5 + x^4 + x^2 + x + 1)$	cykl = 31
$p_7(x) = x^6 + x^4 + x^3 + x^2 + x + 1$	$= (x+1)^2(x^4 + x + 1)$	cykl = 30
$p_8(x) = x^6 + x^5 + x^4 + x^3 + 1$	$= (x^2 + x + 1)(x^4 + x + 1)$	cykl = 34
$p_9(x) = x^6 + x^5 + x + 1$	$= (x+1)^2(x^4 + x^3 + x^2 + x + 1)$	cykl = 10
$p_{10}(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$	$= (x^3 + x + 1)(x^3 + x^2 + 1)$	cykl = 7
$p_{11}(x) = x^6 + x^4 + 1$	$= (x^3 + x^2 + 1)^2$	cykl = 14
$p_{12}(x) = x^6 + 1$	$= (x+1)^2(x^2 + x + 1)^2$	cykl = 6

W celu zilustrowania tych różnic w [Hławi93b] przeprowadzono analizę porównawczą krzywych $P_{al}(m)$ dla rejestrów c-SISR- $p(x)$ związanych z wielomianami charakterystycznymi $p(x)$ szóstego stopnia przedstawionymi w tabeli 4.1. Dla pierwotnego wielomianu charakterystycznego $p_1(x)$ istnieją struktury IED, EED, cztery różne struktury typu LTD cztery różne struktury TBD, dwie struktury typu CAa i CAb (101001; 100101) oraz wiele struktur IEDT i EEDT. Również wiele struktur można zrealizować w oparciu o pozostałe rozkładalne i pierwsze wielomiany charakterystyczne. Obliczenia zrealizowano dla wszystkich istniejących rejestrów c-SISR. Dla wszystkich struktur sprzężeń liniowych związanych z wybranym wielomianem charakterystycznym uzyskano identyczne wykresy $P_{al}(m)$, co potwierdza tezę zawartą we wniosku 4.2.



Rys. 4.2. Wykresy krzywych $P_{al}(m)$ dla rejestrów c-SISR szóstego stopnia a) dla $p_1(x)$ oraz $b = 0.9$; b) dla $p_{12}(x)$ oraz $b = 0.9$; c) dla $p_1(x)$ i $p_{12}(x)$ oraz $b = 0.5$; d) dla $p_1(x)$ i $p_{12}(x)$ oraz $b = 0.01$
 Fig. 4.2. Aliasing probability $P_{al}(m)$ for registers c-SISR with six stages: for $p_1(x)$ and $b = 0.9$ (a), for $p_{12}(x)$ and $b = 0.9$ (b), for $p_1(x)$, $p_{12}(x)$ and $b = 0.5$ (c), for $p_1(x)$, $p_{12}(x)$ and $b = 0.001$ (d)

Na wykresach przedstawionych na rys. 4.2 umieszczono krzywe $P_{al}(m)$ dla dwóch rejestrów c-SISR z pierwotnym wielomianem charakterystycznym $p_1(x)$ i rozkładalnym wielomianem charakterystycznym $p_{12}(x)$. Analiza tych wykresów potwierdza, że dla $m > n$ prawdopodobieństwo maskowania błędów przyjmuje wartości nieustalone (prawdopodobieństwo dynamiczne), natomiast dla $m \gg n$ to prawdopodobieństwo asymptotycznie zbiega do wartości ustalonej 2^{-n} (prawdopodobieństwo statyczne) niezależnie od rodzaju wielomianu charakterystycznego opisującego liniowe sprzężenia zwrotne rejestru c-SISR i niezależnie od rodzaju struktury tego sprzężenia. Wyrażna zależność przebiegu omawianych krzywych od rodzaju wielomianu charakterystycznego rejestru c-SISR zaznacza się w obszarze, gdzie prawdopodobieństwo maskowania błędów przyjmuje wartości nieustalone. Rozrzut wartości stanów nieustalonych prawdopodobieństwa maskowania $P_{al}(m)$ dla $b \approx 0$ lub $b \approx 1$ też silnie zależy od rodzaju wielomianu charakterystycznego rejestru c-SISR. Najmniejszy rozrzut wykazują krzywe związane z wielomianami charakterystycznymi pierwotnymi, największy zaś krzywe związane z większością wielomianów rozkładalnych.

Na podstawie wyników wcześniej wspomnianych prac można określić następujący wniosek:

Wniosek 4.6

Dynamiczne prawdopodobieństwo maskowania błędów wprowadzonych na wejście n-bitowego rejestru c-SISR-p(x) dąży najszybciej do asymptoty 2^{-n} wykresu funkcji $P_{al}(m)$ i jest optymalne z punktu widzenia minimalizacji tego prawdopodobieństwa wtedy, gdy wielomian charakterystyczny p(x) związany z rejestrem c-SISR-p(x) jest wielomianem pierwotnym.

Williams i inni w [WillD86] za pomocą transformaty Z i dla założonego b uzyskali opis funkcji określającej przebieg krzywej zbiegającej do stanu ustalonego 2^{-n} i przybliżonej do granicznych wartości prawdopodobieństwa $P_{al}(m)$ w stanie nieustalonym uzyskiwanym w symulacji procesu Markowa. W pracy [HławM89] przeprowadzono podobne obliczenia. Dokładniejszy opis tej funkcji, również przy użyciu transformaty Z, uzyskano w pracy [SeirV91]. Podobny rezultat uzyskał również Gupta w [GuptP88] przy użyciu teorii kodów cyklicznych. W pracy [HławM89] zrealizowano także próbę określenia wyrażenia matematycznych opisujących prawdopodobieństwo maskowania błędów w funkcji długości cykli rejestru c-SISR. Długość cykli jest ściśle związana z rodzajem wielomianu charakte-

rystycznego p(x) rejestru c-SISR-p(x). Otrzymane w [HławM89] wyrażenia umożliwiają analizę wpływu długości cykli na szybkość zbiegania krzywej $P_{al}(m)$ do wartości 2^{-n} , co ma istotne znaczenie przy wyborze długości testu podawanego na wejścia TUC. Funkcje $P_{al}(m)$ dla rejestrów c-SISR-p(x) związanych z wielomianem pierwotnym posiadającym najdłuższy cykl równy $2^n - 1$ zbiegają najszybciej do wartości 2^{-n} już mniej więcej przy m zbliżonym do wartości $2^n - 1$. Obserwacje te stały się motywacją do poszukiwania najlepszych testerów wewnątrzukładowych ze sprzężeniem nieliniowym wśród tych, które charakteryzowały się najdłuższym cyklem pracy [Badur92]. Obserwacje te jak również wcześniej opisane właściwości dynamiczne krzywych $P_{al}(m)$ potwierdzają także słuszność wyboru wielomianu charakterystycznego rejestru EED-SISR dokonanego przez wynalazców pierwszego analizatora sygnatur 5004 [Froh77]. Odrzucili oni rozkładalne wielomiany typu CRC-12, CRC-16, CRC-CCITT, CRC-16 REVERSE, CRC-CCITT REVERSE, które wcześniej okazały się najlepsze do wykrywania błędów pojedynczych, podwójnych, potrójnych, nieparzystych i grupowych (spowodowanych zakłóceniami impulsowymi, krótkimi przerwami transmisji, uszkodzeniami powłok magnetycznych pamięci dyskowych itp.) podczas przesyłania, zapisywania, przechowywania lub odczytu zakodowanej informacji źródłowej [Socha66, Buck175, Ralla78, Budko79, Seidl83, Ralla78, Budko79, Gutma79]. Każdy z tych odrzuconych wielomianów posiada czynnik $1+x$, który umożliwia wykrywanie wszystkich błędów nieparzystych. Wynalazcy wspomnianego analizatora sygnatur zaproponowali rejestr EED-SISR-p(x) związany z nowym nierozkładalnym wielomianem typu HP [Froh77]. Jest to wielomian pierwotny umożliwiający pracę rejestru SISR jako licznika o cyklu $2^n - 1$ w specjalnych rodzajach pracy analizatora sygnatur. Spośród ponad dwóch tysięcy wielomianów pierwotnych szesnastego stopnia wybrali oni do analizatora sygnatur 5004 taki wielomian, który nie wyróżnia ani ciągów błędów z parzystą, ani z nieparzystą liczbą błędnych bitów i umożliwia wykrywanie błędów zależnych charakterystycznych dla testowanych systemów mikroprocesorowych.

Fakt, że rejestry c-SISR związane z różnymi wielomianami pierwotnymi nie są sobie równoważne zauważył Robinson w pracy [Robin91]. Zaobserwował on w niej, że rejestry c-SISR-p(x) tego samego stopnia n, ale związane z różnymi wielomianami pierwotnymi p(x), mają różne krzywe $P_{al}(m)$, szczególnie dla $b \approx 1$. Po licznych symulacjach procesu Markowa zauważył on także, że dla różnych wartości b najlepsze właściwości dynamiczne

posiadają krzywe $P_{al}(m)$ rejestrów c-SISR-p(x) związanych z wielomianami pierwotnymi zawierającymi około połowę niezerowych współczynników.

Wymienione dotąd prace poświęcone dynamicznemu prawdopodobieństwu maskowania błędów zawierały przykłady, w których ze względu na najszybciej zbiegające do wartości 2^n górne graniczne wartości pików rejestry c-SISR związane z wielomianami pierwotnymi były bardziej preferowane od rejestrów związanych z wielomianami niepierwotnymi. Pozwoliło to dla wielomianów pierwotnych skrócić długość testów gwarantujących osiągnięcie poziomu statycznego prawdopodobieństwa do wartości $m \geq 2^n$. W pracy [OlivD93] na podstawie rezultatów symulacji procesu Markowa udowodniono, że rejestry c-SISR-p(x) z wielomianem p(x) typu $(1+x)g(x)$ lub bazującym na kodach Fire'a typu $(1+x^a)g(x)$ odzwierciedlają dla długości testu $m < 2^n$ zachowanie rejestrów c-SISR związanych z wielomianem pierwotnym. W wielomianach tych g(x) jest wielomianem nieredukowalnym stopnia n - a. Autorzy tej pracy wykazali, że jeśli g(x) jest wielomianem pierwotnym (długi cykl) i długość testu m jest krótsza od 2^{n-a} , wówczas prawdopodobieństwo maskowania standardowych klas uszkodzeń $P_{al}(m)$ reprezentowane przez ten rejestr jest mniejsze od tego, które reprezentuje rejestr związany z wielomianem pierwotnym. Długość 2^{n-a} jest w rzeczywistości długością maksymalnego cyklu wynikającego z wielomianu g(x). W pracy tej jednocześnie zauważono, że gdy $m > 2^{n-a}$, to prawdopodobieństwo to dla $b < 0.5$ "ekspłduje" do bardzo wysokich wartości (większych od 2^{-n}), których dla tych długości nie posiada krzywa $P_{al}(m)$ związana z wielomianem pierwotnym. Jeżeli rejestr c-SISR posiada odpowiednią liczbę n komórek, wówczas wartość graniczna długości testu będzie większa niż jakakolwiek praktyczna długość testu m. Na przykład, dla 32-bitowego rejestru z wielomianem $p(x) = (1+x)g(x)$ graniczna długość testu może dochodzić do $2^{31} \cong 10^9$, czyli może przekraczać wszelkie realne długości testu. Przy założeniu nieingerowania w przyjętą wcześniej liczbę bitów rejestru c-SISR ostateczny wybór pomiędzy wielomianem pierwotnym a wielomianem $p(x) = (1+x^a)g(x)$ zależeć będzie od długości testu gwarantującego odpowiedni współczynnik pokrycia uszkodzeń TUC.

W pracach [DamiO89a, Hławi89a, Hławi89b, WillD89, DamiO89b, IvanA89, Hławi89c, Hławi90a, DamiO90, IvanP90, PradG91, DamiO91, Hławi92b, OlivD93, PilaK95] przedstawiono wyniki badań przy użyciu symulacji procesu Markowa wpływu wielomianu p(x) na dynamiczne prawdopodobieństwo maskowania błędów przez rejestry c-MISR-p(x). Zakres tych badań w [Hławi89a, Hławi89b, Hławi89c, Hławi90a, Hławi92b]

ograniczono do możliwości wystąpienia błędów ze standardowej klasy uszkodzeń tylko na r-tym wejściu rejestru c-MISR-p(x) (testery wewnętrzne STUMPS i LOCST [PilaK95]). Wyniki tych badań pozwoliły stwierdzić, że dynamiczne prawdopodobieństwo maskowania błędów na r-tym wejściu rejestru c-MISR-p(x) z danym nierozkładalnym charakterystycznym wielomianem p(x) jest podobne do dynamicznego prawdopodobieństwa maskowania błędów dla rejestrów c-SISR-p(x) z takim samym wielomianem charakterystycznym p(x). W pracach tych na podstawie rezultatów symulacji procesu Markowa potwierdzona została również zależność wartości statycznego prawdopodobieństwa maskowania błędów na r-tym wejściu rejestru c-MISR-p(x) od podzielności wielomianu $p_{cr}(x)$ przez rozkładalny ułomny wielomian charakterystyczny p(x) tego rejestru. Ilustrują to krzywe $P_{al}(m)$ [Hławi89c, Hławi90a], przedstawione na rysunku 4.3 i określone dla rejestru EED-MISR-p(x) związanego z wielomianem ułomnym

$$p(x) = 1 + x + x^2 + x^3 + x^4 + x^8 = (1+x)^3(1+x+x^2)(1+x+x^3).$$

Symulacja została zrealizowana dla różnych wartości b ($b = 0.1$, $b = 0.5$, $b = 0.9$) i dla różnych wejść ($r = 4$, $r = 5$, $r = 6$) związanych z następującymi wielomianami

$$P_{EED4} = (1+x)^4, P_{EED5} = (1+x+x^2)(1+x^2+x^3),$$

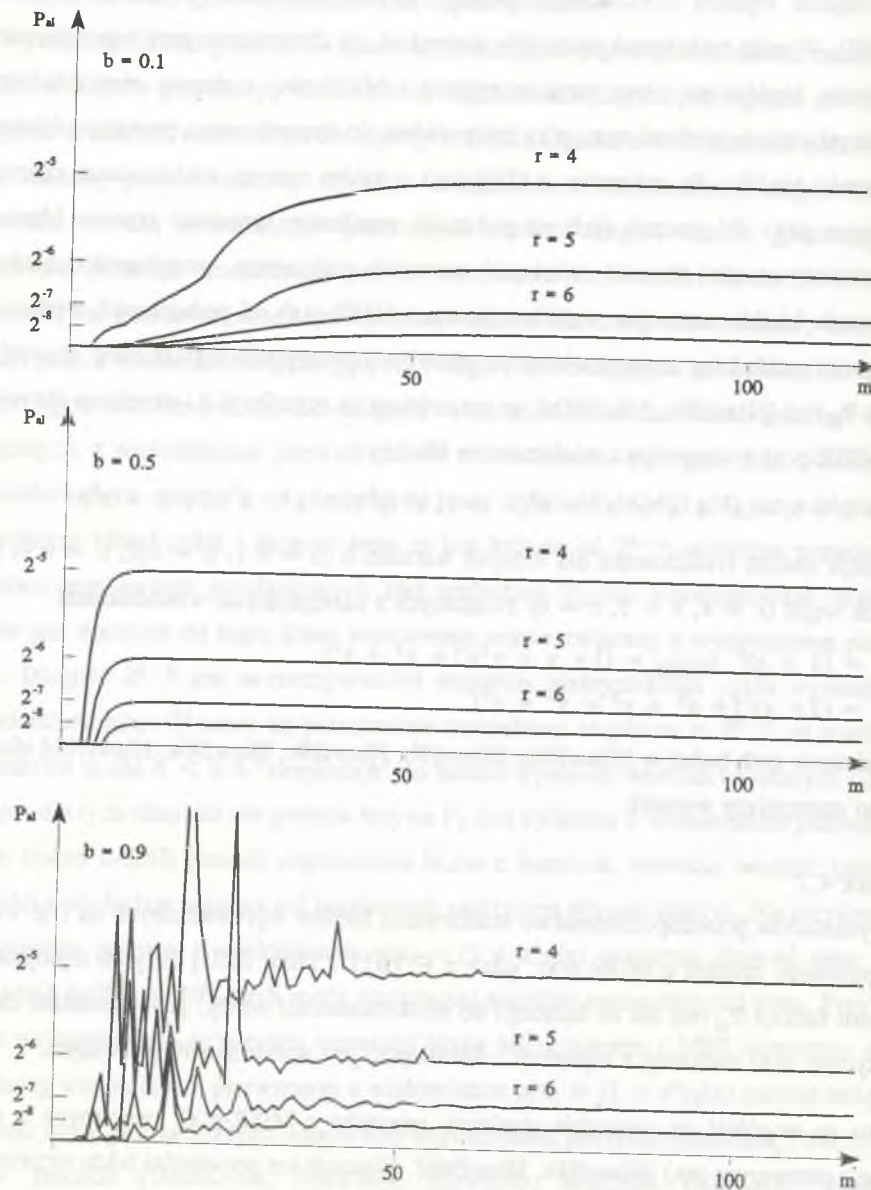
$$P_{EED6} = (1+x)(1+x^2+x^3+x^4+x^5).$$

Na podstawie tych badań w [Hławi89a, Hławi89b, Hławi89c, Hławi90a, Hławi92b] sformułowano następujący wniosek:

Wniosek 4.7

Dynamiczne prawdopodobieństwo maskowania błędów wprowadzonych na r-te wejście n-stopniowego rejestru c-MISR-p(x), gdzie $c \in \{BTD, TBD, EED\}$ dąży do asymptoty 2^{-n} wykresu funkcji $P_{al}(m)$ dla m dążącego do nieskończoności wtedy, gdy wielomian charakterystyczny p(x) związany z rejestrem c-MISR-p(x) jest wielomianem pierwszym.

Można go uogólnić na wszystkie struktury rejestrów c-MISR-p(x) związanych z wielomianem pierwszym p(x) [Hławi93b, Hławi96b]. Wniosek ten potwierdza także wcześniejsze wnioski 4.4 i 4.5. Wśród wielomianów pierwszych te, które były pierwotne, posiadały najlepsze właściwości dynamiczne krzywych $P_{al}(m)$. Niestety ograniczenie badań dynamicznego prawdopodobieństwa maskowania błędów do błędów wprowadzanych wyłącznie na jedno wejście rejestru c-MISR-p(x) uniemożliwiało stwierdzenie, które wielomiany p(x)



Rys. 4.3. Dynamiczne prawdopodobieństwo maskowania błędów $P_{al}(m)$ na wejściu r rejestru EED-MISR- $p(x)$ związanego z wielomianem ułomnym $p(x) = 1 + x + x^2 + x^3 + x^4 + x^8$

Fig. 4.3. Dynamic probability of aliasing $P_{al}(m)$ on r -th input of register EED-MISR- $p(x)$ with $p(x) = 1 + x + x^2 + x^3 + x^4 + x^8$

są najlepsze dla celów testowania w przypadku stosowania kompaktora c-MISR- $p(x)$ w testerach wewnątrzukładowych innych niż np. STUMPS czy też LOCST. Ten problem został szczegółowo zbadany przez Damianiego i innych w pracach [DamiO89a, DamiO89b, DamiO90, DamiO91]. Szczególnie praca [DamiO91] wymazała większość znaków zapytania związanych z tym zagadnieniem. Badania w [DamiO91] przeprowadzono dla takich macierzy błędów, w których każdy wektor błędu posiada swoje własne niezależne prawdopodobieństwo pojawienia się. W pracy tej wyprowadzono formalne równania opisujące model łańcucha Markowa dla liniowych automatów oraz narzędzia matematyczne potrzebne do rozwiązania tych równań. Na tej podstawie udowodniono, że zachowanie równoległych n -bitowych kompaktorów liniowych w postaci dowolnego automatu liniowego związanego z pierwotnym wielomianem charakterystycznym jest optymalne z punktu widzenia minimalizacji prawdopodobieństwa maskowania błędów. W pracy tej pokazano, że dynamiczne prawdopodobieństwo maskowania błędów w przypadku takich automatów liniowych dąży najszybciej do asymptoty 2^{-n} wykresu funkcji $P_{al}(m)$. Na podstawie wyników tej pracy można ostatecznie określić następujące wnioski:

Wniosek 4.8

Dynamiczne prawdopodobieństwo maskowania macierzy błędów przez n -bitowy rejestr c-MISR- $p(x)$ dąży najszybciej do asymptoty 2^{-n} wykresu funkcji $P_{al}(m)$ i jest optymalne z punktu widzenia minimalizacji tego prawdopodobieństwa wtedy, gdy wielomian charakterystyczny $p(x)$ związany z rejestrem c-MISR- $p(x)$ jest wielomianem pierwotnym.

Wniosek 4.9

Dynamiczne prawdopodobieństwo maskowania macierzy błędów przez n -stopniowy rejestr c-MISR- $p_c(x)$ związany z wielomianem pierwotnym $p_c(x)$ jest niezależne od rodzaju struktury sprzężenia liniowego wynikającego z tego wielomianu.

Badania właściwości dynamicznych funkcji $P_{al}(m)$ realizowano dla wielu różnych modeli błędów. Opis większości z tych modeli można znaleźć w pracy [PilaK95]. Modele ujęte w tej pracy i inne modele błędów np. model błędów skorelowanych w czasie i przestrzeni [EdirR93] czy też uniwersalny model błędów [KarGP91] są jedynie pewnymi lepszymi lub gorszymi przybliżeniami rzeczywistych klas uszkodzeń. W związku z tym również uzyskiwane dla tych modeli wykresy probabilistycznej funkcji $P_{al}(m)$ są tylko

lepiej lub gorzej zbliżone do rzeczywistości. Dlatego służą one jedynie do oszacowania przybliżonej skuteczności kompaktora liniowego. Może się więc zdarzyć, że rejestr c-MISR-p(x) ze sprzężeniem opisanym za pomocą pierwotnego wielomianu p(x) będzie miał dla specyficznej klasy rzeczywistych błędów gorsze właściwości detekcyjne od rejestru z niepierwotnym wielomianem charakterystycznym. To spostrzeżenie potwierdziły rezultaty badań opisanych w [AhmNG90]. Autorzy tej pracy zamiast podejścia probabilistycznego zastosowali deterministyczną analizę skuteczności kompaktorów liniowych opartą na komputerowej symulacji uszkodzeń oraz komputerowej symulacji sygatur. To podejście stało się ostatnio bardzo popularne, czego najlepszym przykładem są prace [RajsT91, LambI91, DebGD92, PomeR95, Frank95, KunHL95]. W [RajsT91, DebGD92] opisano praktyczne eksperymenty zrealizowane przy użyciu wzorcowych układów ISCAS 85 i przy zastosowaniu pełnej symulacji uszkodzeń i symulacji sygatur. Potwierdziły one, że kompaktory liniowe MISR-p(x) związane z pierwotnymi wielomianami charakterystycznymi p(x) gwarantują najszybsze zbieganie krzywej $P_{a1}(m)$ do wartości 2^{-n} dla błędów zależnych wywołanych uszkodzeniami typu s-z-x. Określenie rzeczywistych wykresów $P_{a1}(m)$ wymaga dokładnego określenia zbioru maskowanych błędów dla dowolnej długości m sekwencji testów oraz dla bardzo dużego zbioru uszkodzeń. Powinno to być realizowane w rozsądnym (akceptowalnym) czasie. Dlatego też stało się to motywacją do opracowywania przez naukowców szybkich procedur symulacji uszkodzeń oraz szybkich procedur obliczania sygatur. W celu przyśpieszenia pracy symulatora uszkodzeń autorzy pracy [PomeR95] wprowadzili do swojego symulatora technikę usuwania uszkodzeń z procesu symulacji nazywaną techniką "fault dropping". Technika ta umożliwiła skuteczne określenie zbioru maskowanych przez rejestry MISR błędów wymuszanych uszkodzeniami s-z-x na wyjściach wzorcowych układów ISCAS 85 oraz 89. Kung z kolegami również opracowali szybki symulator uszkodzeń BISTSIM [KunHL95]. Symulator ten umożliwił kilkukrotne przyśpieszenie symulacji uszkodzeń w porównaniu z symulatorem FSIM opracowanym wcześniej przez Lee oraz Ha [LeeH91]. W pracy [KunHL95] przedstawiono także dwie niezależne procedury szybkiego obliczania sygatur dla rejestrów EED-MISR, które razem z symulatorem BISTSIM umożliwiają dokładne określanie zbioru maskowanych błędów dla projektowanych układów scalonych z wewnątrzukładowym testerem. Autorzy pracy [KunHL95] eksperymentalnie dowiedli przy użyciu wzorcowych układów ISCAS 85, że pomimo wprowadzenia szybkich procedur wyznaczania sygatur czas obliczania sygatur przez te proce-

dury był prawie dwukrotnie wolniejszy od czasu pracy symulatora BISTSIM. Czasochłonność procesu symulacji sygatur zmobilizowała także innych badaczy do poszukiwania szybszych algorytmów. Na przykład w pracy [LambI91] opisano procedurę umożliwiającą bardzo szybką symulację sygatur dla kompaktora liniowego o dowolnej strukturze, natomiast w [Frank95] przedstawiono technikę szybkiej symulacji sygatur dla rejestrów MISR typu CA. Techniki określania sygatur oraz techniki ich konwersji przedstawione w rozdziale 2.3.2 niniejszej pracy również mogą służyć jako podstawa do opracowania algorytmu umożliwiającego szybkie wyznaczanie sygatur.

4.2.5. Parametry dobrze zaprojektowanego kompaktora liniowego

Wszystkie wyniki badań dynamicznego prawdopodobieństwa maskowania błędów wskazują, że nie ma kompaktora liniowego gwarantującego monotoniczne przejście wykresu jego funkcji $P_{a1}(m)$ w kierunku poziomego wykresu statycznego prawdopodobieństwa maskowania błędów. Tak więc schemat n-bitowego kompaktora liniowego powinien być zaprojektowany w taki sposób, ażeby wymusić na krzywej dynamicznego prawdopodobieństwa maskowania błędów $P_{a1}(m)$, aby jak najbardziej przylegała do poziomego wykresu 2^{-n} i ażeby była optymalna z punktu widzenia minimalizacji tego prawdopodobieństwa [Gupta92, OlivD93]. Można się starać zrealizować to przy najmniejszej liczbie m testów wejściowych. Wybór rejestru c-MISR-p(x) związanego z pierwotnym wielomianem charakterystycznym p(x) zapewni najkrótszy czas zaniku pików krzywej $P_{a1}(m)$ i osiągnięcie przez nią poziomu 2^{-n} . Ponieważ wszystkie rejestry c-MISR związane z wielomianem pierwotnym są sobie równoważne, dlatego użyteczny będzie wybór takiego wielomianu pierwotnego, który zapewnia najtańszą realizację struktury sprzężenia liniowego (minimalna liczba bramek XOR), najkrótszy czas propagacji sygnału pomiędzy sąsiednimi przerzutnikami, najmniejszą liczbę połączeń w sprzężeniu zwrotnym oraz najmniejszą obciążalność wyjść przerzutników (rozdział 3.1). Przed przystąpieniem do projektowania rejestru liniowego istotne jest więc nie tylko określenie wielomianu charakterystycznego, ale także ważne jest określenie ww. parametrów. Wymienione parametry projektu rejestru liniowego, a także obecność i rozkład komórek (\oplus 'D) lub T w tym rejestrze ostatecznie decydują o wyborze najlepiej przystającej struktury sprzężenia liniowego do architektury logicznej oraz sieci połączeń oferowanej przez układ scalony, w którym implementowany jest kompaktor liniowy.

Optymalny projekt rejestru c-SISR- $p(x)$ wymaga wyboru wielomianu $p(x)$ z rodziny wielomianów pierwotnych, wielomianów typu $(1+x)g(x)$ i bazujących na kodzie Fire'a. Dodatkowym kryterium wyboru rejestru c-SISR powinny być wymienione wyżej parametry związane z kosztem realizacji sprzętu, szybkością pracy, obciążalnością itp.

Komputerowa symulacja uszkodzeń połączona z szybką symulacją sygatur jest obecnie najlepszym narzędziem umożliwiającym uzyskanie rzeczywistego (dokładnego) wykresu funkcji $P_{al}(m)$. Na tej podstawie można ostatecznie stwierdzić, czy wielomian $p(x)$ - wstępnie dobrany do projektu kompaktora liniowego w oparciu o wcześniej podane obserwacje, wnioski oraz metodę wyznaczania probabilistycznej, przybliżonej funkcji $P_{al}(m)$ - odpowiada najlepiej potrzebom testowania. Wprowadzenie kosztownej metody deterministycznej do optymalnego wyboru wielomianu $p(x)$ zależy od możliwości finansowych producenta decydującego się na wprowadzenie do swoich układów testerów wewnętrznych. Zależy to także od wymagań niezawodnościowych stawianych systemowi komputerowemu, w którym mają być zastosowane te układy. Jeżeli koszty metody deterministycznej są zbyt duże, wówczas producent prawdopodobnie poprzestanie na określaniu przybliżonych probabilistycznych funkcji $P_{al}(m)$ [IvanS91].

Typowe długości n rejestrów c-MISR lub c-SISR mieszczą się mniej więcej w przedziale pomiędzy $n = 16$ a $n = 32$. Wynika z nich, że statyczne prawdopodobieństwo maskowania błędów mieścić się będzie w granicach pomiędzy $2^{-16} \cong 1.5 \times 10^{-5}$ a $2^{-32} \cong 2 \times 10^{-10}$. Prawdopodobieństwo to wpłynie na pomniejszenie współczynnika pokrycia uszkodzeń gwarantowanego przez m testów wejściowych. Zauważmy, że ułamek, o który zostanie pomniejszony ten współczynnik, jest niezwykle mały, szczególnie jeśli go porównać z ułamkiem określającym udział uszkodzeń niewykrywalnych przez wejściową sekwencję testów. W większości zastosowań takie nieznaczne maskowanie błędów jest akceptowane. W przypadku wiarygodnych systemów komputerowych poszukuje się jednak rozwiązań w znaczący sposób pomniejszających maskowania błędów. Dobrym rozwiązaniem jest zastosowanie w takich systemach analizy k-sygnaturowej. Najlepsze rozwiązania powinny jednak gwarantować likwidację zjawiska maskowania błędów.

4.3. Zwiększanie skuteczności analizy sygnaturowej

Konwencjonalna jednosygnaturowa analiza (1SA) mikrokomputerów przedstawiona w pracach [Froh77, HławJ81, HławN83] polega na realizacji jednego pomiaru ciągu danych diagnostycznych w każdym testowanym punkcie, odczytaniu oddzielnych pojedynczych sygatur związanych z testowanymi punktami i porównaniu ich z pojedynczymi sygaturami wzorcowymi. Prawdopodobieństwo statyczne niewykrycia niezależnych błędów w mierzonym ciągu zależy w takiej analizie sygnaturowej od długości n stosowanego rejestru liniowego c-SISR i równe jest 2^{-n} . Podobnie jest w przypadku testerów wewnętrznych wykorzystujących rejestry MISR. Wtedy jednak, w przypadku kiedy liczba n wyprowadzeń TUC podłączonych do rejestru MISR jest zbyt mała, nie uzyskuje się pożądanego niskiego współczynnika maskowania błędów. Najprostszym sposobem zmniejszenia tej wartości jest wydłużenie rejestru liniowego o dodatkowe d komórek. Wówczas statyczne prawdopodobieństwo maskowania błędów $P_{al} = 2^{-(n+d)}$ będzie 2^d razy mniejsze od wartości 2^{-n} . Poważną wadą zastosowania tego rozwiązania w testerach wewnętrznych jest znaczący koszt dodatkowego sprzętu niezbędnego do wydłużenia rejestru. W przypadku dostępnych w handlu autonomicznych analizatorów sygatur 5005B, 5006A [Hewle97] takie wydłużanie rejestru stwarza dodatkowe problemy, co w efekcie czyni ten sposób zmniejszania maskowania błędów całkowicie niepraktycznym rozwiązaniem. Czy w związku z tym istnieje jakaś inna metoda zwiększania skuteczności analizy sygnaturowej?

4.3.1. Zmniejszanie maskowania błędów za pomocą analizy k-sygnaturowej z redundancją czasu

Wielosygnaturową analizę realizują, nie zdając sobie z tego sprawy, technicy w trakcie pomiaru sygatur na wszystkich liniach jednobajtowej magistrali danych systemu mikroprocesorowego w trybie testowania za pomocą swobodnego obiegu przestrzeni adresowej (free run) [HławM93]. Wówczas uszkodzenie np. s-z-0 jednej z linii magistrali adresowej wymusza błędy często na wszystkich ośmiu liniach magistrali danych. Objawić się one mogą w ośmiu sygaturach zmierzonych kolejno na liniach magistrali danych. O takich uszkodzeniach mówi się, że są 8-wykrywalne (ogólnie t-wykrywalne) [Hławi86d, Hławi86e, HławM86, Hławi89g]. Jeżeli część z t sygatur jest równa sygaturom wzorcowym, to odpowiadające im ciągi błędów są maskowane. Jeżeli wszystkie sygatury są identyczne

z sygnaturami wzorcowymi, wówczas uszkodzenie nie zostanie wykryte. Statyczne prawdopodobieństwo maskowania błędów wywołanych uszkodzeniem t-wykrywalnym opisane w [HławM86, Hławi89g, Hławi90e, Hławi93b] określa wyrażenie 2^{-tn} , chociaż w świadomości inżynierów prawdopodobieństwo to równe jest 2^{-n} . Jest ono równoważne statycznemu prawdopodobieństwu maskowania błędów P_{aj} rejestru liniowego o długości $n + d$, gdzie $d = n(t-1)$. W [Froh77, Smith80] nie zauważono tej dodatkowej zależności wartości statycznego prawdopodobieństwa maskowania błędów od uszkodzeń t-wykrywalnych. Większość uszkodzeń pakietów cyfrowych jest t-wykrywalna. Tylko nieduża ich część jest 1-wykrywalna.

Skuteczność lokalizacji uszkodzenia w procesie śledzenia wstecz zależy od tego, na ilu wyjściach każdego układu scalonego umieszczonego w ścieżce propagacji błędnego sygnału (od jego źródła, czyli uszkodzenia) można zmierzyć sygnatury. Jeżeli wzdłuż całej takiej ścieżki na wyjściach każdego z układów scalonych można zmierzyć co najmniej t sygnatur, to wówczas o uszkodzeniu powiemy, że jest t-lokalizowalne. Dzięki nieoczekiwaniu bardzo wysokiej skuteczności analizy sygnaturowej w procesie śledzenia wstecz są właściwie określane miejsca wszystkich uszkodzeń t-lokalizowalnych. Niestety część uszkodzeń jest tylko 1-lokalizowalna. Takie uszkodzenia lokalizowane są ze skutecznością o kilka rzędów mniejszą. Jeżeli podczas takiego procesu diagnostycznego po całej serii np. r niepoprawnych sygnatur następną, tzn. r+1 sygnatura okaże się poprawna, wówczas wskazuje ona, że między miejscem, w którym ją zmierzono, a poprzednim punktem, w którym odczytano niepoprawną sygnaturę, znajduje się uszkodzenie 1-lokalizowalne. Jeżeli jednak okaże się, że wskutek maskowania błędów np. r-1 sygnatura jest poprawna (choć powinna być niepoprawna), wówczas uszkodzenie 1-lokalizowalne zostanie niewłaściwie umiejscowione.

W celu zwiększenia skuteczności umiejscawiania uszkodzeń 1-lokalizowalnych oraz wykrywania uszkodzeń 1-wykrywalnych wprowadzono w [HławK81a] analizę wielosygnaturową. Polega ona na realizacji kolejno kilku (k) pomiarów różnych ciągów tych samych danych diagnostycznych w punkcie związanym z obserwacją błędów wymuszonych uszkodzeniami 1-lokalizowalnymi. Konkatenacja k sygnatur uzyskanych w tych pomiarach może być traktowana jako jedna sygnatura złożona z kn bitów. Sygnatura ta jest następnie porównywana z konkatenacją k wzorcowych n-bitowych sygnatur. Technika ta pod nazwą k-sygnaturowa analiza (kSA) została po raz pierwszy na świecie przedstawiona w pracach [HławK81a, HławK81b (patent)], następnie powtórzona trzy lata później w pracy

[HassM84a] pod nazwą technika MSA (Multiple Signature Analysis). Technika analizy k-sygnaturowej może być zarówno stosowana w zewnętrznej analizie sygnaturowej, jak również może być wprowadzana do testerów wewnętrznych z kompaktorem w postaci rejestrów s-SISR lub c-MISR.

Ograniczmy na początku naszą uwagę do analizy 2SA realizowanej przy użyciu n-stopniowego rejestru c-SISR- $p_1(x)$. Załóżmy, związany z pierwszym pomiarem, wielomian błędów $e_1(x) = e(x)$ oraz wielomian błędów $e_2(x) = e^+(x)$ związany z drugim pomiarem, w którym testy podawane są w odwrotnej kolejności niż podczas pomiaru pierwszego. Tę technikę analizy 2SA wyjaśnia poniższy przykład.

Przykład 4.7

Założmy, że rejestr IED-SISR- $p_1(x)$ związany jest z wielomianem pierwotnym $p_1(x) = 1 + x + x^2 + x^4 + x^5$. Załóżmy, że w pierwszym pomiarze na wejście rejestru SISR wprowadzany jest ciąg zawierający następujący wielomian błędów

$$e(x) = 1 + x + x^2 + x^3 + x^7 + x^8 + x^9 + x^{12} + x^{13} + x^{15} = \\ = (1 + x)^3(1 + x + x^2 + x^4 + x^5)(1 + x + x^3 + x^5 + x^7)$$

natomiast w drugim pomiarze wprowadzany jest ciąg odwrotny z odwróconym wielomianem błędów

$$e^+(x) = 1 + x^2 + x^3 + x^6 + x^7 + x^8 + x^{12} + x^{13} + x^{14} + x^{15} = \\ = (1 + x)^3(1 + x + x^3 + x^4 + x^5)(1 + x^2 + x^4 + x^6 + x^7).$$

Wielomian błędów $e(x)$ jest podzielny przez wielomian $p_1(x)$, natomiast wielomian błędów $e^+(x)$ jest niepodzielny przez $p_1(x)$. Oznacza to, że w pierwszym pomiarze błędy nie zostaną wykryte, natomiast w drugim pomiarze ich detekcja będzie skuteczna.

Wielomiany błędów niewykrywane w obu pomiarach przyjmują postaci $e_1(x) = q_1(x) p_1(x)$ oraz $e_2(x) = q_2(x) p_1(x)$. W związku z tym, że $e_2^+(x) = q_2^+(x) p_1^+(x)$, otrzymujemy $e_2^+(x) = (e^+(x))^+ = e(x) = q_2^+(x) p_1^+(x)$ oraz $e(x) = q_1(x) p_1(x)$. Na tej podstawie w [HławK81b] udowodniono, że skuteczność takiej dwusygnaturowej analizy można badać przy użyciu wyrażenia

$$e(x) / p_1(x) p_1^+(x) = q(x) + r(x) / p_1(x) p_1^+(x) \quad (4.6)$$

opisującego proces kompaktacji liniowej podczas jednego pomiaru sygnatury za pomocą rejestru c-SISR- $p(x)$ związanego z wielomianem charakterystycznym $p(x) = p_1(x) p_1^+(x)$. Jest to nieprawda w przypadku symetrycznych ciągów błędów zależnych charakterystycz-

nych dla opisanych w [Hławi86a] uszkodzeń typu sklejenie w kombinacyjnych układach liniowych. W przypadku pojawienia się takich błędów w obu pomiarach dzielony jest identyczny wielomian błędów $e(x) = e^+(x)$. Szeroko ten problem omówiono w pracy [Hławi86a].

Innym sposobem analizy k-sygnaturowej jest pomiar k sygnatur za pomocą k rejestrów c-SISR każdy o innym liniowym sprzężeniu zwrotnym [Hławn84]. Skuteczność dwusygnaturowej analizy przy użyciu rejestru c-SISR związanego w pierwszym pomiarze z wielomianem pierwotnym $p_1(x)$, natomiast w drugim pomiarze z wielomianem pierwotnym $p_2(x)$ można badać przy użyciu wyrażenia

$$e(x) / p_1(x) p_2(x) = q(x) + r(x) / p_1(x) p_2(x) \quad (4.7)$$

opisującego proces kompaktacji liniowej podczas jednego pomiaru sygnatury za pomocą rejestru c-SISR- $p_1(x)p_2(x)$. Wyjaśnia to przykład.

Przykład 4.8

Wprowadźmy w przykładzie 4.7 drugi pomiar realizowany za pomocą rejestru IED-SISR- $p_2(x)$ związanego z wielomianem $p_2(x) = 1 + x + x^3 + x^4 + x^5$. Wielomian $e(x) = 1 + x + x^2 + x^3 + x^7 + x^8 + x^9 + x^{12} + x^{13} + x^{15} = (1 + x)^3(1 + x + x^2 + x^4 + x^5)(1 + x + x^3 + x^5 + x^7)$ zgodnie z przykładem 4.7 jest podzielny przez wielomian $p_1(x)$, natomiast nie jest on podzielny przez wielomian $p_2(x)$. W efekcie błędy zostaną wykryte za pomocą sygnatury uzyskanej w drugim pomiarze. Ten sam rezultat osiąga się za pomocą kompaktacji liniowej zrealizowanej w rejestrze c-SISR- $p_1(x)p_2(x)$. Ilustruje to poniższe dzielenie, którego reszta jest różna od zera.

$$\frac{(1+x)^3(1+x+x^2+x^4+x^5)(1+x+x^3+x^5+x^7)}{(1+x+x^2+x^4+x^5)(1+x+x^3+x^4+x^5)}$$

Na podstawie wniosku 4.3 statyczne prawdopodobieństwo maskowania błędów P_{al} przy wykorzystywaniu opisanych wyżej dwóch technik 2SA można określić wyrażeniem 2^{-2n} [Hławk81a, Hławk81b]. Skuteczność analizy kSA przy wykorzystaniu k różnych wielomianów pierwotnych $p_i(x)$; $i = 1, 2, \dots, k$ można badać za pomocą wyrażenia ilustrującego proces kompaktacji za pomocą rejestru c-SISR- $p_1(x)p_2(x) \dots p_1(x) \dots p_k(x)$. W tym przypadku statyczne prawdopodobieństwo niewykrycia błędów $P_{al} = 2^{-kn}$. Ostatecznie w oparciu o [Hławk81a, BhavK84] można określić następujący wniosek.

Wniosek 4.10

Statyczne prawdopodobieństwo niewykrycia jednakowo prawdopodobnych ciągów błędów w trakcie analizy kSA, realizowanej za pomocą n-stopniowego rejestru c-SISR przy użyciu k różnych sprzężeń liniowych bądź k różnych ciągów testów, jest równe 2^{-kn} .

W każdej z opisanych technik analizy kSA zamiast dużego nadmiaru sprzętowego wprowadzona jest redundancja czasu związana z pomiarem dodatkowych k-1 sygnatur. Nie ma ona znaczenia wszędzie tam, gdzie czas testowania nie jest istotny, np. podczas serwisu systemów cyfrowych o niezawodności czy też dyspozycyjności, do której nie przywiązuje się większej wagi. Praktycznym przykładem takiego zastosowania analizy dwusygnaturowej jest serwisowy analizator sygnatur PAS80 [Hławn84, Hławn85a, Hławn85b] wyposażony w 16-stopniowy rejestr EED-SISR posiadający dwa różne sprzężenia liniowe związane odpowiednio z wielomianem pierwotnym $p_1(x) = 1 + x^4 + x^7 + x^9 + x^{16}$ oraz z wielomianem pierwotnym $p_2(x) = 1 + x^4 + x^7 + x^{10} + x^{16}$. Sprzężenia te można zmieniać z pulpitu operatora ww. analizatorów sygnatur. Statyczne prawdopodobieństwo maskowania przez te analizatory błędów zmniejsza się o pięć rzędów przechodząc od wartości 1.5×10^{-5} po pierwszym pomiarze do wartości 2×10^{-10} po drugim pomiarze. Innym przykładem takiego zastosowania analizy 2SA jest weryfikator sygnatur WAS80 [Hławi88b]. Opisana technika analizy k-sygnaturowej może być także wprowadzana do testerów wewnętrznych z kompaktorami w postaci rejestrów c-SISR.

Badaniem skuteczności k-sygnaturowej analizy przy wykorzystaniu równoległych kompaktorów liniowych zajmowano się w pracach [BhavK84, Hławi86a, Hławi86b]. W pracy [BhavK84] ograniczono się jedynie do badania skuteczności wykrywania macierzy błędów niezależnych (jednakowo prawdopodobnych). W pracach [Hławi86a, Hławi86b] do zbadania skuteczności wykorzystano pierścień wielomianów nad ciałem GF(2). Umożliwiło to zbadanie wpływu różnych struktur sprzężeń liniowych rejestrów c-MISR na skuteczność analizy k-sygnaturowej.

Zauważmy, że wielomian błędów $e_{IEDf}(x)$, ilustrujący skompresowaną w przetworniku $P1_{IED}$ (rys. 2.7) macierz błędów \mathbf{E}_f , jest jednakowy w przypadku każdego z k pomiarów. Wynika to z faktu, że wielomian $p_{IEDr}(x)$ jest stały (x^r) niezależnie od rodzaju wielomianu $p_i(x)$. Tej właściwości nie posiadają rejestry c-MISR o strukturach innych niż struktura IED. Na tej podstawie w pracach [Hławi86a, Hławi86b] udowodniono, że skuteczność

detekcji macierzy błędów \mathbf{E}_f za pomocą jednosygnaturowej analizy przy użyciu rejestru IED-MISR- $p(x)$ związanego z wielomianem charakterystycznym

$p(x) = p_1(x)p_2(x)\dots p_i(x)\dots p_k(x)$ stopnia kn (reprezentowanej dzieleniem wielomianu błędów $e_{IEDf}(x)$ przez $p(x)$) jest identyczna ze skutecznością analizy k -sygnaturowej realizowanej za pomocą rejestru IED-MISR związanego w każdym z k pomiarów z innym pierwotnym wielomianem charakterystycznym $p_i(x)$ stopnia n ze zbioru $\{p_1(x), p_2(x), \dots, p_i(x), \dots, p_k(x)\}$. W pracach [Hławi86a, Hławi86b] udowodniono również, że analiza k -sygnaturowa za pomocą n -bitowego rejestru IED-MISR z użyciem k jego różnych sprzężeń liniowych wykrywa każdą paczkę błędnych wektorów $\mathbf{E}(kn, t)$. Ten ostatni rezultat jest szczególnie atrakcyjny w przypadku stosowania analizy k -sygnaturowej w testerach wewnętrznych układowych typu STUMPS.

W pracach [Hławi86a, Hławi86b, HławN89] zajmowano się także, realizowaną przy użyciu n -bitowego rejestru IED-MISR- $p(x)$, analizą 2SA z użyciem dwóch różnych macierzy odpowiedzi TUC wymuszanych przez generator testów pracujący w przód, a następnie wstecz. W pracy [Hławi86b] pokazano sposób wymuszania obiegu przestrzeni adresowej mikroprocesora w kierunku odwrotnym do normalnie wymuszanego przez licznik instrukcji (IR). Zrealizowano to za pomocą umieszczenia na liniach magistrali adresowej bloku bramek XOR pełniących rolę sterowanych negacji. Tę technikę praktycznie zastosowano do realizacji analizy 2SA w trybie testowania "free run" w samotestowalnym mikrokomputerowym urządzeniu pomiarowym CRISTALDIGRAF NC-ST [HławN89]. W pracach [Hławi86a, Hławi86b] udowodniono, że skuteczność takiej dwusygnaturowej analizy w przypadku niesymetrycznych wielomianów $e_{IEDf}(x)$ można badać przy użyciu wyrażenia

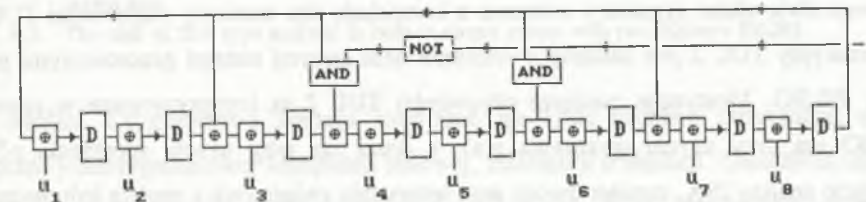
$$e_{IEDf}(x) / p(x) p^*(x) = q(x) + r(x) / p(x) p^*(x) \quad (4.8)$$

Oczywiście statyczne prawdopodobieństwo maskowania błędów $P_{al} = 2^{-2n}$. W ww. pracach pokazano również, że omawiana analiza dwusygnaturowa wykrywa każdą paczkę błędnych wektorów $\mathbf{E}(2n, t)$. Przedstawionych powyżej właściwości nie posiadają rejestry c-MISR o strukturach innych niż struktura IED [Hławi86b]. Tym niemniej na podstawie prac [BhavK84, Hławi86a, Hławi86b] można sformułować następujący wniosek dotyczący rejestrów c-MISR o wszystkich wcześniej zdefiniowanych strukturach:

Wniosek 4.11

Statyczne prawdopodobieństwo niewykrycia macierzy błędów \mathbf{E} w trakcie analizy kSA realizowanej za pomocą n -stopniowego rejestru c-MISR i przy użyciu k różnych ciągów testów bądź k różnych sprzężeń liniowych związanych z różnymi wielomianami pierwotnymi jest równe 2^{-kn} .

Rejestr IED-MISR związany z k różnymi wielomianami charakterystycznymi $p_i(x)$; $i = 1, 2, \dots, k$ umożliwiającymi k -sygnaturową analizę nazwano w pracy [Hławi89e] rejestrem kMISR. W pracy tej podano sposób doboru k wielomianów pierwotnych dla rejestru kMISR umożliwiający uzyskanie minimalnej redundancji sprzętowej związanej z realizacją analizy kSA. Przykład 8-stopniowego rejestru 2MISR z dwoma dobranymi tym sposobem wielomianami pierwotnymi $p_1(x) = 1 + x^2 + x^3 + x^6 + x^8$ oraz $p_2(x) = 1 + x^2 + x^5 + x^6 + x^8$ przedstawia rys. 4.4. Wielomiany te, różniąc się tylko na jednej pozycji, umożliwiają ograniczenie dodatkowego kosztu związanego z włączaniem drugiego sprzężenia liniowego ($p_2(x)$) do dwóch bramek AND, jednej negacji oraz jednej bramki XOR. Przykładem zastosowania tego rejestru 2MISR jest analizator sygnatur wbudowany w samotestowalne mikrokomputerowe urządzenie pomiarowe CRISTALDIGRAF NC-ST [HławN89].



Rys. 4.4. Schemat ideowy 8-stopniowego rejestru 2MISR
Fig. 4.4. Scheme of register 2MISR with 8 stages

Zakładając, że przedstawiony na rys. 4.4 rejestr 2MISR posiada dodatkowy sprzęt umożliwiający przekształcanie go w rejestr przesuwający z wejściem SI oraz wyjściem SO, można przedstawić następujący scenariusz jego pracy w trakcie analizy 2-sygnaturowej. Przed kompaktacją odpowiedzi TUC przy użyciu sprzężenia związanego z wielomianem $p_1(x)$ do rejestru 2MISR zostaje wprowadzony poprzez wejście SI jego stan początkowy, a na wejściu c sterującym przełączaniem sprzężenia liniowego podana zostaje wartość 0. Po pierwszej kompaktacji stan rejestru w postaci sygnatury s_1 zostaje wysunięty na zewnątrz

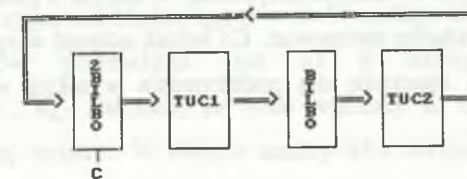
poprzez wyjście SO do zewnętrznego komparatora sygnatur. Równocześnie z wyprowadzaniem na zewnątrz sygnatury zostaje wprowadzony do rejestru 2MISR nowy stan początkowy, a na wejście c zostaje podany stan 1 włączający nowe sprzężenie liniowe związane z wielomianem $p_2(x)$. Po zakończeniu drugiej kompaktacji sygnatura s_2 zostaje wysunięta do zewnętrznego kompaktora.

Podobną minimalną redundancję sprzętową można uzyskać projektując rejestr 2LFSR wyposażony w dwa sprzężenia liniowe związane z różniącymi się na jednej pozycji wielomianami $p(x)$ oraz $p^+(x)$. Rejestr 2LFSR umożliwia generację dwóch wzajemnie odwrotnych ciągów testów pseudoprzypadkowych [Hławi86a, Hławi86b].

Rejestry kMISR (kLFSR) można także wykorzystać do budowy rejestrów kBILBO (kMBILBO). Zaczerpnięty z prac [Hławi86a, Hławi86b] przykład zastosowania tak zmodyfikowanego rejestru 2BILBO do budowy klasycznego testera wewnątrzukładowego umożliwiającego realizację analizy 2SA przedstawia rys. 4.5. Rejestr 2BILBO posiada dwa różne sprzężenia zwrotne związane z wielomianami $p(x)$ oraz $p^+(x)$, które można wybierać za pomocą zewnętrznego sygnału c. Natomiast rejestr BILBO jest wykonany tradycyjnie i posiada tylko jedno sprzężenie zwrotne. Rejestr 2BILBO generuje dwukrotnie testy dla TUC 1. Pierwszy raz przy sprzężeniu $p(x)$, a drugi raz w odwrotnej kolejności przy sprzężeniu $p^+(x)$. Rejestr BILBO pozwala zmierzyć przy użyciu tego samego sprzężenia liniowego dwie różne sygnatury związane z kompaktacją obu macierzy odpowiedzi TUC 1. Kombinacyjny TUC 2 jest zasilany dwukrotnie tymi samymi testami generowanymi przez rejestr BILBO. Identyczne macierze odpowiedzi TUC 2 są komprimowane w rejestrze 2BILBO raz przy użyciu sprzężenia $p(x)$, a drugi raz przy użyciu sprzężenia $p^+(x)$. W efekcie analiza 2SA, zamiast dwóch sesji testowania związanych z analizą jednosygnaturową, wymaga czterech różnych sesji testowania, ale tylko jednego rejestru 2MISR zawartego w rejestrze 2BILBO. Liczbę sesji można ograniczyć do dwóch, jeżeli stany rejestrów 2MISR w 2BILBO oraz rejestru MISR w BILBO będą w trakcie kompaktacji podawane jako testy pseudoprzypadkowe odpowiednio do TUC 1 oraz TUC 2.

Pewnym utrudnieniem w analizie kSA jest konieczność wyprowadzania do zewnętrznego komparatora k różnych sygnatur oraz konieczność pamiętania k różnych sygnatur wzorcowych. W celu wyeliminowania tego kłopotu w pracach [Hławi87b, Hławi88c, Hławi89e] zaproponowano konstrukcję specjalnego k-sygnaturowego weryfikatora BIKEV (Built-in Evaluator Using Multiple (k) Compaction) zawierającego n-stopniowy rejestr kMISR oraz

n-wejściową bramkę AND (OR) podłączoną do wszystkich n wyjść przerzutników rejestru kMISR i dekodującą stan 111...11 (000...00) tego rejestru. Wspomniana bramka AND może być traktowana jako komparator, jeżeli założymy, że przed każdym z k procesów kompaktacji wprowadzany jest do rejestru kMISR taki stan początkowy, że stan końcowy (sygnatura) po każdej z k kompaktacji będzie równy stanowi 111...11. Wówczas wyjście bramki AND w każdym z k momentów dekodowania takiej sygnatury będzie sygnalizowało stanem 1, że nie ma błędów w macierzy odpowiedzi TUC. Technika obliczania stanu początkowego przy założeniu znajomości stanu końcowego rejestru MISR oraz znajomości macierzy odpowiedzi TUC opracowano w [Hławi87a, Hławi88a]. Dokładnie ją opisano w rozdziale 2.3.3 i przykładzie 2.9. Technika tę można wykorzystać do określania k różnych stanów początkowych, które wprowadzane do rejestru kMISR w weryfikatorze BIKEV ułatwią proces weryfikacji k-sygnatur.



Rys. 4.5. Schemat blokowy analizy 2SA w testerach wewnątrzukładowych z dwoma rejestrami BILBO (MBILBO)

Fig. 4.5. The idea of 2SA type analysis in built-in circuit testers with two registers BILBO

Analiza k-sygnaturowa pomaga rozwiązać nie tylko problem odzyskania, traconej podczas jednosygnaturowej kompaktacji liniowej, informacji o błędach. Umożliwia ona także zwiększanie rozdzielczości uszkodzeń, która w przypadku analizy jednosygnaturowej często okazuje się zbyt mała. W szczególności konieczność rozwiązania tego problemu wystąpi w przypadku procesu diagnostycznego wykorzystującego słownik sygnatur charakterystycznych dla lokalizowanych uszkodzeń. Wówczas uszkodzenia, chociaż wykrywalne przez kompaktor liniowy i całkowicie rozróżnialne na podstawie różnic w błędach zależnych występujących w komprimowanych macierzach błędów, stają się nierozróżnialne wskutek maskowania tych różnic w kompaktorze. Jakość diagnostyki przy użyciu analizy sygnaturowej zależy od tego, jak duża liczba różnych sygnatur może być generowana przy danym zbiorze uszkodzeń (liczność tego zbioru) i przy danej długości n kompaktora liniowego [Hławi87c, Hławi88b, RajsT90, RajsT91b]. Problemem zwiększania rozdzielczości

diagnostycznej analizy sygnaturowej zajmowano się w pracach [Hławi87c, Hławi88b]. Do rozwiązania tego problemu zastosowano analizę k-sygnaturową. W pracach tych wykorzystano prawdopodobieństwo rozróżnialności zdefiniowane w [David78, David80] do określenia rozróżnialności uszkodzeń w analizie k-sygnaturowej. Za pomocą tego aparatu matematycznego udowodniono, że przy liczbie 2^n możliwych uszkodzeń i k-sygnaturowej analizie z użyciem n-bitowego kompaktora liniowego i przy założeniu $n \geq 8$ i $k \geq 3$, prawdopodobieństwo rozróżnienia tych uszkodzeń jest prawie równe 1. W pracach tych przedstawiono ponadto techniki wykorzystania wbudowanych weryfikatorów BIKEV do lokalizacji przy użyciu analizy kSA uszkodzonych na płycie krzemowej struktur, uszkodzonych na pakiecie cyfrowym układów scalonych, a także lokalizacji uszkodzonych połączeń różnych układów ASIC umieszczonych na pakiecie cyfrowym. Podstawową wadą tych technik analizy k-sygnaturowej jest k-krotne wydłużenie czasu testowania. Nie ma to jednak większego znaczenia w każdym przypadku, w którym z punktu widzenia ekonomicznego opłaca się taką technikę zastosować. Co jednak uczynić w sytuacji, gdy krótki czas testowania ma żywotne znaczenie dla podtrzymania wysokiej wiarygodności systemu komputerowego?

4.3.2. Zmniejszanie maskowania błędów za pomocą analizy k-sygnaturowej bez redundancji czasu

Jeżeli czas testowania ma istotne znaczenie, wówczas omówione poprzednio techniki analizy k-sygnaturowej można zastąpić pewną odmianą analizy k-sygnaturowej. Jest nią technika odczytu w trakcie jednej sesji testowania k sygnatur pojawiających się sukcesywnie w rejestrze c-MISR w trakcie procesu kompaktacji, po określonych wcześniej k różnych liczbach taktów zegarowych. Oznacza to, że macierz wektorów na wyjściu TUC jest abstrakcyjnie dzielona na k segmentów. Oznacza to również, że sygnatura uzyskana po kompaktacji j takich segmentów traktowana jest jako stan początkowy rejestru c-MISR przed kompaktacją j + 1 segmentu. Tak więc wszystkie sygnatury otrzymuje się w normalnym czasie przeznaczonym na wygenerowanie całej macierzy odpowiedzi TUC. Technika ta opisana w [BhavK84, Hławi90e, Hławi91a] umożliwia realizację analizy k-sygnaturowej bez redundancji czasu. Wymaga ona nieco większych nakładów sprzętowych [Hławi90e, Hławi91a] niż w przypadku analizy kSA z redundancją czasu. Prawdopodobieństwo masko-

wania błędów na podstawie prac [BhavK84, Hławi90e, Hławi91a, Youn91] określa następujący wniosek:

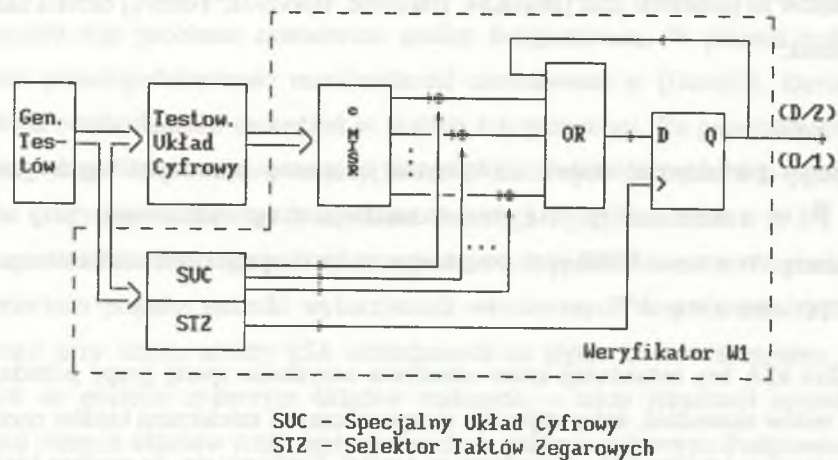
Wniosek 4.12

Statyczne prawdopodobieństwo niewykrycia jednakowo prawdopodobnych macierzy błędów \mathbf{E} w trakcie analizy kSA bez redundancji czasu realizowanej przy użyciu n-stopniowego rejestru c-MISR-p(x) związanego z charakterystycznym wielomianem pierwotnym p(x) jest równe 2^{-kn} .

Analiza kSA bez redundancji czasu umożliwia odzyskanie sporej grupy pobudzonych ciągami testów uszkodzeń, które objawiają się specyficznymi macierzami błędów normalnie maskowanymi w analizie 1SA. Wyjaśnimy to poniżej. Umówmy się, że macierz błędów \mathbf{E} jest maskowana w analizie jednosygnaturowej. Oznacza to, że po jej kompaktacji w rejestrze c-MISR otrzymujemy sygnaturę błędów $s_{cc}(x) = 0$. W analizie kSA bez redundancji czasu ta macierz błędów podzielona jest na k następujących segmentów $\mathbf{E}_1, \mathbf{E}_2, \dots, \mathbf{E}_i, \dots, \mathbf{E}_j, \dots, \mathbf{E}_k$. Załóżmy, że tylko segmenty \mathbf{E}_i oraz \mathbf{E}_j zawierają błędy. Pozostałe segmenty są zerowe. W efekcie analizy kSA otrzymujemy k następujących sygnatur błędów $s_{ce1}(x), s_{ce2}(x), \dots, s_{cei}(x), \dots, s_{cej}(x), \dots, s_{cek}(x)$. Wszystkie sygnatury związane z segmentami od 1 do i-1 są równe zero. Pierwszą różną od zera jest sygnatura $s_{cei}(x)$. Również następne są różne od zera. Kolejna zerowa sygnatura błędów pojawi się po kompaktacji segmentu $\mathbf{E}_j \neq 0$. Oznacza to, że błędy zawarte w tym segmencie skasowały w trakcie kompaktacji błędy pochodzące z segmentu \mathbf{E}_i i zawarte w sygnaturze powstałej po kompaktacji segmentu $\mathbf{E}_{j-1} = 0$.

W dalszym ciągu niewykrywane są te pobudzone testami uszkodzenia, które wymuszają takie macierze błędów \mathbf{E} , że każdy z jej k segmentów $\mathbf{E}_1, \mathbf{E}_2, \dots, \mathbf{E}_i, \dots, \mathbf{E}_j, \dots, \mathbf{E}_k$ komprimowany jest, niezależnie od innych segmentów, w zerową sygnaturę błędów.

Przedstawiony na rys. 4.6 weryfikator W1 umożliwia realizację techniki analizy k-sygnaturowej bez redundancji czasu. Weryfikator ten zawiera n-bitowy kompaktor c-MISR. Ponadto zawiera on selektor taktów zegarowych STZ, który wśród m taktów zegarowych niezbędnych do skomprimowania m wektorów macierzy odpowiedzi TUC wybiera k takich taktów, po których odczytywane są sygnatury. W ten sposób m-wektorowa macierz zostaje rozdzielona na k segmentów zawierających odpowiednio $m_1, m_2, \dots, m_i, \dots, m_k$ wektorów ($m = m_1 + m_2 + \dots + m_i + \dots + m_k$). Jeżeli długość



Rys. 4.6. Schemat blokowy weryfikatora W1 z kompaktorem do analizy k-sygnaturowej bez redundancji czasu

Fig. 4.6. Scheme of verifier W1 with compactor for k signature analysis without time redundancy

każdego z k segmentów będzie identyczna, wówczas układ STZ można zrealizować np. w postaci licznika liniowego opisanego w [Hławi97a]. Każde j kolejnych segmentów zostaje zakodowane w oddzielnej sygnaturze. Weryfikator W1 zawiera także specjalny układ cyfrowy SUC, który pobudzony przez generator testów, w odpowiednich k taktach zegarowych, wytwarza na swoim wyjściu k wzorcowych sygnatur. Sygnatury te są sukcesywnie porównywane w komparatorze (n bramek XOR) z kolejno mierzonymi sygnaturami. Jedynka na wyjściu bramki OR oznacza niezgodność sygnatur, którą wyłapuje specjalna pułapka złożona z przerzutnika D wyzwalanego zboczem. Czas weryfikacji każdej z k sygnatur równy jest zero. Skuteczność kompaktora zawartego w weryfikatorze W1 można określić w oparciu o 4.1b za pomocą następującego wyrażenia:

$$EF_1 = 1 - \frac{(2^{mn-kn} - 1)}{(2^{mn} - 1)} \quad (4.9)$$

które dla $m \gg n$ przyjmuje postać $EF_1 = 1 - 2^{-kn}$. Możliwe do uzyskania wartości skuteczności EF zawarte są w przedziale:

$$1 - 2^{-n} \leq EF_1 \leq 1 \text{ gdzie: } EF_1 = 1 - 2^{-kn} \text{ dla } k = 1, 2, \dots, m$$

Ze względu na to, że rejestr c-MISR-p(x) związany z wielomianem pierwotnym wykrywa każdą paczkę błędnych wektorów $\mathbb{E}(n, t)$, można, przy założeniu pomiaru m sygnatur ($k = m$), uzyskać całkowitą likwidację zjawiska maskowania błędów. Zwiększanie

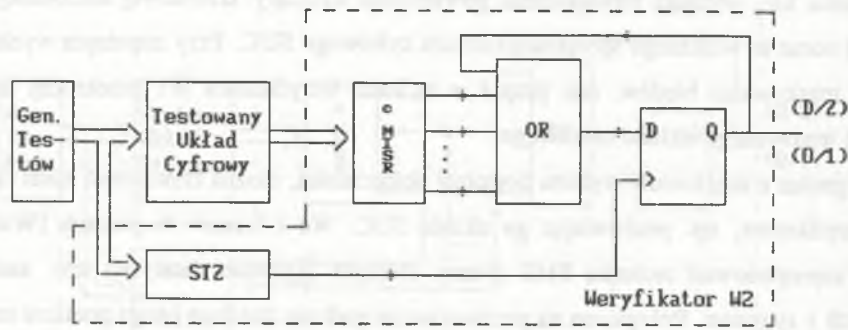
skuteczności EF_1 wymaga powiększenia powierzchni struktury krzemowej niezbędnej do realizacji coraz to większego specjalnego układu cyfrowego SUC. Przy pełnym wyeliminowaniu maskowania błędów, tzn. przy $k = m$ koszt weryfikatora W1 przekroczy koszt realizacji testowanego układu cyfrowego.

Rezygnując z możliwości wyboru dowolnej skuteczności, można zmniejszyć koszt realizacji weryfikatora, np. pozbawiając go układu SUC. Wu i Ivanow w pracach [WuI92, WuI95] zaproponowali technikę FMS (Fuzzy Multiple Signature Analysis) tzw. analizy rozmytych k sygnatur. Polega ona na porównywaniu podczas każdego i-tego pomiaru zmierzonej i-tej sygnatury ze wszystkimi k wzorcowymi sygnaturami. Technika takiego pomiaru jest identyczna z techniką analizy sygnaturowej bez korelacji stosowaną od dawna w zewnętrznych weryfikatorach sygnatur 1310A [Robin85] oraz WAS80 [Hławi88b]. Różnica polega jedynie na tym, że zamiast np. pamięci PROM zawierającej k wzorcowych sygnatur Wu oraz Ivanov zastosowali jednowyjściowy układ kombinacyjny realizujący funkcję w postaci zminimalizowanej sumy logicznej k implikantów odzwierciedlających postaci k wzorcowych sygnatur i zawierających n zmiennych wyjściowych n-stopniowego rejestru c-MISR. Układ ten podłączony do n wyjść rejestru c-MISR sygnalizuje zgodność zmierzonej i-tej sygnatury z sygnaturą wzorcową. Niestety sygnalizuje on taką zgodność również w sytuacji, w której zmierzona i-ta sygnatura równa jest wzorcowej j-tej sygnaturze skorelowanej z innym j-tym taktem pomiarowym. Weryfikator W1 wskazuje w takim przypadku niezgodność sygnatur. Tak więc technika analizy rozmytych k sygnatur na pewno posiada mniejszą skuteczność niż weryfikator W1. Skuteczność tę ze względu na podobieństwo techniki FMS do techniki stosowanej w zewnętrznych weryfikatorach sygnatur określono już wcześniej w pracach [HławiM86, Hławi88b, Hławi89g]. Określa ją następujące wyrażenie

$1 - v^l 2^{-ln}$, gdzie v jest liczbą wzorcowych sygnatur, natomiast l oznacza liczbę pomiarów l sygnatur. Zakładając, że $v = l = k$ otrzymujemy skuteczność

$$EF_{FMS} = 1 - k^k 2^{-kn} \quad (4.10)$$

Jest ona identyczna z uzyskaną w [WuI92, WuI95]. Przy założeniu $n = 16$ oraz $k = 4$ skuteczność weryfikatora W1 określa wyrażenie $EF_1 = (1 - 2^{-64})$. Skuteczność techniki FMS przy tych samych założeniach określa wyrażenie $EF_{FMS} = (1 - 4^4 2^{-64}) = (1 - 2^{-56})$. Tak więc prawdopodobieństwo maskowania błędów w przypadku weryfikatora W1 jest 2^8 razy mniejsze niż w przypadku techniki analizy rozmytych k sygnatur.



STZ - Selektor Taktów Zegarowych

Rys. 4.7. Schemat blokowy weryfikatora W2 umożliwiający analizę k identycznych sygnatur
 Fig. 4.7. Scheme of verifier W2 enabling the analysis of k identical signatures

Wady tej nie posiada schemat weryfikatora W2 przedstawiony na rys. 4.7 i po raz pierwszy opisany w pracach [Hławi90e, Hławi91a], a następnie w pracach [Hławi93b, WuI93a, WuI93b]. Nie zawiera on również bloku n bramek XOR oraz specjalnego układu cyfrowego SUC generującego różne wzorcowe sygnatury. Weryfikator W2 w wybranych przez selektor STZ taktach zegarowych dokonuje pomiaru k identycznych sygnatur, które porównuje zawsze z tą samą sygnaturą wzorcową. Zmieniając stan początkowy rejestru MISR i/lub jego sprzężenie zwrotne, można tak dobrać mierzoną k razy sygnaturę, że liczba k określająca krotność jej wystąpienia będzie większa od jedności. Przy optymalnym wyborze stanu początkowego rejestru MISR i jego sprzężenia można wybrać taką sygnaturę, która pozwala uzyskać maksymalną, możliwą do otrzymania tą metodą, wartość liczby $k = k_{\max}$ gwarantującą uzyskanie stosunkowo dużej skuteczności

$$EF_2 = 1 - 2^{-nk_{\max}} \quad (4.11)$$

Optymalny wybór powinien także zapewniać rozdzielanie macierzy odpowiedzi TUC na k segmentów o identycznej długości, co pozwala zastosować układ STZ w postaci prostego licznika liniowego [Hławi97a]. Taki optymalny wybór może być realizowany za pomocą specjalnych pakietów programów, jakich użyto w pracy [WuI93a]. W wyborze tym może być także pomocna możliwość zmiany struktury sprzężenia liniowego rejestru c-MISR związanego z wielomianem charakterystycznym $p(x)$. W pracy [WuI93b] do zmiany sygnatur uzyskiwanych w rejestrze MISR zastosowano generator testów LFSR- $p(x)$. Specjalny program opracowany przez Wu oraz Ivanova optymalizuje wybór k identycznych

sygnatur poprzez poszukiwanie właściwego stanu początkowego rejestru LFSR i odpowiedniego wielomianu charakterystycznego $p(x)$ związanego z jego sprzężeniem. Wadą tej metody jest możliwość wystąpienia sprzeczności pomiędzy stanem początkowym (seed) wybieranym w celu skrócenia ciągu testów pseudoprzypadkowych a stanem początkowym umożliwiającym optymalny wybór k identycznych sygnatur. Najczęściej są to dwa różne stany. Który z nich wybrać, to znany problem tzw. "krótkiej koldry". Inną wadą weryfikatora W2 jest zbyt mała liczba k w stosunku do możliwości, jakie daje długość m macierzy odpowiedzi TUC. Liczba k identycznych i równomiernie rozłożonych w czasie sygnatur zależy od długości testu generowanego przez c-LFSR- $p(x)$. Zwiększanie tej liczby wpływa na niepożądane wydłużanie czasu testowania.

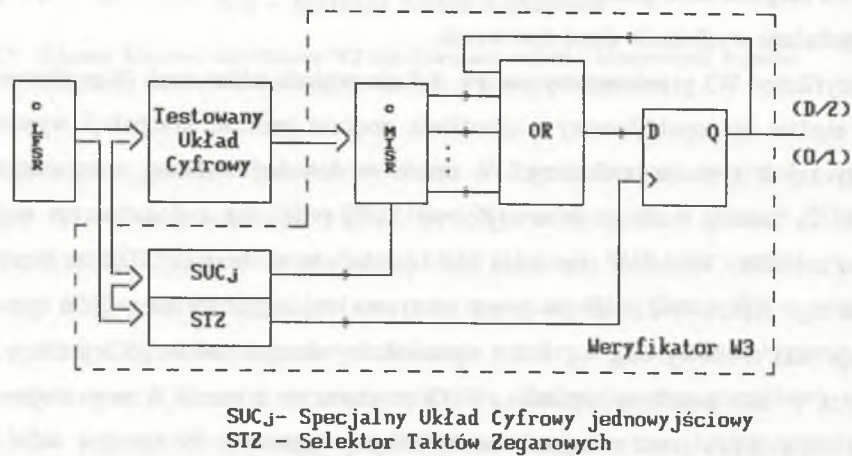
Weryfikator W3 przedstawiony na rys. 4.8 nie posiada takich wad. Weryfikator ten - dotąd nigdzie nie opublikowany - umożliwia podczas procesu kompaktacji wymuszanie k identycznych sygnatur rozłożonych w czasie w dowolny wybrany przez projektanta sposób. To zadanie realizuje jednowyjściowy SUCj połączony z dodatkowym wejściem rejestru c-MISR. Wejściem tym może być np. dodatkowa bramka XOR na pierwszym stopniu tego rejestru. Na n taktów przed odczytem i-tej sygnatury na wejściu tym SUCj generuje taki n-bitowy ciąg w_i , który wprowadzany do n-stopniowego rejestru c-MISR wymusza w nim pożądaną sygnaturę. SUCj powtarza tę czynność k razy zawsze na n taktów zegarowych przed odczytem każdej kolejnej sygnatury. W zasadzie układ SUCj generuje k różnych n-bitowych ciągów w_i , co pozwala uzyskać następującą maksymalną liczbę mierzonych sygnatur $k_{\max} = \lceil m/n \rceil$. Weryfikator W3 umożliwia więc wybór dowolnej liczby k ze zbioru $\{1, 2, \dots, \lceil m/n \rceil\}$ bez potrzeby ingerowania w długość ciągu testującego. Pozwala to na pewną swobodę w ustalaniu pożądanej wartości skuteczności

$$EF_3 = 1 - 2^{-kn} \quad (4.12)$$

Liczba $\lceil m/n \rceil$ jest niezależna od stanów początkowych i wielomianów charakterystycznych rejestrów c-LFSR oraz c-MISR oraz niezależna od funkcji realizowanej przez TUC. Jeżeli m jest podzielne przez n, wówczas maksymalną skuteczność weryfikatora W3 określa wyrażenie $EF = 1 - 2^{-m}$.

W przypadku $m = 1000$ skuteczność $EF = 1 - 2^{-1000}$. Wartość ta powinna satysfakcjonować projektantów większości wysoce wiarygodnych systemów komputerowych. Pewnymi problemami, które stwarza weryfikator W3, jest technika określania funkcji układu SUCj (ciągów w_i) oraz jej minimalizacja. Pierwszy z problemów, czyli sposób określania

sekwencji wejściowej ustawiającej pożądaną sygnaturę w rejestrze liniowym z ustalonym stanem początkowym np. w postaci poprzedniej sygnatury, rozwiązano w rozdziale 2.3.4. oraz wyjaśniono w przykładzie 2.10 tego rozdziału. Natomiast na minimalizację kosztów realizacji SUC_j duży wpływ ma wybór wielomianu $p(x)$ związanego z kompaktorem c-MISR oraz wybór struktury sprzężenia związanego z wybranym wielomianem. Nie bez znaczenia jest także wybór stanu początkowego tego rejestru. Dodatkowy wpływ na minimalizację układu SUC_j może mieć wybór struktury, wielomianu $p(x)$ oraz stanu początkowego generatora testów c-LFSR. Ten złożony problem oczekuje jednak rozwiązania.



Rys. 4.8. Schemat blokowy weryfikatora W3 umożliwiający analizę k identycznych sygnatur
Fig. 4.8. Scheme of verifier W3 enabling the analysis of k identical signatures

Można projektować również hybrydowe rozwiązania weryfikatorów do k-sygnaturowej analizy bez redundancji czasu. Na przykład można wybrać dwie różne i najczęściej pojawiające się wzorcowe sygnatury s_1 oraz s_2 występujące odpowiednio k_1 oraz k_2 razy w rejestrze c-MISR zawartym w weryfikatorze W2. Pozwoli to na zwiększenie liczby k_{max} do wartości $k_1 + k_2$. Oczywiście w takim przypadku należy bramkę OR na wyjściu rejestru c-MISR w weryfikatorze W2 zastąpić, stosowanym w technice analizy FSM, układem kombinacyjnym dekodującym dwie sygnatury s_1 oraz s_2 . Skuteczność takiego hybrydowego weryfikatora określa następujące wyrażenie:

$$EF_{h1} = 1 - 2^{-2} 2^{-(k_1 + k_2)n} \quad (4.13)$$

Wartość skuteczności EF_{h1} znacznie przewyższy wartość EF_2 , jeżeli, wybrana dla niehybrydowego weryfikatora W2, wzorcowa sygnatura pojawi się $k = k_1$ razy na wyjściu rejestru c-MISR w trakcie procesu kompaktacji odpowiedzi TUC. To zwiększenie skuteczności wymaga jednak poniesienia nieznacznego dodatkowego kosztu sprzętu niezbędnego przy realizacji dekodera dwóch sygnatur s_1 oraz s_2 .

Innym hybrydowym rozwiązaniem może być weryfikator W2 uzupełniony znacznie okrojonym układem SUC_j z weryfikatora W3. Dodana uproszczona wersja układu SUC_j pozwala na wymuszenie k_2 dodatkowych sygnatur identycznych do tych, które w rejestrze c-MISR konwencjonalnego weryfikatora W2 występują k_1 razy. Wówczas skuteczność określona wyrażeniem

$$EF_{h2} = 1 - 2^{-(k_1 + k_2)n} \quad (4.14)$$

będzie większa od skuteczności EF_{h1} .

Wszystkie przedstawione techniki k-sygnaturowej analizy bez redundancji czasu pozwalają znacząco zmniejszyć zjawisko maskowania błędów. Na przykład przy założeniu $n = 16$ oraz $k = 20$ prawdopodobieństwo niewykrycia błędów jest równe $2^{-320} \cong 4.6817 \times 10^{-97}$. Jest to wartość praktycznie rzecz biorąc równa zero. Oznacza ona, że na 2.136×10^{96} macierzy błędów tylko jedna może zostać niewykryta. Techniki k-sygnaturowej analizy bez redundancji czasu nie tylko drastycznie zmniejszają maskowanie błędów. Umożliwiają one także w przypadku TUC ze zbyt małą liczbą wyjść pierwotnych, podłączonych w związku z tym do rejestru c-MISR z małą liczbą stopni wejściowych np. $n < 16$, uzyskanie skuteczności osiąganą normalnie dla szesnasto- i więcejstopniowych kompaktorów. Techniki te ponadto umożliwiają skrócenie czasu testowania [Lee88]. Po odczycie pierwszej sygnatury niezgodnej ze wzorcową proces testowania może zostać zakończony. Nie trzeba w związku z tym tracić czasu na zbędne podawanie dalszych testów, w przypadku jeżeli już pierwsze testy pobudziły uszkodzenie i jedna z pierwszych zmierzonych sygnatur już je wskazała. Łatwiejsze jest także obliczanie współczynnika pokrycia uszkodzeń [Lambi91]. Podobnie jak w przypadku analizy kSA z redundancją czasu opisana powyżej k-sygnaturowa analiza bez nadmiaru czasu pozwala również zwiększać rozdzielczość diagnostyczną procesu analizy sygnaturowej uszkodzeń [Waicu87].

4.3.3. Zmniejszanie maskowania błędów za pomocą analizy sygnatury i ilorazu

Oprócz odczytu w rejestrze c-MISR końcowej n-bitowej sygnatury można także obserwować (m-1)-bitowy ciąg, który w trakcie kompaktacji m-wektorowej macierzy odpowiedzi TUC wysuwany jest na zewnątrz poprzez wyjście rejestru c-MISR. Ciąg ten jest ilorazem z dzielenia 2.1. Jego wielomianową postać przedstawia wyrażenie $q_c(x) = [w_c(x) + x^m h_c(x)]/p_c(x) + r_c(x)/p_c(x)$. Iloraz ten wraz z sygnaturą stanowi (m+n-1)-bitowe słowo, w które zakodowana zostaje za pomocą rejestru c-MISR (m x n)-bitowa macierz odpowiedzi TUC. Uzyskanie informacji o uszkodzeniu TUC wymaga porównania tego słowa ze wzorcową sygnaturą oraz wzorcowym ilorazem. Technika analizy sygnaturowej i ilorazowej (ASI) została przedstawiona w [Sridh82], a następnie w zmodyfikowanej postaci w pracach [Hławi90e, Hławi91a, Hławi93b]. Jej skuteczność w oparciu o wspomniane prace określa wyrażenie

$$EF_3 = 1 - \frac{2^{mn - (m \cdot n - 1)} - 1}{2^{mn} - 1} \quad (4.15)$$

które dla $m \geq n$ przyjmuje uproszczoną postać:

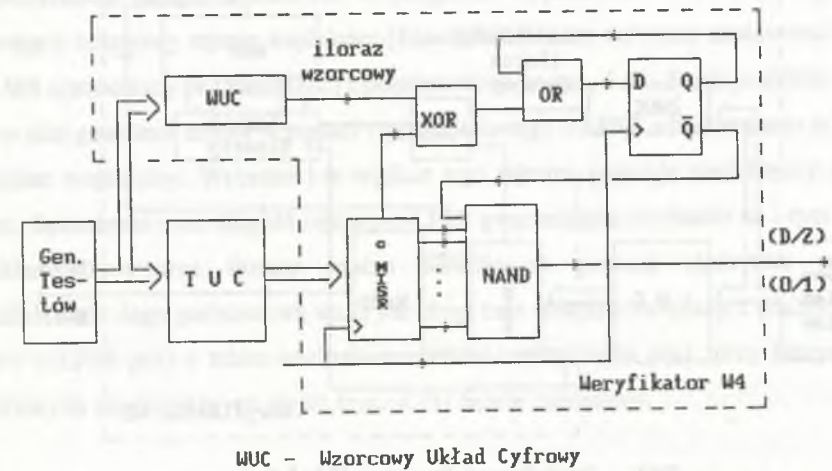
$$EF_4 = 1 - 2^{-(m \cdot n - 1)} \quad (4.16)$$

Wzięty z prac [Hławi90e, Hławi91a, Hławi93b] i przedstawiony na rys. 4.9. weryfikator W4 umożliwia realizację techniki ASI. Pobudzony przez generator testów kombinacyjny wzorcowy układ cyfrowy WUC tworzy na swoim wyjściu w czasie samotestowania wzorcowy iloraz. Dzięki połączeniu wyjścia pułpki (przerzutnik D i bramka OR) z wejściem bramki NAND sygnał D/Z zawiera również informacje o zgodności (niezgodności) wzorcowego ilorazu z ilorazem zmierzonym. Zastosowanie bramki NAND redukuje czas weryfikacji do zera. Niestety weryfikator w opisanej postaci sygnalizuje wykrycie błędu na końcu sesji testowania, co nie pozwala na skrócenie czasu testowania.

Rezygnując z porównywania sygnatury zmierzonej ze wzorcową, można usunąć z weryfikatora W4 bramkę NAND. Sygnał D/Z w takim przypadku odczytywany jest na wyjściu Q przerzutnika D. Wówczas niezgodność pomiędzy ilorazem wzorcowym oraz mierzonym jest sygnalizowana na bieżąco. W efekcie po wykryciu pierwszego niezgodnego bitu ilorazu można przerwać testowanie. Wyeliminowanie bramki NAND zmniejsza nieznacznie skuteczność, którą można określić w tym przypadku za pomocą następującego wyrażenia:

$$EF_4' = 1 - 2^{-(m-1)} \quad (4.17)$$

Jeżeli na przykład $m = 1000$ oraz $n = 20$, wówczas w oparciu o wyrażenie 4.17 otrzymujemy $EF_4' = 1 - 2^{-999} \cong 1$, natomiast w oparciu o 4.16 otrzymujemy $EF_4 = 1 - 2^{-1019} \cong 1$. Różnica pomiędzy EF_4' i EF_3 jest tak znikoma, że można ją praktycznie pominąć.

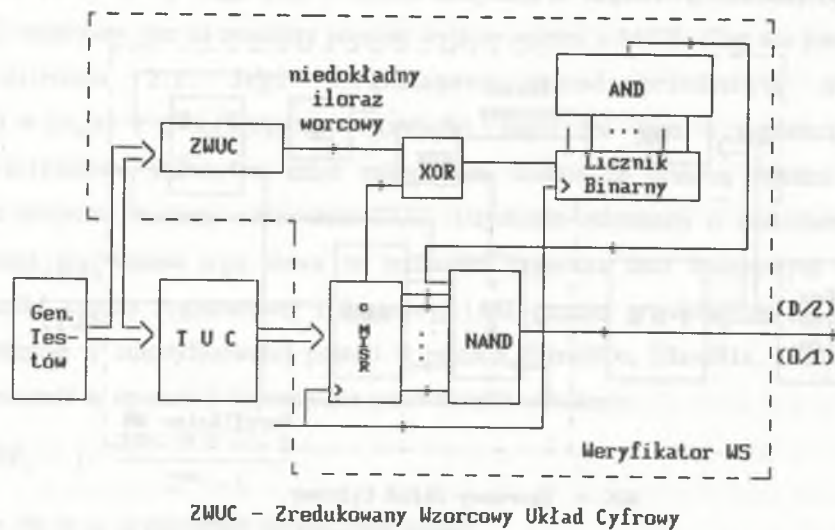


Rys. 4.9. Schemat blokowy weryfikatora W4 z kompaktorem typu c-MISR i wzorcowym układem cyfrowym WUC

Fig. 4.9. Scheme of verifier W4 with type c-MISR compactor and reference digital circuit WUC

Dalsza redukcja powierzchni struktury krzemowej zajmowanej przez weryfikator wymaga okrojenia struktury logicznej wzorcowego układu cyfrowego WUC. Jak zbudować weryfikator, aby redukcja tego układu miała tylko nieznaczny wpływ na zmniejszenie się skuteczności EF? Odpowiedź na to pytanie ilustruje schemat blokowy weryfikatora W5 wzięty z [Hławi90e, Hławi91a, Hławi93b] i przedstawiony na rys. 4.10. Główna idea pracy tego weryfikatora została opracowana w [ZoriA86]. Zredukowany wzorcowy układ cyfrowy ZWUC nie generuje na swoim wyjściu dokładnego wzorca ilorazu. Na wyjściu bramki XOR w miejsce ciągu samych zer pojawi się ciąg z bitami jedynekowymi na tych pozycjach ciągu, dla których niedokładny wzorec ilorazu różni się od dokładnego wzorca ilorazu. Oznaczmy liczbę tych jedynek przez t. Będzie ona charakteryzować ciąg na wyjściu bramki XOR, gdy testowany układ cyfrowy jest nieuszkodzony i nazywana będzie wzorcowym syndromem tego ciągu. Syndrom obliczany jest w liczniku binarnym, który pełni rolę nieliniowego kompaktora [Savir80] o pojemności $\lg(m)$. Wzorcowy syndrom zakodowany jest za pomocą bramki AND połączonej z wyjściem licznika binarnego. Jeżeli testowany układ cyfrowy jest uszkodzony, wówczas zmierzony syndrom z bardzo dużym prawdopodo-

bieństwem będzie różny od t . Jakkolwiek różnica pomiędzy syndromem wzorcowym a syndromem zmierzonym jest sygnalizowana na wyjściu D/Z.



ZWUC - Zredukowany Wzorcowy Układ Cyfrowy

Rys. 4.10. Schemat blokowy weryfikatora W5 zawierającego rejestr c-MISR, licznik binarny i zredukowany wzorcowy układ cyfrowy

Fig. 4.10. Scheme of verifier W5 including register c-MISR, binary counter and a reduced reference digital circuit

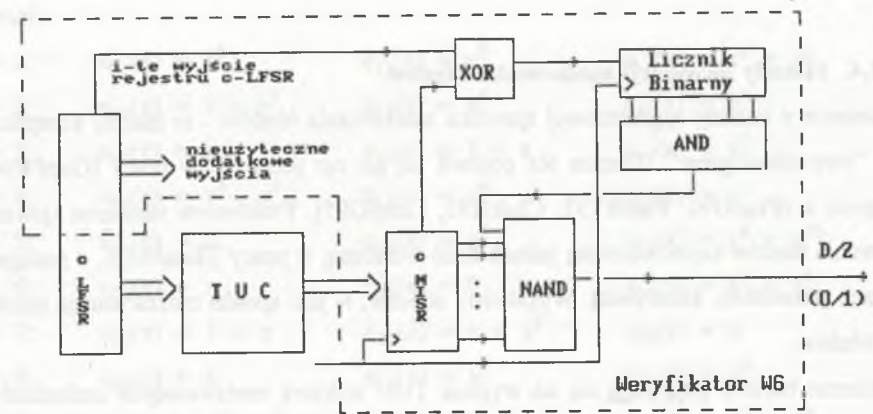
Skuteczność symultanicznego kompaktora zawartego w weryfikatorze W5 określono w pracach [Hławi90e, Hławi91a, Hławi93b] na podstawie [ZoriA86]. Przedstawia ją następujące wyrażenie:

$$EF_5 = 1 - \frac{\binom{m-1}{t} 2^{mn - (m+n-1)} - 1}{2^{mn} - 1} \quad (4.18)$$

Dla $t=0$, czyli dla dokładnego ilorazu wzorcowego, dwumian Newtona zawarty w tym wyrażeniu przyjmuje wartość 1. W ten sposób całe wyrażenie staje się podobne do wyrażenia 4.15. W konsekwencji weryfikator W5 można w takim przypadku zastąpić weryfikatorem W4.

Syndrom wzorcowy jest funkcją struktury zredukowanego wzorcowego układu cyfrowego. Im formuła boolowska opisująca strukturę tego układu posiada mniej implikantów generujących jedynki w tych taktach zegarowych, w których iloraz na wyjściu c-MISR posiada jedynki, tym liczba t jest większa. Wpływa to na zmniejszanie się skuteczności EF_5 . W efekcie $EF_5 \leq EF_4$.

Wadą weryfikatora W5 jest konieczność budowy nie tylko licznika binarnego, ale także dodatkowego układu kombinacyjnego ZWUC. Wymaga to zarezerwowania pewnej powierzchni struktury krzemowej przeznaczonej na realizację tego weryfikatora. Tę powierzchnię można zredukować w przypadku wyposażenia TUC w sprzęg 1149.1 zawierający brzegowy rejestr wejściowy [Hławi93b]. Należy wówczas zastosować weryfikator W6 opracowany w [BaduH88c] i przedstawiony na rys. 4.11. Funkcję układu ZWUC pełni w nim generator testów w postaci rejestru liniowego c-LFSR wbudowanego w brzegowy rejestr wejściowy. Wybrane i -te wyjście tego rejestru generuje niedokładny wzorec ilorazu. Sprzężenie oraz długość rejestru c-LFSR gwarantująca uzyskanie na i -tym wyjściu niedokładnego wzorca ilorazu można określić za pomocą algorytmu podanego w [BaduH88c]. Jego podstawową wadą jest długi czas obliczeń związany z poszukiwaniem rejestru c-LFSR- $p(x)$ z takim wielomianem charakterystycznym $p(x)$, przy którym liczba dodatkowych nieużytecznych wyjść (rys. 4.11) będzie minimalna.



Rys. 4.11. Schemat blokowy weryfikatora W6 zawierającego rejestr c-MISR, licznik binarny i rejestr liniowy (GT) zastępujący układ ZWUC

Fig. 4.11. Scheme of verifier W6 including register c-MISR, binary counter and linear register (GT) which replaces circuit ZWUC

Analiza sygnatury i ilorazu znacząco zmniejsza maskowanie błędów. Poziom skuteczności weryfikatorów wykorzystujących technikę ASI jest podobny do poziomu skuteczności analizy k -sygnaturowej, w szczególności analizy bez redundancji czasu. Zwiększona jest także rozdzielczość diagnostyczna uzyskiwana w efekcie stosowania

techniki ASI. Niestety modyfikacje tej techniki zastosowane w weryfikatorach W5 oraz W6 uniemożliwiają skracanie czasu testowania.

Równoczesne stosowanie techniki ASI oraz analizy k-sygnaturowej bez redundancji czasu umożliwia znaczące zmniejszenie dotąd otrzymywanych liczb wyrażających prawdopodobieństwo maskowania błędów. Niestety wymaga to jednocześnie podwojenia kosztu dodatkowego sprzętu. Całkowita likwidacja maskowania błędów jest możliwa w przypadku zastosowania weryfikatora W1 z liczbą $k = m$ sygnatur. Przy takiej liczbie pomiarów sygnatur weryfikator W1 wykrywa, jak już wcześniej wspomniano, każdą macierz błędów bez względu na rodzaj funkcji TUC, zbiór jego rzeczywistych uszkodzeń czy też kolejność podawanych na wejścia TUC testów. Jest więc on dostosowany do wykrywania wszystkich jednakowo prawdopodobnych błędów. Niestety ta uniwersalność jest nieopłacalna z ekonomicznego punktu widzenia. Tak więc znalezienie tańszych technik likwidacji maskowania błędów jest istotnym zagadnieniem i ma szczególne znaczenie dla wysoce wiarygodnych systemów komputerowych.

4.3.4. Metody likwidacji maskowania błędów

Usunięcie z analizy sygnaturowej zjawiska maskowania błędów - to inaczej kompaktacja z tzw. "zero-aliasingiem". Termin ten pojawił się po raz pierwszy w pracy [GupPR90], a następnie w [PradG91, PomRT92, ChakH93, LempG95]. Problemem usunięcia zjawiska maskowania błędów zajmowano się jednak dużo wcześniej w pracy [HassM83], a następnie w pracach [Hławi84b, Hławi84d]. Wyjaśnimy obecnie, w jaki sposób można usunąć maskowanie błędów.

Macierze błędów pojawiają się na wyjściu TUC wskutek rzeczywistych uszkodzeń f . Oznaczmy takie macierze przez \mathbf{E}_f , a ich pełny zbiór przez ε_f . Liczność tego zbioru $|\varepsilon_f| < |\varepsilon|$. Oznaczmy przez $\varepsilon_{fm} = \varepsilon_f \cap \varepsilon_m$ podzbiór macierzy $\mathbf{E}_f \neq 0$ odwzorowywanych w sygnaturę zerową charakterystyczną dla macierzy $\mathbf{E}_0 = 0$. Na tej podstawie i w oparciu o 4.1a rzeczywiste prawdopodobieństwo niewykrycia błędów $P_{alf}(m)$ można określić za pomocą następującego wyrażenia:

$$P_{alf} = \frac{|\varepsilon_{fm}|}{|\varepsilon_f|} \quad (4.19)$$

Zauważmy, że w praktyce licznosc $|\varepsilon_{fm}| < |\varepsilon_m|$. Usunięcie maskowania błędów oznacza, że $|\varepsilon_{fm}| = 0$. Wówczas każda macierz błędu \mathbf{E}_f jest wykrywana. Można to uzyskać wtedy,

gdy $R[e_{cf}(x), p_c(x)] \neq 0$ dla każdego uszkodzenia f . Uzyskanie $|\varepsilon_{fm}| = 0$ zależy więc przede wszystkim od zbioru zamodelowanych uszkodzeń [PomRT92]. Zależy ono także od kolejności, w jakiej podane zostaną testy na wejścia TUC [PomRT92], od wielomianu charakterystycznego $p(x)$ związanego z kompaktorem c -MISR- $p(x)$ [PomRT92] oraz od struktury c jego sprzężenia liniowego. Dobierając którykolwiek z tych trzech parametrów, można uzyskać wartość $|\varepsilon_{fm}| = 0$. Wyjaśnia to przykład 4.9, w którym kompaktorem jest rejestr IED-MISR- $p(x)$. Wielomiany błędów, przy takiej strukturze równoległego kompaktora, są niezależne od rodzaju wielomianu $p(x)$. Można więc dobrać taki wielomian $p(x)$, który nie będzie dzielnikiem żadnego z wielomianów błędów.

Przykład 4.9

Załóżmy trójwyjściowy TUC z dziewięcioma uszkodzeniami $f = 1, 2, \dots, 9$. Uszkodzenia te pobudzone testami objawiają się na wejściach 0, 1 i 2 rejestru IED-MISR wielomianami błędów $e_{0f}(x)$, $e_{1f}(x)$, $e_{2f}(x)$, które dla poszczególnych uszkodzeń przyjmują następujące postaci:

$$\begin{array}{lll} f = 1; & e_{01}(x) = x^5, & e_{11}(x) = x^5, & e_{21}(x) = x^6 + x^8 \\ f = 2; & e_{02}(x) = 1 + x^2, & e_{12}(x) = x^3, & e_{22}(x) = x^3 \\ f = 3; & e_{03}(x) = 1, & e_{13}(x) = 1 + x, & e_{23}(x) = x^2 \\ f = 4; & e_{04}(x) = x^3 + x^4, & e_{14}(x) = x^4, & e_{24}(x) = x^6 \\ f = 5; & e_{05}(x) = 0, & e_{15}(x) = x^5, & e_{25}(x) = x^6 + x^{10} \\ f = 6; & e_{06}(x) = x, & e_{16}(x) = 0, & e_{26}(x) = x^3 + x^4 \\ f = 7; & e_{07}(x) = 1 + x, & e_{17}(x) = x + x^2, & e_{27}(x) = 0 \\ f = 8; & e_{08}(x) = x, & e_{18}(x) = x^2, & e_{28}(x) = x^4 \\ f = 9; & e_{09}(x) = x^2 + x^4, & e_{19}(x) = 0, & e_{29}(x) = x^3 + x^4 \end{array}$$

Na wyjściu abstrakcyjnego przetwornika $P1_{IED}$ (rys. 2.7) w schemacie zastępczym kompaktora IED-MISR- $p(x)$ wielomiany te tworzą sumę

$e_{IEDf}(x) = e_{0f}(x) + e_{1f}(x)x + e_{2f}(x)x^2$, która dla dziewięciu przykładowych uszkodzeń przyjmuje następujące formy:

$$\begin{array}{ll} f = 1; & e_{IED1}(x) = x^{10} + x^8 + x^6 + x^5 = x^5(x^4 + x^3 + 1)(x + 1) \\ f = 2; & e_{IED2}(x) = x^5 + x^4 + x^2 + 1 = (x^4 + x + 1)(x + 1) \\ f = 3; & e_{IED3}(x) = x^4 + x^2 + x + 1 = (x^3 + x^2 + 1)(x + 1) \\ f = 4; & e_{IED4}(x) = x^8 + x^5 + x^4 + x^3 = x^3(x^3 + x + 1)(x + 1)^2 \end{array}$$

$$\begin{aligned}
 f = 5; & \quad e_{\text{IED}_5}(x) = x^{12} + x^8 + x^6 = x^6(x^3 + x + 1)^2 \\
 f = 6; & \quad e_{\text{IED}_6}(x) = x^6 + x^5 + x = x(x^3 + x + 1)(x^2 + x + 1) \\
 f = 7; & \quad e_{\text{IED}_7}(x) = x^3 + x^2 + x + 1 = (x + 1)^3 \\
 f = 8; & \quad e_{\text{IED}_8}(x) = x^6 + x^2 + x = x(x^3 + x^2 + 1)(x^2 + x + 1) \\
 f = 9; & \quad e_{\text{IED}_9}(x) = x^6 + x^5 + x^4 + x^2 = x^2(x^3 + x + 1)(x + 1)
 \end{aligned}$$

Poszukiwany jest taki pierwotny wielomian $p(x)$ związany z kompaktorem IED-MISR- $p(x)$, który nie będzie dzielnikiem powyższych wielomianów błędów. Wielomiany pierwotne $p_1(x) = x + 1$, $p_2(x) = x^2 + x + 1$, $p_3(x) = x^3 + x + 1$, $p_4(x) = x^3 + x^2 + 1$, $p_5(x) = x^4 + x^3 + 1$, $p_6(x) = x^4 + x + 1$ są dzielnikami wielomianów błędów wymuszonych odpowiednio następującymi zbiorami uszkodzeń: $\{1,2,4,7,9,10\}$, $\{6,8\}$, $\{4,5,6,9\}$, $\{3,8\}$, $\{1\}$, $\{2\}$. Wszystkie wielomiany pierwotne do stopnia $n = 4$ są podzielnikami. Natomiast nie ma wśród podzielników wielomianów pierwotnych stopnia 5. Można więc wybrać na przykład wielomian pierwotny $p_3(x) = x^3 + x + 1$. Umożliwia on realizację sprzężenia rejestru IED-MISR gwarantującego zlikwidowanie maskowania błędów dla założonego zbioru błędów.

Autorzy pracy [HassM83], chcąc uzyskać dokładny opis wielomianów błędów związanych z uszkodzeniami w układach PLA, najpierw określili postać wielomianów opisujących ciągi pojawiające się na wyjściu źródła testów pseudoprzykładowych generowanych przez rejestr LFSR. Następnie przyjęli model uszkodzeń typu sklejenie s-z-0 oraz s-z-1 i na tej podstawie określili dokładną postać wielomianową błędów zależnych. Po czym zbadali ich podzielność przez wielomian charakterystyczny związany z rejestrem IED-MISR- $p(x)$ i wybrali taki wielomian $p(x)$, który gwarantował wykrycie wszystkich modelowanych uszkodzeń. W pracy [Jou95] zastosowano identyczną technikę, ale przy założeniu generatora testów deterministycznych. Również w [NagvK92] zastosowano podobną technikę do analizy sygnaturowej błędów na wyjściu dwupoziomowych układów logicznych AND-OR lub AND-XOR z wewnątrzukładowym testerem w postaci generatora testów LFSR- $p(x)$ i kompaktora MISR- $p(x)$ z identycznym wielomianem charakterystycznym $p(x)$.

Podobne podejście zastosowano także w pracach [Hławi84b, Hławi84d] do zbadania możliwości likwidacji maskowania błędów w analizie sygnaturowej magistrali adresowej podczas testu typu "free run". W celu uproszczenia rozważań znacznie zawężono model uszkodzeń, ograniczając go do sklejeń pojedynczych linii tej magistrali. W pracach tych opisano za pomocą wielomianów ciągi testów generowanych na i-tych wyjściach k-bitowego

licznika binarnego (licznika programu). Następnie określono wielomiany błędów generowanych na i-tych wyjściach licznika w obecności uszkodzenia s-z-0 lub s-z-1. Ich postaci odpowiednio dla s-z-0 oraz s-z-1 przedstawiają poniższe wyrażenia

$$e_{\text{ki}0}(x) = x^{2i}(1+x)^{(2^k-2)-(2^i-1)} \quad e_{\text{ki}1}(x) = (1+x)^{(2^k-2)-(2^i-1)}$$

W [Hławi84b, Hławi84d] wykazano, że reszty $R[e_{\text{ki}0}(x), p(x)]$ oraz $R[e_{\text{ki}1}(x), p(x)]$ są różne od zera już dla każdego wielomianu pierwotnego $p(x)$ stopnia $n \geq 2$. Podobnie w [Hławi84d] udowodniono, że kompaktor IED-MISR- $p(x)$ z wielomianem pierwotnym $p(x)$ stopnia $n = k$ wykrywa podczas testu "free run" każde z założonych na magistrali adresowej uszkodzeń s-z-0 oraz s-z-1.

W podobny sposób można badać również realną skuteczność analizy sygnaturowej połączeń między układami ASIC wyposażonych w brzegową ścieżkę sterująco-obszerną z wbudowanym weń kompaktorem liniowym typu IED-MISR. Zarówno struktura logiczna takich połączeń, jak również modele uszkodzeń np. typu sklejenie lub model AND dla zwarć pozwalają na wyprowadzanie ogólniejszych wniosków. Przykładem jest zwarcie wielu linii zasilanych testem w postaci kroczonej jedynki. Załóżmy, że bramka AND jest modelem tego uszkodzenia. Tego typu zwarcie ilustruje rys. 4.12, na którym, pracujący jako rejestr przesuwający, brzegowy rejestr wyjściowy OBR (Output Boundary Register) z układu ASIC 1 generuje test kroczonej jedynki. Odpowiedzi testowanych połączeń wprowadzane są do układu ASIC 2 do jego wejściowego rejestru brzegowego IBR (Input Boundary Register) skonfigurowanego w rejestr IED-MISR- $p(x)$. Zwarcie powoduje, że na wejściach 0, j, j+1, r kompaktora IED-MISR- $p(x)$ pojawiają się odpowiednio następujące wielomiany błędów x^i , x^j , x^{i+k} , x^{i+v} . W efekcie proces kompaktacji tych błędów można opisać następującym dzieleniem

$$(x^i + x^j x^k + x^{i+k} x^{j+1} + x^{i+v} x^r)/p(x) = x^i(1 + x^j + x^{k+j+1} + x^{v+r})/p(x).$$

Gdy wielomian $p(x)$ jest wielomianem pierwszym, a $\deg p(x) > v+r$, wówczas $R[x^i(1 + x^j + x^{k+j+1} + x^{v+r}), p(x)] \neq 0$. Tak więc każde zwarcie linii wyprowadzonych z wyjść rejestru OBR odległych od siebie nie więcej niż o v i wprowadzonych do wejść rejestru IED-MISR- $p(x)$ odległych od siebie nie więcej niż o r zostanie wykryte, jeżeli jest realizowany test kroczonej jedynki i jeżeli stopień wielomianu pierwszego $p(x)$, $\deg p(x)$, jest większy od $v+r$.

$$\begin{aligned}
f = 5; & \quad e_{\text{IED}_5}(x) = x^{12} + x^8 + x^6 = x^6(x^3 + x + 1)^2 \\
f = 6; & \quad e_{\text{IED}_6}(x) = x^6 + x^5 + x = x(x^3 + x + 1)(x^2 + x + 1) \\
f = 7; & \quad e_{\text{IED}_7}(x) = x^3 + x^2 + x + 1 = (x + 1)^3 \\
f = 8; & \quad e_{\text{IED}_8}(x) = x^6 + x^2 + x = x(x^3 + x^2 + 1)(x^2 + x + 1) \\
f = 9; & \quad e_{\text{IED}_9}(x) = x^6 + x^5 + x^4 + x^2 = x^2(x^3 + x + 1)(x + 1)
\end{aligned}$$

Poszukiwany jest taki pierwotny wielomian $p(x)$ związany z kompaktem IED-MISR- $p(x)$, który nie będzie dzielnikiem powyższych wielomianów błędów. Wielomiany pierwotne $p_1(x) = x + 1$, $p_2(x) = x^2 + x + 1$, $p_3(x) = x^3 + x + 1$, $p_4(x) = x^3 + x^2 + 1$, $p_5(x) = x^4 + x^3 + 1$, $p_6(x) = x^4 + x + 1$ są dzielnikami wielomianów błędów wymuszonych odpowiednio następującymi zbiorami uszkodzeń: $\{1, 2, 4, 7, 9, 10\}$, $\{6, 8\}$, $\{4, 5, 6, 9\}$, $\{3, 8\}$, $\{1\}$, $\{2\}$. Wszystkie wielomiany pierwotne do stopnia $n = 4$ są dzielnikami. Natomiast nie ma wśród dzielników wielomianów pierwotnych stopnia 5. Można więc wybrać na przykład wielomian pierwotny $p_a(x) = x^5 + x^3 + 1$. Umożliwia on realizację sprzężenia rejestru IED-MISR gwarantującego zlikwidowanie maskowania błędów dla założonego zbioru błędów.

Autorzy pracy [HassM83], chcąc uzyskać dokładny opis wielomianów błędów związanych z uszkodzeniami w układach PLA, najpierw określili postać wielomianów opisujących ciągi pojawiające się na wyjściu źródła testów pseudoprzypadkowych generowanych przez rejestr LFSR. Następnie przyjęli model uszkodzeń typu sklejenie s-z-0 oraz s-z-1 i na tej podstawie określili dokładną postać wielomianową błędów zależnych. Po czym zbadali ich podzielność przez wielomian charakterystyczny związany z rejestrem IED-MISR- $p(x)$ i wybrali taki wielomian $p(x)$, który gwarantował wykrycie wszystkich modelowanych uszkodzeń. W pracy [Jou95] zastosowano identyczną technikę, ale przy założeniu generatora testów deterministycznych. Również w [NagvK92] zastosowano podobną technikę do analizy sygnaturowej błędów na wyjściu dwupoziomowych układów logicznych AND-OR lub AND-XOR z wewnątrzukładowym testerem w postaci generatora testów LFSR- $p(x)$ i kompaktora MISR- $p(x)$ z identycznym wielomianem charakterystycznym $p(x)$.

Podobne podejście zastosowano także w pracach [Hławi84b, Hławi84d] do zbadania możliwości likwidacji maskowania błędów w analizie sygnaturowej magistrali adresowej podczas testu typu "free run". W celu uproszczenia rozważań znacznie zawężono model uszkodzeń, ograniczając go do sklejeń pojedynczych linii tej magistrali. W pracach tych opisano za pomocą wielomianów ciągi testów generowanych na i-tych wyjściach k-bitowego

licznika binarnego (licznika programu). Następnie określono wielomiany błędów generowanych na i-tych wyjściach licznika w obecności uszkodzenia s-z-0 lub s-z-1. Ich postaci odpowiednio dla s-z-0 oraz s-z-1 przedstawiają poniższe wyrażenia

$$e_{\text{ki}0}(x) = x^{2i}(1+x)^{(2^k-2)-(2^i-1)} \quad e_{\text{ki}1}(x) = (1+x)^{(2^k-2)-(2^i-1)}$$

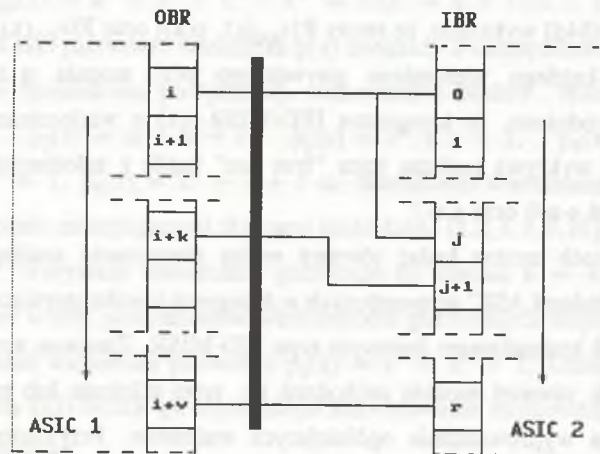
W [Hławi84b, Hławi84d] wykazano, że reszty $R[e_{\text{ki}0}(x), p(x)]$ oraz $R[e_{\text{ki}1}(x), p(x)]$ są różne od zera już dla każdego wielomianu pierwotnego $p(x)$ stopnia $n \geq 2$. Podobnie w [Hławi84d] udowodniono, że kompaktor IED-MISR- $p(x)$ z wielomianem pierwotnym $p(x)$ stopnia $n = k$ wykrywa podczas testu "free run" każde z założonych na magistrali adresowej uszkodzeń s-z-0 oraz s-z-1.

W podobny sposób można badać również realną skuteczność analizy sygnaturowej połączeń między układami ASIC wyposażonych w brzegową ścieżkę sterująco-obszerną z wbudowanym weń kompaktem liniowym typu IED-MISR. Zarówno struktura logiczna takich połączeń, jak również modele uszkodzeń np. typu sklejenie lub model AND dla zwarć pozwalają na wyprowadzanie ogólniejszych wniosków. Przykładem jest zwarcie wielu linii zasilanych testem w postaci kroczonej jedyńki. Założmy, że bramka AND jest modelem tego uszkodzenia. Tego typu zwarcie ilustruje rys. 4.12, na którym, pracujący jako rejestr przesuwający, brzegowy rejestr wyjściowy OBR (Output Boundary Register) z układu ASIC 1 generuje test kroczonej jedyńki. Odpowiedzi testowanych połączeń wprowadzane są do układu ASIC 2 do jego wejściowego rejestru brzegowego IBR (Input Boundary Register) skonfigurowanego w rejestr IED-MISR- $p(x)$. Zwarcie powoduje, że na wejściach 0, j, j+1, r kompaktora IED-MISR- $p(x)$ pojawiają się odpowiednio następujące wielomiany błędów x^i , x^j , x^{i+k} , x^{j+1} , x^{i+v} , x^r . W efekcie proces kompaktacji tych błędów można opisać następującym dzieleniem

$$(x^i + x^j x^j + x^{i+k} x^{j+1} + x^{i+v} x^r) / p(x) = x^i (1 + x^j + x^{k+j+1} + x^{v+r}) / p(x).$$

Gdy wielomian $p(x)$ jest wielomianem pierwszym, a $\deg p(x) > v+r$, wówczas $R[x^i (1 + x^j + x^{k+j+1} + x^{v+r}), p(x)] \neq 0$. Tak więc każde zwarcie linii wyprowadzonych z wyjść rejestru OBR odległych od siebie nie więcej niż o v i wprowadzonych do wejść rejestru IED-MISR- $p(x)$ odległych od siebie nie więcej niż o r zostanie wykryte, jeżeli jest realizowany test kroczonej jedyńki i jeżeli stopień wielomianu pierwszego $p(x)$, $\deg p(x)$, jest większy od $v+r$.

Uzyskany powyżej wynik - dotąd nie opublikowany - jest tylko cząstkowym rozwiązaniem problemu eliminacji maskowania błędów spowodowanych uszkodzeniami w sieciach połączeń pomiędzy układami ASIC. Zagadnienie to nadal oczekuje całkowitego rozwiązania.



Rys. 4.12. Przykład zwarcia linii łączących układ ASIC 1 z układem ASIC 2
Fig. 4.12. An example of a short circuit in the lines connecting circuit ASIC 1 with circuit ASIC 2

Omówione dotąd techniki eliminacji maskowania błędów dotyczyły prostych TUC i małego zbioru zamodelowanych uszkodzeń. Polegały one głównie na takim doborze źródeł testów, aby wielomianowy opis błędów, zależnych od uszkodzeń TUC i wymuszonych tymi testami, był uniwersalny i na tyle prosty, aby umożliwiał wyprowadzanie ogólnych wniosków o jego podzielności przez wielomiany pierwotne $p(x)$ bez konieczności określania niezerowych współczynników p_i wielomianów $p(x)$. To podejście natrafia jednak na poważne trudności przy zastosowaniu go do badania skuteczności analizy sygnaturowej wewnątrz współczesnych układów ASIC. Podstawową przyczyną jest duża liczba różnych uszkodzeń, które należałoby zamodelować i dla których należałoby dobrać takie źródło testów, aby możliwe było formułowanie ogólnych wniosków związanych z podzielnością wielomianów błędów.

Rozwiązanie tego problemu niezależne od rodzaju źródła testów i nie wymagające ograniczania zbioru zamodelowanych uszkodzeń zostanie opisane poniżej przy użyciu wielomianów błędów podanych w przykładzie 4.9. Rezultaty uzyskane w tym przykładzie są

efektem doświadczenia badacza. Te same wyniki można również uzyskać, stosując pewną określoną metodę postępowania. W celu wyjaśnienia jej ogólnej idei założymy, że wszystkie wielomiany pierwotne i -tego stopnia oznaczamy będziemy tą samą literą alfabetu z indeksem liczbowym rozróżniającym je między sobą. Założymy także, że wielomiany pierwotne stopnia $i+k$, gdzie $k = 1, 2, \dots, j$ oznaczane będą innymi najlepiej kolejnymi literami alfabetu również z indeksami liczbowymi. W związku z tym wszystkie wielomiany pierwotne na przykład stopnia $n \leq 5$ można oznaczyć w następujący sposób:

$$\begin{aligned} a_1 &= (x + 1), & b_1 &= (x^2 + x + 1), & c_1 &= (x^3 + x + 1), & c_2 &= (x^3 + x^2 + 1), \\ d_1 &= (x^4 + x + 1), & d_2 &= (x^4 + x^3 + 1), & g_1 &= (x^5 + x^3 + 1), & g_2 &= (x^5 + x^2 + 1), \\ g_3 &= (x^5 + x^4 + x^3 + x^2 + 1), & g_4 &= (x^5 + x^3 + x^2 + x + 1), \\ g_5 &= (x^5 + x^4 + x^2 + x + 1), \\ g_6 &= (x^5 + x^4 + x^3 + x + 1). \end{aligned}$$

Oznaczmy ich zbiór przez $P_n = \{a_1, b_1, c_1, c_2, d_1, d_2, g_1, g_2, g_3, g_4, g_5, g_6\}$, a ich iloczyn przez $P_n(x) = a_1 b_1 c_1 c_2 d_1 d_2 g_1 g_2 g_3 g_4 g_5 g_6$ ($\deg P_n(x) = 47$ dla $n = 5$). Natomiast zbiór wszystkich wielomianów pierwotnych stopnia $n \leq 4$ oznaczmy przez $P_{n-1} = \{a_1, b_1, c_1, c_2, d_1, d_2\}$. Ich iloczyn $P_{n-1}(x) = a_1 b_1 c_1 c_2 d_1 d_2$, a stopień $\deg P_{n-1}(x) = 17$ dla $n-1 = 4$. Wykorzystując te same oznaczenia, można wielomiany błędów podane w przykładzie 4.9 zapisać w następujący sposób:

$$\begin{aligned} e_{IED1}(x) &= x^5 a_1 d_2; & e_{IED2}(x) &= a_1 d_1; & e_{IED3}(x) &= a_1 c_2; & e_{IED4}(x) &= x^3 a_1^2 c_1; \\ e_{IED5}(x) &= x^6 c_1^2; & e_{IED6}(x) &= x c_1 b_1; & e_{IED7}(x) &= a_1^3; & e_{IED8}(x) &= x b_1 c_2 \\ e_{IED9}(x) &= x^2 a_1 c_1. \end{aligned}$$

Iloczyn wszystkich przykładowych wielomianów błędów przedstawia wielomian $x^{18} E_f(x)$, gdzie $E_f(x) = a_1^9 b_1^2 c_1^5 c_2^2 d_1 d_2$; $\deg E_f(x) = 42$ dla $f = 9$. Ponieważ spełniona jest nierówność $\deg P_{n-1}(x) < \deg E_f(x) < \deg P_n(x)$, dlatego też tylko w zbiorze P_n istnieje co najmniej jeden wielomian $p(x)$ stopnia $n \leq 5$, który nie jest dzielnikiem wszystkich wielomianów błędów.

Obecnie przedstawimy technikę określania niepodzielników w zbiorze P_n . Dzieląc pierwszy wielomian błędów $e_{IED1}(x) = x^5 a_1 d_2$ przez wielomiany pierwotne pobierane kolejno ze zbioru P_n , znajdujemy te wielomiany, które są dzielnikami. Wielomiany te usuwamy ze zbioru P_n . Zredukowany w ten sposób zbiór oznaczamy przez P^I . Następnie wybieramy drugi wielomian błędów. Opisany proces powtarzamy wybierając dzielniki ze zbioru P^I . Te z nich, które były dzielnikami, usuwamy z tego zbioru. W efekcie otrzymujemy

mujemy zbiór P^{II} . Ten proces jest powtarzany dla każdego kolejnego wielomianu błędów. Po podzieleniu tą metodą ostatniego, czyli dziewiątego wielomianu błędów, otrzymujemy zbiór P^{IX} zawierający tylko takie wielomiany pierwotne, które nie są dzielnikami żadnego z przykładowych wielomianów błędów. Proces ten zilustrowano poniżej

$$\begin{aligned} e_{\text{IED1}}(x) &= x^5 a_1 d_2; & P^{\text{I}} &= \{-, b_1, c_1, c_2, d_1, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED2}}(x) &= a_1 d_1; & P^{\text{II}} &= \{-, b_1, c_1, c_2, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED3}}(x) &= a_1 c_2; & P^{\text{III}} &= \{-, b_1, c_1, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED4}}(x) &= x^3 a_1^2 c_1; & P^{\text{IV}} &= \{-, b_1, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED5}}(x) &= x^6 c_1^2; & P^{\text{V}} &= \{-, b_1, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED6}}(x) &= x c_1 b_1; & P^{\text{VI}} &= \{-, -, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED7}}(x) &= a_1^3; & P^{\text{VII}} &= \{-, -, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED8}}(x) &= x b_1 c_2; & P^{\text{VIII}} &= \{-, -, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \\ e_{\text{IED9}}(x) &= x^2 a_1 c_1; & P^{\text{IX}} &= \{-, -, -, -, -, -, g_1, g_2, g_3, g_4, g_5, g_6\} \end{aligned}$$

Każdy wielomian ze zbioru $\{g_1, g_2, g_3, g_4, g_5, g_6\}$ nie jest dzielnikiem wielomianów błędów i może być wykorzystany do budowy kompaktora IED-MISR. Ten sam rezultat można uzyskać dzieląc $P_n(x)$ przez $E_f(x)$. Wyjaśnia to poniższe wyrażenie

$$P_n(x)/E_f(x) = a_1 b_1 c_1 c_2 d_1 d_2 g_1 g_2 g_3 g_4 g_5 g_6 / a_1^9 b_1^2 c_1^5 c_2^2 d_1 d_2 = g_1 g_2 g_3 g_4 g_5 g_6 / a_1^8 b_1 c_1^4 c_2.$$

Dzieląc przez $E_f(x)$ wielomian $P_{n-1}(x)$, którego stopień $\deg P_{n-1}(x) < \deg E_f(x)$, otrzymujemy zero w liczniku, co potwierdza, że warunkiem znalezienia co najmniej jednego niepodzielnika jest $\deg E_f(x) < \deg P_n(x)$.

Niepodzielniki ze zbioru $\{g_1, g_2, g_3, g_4, g_5, g_6\}$ są wielomianami piątego stopnia. W efekcie pięciobitowy rejestr IED-MISR zbudowany w oparciu o jeden z tych wielomianów będzie o dwa przerzutniki za duży w stosunku do potrzeb wynikających z założonej w przykładzie 4.9 liczby wyjść TUC. Tańsze rozwiązanie można znaleźć wśród rejestrów IEDT-MISR. Spróbujmy więc zastosować opisaną metodę do poszukiwania niepodzielników wśród wielomianów pierwotnych $p(x)$ związanych z takimi rejestrami. Jest to możliwe tylko wtedy, gdy występujące na wyjściu abstrakcyjnego przetwornika $P1_{\text{IEDT}}$ wielomiany błędów

$$e_{\text{IEDTf}}(x) = \sum_{r=0}^{n-1} \oplus e_{r,f}(x) P_{\text{IEDTf}}(x) \quad \text{gdzie:} \quad P_{\text{IEDTf}}(x) = \prod_{j=0}^{r-1} (k_j + x), \quad P_{\text{IEDT0}}(x) = 1$$

będą niezależne od rodzaju zastosowanego wielomianu charakterystycznego $p(x)$. Efekt ten można uzyskać w przypadku takich rejestrów IEDT-MISR- $p(x)$, które niezależnie od związanego z nimi wielomianu charakterystycznego $p(x)$ posiadają komórki $\oplus' \mathbf{D}(\mathbf{T})$ umiejscowione stale w tych samych stopniach rejestru. Optymalne wśród nich są te, które zawierają tylko jedną komórkę $\oplus' \mathbf{D}(\mathbf{T})$ ulokowaną stale, np. w zerowym stopniu rejestru. Wówczas każdy wielomian charakterystyczny $p(x)$ o nieparzystej liczbie niezerowych współczynników p_i można przekształcić w postać $p(x) = 1 + x^a(1+x)d(x) = 1 + x^a(1+x)[1 + x^b[1 + x^c[1 + \dots + x^d[1 + \dots + x^s[1 + x^w] \dots] \dots]]$, której odpowiada rejestr o strukturze $\mathbf{T} \mathbf{D}^a \oplus \mathbf{D}^b \oplus \mathbf{D}^c \oplus \dots \oplus \mathbf{D}^j \oplus \dots \oplus \mathbf{D}^s \oplus \mathbf{D}^w$ (zob. rozdział 2.2). Ponieważ w takich rejestrach tylko współczynnik $k_0 = 1$, dlatego też wielomiany związane z wejściami tych rejestrów przyjmują postać $P_{\text{IEDTf}}(x) = (1+x)x^{r-1}$ (zob. tab. 2.1). W efekcie wielomiany błędów $e_{\text{IEDTf}}(x)$ posiadają stałą postać

$$e_{\text{IEDTf}}(x) = e_{0f}(x) + (1+x) \sum_{r=0}^{n-2} \oplus e_{r+1,f}(x) x^r \quad (4.20)$$

niezależnie od rodzaju wielomianu $p(x)$. Podobnie jest w przypadku, w którym komórka $\oplus' \mathbf{D}(\mathbf{T})$ znajduje się nie w zerowym stopniu, ale stale w pierwszym stopniu rejestru IEDT-MISR. Wówczas współczynnik $k_1 = 1$, natomiast pozostałe współczynniki k_r są równe zeru. W efekcie wielomiany błędów $e_{\text{IEDTf}}(x)$, niezależnie od rodzaju wielomianu $p(x)$, posiadają następującą niezmienną formę

$$e'_{\text{IEDTf}}(x) = e_{0f}(x) + e_{1f}(x)x + (1+x) \sum_{i=1}^{n-2} \oplus e_{i+1,f}(x) x^i \quad (4.21)$$

Zastosowanie wyrażenia 4.21 lub 4.20 w procesie poszukiwania niepodzielników zilustrujemy w poniższym przykładzie.

Przykład 4.10

Wykorzystując wyrażenie 4.21 i podane w przykładzie 4.9 wielomiany błędów $e_{0f}(x)$, $e_{1f}(x)$, $e_{2f}(x)$ określimy obecnie zbiór dziewięciu wielomianów błędów $e'_{\text{IEDTf}}(x)$ występujących w schemacie zastępczym kompaktora IEDT-MISR- $p(x)$ na wyjściu abstrakcyjnego przetwornika $P1_{\text{IEDT}}$. Wielomiany $e_{0f}(x)$, $e_{1f}(x)$, $e_{2f}(x)$ tworzą sumę $e'_{\text{IEDTf}}(x) = e_{0f}(x) + e_{1f}(x)x + e_{2f}(x)(1+x)$, która dla dziewięciu przykładowych uszkodzeń przyjmuje następujące formy:

$$\begin{aligned} f = 1; & \quad e'_{\text{IEDT1}}(x) = x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 = x^5(x^2+x+1)^2(x+1) = x^5 a_1 b_1^2 \\ f = 2; & \quad e'_{\text{IEDT2}}(x) = x^5 + x^2 + 1 = g_2 \end{aligned}$$

$$\begin{aligned}
f = 3; e'_{\text{IEDT3}}(x) &= x^4 + x^3 + x^2 + x + 1 && (\text{wielomian pierwszy}) && = d_3 \\
f = 4; e'_{\text{IEDT4}}(x) &= x^8 + x^7 + x^5 + x^4 + x^3 && = x^3(x^5 + x^4 + x^2 + x + 1) && = x^3 g_5 \\
f = 5; e'_{\text{IEDT5}}(x) &= x^{12} + x^{11} + x^8 + x^7 + x^6 && = x^6(x^6 + x^5 + x^2 + x + 1) && = x^6 h_1 \\
f = 6; e'_{\text{IEDT6}}(x) &= x^6 + x^4 + x && = x(x^5 + x^3 + 1) && = x g_1 \\
f = 7; e'_{\text{IEDT7}}(x) &= x^3 + x^2 + x + 1 && = (x + 1)^3 && = a_1^3 \\
f = 8; e'_{\text{IEDT8}}(x) &= x^6 + x^5 + x^3 + x && = x(x^4 + x + 1)(x + 1) && = x a_1 d_1 \\
f = 9; e'_{\text{IEDT9}}(x) &= x^6 + x^2 && = x^2(x + 1)^4 && = x^2 a_1^4
\end{aligned}$$

Zauważmy, że iloczyn wszystkich dziewięciu wielomianów $e'_{\text{IEDT}f}(x)$ przyjmuje postać wielomianu $x^{18} E_f(x)$, w którym $E_f(x) = a_1^9 b_1^2 d_1 d_3 g_1 g_2 g_5 h_1$ ($\deg E_f(x) = 42$). Ponieważ $\deg E_f(x) < \deg P_n(x)$ dla $n = 5$, dlatego można zastosować technikę dzielenia $P_n(x)$ przez $E_f(x)$. W efekcie otrzymujemy $P_n(x)/E_f(x) =$

$$= a_1 b_1 c_1 c_2 d_1 d_2 g_1 g_2 g_3 g_4 g_5 g_6 / a_1^9 b_1^2 d_1 d_3 g_1 g_2 g_5 h_1 = c_1 c_2 d_2 g_3 g_4 g_6 / a_1^8 b_1 d_3 h_1.$$

Niepodzielnikami są więc wszystkie wielomiany pierwotne, które pozostały w liczniku. Są to wielomiany

$$c_1 = x^3 + x + 1, c_2 = x^3 + x^2 + 1, d_2 = x^4 + x^3 + 1,$$

$$g_3 = (x^5 + x^4 + x^3 + x^2 + 1), g_4 = (x^5 + x^3 + x^2 + x + 1) \text{ oraz}$$

$g_6 = (x^5 + x^4 + x^3 + x + 1)$ związane odpowiednio z następującymi rejestrami IEDT-MISR:

$$\begin{aligned}
&\mathbf{D}(\oplus' \mathbf{D})_{\oplus} \mathbf{D}, \mathbf{D}(\oplus' \mathbf{D}) \mathbf{D}, \mathbf{D}(\oplus' \mathbf{D}) \mathbf{D} \mathbf{D}, \mathbf{D}(\oplus' \mathbf{D}) \mathbf{D}_{\oplus} \mathbf{D} \mathbf{D}, \mathbf{D}(\oplus' \mathbf{D})_{\oplus} \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D}, \\
&\mathbf{D}(\oplus' \mathbf{D})_{\oplus} \mathbf{D}_{\oplus} \mathbf{D} \mathbf{D} \text{ lub } \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D}, \mathbf{D} \mathbf{D} \mathbf{D}, \mathbf{D} \mathbf{D} \mathbf{D} \mathbf{D}, \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D} \mathbf{D}, \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D}, \mathbf{D} \mathbf{D}_{\oplus} \mathbf{D}_{\oplus} \mathbf{D} \mathbf{D}.
\end{aligned}$$

Ostatecznie do analizy sygnaturowej wybieramy trójwejściowy najtańszy z punktu widzenia sprzętu rejestr $\mathbf{D}(\oplus' \mathbf{D}) \mathbf{D}$ lub $\mathbf{D} \mathbf{D} \mathbf{D}$ związany z wielomianem pierwotnym $c_2 = p_b(x) = x^3 + x^2 + 1$.

Niepodzielniki można również znaleźć wśród wielomianów redukowalnych. Spróbujmy je odnaleźć wśród wielomianów związanych z rejestrami IED-MISR. Uwagę naszą ograniczymy tylko do rejestrów stopnia $n \leq 4$ zawartych w zbiorze $R = \{a_1 b_1, a_1 c_1, a_1 c_2\}$. Stosując identyczną do poprzedniej technikę poszukiwania niepodzielników, otrzymujemy zredukowany po dziewiątym kroku zbiór $R^{\text{IX}} = \{a_1 b_1, -, -\}$. Proces poszukiwania zilustrowano poniżej

$$e_{\text{IED1}}(x) = x^5 a_1 d_2; \quad R^{\text{I}} = \{a_1 b_1, a_1 c_1, a_1 c_2\}$$

$$e_{\text{IED2}}(x) = a_1 d_1; \quad R^{\text{II}} = \{a_1 b_1, a_1 c_1, a_1 c_2\}$$

$$e_{\text{IED3}}(x) = a_1 c_2; \quad R^{\text{III}} = \{a_1 b_1, a_1 c_1, -\}$$

$$e_{\text{IED4}}(x) = x^3 a_1^2 c_1; \quad R^{\text{IV}} = \{a_1 b_1, -, -\}$$

$$e_{\text{IED5}}(x) = x^6 c_1^2; \quad R^{\text{V}} = \{a_1 b_1, -, -\}$$

$$e_{\text{IED6}}(x) = x c_1 b_1; \quad R^{\text{VI}} = \{a_1 b_1, -, -\}$$

$$e_{\text{IED7}}(x) = a_1^3; \quad R^{\text{VII}} = \{a_1 b_1, -, -\}$$

$$e_{\text{IED8}}(x) = x b_1 c_2; \quad R^{\text{VIII}} = \{a_1 b_1, -, -\}$$

$$e_{\text{IED9}}(x) = x^2 a_1 c_1; \quad R^{\text{IX}} = \{a_1 b_1, -, -\}$$

Uzyskany niepodzielnik $a_1 b_1$ jest rozkładalnym wielomianem

$p_c(x) = x^3 + 1 = (x^2 + x + 1)(x + 1)$. Koszt realizacji rejestru IED-MISR związanego z wielomianem $p_c(x)$ jest podobny do kosztu sprzętu rejestru IEDT-MISR- $p_b(x)$.

Zauważmy, że iloczyn dwóch wielomianów pierwotnych o nieparzystej liczbie niezerowych współczynników również zawiera nieparzystą liczbę niezerowych współczynników. Wyjaśnia to następujący przykład:

$$(x^2 + x + 1)(x^3 + x + 1) = (x^5 + x^4 + 1) = 1 + x^4(1 + x) \equiv \mathbf{DTDDD}.$$

W związku z tym również w przypadku stosowania kompaktorów IEDT-MISR z komórkami $\oplus' \mathbf{D}$ (\mathbf{T}) ułożonymi w zerowym albo w pierwszym stopniu tych rejestrów można poszukiwać niepodzielników w postaci takich iloczynów.

Znając liczbę m określającą długość testu oraz liczbę w określającą ilość wyjść TUC, można określić jako $s = m + w - 1$ (zob. wyrażenie 2.1) maksymalny możliwy do uzyskania stopień wielomianu błędów $e_{\text{IED}f}(x)$ lub $e_{\text{IEDT}f}(x)$. Załóżmy, że TUC posiada f uszkodzeń. Przyjmijmy również, że $\deg e_{\text{IED}f}(x) = s$ ($\deg e_{\text{IEDT}f}(x) = s$) dla każdego f . Wówczas maksymalny możliwy do uzyskania stopień wielomianu $E_f(x)$ równy jest $\deg_{\max} E_f(x) = f(m + w - 1)$. Jeżeli $\deg P_{n-1}(x) \leq \deg E_f(x) < \deg P_n(x)$, wówczas wśród wielomianów pierwotnych zawartych w iloczynie $P_n(x)$ można znaleźć co najmniej jeden wielomian pierwotny stopnia $\deg p(x) \leq n$ umożliwiający konstrukcję rejestru IED-MISR- $p(x)$ lub IEDT-MISR- $p(x)$ wykrywającego wszystkie f macierze błędów, które pojawiają się na w wyjściach TUC zasilanego ciągiem m testów.

W [LempG95] stwierdzono, że iloczyn liczby uszkodzeń oraz liczby określającej ilość wektorów testowych podawanych w czasie testowania obecnie produkowanych układów scalonych jest zawsze mniejszy od liczby $M = 1.4 \times 10^{16}$. W tej samej pracy udowodniono, że wśród wielomianów pierwotnych stopnia $n \leq 53$ zawsze można znaleźć wielomian umożliwiający zbudowanie rejestru IED-MISR gwarantującego likwidację maskowania błędów. Wynik ten potwierdza następujące rozumowanie. W oparciu o podane w [Gołom82] tablice określające liczebności zbiorów wszystkich wielomianów pierwotnych

danego stopnia n można obliczyć, że $\deg P_n(x) > M$ dla $n = 53$. Zakładając, że iloczyn $f(m+w-1) < M$, otrzymujemy warunek $\deg P_n(x) \leq \deg_{\max} E_f(x) < M < \deg P_n(x)$, który jest spełniony dla $n = 53$.

W oparciu o opisaną metodę poszukiwania niepodzielników można opracować programy komputerowe do określania wielomianów pierwotnych $p(x)$ najniższego stopnia gwarantujących likwidację maskowania błędów zależnych od założonego zbioru modelowanych uszkodzeń. W pracy [LempG95] przedstawiono specjalne procedury przyspieszające proces poszukiwania niepodzielników. Procedury te można stosować niezależnie od rodzaju źródła testów i liczby modelowanych uszkodzeń. Można je stosować w przypadku wszystkich rejestrów c-SISR niezależnie od ich struktury. Chociaż były one tworzone z myślą o projektowaniu sprzężeń liniowych tylko dla rejestrów IED-MISR, to można je również stosować w przypadku rejestrów IEDT-MISR. Dzięki temu powstała nieznana dotąd możliwość badania podzielności co najmniej trzech różnych zbiorów wielomianów błędów $\{e_{IEDf}(x)\}$, $\{e_{IEDTf}(x)\}$ (4.20) oraz $\{e'_{IEDTf}(x)\}$ (4.21) bez konieczności zmiany źródła testów. Pozwala to poszukiwać takiego niepodzielnika, który gwarantuje najtańsze rozwiązanie w zbiorze rejestrów IED-MISR i IEDT-MISR przy raz dobranej sekwencji testów. Likwidacja maskowania błędów w procesie kompaktacji realizowanym przez rejestry c-MISR- $p(x)$ o innych strukturach liniowych sprzężeń zwrotnych jest niezwykle pracochłonna, ponieważ każdy wielomian $p(x)$ wymusza inny zbiór wielomianów błędów $\{e_{cf}(x)\}$. Istnieje więc w tym przypadku problem zmniejszenia pracochłonności związanej z wielokrotnym określaniem zbiorów $\{e_{cf}(x)\}$. Zagadnienie to oczekuje rozwiązania.

Do wyeliminowania maskowania błędów przez kompaktory zapewniające "zero-aliasing" wystarcza analiza jednosygnaturowa. Analiza kSA w przypadku takich kompaktorów jest zbędna, ponieważ jeden pomiar sygnatury gwarantuje wykrycie wszystkich błędów zależnych. Jednakże analiza kSA umożliwia skracanie czasu testowania. Z tego też powodu pożyteczne jest projektowanie weryfikatorów W2 oraz W3 służących do analizy k identycznych sygnatur (zob. rys. 4.7 oraz 4.8) w taki sposób, aby ich kompaktory liniowe likwidowały maskowanie błędów. Wówczas pomiar $k-1$ sygnatur poprzedzających pomiar sygnatury ostatniej (k -tej) umożliwia w razie wykrycia błędów zakończenie testowania przed czasem, natomiast pomiar k -tej sygnatury gwarantuje wykrycie wszystkich błędów zależnych od modelowanych uszkodzeń.

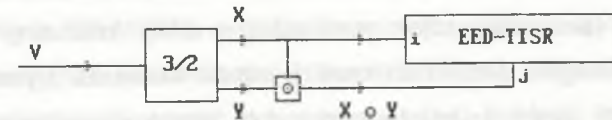
Opisana technika poszukiwania kompaktorów liniowych, które zapewniają brak maskowania błędów zależnych, charakteryzuje się dużą czasochłonnością obliczeń komputerowych. Nawet w przypadku zastosowania w tych obliczeniach procedur przyspieszających podanych w [LempG95] jest ona większa od czasochłonności związanej z komputerową symulacją uszkodzeń oraz z szybką symulacją sygnatur służących do określenia dokładnego wykresu probabilistycznej funkcji $P_{a1}(m)$. Dlatego też od wymagań niezawodnościowych stawianych projektowanemu systemowi komputerowemu zależy ostateczny wybór metody postępowania przy wyborze kompaktorów dla testerów wewnętrznych zastosowanych w projekcie tego systemu komputerowego.

4.3.5. Zmniejszanie maskowania błędów w macierzach odpowiedzi zawierających trójstanowe lub wielowartościowe dane

Kompaktory liniowe ze względu na powszechnie stosowaną w technologii układów ASIC logikę binarną są układami cyfrowymi realizowanymi w logice dwuwartościowej. W związku z tym kompaktacja ciągów binarnych przez takie układy cyfrowe nie stwarza żadnych nie wyjaśnionych dotąd zagadnień związanych ze skutecznością wykrywania błędów. Inaczej ma się rzecz w przypadku kompaktacji przez binarne układy cyfrowe ciągów danych trójstanowych zawierających także stan wysokiej impedancji HZ. Błędy zależne w takich układach wynikają z modeli uszkodzeń dla sieci buforów trójstanowych [Powel96]. Kompaktacja takich błędów stwarza nowe nieznane dotąd problemy. Podobnie będzie także w sytuacji komprimowania ciągów danych trójwartościowych (0,1,2) i ciągów danych czterowartościowych (0,1,2,3) występujących w przypadku stosowania układów ASIC typu BIMLON (Binary Implemented Multi-Valued Logic Network) z binarnie implementowaną logiką wielowartościową [EtieI77, MuziW86, RudeS87, EtieI88, Sasao88]. W podanych układach cyfrowych kompaktacja błędów związanych z niektórymi uszkodzeniami, np. typu skłenie $s-z-x$ gdzie $x \in \{0,1,2,3\}$ stwarza nowe problemy. W celu ich uniknięcia autorzy pracy [SerrM85] zaproponowali zastosowanie liniowego kompaktora o logice wielowartościowej [SerrM85]. Algebraiczny model pracy takiego kompaktora opisywany jest za pomocą pierścienia wielomianów nad ciałem $GF(w)$ [SerrM85], gdzie np. w przypadku logiki czterowartościowej $w = 4$. Ze względu jednak na konieczność binarnej implementacji logiki wielowartościowej racjonalnym rozwiązaniem kompaktacji ciągów wielostanowych (wielowartościowych) danych może być zastosowanie dekodery w/2 ciągu wielowartościowego

wych danych na ciągi binarne komprimowane następnie przez binarne kompaktory liniowe. Przykładem takiego podejścia do kompaktacji ciągów trójstanowych danych jest szeregowy kompaktor w postaci układu dekodera 3/2 - przerzutnik JK - rejestr SISR stosowany powszechnie w multimetrach z analizą sygnaturową 5005B [Hewle97] oraz w analizatorach sygatur typu 5006A [Hewle97], a także w ich odpowiednikach, np. PAS 80 [HławN84, HławN85b]. Innym przykładem jest zastosowany w testerach układów scalonych MASTER 86 i opisany w [MitaH88] równoległy n-stopniowy kompaktor w postaci n dekodów 3/2 z n przerzutnikami JK podłączonych do n wejść rejestru c-MISR. W kompaktorach tych ciągi binarne z wyjść dekodera 3/2 wprowadzane są na wejścia J oraz K przerzutnika JK, który przetwarza je w jeden ciąg binarny wprowadzany na wejście rejestru liniowego SISR lub wejścia rejestru MISR. Taki kompaktor ciągów trójstanowych posiada kilka istotnych wad opisanych w [HławP83, Hławi84c, Hławi86c]. Należy do nich np. tworzenie niepowtarzalnych sygatur w sytuacjach, w których pierwszy bit ciągu trójstanowych danych przyjmuje wartość stanu HZ. Komplikuje to niepotrzebnie proces analizy sygnaturowej. Inną wadą jest utrata istotnych informacji diagnostycznych zawartych w ciągach błędów wskutek ich maskowania przez przerzutnik JK (zob. tab. V w [Hławi86c]) lub ich nieoczekiwanego wstawiania przez ten przerzutnik (zob. tab. V w [Hławi86c]). Może to prowadzić do niewykrycia uszkodzeń typu sklejenie s-z-x ($x \in \{0,1,HZ\}$) lub do niewłaściwej ich lokalizacji. Przyczyną tych niedostatków jest przerzutnik JK, który połączony z dekodrem 3/2 realizuje niedwracalną funkcję przetwarzającą ciągi trójstanowych danych w ciągi binarne. Na przykład każdy ciąg ze zbioru $\{0001H, 00H11, 00H1H, 0H011, 0H01H, 0HH11, 0HH1H, 00011\}$ zawsze jest przetwarzany w ciąg 00011. Konkretnym przykładem uszkodzenia, którego nie można wykryć z tych powodów jest s-z-0 na linii sterującej takim buforem trójstanowym, w którym linia ta jest normalnie aktywna w stanie 0. W przypadku gdy wejście danych takiego bufora i wspomniana linia sterująca są zasilane testami podawanymi z wyjść licznika binarnego (np. w trakcie testu "free run") w taki sposób, że mniej znaczący bit licznika zasila linię sterującą, a bardziej znaczący zasila wejście danych, wówczas uszkodzenie s-z-0 jest niewykrywalne. W podanym przykładzie na wyjściu przerzutnika JK będą identyczne ciągi zarówno w sytuacji obecności wspomnianego s-z-0, jak również w przypadku jego braku. Szeroko te wady omówiono w pracach [Hławi84c, Hławi86c]. Eliminacja tych niedostatków wymaga ingerencji w kolejność testów i zaproponowanie takiej ich sekwencji, aby problem zniknął. To podejście jest niewygodne

i pracochłonne. Dlatego też w pracy [HławP83] usunięto przerzutnik JK z kompaktora i w jego miejsce wprowadzono multiplexer umożliwiający sekwencyjną kompaktację w rejestrze SISR obu ciągów uzyskiwanych na wyjściach dekodera 3/2. Wadą tego rozwiązania jest konieczność podwojenia czasu potrzebnego na kompaktację. Można ten problem wyeliminować, stosując dla obu wyjść dekodera 3/2 dwa oddzielne pomiary sygatur [HławP83, Hławi84c]. Oznacza to jednak konieczność pamiętania dwóch sygatur. Sposób całkowitego wyeliminowania takich niedostatków z binarnie implementowanej kompaktacji ciągów trójstanowych opisano w pracach [Hławi84c, Hławi86c]. Opracowano w nich nowy kompaktor ciągów trójstanowych danych wykorzystujący dekodera 3/2 połączony bezpośrednio z dwuwejściowym rejestrem liniowym EED-TISR. Kompaktor ten przedstawiono na rys. 4.13.



Rys. 4.13. Kompaktor ciągów danych trójstanowych
Fig. 4.13. Compactor of three-state data sequences

Dzięki zastosowaniu na j-tym wejściu rejestru EED-TISR bramki IOR realizującej funkcję $X \odot Y$ ciągi binarne V są komprimowane w sygatury identyczne z sygaturami otrzymywanymi w kompaktorze wykorzystującym przerzutnik JK. Tylko w przypadku wprowadzania ciągów trójstanowych danych stan HZ kodowany jest w stan dwóch zer na wyjściach X oraz Y, które na wyjściu bramki IOR objawiają się w postaci jedynki wprowadzanej do rejestru EED-TISR. W ten sposób wszystkie ciągi z przykładowego zbioru $\{0001H, 00H11, 00H1H, 0H011, 0H01H, 0HH11, 0HH1H, 00011\}$ zostaną odwzorowane w różniące się między sobą sygatury. W pracach [Hławi84c, Hławi86c] przedstawiono algebraiczny model pracy kompaktora dekodera 3/2 - rejestr EED-TISR (zob. 2.6) oraz zbadano przy użyciu tego modelu właściwości detekcyjne tego kompaktora dla niektórych wybranych błędów niezależnych występujących hipotetycznie w komprimowanym ciągu binarnym. W pracach tych zbadano także skuteczność ww. kompaktora w wykrywaniu tzw. błędów zależnych wynikających z przyjętego modelu uszkodzeń typu sklejenie s-z-x, gdzie $x \in \{0,1,HZ\}$ i występujących w rzeczywistości w przekształcanym ciągu trójstanowych danych. Udowodniono, że właściwy dobór wejść i oraz j w rejestrze TISR umożliwia

znaczącą minimalizację zjawiska wzajemnego kasowania się błędów (zob. rozdział 4.2.1), a nawet jego zupełną eliminację. Pozwoliło to na określenie zasad doboru wejść i oraz rejestru EED-TISR oraz opracowanie dla analizatora sygnatur PAS 80 [HławN85a, HławN85b] oraz weryfikatora sygnatur WAS 80 [Hławi88b] konkretnego schematu rejestru EED-TISR łatwo implementowalnego przy użyciu układów scalonych typu MSI i gwarantującego wysoką skuteczność w wykrywaniu ww. błędów zależnych.

Ważnym aspektem zastosowań opisanej w pracach [Hławi84c, Hławi86c] idei wykorzystania rejestru TISR do kompaktacji ciągów trójstanowych danych jest możliwość aplikacji jej do kompaktacji ciągów 3- lub 4-wartościowych danych na wyjściach układów ASIC typu BITLON (ang. Binary Implemented Ternary LOGic Network) [McClu82] lub BIQLON (ang. Binary Implemented Quadruple LOGic Network) [McClu82]. Tego typu układy ASIC stanowią dla dzisiejszych wytwórców układów scalonych alternatywę umożliwiającą wybór projektu gwarantującego większy uzysk powierzchni struktury krzemowej [MuziW86, RudeS87, EtieI88, Sasao88]. Omówienie teorii i techniki stosowania rejestru TISR do szeregowej kompaktacji ciągów 3- lub 4-wartościowych danych można znaleźć w pracy [Hławi84c]. Zastosowanie idei rejestru TISR do budowy rejestru MTISR (Multi-Two Input Signature Register) umożliwiającego równoległą kompaktację wielu ciągów wielowartościowych przedstawiono w pracy [Hławi86f]. Rejestr MTISR może więc stanowić podstawowy blok testera wewnątrzukładowego [Hławi86f]. Technikę stosowania brzegowej ścieżki sterująco-obszernyjnej [IEEE90] dla układów ASIC typu BITLON lub BIQLON i sposobu jej podłączenia do rejestru TISR realizującego kompaktację danych 3- lub 4-wartościowych wprowadzanych do ścieżki brzegowej podczas testowania pakietu zawierającego ww. układy scalone opisano w pracy [BaduH88a]. Problemem wymagającym rozwiązania jest zbadanie skuteczności wykrywania przez rejestr TISR uszkodzeń połączeń pomiędzy układami ASIC typu BITLON lub BIQLON umieszczonych na pakiecie wyposażonym w wyżej opisaną brzegową ścieżkę sterująco-obszernyjną. Ciekawym problemem do rozwiązania jest także wskazanie, która ze struktur "c" rejestrów c-TISR nadaje się najlepiej do kompaktacji ciągów danych 3- lub 4-wartościowych.

5. PODSUMOWANIE

Rozwój teorii i technik wykorzystywania rejestrów liniowych LFSR, SISR oraz MISR do upraszczania problemów związanych z testowaniem, a zwłaszcza do testowania wewnątrzukładowego, rozpoczął się w drugiej połowie lat siedemdziesiątych [Benow75, Frohw77, David78, KoneM79]. Rozwój ten trwa nadal, czego praktycznym przykładem jest nowy mikroprocesor RISC typu Alpha 21164, w którym do szybkiej równoległej kompaktacji sygnałów pochodzących z ponad 550 wewnętrznych węzłów zastosowano 27 różnych rejestrów liniowych IED-MISR o specjalnej konstrukcji przedstawionej w [BhavE97].

Niniejsza monografia jest skróconym opisem wkładu autora w ogólnowiedowy dorobek w dziedzinie wspomnianej powyżej, dokonanego w okresie od roku 1981 [HławK81a] do chwili obecnej [HłGS97]. W szczególności dotyczy on analizy zjawiska maskowania błędów przez rejestry liniowe SISR, TISR i MISR oraz takiego ich projektowania i stosowania, aby analizę sygnaturową odpowiedzi testowanych układów cyfrowych cechowała wysoką skuteczność. Zasadniczym tematem pracy jest próba wskazania zasad wyboru właściwego sprzężenia liniowego pozwalającego na efektywne implementowanie rejestru liniowego w technologii, w której są realizowane np. niektóre programowalne układy FPGA, a także umożliwiającego uzyskanie testowania i samotestowania o wysokiej jakości i o niskim koszcie. Pokazano, że właściwy wybór sprzężenia rejestru liniowego zależy nie tylko od postaci założonego wielomianu charakterystycznego związanego z tym sprzężeniem, ale w wielu wypadkach także od rodzaju struktury wybranej do realizacji tego sprzężenia.

Opracowany w postaci dzielenia wielomianów algebraiczny opis pracy rejestrów liniowych o różnych, wynikających z tego samego wielomianu charakterystycznego, strukturach sprzężeń liniowych (rys. 2.7) umożliwił określenie wpływu tych struktur na:

- proces generacji testów pseudolosowych przez rejestry LFSR (2.7),

- proces kompaktacji realizowany przez rejestry SISR (2.5), TISR (2.6) oraz MISR (2.1), a także ich wpływu na
- zależność pomiędzy resztą z dzielenia a stanem wewnętrznym rejestrów (2.11).

Zaproponowano techniki projektowania w oparciu o założony wielomian charakterystyczny wielu różnych struktur sprzężeń liniowych, np. IED, EED, BTB, TBD oraz wielu nowych nieznanych dotąd struktur zawierających także przerzutniki T, np. IEDT, EEDT, DT, BTDT, TBDB. Polegają one na przekształcaniu źródłowego wielomianu charakterystycznego $p(x)$ w tzw. strukturalną jego postać $p_{cw}(x)$ (tabela 3.1), która jest ściśle skorelowana ze schematem ideowym struktury projektowanego rejestru liniowego i z której wprost można ten schemat odczytać. Dzięki temu można również na podstawie schematu struktury rejestru liniowego bardzo prosto określić wielomian charakterystyczny związany ze sprzężeniem liniowym.

Biorąc pod uwagę kryteria przydatności wielomianów charakterystycznych w procesie kompaktacji, zaproponowano metodę szybkiego projektowania takich ciągów komórek rejestrów liniowych typu CA, które są związane z wielomianami nieułamnymi i nieparzystymi (graf G_u/G_p na rys. 3.4). Umożliwiło to częściową redukcję trudności, które ze względu na iteracyjny sposób określania wielomianu (tabela 2.1) związanego z rejestrem o strukturze CADT występują podczas projektowania tego rejestru liniowego. Dalszą redukcję czasochłonności związanych z projektowaniem takich rejestrów uzyskano dzięki opracowaniu metody wyboru wielokomórkowych modułów plasterkowych (twierdzenie 3.1 oraz wniosek 3.3) służących do efektywnego ich sklejanie w łańcuchy komórek typu CA związanych z nieułamnymi i nieparzystymi wielomianami charakterystycznymi. Opracowane na bazie tej metody specjalne programy umożliwiają, w oparciu o biblioteki wybranych wielokomórkowych modułów plasterkowych, sklejanie ich w ciągi komórek rejestrów liniowych typu CA przydatnych w testowaniu wewnątrzukładowym.

Zastąpiono graf przejść opisujący funkcje przejść rejestru liniowego LFSR, SISR i MISR o dowolnej strukturze specjalną techniką określania ich stanów wewnętrznych wynikającą wprost z algebraicznego opisu ich pracy i schematu zastępczego przedstawionego na rys. 2.7. Technika ta umożliwia znaczne zmniejszenie trudności związanych z określaniem stanu końcowego rejestru liniowego przy znanym jego stanie początkowym i znanej sekwencji wektorów wejściowych (2.8). Umożliwia ona także stosunkowo proste określanie stanu początkowego, który należy wprowadzić do rejestru liniowego, aby po podaniu zna-

tego ciągu wektorów wejściowych uzyskać pożądany stan końcowy (2.9) oraz określanie sekwencji wejściowej, która ustawia rejestr liniowy w pożądany stan końcowy przy założonym dowolnym stanie początkowym (2.10). Technika ta pozwala także na realizację procesu konwersji (podrozdział 2.3.2) stanów wewnętrznych rejestrów liniowych o różnych strukturach związanych z tym samym wielomianem charakterystycznym. Wszystkie opisane zastosowania wspomnianej techniki są bardzo przydatne w czasie projektowania testerów wewnątrzukładowych, o czym wielokrotnie wspomniano w podrozdziale 2.3 oraz w wielu miejscach rozdziału 4. Przedstawione w podrozdziale 2.3.2 techniki określania sygnatur oraz techniki ich konwersji mogą być podstawą do opracowania algorytmu umożliwiającego szybkie wyznaczanie sygnatur dla kompaktorów SISR oraz MISR o dowolnej strukturze sprzężeń liniowych.

Część zaproponowanych w tej pracy struktur sprzężeń liniowych, w porównaniu ze strukturami klasycznymi np. typu IED, zawiera pętle ze znacznie zredukowanym współczynnikiem rozptywu (fanout). Wyraźnie ilustrują to przykłady 3.14, 3.15, 3.19 oraz tabele 3.2, 3.4 i 3.5. Możliwość zredukowania wartości współczynnika rozptywu pozwala na dopasowanie struktury sprzężenia liniowego do tych reguł projektowania układów FPGA, które dzięki tej redukcji pozwalają uzyskać duży współczynnik wydajności [Hławi97a]. Dobrym tego przykładem są układy FPGA firmy ACTEL, w których czas opóźnienia pomiędzy dwoma logicznymi modułami CLB zwiększa się, między innymi wraz ze wzrostem liczby sterowanych wejść [ACTEL, Hławi97a]. Możliwość projektowania struktur sprzężeń liniowych ze zredukowanym współczynnikiem rozptywu wętli pozwala na stosowanie w testowaniu wewnątrzukładowym sprzężeń liniowych związanych z wielomianami charakterystycznymi, które dotąd, ze względu na zbyt dużą liczbę niezerowych współczynników tego wielomianu, były a priori odrzucane (przykłady 3.14, 3.15, 3.19, tabela 3.2). Ma to szczególnie istotne znaczenie w metodzie likwidacji maskowania błędów, w której znaleziony niepodzielnik może okazać się wielomianem zawierającym dużą liczbę niezerowych współczynników. Ma to także ważne znaczenie w przypadku poszukiwania generatora krótkich sekwencji testów pseudolosowych zawierających wszystkie testy deterministyczne. Rejestry liniowe o sprzężeniach IEDT, ze względu na podane zalety, można także wykorzystywać do projektowania, pracujących w cyklu 2^n , liczników liniowych [Hławi97a] stosowanych do budowy układów sterujących testowaniem wewnątrzukładowym.

Za pomocą opisanej w rozdziale 3.2.6, techniki projektowania rejestrów liniowych w formie przeplatanki można zlikwidować wszystkie długie połączenia. Technika ta pozwala więc zmniejszać opóźnienia wnoszone przez pętle. Wyjaśniono to w przykładzie 3.22.

Również problem nadmiernej liczby bramek XOR w tego typu sprzężeniach został w tej pracy rozwiązany. Przedstawione w rozdziale 3 techniki projektowania umożliwiają redukcję liczby bramek XOR w strukturach sprzężeń liniowych. Ilustruje to cały rozdział 3.2, a w szczególności przykłady 3.14, 3.15, 3.19, tabele 3.4 i 3.5, a także tabela 3.2 w rozdziale 3.1.

W tab. 5.1 przedstawiono różne strukturalne postaci wielomianów charakterystycznych $p_{cw}(x)$ umożliwiających zaprojektowanie rejestrów liniowych c-LFSR-p(x), c-SISR-p(x) oraz c-MISR-p(x) o takich strukturach sprzężeń liniowych, które gwarantują jednoczesne uzyskanie niskiego współczynnika rozplywu WR, minimalnej liczby LX bramek XOR w sprzężeniu, minimalnego czasu propagacji CZ (liczba warstw bramek XOR), a także likwidację długich połączeń (LP = 0). W tabeli tej podano także liczby określające LX, WR, LP oraz CZ dla różnych realizacji struktur sprzężeń liniowych. Zauważmy, że parametry LX, WR, LP, CZ projektu rejestru liniowego opartego na strukturach wynikających z wielomianów $p_{cw}(x)$ podanych w wierszach od 1 do 13 są na ogół lepsze od parametrów projektów opartych na klasycznych strukturach IED oraz EED podanych w wierszach 16 i 17 tej tabeli. Zdecydowanie najlepsze parametry zapewniają te projekty rejestrów liniowych, w których przerzutniki T nie muszą być budowane w postaci przerzutnika D z bramką XOR na wejściu (wiersze 1,4-6) oraz te projekty, które zawierają tylko jedną bramkę XOR w sprzężeniu (wiersze 2 i 3). Tab. 5.1 może stanowić dla konstruktora układów cyfrowych pewnego rodzaju zbiór wskazówek do projektowania optymalnej struktury sprzężenia liniowego na podstawie założonego wielomianu charakterystycznego $p(x)$. Niestety większość wielomianów strukturalnych $p_{cw}(x)$ zawartych w tej tabeli można wyprowadzić z wielomianów charakterystycznych $p(x)$ zawierających małą liczbą niezerowych współczynników. W przypadku wielomianów z dużą liczbą niezerowych współczynników należy sięgnąć do metod projektowania przedstawionych w rozdziale 3.

Struktury sprzężeń liniowych typu IEDT oraz IED są najlepiej dostosowane do układów FPGA (ACT-3) firmy ACTEL, co częściowo udokumentowano na podstawie [Hławi97a] w podrozdziale 3.1.7. Podobnie najlepiej dopasowane do implementacji w układach FPGA firmy ALTERA są projekty rejestrów MISR-MR o strukturach IEDT [HławB94a,

HławB94b]. Z punktu widzenia nadmiaru układowego stosowanie w obu przypadkach komórkowych bezpętlowych rejestrów liniowych typu CA jest nieopłacalne. Podane w pracy [Hławi97a] przykłady komórek standardowych do realizacji różnych struktur sprzężeń liniowych wskazują wyraźnie, że układy ACT-3 zdecydowanie najgorzej dostosowane są do realizacji wszelkich komórkowych rejestrów liniowych CA. Przyczyną jest konieczność dodawania, w porównaniu z rejestrami liniowymi o strukturach IEDT oraz IED, dodatkowych modułów "c" typu AX1C [Hławi97a] realizujących funkcje XOR. Jest to wyraźnie sprzeczne z regułami optymalnego projektowania narzucającymi w przypadku układów FPGA firmy ACTEL konieczność zmniejszania liczby warstw logiki i liczby modułów CLB.

Tabela 5.1

Lp.	Struktura c	Realizacja przerz. T	$P_{cw}(x)$	LX	WR	LP	CZ
1	DT	T	$1+x^b(1+x)^f$	0	1	0	0
2	IED, EED	-	$1+x^b[1+x^c]$	1	2	0	1
3	DT	$T=(\oplus'D)$	$1+x^b(1+x)$	1	2	0	1
4	IEDT, EEDT	T	$1+x^b(1+x)^f[1+x^c]$	1	2	0	1
5	IEDT, EEDT	T	$1+x^b[1+x^c(1+x)^g]$	1	2	0	1
6	IEDT, EEDT	T	$1+x^b(1+x)^f[1+x^c(1+x)^g]$	1	2	0	1
7	DT	$T=(\oplus'D)$	$1+x^b(1+x)^2$	2	2	0	1
8	IEDT, EEDT	$T=(\oplus'D)$	$1+x^b(1+x)[1+x^c]$	2	2	0	1
9	IEDT, EEDT	$T=(\oplus'D)$	$1+x^b[1+x^c(1+x)]$	2	2÷3	0	1÷2
10	TBD	-	$1+[x^a[1+x^b]][1+x^c]$	2	2	1÷2	1
11	BTD	-	$x^n+[1+x^a][1+x^b]$	2	2	1÷2	1
12	DT	$T=(\oplus'D)$	$1+x^b(1+x)^3$	3	2	0	1
13	IEDT, EEDT	$T=(\oplus'D)$	$1+x^b(1+x)^2[1+x^c]$	3	2	0	1
14	IEDT, EEDT	$T=(\oplus'D)$	$1+x^b[1+x^c(1+x)^2]$	3	2÷3	0	1÷2
15	IEDT, EEDT	$T=(\oplus'D)$	$1+x^b(1+x)[1+x^c(1+x)]$	3	2÷3	0	1÷2
16	IED	-	$1+x^a[1+x^b[1+x^c[1+x^d]]]$	3	4	0÷1	1
17	EED	-	$1+x^a[1+x^b[1+x^c[1+x^d]]]$	3	2	1÷4	3

W pracy omówiono także zagadnienie projektowania rejestrów liniowych o strukturach sprzężeń liniowych mało wrażliwych na modyfikacje w czasie prototypowania logiki funkcjonalnej TUC. Modyfikacje te są głównie związane ze zwiększaniem lub zmniejszaniem liczby wejść i wyjść pierwotnych TUC i na ogół wymagają w takim przypadku całkowitej wymiany rejestru liniowego. Dotąd wydawało się, że komórkowe rejestry liniowe CA są najlepiej dostosowane do takich zmian. W rozdziale 3.4 zbadano wrażliwość komórkowych rejestrów CA (tabela 3.13) oraz rejestrów o strukturach IEDT i EEDT na modyfikacje związane ze zmianą liczby wyprowadzeń TUC. Wykazano w nim na konkretnych przykładach, że w porównaniu z rejestrami typu CADT rejestry liniowe o nowych strukturach IEDT (rys. 3.9) czy też EEDT (rys. 3.10) są równie mało wrażliwe na modyfikacje logiki funkcjonalnej. Wydaje się, że wśród tych rejestrów należy szukać struktur o najmniejszej wrażliwości.

Odrębnym zagadnieniem opracowanym w tej pracy jest opis algebraiczny pracy rejestrów COPMISR przechowujących rozdzielny kod liniowy (rozdział 2.4) oraz technika ich projektowania (rozdział 3.5). Zastosowanie tego rejestru do budowy samotestowalnych układów samokontrolujących się (self-testing self-checking checker) i wykorzystywanie tych układów w wysoce niezawodnych systemach tolerujących uszkodzenia opisano szczegółowo w pierwszym rozdziale tej pracy (rys. 1.10). Rejestr ten wraz z układem samokontrolującym się (rys. 1.11) może być także wykorzystywany np. do budowy rejestrów COPBILBO (COde Preserving BILBO) lub COPCBILBO (COde Preserving CBILBO), w których błędy sygnalizuje sygnatura lub wyjścia r_1 , r_2 . Mogą być one stosowane, podobnie jak to pokazano dla konwencjonalnych układów kombinacyjnych w rozdziale 1 (na rys. 1.2 oraz rys. 1.3), w testowaniu wewnątrzukładowym układów kombinacyjnych zaprojektowanych z myślą o wykrywaniu ich uszkodzeń podczas normalnej pracy [Touba94].

Opracowane wcześniej przez autora zagadnienia związane z wykorzystaniem rejestrów liniowych SISR, TISR oraz MISR do kompaktacji odpowiedzi testowanych układów cyfrowych zarówno w testerach wewnątrzukładowych, jak również przy użyciu analizatorów i weryfikatorów sygnatur, usystematyzowano, uzupełniono i rozwinięto w rozdziale 4. W rozdziale tym określono dwie przyczyny maskowania błędów i wykorzystano statyczne prawdopodobieństwo tego maskowania ($P_{al} = 2^{-n}$) jako podstawową miarę maskowania błędów. Na jej podstawie i na podstawie wynikającego z wyrażenia 2.1 algebraicznego

opisu kompaktacji liniowej określono kryteria wyboru sprzężeń gwarantujących minimalne maskowania błędów w procesie kompaktacji. Pozwoliły one, w pierwszej fazie badań autora nad skutecznością, ograniczyć zbiór przydatnych w testowaniu wewnątrzukładowym wielomianów charakterystycznych do zbioru nieułamnych rozkładalnych i pierwszych wielomianów związanych z różnymi strukturami rejestrów SISR (wniosek 4.3) oraz zbioru nierozkładalnych wielomianów związanych z różnymi strukturami rejestrów MISR (wnioski 4.4 oraz 4.5). W drugiej fazie tych badań, na podstawie wprowadzonego modelu błędów (definicja 4.1 modelu standardowej klasy uszkodzeń), uściślonego wyrażenia (4.2) na prawdopodobieństwo maskowania tych błędów oraz wykresów ilustrujących dynamiczny przebieg prawdopodobieństwa ich maskowania ($P_{al}(m)$), wskazano wielomiany pierwotne (wnioski 4.6 i 4.8) jako najbardziej przydatne do projektowania skutecznych kompaktorów liniowych. Udowodniono także, że różnią się między sobą zbiory macierzy błędów niewykrywanych przez rejestry MISR o różnych strukturach sprzężeń liniowych, chociaż związanych identycznym wielomianem charakterystycznym. To odkrycie okazało się bardzo przydatne podczas badań nad zupełną likwidacją maskowania błędów.

W większości zastosowań testerów wewnątrzukładowych akceptowane jest maskowanie błędów wynikające z zastosowania sprzężenia liniowego związanego z wielomianem pierwotnym. Jednak dla wysoce wiarygodnych systemów komputerowych poszukuje się rozwiązań w znaczący sposób pomniejszających to prawdopodobieństwo maskowania błędów. Cały podrozdział 4.3 jest poświęcony temu zagadnieniu. Omówiono w nim różne techniki analizy k-sygnaturowej oraz analizy sygnaturowej połączonej z analizą ilorazową. Wszystkie te metody umożliwiają sprowadzenie zjawiska maskowania błędów do poziomu, w którym np. na 2.136×10^{96} macierzy błędów w przypadku 20-krotnego odczytywania sygnatur w 16-bitowym kompaktorze tylko jedna macierz błędów może zostać niewykryta. Techniki k-sygnaturowej analizy bez redundancji czasu umożliwiają ponadto w przypadku TUC ze zbyt małą liczbą wyjść pierwotnych uzyskanie skuteczności osiągananej normalnie dla wielobitowych kompaktorów. Umożliwiają one również znaczne skrócenie czasu testowania oraz zwiększenie rozdzielczości uszkodzeń.

Najlepsze rozwiązania w przypadku systemów komputerowych o wysokiej niezawodności powinny zupełnie likwidować zjawisko maskowania błędów. Ten problem szczególnie omówiono w podrozdziale 4.3.4. Zaproponowano w nim między innymi metodę poszukiwania niepodzielników wielomianów błędów wśród wielomianów charakte-

rystycznych związanych ze strukturami kompaktorów IEDT-MISR. Udowodniono, że poszukiwanie niepodzielników dla tych kompaktorów może być realizowane w prosty sposób przy raz dobranej sekwencji testów gwarantujących wysoki współczynnik pokrycia uszkodzeń.

Rezultaty badań wpływu postaci wielomianu charakterystycznego $p(x)$ oraz struktury c sprzężenia liniowego na skuteczność wykrywania błędów przez rejestry liniowe c -SISR- $p(x)$ oraz c -MISR- $p(x)$ zestawiono w tab. 5.2. W kolumnie 2 tej tabeli wskazano te wielomiany charakterystyczne $p(x)$ i wynikające z nich struktury sprzężeń liniowych, które gwarantują uzyskanie przez n -bitowe rejestry liniowe statycznego prawdopodobieństwa maskowania błędów $P_{al} = 2^{-n}$. W kolumnie 3 tej tabeli podano natomiast, które z wielomianów $p(x)$ ze zbioru wskazanego w kolumnie 2 i jakie związane z nimi struktury sprzężeń zapewniają taki przebieg krzywej dynamicznego prawdopodobieństwa maskowania błędów $P_{al}(m)$, który najszybciej zbiega do poziomu $P_{al} = 2^{-n}$. Wreszcie kolumna 4 tej tabeli sygnalizuje, że do zaprojektowania kompaktora liniowego wykrywającego wszystkie błędy ($P_{al} = 0$) można wybrać każdy wielomian charakterystyczny $p(x)$, który okaże się niepodzielnikiem zbioru wielomianów błędów. Tab. 5.2 podobnie jak i tab. 5.1 może stanowić pewną formę uproszczonego poradnika dla projektanta samotestowalnych układów cyfrowych. Aspekty ekonomiczne testowania ostatecznie zdecydują o tym, która z kolumn tab. 5.2 będzie brana pod uwagę przy wyborze wielomianu $p(x)$ do projektu kompaktora liniowego.

W podrozdziale 4.3.5 w skrócie opisano zastosowanie rejestru TISR (MTISR) do rozwiązania niektórych problemów związanych ze zmniejszaniem maskowania błędów w macierzach odpowiedzi TUC zawierających bądź to trójstanowe dane charakterystyczne dla buforów trójstanowych wprowadzających stan wysokiej impedancji HZ, bądź też wielowartościowe dane charakterystyczne dla układów ASIC typu BIMLON z binarnie implementowaną logiką wielowartościową.

Zaproponowane w tej pracy nowe struktury sprzężeń rejestrów liniowych zawierających także przerzutniki T umożliwiają tworzenie znacznie liczniejszych zbiorów rejestrów liniowych generujących różne ciągi testów pseudolosowych PR. Pewnym tego przykładem jest tabela 2.4. Zbiory te zwiększają szansę wyboru struktury sprzężenia rejestru liniowego LFSR generującego wszystkie niezbędne testy deterministyczne w znacznie krótszej sekwencji pseudolosowej. Pozwoliłoby to na lepsze dopasowanie rejestrów liniowych do

Kompaktor Liniowy		$P_{al} = 2^{-n}$	$P_{al}(m)$	$P_{al} = 0$
1		2	3	4
c-SISR- $p(x)$	c	dowolna struktura sprzężenia liniowego	dowolna struktura sprzężenia liniowego	dowolna struktura sprzężenia liniowego
	$p(x)$	dowolny wielomian pierwszy lub rozkładalny	dowolny wielomian pierwotny lub oparty na kodach Fire'a	dowolny wielomian będący niepodzielnikiem wielomianów błędów
c-MISR- $p(x)$	c	dowolna struktura sprzężenia liniowego	dowolna struktura sprzężenia liniowego	IED, IEDT
	$p(x)$	dowolny wielomian pierwszy	dowolny wielomian pierwotny	dowolny wielomian będący niepodzielnikiem wielomianów błędów

potrzeb związanych ze skracaniem czasu testowania i zmniejszaniem nadmiaru układowego (zmniejszaniem kosztów samotestowania) przy jednoczesnym zachowaniu wysokiej jakości tego testowania. Zachęcające rezultaty w tym zakresie uzyskano w pracy [Gucwa94], w której krzywe określające zależność współczynnika pokrycia uszkodzeń od liczby podanych testów pseudolosowych generowanych przez rejestry IEDT-LFSR dla wybranych układów ISCAS (6228, C1908) wyraźnie wskazują na możliwość skrócenia czasu testowania. Te nowe struktury sprzężeń liniowych powinny więc stanowić podstawę do dalszych badań w dziedzinie projektowania liniowych generatorów testów deterministycznych. Powinny one być również uwzględnione przy tworzeniu nowych narzędzi do automatycznej syntezy samotestowania wykorzystujących podane w [LempG94] techniki optymalnego wyboru rejestru LFSR umożliwiającego wygenerowanie założonego zbioru testów deterministycznych w najkrótszym możliwym do uzyskania czasie. Powinny być kontynuowane prace teoretyczne nad zastosowaniami rejestrów liniowych IEDT-MISR z przerzutnikami T do kompaktacji odpowiedzi TUC bez maskowania błędów. Ma to szczególne znaczenie w przypadku stosowania, zawartych w samotestowalnych układach samokontrolujących się, rejestrów COPMISR, w których rejestr EED-MISR zastąpiono rejestrem IEDT-MISR (macierz C przedstawiona w 2.22 oraz 2.23). Dalsze prace w tej dziedzinie powinny zostać skoncentrowane nad zastosowaniem procedur przyspieszonego poszukiwania niepodzielników wielomianów błędów [LempG95] do stworzenia narzędzi programowych do automa-

tycznej syntezy samotestowania, umożliwiającej wykorzystywanie rejestrów IEDT-MISR do kompaktacji bez maskowania błędów, która - co należy mocno podkreślić - jest niezależna od długości sekwencji testów podawanych na wejścia TUC. Wspomniana niezależność pozwala na równoczesne stosowanie rejestrów LFSR (między innymi z przerzutnikami T) do efektywnej generacji testów deterministycznych oraz rejestrów IED-MISR oraz IEDT-MISR do kompaktacji odpowiedzi TUC bez maskowania błędów.

Nadal oczekuje rozwiązania problem eliminacji maskowania błędów spowodowanych uszkodzeniami w sieciach połączeń pomiędzy układami ASIC wyposażonych w brzegową ścieżkę sterująco-obszerną z wbudowanym weń kompaktorem liniowym typu IEDT-MISR lub IED-MISR. W rozdziale 4.3.4 wykazano, że w tym zakresie badań istnieje możliwość wyprowadzania ogólniejszych wniosków związanych z niepodzielnymi wielomianów błędów.

Przedstawione we wprowadzeniu techniki wykorzystania rejestrów liniowych w testowaniu wewnątrzukładowym stanowią przykład wyspecjalizowanych technik samotestowania. Przyszłe prace, podobnie jak uczyniono to już w [Kraśn96] dla techniki pierścienia testującego, należałoby również skoncentrować nad opracowaniem technik uniwersalnego stosowania rejestrów LFSR, SISR oraz MISR, o różnych strukturach sprzężeń liniowych, do testowania w jednym układzie nie tylko wielu klasycznych układów kombinacyjnych i sekwencyjnych, ale także np. struktur pamięci RAM itp.

Nowe zaproponowane w niniejszej pracy struktury sprzężeń rejestrów liniowych, ich algebraiczny opis pracy, opracowane nowe techniki ich projektowania, wspomniana poprzednio możliwość znaczącego skracania długości sekwencji pseudolosowych zawierających testy deterministyczne oraz możliwość zupełnej likwidacji maskowania błędów stanowią dobrą podstawę do prowadzenia dalszych badań naukowych w dziedzinie zastosowań rejestrów liniowych w testowaniu. Ich prawdopodobnym efektem mogą być optymalne techniki testowania na bieżąco i testowania wewnątrzukładowego o wysokiej jakości i małym koszcie i rozwiązujących w sposób kompletny i praktyczny większość problemów związanych z upraszczaniem testowania projektowanych systemów i układów cyfrowych o wielkiej skali integracji. Autor ma także nadzieję, że część rezultatów tej pracy będzie mogła być również wykorzystana w innych dziedzinach, w których stosuje się rejestry liniowe.

BIBLIOGRAFIA

- [AbraB90] Abramovici M, Breuer M.A., Friedman A.D.: Digital Systems Testing and Testable Design. IEEE Press, 1990.
- [ACTEL] ACTEL - FPGA Data Book and Design Guide. 1994.
- [AgraK93a] Agrawal V.D., Kime C.R., Saluja K.K.: A Tutorial on Built-In Self-Test - Part 1: Principles. IEEE Design and Test of Computers, March 1993, vol. 10, No 1, str. 73-82.
- [AgraK93b] Agrawal V.D., Kime C.R., Saluja K.K.: A Tutorial on Built-In Self-Test Part 2: Applications. IEEE Design and Test of Computers, June 1993, str. 69-77.
- [AhmN90] Ahmad A., Nanda N.K., Garg K.: Are Primitive Polynomials Always Best in Signature Analysis? IEEE Design and Test of Computers, August 1990, str. 36-38.
- [ALTERA] ALTERA; MAX 7000 Data Book. 1993.
- [AshaR78] Ashajee M.J., Reddy S.M.: On Totally Self-Checking Checkers for Separable Codes. IEEE Trans. on Computers, vol. C-26, No 8, August 1978, str. 737-744.
- [BaduH88a] Badura D., Hławiczka A.: Multi-Valued Logic Circuit Testing with Boundary Scan Path. Proc. of 11th Conference on FTSD, Suhl, June 1988, str. 253-258.
- [BaduH88c] Badura D., Hławiczka A.: A Linear Feedback Shift Register as Modifier to Optimize Error Masking In-Built Self-Testing. Proc. of 11th Conference on FTSD, Suhl, June 1988, str. 209-214.
- [BaduH96a] Badura D., Hławiczka A.: Condensed Circular Self-Test Path: A Low-Cost Circular BIST. Proc. of IEEE European Test Workshop - ETW'96, June 12-14 1996, Sete, Francja, str. 65-69.
- [BaduH96b] Badura D., Hławiczka A.: How to Reduce Cost of Circular Self-Test Path. Proc. of EDCC-2 Companion Workshop, May 15, Gliwice 1996, str. 147-152.
- [BaduH96c] Badura D., Hławiczka A.: Skondensowany pierścień testujący: Nowa niedroga technika BIST. ELEKTRONIKA nr 8, 1996, str. 17-20.
- [BaduH97] Badura D., Hławiczka A.: Low Cost BIST for EDA/C Circuits. Proc. of the Sixth Asian Test Symposium ATS'97, November 1997, Akita, Japan, IEEE Computer Society Press.

- [Badur89] Badura D.: Efficiency of Self-Test Path as a Test Pattern Generator and Test Response Compactor. Proc. 1989 4th International FTCS Conference 89, Baden-Baden, September 1989, str. 365-375.
- [Badur90a] Badura D.: Self-Test Path in Self-Diagnostic Systems. Proc. of COGNITIV'90, Madryt, November, 1990.
- [Badur90b] Badura D.: Design of Circuit with Self-Test Path. Proc. of 13th FTSD, Varna, June 1990.
- [Badur92] Badura D.: Techniki projektowania samotestowalnych układów i pakietów cyfrowych wykorzystujące rejestry szeregowo z nieliniowym sprzężeniem zwrotnym. Prace Naukowe Uniwersytetu Śląskiego nr 1280, Katowice 1992.
- [Barde90] Bardell P.H.: Analysis of Cellular Automata Used as Pseudorandom Pattern Generators. Proc. 1990 IEEE International Test Conference, 1990, str. 762-768.
- [BardM82] Bardell P.H., McAnney W.H.: Self-Testing of Multichip Logic Modules. Proc. 1982 International Test Conference, 1982.
- [BardM83] Bardell P.H., McAnney W.H.: Self-Testing of Multichip Logic Modules. Test and Measurement World, March 1983, str. 26-29.
- [BardM86] Bardell P.H., McAnney W.M.: Pseudorandom Arrays for Built-In Test. IEEE Transactions on Computers, vol. C-35, No 7, July 1986, str. 653-662.
- [BardM87] Bardell P.H., McAnney W.H., Savir J.: Built-In Test for VLSI: Pseudorandom Technique. John Wiley & Sons Press, 1987.
- [Beenk87] Beenker F.P.M.: Systematic and Structured Methods for Digital Board Testing. VLSI Systems Design, January 1987, str. 52-58, a także Proc. 1985 International Test Conference, str. 380-385.
- [Benne84] Bennets R.G.: Design of Testable Logic Circuits. Addison-Wesley Publ. Comp. 1984.
- [Benow75] Benowitz N. et al: An Advanced Fault Isolation System for Digital Logic. IEEE Transactions on Comput, vol. C-24, May 1975, str. 489-497.
- [BhavE97] Bhavsar D.K., Edmondson J.H.: Alpha 21164 Testability Strategy. IEEE Design and Test of Computers, vol. 14, No 1, January-March, 1997, str. 25-33.
- [BhavH81] Bhavsar D.K., Heckelman R.W.: Self-Testing by Polynomial Division. Proc. 1981 IEEE International Test Conference, 1981, str. 208-216.
- [BhavK84] Bhavsar D.K., Krishnamurthy B.: Can We Eliminate Fault Escape In-Self Testing by Polynomial Division (Signature Analysis)? Proc. 1984 International Test Conference, 1984, str. 134-139.
- [Bhavs85] Bhavsar D.K.: Concatenable Polydividers: Bit-Sliced LFSR Chips for Board Self-Test. Proc. IEEE International Test Conference, 1985, str. 88-93.

- [BoubK93] Bouberazi S., Kaminska B.: Cellular Automata Synthesis Based on Precomputed Tests Vectors for Built-In Test. International Conference on CAD, 1993, str. 578-585.
- [BrynA90] Brynestad O., Aas E.J., Vallestad A.E.: State Transition Graph Analysis as Key to BIST Fault Coverage. Proc. of International Test Conference, 1990, str. 537-543.
- [Buckl75] Buckley J.E.: IBM Protocols Part2: SDLC. Computer Design, February, 1975, str. 14-16.
- [Budko79] Budkowski S. i inni: Zespoły i urządzenia cyfrowe. WNT, Warszawa 1979.
- [CattM96] Cattel K., Muzio J.C.: Synthesis of One-Dimensional Linear Hybrid Cellular Automata. IEEE Transactions on CAD of Integrated Circuits and Systems, vol. 15, No 3, March 1996, str. 325-335.
- [ChakH93] Charabarty K., Hayes J.P.: Aliasing-Free Error Detection (ALFRED). Proc of VLSI Test Symposium, 1993, str. 260-266.
- [ChuaG89] Chuang C.C., Gupta A.K.: The Analysis of Parallel BIST by the Combined Markov Chain (CMC) Model. Proceedings of International Test Conference, 1989, IEEE Computer Society Press, str. 337-343.
- [CoIAR88] Cox H., Ivanov A., Agarwal V.K., Rajski J.: On Multiple Fault Coverage and Aliasing Probability Measures. Proc. 1988 IEEE International Test Conference, September 1988, IEEE Computer Society Press, str. 314-321.
- [DamiO89a] Damiani M., Olivo P., Favalli M., Ercolani S., Ricco B.: Aliasing in Signature Analysis Testing with Multiple-Input Shift-Registers. Proc. of 1 st European Test Conference, Paris, IEEE Computer Society Press, April 1989, str. 346-353.
- [DamiO89b] Damiani M., Olivo P., Favalli M., Ricco B.: An Analytical Model for the Aliasing Probability in Signature Analysis Testing. IEEE Transactions on Computer-Aided Design, vol. 8, No 11, November 1989, str. 1133-1144.
- [DamiO90] Damiani M., Olivo P. et al.: Aliasing in Signature Analysis Testing with Multiple Input Shift Registers. IEEE Trans. on Computer-Aided Design, vol. 9, No 12, December 1990, str. 1344-1353.
- [DamiO91] Damiani M., Olivo P., Ricco B.: Analysis and Design of Linear Finite State Machines for Signature Analysis Testing. IEEE Transactions on Computers, vol. 40, No 9, September 1991, str. 1034-1045.
- [David78] David R.: Feedback shift register testing. Proc. Fault Tolerant Computing Systems Conf., June 1978, IEEE Computer Society Press, str. 103-107.
- [David80] David R.: Testing by Feedback Shift Register. IEEE Transactions on Computers, vol. C-29, No 7, July 1980, str. 668-673.
- [David85] David R.: Survey of Compact Testing by Means of Signature Analysis. Proc. of FTSD'85, Kokotek, September 1985, str. 75-85.
- [David86] David R.: Signature Analysis for Multiple-Output Circuits. IEEE Transactions on Computers, vol. C-35, No 9, September 1986, str. 830-837.

- [DebGD92] Debany W.H., Gorniak M.J., Daskiwich D.E.: Empirical Bounds on Fault Coverage Loss Due to LFSR Aliasing. Proc. of VLSI Test Symposium, 1992, str. 143-148.
- [EdirR93] Edirisooriya G., Robinson J.P.: Time and Space Correlated Errors in Signature Analysis. Proc of European Conference on Design Automation, EDAC'93, 1993, str. 275-281.
- [ElsPa59] Elspas B.: Theory of Autonomous Linear Sequential Networks. IRE Transaction on Circuit Theory, March 1959.
- [Esche92a] Eschermann B.: Synthesis and self-test of random logic control units. Integration, the VLSI journal, No 13, 1992, str. 209-230.
- [Esche92b] Eschermann B.: An Implicitly Testable Boundary Scan Tap Controller. Journal of Electronic Testing: Theory and Application, No 3, 1992, str. 159-169.
- [EschW91] Eschermann B., Wunderlich H.J.: Parallel Self-Test and Synthesis of Control Units. Proc. of 2nd ETC, 1991.
- [EschW92] Eschermann B., Wunderlich H.J.: Optimized Synthesis Techniques for Testable Sequential Circuits. IEEE Trans. on CAD. vol. 11, No 3, March, 1992, str. 301-312.
- [EtieI77] Etiemble D., Israel M.: Implementation of Ternary Circuits with Binary Integrated Circuits. IEEE Transactions on Computers, vol. C-26, No 12, December 1977, str. 1222-1232.
- [EtieI88] Etiemble D., Israel M.: Comparison of Binary and Multivalued ICs According to VLSI Criteria. Computer, April 1988, IEEE Computer Society Press, str. 28-42.
- [EXEL] EXEL EEPROMs Data Book, section 5, Programmable Logic Devices (5-1-5-32), Exel Microelectronics, A Division of Rohm Corp., San Jose, California, 1993.
- [Frank95] Franklin M.: Fast Computation of C-MISR Signatures. Proc of Fourth Asian Test Symposium, 1995, str. 293-297.
- [FrohW77] Frohwerk R.A.: Signature Analysis: A new Digital Field Service Method. Hewlett-Packard Journal, May 1977, str. 2-8.
- [Garbo95] Garbolino T.: Optimised Synthesis of Self-testable Finite State Machine Using BIST-PAT Structures in PLDs. Proc. of Inter. Conf. on Programmable Devices and Systems (PDS'95), Gliwice, listopad 1995, str. 51-58.
- [Garbo96] Garbolino T.: Wbudowane generatory testów deterministycznych dla samotestowalnych układów ASIC. ELEKTRONIKA, nr 6, 1996, str. 21-26.
- [Gill67] Gill A.: Linear Sequential Circuits. McGraw Hill, 1967.
- [GlosB89] Gloster C.S., Brglez F.: Boundary Scan with Built-In Self-Test. IEEE Design and Test of Computers, February 1989, str. 36-44.
- [Goeri87] Goering R.: Boundary-Scan Technique Targets Board-Level Testability. Computer Design, 1 October 1987, str. 47-49.

- [GoesS96] Goessel M., Sogomonyan E.S.: A Parity-Preserving Multi-Input Signature Analyzer and Its Application for Concurrent Checking and BIST. Journal of Electronic Testing: Theory and Applications, No 8, 1996, str. 165-177.
- [Golom82] Golomb S.W.: Shift Register Sequences. Revised Edition, Aegean Park Press, Laguna Hills, California, 1982.
- [Gości96] Gościński I.: Rejestry liniowe z mieszanym sprzężeniem zwrotnym. Pomiary, Automatyka, Kontrola, nr 1, 1996, str. 4-9.
- [GórkH76] Górkiewicz J., Hławiczka A.: Diagnostyka układów cyfrowych - automatyczne generowanie testów. Proc. of Intern. Symposium on Fault Diagnosis of Digital Networks and Fault-Tolerant Computing, Wisła, May 1976, str. 11-15.
- [GórkH77] Górkiewicz J., Hławiczka A.: Diagnostyka układów cyfrowych z wykorzystaniem automatycznego generowania testów. Informatyka, nr 1, 1977, str. 15-18.
- [Gucwa94] Gucwa K.: Wykorzystanie metody symulacji do oceny pokrycia uszkodzeń w układach cyfrowych. Zeszyty Naukowe Politechniki Śląskiej, Elektronika, z. 3, Gliwice 1994, str. 157-181.
- [GupP92] Gupta S.K., Pradhan D.K.: Can Concurrent Checkers Help BIST? Proc. of International Conference, 1992, str. 140-150.
- [GupP96] Gupta S.K., Pradhan D.K.: Utilization of On-Line (Concurrent) Checkers During Built-In Self-Test and Vice Versa, IEEE Trans. on Computers, vol. 45, Nr 1, January 1996, str. 63-73.
- [GuPR90] Gupta S.K., Pradhan D.K., Reddy S.M.: Zero Aliasing Compression. Proc of 20th Fault Tolerant Computing Systems Conference FTCS-20, June 1990, str. 254-263.
- [Gupta92] Gupta S.K.: Recent Advances in BIST. Proc. of VLSI Test Symposium, 1992, str. 235-240.
- [GuptP88] Gupta S.K., Pradhan D.K.: A New Framework for Designing and Analyzing BIST Techniques: Computation of Exact Aliasing Probability. Proc. 1988 International Test Conference, Washington, September 1988, str. 329-341.
- [Gutma79] Gutman F.: 8080 Routine Generates CRC character. EDN, June, No 5, 1979, str. 84.
- [Hassa82] Hassan S.Z.: Algebraic Analysis of Parallel Signature Analyzers. CRC Technical Report No 82-5, Center for Reliable Computing, Computer Systems Laboratory, Stanford University, June 1982.
- [HassL83] Hassan S.Z., Lu D.J., McCluskey E.J.: Parallel Signature Analyzers Detection Capability and Extensions. Proc. of COMPCON, Spring, 1983, str. 440-445.
- [HassM83] Hassan S.Z., McCluskey E.: Testing PLAs Using Multiple Parallel Signature Analyzers. str. 422-425.
- [HassM84a] Hassan S.Z., McCluskey E.: Increased Fault Coverage Through Multiple Signatures. FTCS'84, str. 354-359.

- [Helle95] Hellebrand S. et al.: Built-In Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial LFSR. IEEE Transaction on Computers, vol. 44, No 2, str. 223.
- [Hewle97] Katalog wyrobów firmy Hewlett Packard, 1997.
- [HłaGS95] Hławiczka A., Goessel M., Sogomonyan E.S.: A Hamming Code Preserving MISA and its Application for On-Line and Off-Line Testing. report No MPI-I-95-604, June 1995, Max-Planck -Society Fault-Tolerant Computing Group at the University of Potsdam,.
- [HłaGS96a] Hławiczka A., Goessel M., Sogomonyan E.S.: On-Line and Off-Line Testing Using a Linear Code-Preserving Signature Analyzer Checker. Proc. of IEEE European Test Workshop - ETW'96, June 12-14 1996, Sete, Francja, str. 14-20.
- [HłaGS96b] Hławiczka A., Goessel M., Sogomonyan E.S.: Memory Testing Using a Linear Code-Preserving Signature Analyzer Checker. Proc. of EDCC-2 Companion Workshop, May 15, 1996, Gliwice, str. 153-175.
- [HłaGS97] Hławiczka A., Goessel M., Sogomonyan E.S.: A Linear Code - Preserving Signature Analyzer COPMISR. Proc. of VLSI Test Symposium (VTS'97), April 27 - May 1, 1997, Monterey, California, USA, IEEE Computer Society Press, str. 350-355.
- [HłaKC96] Hławiczka A., Kopeć M., Chachulski M.: Plasterkowy rejestr komórkowy o sprzężeniu liniowym. Opis patentowy PL 168495, 29.02.1996 (WUP 02/96).
- [HławB87] Hławiczka A., Badura D.: Universal Test Controller Chip for Board Self Test. Proc. of 3rd International GI/ITG/GMA Conference on Fault-Tolerant Computing Systems, Bremerhaven, September 1987, Springer-Verlag, str. 165-175.
- [HławB94a] Hławiczka A., Binda J.: The Optimized Synthesis of Self-Testable Finite State Machines Using BIST-PST Structures in ALTERA Structures. Proc. of 4th International Workshop on Field Programmable Logic and Applications, September 7-9, 1994, Prague, Lecture Notes in Computer Science No 849, Springer Verlag Press, str. 120-122.
- [HławB94b] Hławiczka A., Binda J.: The Optimized Synthesis of the Self-Testable Sequential Circuits Based on the ALTERA Circuits. Proc. of XVII KKTOiUE, 19-21 październik 1994, Polanica Zdrój, Prace Naukowe Instytutu Telekomunikacji i Akustyki Politechniki Wrocławskiej nr 79, seria: konferencje nr 25, str. 561-566.
- [HławG77] Hławiczka A., Górkiewicz J.: Z międzynarodowego sympozjum w Wiśle: Diagnostyka urządzeń cyfrowych - problem wymagający rozwiązania. Informatyka, nr 1, 1977, str. 29-30.
- [HławH82] Hławiczka A., Huetter L., Pach A.: SAS-80 - Exerciser for P Based Systems Fault Signature Analysis. Proc. of 5th International FTSD Conference, Katowice, September, 1982, str. 83-93.

- [HławH83] Hławiczka A., Huetter L., Pach A.: Stymulator do analizy sygnaturowej uszkodzeń w systemach mikroprocesorowych. PAK nr 6, 1983, str. 196-199.
- [Hławi84a] Hławiczka A.: Binary Implemented Compression of Multiple-Valued Data Serial Streams. Proceedings of 14th ISMVL, Winnipeg, May 1984, IEEE Computer Society Press, str. 109-117.
- [Hławi84b] Hławiczka A.: Fault Signature Effectiveness of Microcomputer Address Bus. Electronics Letters, No 16, vol. 20, August 1984, str. 645-646.
- [Hławi84c] Hławiczka A.: Compression of Multi-Valued Data Serial Streams by Means of Parallel LFSR Signature Analyser. Proceedings of 2nd ICFTCS, Informatik-Fachberichte No 84, Springer-Verlag, Bonn, September 1984, str. 404-416.
- [Hławi84d] Hławiczka A.: Fault Signature Effectiveness of Microcomputer Address Bus. Proceedings of 7th International FTSD Conference, Sofia, October 1984, str. 216-223.
- [Hławi85a] Hławiczka A.: Własności detekcyjne różnych metod równoległej kompresji przy pomocy rejestru liniowego. Proceedings of 8th FTSD Conference, Kokotek, September 1985, str. 145-152.
- [Hławi85b] Hławiczka A.: Binary Implemented Exhaustive Testing of Multi-Valued Logic Networks. Proceedings of 8th International FTSD Conference, Kokotek, September 1985, str. 181-191.
- [Hławi86a] Hławiczka A.: Wielosygnaturowa równoległa analiza uszkodzeń w systemie cyfrowym. Zeszyty Naukowe Politechniki Śląskiej, z.83, Automatyka, nr kol. 888, str. 9-31.
- [Hławi86b] Hławiczka A.: Parallel Multisignature Analysis of Faults in the Multi-Output Digital System. Proc. of 16th FTCS Conference, Vienna July 1986, IEEE Computer Society Press, Washington, str. 398-403.
- [Hławi86c] Hławiczka A.: Compression of Three-State Data Serial Streams by Means of a Parallel LFSR Signature Analyzer. IEEE Transaction on Computers, vol. C-35, No 8, August 1986, str. 732-741.
- [Hławi86d] Hławiczka A.: The Survey on Signature Testing Methods. Proc. of 9th International FTSD Conference, Brno, September 1986, str. 21-30.
- [Hławi86e] Hławiczka A.: Signature Testing Methods for Microcomputers - A survey. Prace Naukowe ICT Pol. Wrocławskiej, nr 73, str. Konferencje, nr 30, Microcomputer 86 - Design, Practice, Education, Wrocław 1986, str. 90-99.
- [Hławi86f] Hławiczka A.: Binary Implemented Exhaustive Testing of Multi-Valued Logic Networks. Proceedings of 16th ISMVL, Blacksburg, May 1986, IEEE Computer Society Press, str. 85-93.
- [Hławi87a] Hławiczka A.: BIST Structure Using Multiinput Shift Register with Initial Value. Electronics Letters, No 9, vol. 23, 23rd April 1987, str. 476-478, a także Proc. of 10th International FTSD Conference, Sofia 1987, str. 336-342.

- [Hławi87b] Hławiczka A.: VLSI Wafers and Boards Diagnostics Using Multisignature Analysis. Proc. of 2nd European Workshop on Fault Diagnostics, Reliability and Related Knowledge-based Approaches, Manchester, April 1987, Pergamon Press.
- [Hławi87c] Hławiczka A.: Podstawy teoretyczne równoległej analizy sygnaturowej z wykorzystaniem dowolnego rejestru przesuwającego ze sprzężeniem liniowym. Prace IPI PAN, nr 601, marzec 1987, str. 279-313.
- [Hławi88a] Hławiczka A.: Podstawy teoretyczne równoległej analizy sygnaturowej z wykorzystaniem dowolnego rejestru przesuwającego ze sprzężeniem liniowym. Archiwum Automatyki i Telemekhaniki, tom XXXIII, z.2, 1988.
- [Hławi88b] Hławiczka A.: Weryfikatory sygnatur - druga generacja analizatorów sygnatur. PAK, 1988, nr 7, str. 156-159.
- [Hławi88c] Hławiczka A.: Built-In Evaluator Using Multiple Compression. 6th European Workshop on Design For Testability, Garderen, June 1988.
- [Hławi89a] Hławiczka A.: Hybrid Design of Parallel Signature Analyzers. Proc. of 1st European Test Conference, Paris, April 1989, IEEE Computer Society Press, Washington, str. 354-360.
- [Hławi89b] Hławiczka A.: Signature Analyzers Testing with Bottom-Top Exclusive OR type MISR. Proc. of 4th International GI/ITG/GMA Conference on FTCS, Baden-Baden, September 1989, Springer-Verlag, str. 356-367.
- [Hławi89c] Hławiczka A.: Use of Built-In Evaluators with Linear Compression. Proc. of 12th International Conference on FTSD, Praha, September 1989, str. 32-45.
- [Hławi89d] Hławiczka A.: Ułatwianie testowania przy użyciu uniwersalnego sprzęgu upraszczającego komunikację pomiędzy układem cyfrowym a jego testerem. Materiały V Krajowego Sympozjum Telekomunikacji KST 89, Bydgoszcz, wrzesień 1989, tom E, str. 81-90.
- [Hławi89e] Hławiczka A.: Built-In Evaluator Using Multiple Compression. Journal of New Generation of Computer Systems, vol. 2, No 4, 1989, Akademie-Verlag, Berlin, str. 295-305.
- [Hławi89f] Hławiczka A.: Metody testowania pakietów cyfrowych w produkcji seryjnej - wady i zalety. ELEKTRONIZACJA, z. 28, WKiŁ, Warszawa 1989, str. 41-51.
- [Hławi89g] Hławiczka A.: Eliminacja krążących pakietów oraz zwiększanie przepustowości testerów za pomocą analizatorów i weryfikatorów sygnatur. ELEKTRONIZACJA, z. 28, WKiŁ, Warszawa 1989, str. 80-91.
- [Hławi90a] Hławiczka A.: Built-In Self-Test Using Time Linear Compression. Journal of the New Generation of Computer Systems, 1990, vol. 3, No 4, Akademie-Verlag Berlin, str. 337-352.
- [Hławi90b] Hławiczka A.: Przyczyny małej popularności stosowania technologii upraszczania testowania. PAK, nr 11, 1990, str. 228-230.
- [Hławi90c] Hławiczka A.: Sprzęgi ułatwiające testowanie. ELEKTRONIKA, nr 7-9, 1990, str. 57-61.

- [Hławi90d] Hławiczka A.: Tester wewnętrzny w postaci komórkowego rejestru liniowego - jedna z jego właściwości. Materiały KKTOIUE, Bielsko-Biała, październik 1990.
- [Hławi90e] praca zbiorowa pod red. A. Hławiczki: Testowanie i projektowanie łatwo testowalnych układów i pakietów cyfrowych - część 1. Skrypt Politechniki Śląskiej, nr 1586, Gliwice 1990.
- [Hławi91a] Hławiczka A.: Wbudowane weryfikatory i ich rola w procesie samotestowania. ELEKTRONIKA, nr 1, 1991, str. 13-16.
- [Hławi91b] Hławiczka A.: Plasterkowe rejestry komórkowe o sprzężeniu liniowym w testerach wewnętrznych. Materiały XI KKA, Białystok-Białowieża, 17-20 września 1991, str. 355 - 362.
- [Hławi92a] Hławiczka A.: D or T Flip-Flop Based Linear Registers. Archives of Control Sciences (d.Archiwum Automatyki i Telemekhaniki), vol. 1 (XXXVII), 1992, No 3-4, str. 249-268.
- [Hławi92b] Hławiczka A.: Parallel Signature Analyzers Using Hybrid Design of their Linear Feedbacks. IEEE Trans. on Computers, vol. 41, No 12, December 1992, str. 1562-1571.
- [Hławi92c] Hławiczka A.: Biblioteka komórek standardowych do automatycznego projektowania liniowych testerów wewnętrznych. Problemy Współczesnej Elektroniki, Materiały jubileuszowej sesji naukowej zorganizowanej z okazji nadania tytułu doktora h.c. Politechniki Śląskiej i 80-lecia urodzin prof.dr inż. Tadeusza Zagajewskiego, Gliwice, wrzesień 1992, str. 67-78.
- [Hławi93a] Hławiczka A.: Biblioteka komórek standardowych do automatycznego projektowania liniowych testerów wewnętrznych. ELEKTRONIKA, nr 2, 1993, str. 11-14.
- [Hławi93b] praca zbiorowa pod red. A. Hławiczki: Łatwo testowalne układy i pakiety cyfrowe - projektowanie i testowanie. WNT, Warszawa 1993.
- [Hławi96a] Hławiczka A.: Projektowanie łatwo testowalnych układów i systemów mikroelektronicznych na świecie i w Polsce. ELEKTRONIKA, nr 4, 1996, str. 21-23.
- [Hławi96b] Hławiczka A.: Struktury testerów wewnętrznych dla samotestowalnych układów sekwencyjnych (część 1) - rola rejestrów liniowych. ELEKTRONIKA, nr 9, 1996, str. 14-18.
- [Hławi96c] Hławiczka A.: Struktury testerów wewnętrznych dla samotestowalnych układów sekwencyjnych (część 2) - Techniki ułatwiania testowania za pomocą rejestrów liniowych. ELEKTRONIKA, nr 10, 1996, str. 20-24.
- [Hławi96d] Hławiczka A.: A Hamming Code-Preserving Signature Analyzer Checker For Memory With 8-bits Data Words. Proc. of Int. Conf. on Programmable Devices and Systems, PDS'96, Ostrava, listopad 1996, str. 119-126.
- [Hławi96e] Hławiczka A.: (redaktor) Proc. of EDCC-2 Companion Workshop on Dependable Computing. Silesian Technical University, Gliwice, May 15, 1996.

- [Hławi97a] Hławiczka A.: Struktury testerów wewnętrznych dla samotestowalnych układów sekwencyjnych (część 3) - Projektowanie Rejestrów Liniowych Przy Użyciu Układów Firmy ACTEL. ELEKTRONIKA, nr 2, 1997, str. 24-29.
- [Hławi97b] Hławiczka A.: Struktury testerów wewnętrznych dla samotestowalnych układów sekwencyjnych (część 4). ELEKTRONIKA, nr 7-8, 1997, str. 33-38.
- [HławiJ81] Hławiczka A., Jeżewski J.: Specjalizowany mikrokomputer do określania jakości żeliw. Materiały z konf. pt. "Zastosowanie komputerów w przemyśle", 1981, wrzesień, Szczecin, tom 2, str. 52-60.
- [HławiJ85] Hławiczka A., Jura J., Sakoł J.: Testowanie systemów mikroprocesorowych za pomocą emulatora układowego sprzężonego z analizatorem sygnatur. PAK nr 2, 1985, str. 40-44.
- [HławiK81a] Hławiczka A., Kubica H.: The Method of Increasing the Probability of Detecting Errors for Signature Analysis. Proc. of 4th International FTSD Conference, Brno, September 1981, str. 273-277.
- [HławiK81b] Hławiczka A., Kubica H.: Sposób zwiększania prawdopodobieństwa wykrywania i lokalizacji uszkodzeń techniką analizy sygnatur w systemach cyfrowych zwłaszcza mikrokomputerowych. Patent nr 137712 z mocą od dnia 21.05.81.
- [HławiK92a] Hławiczka A., Kopeć M.: Concatenable Cellular Automata Register Design for Built-In Self-Test. Proc. of the European Conference on Design Automation, Brussels, 16-19 March 1992, IEEE Computer Society Press, str. 164-168.
- [HławiK92b] Hławiczka A., Kopeć M.: Ułatwianie testowania systemów cyfrowych przy użyciu standardu P1149. ELEKTRONIKA, nr 7, 1992, str. 8-14.
- [HławiK93a] Hławiczka A., Kopeć M.: Properties of the Rule 150/90 Cellular Automata - Based MISRs and LFSRs used in Built-In Self-Test. Archiwum Informatyki Teoretycznej i Stosowanej, tom 5, zeszyt 1, 1993, str. 181-203.
- [HławiK93b] Hławiczka A., Kopeć M., Chachulski M., Boszko M.: Projekty nowych układów scalonych do realizacji sprzęgu P1149.3. ELEKTRONIKA, nr 5, 1993, str. 10-15.
- [HławiM86] Hławiczka A., Mostowski K.: Signature Verifier vs Signature Analyzer-Probability of Errors Detecting. Proc. of MICROELECTRONIK Conference, Ploudiv, October, 1986.
- [HławiM89] Hławiczka A., Mostowski K.: Wpływ liniowych sprzężeń zwrotnych na skuteczność analizy sygnaturowej uszkodzeń. Prace IPI PAN, marzec 1989, nr 655.
- [HławiM93] Hławiczka A., Mitas A., Ostas P.: Analizatory sygnatur - Poradnik dla użytkownika. Skrypt Politechniki Śląskiej, nr 1706, Gliwice 1993.

- [HławiN83] Hławiczka A., Nowiński M., Jeżewski J.: MIT 80 - Jednokartowy mikrokomputer do oceny jakości żeliw. Zeszyty Naukowe Politechniki Śląskiej, s. Automatyka, z. 66, nr 752, 1983, str. 151-165, a także Materiały z Konferencji pn. Metoda ATD i jej zastosowanie w praktyce, Gliwice 1983.
- [HławiN84] Hławiczka A., Nowiński M.: PAS 80 - Programowalny analizator sygnatur współpracujący z mikrokomputerem. PAK nr 7, 1985, str. 169-171, a także Materiały konferencji pn. Mikrokomputery w automatyce i technice systemów, tom 1, Wrocław, wrzesień 1984, str. 169-176.
- [HławiN85a] Hławiczka A., Nowiński M.: Easy-To-Use Programmable Signature Analyzer PAS 80. Proceedings of MICROSYSTEM Conference, Tabor, October 1985, a także Proceedings of 6th RELECTRONIC Symposium, Budapest, August 1985, str. 98-102.
- [HławiN85b] Hławiczka A., Nowiński M.: PAS-80 - Programowalny analizator sygnatur współpracujący z komputerem. PAK nr 7, 1985, str. 169-171.
- [HławiN89] Hławiczka A., Nikiel J.: Samotestowalne mikrokomputerowe urządzenie pomiarowe CRYSTALDIGRAF NC-ST. ELEKTRONIZACJA, z. 28, WKiŁ, Warszawa 1989, str. 66-80.
- [HławiP83] Hławiczka A., Pach A.: Method for Compression of Serial of Test Result Three State Data. Proceedings of 6th International FTSD Conference, Brno, Septemeber 1983, str. 162-168.
- [HortM89] Hortensius P.D., McLeod R.D., Pries W., Miller D.M., Card H.C.: Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test. IEEE Transactions on CAD, vol. 8, No 8, August 1989.
- [HortM90] Hortensius P.D., McLeod R.D., Card H.C.: Cellular Automata-Based Signature Analysis for Built-In Self-Test. IEEE Transactions on Computers, vol. 39, No 10, October 1990, str. 1273-1283.
- [HunTZ96] Hunter A., et al: Safety Critical Design and BIST for Front Line Reactor Protection System. Proc. of Test'96 Exhibition and Conference, 1996.
- [IEEE90] IEEE Standard Test Access Port and Boundary-Scan Architecture. IEEE Std 1149.1-1990, IEEE Press, May, 1990.
- [IvanA87] Ivanov A., Agarwal V.K.: On a fast method to monitor the behaviour of signature analysis register. Proc. 1987 Int. Test Conf., September 1987, IEEE Computer Society Press, str. 645-655.
- [IvanA88] Ivanov A., Agarwal V.K.: An Iterative Technique for Calculating the Aliasing Probability of Linear Feedback Signature Registers. Proc. of Fault Tolerant Computing Systems Conf., June 1988, IEEE Computer Society Press, str. 70-75.
- [IvanA89] Ivanov A., Agarwal V.K.: An Analysis of the Probabilistic Behavior of Linear Feedback Signature Registers. IEEE Transactions on Computer-Aided Design, vol. 8, No 10, October 1989, str. 1074-1088.

- [IvanP90] Ivanov A., Pilarski S.: Signature Analysis for VLSI Built-In Self-Test: A Survey. Raport Centre for Integrated Computer Systems Research, CICS R TR90-3, The University of British Columbia, March 1990.
- [IvanS91] Ivanov A., Starke C.W., Agarval V.K., i in: Iterative Algorithms for Computing Aliasing Probabilities. IEEE Transactions on Computer-Aided Design, vol. 10, No 2, February 1991, str. 260-265.
- [Jou95] Jou J.: An Effective BIST Design for PLA. Proc of Fourth Asian Test Symposium, 1995, str. 298-302.
- [Kalis77] Kalisz J.: Cyfrowe układy scalone w technice systemowej. WMON, Warszawa 1977.
- [KarGP91] Karpovsky M.G., Gupta S.K., Pradhan D.K.: Aliasing and Diagnosis Probability in MISR and STUMPS Using a General Error Model. International Test Conference 1991, str. 828-839.
- [KatN91] Katozi M., Norsieck A.: Built-In Testable Error Detection and Correction. IEEE Journal of Solid-State Circuits, vol. 27, No 1, January 1991, str. 59-66.
- [Kautz65] Kautz W.H. (edytor): Linear Sequential Switching Circuits. Holden-Day Inc., 1965.
- [KhalK95] Khaled S., et al: Frequency-Based BIST for Analog Circuit Testing. Proc. of 13th VLSI Test Symposium, 1995, str. 54-59.
- [Kim88] Kim K., Ha D. S., Tront J.G.: On Using Signature Registers as Pseudorandom Pattern Generators in Built-In Self-Testing. IEEE Trans. on CAD, vol. 7, No 8, August 1988, str. 919-928.
- [KonBJ96] Koenemann B., Bennets B., Jarwala N., Nadeau-Dostie B.: Built-In Self-Test Assuring System Integrity. Computer, November 1996, vol. 29, No 11, str. 39-45.
- [KoneM79] Koenemann B., Mucha J., Zwiechoff G.: Built-In Logic Block Observation Techniques. Proc. 1979 IEEE International Test Conference, str. 37-41, 1979.
- [KoneM80] Koenemann B., Mucha J., Zwiechoff G.: Built-In Test for Complex Digital Integrated Circuits. IEEE Journal of Solid-State Circuits, vol. SC-15, No 3, June 1980, str. 315-319.
- [Kopeć94] Kopeć M.: Analiza i zwiększanie skuteczności pierścieni testujących. Praca doktorska. Wydział AEI Politechniki Śląskiej, Instytut Elektroniki, Gliwice 1994.
- [Kopeć96] Kopeć M.: Zwiększanie skuteczności pierścieni testujących. ELEKTRONIKA, nr 4, 1996, str. 24-28.
- [Kraśn89] Kraśniewski A.: Projektowanie samotestowalnych układów cyfrowych wielkiej skali integracji. Prace Naukowe, Elektronika z. 83, Wyd. Politechniki Warszawskiej, 1989.

- [Kraśn96] Kraśniewski A.: Design of Dependable Hardware: What BIST Is Most Efficient? Proc. of the Second European Dependable Computing Conference, Lecture Notes in Computer Science No 1150, Springer-Verlag, str. 233-245.
- [KraśP87a] Kraśniewski A., Pilarski S.: Effectiveness of Using a Test Response Compactor for Test Pattern Generation. Proc. 10th International Conference FTSD, Varna 1987, str. 284-289.
- [KraśP87b] Kraśniewski A., Pilarski S.: Circular Self-Test Path: A Low-Cost BIST Technique. Proc. of 24th Design Automation Conf., June, 1987, 407-415.
- [KraśP88] Kraśniewski A., Pilarski S.: Experiments with Test Pattern Generation Using a Test Response Compactor. Proc. 11th International Conference FTSD, Suhl 1988, str. 197-202.
- [KraśP89] Kraśniewski A., Pilarski S.: Circular Self-Test Path: A Low-Cost BIST Technique for VLSI Circuits. IEEE Transaction on CAD, vol. 8, No 1, January 1989, str. 46-55.
- [KraśP92] Kraśniewski A., Pilarski S.: Testing Random Access Memories Using Circular Self-Test Path. Technical Report CSS/LCCR TR 92-02, 1992, Simon Fraser University, Canada.
- [Krawc95] Krawczyk H.: Dependable Processing. Technical University of Gdańsk, TEMPUS JEP 3470, Gdańsk 1995.
- [Kubiś84] Kubiś M.A.: Analiza sygnatur. Elektronizacja, zeszyt nr 20, 1984.
- [KunHL95] Kung C., Huang C., Lin C.: Fast Fault Simulation for BIST Applications. Proc of Fourth Asian Test Symposium, 1995, str. 93-99.
- [Lambi91] Lambidonis D., Ivanov A., Agarval V.K.: Fast Signature Computation for Linear Compactors. Proc. of Intern. Test Conference, IEEE CS Press, Los Alamitos, Calif., 1991, str. 808-817.
- [Lapri91] Laprie J-C. et al.: Dependability: Basic Concepts and Terminology - Dependable Computing and Fault Tolerant Systems. Volume 5, Springer-Verlag, Wien, New York, 1991.
- [LeBla84] LeBlanc J.J.: LOCST: A Built-In Self-Test Technique. The IEEE Design and Test, vol. 1, No 4, November 1984, str. 45-52.
- [Lee88] Lee Y.H. et al.: Optimal Scheduling of Signature Analysis for VLSI Testing. Proc. of Intern. Test Conference, 1988, str. 443-447.
- [LeeH91] Lee H.K., Ha D.S.: An Efficient, Forward Fault Simulation Algorithm Based on The Parallel Pattern Single Fault Propagation. Proc. of International Test Conference, 1991, str. 946-955.
- [LemGB94] Gupta S.K., Breuer M.A.: Test Embedding with Discrete Logarithms. Internal Report, Computer Science Department, University of South California, Los Angeles, April 19, 1994.
- [LempG94] Lempel M., Gupta S.K., Breuer M.A.: Test Embedding with Discrete Logarithms. VLSI Test Symposium, 1994, str. 74-80.

- [LempG95] Lempel M., Gupta S.K.: Zero-Aliasing for Modeled Faults. IEEE Transaction on Computers, vol. 44, No 11, November, 1995, str. 1283-1295.
- [Maund87] Maunder C.: Testing Surface-Mount Boards: a Growing Need for Testability Standards. Electronic Design Automation, January 1987, str. 8-17.
- [Maund95] Maunder Colin: Bringing down more barriers with design for test. IEEE Spectrum, January 1995, str. 54-55.
- [MaunB87] Maunder C., Beenker F.: A Framework for Structured Design for-Test. Proc. 1987 IEEE International Test Conference, 1987, IEEE Computer Society Press, str. 714-723.
- [MaunT89] Maunder C.M., Tullos R.E.: The Test Access Port and Boundary Scan Architecture. IEEE Computer Society Press, Los Alamitos, California, 1989.
- [Maxwe88] Maxwell P.C.: Comparative Analysis of Different Implementations of Multiple-Input Signature Analyzers. IEEE Transactions on Computers vol. 37, No 11, November 1988, str. 1411-1414.
- [McClB81] McCluskey E.J., Bozorgui-Nesbat S.: Design for Autonomous Test. IEEEETC, vol. c-30, No 11, str. 866-875.
- [McClu82] McCluskey E.J.: A Discussion of Multiple-Valued Logic Circuits. Proc. of the Twelfth Intern. Symposium on Multiple-Valued Logic, May 25-27, Paris 1982, IEEE Computer Society Press, str. 200-205.
- [McClu85a] McCluskey E.J.: Built-In Self-Test Techniques. Journal IEEE Design and Test, April 1985, str. 21-28.
- [McClu85b] McCluskey E.J.: Built-In Self-Test Structures. Journal IEEE Design and Test, April 1985, str. 29-36.
- [MitaH88] Mitas A., Hławiczka A., Polok D.: Automatyczny tester funkcjonalny układów scalonych - MASTER 86, własności, technika diagnostyczna, oprogramowanie. PAK, nr 9, 1988, str. 209-211.
- [MuchD86] Mucha J.P., Daehn W., Gross J.: Self-Test in a Standard Cell Environment. IEEE Design and Test of Computers, December 1986, str. 35-41.
- [MurrH96] Murray B.T., Hayes J.P.: Testing ICs: Getting to the Core of the Problem. Computer, November 1996, vol. 29, No 11, str. 32-38.
- [MuziW86] Muzio J.C., Wesselkamper T.C.: Multiple-Valued Switching Theory. Adam Hilder Ltd Press, 1986.
- [NagvK92] Nagvajara P., Karpovsky M.G.: Coset Error Detection in BIST Design. Proc. Od VLSI Test Symposium, 1992, str. 79-83.
- [Nebus83] Nebus J.F.: Parallel Data Compression for Fault Tolerance. Computer Design, April 5, 1983, str. 127-134.
- [OlivD93] Olivo P., Damiani M., Ricco B.: Aliasing Minimization in Signature Analysis Testing. Proc. of European Test Conference, 1993, str. 451-456.
- [P1149.4] P1149.4 Standard for a Mixed-Signal Test Bus, IEEE, DO2., March 1995.

- [Parke86] Parker K.P.: Testability: Barriers to Acceptance. IEEE Design and Test of Computers, October 1986, str. 11-15.
- [PetW94] Peterson W.W., Weldon E.J.: Error-Correcting Codes. The MIT Press, 1994.
- [PieńT80] Pieńkos J., Turczyński J.: Układy scalone TTL w systemach cyfrowych. WKŁ, Warszawa 1980.
- [Piest95] Piestrak J.P.: Design of self-testing checkers for unidirectional error detecting codes. Prace Naukowe ICT Pol. Wrocław. nr 92, seria monografie nr 24, 1995.
- [PilaK90] Pilarski S., Kraśniewski A., Kameda T.: Estimating Testing Effectiveness of the Circular Self-Test Path Technique. Raport CSS/LCCR (Simon Fraser Univesristy, Burnaby ,B.C. Canada) TR 90-10, 1990.
- [PilaK95] Pilarski S., Kameda T.: A Probabilistic Analysis of Test-Response Compaction. IEEE Computer Society Press, 1995.
- [PomeR95] Pomeranz I., Reddy S.M.: Aliasing Computation UUsing Fault Simulation with Fault Dropping. IEEE Trans. on Computers, vol. 44, No 1, January 1995, str. 139-144.
- [PomRT92] Pomeranz I., Reddy S.M., Tangirala R.: On Achieving Zero Aliasing for Modeled Faults. Proc of European Conference on Design Automation, EDAC'92, 1992, str. 291-299.
- [Powel96] Powell T.J.: Consistently Dominant Fault Model for Tristate Buffer Nets. Proc. of 14th VLSI Test Symposium, 1996, str. 400-404.
- [PradG91] Pradhan D.K., Gupta S.K.: A New Framework for Designing and Analyzing BIST Techniques and Zero Aliasing Compresion. IEEE Transactions on Computers, vol. 40, No 6, June 1991, str. 743-763.
- [PradH78] Pradhan D.K., Hsiao M.Y., Patel A.M., Su S.Y.: Shift Registers Designed for On-Line Fault Detection. Proc. of Eighth International Conference on Fault-Tolerant Computing Systems FTCS- 8, Toulouse, France, 1978, str. 173-178.
- [Pradh95] Pradhan D.K.: Fault-Tolerant Computer System Design. Prentice Hall PTR, 1995.
- [RajsT90] Rajski J., Tyszer J., Salimi B.: On the Diagnostic Resolution of Signature Analysis. str. 364-367.
- [RajsT91a] Rajski J., Tyszer J.: Experimental Analysis of Fault Coverage in Systems with Signature Registers. ETC'91, str. 45-51.
- [RajsT91b] Rajski J., Tyszer J.: On the Diagnostic Properties of Linear Feedback Shift Registers. IEEE Transactions on Computer-Aided Design, vol. 10, No 10, October 1991, str. 1316-1322.
- [RajsT91] Rajski J., Tyszer J.: Experimental of Fault Coverage in Systems with Signature Registers. Proc. of European Test Conference, 1991, str. 45-51.
- [Ralla78] Rallapalli K.: CRC Error-Detection Schemes Ensure Data Accuracy. EDN, Sepetember, No 8, 1978, str. 119-123.

- [RaoF89] Rao T.R.N., Fujiwara E.: Error Control Coding for Computer Systems. Prentice Hall, New York, 1989.
- [Robin85] Robinson J.P.: Modern Digital Troubleshooting - an Implementation Guide. Data I/O Corporation 1985.
- [Robin91] Robinson J.P.: Aliasing Probability for Feedback Signature Compression of Test Data. IEEE Trans. on Computers, vol. 40, No 7, July 1991, str. 867-873.
- [RudeS87] Rudell R.L., Sangiovanni-Vincentelli A.: Multiple-Valued Minimization for PLA Optimization. IEEE Transactions on Computer-Aided Design, vol. CAD-6, No 5, September 1987, str. 727- 750.
- [SaluC92] Saluja K.K., Chin-Foo See: An Efficient Signature Computation Method. IEEE Design and Test of Computers, December 1992, str. 22-26.
- [Sapie87] Sapiecha K.: Testowanie i diagnostyka systemow cyfrowych. PWN, Warszawa 1987.
- [Sasao88] Sasao T.: Multiple-Valued Logic and Optimization of Programmable Logic Arrays. Computer, April 1988, IEEE Computer Society Press, str. 71-80.
- [SavaW95] Savage W.: Efficient BIST Techniques for Field Programmable Gate Arrays. Catalog ACTEL, May 1995, str. 11-17 do 11-25.
- [SaviB93] Savir J., Bardell P.H.: Built-In Self-Test: Milestones and Challenges. VLSI Design, 1993, vol. 1, No 1, str. 23-44.
- [Savir80] Savir J.: Syndrome-Testable Design for Combinational Circuits. IEEE Transactions on Computers, vol. C-29, June 1980, str. 442-451.
- [SavTI96] Savaria Y., Thibeault C., Ivanov A.: IEEE VLSI Test Symposium: Meeting the Quality Challenge. IEEE Design and Test of Computers, Fall 1996, str. 110-112.
- [SeeS92] See C.F., Saluja K.K.: An Efficient Method for Computation of Signatures. Proc. of Fifth Intern. Conference on VLSI Design, IEEE CS Press, 1992, str. 245-250.
- [Seidl83] Seidler J.: Nauka o informacji. tom 2 pt. Sygnały niosące informację i jej odtwarzanie. WNT, Warszawa 1983.
- [SeirV91] Seireg R.H., Vacroux A.G.: Comparison Between the Dynamic Behaviour of Maximum Length and Cyclic Shift Registers. Proc. of VLSI Test Symposium, 1991, str. 260-264.
- [SerrM85] Serra M., Muzio J.C.: Multiple-Valued Linear Feedback Shift Register. University of Victoria, Report Number: 08-1985.
- [SerrM88] Serra M., Miller M., Muzio J.C.: Linear Cellular Automata and LFSRs are Isomorphic. Proc. of Third Tech. Workshop-New Directions for IC Testing, October 1988, str. 213-223.
- [SerrS90] Serra M., Slater T., Muzio J.C., Miller M.: The analysis of One-Dimensional Linear Cellular Automata and Their Aliasing Properties. IEEE Transactions on Computer-Aided Design, vol. 9, No 7, July 1990, str. 767-778.

- [SiewS92] Siewiorek D.P., Swarz R.S.: Reliable Computer Systems-Design and Evaluation. Digital Press 1992.
- [Smith80] SMITH J.E.: Measures of the effectiveness of fault signature analysis. IEEE Trans. Comput. vol. C-29, No 6, June 1980, str. 510-514.
- [Socha66] Sochacki J.: Transmisja danych. Problemy Telekomunikacji, nr 10, WKiL, Warszawa 1966.
- [SogG95] Sogomonyan E.S., Goessel M.: A New Parity-Preserving Multi-Input Signature Analyser. Proc. of 1st IEEE Int. On-Line Testing Workshop, Nice, France, 1995, str. 211-215.
- [SogG96] Sogomonyan E.S., Goessel M.: Concurrently Self-Testing Embedded Checkers for Ultra-Reliable Fault-Tolerant Systems. Proc. of VLSI Test Symposium, 1996, str. 138-144.
- [SridH82] Sridhar T., Ho D.S., Powell T.J., Thatte S.M.: Analysis and Simulation of Parallel Signature Analyzers. Proc. 1982 IEEE International Test Conference, str. 656-661.
- [Tan87] Tan S.B. et al.: A Fast Signature Simulation Tool for BIST. Proc. of 24th Design Automation Conf, IEEE CS Press, 1987, str. 17-25.
- [Toub93] Touba N.A.: Logic Synthesis for Concurrent Error Detection. Technical Report of The Center for Reliable Computing, Computer Systems Laboratory, Department of Electrical Engineering and Computer Science, Stanford University, No CSL TN # 93-x, February 1993.
- [Touba94] Touba N.A., McCluskey E.J.: Logic Synthesis Techniques for Reduced Area Implementation of Multilevel Circuits with Concurrent Error Detection. Proc. of Intern. Conference on CAD (ICCAD), 1994, str. 651-654.
- [Turin84] Turino J.: A Totally Universal Reset, Initialization and Nodal Observation Circuit. Proc. IEEE International Test Conference, 1984, str. 878-883.
- [Turin85] Turino J.: Enhancing Built-In Test on SMT Boards. Evaluation Engineering, June 1985.
- [VoelP88] Voelkel L., Pliquet: Signaturanalyse. Akademie-Verlag Berlin, 1988.
- [Waicu87] Waicukauski J.A. et al.: Diagnosis of BIST Failures by PPSFP Simulation. Proc. of Intern. Test Conference, 1987, str. 480-484.
- [Wang87] Wang L.-T.: Circuits for Built-In Self-Test. Center for Reliable Computing Stanford University, CRC Technical Report, No 87-14, July 1987.
- [WangM86b] Wang L.-T., McCluskey E.J.: A Hybrid Design of Maximum Length Sequence Generators. Proc. IEEE International Test Conference 1986, September 86, str. 38-45.
- [WangM86c] Wang L.-T., McCluskey M.: Feedback Shift Registers for Self-Testing Circuits. VLSI Systems Design, December 1986, str. 50-58.
- [WangM86d] Wang L.-T., McCluskey M.: Concurrent Built-In Logic Block Observer (CBILBO). Digest of Papers, IEEE 1986 Int'l Symposium on Circuits and Systems (ISCAS-86), vol. 3 of 3, San Jose, CA, May 5-7, str. 1054-1057.

- [WangM87b] Wang L.-T., McCluskey E.J.: Built-In Self-Test for Sequential Machines. Proc. 1987 International Test Conference, Washington, September 1987, str. 334-341.
- [WangM88] Wang L.T., McCluskey E.J.: Hybrid designs generating maximum-length sequences. IEEE Trans. Comput. Aided Design, January 1988, vol. 7, No 1, str. 91-99.
- [WangM95] Wang L-C, Mercier M.R., Williams T.W.: On Efficiently and Reliably Achieving Low Defective Part Levels. Proc. of ITC 1995, str. 616-625.
- [WillD86] Williams T.W., Daehn W., Gruetzner M., Starke C.W.: Comparison of Aliasing Errors for Primitive and Non-Primitive Polynomials. Proc. 1986 International Test Conference, September 1986, str. 282-288.
- [WillD87a] Williams T.W., Daehn W., Gruetzner M., Starke C.W.: Aliasing Errors in Signature Analysis Registers. IEEE Design and Test of Computers, April 1987, str. 39-45.
- [WillD87b] Williams T.W., Daehn W., Gruetzner M., Starke C.W.: Bounds on Aliasing Errors in Linear Feedback Shift Registers. CompEuro, May 1987, str. 373-377.
- [WillD89] Willtams T.W., Daehn W.: Aliasing errors in multiple input signature analysis registers. Proc. 1989 European Test Conf., Paris, April 1989, IEEE Computer Society Press, str. 338-344.
- [WillP82] Williams T.W., Parker K.P.: Design for Testability - a Survey. IEEE Transactions on Computers, January, 1982.
- [WuI92] Wu Y., Ivanov A.: A Fuzzy Multiple Signature Compaction Scheme for BIST. Proc. of 1-st Asian Test Symposium, 1992, str. 247-252.
- [WuI93a] Wu Y., Ivanov A.: Achieving Minimal Hardware Multiple Signature Analysis for BIST. Proc. of 2-nd Asian Test Symposium, 1993, str. 311-316.
- [WuI93b] Wu Y., Ivanov A.: Minimal Hardware Multiple Signature Analysis for BIST. Proc. of VLSI Test Symposium, 1993, str. 17-20.
- [WuI95] Wu Y., Ivanov A.: Reducing Hardware with Fuzzy Multiple Signature Analysis. IEEE Design and Test of Computers, spring 1995, str. 68-74.
- [XILINX94] XILINX - The Programmable Logic Data Book, 1994.
- [Youn91] Youn H.Y.: Compact Testing with Intermediate Signature Analysis. Proc. of VLSI test Symposium, 1991, str. 47-52.
- [Yu96] Yu A.: The Future of Microprocessors. IEEE MICRO, December 1997, vol. 16, No 6, str. 46-53.
- [ZoriA86] Zorian Y., Agarwal Y.K.: A General Scheme to Optimize Error Masking in Built-In Self-Testing. Proc. of FTCS, Vienna, June 1986, str. 410-415.

REJESTRY LINIOWE - ANALIZA, SYNTEZA I ZASTOSOWANIA W TESTOWANIU UKŁADÓW CYFROWYCH

Streszczenie

Opisano użyteczność stosowania autonomicznych oraz jedno- i wielowjęściowych rejestrów liniowych w testerach wewnątrzukładowych, w upraszczaniu zewnętrznego testowania układów cyfrowych oraz w układach samokontrolujących się. Uzasadniono ich przydatność w kompaktacji odpowiedzi testowanych układów cyfrowych, w generacji testów pseudolosowych oraz w sterowaniu procesem testowania.

Głównym tematem rozważań jest właściwy dobór sprzężenia liniowego oraz jego struktury mający na celu zwiększenie jakości testowania, zmniejszenie kosztów samotestowania oraz efektywne wykorzystywanie zasobów układów FPGA, w których rejestry liniowe są implementowane. W związku z tym wprowadzono nowe struktury sprzężeń rejestrów liniowych zawierających także przerzutniki T i znacznie rozszerzających licznosc zbioru rejestrów związanych z tym samym wielomianem charakterystycznym.

Na bazie pierścienia wielomianów nad ciałem $GF(2)$ przedstawiono jednolity opis algebraiczny pracy tych rejestrów. Stał się on podstawą zaproponowania ich uniwersalnego schematu zastępczego. W pracy podano praktyczne przykłady stosowania tego schematu zastępczego, zamiast skomplikowanego grafu przejść rejestrów liniowych, przy analizie rozwiązań testowania wewnątrzukładowego. Umożliwił on także zbadanie w pracy przyczyn maskowania błędów w kompaktorach liniowych oraz określenie cech sprzężeń liniowych kompaktorów gwarantujących minimalne maskowanie błędów. Wykorzystano go również do zaproponowania technik znaczącego zmniejszania maskowania błędów lub nawet jego całkowitej likwidacji.

W pracy przedstawiono także techniki projektowania różnych struktur sprzężeń liniowych w oparciu o założony wielomian charakterystyczny oraz w przypadku rejestrów komórkowych w oparciu o kilkukomórkowe moduły plasterkowe.

Ponadto w pracy podano struktury, opis algebraiczny i techniki projektowania rejestrów liniowych przechowujących wyłącznie słowa kodowe należące do rozdzielnego kodu liniowego przydatnego w wysoce wiarygodnych systemach komputerowych.

Reasumując, w pracy usystematyzowano i ujednolicono teorię i praktykę projektowania rejestrów liniowych akcentując dorobek autora w dziedzinie analizy i syntezy łatwo testowalnych układów cyfrowych. Zaproponowano nowe oryginalne rozwiązania mające duże znaczenie dla rozwoju wiarygodnych systemów przetwarzania informacji.

LINEAR REGISTERS - ANALYSIS, SYNTHESIS AND APPLICATIONS IN DIGITAL CIRCUITS TESTING

Abstract

In this book has been described the usefulness of applying autonomous as well as single and multi input linear registers MISR in built-in circuit testers, in simplifying external digital circuit testing and in checkers. Their benefit in tested digital circuits responses compaction, in the generator of pseudo-random tests and in testing control has been accounted for.

The main topic of the book is the proper selection of linear feedback and its structure aiming at increasing testing quality, decreasing self-testing costs and the effective use of resources type FPGA circuits in which linear registers are implemented. In connection with this have been introduced new structures of linear register feedback containing also flip-flops T and a significantly widening the power of the set of registers connected with the same characteristic polynomial.

The uniform algebraic description of the operation of these registers has been introduced on the basis of ring of polynomials over a field $GF(2)$. It became the basis for proposing their universal substitute scheme. In the book practical examples of that substitute scheme - instead of complicated state transition graphs of linear registers - in built-in circuit testing analysis have been given. This also made it possible to examine the reasons for error masking in linear compactors and to determine the characteristics of linear compactor feedbacks which ensure minimal error aliasing. It was also used to suggest a techniques which significantly decrease an error masking or even its total elimination.

The book also introduces designing techniques of various linear feedback structures using the assumed characteristic polynomial as well as in the case of cellular automata registers based on slice moduls which contain several cells.

Furthermore, the book presents structures, algebraic description and designing technique of linear registers, preserving only a separable linear code. These linear registers are useful in highly dependable computer systems.

Summing up, the book systemises and unifies both the theory as well as designing of linear registers, emphasizing the author's scientific output in the field of analysis and synthesis of easily tested digital circuits. New, original solutions have been proposed, instrumental for the development of dependable data processing systems.

BIBLIOTEKA GŁÓWNA
Politechniki Śląskiej

P.4474/97/9

Druk: Drukarnia Gliwice, ul. Zwycięstwa 27, tel. 252 49 50