Na prawach rękopisu

INSTYTUT CYBERNETYKI TECHNICZNEJ
POLITECHNIKI WROCŁAWSKIEJ
Raport Serii PREPRINTY Nr 13/82

METODY PROJEKTOWANIA SAMOTESTOWALNYCH UKŁADÓW KONTROLNYCH KODÓW STAŁOWAGOWYCH

Praca doktorska

Stanisław PIESTRAK

Promotor:
doc.dr hab. W. Zamojski

Słowa kluczowe: diagnostyka, kod nieuporządkowany, kod stałowagowy, niezawodność, system cyfrowy, testowanie układów cyfrowych, układ kombinacyjny, układ kontrolny, układ samosprawdzalny, układ samotestowalny, wykrywanie uszkodzeń

Wydawnictwo Politechniki Wrocławskiej
Wrocław 1982

SPIS TREŚCI

WYKAZ WAŻNIEJSZYCH OZNACZEŃ

Br - liczba bramek logicznych

C_X, C_Z - kod wejść, kod wyjść układu kontrolnego

$C_{m/n}$ - zbiór słów kodu m/n

f - symbol uszkodzenia

f^* - symbol stanu niezawodnościowego układu $\mathbb{H}$

f^z, f^z_u - uszkodzenie pojedyncze (jednokierunkowe) typu s/z, $z = \{0,1\}$

F - zbiór uszkodzeń (najbardziej prawdopodobnych) układu $\mathbb{H}$

F_p, F^i_p - zbiór wszystkich uszkodzeń pojedynczych typu s/z układu $\mathbb{H}$, $\mathbb{H}^i$

F^* - zbiór stanów niezawodnościowych układu $\mathbb{H}$; $F^* = \{\varphi\} \cup F$

F_u - zbiór uszkodzeń jednokierunkowych

$\mathbb{H}^i$ - symbol układu kontrolnego kodu m/n

$\mathbb{H}^i$ - symbol i-tego bloku układu $\mathbb{H}$

$H^i = \{H^i_1, H^i_2, \ldots, H^i_{n_i}\}$ - wektorowa funkcja boole'owska realizowana przez układ $\mathbb{H}^i$

$H = \{H_1, H_2, \ldots, H_q\}$ - wektorowa funkcja boole'owska realizowana przez układ $\mathbb{H}$

L - liczba poziomów bramek

m - waga kodu m/n

m/n - skrócone oznaczenie kodu "m" z n"

m_i/n_i - kod, którego słowa występują na wyjściu układu $\mathbb{H}^i$

m_j - implikant

$m_j(x_i)$ - dzielnik implikantu m_j względem zmiennej wejściowej x_i (def. 2.4.2)

$m_j \leftrightarrow X_j$ - relacja "implikant m_j odpowiada słowu X_j" (def. 2.3.3)

M_i - zbiór implikantów funkcji H_i

M - zbiór wszystkich implikantów funkcji H

n - długość słowa kodu m/n

s/z - symbol uszkodzenia typu sklejenie z "z", $z = \{0,1\}$

STC - skrót: samotestowalny układ kontrolny

$T(a \geqslant i)$ - symbol funkcji progowej o progu i, określonej na zmiennych ze zbioru A

T^n_i - symbol funkcji progowej o progu i, określonej na zbiorze n dowolnych zmiennych

$\mathbb{T}$ - zbiór testów

TSC - skrót: całkowicie samosprawdzalny

We - łączna liczba wejść bramek układu

$\{x_1, x_2, \ldots, x_n\}$ - zbiór zmiennych wejściowych układu $|H$

X - słowo wejściowe układu $|H$, $X = \{x_1, x_2, \ldots, x_n\}$

$\mathbb{X}$ - zbiór wejściowych słów kodowych układu $|H$

$\overline{\mathbb{X}}$ - zbiór wejściowych słów niekodowych układu $|H$

$\mathbb{X}_0$ - przestrzeń wejść układu $|H$

$\overline{\mathbb{X}}_G$ - zbiór wyjściowych słów niekodowych generowanych przez układ $\mathbb{G}$

Z - słowo wyjściowe układu $|H$

$\mathbb{Z}$ - zbiór wyjściowych słów kodowych układu $|H$, $|H^i$

$\overline{\mathbb{Z}}$ - zbiór wyjściowych słów niekodowych układu $|H$, $|H^i$; $\overline{\mathbb{Z}} = \mathbb{Z}_0 \setminus \mathbb{Z}$

$\mathbb{Z}_0$ - przestrzeń wyjść układu $|H$

φ - symbol braku uszkodzenia w układzie

x - symbol operacji złożenia kodów (def. 2.2.4)

$\lfloor a \rfloor$ - największa liczba całkowita mniejsza lub równa a

$\lceil a \rceil$ - najmniejsza liczba całkowita większa lub równa a

$\{x, y, \ldots, z\}$ - zbiór zawierający wyszczególnione leementy

$\{\overline{a, b}\}$ - zbiór wszystkich liczb całkowitych od a do b włącznie

$(A|B)$ - A jest takie, że B

W pracy przedstawiono nowe metody projektowania samotestowalnych układów kontrolnych kodów stałowagowych. Podano algorytm projektowania wielopoziomowych układów progowych stosowanych w układach kontrolnych. W porównaniu z analogicznymi układami zaprojektowanymi wg innych metod, układy te charakteryzują się mniejszą liczbą bramek i łączną liczbą wejść bramek oraz wymagają mniejszej liczby testów potrzebnych do wykrycia wszystkich uszkodzeń pojedynczych typu sklejenie z "z", z = {0,1}. Podano ogólne zasady wyznaczania miar niezawodności samotestowalnych układów kontrolnych.

1. WPROWADZENIE

1.1. Sformułowanie problemu

Kod stałowagowy m z n (kod m/n) jest to taki kod detekcyjny, którego każde słowo zawiera dokładnie m jedynek i n-m zer. Kody m/n mają zdolność detekcji wszystkich błędów jednokierunkowych [24] (wszystkie przekłamane bity są wyłącznie typu 0 → 1 albo wyłącznie typu 1 → 0). Błędy takie są charakterystyczne dla wielu technologii wykonania urządzeń cyfrowych [7, 13, 39, 41, 42, 44, 55].

Ideę projektowania komputerów całkowicie samotestowalnych w czasie normalnej pracy po raz pierwszy sformułowano w [3] . Opiera się ona na szerokim zastosowaniu kodów detekcyjnych do kodowania wejść, wyjść i stanów wewnętrznych układów cyfrowych, zwanych wtedy układami samosprawdzalnymi (ang. self-checking).

W [10] wprowadzono pojęcie układu całkowicie samosprawdzalnego (ang. totally self-checking).

Układ całkowicie samosprawdzalny (TSC) dla uszkodzeń ze zbioru F jest to układ, który jest:

1/ samotestowalny (tj. taki, którego każde uszkodzenie ze zbioru F jest automatycznie wykrywane przez słowa kodowe występujące na wejściu układu w czasie normalnej pracy) i

2/ zabezpieczony przed uszkodzeniami (ang. fault-secure) (jeżeli w układzie takim jest uszkodzenie ze zbioru F, a na wejściu jest słowo kodowe, to na wyjściu nie występują niewykrywalne błędy).

Jako zbiór uszkodzeń F często przyjmuje się zbiór wszystkich uszkodzeń pojedynczych lub wielokrotnych jednokierunkowych typu "sklejenie z zerem" (s/0) lub "sklejenie z jedynką" (s/1). Zastosowanie układów TSC jest pożądane w systemach cyfrowych o podwyższonej niezawodności, w których wystąpienie niewykrywalnych błędów jest niedopuszczalne [4, 9, 12, 28, 42, 43, 45, 56]. Układy TSC, stanowiące bardzo ważną podklasę układów samosprawdzalnych (ich klasyfikację i własności omówiono dokładnie w [54, 56]) posiadają m. in. następujące zalety:

1/ wykrywane są zarówno uszkodzenia trwałe jak i przemijające,
2/ błędy są wykrywane natychmiasotowo po wystąpieniu, co zapobiega gromadzeniu się błędnych informacji,
3/ wymagana diagnostyka software'owa może być wyeliminowana lub znacznie uproszczona.

Warto tutaj odnotować fakt, że według danych z [51] uszkodzenia przemijające stanowią około 80 - 90 % uszkodzeń występujących we współczesnych systemach cyfrowych, a ich diagnostyka pochłania ponad 90 % kosztów odnowy systemów cyfrowych.

Sprawdzanie poprawności działania układu TSC zapewnia podłączony do jego wyjścia układ kontrolny, który sygnalizuje wystąpienie słów niekodowych. Ze względu na możliwość wystąpienia uszkodzeń w samym układzie kontrolnym projektuje się go jako układ samotestowalny; własność zabezpieczenia przed uszkodzeniami nie jest istotna dla poprawnego funkcjonowania układu kontrolnego - wystarczy aby każdorazowemu wystąpieniu słowa niekodowego na jego wejściu odpowiadało wygenerowanie przezeń dowolnego z wyjść stanowiących sygnał alarmowy.

Pożądane cechy samotestowalnego układu kontrolnego (w skrócie: STC) to: małe rozmiary, krótki czas propagacji sygnału z wejścia na wyjście oraz łatwa testowalność.

Pierwszym poważniejszym obiektem aplikacji kodów m/n i przeznaczonych dla nich ST układów kontrolnych był samosprawdzalny komputer ESS wyprodukowany przez firmę Bell Telephone dla potrzeb telefonii [11].

Od tego czasu powstało kilka innych opracowań samosprawdzalnych mini- i mikrokomputerów, w których użyto kody m/n i odpowiednie STC [14 - 16, 34]. Ważnymi obiektami zastosowań STC kodów m/n są TSC mikroprogramowane układy sterujące [13, 22], TSC liczniki [25] oraz inne TSC układy sekwencyjne - synchroniczne [21] i asyn-

chroniczne [19, 37, 46] (obszerniejszą bibliografię na ten temat można znaleźć w cytowanych pracach). Ostatnio zastosowano je również jako podukłady ST układów kontrolnych bardziej złożonych kodów detekcyjnych i korekcyjnych [41].

Przeważająca większość metod projektowania STC kodów m/n dotyczy układów dwuwyjściowych i realizowanych z użyciem pojedynczych bramek [3, 10, 18, 35, 44, 48], które będą rozważane również w tej pracy. Jedynie metoda z [27] dotyczy układów trzywyjściowych. We wszystkich pracach dotyczących metod projektowania takich układów (również w niniejszej pracy) przyjmuje się, że wyjścia (01), (10) - słowa kodu 1/2 - oznaczają wystąpienie słowa kodowego na wejściu układu kontrolnego i brak uszkodzeń wewnątrz samego układu; natomiast wyjścia (00) lub (11) oznaczają wystąpienie słowa niekodowego na wejściu układu kontrolnego albo wykrycie uszkodzenia w układzie kontrolnym.
Metody te z powodzeniem zastosowano do realizacji STC w układach LSI wykonanych w technologii CMOS z pojedynczym kanałem typu p [14, 15], charakteryzującej się m. in. tym, że nie każdemu funktorowi występującemu w zapisie funkcji logicznej układu odpowiada bramka logiczna w rzeczywistym układzie [26]. W przypadku implementacji STC kodów m/n z zastosowaniem PLA [57] stosowalność wyżej wymienionych metod projektowania jest częściowo ograniczona ze względu na występowanie w PLA nie tylko uszkodzeń modelowanych jako s/z (np. zwarcie między sąsiednimi liniami). Warto odnotować również udane próby realizacji STC kodów m/n z zastosowaniem technologii wielowartościowych układów I^2L [23], oraz układów progowych I^2L 52, oczywiście z zastosowaniem diametralnie różnych metod wymagających wniknięcia w strukturę elektryczną układu.

Pierwsze ogólne metody projektowania STC kodów m/n, z wyjątkiem kodów 1/3, 1/7 i dualnych do nich kodów 2/3 i 6/7, zostały podane przez Andersona i Metze'a w [3]. Większość opublikowanych prac dotyczy STC kodów kompletnych (tzn. takich, o których się zakłada, że na wejściu układu występują wszystkie $\binom{n}{m}$ słowa kodowe). Założenie kompletności kodu jest uzasadnione, gdy nie jest znany à priori zbiór słów kodowych występujących w czasie normalnej pracy [48]. Znane są także metody projektowania ST układów kontrolnych pewnej klasy kodów stałowagowych niekompletnych, tj. kodów złożonych z innych kodów stałowagowych (kodów złożonych

w sensie definicji Smith'a [48]); np. w [10] podano rozwiązanie dla kodu z inwersyjnym dopełnieniem (ang. double-rail code), a w [48] podano ogólne rozwiązanie dla dowolnego kodu złożonego z kodów nieuprządkowanych (jest to klasa kodów detekcyjnych mających zdolność wykrywania wszystkich błędów jednokierunkowych. Najważniejsze z nich są kody m/n oraz kody Bergera [5]).

Większość cytowanych prac prezentuje metody projektowania STC jako układów kombinacyjnych. Jedynie dla kodów 1/3 i 2/3 układ kontrolny samotestowalny dla każdego uszkodzenia pojedynczego typu s/z, z = {0,1}, nie może być kombinacyjny, co dowiedziono w [47]. Przykład realizacji sekwencyjnej STC tych kodów podano w [18].

Dla oceny jakości układów zaprojektowanych różnymi metodami przyjmowano następujące parametry układu:

1/ liczbę bramek,
   a/ jako miarę złożoności układu [35, 48],
   b/ jako miarę poboru mocy [14, 15],
2/ liczbę wejść bramek jako miarę złożoności układu [35, 48],
3/ liczbę tranzystorów jako miarę złożoności układu [14, 15],
4/ liczbę poziomów bramek, jako miarę opóźnienia wnoszonego przez układ [14, 15, 35, 48],
5/minimalną liczbę testów będących słowami danego kodu m/n wystarczających do wykrycia wszystkich uszkodzeń pojedynczych typu s/z, jako miarę diagnozowalności układu (jest to istotne np. w czasie testowania poprodukcyjnego lub, gdy w czasie pracy systemu układ wygeneruje sygnał alarmowy) [35, 48].

Analiza literatury problemu projektowania samosprawdzalnych układów cyfrowych wykorzystujących kody nieuporządkowane oraz udane próby rozwiązania dla prostych układów zainspirowały autora do poszukiwania nowych metod projektowania STC kodów m/n i sformułowania tezy:

"możliwe jest zaprojektowanie STC kodów m/n o lepszych parametrach niż układy otrzymane przez zastosowanie dotychczas znanych metod".

Na tle prowadzonych badań [40] można również przypuszczać, że STC kodów m/n zaprojektowane nowymi metodami można efektywnie zastosować do projektowania STC kodów Bergera.

1.2. Cel, zakres i treść pracy

Celem przedstawionej pracy jest opracowanie metod projektowania struktury logicznej samotestowalnych układów kontrolnych (STC) kodów stałowagowych m/n, pozwalających uzyskać układy o mniejszej złożoności i łatwiej testowalne niż analogiczne układy zaprojektowane dotychczas znanymi metodami.

Przyjmuje się następujące założenia projektowe:

1) STC jest kombinacyjny o dwóch wyjściach kodowanych kodem 1/2,
2) kod m/n jest kompletny,
3) wszystkie uszkodzenia występujące w układzie (zbiór F) są modelowane jako s/z, z = {0,1} - pojedyncze lub wielokrotne jednokierunkowe,
4) funkcję przełączającą realizowaną przez układ podaje się w wersji bezinwersyjnej, tj. z użyciem wyłącznie funktorów AND i OR.

Praca składa się z sześciu rozdziałów. W rozdziale drugim omówiono podstawowe pojęcia dotyczące STC i podano najważniejsze własności kodów m/n, funkcji pozytywnie gładkich i układów bezinwersyjnych.

W ostatnich dwóch częściach tego rozdziału (pp. 2.4 i 2.5) wykazano cztery ogólne własności dotyczące kombinacyjnych STC kodów m/n .

W rozdziale trzecim podano algorytm projektowania wielopoziomowych układów progowych, w których wszystkie wagi wejść są równe 1. Układy te są mniej złożone niż analogiczne układy zaprojektowane dotychczas znanymi metodami.

Zasadniczą część pracy stanowi rozdział 4, w którym przedstawiono nowe metody projektowania STC wyróżnionych klas kodów m/n. Zasada działania tych STC polega na translacji kodu m/n w kod 2/4 przy pomocy t-1 ($t \geqslant 2$). bloków pośrednich translatorów, na których wejściach i wyjściach występują kody stałowagowe, oraz zastosowaniu STC kodu 2/4 jako bloku wyjściowego. Istotne novum proponowanych metod projektowania polega na tym, że (oprócz wielostopniowej struktury układu i doboru kodów pośrednich) w przeciwieństwie do wszystkich metod znanych z literatury przedmiotu, w funkcjach przełączających realizowanych przez translatory występują iloczyny mniejszej liczby zmiennych niż waga kodu występującego na wyjściu translatora.

Takie rozwiązanie translatorów wraz z zastosowaniem wielopoziomowych układów progowych wykorzystujących wspólne podukłady (nie dotyczy pierwszego bloku dla kodów 1/n i (n-1)/n) ma istotny wpływ na otrzymane wyniki projektowania. Wyniki te zamieszczono łącznie dla STC tych samych kodów, otrzymanych według metod proponowanych w tej pracy i układów o najmniejszej złożoności zaprojektowanych metodami znanymi z literatury, głównie z [35]. Porównania tego dokonano dla następujących parametrów:

1) liczba bramek,
2) liczba wejść bramek,
3) liczba poziomów bramek,
4) minimalna liczba testów.

Oszacowania tych parametrów (w oparciu o ogólne zależności) podano dla układów bezinwersyjnych zakładając brak ograniczeń na liczbę wejść pojedynczej bramki i jej obciążalności.

W rozdziale piątym zdefiniowano miary niezawodności STC, podano ogólne zasady ich wyznaczania, oraz zastosowano je do porównania kilku wybranych STC otrzymanych różnymi metodami.

Najważniejsze wyniki otrzymane w pracy, możliwości ich praktycznego zastosowania oraz przewidywane kierunki badań omówiono w rozdziale szóstym.

2. POJĘCIA PODSTAWOWE

2.1. Podstawowe definicje dotyczące kombinacyjnych układów samotestowalnych i kontrolnych

Niech $\mathbb{H}$ będzie n-wejściowym i q-wyjściowym układem kombinacyjnym, $\mathcal{X}_0 = \{(x_1, x_2 \ldots x_n): x_i \in \{0,1\}\}$ - przestrzenią wejść, $\mathcal{Z}_0 = \{(z_1, z_2 \ldots z_q): z_j \in \{0,1\}\}$ - przestrzenią wyjść układu.

Oznaczmy przez F zbiór niektórych (najbardziej prawdopodobnych) uszkodzeń logicznych układu $\mathbb{H}$, zaś przez φ - brak uszkodzenia. Zbiór $F^* = \{\varphi\} \cup F$ nazwiemy zbiorem stanów niezawodnościowych układu.

Układ $\mathbb{H}$ odwzorowuje przestrzeń wejść $\mathcal{X}_0$ i zbiór F^* w przestrzeń wyjść $\mathcal{Z}_0$

$$H: \quad \mathcal{X}_0 \times F^* \longrightarrow \mathcal{Z}_0 \ ,$$

tzn.

$$(\forall X \in \mathcal{X}_0) \quad (\forall f \in F^*) \left(H(X,f) = Z \in \mathcal{Z}_0\right).$$

W przestrzeni wejść $\mathcal{X}_0$ wyróżnić można pewien jej podzbiór $\mathcal{X}$, zwany wejściową przestrzenią kodową, zawierający wszystkie wektory wejściowe X występujące w czasie normalnego (bez uszkodzeń) funkcjonowania układu, tzw. wejściowe słowa kodowe. Zbiór wyjść nieuszkodzonego układu $\mathbb{H}$, odpowiadających wejściowym słowom kodowym, zwanych wyjściowymi słowami kodowymi, tworzy wyjściową przestrzeń kodową $\mathcal{Z} \subset \mathcal{Z}_0$.

Gdy układ $\mathbb{H}$ jest nieuszkodzony, relację: "układ $\mathbb{H}$ odwzorowuje wejście X w wyjście Z", zapisujemy w formie uproszczonej

$$H(X) = Z \ .$$

Wpływ uszkodzeń wewnętrznych na funkcjonowanie układu $\mathbb{H}$

Załóżmy, że w układzie $\mathbb{H}$ jest uszkodzenie $f \in F$, a na jego wejściu występuje wektor $X \in \mathcal{X}$. Wpływ uszkodzenia f na działanie układu $\mathbb{H}$ może objawiać się na wyjściu układu na trzy różne sposoby:

1) $H(X,f) = H(X)$

- układ działa poprawnie, a o uszkodzeniu f mówi się, że jest maskowane dla wektora wejściowego X

W szczególności jeżeli dla pewnego uszkodzenia f zachodzi relacja

$$(\forall X \in \mathbb{X})\left(H(X,f) = H(X)\right),$$

to mówimy, że układ $\mathbb{H}$ maskuje uszkodzenie f dla wejściowej przestrzeni kodowej $\mathbb{X}$. Natomiast jeżeli dla uszkodzenia f zachodzi relacja

$$(\forall X \in \mathbb{X}_0)\left(H(X,f) = H(X)\right)$$

to połączenie, w którym ono wystąpiło, nazywa się całkowicie redundancyjnym dla wejściowej przestrzeni kodowej $\mathbb{X}_0$. Układ $\mathbb{H}$, w którym nie ma ani jednego połączenia całkowicie redundancyjnego dla przestrzeni wejść $\mathbb{X}_0$ nazywamy <u>układem nieredundancyjnym</u>.

2) $\left(H(X,f) \neq H(X)\right) \wedge \left(H(X,f) \in \mathbb{Z}\right)$

- na wyjściu układu $\mathbb{H}$ występuje słowo niekodowe (lub: błąd wykrywalny). O uszkodzeniu f mówi się, że jest <u>testowalne</u> przez wektor wejściowy X, który jest <u>testem</u> dla uszkodzenia f.

3) $\left(H(X,f) \neq H(X)\right) \wedge \left(H(X,f) \in \mathbb{Z}\right)$

- na wyjściu układu występuje błędne słowo kodowe (lub: niewykrywalny błąd).

<u>Definicja 2.1.1</u> [3]
Układ $\mathbb{H}$ nazywa się <u>układem samotestowalnym dla zbioru uszkodzeń F</u> (w skrócie: układem ST), jeżeli spełnia warunek

$$\left(\forall f \in F\right)\left(\exists X \in \mathbb{X} \mid H(X,f) \in \overline{\mathbb{Z}}\right).$$

W układzie ST dla każdego uszkodzenia f ze zbioru F istnieje wejściowe słowo kodowe, które jest testem dla tego uszkodzenia. Zatem w układzie ST każde uszkodzenie f ze zbioru F jest wykrywane przez pewne słowo kodowe w czasie pracy układu.

<u>Wpływ błędów wejściowych na funkcjonowanie sprawnego układu $\mathbb{H}$</u>

Załóżmy, że układ $\mathbb{H}$ jest dołączony do wyjścia innego układu - $\mathbb{G}$, którego wejściową przestrzeń kodową (niekodową) stanowi

zbiór $\mathbb{Y}$ ($\overline{\mathbb{Y}}$). Wskutek wystąpienia uszkodzeń w układzie $\mathbb{G}$ lub słów niekodowych na wejściu układu $\mathbb{G}$, na jego wyjściu mogą występować błędy.

Oznaczmy przez $\overline{\mathbb{X}}_G$ zbiór wszystkich słów niekodowych występujących na wyjściu układu $\mathbb{G}$ wskutek dwóch niejednoczesnych zdarzeń: uszkodzeń ze zbioru F_G albo wskutek wystąpienia słów niekodowych na jego wejściu, tzn.

$$\overline{\mathbb{X}}_G = \left\{ X \in \overline{\mathbb{X}} \mid \left(X = G(Y,f) \;,\; Y \in \mathbb{Y} \;,\; f \in F_G\right) \vee \left(X = G(Y),\; Y \in \overline{\mathbb{Y}}\right) \right\}.$$

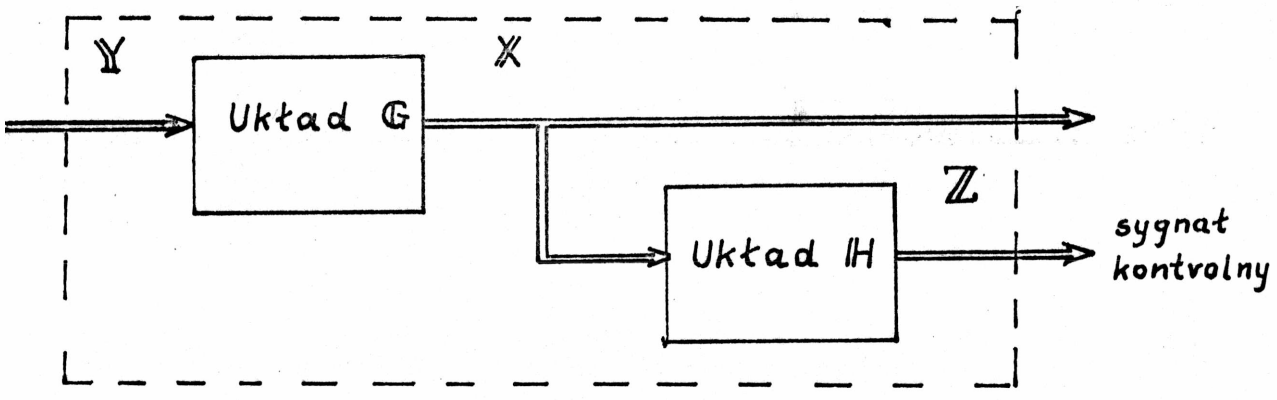
Zbiór $\overline{\mathbb{X}}_G$ nazywamy <u>zbiorem potencjalnych błędów generowanych przez układ $\mathbb{G}$</u>.

Jednakże w odróżnieniu od analogicznej definicji z [48], zbiór ten charakteryzuje układ $\mathbb{G}$ nie tylko pod względem jego możliwości generowania wykrywalnych błędów wyjściowych wskutek uszkodzeń wewnętrznych, ale również spowodowanych wystąpieniem słów niekodowych na jego wejściu.

Ogólnie - słowo występujące na wyjściu układu $\mathbb{G}$ (a na wejściu układu $\mathbb{H}$) można zakwalifikować ze względu na poprawność jako jeden z trzech następujących przypadków:

1) poprawne słowo kodowe,
2) błędne słowo kodowe (niewykrywalny błąd),
3) słowo niekodowe (wykrywalny błąd).

Zaprojektowanie układu $\mathbb{G}$ np. jako układu całkowicie samosprawdzalnego dla jego najbardziej prawdopodobnych uszkodzeń wewnętrznych ma na celu wyeliminowanie przypadku 2. Istnieje wtedy możliwość wykrycia błędów na wyjściu układu $\mathbb{G}$ poprzez sprawdzanie przynależności wektorów wyjściowych układu $\mathbb{G}$ do zbioru $\mathbb{X}$ lub $\overline{\mathbb{X}}_G$. Przypuśćmy, że funkcję tą wykonuje układ $\mathbb{H}$. Schemat połączenia układów $\mathbb{G}$ i $\mathbb{H}$ przedstawia rys. 2.1.1.



Rys. 2.1.1. Ogólny schemat systemu samosprawdzalnego

Reakcja układu $\mathbb{H}$ na słowo niekodowe X generowane przez układ $\mathbb{G}$ może być dwojakiego rodzaju:

1) $H(X) \in \mathbb{Z}$ - słowo niekodowe X jest niewykrywalne (maskowane) przez układ $\mathbb{H}$,

2) $H(X) \notin \mathbb{Z}$ - słowo niekodowe X jest wykrywalne przez układ $\mathbb{H}$.

Podział ten stanowi podstawę do wyróżnienia następujących klas układów - definicje 2.1.2 ÷ 2.1.4.

Definicja 2.1.2 [48]

Układ $\mathbb{H}$ nazywa się układem kontrolnym układu $\mathbb{G}$, jeżeli spełnia warunek

$$(\forall X \in \mathbb{X})\left(H(X) \in \mathbb{Z}\right) \wedge \left(\forall X \in \mathbb{X}_G\right)\left(H(X) \notin \mathbb{Z}\right).$$

Układ taki każde wejściowe słowo kodowe odwzorowuje w wyjściowe słowo kodowe, a każde wejściowe słowo niekodowe ze zbioru $\mathbb{X}_G$, generowane przez układ $\mathbb{G}$ odwzorowuje w wyjściowe słowo niekodowe. Układ tego typu jest przydatny tam, gdzie à priori znane są możliwości generowania błędów przez układ $\mathbb{G}$, którego wyjścia mają być kontrolowane przez układ $\mathbb{H}$. Jeżeli zbiór $\mathbb{X}_G$ nie jest znany przyjmuje się, że $\mathbb{X}_G = \overline{\mathbb{X}}$.

Definicja 2.1.3 [20]

Układ $\mathbb{H}$ nazywa się kodowo-rozłącznym (ang. code-disjoint) translatorem przestrzeni kodowej $\mathbb{X}$ na przestrzeń kodową $\mathbb{Z}$, jeżeli spełnia warunek

$$(\forall X \in \mathbb{X})\left(H(X) \in \mathbb{Z}\right) \wedge \left(\forall X \in \overline{\mathbb{X}}\right)\left(H(X) \notin \mathbb{Z}\right).$$

Kodowo-rozłączny translator jest szczególnym przypadkiem (najczęściej wykorzystywanym) układu kontrolnego dla zbioru błędów $\mathbb{X}_G$ - wtedy, gdy $\mathbb{X}_G = \overline{\mathbb{X}}$. Układ taki każde wejściowe słowo kodowe odwzorowuje w wyjściowe słowo kodowe, a każde wejściowe słowo niekodowe - w wyjściowe słowo niekodowe.
Szczególnym przypadkiem kodowo-rozłącznego translatora jest układ kontrolny.

Definicja 2.1.4.

Układ $\mathbb{H}$ nazywa się układem kontrolnym kodu C_X, jeżeli

$\mathcal{X} = C_X$ i kod wyjściowy jest kodem 1/2.

Wprowadzona klasyfikacja ma charakter umowny, a jej celem jest precyzyjne określenie warunków koniecznych i wystarczających dla poprawnego funkcjonowania układów kontrolnych o strukturze wielostopniowej, którą mają układy rozważane w dalszej części pracy.

Wpływ uszkodzeń wewnętrznych na funkcjonowanie układu kontrolnego $\mathbb{H}$

Załóżmy, że rozpatrywany układ $\mathbb{H}$ jest układem kontrolnym układu $\mathbb{G}$ (lub kodu C_X). Zastosowanie układu $\mathbb{H}$ ma na celu zapobieżenie dalszej propagacji wykrywalnych błędów poprzez sygnalizowanie wykrycia słów niekodowych na wyjściu układu $\mathbb{G}$.

W związku z tym mówimy, że:

1) układ kontrolny $\mathbb{H}$ funkcjonuje poprawnie, jeżeli wystąpienie słowa niekodowego na jego wejściu jest każdorazowo sygnalizowane,

i alternatywnie

2) układ kontrolny $\mathbb{H}$ funkcjonuje błędnie, jeżeli nie sygnalizuje wystąpienia słowa niekodowego na swoim wejściu.

Łatwo stwierdzić, że układ $\mathbb{H}$ bez uszkodzeń, który spełnia warunki definicji 2.1.2 (lub 2.1.4) zawsze działa poprawnie.

Przyczyną zawodnego działania układu kontrolnego $\mathbb{H}$ są uszkodzenia wewnętrzne. Jeżeli są spełnione warunki: $(X \notin \mathcal{X} \wedge f \in F)$ - konieczny, i $H(X,f) \in \mathcal{Z}$ - wystarczający, to układ kontrolny $\mathbb{H}$ funkcjonuje błędnie. Należy tu podkreślić, że wystąpienie zdarzenia $\{X \in \mathcal{X}, f \in F\}$ nie jest równoznaczne z błędnym funkcjonowaniem układu kontrolnego $\mathbb{H}$.

Wpływ uszkodzeń wewnętrznych można prawie całkowicie wyeliminować projektując układ $\mathbb{H}$ jako układ samotestowalny dla uszkodzeń najbardziej prawdopodobnych (ze zbioru F).

Jest wtedy wysoce prawdopodobne, że:

1) uszkodzenie układu $\mathbb{H}$ albo ustąpi samo (jeżeli ma charakter przemijający) albo zostanie wykryte zanim wystąpi wejściowe słowo niekodowe,

albo

2) pomimo uszkodzenia wewnętrznego wejściowe słowo niekodowe zostanie wykryte.

Zatem, podobnie jak w cytowanych pracach, przy projektowaniu samotestowalnych układów kontrolnych (rozdz. 4) nie bierze się pod uwagę możliwości wystąpienia zdarzenia $\{X \in \overline{X} \wedge f \in F\}$. Natomiast uwzględniono je przy ocenie niezawodności samotestowalnych układów kontrolnych w rozdz. 5.

2.2. Własności kodów stałowagowych (kodów m/n)

Definicja 2.2.1 [48]

Kod C_X, który spełnia warunek

$$(\forall X_1, X_2 \in C_X)\ (X_1 \leqslant X_2 \wedge X_2 \leqslant X_1)$$

nazywa się kodem nieuporządkowanym.

Kody nieuporządkowane mają zdolność wykrywania wszystkich błędów jednokierunkowych [48]. Do klasy kodów nieuporządkowanych należą m. in. kody stałowagowe.

Definicja 2.2.2 [56]

Kod stałowagowy (kod m/n) jest to podzbiór zbioru wektorów n-bitowych taki, że wektor jest słowem kodowym wtedy i tylko wtedy, jeżeli dokładnie m jego bitów ma wartość 1 (ma wagę m).
Spośród wszystkich kodów mających zdolność wykrywania wszystkich błędów jednokierunkowych największą pojemność przy ustalonej długości słowa n ma kod m/n taki, że $m = \lfloor n/2 \rfloor$ [24]. Pojemność maksymalna liczba słów kodowych kodu m/n wynosi

$$\left| C_{m/n} \right| = \binom{n}{m} = \frac{n!}{(n-m)! \cdot m!} \quad .$$

Kod m/n, który zawiera wszystkie $\binom{n}{m}$ słów kodowych będziemy nazywać kodem kompletnym; w przeciwnym razie nazywamy go kodem niekompletnym.
Podana niżej definicja złożenia kodów (ang. concatenation of codes) została wprowadzona przez Smitha [48] i nieco różni się od klasycznej definicji znanej z teorii kodowania, np. [33].

Definicja 2.2.3 [48]

Złożeniem wektora binarnego U o wymiarze u z wektorem binarnym V o wymiarze v nazywamy wektor binarny W o wymiarze w= u+v, w którym pierwsze u pozycji są takie jak w wektorze U, a pozostałe v pozycji są takie jak w wektorze V.

Definicja 2.2.4 [48]

Złożeniem dwóch kodów C_U i C_V o pojemnościach odpowiednio $|C_U|$ i $|C_V|$ jest kod C_W o pojemności $|C_W| = |C_U| \cdot |C_V|$ utworzony przez złożenie każdego słowa kodowego kodu C_U z każdym słowem kodowym kodu C_V.
Operację złożenia dwóch kodów można uogólnić na większą liczbę kodów.
Zbiór słów kodu złożonego C_W oznaczamy przez $C_U \times C_V$.

2.3. Własności funkcji pozytywnie gładkich i układów bezinwersyjnych

Funkcje przełączające realizowane przez samotestowalne układy kontrolne kodów m/n są pozytywnie gładkie (patrz własność W. 2.3.4), a układy zaprojektowane wg metod podanych m. in. w [3, 35, 44, 48] i w rozdz. 4 są bezinwersyjne, tzn. zawierają wyłącznie bramki AND i OR. Łatwość testowania układów realizujących funkcje gładkie [6, 17, 59] ma istotny wpływ na zastosowanie kodów m/n w układach samotestowalnych i stosowanie bezinwersyjnych realizacji tychże układów.

Definicja 2.3.1 [29]

Funkcja boole'owska p zmiennych $H_i(x_1, x_2, \ldots x_p)$ jest monotonicznie rosnąca wtedy i tylko wtedy, jeżeli jest spełniony warunek

$$\left(X_1 \leqslant X_2\right) \Rightarrow \left(H_i(X_1) \leqslant H_i(X_2)\right).$$

Szczególnym przypadkiem funkcji monotonicznie rosnącej jest funkcja pozytywnie gładka [17].

Definicja 2.3.2 [29]

Funkcja pozytywnie gładka (ang. positive unate function) jest to funkcja, w której postaci normalnej występują tylko zmienne proste.

Własności

W. 2.3.1) Jeżeli układ kombinacyjny jest bezinwersyjny, to realizuje funkcję pozytywnie gładką [29], lub ogólniej: funkcję monotonicznie rosnącą.

W. 2.3.2) Jeżelu układ $\mathbb{H}$ jest bezinwersyjny, to prawdziwa jest relacja ([48])

$$(\forall X \in \mathbb{X})\left(H(X,f_u^0) \leqslant H(X) \leqslant H(X,f_u^1)\right) ,$$

gdzie f_u^0, f_u^1 jest uszkodzeniem jednokierunkowym typu sklejenie z 0 lub z 1.

W. 2.3.3) Zbiór testów $\mathbb{T}_i^z$ dla uszkodzenia pojedynczego f_i^z, $z=\{0,1\}$, układu bezinwersyjnego, wykrywa również każde uszkodzenie jednokierunkowe $f_u^z = (f_i, f_j)$ [6].

W. 2.3.4) Każda funkcja przełączająca, której wejściowa przestrzeń kodowa składa się ze słów kodu nieuporządkowanego jest realizowalna przez układ bezinwersyjny [48].

W. 2.3.5) Istnieje dokładnie jedna postać minimalna funkcji pozytywnie gładkiej w postaci sumy iloczynów [29].

Podajemy teraz sposób, który ułatwia otrzymanie funkcji pozytywnie gładkiej w postaci minimalnej, w przypadku gdy $\mathbb{X} \subset C_{m/n}$.

Definicja 2.3.3

Mówimy, że implikant $m_j = x_{j1} \cdot x_{j2} \cdot \ldots \cdot x_{jm}$ odpowiada słowu X_j, co zapisujemy w postaci "$m_j \leftrightarrow X_j$", jeżeli na wszystkich pozycjach j_l, $l \in \{\overline{1,m}\}$, słowa X_j występują jedynki, a na pozostałych - zera.

Odpowiedni dobór wektorów wejściowych ze zbioru $\overline{\mathbb{X}}$ (traktowanych jako stany obojętne) umożliwia podanie funkcji H_i, $i \in \{\overline{1,q}\}$, w postaci równań [52]:

$$H_i = \sum_{m_j \in M_i'} m_j = \sum x_{j1} \cdot x_{j2} \cdot \ldots \cdot x_{jm}, \; i \in \{\overline{1,q}\} \qquad (2.3.1)$$

gdzie

$$M_i' = \{ m_j \mid (m_j \leftrightarrow X_j) \wedge (X_j \in \mathbb{X}) \wedge (H_i(X_j) = 1) \} .$$

W [20, 21] podano metodę modyfikacji funkcji układu $\mathbb{H}$ zadanej w postaci (2.3.1) do takiej postaci, która zapewnia samotestowalność układu $\mathbb{H}$ dla wszystkich uszkodzeń pojedynczych typu s/z. Można zauważyć, że metoda ta de facto doprowadza funkcje H_i do postaci minimalnej. Po takiej modyfikacji każda funkcja H_i zadana wzorem (2.3.1) może przybrać nową postać, w której wystąpią implikanty będące iloczynem mniej niż m zmiennych wejściowych lub nawet liczba tych implikantów zmniejszy się. Funkcję H_i zapisujemy teraz w nowej postaci

$$H_i = \sum_{m_j \in M_i} m_j , \qquad (2.3.2)$$

gdzie: $M_i = \{ m_j \mid m_j \in H_i \}$ oznacza zbiór implikantów występujących w funkcji H_i, w zasadzie po modyfikacji, chociaż nie jest to konieczne (wtedy funkcję H_i w postaci (2.3.1) traktuje się jako szczególny przypadek funkcji H_i w postaci (2.3.2)).

Np. funkcje: $H_1 = x_1x_3 + x_2x_3$, $H_2 = x_1x_2 + x_2x_4 + x_3x_4$, określone na zbiorze $\mathbb{X} = C_{2/4} \setminus \{(1001)\}$ $((1001) \leftrightarrow x_1x_4)$, po zmodyfikowaniu wg metody z [20, 21] przyjmują postać:

$$H_1 = x_1x_3 + x_2x_3, \qquad H_2 = x_1x_2 + x_4 .$$

2.4. Własności kombinacyjnych układów samotestowalnych wykorzystujących kody stałowagowe

Rozważamy układ $\mathbb{H}$ zdefiniowany w p. 2.1 taki, że $\mathbb{X} \subset C_{m/n}$ i $\mathbb{Z} \subset C_{p/q}$. Zakładamy, że funkcje układu $\mathbb{H}$ - H_i, $i \in \overline{1,q}$, są zadane w postaci (2.3.2). Uwzględnia się uszkodzenia logiczne typu s/z, z = {0,1}, pojedyncze (zbiór F_p) i jednokierunkowe (zbiór F_u), które występują w liniach wejściowych, wewnętrznych lub wyjściowych układu $\mathbb{H}$. Uszkodzenia te mogą mieć charakter trwały lub przemijający.

Warunki konieczne i wystarczające samotestowalności dla uszkodzeń pojedynczych układu $\mathbb{H}$ składającego się z rozłącznych

podukładów AND OR podane przez Diaza w [20, 21] zebrano i sformułowano łącznie jako tw. 2.4.1. Zakłada się, że funkcja H jest w postaci (2.3.2), i że każdemu implikantowi $m_j \in M_i$, $i \in \{\overline{1,q}\}$, odpowiada jedna bramka AND.

Definicja 2.4.1 [54]

Wielowyjściowy układ kombinacyjny $\mathbb{H}$ jest układem o rozłącznych podukładach (ang. bit-sliced circuit), jeżeli każdy podukład realizujący funkcję H_i, $i \in \{\overline{1,q}\}$, ma wspólne z innymi podukładami jedynie wejściowe linie pierwotne. W przeciwnym razie nazywa się układem o wspólnych podukładach (ang. shared logic).

Definicja 2.4.2 [20, 21]

Dzielnik $m_l(x_i)$ implikantu m_l względem zmiennej wejściowej x_i jest to wyrażenie otrzymane przez podstawienie $x_i=1$ w m_l.
Niech $m_i = x_{i1} \cdot x_{i2} \cdot \ldots \cdot x_{ia}$ oraz $m_j = x_{j1} \cdot x_{j2} \cdot \ldots \cdot x_{jb}$.
Relację " $\subset$ " zawierania się implikantów definiuje się następująco:

Definicja 2.4.3 [20, 21]

$$\left(m_i \subset m_j\right) \Longleftrightarrow \left(\left(\forall x_{ir},\ r \in \{1,a\}\right)\ \left(\exists x_{js},\ s \in \{1,b\} \mid x_{ir} = x_{js}\right)\right).$$

Wprowadzamy oznaczenia:

$$M' = \left\{m_j \mid (m_j \leftrightarrow x_j) \wedge (x_j \in \mathcal{X})\right\},$$

$$\left\{M' \setminus M_i'\right\} = \left\{m_j \mid (m_j \leftrightarrow x_j) \wedge (x_j \in \mathcal{X}) \wedge (m_j \notin M_i')\right\},$$

$\mathcal{X}$ - zbiór zmiennych x_i w m_l,

$m_l(\mathcal{X}) = \left\{m_l(x_i) \mid x_i \in \mathcal{X}\right\}$ - zbiór dzielników implikantu m_l względem wszystkich jego zmiennych.

Twierdzenie 2.4.1 [20, 21]

Dwupoziomowy AND-OR układ $\mathbb{H}$ o rozłącznych podukładach jest samotestowalny dla uszkodzeń ze zbioru F_p wtedy i tylko wtedy, jeżeli są spełnione warunki:

1) $\left(\forall f^z \in F_p\right)\ \left(\exists x_j \in \mathcal{X}_0 \mid H(x_j, f^z) \neq H(x_j)\right)$,

2) $$\left(\forall M_i,\ i \in \{\overline{1,q}\}\right)\left(\forall m_k \in M_i\right)\left(\exists x_j \in X \mid (m_k \subset m_j) \wedge \right.$$
$$\left. \wedge \sim(\exists m_p \in \{M_i \setminus \{m_k\}\} \mid m_p \subset m_j)\right),$$

3) $$\left(\forall M_i,\ i \in \{\overline{1,q}\}\right)\left(\forall m_k \in M_i\right)\ \left(\forall m_k(x_l) \in m_k(x)\right)$$
$$\left(\exists m_p \in \{M' \setminus M_i'\} \mid m_k(x_l) \subset m_p\right).$$

W tw. 2.4.1 war. 1 jest formalnym zapisem założenia, że układ jest nieredundancyjny, a warunki 2 i 3 określają kiedy układ jest samotestowalny odpowiednio dla uszkodzeń typu s/0 i s/1.
W [20] wykazano następującą własność, która rozszerza zakres stosowalności tw. 2.4.1 na uszkodzenia jednokierunkowe:

W. 2.4.1) Jeżeli dwupoziomowy AND-OR układ $\mathbb{H}$ o rozłącznych podukładach jest samotestowalny dla uszkodzeń pojedynczych, to układ ten jest również samotestowalny dla uszkodzeń jednokierunkowych, niezależnie od tego czy podukłady są wspólne, czy nie.

Rozszerzenie zakresu stosowalności tw. 2.4.1, np. dla układów wielopoziomowych o wspólnych podukładach, wynika z poniższego twierdzenia:

Twierdzenie 2.4.2

Jeżeli dwupoziomowy AND-OR układ $\mathbb{H}$ o rozłącznych podukładach realizujący funkcję H jest samotestowalny dla uszkodzeń pojedynczych (ze zbioru F_p), to każdy inny nieredundancyjny bezinwersyjny układ $\mathbb{H}'$ realizujący tą samą funkcję H, jest samotestowalny dla uszkodzeń jednokierunkowych (ze zbioru F_u).

Dowód

Załóżmy, że $\mathbb{H}$ jest dwupoziomowym AND-OR układem o rozłącznych podukładach, który jest samotestowalny dla uszkodzeń ze zbioru F_p. Oznacza to, że spełnione są warunki określone w tw. 2.4.1.

Rozważmy podukład układu $\mathbb{H}$ realizujący funkcję H_i, $i \in \{\overline{1,q}\}$, podaną w postaci (2.3.2).

Wprowadzamy oznaczenia:

$$\mathbb{X}_i = \{x_j \in \mathbb{X} \mid H_i(X_j) = 1\},$$

$$\mathbb{X}_i^0 = \{x_j \in \mathbb{X}_i \mid (\exists! \, m_k \in M_i \mid m_k \subset m_j)\},$$

$$\mathbb{X}_i^1 = \left\{x_j \in \{\mathbb{X} \setminus \mathbb{X}_i\} \middle| \left((\exists m_k \in M_i) \wedge (\exists m_k(x_1) \in m_k(\infty))\right) \middle| \left(m_k(x_1) \subset m_j\right)\right\}.$$

W zbiorze F_p wyróżniamy podzbiory rozłączne uszkodzeń:

$$F_p = F_{WP} \cup \left(\bigcup_{i=1}^{q} F_{pi}\right),$$

gdzie: F_{WP} - zbiór uszkodzeń pierwotnych linii wejściowych układu,
F_{pi} - zbiór uszkodzeń podukładu realizującego funkcję H_i.

Na mocy tw. 2.4.1 zbiór $\mathbb{X}_i^z$, $z = \{0,1\}$ zawiera wszystkie słowa kodowe będące testami dla uszkodzeń typu s/z ze zbioru F_{pi}, a zbiór $\mathbb{X}$ ogólnie wszystkie testy, w tym również dla uszkodzeń ze zbioru F_{WP}.
Załóżmy, że układ $\mathbb{H}'$ jest nieredundancyjny i bezinwersyjny. Tymczasowo zakładamy również, że $\mathbb{H}'$ jest układem o rozłącznych podukładach.
Analogicznie jak w przypadku układu $\mathbb{H}$, w zbiorze F_p' uszkodzeń układu $\mathbb{H}'$ wyróżniamy podzbiory rozłączne:

$$F_p' = F_{WP}' \cup \left(\bigcup_{i=1}^{q} F_{pi}'\right).$$

W pierwszej kolejności rozważymy uszkodzenie $f^z \in F_{pi}'$.
Oznaczmy przez g funkcję realizowaną w połączeniu, w którym wystąpiło uszkodzenie f^z. Funkcja ta jest oczywiście pozytywnie gładka i w ogólnym przypadku zależy od wszystkich zmiennych wejściowych: $g = g(x_1, x_2, \ldots, x_n)$.
Ponieważ z założenia układ $\mathbb{H}'$ jest nieredundancyjny, więc funkcja $g(x_1, x_2, \ldots, x_n)$ nie jest tożsamościowo równa 0 ani 1.
Funkcję H_i można zatem zapisać w postaci ogólnej $H_i =$

$= H_i(x_1, x_2, \ldots, x_n, g)$, a następnie w postaci $H_i = g \cdot H_{i1} + H_{i2}$, gdzie H_{i1}, H_{i2} są funkcjami pozytywnie gładkimi. Funkcje te spełniają warunki:

$$1)\quad \left(\forall X_j \in \mathcal{X}_i\right)\left(\left(H_{i2}(X_j) = 1\right) \Rightarrow \left(H_i(X_j) = 1\right) \wedge \left(X_k \in \mathcal{X}_i \mid H_{i2}(X_k) = 0\right)\right), \tag{2.4.1}$$

$$2)\quad \left(\forall X_j \in \mathcal{X}\right)\left(\left(H_{i1}(X_j) = 1\right) \Rightarrow \left(H_i(X_j) = 1\right) \wedge \left(\exists X_k \in \mathcal{X}_0 \mid H_{i1}(X_k) = 1\right) \wedge \left(H_i(X_k) = 0\right)\right). \tag{2.4.2}$$

Oznaczmy przez M_{i1}, M_{i2} zbiory implikantów występujących w funkcjach H_{i1}, H_{i2}. Spełnione są warunki:

$$1)\quad \left(M_{i2} \subset M_i'\right) \wedge \left(M_{i2} \neq M_i'\right) \wedge \left(M_{i2} \neq \emptyset\right), \tag{2.4.3}$$

$$2)\quad \left(\forall m_j \in M_{i1}\right)\left(\exists m_k \in \{M_i' \setminus M_{i2}\} \mid m_j \subset m_k\right), \tag{2.4.4}$$

$$3)\quad \left(\exists m_1 \in M_{i1} \mid \left(m_1 \subset m_p,\ m_p \in \{M_i' \setminus M_{i2}\}\right) \Rightarrow \left(m_1 \neq m_p\right)\right). \tag{2.4.5}$$

Rozważmy teraz dwa przypadki uszkodzenia f^z.

1. z=0

Funkcja g jest tożsamościowo równa 0, toteż $H_i = H_{i2}$. Każde słowo $X_1 \in \mathcal{X}_i^0$ takie, że $\left(\left(\exists! m_p \in \{M_i' \setminus M_{i2}\}\right) \wedge \left(m_p \subset m_1\right)\right)$ jest testem dla uszkodzenia f^0. Z warunków 1 i 2 tw. 2.4.1 oraz (2.4.3) i (2.4.4) wynika, że słowo takie zawsze istnieje.

2. z=1

Funkcja g jest tożsamościowo równa 1 i stąd $H_i = H_{i1} + H_{i2}$. Z (2.4.5) wynika, że

$$\left(\left(m_1 \in M_{i1}\right) \wedge \left(m_p \in \{M_i' \setminus M_{i2}\}\right) \wedge \left(m_1 \neq m_p\right)\right) \Rightarrow \left(\exists m_p(x) \in m_p(x) \mid m_1 \subset m_p(x)\right). \tag{2.4.6}$$

Z war. 2 tw. 2.4.1 wynika, że $\left(\exists X_j \in \mathcal{X}_i^1 \mid m_p(x) \subset m_j\right)$.

Ponieważ $m_1 \subset m_p(x)$ i $m_p(x) \subset m_j$ więc $m_1 \subset m_j$. Stąd wynika, że X_j jest testem dla uszkodzenia f^1.

Ponieważ powyższe rozważania dotyczą dowolnego uszkodzenia pojedynczego $f^z \in F_{pi}$, więc każde uszkodzenie podukładu realizującego funkcję H_i jest wykrywane przez pewne słowo kodowe ze zbioru $\mathbb{X}_i \subset \mathbb{X}$.

Ponieważ założono, że $\mathbb{H}'$ składa się z rozłącznych podukładów, więc uszkodzenie f^z i-tego podukładu może spowodować błąd co najwyżej na jednym (i-tym) wyjściu układu $\mathbb{H}'$. Tym samym wykrycie uszkodzenia f^z na i-tym wyjściu jest równoważne wystąpieniu słowa niekodowego kodu $(m-1)/n$, jeżeli $z=0$, a kodu $(m+1)/n$, jeżeli $z=1$. Oznacza to, że układ $\mathbb{H}'$ jest samotestowalny dla uszkodzeń ze zbioru F'_{pi}.

Z kolei, ponieważ rozważania te odnoszą się do dowolnego z q rozłącznych podukładów wchodzących w skład układu $\mathbb{H}'$, więc tym samym układ $\mathbb{H}'$ jest samotestowalny dla uszkodzeń ze zbioru

$$\bigcup_{i=1}^{q} F'_{pi}.$$

Jeżeli chodzi o uszkodzenia ze zbioru F'_{WP}, to każde uszkodzenie pojedyncze typu s/z wejściowej linii pierwotnej jest równoważne wystąpieniu uszkodzeń jednokierunkowych typu s/z we wszystkich rozgałęzieniach tej linii, z których każde należy do pewnego zbioru F'_{pi}.

Ponieważ układ $\mathbb{H}'$ jest samotestowalny dla uszkodzenia ze zbioru

$\bigcup_{i=1}^{q} F'_{pi}$, więc na mocy własności W. 2.3.2 i W. 2.3.3 jest samotestowalny również dla uszkodzeń ze zbioru F'_{WP}.

Jeżeli $\mathbb{H}'$ jest układem o wspólnych podukładach, wystąpienie uszkodzenia s/z we wspólnym podukładzie jest równoważne wystąpieniu uszkodzeń jednokierunkowych s/z w odpowiedniej linii każdego podukładu rozłącznego (gdy $\mathbb{H}'$ składa się z rozłącznych podukładów). Zatem z tych samych względów co w przypadku uszkodzeń ze zbioru F'_{WP}, układ $\mathbb{H}'$ o wspólnych podukładach jest samotestowalny zarówno dla uszkodzeń ze zbioru F_p jak i F_u.

Q.E.D.

2.5. Własności układów kontrolnych kodów stałowagowych

Rozważamy układ $\mathbb{H}$ zdefiniowany w p. 2.1 taki, że $\mathbb{X} \subset C_{m/n}$ i $\mathbb{Z} \subset C_{p/q}$.
Zbiór $\overline{\mathbb{X}}$ dzielimy na trzy podzbiory rozłączne:

$$\overline{\mathbb{X}}^{-} = \left\{ X_j \in \overline{\mathbb{X}} \,\middle|\, (\exists X_k \in \mathbb{X} \mid X_j < X_k) \right\},$$

$$\overline{\mathbb{X}}^{\phi} = \left\{ C_{m/n} \setminus \mathbb{X} \right\} \; ; \; (\text{jeżeli } \mathbb{X} = C_{m/n}, \text{ to } \overline{\mathbb{X}}^{\phi} = \emptyset),$$

$$\overline{\mathbb{X}}^{+} = \left\{ X_j \in \overline{\mathbb{X}} \,\middle|\, (\exists X_k \in \mathbb{X} \mid X_k < X_j) \right\}.$$

Podobnie dzieli się zbiór $\overline{\mathbb{Z}}$, a jego podzbiory $\overline{\mathbb{Z}}^{-}$, $\overline{\mathbb{Z}}^{\phi}$, $\overline{\mathbb{Z}}^{+}$ definiuje się analogicznie.

Twierdzenie 2.5.1

Jeżeli układ $\mathbb{H}$ realizuje funkcję pozytywnie gładką H i jest kodowo-rozłącznym translatorem przestrzeni kodowej $\mathbb{X}$ na przestrzeń kodową $\mathbb{Z}$, to:

1) $\left(\forall X_j \in \overline{\mathbb{X}}^{-}\right)\left(H(X_j) \in \overline{\mathbb{Z}}^{-}\right)$,
2) $\left(\forall X_j \in \overline{\mathbb{X}}^{+}\right)\left(H(X_j) \in \overline{\mathbb{Z}}^{+}\right)$.

Dowód

Jeżeli układ spełnia założenia twierdzenia, to spełnione są warunki:

i) $\left(X_1 \leqslant X_2\right) \Rightarrow \left(H(X_1) \leqslant H(X_2)\right)$ (z def. 2.3.1),
ii) $\left(\forall X_j \in \overline{\mathbb{X}}\right)\left(H(X_j) \in \overline{\mathbb{Z}}\right)$ (z def. 2.1.3).

Ad 1) $\left(X_j \in \overline{\mathbb{X}}^{-}\right) \Rightarrow \left(\exists X_k \in \mathbb{X} \mid X_j < X_k\right) \Rightarrow \left(H(X_j) < H(X_k)\right)$.

Ponieważ równocześnie zachodzi

$$H(X_j) \in \overline{\mathbb{Z}} \;\wedge\; H(X_k) \in \mathbb{Z},$$

więc $H(X_j) \in \overline{\mathbb{Z}}^{-}$.

Ad 2) $\left(X_j \in \overline{\mathbb{X}}^+\right) \Rightarrow \left(\exists X_k \in \mathbb{X} \mid X_k < X_j\right) \Rightarrow \left(H(X_k) < H(X_j)\right)$.

Ponieważ równocześnie zachodzi

$$H(X_k) \in \mathbb{Z} \quad \wedge \quad H(X_j) \in \overline{\mathbb{Z}},$$

więc $H(X_j) \in \overline{\mathbb{Z}}^+$.

Q.E.D.

Z tw. 2.5.1 wynika, że jeżeli $\mathbb{X} = C_{m/n}$, a układ $\mathbb{H}$ realizuje funkcję pozytywnie gładką i jest kodowo-rozłącznym translatorem kodów stałowagowych, to

$$\sim\left(\exists X_j \in \mathbb{X}_0 \mid H(X_j) \in \mathbb{Z}^{\phi}\right) \text{ , gdyż } \quad \overline{\mathbb{X}}^{\phi} = \emptyset.$$

Twierdzenie 2.5.2

Jeżeli układ $\mathbb{H}$ spełnia warunki:

1) jest bezinwersyjny,
2) jest układem kontrolnym układu $\mathbb{G}$, na którego wyjściu występują:
   a) słowa kodowe ze zbioru $\mathbb{X} \subset C_{m/n}$,
   b) słowa niekodowe ze zbioru $\overline{\mathbb{X}}_G = \overline{\mathbb{X}}^- \cup \overline{\mathbb{X}}^+$,
3) w układzie $\mathbb{H}$ występują uszkodzenia jednokierunkowe,
4) nie występuje równocześnie wejściowe słowo niekodowe i uszkodzenie układu $\mathbb{H}$,

to $\overline{\mathbb{Z}}_H = \overline{\mathbb{Z}}^- \cup \overline{\mathbb{Z}}^+$;

(lub: w zbiorze $\overline{\mathbb{Z}}_H$ potencjalnych błędów generowanych przez układ $\mathbb{H}$ nie występują słowa niekodowe ze zbioru $\overline{\mathbb{Z}}^{\phi}$, niezależnie od tego czy $\mathbb{Z} = C_{p/q}$, czy nie).

Dowód opiera się na zastosowaniu tw. 2.5.1 oraz własności W. 2.3.1 i W. 2.3.2.

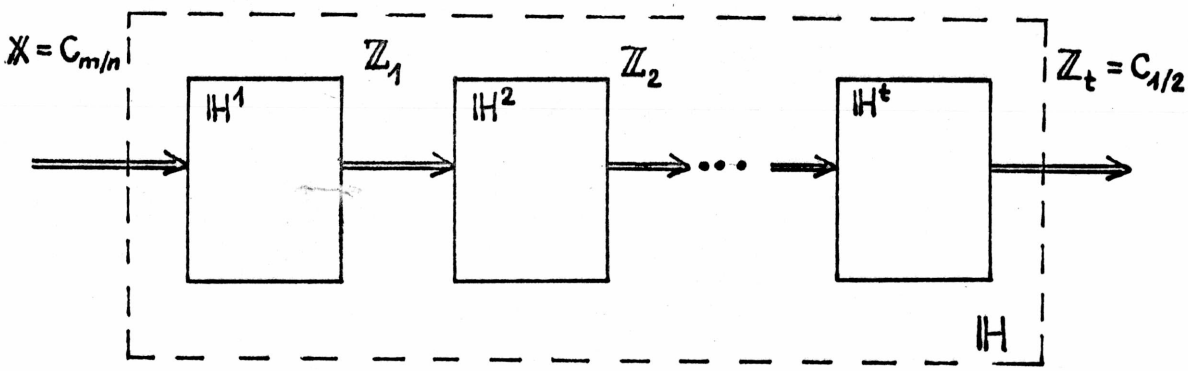
Wprowadzamy oznaczenia:

$H^i = \left\{H_1^i, H_2^i, \ldots, H_{n_i}^i\right\}$ - wektorowa funkcja **boole'owska realizowana** przez układ $\mathbb{H}^i$,

$\mathbb{Z}_i$ - wyjściowa przestrzeń kodowa układu $\mathbb{H}^i$, tzn. zbiór wektorów występujących na wyjściu układu $\mathbb{H}^i$, gdy na wejś-

ciu układu $\mathbb{H}$ występują słowa kodu m/n, a układ $\mathbb{H}$ jest nieuszkodzony,

F_p^i - zbiór uszkodzeń pojedynczych typu s/z układu $\mathbb{H}^i$.



Rys. 2.5.1. Struktura szeregowa samotestowalnego układu kontrolnego **kodu m/n**

Załóżmy, że: 1) n-wejściowy i dwuwyjściowy układ $\mathbb{H}$ ma strukturę jak na rys. 2.5.1 (tzn. składa się z połączonych szeregowo bloków $\mathbb{H}^i$, $i \in \{\overline{1,t}\}$, $t \geqslant 2$, z których tylko układ $\mathbb{H}^1$ ma wejścia pierwotne, 2) $\forall\, \mathbb{Z}_i$, $i \in \{\overline{1,t}\}$, $\mathbb{Z}_i \subset C_{m_i/n_i}$, przy czym $m_t=1$ i $n_t=2$, 3) $\mathbb{X} = C_{m/n}$.

Twierdzenie 2.5.3

Układ $\mathbb{H}$ spełniający przyjęte powyżej założenia jest samotestowalnym dla uszkodzeń jednokierunkowych układem kontrolnym kodu m/n, jeżeli jest bezinwersyjny, a bloki $\mathbb{H}^i$, $i \in \{\overline{1,t}\}$, spełniają warunki:

1) $\mathbb{H}^1$ jest samotestowalnym dla uszkodzeń ze zbioru F_p^1 kodowo-rozłącznym translatorem kodu m/n w kod m_1/n_1,
2) każdy układ $\mathbb{H}^i$, $i \in \{\overline{2,t}\}$, jest układem kontrolnym układu $\mathbb{H}^{i-1}$,
3) każdy zbiór $\mathbb{Z}_{i-1}$, $i \in \{\overline{2,t}\}$, zawiera wszystkie testy wystarczające do wykrycia uszkodzeń ze zbioru F_p^i.

Dowód

Warunek 1 zapewnia, że: a) każde słowo niekodowe na wejściu ukła-

du $\mathbb{H}$ spowoduje wystąpienie słowa niekodowego na wyjściu układu $\mathbb{H}^1$, b) dla każdego uszkodzenia pojedynczego w zbiorze $\mathbb{X}$ istnieje co najmniej jeden test, co zapisujemy formalnie w następującej postaci:

$$(\forall x \in \mathbb{X})(H^1(x) \in \mathbb{Z}_1) \wedge (\forall x \in \overline{\mathbb{X}})(H^1(x) \in \overline{\mathbb{Z}}_1) \qquad (2.5.1)$$

oraz

$$(\forall f \in F_p^1)\ (\exists x \in \mathbb{X} \mid H^1(x,f) \in \overline{\mathbb{Z}}_1)\ . \qquad (2.5.2)$$

Z (2.5.2) i własności W. 2.3.3 wynika, że

$$(\forall f \in F_u^1)\ (\exists x \in \mathbb{X} \mid H^1(x,f) \in \overline{\mathbb{Z}}_1)\ .$$

Z tw. 2.5.2 wynika, że $\overline{\mathbb{Z}}_{H_1} = \overline{\mathbb{Z}}_1^- \cup \overline{\mathbb{Z}}_1^+$.

Z warunku 2 wynika, że

$$(\forall z \in \overline{\mathbb{Z}}_{H_1})\ (H^2(z) \in \overline{\mathbb{Z}}_2). \qquad (2.5.3)$$

Z warunku 3 wynika, że

$$(\forall f \in F_p^1)\ (\exists x \in \mathbb{X} \mid H^2[H^1(x), f] \in \overline{\mathbb{Z}}_2)\ . \qquad (2.5.4)$$

Z (2.5.4) i własności W. 2.3.3 wynika, że

$$(\forall f \in F_u^2)\ (\exists x \in \mathbb{X} \mid H^2[H^1(x, f)] \in \overline{\mathbb{Z}}_2)\ .$$

Z tw. 2.5.2 wynika, że $\overline{\mathbb{Z}}_{H_2} = \overline{\mathbb{Z}}_2 \quad \overline{\mathbb{Z}}_2$.

Podobnie jest dla każdego układu $\mathbb{H}^i$, $i \in \{\overline{2,t}\}$ skąd wynika prawdziwość twierdzenia.

Q.E.D.

Uwaga: Podobne twierdzenie sformułowane zostało przez Andersona [2] (cyt. w [35] jako tw. 1), w którym od układów $\mathbb{H}^i$, $i \in \{\overline{2,t}\}$, wymaga się kodowej rozłączności. Na mocy tw. 2.5.3 - war. 2 - wystarcza, aby każdy układ $\mathbb{H}^i$, $i \in \{\overline{2,t}\}$, był układem kontrolnym układu $\mathbb{H}^{i-1}$.

3. WŁASNOŚCI, ZASTOSOWANIA I METODY PROJEKTOWANIA UKŁADÓW PROGOWYCH

Przy projektowaniu samotestowalnych układów kontrolnych kodów m/n stosuje się [3, 35, 44] (również w metodach podanych w rozdz. 4) układy progowe, w których wszystkie uwagi wejść są równe 1. O zastosowaniu takich układów, szczególnie w wersji wielopoziomowej, decydują duże możliwości stosowania wspólnych podukładów i ich łatwość testowania [44, 59].

3.1. Własności i zastosowania układów progowych

Niech $A = \{x_1, x_2, \ldots, x_n\}$ oznacza zbiór n wejściowych zmiennych binarnych, i - liczbę jedynek występujących w danej chwili na pozycjach ze zbioru A.

Definicja 3.1.1 [3]

Funkcja progowa $T(a \geq i)$ o jednakowych wagach sygnałów wejściowych równych 1 jest to funkcja określona wyrażeniem

$$T(a \geq i) = \sum_{\substack{j_1 \in \{\overline{1,n}\} \\ j_{11} \neq j_{12}}} x_{j1} \cdot x_{j2} \cdot \ldots \cdot x_{ji}, \qquad (3.1.1)$$

która przybiera wartość 1, jeżeli liczba jedynek na pozycjach ze zbioru A osiągnie lub przekroczy zadany próg i; (funkcja ta jest pozytywnie gładka [29]).

Dla funkcji progowej $T(a \geq i)$ stosowane bywa ([44, 59]) również inne oznaczenie: T_i^n. Będzie ono również stosowane w tej pracy, wtedy - gdy nie są istotne: nazwa zbioru zmiennych wejściowych oraz symbole i indeksacja zmiennych wejściowych.

Najważniejsze metody projektowania STC kodów m/n [3, 35, 44, 48] oparte są na wykorzystaniu układów progowych dwu- lub wielopoziomowych.

W [3] po raz pierwszy podano sposób wykorzystania dwupoziomowych układów progowych do projektowania STC. Podano tam również ogólne zasady projektowania wielopoziomowych układów progowych i zasuge-

rowano możliwość ich zastosowania w STC kodów m/n.
W [59] podano algorytm projektowania wielopoziomowych układów progowych realizujących pojedynczą funkcję $T(a \geq i)$ lub układów realizujących równocześnie wszystkie n funkcji $T(a \geq i)$, $i \in \{\overline{1,n}\}$, z użyciem dwuwymiarowych łańcuchów iteracyjnych. Tak zaprojektowane wielopoziomowe układy progowe zastosowano następnie w STC . kodów $m/2 \cdot m$ [44].
W [35] podano najefektywniejsze jak dotąd algorytmy projektowania STC kodów m/n ($m > 1$ i $n \neq 2 \cdot m$), również z zastosowaniem układów progowych dwupoziomowych. Zasugerowano tam również możliwość zastosowania układów progowych wielopoziomowych, których ogólne zasady projektowania sformułowano w [3].

Podamy teraz dwie metody projektowania układów progowych: pierwsza z nich, dotycząca układów dwupoziomowych, została podana w [3], druga zaś dotycząca układów wielopoziomowych - podana jako algorytm 3.3.2 - jest oryginalna i daje korzystniejsze wyniki niż metoda podana w [59] (podrozdział 3.5) . Układy progowe ocenia się na podstawie tych samych parametrów co STC kodów m/n, których metody projektowania podane w rozdz. 4 są zresztą oparte na zastosowaniu układów progowych zaprojektowanych wg algorytmu 3.3.2. Parametry charakteryzujące n-wejściowy układ progowy o progu i oznaczamy następująco:
- $Br(T_i^n)$ - liczba bramek,
- $We(T_i^n)$ - łączna liczba wejść bramek,
- $L(T_i^n)$ - liczba poziomów bramek,
- $|\mathbb{T}(T_i^n)|$ - liczba testów wykrywających wszystkie uszkodzenia pojedyncze typu s/z, $z = \{0,1\}$.

3.2. Układy progowe dwupoziomowe [3, 59]

Realizację dwupoziomową AND-OR funkcji $T(a \geq i)$ otrzymuje się przez podanie wszystkich słów kodu i/n na wejściu $\binom{n}{i}$ bramek AND. Wyjścia bramek AND są równocześnie wejściami pojedynczej bramki OR.
Np. dla i=2, n=4, $A = \{x_1, x_2, x_3, x_4\}$

$$T(a \geq 2) = x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4 .$$

Parametry tych układów wyrażają się wzorami (pomijamy możliwość wystąpienia ograniczeń na liczbę wejść bramki OR lub liczbę

rozgałęzień wejściowych linii pierwotnych):

$$Br\left(T_i^n\right) = \binom{n}{i} + 1 \,, \tag{3.2.1}$$

$$We\left(T_i^n\right) = (i+1)\cdot\binom{n}{i}, \tag{3.2.2}$$

$$L\left(T_i^n\right) = 2 \,, \tag{3.2.3}$$

$$\left|\mathbb{T}\left(T_i^n\right)\right| = \binom{n}{i} + \binom{n}{i-1} = \binom{n+1}{i} \,. \tag{3.2.4}$$

Oczywiście w szczególnych przypadkach, gdy i=1 lub i=n, układ progowy stanowi pojedyncza n-wejściowa bramka OR (gdy i=1) lub AND (gdy i=n), a liczba testów jest równa n+1.
Dla większych parametrów n oraz i stosowanie dwupoziomowej realizacji układu progowego na ogół jest nieopłacalne ze względu na nadmierną liczbę bramek i połączeń. Ponadto układy te wymagają stosunkowo dużej liczby testów wynoszącej $\binom{n+1}{i}$ [59]. Tamże wykazano, że $\left|\mathbb{T}\left(T_i^n\right)\right| = C_{i/n} \cup C_{(i-1)/n}$.

3.3. Układy progowe wielopoziomowe

Podzielmy zbiór A na dwa niepuste rozłączne podzbiory $A_{1,1}$, $A_{1,2}$ o licznościach odpowiednio $n_{1,1}$, $n_{1,2}$. Jeżeli $n \geq 3$, to każdą funkcję progową $T(a \geq i)$ można przedstawić jako złożenie bardziej elementarnych funkcji progowych w następującej postaci [3]

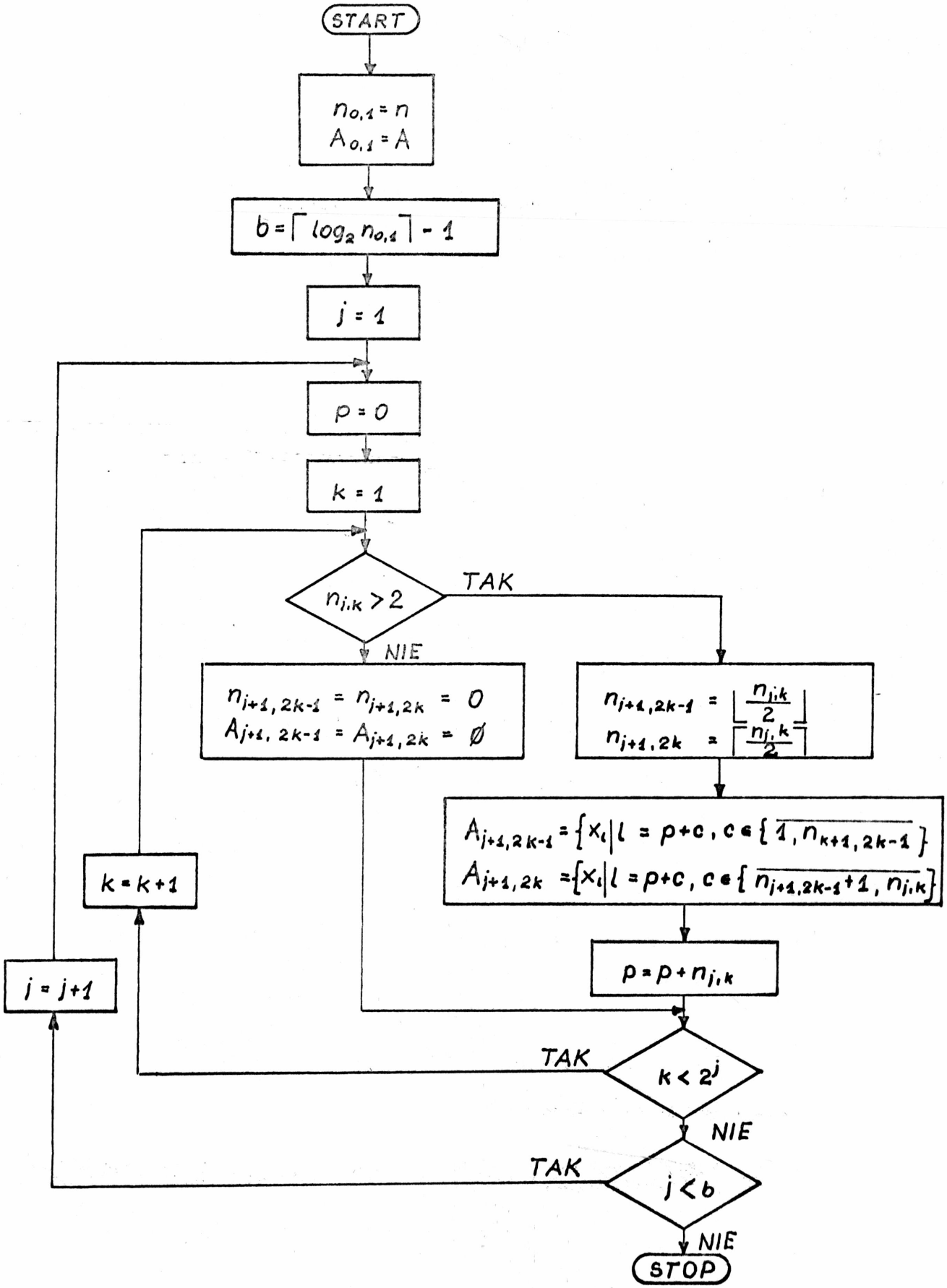
$$T(a \geq i) = \sum_{l=\underline{l}}^{\overline{l}} T(a_{1,1} \geq l) \cdot T(a_{1,2} \geq i-l) \,, \tag{3.3.1}$$

gdzie:

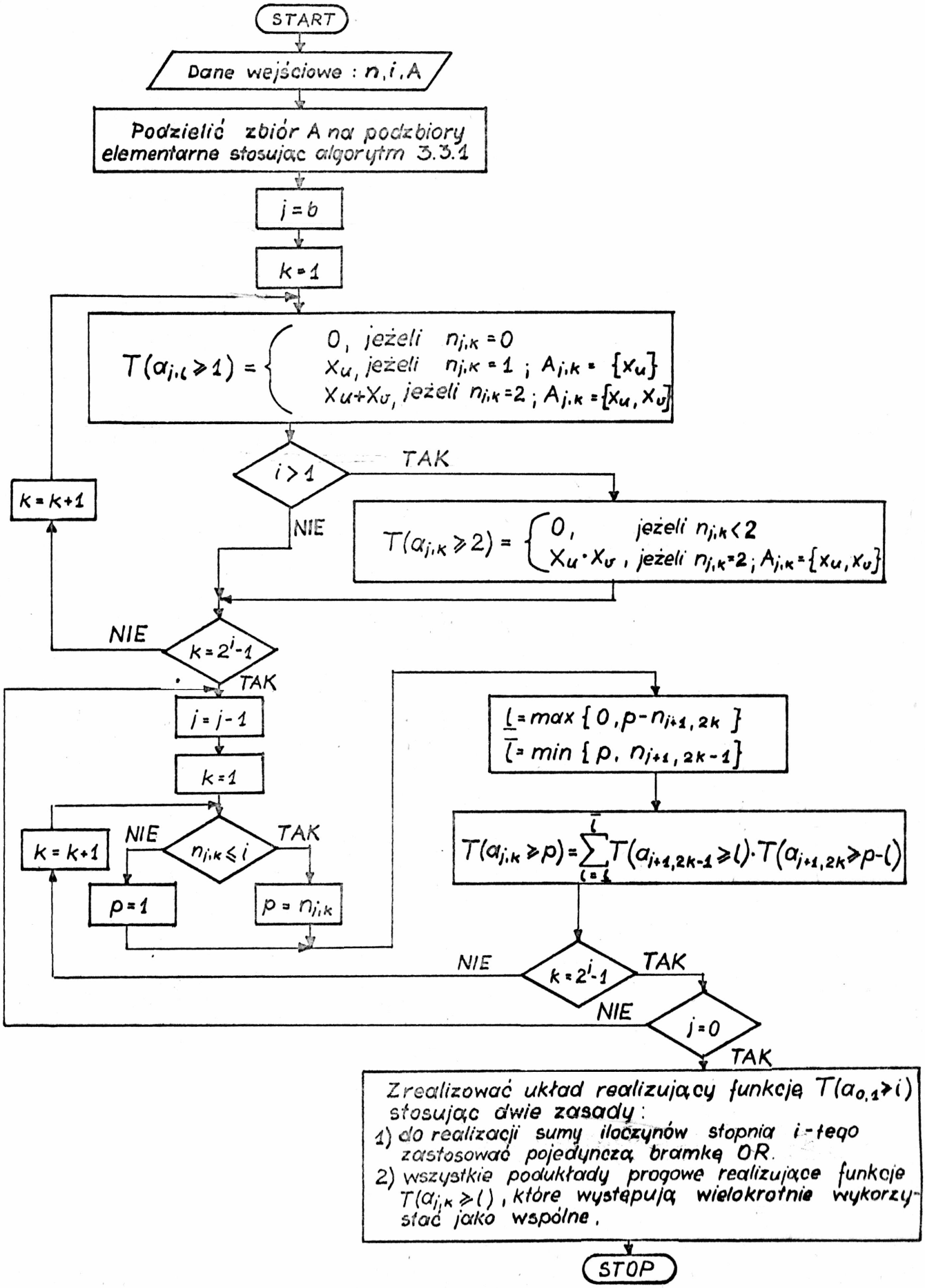
$$\underline{l} = \max\{0,\ i-n_{1,2}\} \,, \tag{3.3.2'}$$

$$\overline{l} = \min\{i,\ n_{1,1}\} \,. \tag{3.3.2''}$$

Podobnie można postąpić w przypadku funkcji $T(a_{1,1} \geq l)$ oraz $T(a_{1,2} \geq i-l)$, $l \in \{\underline{l}, \overline{l}\}$. Układ, który taką funkcję realizuje ma dla $n \geq 4$ strukturę wielopoziomową (tj. o liczbie poziomów bramek większej od 2). Podamy teraz dwa algorytmy: algorytm 3.3.1 - podziału zbioru A na podzbiory oraz algorytm 3.3.2 - projektowania wielopoziomowego układu progowego, które w maksymalnym stop-



Rys. 3.3.1. Schemat blokowy algorytmu 3.3.1



Rys. 3.3.2. Schemat blokowy algorytmu 3.3.2.

niu zapewniają stosowanie wspólnych podukładów i tym samym zmniejszenie złożoności układu progowego. Algorytmy te podano w formie schematów blokowych na rysunkach 3.3.1 i 3.3.2. Dane wejściowe tych algorytmów stanowią: n - liczność zbioru A ($n \geq 3$) oraz $A = \{x_1, x_2, \ldots, x_n\}$ - zbiór zmiennych wejściowych układu progowego - dla obydwu algorytmów, oraz dodatkowo dla algorytmu 3.3.2: i - próg układu.

Przykład 3.3.1

Stosując algorytm 3.3.2 zaprojektować wielopoziomowy układ progowy dla następujących danych:

$n=7$, $i=3$, $A = \{x_1, x_2, \ldots, x_7\}$.

1. Oto kolejne wyniki podziału zbioru A na podzbiory elementarne po zastosowaniu algorytmu 3.3.1:

i) $A_{1,1} = \{x_1, x_2, x_3\}$, $A_{1,2} = \{x_4, x_5, x_6, x_7\}$;

ii) $A_{2,1} = \{x_1\}$, $A_{2,2} = \{x_2, x_3\}$, $A_{2,3} = \{x_4, x_5\}$, $A_{2,4} = \{x_6, x_7\}$.

2. Oto zbiór funkcji progowych określonych na różnych zbiorach $A_{j,k}$ otrzymywanych w wyniku zastosowania algorytmu 3.3.2:

i) $T(a_{2,1} \geq 1) = x_1$,

$T(a_{2,2} \geq 1) = (x_2+x_3)$, $T(a_{2,2} \geq 2) = (x_2 \cdot x_3)$,

$T(a_{2,3} \geq 1) = (x_4+x_5)$, $T(a_{2,3} \geq 2) = (x_4 \cdot x_5)$,

$T(a_{2,4} \geq 1) = (x_6+x_7)$, $T(a_{2,4} \geq 2) = (x_6 \cdot x_7)$;

ii) $T(a_{1,1} \geq 1) = [x_1 + (x_2+x_3)]$,

$T(a_{1,1} \geq 2) = [x_1 \cdot (x_2+x_3) + (x_2 \cdot x_3)]$,

$T(a_{1,1} \geq 3) = [x_1 \cdot (x_2 \cdot x_3)]$,

$$T(a_{1,2} \geq 1) = [(x_4+x_5) + (x_6+x_7)],$$

$$T(a_{1,2} \geq 2) = [(x_4 \cdot x_5) + (x_4+x_5) \cdot (x_6+x_7) + (x_6 \cdot x_7)],$$

$$T(a_{1,2} \geq 3) = [(x_4+x_5) \cdot (x_6 \cdot x_7) + (x_4 \cdot x_5) \cdot (x_6+x_7)];$$

iii) $$T(a_{0,1} \geq 3) = T(a \geq 3) = [x_1 \cdot (x_2 \cdot x_3)] + [x_1 \cdot (x_2+x_3) +$$

$$+ (x_2 \cdot x_3)] \cdot [(x_4+x_5) + (x_6+x_7)] + [x_1 + (x_2+x_3)] \cdot [(x_4 \cdot x_5) +$$

$$+ (x_4+x_5) \cdot (x_6+x_7) + (x_6 \cdot x_7)] + [(x_4+x_5) \cdot (x_6 \cdot x_7) +$$

$$+ (x_4 \cdot x_5) \cdot (x_6+x_7)] .$$

W wyrażeniu na funkcję $T(a \geq 3)$ podkreślono wszystkie wykorzystane funktory logiczne; podwójną linią podkreślono te, które występują wielokrotnie, i w związku z tym odpowiadające im podukłady (w tym przypadku są to pojedyncze bramki OR i AND) stosuje się jako wspólne.

3.4. Wyznaczanie parametrów wielopoziomowych układów progowych zaprojektowanych wg algorytmu 3.3.2

Wzory, z których wyznacza się wyszczególnione poprzednio parametry układów progowych, stosuje się dla $n \geq 3$ oraz $1 \leq i \leq n$. Podano je osobno dla trzech przypadków: 1) i=1 lub i=n, 2) i=2, 3) $3 \leq i \leq n-1$.

1) i=1 lub i=n

$$Br(T_1^n) = n-1 , \qquad (3.4.1)$$

$$We(T_1^n) = 2 \cdot (n-1) , \qquad (3.4.2)$$

$$L(T_1^n) = \lceil \log_2 n \rceil , \qquad (3.4.3)$$

$$|\mathbb{T}(T_i^n)| = n+1 \,. \qquad (3.4.4)$$

W zależności od parametru i są to 2-wejściowe bramki OR (gdy i=1) lub AND (gdy i=n).

$$\mathbb{T}(T_1^n) = C_{1/n} \cup C_{0/n} \,,$$

$$\mathbb{T}(T_n^n) = C_{(n-1)/n} \cup C_{n/n} \,.$$

2) <u>i=2</u>

$$Br(T_2^n) = 2\cdot(n-1) \,, \qquad (3.4.5)$$

$$We(T_2^n) = 5\cdot(n-1) - 2 \,, \qquad (3.4.6)$$

$$L(T_2^n) = \lceil \log_2 n \rceil + 1 \,, \qquad (3.4.7)$$

$$|\mathbb{T}(T_2^n)| = 2\cdot n \,. \qquad (3.4.8)$$

W skład układu wchodzą:

n-1 2-wejściowych bramek AND,
n-2 2-wejściowych bramek OP,
1 n-1 -wejściowa bramka OR.

Zbiór $\mathbb{T}(T_n^2)$ stanowią słowa kodu 1/n oraz n słów kodu 2/n, w których na dwóch kolejnych pozycjach (pozycję n-tą i 1-szą uważa się za kolejne) cyklicznie występują jedynki.

3) <u>i ≥ 3</u>

W tym przypadku parametry układu wyznacza się ze wzorów rekurencyjnych.

A) <u>liczba bramek i połączeń</u>

Układ składa się z:

i) n-i+1 2-wejściowych bramek AND realizujących iloczyny i zmiennych,

ii) n-i+1 -wejściowej bramki OR realizującej sumę iloczynów i zmiennych,

iii pewnej liczby bramek realizujących funkcje $T(a_{1,1} \geq 1)$ oraz $T(a_{1,2} \geq 1)$, $1 \in \{\underline{1}, \overline{1}\}$, korzystamy tutaj z zapisu funkcji $T(a \geq i)$ w postaci (3.3.1), określonej wzorami:

$$Br' = Br\left(T_{\underline{1}'}^{n_{1,1}}\right) + \sum_{1=\underline{1}'+1}^{\overline{1}'} Br\,(n_{1,1},\ 1), \tag{3.4.9}$$

$$Br'' = Br\left(T_{i-\overline{1}'}^{n_{1,2}}\right) + \sum_{1=i-\overline{1}'+1}^{i-\underline{1}'} Br(n_{1,2},\ 1), \tag{3.4.10}$$

$$We' = We\left(T_{\underline{1}'}^{n_{1,1}}\right) + \sum_{1=\underline{1}'+1}^{\overline{1}'} We(n_{1,1},\ 1), \tag{3.4.11}$$

$$We'' = We\left(T_{i-\overline{1}'}^{n_{1,2}}\right) + \sum_{1=i-\overline{1}'+1}^{i-\underline{1}'} We(n_{1,2},\ 1). \tag{3.4.12}$$

gdzie:

$$\underline{1}' = \max\left\{1,\ i-n_{1,2}\right\}, \tag{3.4.13'}$$

$$\overline{1}' = \min\left\{i-1,\ n_{1,1}\right\}. \tag{3.4.13''}$$

$Br(n_{1,1},\ 1)$ jest liczbą bramek AND realizujących iloczyny 1 zmiennych oraz bramek OR realizujących sumy iloczynów 1 zmiennych, w $n_{1,1}$-wejściowym układzie progowym o progu 1; $We(n_{1,1},\ 1)$ oznacza łączną liczbę wejść tych bramek.
Podobne znaczenie mają: $Br(n_{1,2},\ 1)$ i $We(n_{1,2},\ 1)$.

Sposób wyprowadzenia tych wzorów omówimy na przykładzie układów realizujących funkcje $T(a_{1,1} \geq 1)$, $1 \in \{\underline{1}, \overline{1}\}$.
Układy te z założenia wykorzystują wspólne podukłady. Zbiór bramek wchodzących w skład tych układów można podzielić na podzbiory w zależności od krotności iloczynu zmiennych występujących na ich wyjściu. Liczbę bramek, na których wyjściu występują iloczyny 1 zmiennych, $1 \leq 1 \leq \underline{1}$, określa pierwsza część wzoru (3.4.9). Każdy kolejny podukład wnosi do tego bilansu tylko liczbę bramek, na których wyjściu występuje iloczyn liczby zmiennych określonej progiem tego podukładu, gdyż wszystkie bramki o liczbie zmiennych mniejszej od progu są wspólne z podukładami o niższych progach

niż próg danego układu. Z analogicznych przesłanek wynika wzór na łączną liczbę wejść tych bramek - (3.4.11), jak też wzory (3.4.10) i (3.4.12), z których wyznacza się parametry układów realizujących funkcje $T(a_{1,2} \geq i\text{-}1)$, $1 \in \{\underline{1}, \overline{1}\}$ ($\underline{1}$, $\overline{1}$ wyznacza się ze wzorów (3.3.2'), (3.3.2'')).

Sposób wyznaczania parametrów Br' i We' objaśnimy wyznaczając je dla układu progowego o parametrach: n=7, i=3, którego funkcję wygenerowano w przykł. 3.3.1 wg algorytmu 3.3.2.

$$n_{1,1} = 3,\ n_{1,2} = 4,\ 1' = 1,\ 1'' = 2;$$

$$Br' = Br\left(T_1^3\right) + Br(3,2);$$

$$T(a_{1,1} \geq 3) = x_1 + (x_2 + x_3) \quad \text{i stąd} \quad Br\ T_1^3 = 2.$$

3-wejściowy układ progowy o progu 2 opisuje funkcja

$$T(a_{1,1} \geq 2) = x_1 \cdot (x_2 + x_3) + (x_1 \cdot x_2) .$$

Funktory AND realizujące iloczyny 2 zmiennych podkreślono jedną kreską, a funktor OR realizujący sumę iloczynów 2 zmiennych podkreślono dwoma kreskami. Bramka OR realizujące funkcję x_2+x_3 jest wspólna i została uwzględniona przez parametr $Br\left(T_1^3\right)$. Zatem $Br(3,2) = 3$. Podobnie wyznacza się liczbę wejść bramek:

$$We' = We\left(T_1^3\right) + We(3,2) = 4+6 = 10 .$$

Wzory (3.4.9) - (3.4.12) można nieco uprościć, jeżeli odrębnie wyznaczy się parametry bramek AND i OR.

$$Br_{AND}(p,1) = p - 1+1, \quad 2 \leq 1 \leq p , \tag{3.4.14}$$

$$We_{AND}(p,1) = 2 \cdot (p - 1+1), \tag{3.4.15}$$

$$Br_{OR}(p,1) = \begin{cases} 1, & \text{dla } \left\lceil \frac{p}{2} \right\rceil \leq 1 \leq p, \\ 1+Br_{OR}\left(\left\lfloor \frac{p}{2} \right\rfloor, 1\right) + Br_{OR}\left(\left\lceil \frac{p}{2} \right\rceil, 1\right) & \text{dla } 1 \leq 1 < \left\lceil \frac{p}{2} \right\rceil ; \end{cases} \tag{3.4.16}$$

$$We_{OR}(p,1) = \begin{cases} p-1+1, & \text{dla } \lceil \frac{p}{2} \rceil \leq 1 \leq p, \\ p+1 + We_{OR}\left(\lfloor \frac{p}{2} \rfloor, 1\right) + We_{OR}\left(\lceil \frac{p}{2} \rceil, 1\right) & \text{dla } 1 \leq 1 < \lceil \frac{p}{2} \rceil . \end{cases} \quad (3.4.17)$$

Ostatecznie więc:

$$Br\left(T(a \geq i)\right) = (n-i+2) + Br\left(T(a_{1,1} \geq \underline{1})\right) + Br\left(T(a_{1,2} \geq i-\overline{1})\right) + \sum_{1=\underline{1}+1}^{\overline{1}} Br\left(\lfloor \frac{n}{2} \rfloor, 1\right) + \sum_{1=i-\overline{1}+1}^{i-\underline{1}} Br\left(\lceil \frac{n}{2} \rceil, 1\right), \quad (3.4.18)$$

$$We\left(T(a \geq i)\right) = 3 \cdot (n-i+1) + We\left(T(a_{1,1} \geq \underline{1})\right) + We\left(T(a_{1,2} \geq i-\overline{1}\right) + \sum_{1=\underline{1}+1}^{\overline{1}} We\left(\lfloor \frac{n}{2} \rfloor, 1\right) + \sum_{1=i-\overline{1}+1}^{i-\underline{1}} We\left(\lceil \frac{n}{2} \rceil, 1\right). \quad (3.4.19)$$

B) <u>liczba poziomów</u>

$$L\left(T(a \geq i)\right) = 2+L\left(T\left(a_{1,2} \geq \lceil \frac{n}{2} \rceil\right)\right). \quad (3.4.20)$$

C) <u>liczba testów</u>

$$\left|\mathbb{T}\left(T(a \geq i)\right)\right| = 2 \cdot n+c \ . \quad (3.4.21)$$

Zbiór $\mathbb{T}$ stanowią:

1) po n słów kodów i/n oraz (i-1)/n, w których na i (lub i-1) kolejnych pozycjach cyklicznie występują jedynki,
2) c dodatkowych słów kodu i/n, których konieczność stosowania i sposób wyznaczania omówiono poniżej.

Rozważmy funkcję

$$T(a_{j,k} \geq 1) = \sum_{p=\underline{p}}^{\overline{p}} T(a_{j+1,\,2k-1} \geq p) \cdot T(a_{j+1,2k} \geq 1-p), \ j \geq 1 \ . \quad (3.4.22)$$

Jeżeli spełnione są warunki:

i) $2 \leq 1 < i$,

ii) $\bar{p} - \underline{p}+1 \geq 3$,

to każda bramka AND realizująca funkcję

$$T(a_{j+1,2k-1} \geq p) \cdot T(a_{j+1,2k} \geq 1-p),\ \underline{p} < p < \bar{p}$$

wymaga dodatkowego testu dla uszkodzeń typu s/0. Można wykazać, że $c > 0$ wtedy gdy równocześnie spełnione są trzy warunki:
a) $i \geq 3$, b) $n \geq 7$, c) $n-i \geq 2$.
Każdy taki test powinien spełniać warunki:

$$1)\quad \Big(T(a_{j+1,2k-1} \geq p) \cdot T(a_{j+1,2k} \geq 1-p) = 1\Big) \wedge$$
$$\wedge \Big(\forall p' \in \{\underline{p}, \bar{p}\},\ p' \neq p\Big)\Big(T(a_{j+1,2k-1} \geq p') \cdot$$
$$\cdot T(a_{j+1,2k-1} \geq 1-p') = 0\Big), \qquad (3.4.23)$$

$$2)\quad T(a \geq i) = 1. \qquad (3.4.24)$$

Przypomnijmy, że $A_{j,k} = \{A_{j+1,2k-1} \cup A_{j+1,2k}\} = \{x_{s+1}, x_{s+2}, \ldots, x_{s+n_{j,k}}\}$, gdzie $s = \sum_{r=1}^{k-1} n_{j,r}$. Niech $u = \left\lfloor \frac{n_{j,k}}{2} \right\rfloor$.

$$3)\quad \Big(x_m \in A_{j,k}\Big) \Rightarrow \Big(x_m = 1 \Leftrightarrow m \in \{\overline{s+u-p+1,\ s+u+1-p}\}\Big). \qquad (3.4.25)$$

Jakkolwiek w ogólnym przypadku każdy dodatkowy test może być słowem kodu r/n, $r \in \{\overline{2,i}\}$, to jednak ze względu na wykorzystanie tych układów w STC kodów stałowagowych, wprowadzamy dodatkowy ostatni warunek:

4) każdy dodatkowy test jest słowem kodu i/n.

Warto nadmienić w tym miejscu, że liczba dodatkowych testów c może być mniejsza od liczby bramek AND wymagających dodatkowych testów. Dzieje się tak, jeżeli zaistnieje możliwość dobrania jednego testu spełniającego warunki 1 - 3 dla więcej niż jednej bramki równocześnie.
Ze względu na nadmierną złożoność pomijamy formalny dowód wystarczalności tak określonego zbioru testów. Zbiór ten wynika przez domniemanie z regularnej struktury układu.

Zbiory dodatkowych testów dla wybranych układów progowych zamieszczono w tabeli 3.4.1.

3.5. Porównanie układów progowych zaprojektowanych różnymi metodami

Prametry wielopoziomowych układów progowych zaprojektowanych wg algorytmu 3.3.2 zostaną porównane z parametrami podobnych układów zaprojektowanych wg metody z [59]. Parametry układów z [59] realizujących funkcję $T(a \geq i)$ wyznacza się z zależności:

1) $i=1$

$$Br\left(T_1^n\right) = n-1 , \tag{3.5.1}$$

$$We\left(T_1^n\right) = 2 \cdot (n-1), \tag{3.5.2}$$

$$L\left(T_1^n\right) = n-1, \tag{3.5.3}$$

$$\left|\mathbb{T}\left(T_1^n\right)\right| = n+1 . \tag{3.5.4}$$

2) $i \geq 2$

$$Br\left(T_i^n\right) = (2 \cdot i-1) \cdot (n-i) + i-1, \tag{3.5.5}$$

$$We\left(T_i^n\right) = 2 \cdot \left((2 \cdot i-1) \cdot (n-i) + i-1\right), \tag{3.5.6}$$

$$L\left(T_i^n\right) = 2 \cdot n-3, \tag{3.5.7}$$

$$\left|\mathbb{T}\left(T_i^n\right)\right| = 2 \cdot n . \tag{3.5.8}$$

Z porównania zależności (3.4.1) - (3.4.4) i (3.5.1) - (3.5.4) wynika, że dla $i=1$ i $n \geq 4$ układy z [59] wymagają większej liczby poziomów, podczas gdy pozostałe parametry są takie same. Z porównania zależności (3.4.5) - (3.4.8) i (3.5.5) - (3.5.8) wynika, że dla $i=2$ i $n \geq 4$ układy z [59] wymagają większej liczby bramek, połączeń i poziomów, przy tej samej liczbie testów.

Parametry wybranych układów takich, że $i \geq 3$, otrzymanych wg algorytmu 3.3.2 oznaczono gwiazdką " $*$ " i metody z [59] zamieszczono w tabeli 3.5.1.

Tabela 3.4.1. Zbiory dodatkowych testów dla wybranych układów progowych

i/n	3/7	3/8	4/8	3/9	4/9	3/10
Testy dodatkowe	$x_1x_5x_6$	$x_2x_3x_6$ $x_2x_6x_7$	$x_2x_3x_6x_7$	$x_2x_3x_6$ $x_2x_6x_7$	$x_2x_6x_7x_8$ $x_2x_3x_6x_7$	$x_3x_7x_8$ $x_2x_3x_6$
i/n	3/11	3/12	4/10	5/10	4/11	5/11
Testy dodatkowe	$x_4x_8x_9$ $x_2x_3x_6$	$x_5x_9x_{10}$ $x_3x_4x_7$	$x_4x_7x_8x_9$ $x_2x_3x_4x_6$ $x_2x_3x_7x_8$	$x_2x_3x_4x_7x_8$ $x_2x_3x_7x_8x_9$	$x_2x_3x_8x_9$ $x_2x_3x_4x_8$ $x_1x_7x_8x_9$ $x_1x_8x_9x_{10}$	$x_1x_7x_8x_9x_{10}$ $x_2x_3x_4x_8x_9$ $x_2x_3x_7x_8x_9$ $x_2x_3x_8x_9x_{10}$

Tabela 3.5.1. Zestawienie paramtrów wybranych układów/progowych zaprojektowanych wg algorytmu 3.3.2 (*) i wg metody z [59]

	3/7		3/8		3/9		3/10		4/8		4/9		4/10		4/11		5/10		5/11	
	*	59	*	[59]	*	[59]	*	[59]	*	[59]	*	[59]	*	[59]	*	[59]	*	[59]	*	[59]
Br	18	22	21	27	25	32	29	37	26	31	31	38	36	45	41	52	41	49	47	58
We	40	44	48	54	57	64	66	74	57	62	69	76	81	90	93	104	90	98	105	116
L	5	11	5	13	7	15	7	17	5	13	7	15	7	17	7	19	7	17	7	19
|T|	15	14	18	16	20	18	22	20	17	16	20	18	22	20	26	22	22	20	26	22

Z porównania parametrów wielopoziomowych układów zaprojektowanych dwoma różnymi metodami, a zamieszczonych w tabeli 3.5.1 dla wybranych wartości i oraz n, wynika, że układy zaprojektowane wg algorytmu 3.3.2 wymagają:

- mniejszej liczby bramek (rzędu kilkunastu procent),
- minimalnie mniejszej łącznej liczby wejść bramek,
- wnoszą znacznie mniejsze opóźnienie (mierzone liczbą poziomów bramek L),
- minimalnie większego zbioru testów,

niż analogiczne układy zaprojektowane wg metody podanej w [59].

Natomiast w porównaniu z układami progowymi dwupoziomowymi układy wielopoziomowe zaprojektowane wg algorytmu 3.3.2 na ogół wymagają znacznie mniejszej liczby bramek, połączeń i testów niż układy dwupoziomowe. Odbywa się to kosztem nieco większego opóźnienia, które wnoszą układy wielopoziomowe.

Na zakończenie tego podrozdziału warto nadmienić, że przedstawiony tutaj algorytm 3.3.2 można w prosty sposób uogólnić na układy realizujące równocześnie wszystkie n funkcji progowych $T(a \geq i)$, $i \in \{\overline{1,n}\}$. Zagadnienie to pomijamy jako, że ma ono raczej marginalne znaczenie dla głównego przedmiotu badań niniejszej pracy.

4. METODY PROJEKTOWANIA SAMOTESTOWALNYCH UKŁADÓW KONTROLNYCH KODÓW STAŁOWAGOWYCH m/n

W niniejszym rozdziale przedstawiono metody projektowania samotestowalnych układów kontrolnych (STC) następujących klas kodów m/n:

i) 2/n, $n \geq 5$ - podrozdz. 4.2,
ii) m/n, $m \geq 3$, $(2m+1) \leq n \leq 4m$ - podrozdz. 4.3,
iii) 1/n, $n \geq 7$ - podrozdz. 4.4,
iv) kody (n-m)/n dualne do w.w. klas kodów (tzn. takie, że m i n spełniają warunki wyszczególnione dla tych klas kodów) - podrozdz. 4.5.

Przyjęcie podziału kodów m/n na klasy i) - iv) zostało podyktowane pewnymi szczególnymi własnościami kodów z poszczególnych klas, pozwalającymi podać względnie proste metody projektowania STC kodów z danej klasy.

Nie uwzględniono następujących klas kodów m/n:

a) 1/n i (n-1)/n, $3 \leq n \leq 6$,
b) m/2m,
c) m/n i (n-m)/n, $m \geq 3$, $n > 4 \cdot m$,

dla których nie udało się opracować metod projektowania dających układy o mniejszej złożoności niż układy znane z literatury.

4.1. Struktura samotestowalnych układów kontrolnych (STC) kodów m/n

Zakładamy, że STC których metody projektowania podajemy w tym rozdziale mają strukturę szeregową przedstawioną na rys. 2.5.1. Strukturę taką mają również układy zaprojektowane wg metod z [3, 35, 48].

Działanie układów z [3] polega na kolejnych translacjach kodów stałowagowych: $m/n \rightarrow 1/\binom{n}{m} \rightarrow k/2 \cdot k \rightarrow 1/2$. Tym samym STC kodów $k/2 \cdot k$, $k \geq 2$ składają się z jednego bloku, kodów 1/n - z dwóch, a pozostałych kodów - z trzech bloków. STC kodów m/n, $m \geq 2$ zaprojektowane wg metody z [35] składają się z trzech bloków wykonujących kolejne translacje: $m/n \rightarrow 1/Z \rightarrow k/2 \cdot k \rightarrow 1/2$ (jeżeli $2 \cdot m+1 \leq n \leq 4 \cdot m$ to $4 \leq Z \leq 6$ i $k=2$; jeżeli $n > 4 \cdot m$ to $Z = 6$ i $k \geq 3$. Dla kodów m/n, $n > 4 \cdot m$ w [35] podano również inne rozwiązanie STC o bardziej skomplikowanej "drzewiastej" strukturze

połączeń bloków.
W [44, 48] podano metody projektowania STC kodów $m/2 \cdot m$ o bardzo zwartej i regularnej strukturze bez użycia bloków pośrednich. Próby zastosowania takiego rozwiązania dla kodów m/n rozważanych w tym rozdziale nie powiodły się, w związku z czym dla wszystkich wyróżnionych klas kodów STC ma strukturę z rys. 2.5.1.
Dane dotyczące struktury STC (o minimalnej złożoności) rozważanych w tym rozdziale zamieszczono w tabeli 4.1.1.

Tabela 4.1.1

Kod	Liczba bloków t	Struktura STC
1/n	$\geqslant 3$	$1/n \rightarrow 2/n_1 \rightarrow \ldots \rightarrow 2/4 \rightarrow 1/2$
	3	$1/n \rightarrow m_1/n_1 \rightarrow 2/4 \rightarrow 1/2,\ m_1 \geqslant 3,\ (2 \cdot m_1+1) \leqslant n_1 \leqslant 4 \cdot m_1$
2/n	2	$2/n \rightarrow 2/4 \rightarrow 1/2$ dla $5 \leqslant n \leqslant 8$
	3	$2/n \rightarrow 2/n_1 \rightarrow 2/4 \rightarrow 1/2,\ 5 \leqslant n_1 \leqslant 8$, dla $9 \leqslant n \leqslant 64$
	$\geqslant 3$	$2/n \rightarrow 2/n_1 \rightarrow \ldots \rightarrow 2/4 \rightarrow 1/2$, dla $n \geqslant 65$
m/n, m 3	2	$m/n \rightarrow 2/4 \rightarrow 1/2$ dla $(2 \cdot m+1) \leqslant n \leqslant 4 \cdot m$

Można zauważyć, że w układach tych, układem $|H^t$ zawsze jest STC kodu 2/4. Układ ten, o strukturze podanej w [3], opisany jest np. funkcjami:

$$H_1^t = \left(H_1^{t-1} + H_3^{t-1}\right) \cdot \left(H_2^{t-1} + H_3^{t-1}\right),$$

$$H_2^t = \left(H_1^{t-1} \cdot H_3^{t-1}\right) + \left(H_2^{t-1} \cdot H_3^{t-1}\right).$$

W związku z powyższym, głównym przedmiotem badań w dalszej części rozdz. 4 są metody projektowania układów $|H^i$ $(i \in \{\overline{1,t-1}\})$ oraz metody ich łączenia w taki sposób, aby cały układ $|H$ był STC kodu m/n (własności, którymi powinny charakteryzować się bloki jednoznacznie określa tw. 2.5.3).

4.2. Kody 2/n, n ≥ 5

STC kodów 2/n mają strukturę jak na rys. 2.5.1. Każdy z układów $|H^i$, $i \in \{\overline{1,t-1}\}$ dokonuje translacji kodu $2/n_{i-1}$ (przyjmujemy, że $n=n_0$) w kod $2/n_i$ taki, że $n_i < n_{i-1}$.

4.2.1. Ogólna postać funkcji realizowanej przez układ $|H^i$, $i \in \{\overline{1,t-1}\}$

Każdy z układów $|H^i$, $i \in \{\overline{1,t-1}\}$ realizuje funkcję wektorową $H^i = \{H_1^i, H_2^i, \ldots, H_{n_i}^i\}$, która dokonuje odwzorowania

$$H^i: \mathbb{Z}_{i-1} \longrightarrow \mathbb{Z}_i,$$

gdzie $\mathbb{Z}_{i-1} \subset C_{2/n_{i-1}}$, $\mathbb{Z}_i \subset C_{2/n_i}$.

Niech A oznacza zbiór zmiennych wejściowych układu $|H^i$. Jeżeli $i=1$ - $A = \{x_1, x_2, \ldots, x_n\}$; jeżeli $i > 1$ - $A = \{H_1^{i-1}, H_2^{i-1}, \ldots, H_{n_{i-1}}^{i-1}\}$. Tam gdzie to rozróżnienie nie jest istotne, dla zmiennej H_j^{i-1} wejściowej układu $|H^i$ stosujemy symbol H_j^{i-1}.

Zbiór A dzielimy na s_i, $2 \leq s_i < n_i$ niepustych podzbiorów rozłącznych $A^{(k)}$, $k \in \{\overline{1,s_i}\}$ takich, że

$$A^{(k)} = \left\{ x_j \in A \;\middle|\; \sum_{l=1}^{k-1} n^{(l)} < j \leq \sum_{l=1}^{k} n^{(l)} \right\}, \qquad (4.2.1)$$

przy czym $n^{(l)}$ oznacza moc zbioru $A^{(l)}$, a $n^{(0)} = 0$.

Niech $\{a_{i1}, a_{i2}\} = \Big\{ X \in C_{2/n_{i-1}} \;\Big|\; \big(\exists!\, H_j^{i-1} \in A^{(i1)} \,|\, H_j^{i-1}=1\big) \wedge \big(\exists!\, H_k^{i-1} \in A^{(i2)} \,|\, H_k^{i-1}=1\big),\ i1,i2 \in \{\overline{1,s_i}\},\ i1 \neq i2 \Big\}$.

Zbiór słów kodu $2/n_{i-1}$ dzielimy na dwa niepuste podzbiory rozłączne C', C'':

$$C' = \bigcup_{\substack{i1,i2 \in \overline{1,s_i} \\ i1 \neq i2}} \{a_{i1}, a_{i2}\},$$

$$C'' = C_{2/n_{i-1}} \setminus C' .$$

Zbiór C'' dzielimy na $r_i = n_i - s_i$ niepustych zbiorów rozłącznych E_j, $j \in \{\overline{s_i+1, n_i}\}$ (oczywiście $E_{n_j} = C''$, jeżeli $r_i = 1$).

Funkcje H_j^i, $j \in \{\overline{1, n_i}\}$, przyjmują postać:

$$H_j^i = \sum_{H_k^{i-1} \in A(j)} H_k^{i-1} \quad , \; j \in \{\overline{1, s_i}\} , \tag{4.2.2}$$

$$H_j^i = \sum m_k, \; j \in \{\overline{s_i+1, n_i}\} , \tag{4.2.3}$$

gdzie $m_k \leftrightarrow X_k$, $X_k \in E_j$.

Funkcje te mają własności:

$$\left(\forall X \in C'\right)\left(\left(\exists H_{j1}^i, H_{j2}^i, j1, j2 \in \{\overline{1, s_i}\} \; , \; j1 \neq j2 \mid H_{j1}^i(X) = H_{j2}^i(X) = 1\right) \wedge \right.$$
$$\left. \wedge \left(\forall l \in \{\overline{1, n_i}\} \setminus \{j1, j2\}\right)\left(H_l^i(X) = 0\right)\right) , \tag{4.2.4}$$

$$\left(\forall X \in C''\right)\left(\left(\exists ! \; H_j^i, \; j \in \{\overline{1, s_i}\} \mid H_j^i(X) = 1\right) \wedge \left(\exists ! \; H_k^i, \; k \in \{\overline{s_i+1, n_i}\} \mid \right.\right.$$
$$\left.\left. H_k^i(X) = 1\right)\right) . \tag{4.2.5}$$

Z własności tych wynika, że

$$\left(\forall X \in C_{2/n_{i-1}}\right)\left(H^i(X) \in C_{2/n_i}\right) , \tag{4.2.6}$$

tzn. układ $|H^i$ realizujący funkcje (4.2.2) i (4.2.3) każde słowo kodu $2/n_{i-1}$ odwzorowuje w pewne słowo kodu $2/n_i$.

O wykrywalności wejściowych słów niekodowych i uszkodzeń układu $|H^i$, jak również o złożoności układu $|H^i$ decydują: sposób podziału zbioru zmiennych wejściowych A, sposób podziału zbioru C'' na podzbiory E_j oraz ostateczna postać funkcji H^i umożliwiająca stosowanie wspólnych podukładów. Zagadnienia te rozwiązano w następnym podrozdziale.

4.2.2. Algorytm projektowania STC kodów 2/n, n ≥ 5

Algorytm 4.2.1 poprzedzimy krótkim omówieniem występujących w nim parametrów. Dla zapewnienia większej przejrzystości i uniknięcia

pomyłek ich znaczenie zilustrowano również na rys. 4.2.1.
Zbiór A zmiennych wejściowych układu $\mathbb{H}^i$, $i \in \{\overline{1,t-1}\}$, dzieli się na s_i zbiorów rozłącznych $A^{(j)}$, $j \in \{\overline{1,s_i}\}$ o licznościach:

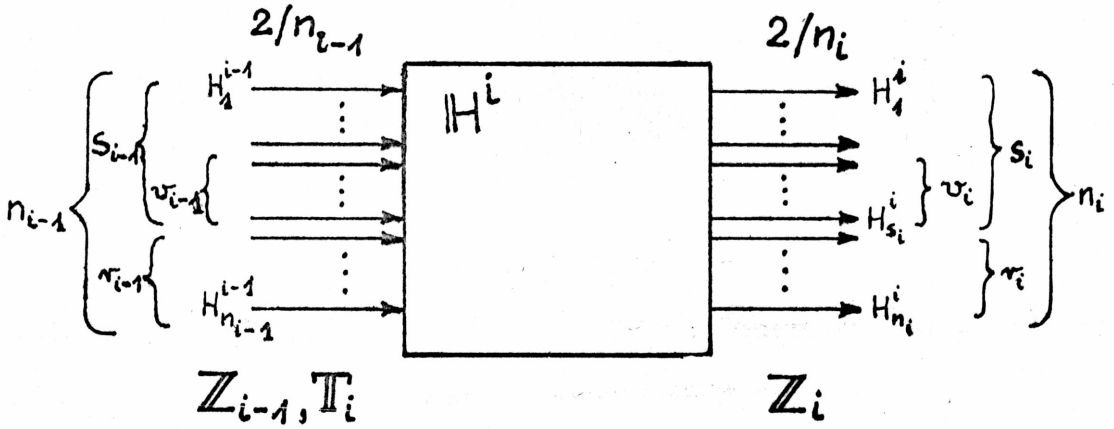
$$b_i = \left\lfloor \frac{n_{i-1}}{s_i} \right\rfloor \text{ - zbiory } A^{(j)} \text{ o indeksach } j \in \{\overline{1, s_i - v_i}\}$$

$$c_i = \left\lceil \frac{n_{i-2}}{s_i} \right\rceil \text{ - zbiory } A^{(j)} \text{ o indeksach } j \in \{\overline{s_i - v_i + 1, s_i}\},$$

gdzie v_i - liczba zbiorów $A^{(j)}$ o liczności c_i.
Spośród n_i funkcji H_j^i:
- każda z $s_i - v_i$ funkcji H_j^i, $j \in \{\overline{1, s_i - v_i}\}$, realizuje sumę logiczną b_i zmiennych wejściowych ze zbioru $A^{(j)}$,
- każda z v_i funkcji H_j^i, $j \in \{\overline{s_i - v_i + 1, s_i}\}$, realizuje sumę logiczną c_i zmiennych wejściowych ze zbioru $A^{(j)}$,
- każda z $r_i = \left\lceil \log_2 \frac{n_{i-1}}{s_i} \right\rceil$ funkcji H_j^i, $j \in \{\overline{s_i + 1, n_i}\}$, realizuje sumę iloczynów dwóch zmiennych wejściowych każdy.

$n_i = s_i + r_i$ jest równocześnie długością słowa kodu wyjść układu $\mathbb{H}^i$.

W zbiorze $\mathbb{Z}_{i-1}$ wyróżnia się zbiór testów $\mathbb{T}_i$ wystarczających do wykrycia wszystkich uszkodzeń ze zbioru F_p^i układu $\mathbb{H}^i$.



Rys. 4.2.1. Ilustracja oznaczeń stosowanych w odniesieniu do układu $\mathbb{H}^i$, $i \in \{\overline{1,t-1}\}$

Uwaga: dla i=1 oznaczeń indeksowanych na rys. 4.2.1 przez i-1 (z wyjątkiem $n_0 = n$ i $\mathbb{Z}_0 = \mathbb{X}$) nie stosuje się.

Wymienione poprzednio parametry spełniają nierówności:

$$1 \leq b_i \leq c_i \quad , \tag{4.2.7}$$

$$2 \leq c_i \leq \left\lceil \frac{n_{i-1}}{2} \right\rceil , \tag{4.2.8}$$

$$2 \leq v_i \leq s_i \quad , \tag{4.2.9}$$

$$2 \leq s_i \leq n_{i-1} \quad . \tag{4.2.10}$$

Algorytm 4.2.1 - projektowania STC kodów 2/n, $n \geq 5$

1. Przyjąć i=1, n_0=n oraz zbiór $\{x_1, x_2, \ldots, x_n\}$ jako zbiór zmiennych wejściowych A.
2. Przyjąć parametr s_i, $2 \leq s_i \leq n_{i-1}-2$, i wyznaczyć parametry r_i oraz n_i takie, aby były spełnione warunki:
   a/ W1 - W4 - jeżeli i=1,
   b/ W1 - W5 - jeżeli $i > 1$.

   W1) $r_i = \left\lceil \log_2 \frac{n_{i-1}}{s_i} \right\rceil$,

   W2) $(s_i + r_i = n_i) \wedge (n_i < n_{i-1})$,

   W3) $r_i \leq \left\lceil \frac{n_i}{2} \right\rceil$,

   W4) $(b_i \neq 2) \vee (c_i \neq 3)$,

   W5) $\left((r_i = 1) \wedge (b_i = 1) \wedge (r_{i-1} > 2)\right) \Rightarrow \left(r_{i-1} \leq v_i + 1\right)$.

3. Zbiór zmiennych wejściowych A podzielić na s_i podzbiorów rozłącznych $A^{(j)}$, $j \in \{\overline{1, s_i}\}$, o licznościach:

$$b_i = \left\lfloor \frac{n_{i-1}}{s_i} \right\rfloor , \; j \in \{\overline{1,\; s_i - v_i}\},$$

$$c_i = \left\lceil \frac{n_{i-1}}{s_i} \right\rceil , \; j \in \{\overline{s_i - v_i + 1,\; s_i}\},$$

gdzie $v_i = n_i - s_i \cdot b_i$,

z zachowaniem uporządkowania zmiennych wejściowych i zbiorów $A^{(j)}$ wg rosnących indeksów.

Uwaga: w zapisie funkcji H_j^i - kroki 4, 5 i 6 - y_u, $u \in \{\overline{1,n_{i-1}}\}$ oznacza: a) zmienną x_u - gdy i=1, b) zmienną H_u^{i-1} - gdy $i > 1$.

4. Funkcje translatora H_j^i, $j \in \{\overline{1,s_i}\}$, o postaci ogólnej

$$H_j^i = \sum_{y_u \in A^{(j)}} y_u \qquad (4.2.11)$$

zapisać w postaci uwzględniającej wielopoziomową strukturę realizujących je układów składających się z $n^{(j)}-1$ dwuwyjściowych bramek OR o strukturze typu "drzewo". Ogólne zasady generowania funkcji H_j^i w tej postaci podaje algorytm 3.3.2, który w tym przypadku upraszcza się do następującej procedury, którą wykonuje się kolejno dla wszystkich zbiorów $A^{(j)}$:

a) przyjmujemy $A_{0,1}^{(j)} = A^{(j)}$,

b) każdy ze zbiorów $A_{p,k}^{(j)}$, $k=2 \cdot l+1$, $l \in \{\overline{0,2^{p-1}-1}\}$, podzielić na dwa podzbiory $A_{p+1,k}^{(j)}$, $A_{p+1,k+1}^{(j)}$ o licznościach $n_{p+1,k}^{(j)} = \left\lfloor \frac{n_{p,k}^{(j)}}{2} \right\rfloor$, $n_{p+1,k+1}^{(j)} = \left\lceil \frac{n_{p,k}^{(j)}}{2} \right\rceil$,

c) zastosować dwuwejściowe bramki OR do realizacji funkcji $H_j^i(A_{p,k}^{(j)}) = H_j^i(A_{p+1,k}^{(j)}) + H_j^i(A_{p+1,k+1}^{(j)})$, $k=2 \cdot l+1$, $l \in \{\overline{0, 2^{p-1}-1}\}$, gdzie $H_j^i(A_{0,1}^{(j)}) = H_j^i$, a jeżeli $A_{p+1,k}^{(j)} = \emptyset$, to $H_j^i(A_{p+1,k}^{(j)}) = 0$; (to samo dotyczy zbioru $A_{p+1,k+1}^{(j)}$).

Kroki b) i c) należy wykonać r_i razy, tj. do momentu gdy wszystkie podzbiory $A_{p,k}^{(j)}$ będą co najwyżej jednoelementowe. r_i-ty podział dotyczy oczywiście tylko tych zbiorów $A_{r_i-1,k}^{(j)}$, $A_{r_i-1,k+1}^{(j)}$, które liczą dwa elementy.

5. Funkcje $H_{s_i+p}^i$, $p \in \{\overline{1,r_i}\}$, zapisać w postaci

$$H_{s_i+p}^i = \sum_{j=1}^{s_i} \sum_{\substack{k=2 \cdot l+1 \\ l \in \{0,\, 2^{p-1}\}}} \left(\sum_{y_u \in A_{p,k}^{(j)}} y_u \right) \cdot \left(\sum_{y_v \in A_{p,k+1}^{(j)}} y_v \right), \qquad (4.2.12)$$

gdzie $A^{(j)}_{p,k}$, $A^{(j)}_{p,k+1}$ - podzbiory zbioru $A^{(j)}$ otrzymane w wyniku p-tego kolejnego podziału zbioru $A^{(j)}$.

6. Jeżeli jest spełniony warunek:

W6) $\left((b_i = 2^l) \wedge (c_i = 2^l+1),\ l = 2,3,4,\ldots\right)$,

to dla każdego $j \in \{\overline{1, s_i - v_i}\}$ z funkcji $H^i_{n_i-1}$ przenieść do funkcji $H^i_{n_i}$ po jednym dowolnym iloczynie

$$\left(\bigvee_{y_u \in A^{(j)}_{r_i-1,k}} y_u\right) \cdot \left(\bigvee_{y_v \in A^{(j)}_{r_i-1,k+1}} y_v\right) .$$

7. Jeżeli $i > 1$ oraz $r_{i-1} > 1$, to w funkcjach H^i_l, $l \in \{\overline{1,n_i}\}$, zamienić miejscami indeksy każdej z w $\left(w = \left\lfloor \frac{r_{i-1}}{2} \right\rfloor\right)$ par zmiennych wejściowych $H^{i-1}_{s_{i-1}+j}$ oraz $H^{i-1}_{s_{i-1}+1-j}$, gdzie:

a) $j = 2 \cdot l+1$, $l \in \{\overline{0,\ w-1}\}$, gdy r_{i-1} jest parzyste,
b) $j = 2 \cdot l$, $l \in \{\overline{1,w}\}$, gdy r_{i-1} jest nieparzyste.

8. Jeżeli $r_i > 1$, to do realizacji funkcji $H^i_{s_i+j}$, $j \in \{\overline{1, r_i-1}\}$, wykorzystać jako wspólne bramki OR występujące w podukładach realizujących funkcje H^i_j, $j \in \{\overline{1,s_i}\}$.

9. Jeżeli $n_i > 4$, należy kolejno:
   a) zwiększyć parametr "i" o 1,
   b) przyjąć nowy parametr n_{i-1} oraz nowy zbiór zmiennych wejściowych H^{i-1}_j, $j \in \{\overline{1,n_{i-1}}\}$,
   c) przejść do kroku 2.

10. Zapisać funkcje układu $\mathbb{H}^t$, $t = i+1$, - STC kodu 2/4 w postaci:

$$H^t_1 = \left(H^i_1 + H^i_3\right) \cdot \left(H^i_2 + H^i_4\right),$$

$$H_2^t = H_1^i \cdot H_3^i + H_2^i \cdot H_4^i \,. \qquad (4.2.13)$$

W algorytmie 4.2.1 można wyróżnić trzy typy procedur:

1) dobór parametrów i wprowadzenie zmiennych wejściowych układu $\mathbb{H}^i$, $i \in \{\overline{1, t-1}\}$, - kroki 1, 2 i 9,
2) generowanie funkcji układu $\mathbb{H}^i$ - kroki 3 - 8,
3) generowanie funkcji układu $\mathbb{H}^t$ (STC kodu 2/4) - krok 10.

Kroki 1 i 10 wykonywane są zawsze dokładnie jeden raz. Kroki 2 - 9 wykonuje się t-1 razy - każdorazowo dla kolejnego układu $\mathbb{H}^i$, $i \in \{\overline{1,t-1}\}$.

W tabeli 4.2.1 podajemy zakres dopuszczalnych wartości t dla różnych parametrów n kodu 2/n (otrzymano je po uwzględnieniu ograniczeń wprowadzanych przez warunki W1 - W5).

Tabela 4.2.1

n	t
5 - 8	2 - (n-3)
9 - 64	3 - (n-3)

W celu wykazania poprawności algorytmu 4.2.1 należy:

- określić zawartość zbiorów $\mathbb{Z}_i$, $i \in \{\overline{1,t-1}\}$ -lemat 4.2.1,
- wykazać, że każdy układ $\mathbb{H}^i$, $i \in \{\overline{1,t-1}\}$, jest kodowo-rozłącznym translatorem przestrzeni kodowej $C_{2/n_{i-1}}$ w przestrzeń kodową C_{2/n_i} - lemat 4.2.2,
- określić zbiór testów $\mathbb{T}_i \subset \mathbb{Z}_{i-1}$, $i \in \{\overline{1,t-1}\}$, wystarczających do wykrycia uszkodzeń ze zbioru F_p^i - lemat 4.2.3,
- wykazać, że układ $\mathbb{H}$ jest układem kontrolnym kodu 2/n samotestowalnym dla uszkodzeń ze zbioru F_u - tw. 4.2.4,
- wykazać, że dla każdego kodu 2/n, $n \geq 5$, możliwe jest zaprojektowanie STC - tw. 4.2.5.

Lemat 4.2.1

Jeżeli układ $\mathbb{H}$ został zaprojektowany wg algorytmu 4.2.1, to dla każdego $i \in \{\overline{1,t-1}\}$ są spełnione warunki:

1) jeżeli $r_i = 1$ to:

a) $(v_i = s_i) \Rightarrow (\mathbb{Z}_i = C_{2/n_i})$,

b) $(v_i < s_i) \Rightarrow (\mathbb{Z}_i = C_{2/n_i-1} \times C_{0/1} \cup C_{0/s_i-v_i} \times C_{1/v_i} \times C_{1/r_i})$;

2) $(r_i > 1) \Rightarrow (\mathbb{Z}_i = C_{2/s_i} \times C_{0/r_i} \cup C_{1/s_i} \times C_{1/r_i})$.

Dowód

Rozważymy osobno dwa przypadki: I) i=1, II) i > 1.

I) i=1

Ponieważ $\mathbb{X} = C_{2/n}$, więc funkcja $H^1: C' \rightarrow C_{2/n_1-1} \times C_{0/1}$ jest surjekcją.

1) $r_1=1$

a) $(v_1=s_1) \Rightarrow (b_1=c_1=2) \Rightarrow \left(H^1_{n_1} = \sum_{u=2 \cdot l+1} (x_u \cdot x_{u+1})\right)$,

gdzie $l \in \{\overline{0, \frac{n}{2}}\}$) $\Rightarrow$ (funkcja $H^1: C'' \rightarrow C_{1/n_1-1} \times C_{1/1}$ jest surjekcją).

Zatem $\mathbb{Z}_1 = C_{2/n_1-1} \times C_{0/1} \cup C_{1/n_1-1} \times C_{1/1} = C_{2/n_1}$.

b) $(v_1 < s_1) \Rightarrow (b_1=1 \wedge c_1=2) \Rightarrow \left(H^1_{n_1} = \sum_{u=s_1-v_1+1+2 \cdot l} (x_u \cdot x_{u+1})\right)$,

gdzie $l \in \{\overline{0, v_1-1}\}) \Rightarrow$ (funkcja $H^1: C'' \rightarrow C_{0/s_1-v_1} \times C_{1/v_1} \times C_{1/1}$ jest surjekcją).

Zatem $\mathbb{Z}_1 = C_{2/n_1-1} \times C_{0/1} \cup C_{0/s_1-v_1} \times C_{1/v_1} \times C_{1/1}$.

2) $r_1 > 1$

$(r_1 > 1) \Rightarrow (b_1 \geq 2 \wedge c_1 \geq 3 \wedge [$z war. W4 alg. 4.2.1: $b_1 \neq 2 \vee c_1 \neq 3]) \Rightarrow (b_1 \geq 3 \wedge c_1 \geq 3)$.

Rozważymy dwa przypadki: i) $(b_1 \neq 2^l \vee c_1 \neq 2^l+1,\ l=2,3,4, \ldots)$,

ii) $(b_1=2^l \wedge c_1=2^l+1,\ l=2,3,4,\ \ldots)$.

Ad i) Ponieważ $|b_1-c_1| \leq 1$, więc w tym przypadku $\lceil \log_2 b_1 \rceil = \lceil \log_2 c_1 \rceil$. Zatem

$$(\forall j \in \{\overline{1,s_1}\})\ (\forall p \in \{\overline{1,r_1}\})(\exists A^{(j)}_{p,k} \neq \emptyset \wedge$$

$$\wedge \exists A^{(j)}_{p,k+1} \neq \emptyset,\ k=2\ l+1,\ l \in \{\overline{0,2^{p-1}-1}\}) \Rightarrow$$

$$\Rightarrow (\forall j \in \{\overline{1,s_1}\})\ (\forall p \in \{\overline{1,r_1}\})(\exists x_u \in A^{(j)}_{p,k} \wedge$$

$$\exists x_{u+1} \in A^{(j)}_{p,k+1}) \Rightarrow ((\forall j \in \{\overline{1,s_1}\})\ (\forall p \in \{\overline{1,r_1}\})$$

$$((H^1_j,\ H^1_{s_1+p}) \in \mathbb{Z}_1) \Rightarrow (\text{funkcja } H^1\colon C'' \longrightarrow C_{1/s_1} \times C_{1/r_1}$$

jest surjekcją).

Uwaga: $(H^1_j,\ H^1_{s_1+p})$ oznacza słowo kodu $2/n_1$, w którym na pozycjach o indeksach j oraz s_1+p występują jedynki.

Ad ii) W tym przypadku przed zmodyfikowaniem funkcji H^1 w kroku 6 alg. 4.2.1, funkcja $H^1\colon C'' \longrightarrow C_{0/s_1-v_1} \times C_{1/v_1} \times$ $\times C_{1/r_1}$ jest surjekcją.

Ponieważ $b^j \geq 4$ więc, dla każdego zbioru $A^{(j)}$, $j \in \{\overline{1,s_1-v_1}\}$, istnieją co najmniej dwie pary niepustych zbiorów

$$A^{(j)}_{r_1-1,k_1},\ A^{(j)}_{r_1-1,k_1+1} \text{ oraz } A^{(j)}_{r_1-1,k_2},\ A^{(j)}_{r_1-1,k_2+1},\ k_1 \neq k_2.$$

Przemiesienie z funkcji $H^1_{n_1-1}$ do funkcji $H^1_{n_1}$ po jednym iloczynie, np. $\left(\sum_{y_u \in A^{(j)}_{r_1-1,k_1}} y_u\right)\cdot\left(\sum_{y_v \in A^{(j)}_{r_1-1,k_1+1}} y_v\right)$, $j \in \{\overline{1,s_1-v_1}\}$,

spowoduje, że do zbioru $C_{0/s_1-v_1} \times C_{1/v_1} \times C_{1/r_1}$ oprócz słów ze zbioru $\mathbb{Z}_i$ należeć również będą słowa kodu $2/n_1$ o postaci $(H^1_j,\ H^1_{n_1})$, $j \in \{\overline{1,s_1-v_1}\}$. Tym samym po zmody-

fikowaniu funkcji H^1, funkcja H^1: $C'' \rightarrow C_{1/s_1} \times C_{1/r_1}$ jest surjekcją.
Zatem w obydwu przypadkach, tj. i) oraz ii),

$$\mathbb{Z}_1 = C_{2/s_1} \times C_{0/r_1} \cup C_{1/s_1} \times C_{1/r_1} .$$

II) $1 < i \leq t-1$

Zakładamy, że zbiór $\mathbb{Z}_{i-1}$ spełnia warunki określone w lemacie 4.2.1. Wykażemy, że zbiór $\mathbb{Z}_i$ określony surjekcją H^i: $\mathbb{Z}_{i-1} \rightarrow \mathbb{Z}_i$ również spełnia warunki określone w lemacie 4.2.1. Rozważymy dwa przypadki ogólne: A) $r_{i-1} = 1$, B) $r_{i-1} > 1$.

A) $r_{i-1} = 1$

$$\left(r_{i-1} = 1\right) \Rightarrow \left(\forall A^{(j)},\ j \in \{\overline{1, s_i}\}\right)\left(\exists H_1^{i-1} \mid \left(1 \in \{\overline{1, n_{i-1}-1}\}\right) \wedge \left(H_1^{i-1} \in A^{(j)}\right)\right),$$

a ponieważ $C_{2/n_{i-1}-1} \times C_{0/1} \subset \mathbb{Z}_{i-1}$, więc

$$\left(\forall j1, j2 \in \{\overline{1, s_i}\},\ j1 \neq j2\right)\left(\exists H_{11}^{i-1} \in A^{(j1)} \wedge \exists H_{12}^{i-1} \in A^{(j2)},\right.$$

$$\left.11, 12 \in \{\overline{1, n_{i-1}-1}\}\right) \Rightarrow \left(\forall j1, j2 \in \{\overline{1, s_i}\},\ j1 \neq j2\right)$$

$$\left((H_{11}^i, H_{12}^i) \in \mathbb{Z}_i\right) \Rightarrow \left(C_{2/s_i} \times C_{0/r_i} \subset \mathbb{Z}_i\right).$$

Dalsza część dowodu zarówno dla $r_i=1$ (przypadki a) i b)), jak i dla $r_i > 1$ jest podobna do dowodu przeprowadzonego dla i=1, w związku z czym pomijamy ją.

B) $r_{i-1} > 1$

Łatwo wykazać, że ponieważ parametr r_{i-1} spełnia warunek W3, tzn. $r_{i-1} \leq \left\lceil \frac{n_{i-1}}{2} \right\rceil$, modyfikacja funkcji H^i przewidziana do wykonania w kroku 7 alg. 4.2.1 zawsze jest możliwa do przeprowadzenia. Funkcję H^i modyfikuje się, ze względu na to, że iloczyny

$$\left(\sum_{H_k^{i-1} \in A_{p,k}^{(j)}} H_u^{i-1}\right) \cdot \left(\sum_{H_k^{i-1} \in A_{p,k+1}^{(j)}} H_v^{i-1}\right)$$

występujące w funkcjach $H^i_{s_i+j}$, $j \in \{\overline{1,r_i}\}$, które spełniają warunek

$$\left(\forall H_u^{i-1} \in A_{p,k}^{(j)},\ u \in \{\overline{s_{i-1}+1,\ n_{i-1}}\}\right) \wedge \left(\forall H_v^{i-1} \in A_{p,k+1}^{(j)},\right.$$

$$\left. v \in \{\overline{s_{i-1}+1,\ n_{i-1}}\}\right),$$

dla żadnego słowa kodowego ze zbioru $\mathbb{Z}_{i-1}$ nie przyjmują wartości 1, gdyż $C_{0/s_{i-1}} \times C_{1/s_{i-1}} \notin \mathbb{Z}_{i-1}$.

Modyfikacja funkcji H^i wykonywana w kroku 7 alg. 4.2.1 polega na tym, że $w = \left\lfloor \frac{r_{i-1}}{2} \right\rfloor$ spośród r_{i-1} zmiennych wejściowych o indeksach $H^{i-1}_{s_i+p}$, $p \in \{\overline{1,r_{i-1}}\}$, zamienia się miejscami z w spośród s_{i-1} zmiennych wejściowych H_p^{i-1}, $p \in \{\overline{1,s_{i-1}}\}$. Zamiana ta dotyczy co drugiej zmiennej, łącznie po w z każdej grupy. Po tej modyfikacji spełnione są dwa warunki:

$$\left(\forall H_l^i,\ l \in \{\overline{1,s_i}\}\right)\left(\exists H_u^{i-1} \mid (u \in \{\overline{1,s_{i-1}}\}) \wedge (H_l^i = H_u^{i-1} + H_l^{i*})\right), \tag{4.2.14}$$

$$\left(\forall A_{p,k}^{(j)*},\ A_{p,k+1}^{(j)*},\ p \in \{\overline{1,r_i}\},\ k=2\cdot l+1,\ l \in \{\overline{0,2^{p-1}-1}\}\right)\left(\exists H_u^{i-1} \mid\right.$$

$$\left.(u \in \{\overline{1,s_{i-1}}\}) \wedge (H_u^{i-1} \in \{A_{p,k}^{(j)*} \cup A_{p,k+1}^{(j)*}\})\right), \tag{4.2.15}$$

gdzie: H_l^{i*} oznacza sumę pozostałych zmiennych wejściowych, od których zależy funkcja H_l^i,
$A_{p,k}^{(j)*}$, $A_{p,k+1}^{(j)*}$ są zmodyfikowanymi podzbiorami $A_{p,k}^{(j)}$, $A_{p,k+1}^{(j)}$ otrzymanymi po zamianie miejscami zmiennych wejściowych w kroku 7 alg. 4.2.1.

Jedynie w przypadku, gdy $((r_i=1) \wedge (b_i=1) \wedge (r_{i-1}=v_{i+1}))$ warunek (4.2.14) przyjmuje postać:

$$\left(\forall H_l^i,\ l \in \{\overline{1,s_i}\},\ l \neq s_i - v_i\right)\left(\exists H_u^{i-1} \mid (u \in \{\overline{1,s_{i-1}}\}) \wedge (H_l^i = \right.$$

$$\left. = H_u^{i-1} + H_l^{i*})\right), \tag{4.2.16'}$$

$$H^{i}_{s_i - v_i} = H^{i-1}_{n_{i-1}-1} \cdot \qquad (4.2.16'')$$

Ponieważ z założenia $\mathbb{Z}_{i-1} = C_{2/s_{i-1}} \times C_{0/r_{i-1}} \cup C_{1/s_{i-1}} \times C_{1/r_{i-1}}$, więc w obydwu przypadkach, tzn. gdy funkcja H^i jest zadana albo w postaci (4.2.14) i (4.2.15), albo w postaci (4.2.16) i (4.2.15), funkcja H^i: $\mathbb{Z}_{i-1} \rightarrow \{C_{2/s_i} \times C_{0/r_i} \cup C_{1/s_i} \times C_{1/r_i}\}$ jest surjekcją.

Wykazaliśmy zatem, prawodziwość lematu 4.2.1 dla i=1, oraz, że jeżeli $2 \leqslant i \leqslant t-1$, to jeżeli zbiór $\mathbb{Z}_{i-1}$ spełnia warunki określone w tym lemacie, to spełnia je również zbiór $\mathbb{Z}_i$. Wynika stąd prawdziwość lematu dla każdego $i \in \{\overline{1, t-1}\}$.

Q.E.D.

Ponieważ niekiedy wygodniej jest operować zbiorem $\mathbb{Z}^{\emptyset}_i$ słów kodu $2/n_i$, które nie występują w czasie normalnej pracy układu $\mathbb{H}$, lemat 4.2.1 podajemy również w postaci alternatywnej:

Lemat 4.2.1

Jeżeli układ $\mathbb{H}$ został zaprojektowany wg algorytmu 4.2.1, to dla każdego $i \in \{\overline{1,t-1}\}$ są spełnione warunki:

1) **jeżeli $r_i = 1$, to:**

   a) $(v_i = s_i) \Rightarrow (\mathbb{Z}^{\emptyset}_i = \emptyset)$,

   b) $(v_i < s_i) \Rightarrow (\mathbb{Z}^{\emptyset}_i = C_{1/s_i - v_i} \times C_{0/v_i} \times C_{1/1})$,

2) $(r_i > 1) \Rightarrow (\mathbb{Z}^{\emptyset}_i = C_{0/s_i} \times C_{2/r_i})$.

Lemat 4.2.2

Każdy układ $\mathbb{H}^i$, $i \in \{\overline{1,t-1}\}$, zaprojektowany wg algorytmu 4.2.1 (kroki 2 - 8) spełnia warunki:

1) $(\forall X \in C_{2/n_{i-1}})\ (H^i(X) \in C_{2/n_i})$,

2) $(\forall X \notin C_{2/n_{i-1}})\ (H^i(X) \notin C_{2/n_i})$.

Dowód

Z lematu 4.2.1 wynika, że pierwszy z tych warunków jest spełnio-

ny. Wykażemy, że warunek drugi też jest spełniony. Rozważmy układ $|H^i$, $i \in \{\overline{1,t-1}\}$, zaprojektowany wg alg. 4.2.1.

Na mocy tw. 2.5.1 należy wykazać, że:

1) $\left(\forall x \in \bigcup_{j=0}^{1} C_{j/n_{i-1}}\right)\left(H^i(x) \in \overline{\mathbb{Z}}_i^-\right)$, gdzie $\overline{\mathbb{Z}}_i^- = \bigcup_{j=0}^{1} C_{j/n_i}$,

2) $\left(\forall x \in \bigcup_{j=3}^{n_{i-1}} C_{j/n_{i-1}}\right)\left(H^i(x) \in \overline{\mathbb{Z}}_i^+\right)$, gdzie $\overline{\mathbb{Z}}_i^+ = \bigcup_{j=3}^{n_1} C_{j/n_i}$.

Ponieważ układ $|H^i$ realizuje funkcję monotonicznie rosnącą, wystarczy zatem wykazać, że:

1) $\left(\forall x \in C_{1/n_{i-1}}\right)\left(H^i(x) \in \overline{\mathbb{Z}}_i^-\right)$,

2) $\left(\forall x \in C_{3/n_{i-1}}\right)\left(H^i(x) \in \overline{\mathbb{Z}}_i^+\right)$.

Ad 1) $\left(\forall x \in C_{1/n_{i-1}}\right)\left(H_j^i(x) = 0,\ j \in \{\overline{s_i+1,\ n_i}\}\right)$.

Ponieważ zbiór $A = \left\{H_{1,2}^{i-1}, H_2^{i-1}, \ldots, H_{n_{i-1}}^{i-1}\right\}$ został podzielony na podzbiory rozłączne $A^{(j)}$, $j \in \{\overline{1,s_i}\}$, więc

$$\left(\forall x \in C_{1/n_{i-1}}\right)\left(\exists!\ H_j^i(x) = 1,\ j \in \{\overline{1,s_i}\}\right).$$

Zatem $\left(\forall x \in C_{1/n_{i-1}}\right)\left(H^i(x) \in C_{1/n_i} \subset \mathbb{Z}_i\right)$.

Ad 2) Niech X - słowo kodu $3/n_{i-1}$. Przez u1, u2, u3, $(u1, u2, u3 \in \{\overline{1,n_i}\})$, oznaczamy indeksy tych pozycji słowa X, na których występują jedynki. Zakładamy, że $u1 < u2 < u3$.

W zależności od przynależności zmiennych x_{u1}, x_{u2}, x_{u3} do zbiorów $A^{(j)}$ wyróżniamy następujące przypadki:

a) $x_{u1}, x_{u2}, x_{u3} \in A^{(j)}$,

b) $x_{u1} \in A^{(j1)}$, $x_{u2}, x_{u3} \in A^{(j2)}$ (oczywiście $j1 < j2$),

c) $x_{u1} \in A^{(j1)}$, $x_{u2} \in A^{(j2)}$, $x_{u3} \in A^{(j3)}$ (oczywiście $j1 < j2 < j3$),

przy czym $j, j1, j2, j3 \in \{\overline{1,s_1}\}$.

Ad a) Z (4.2.11) wynika, że $\left(H^i_j(X) = 1\right)$ i $\left((\forall p \in \overline{\{1,s_1\}}\ ,\ p \neq j)\ H^i_p(X) = 0\right)$.

Oznaczmy przez p1, p2 $\left(p1, p2 \in \overline{\{1,r_1\}}\ ,\ p1 < p2\right)$ liczby porządkowe podziałów zbioru $A^{(j)}$, w wyniku których $\left(x_{u1} \in A^{(j)}_{p1,k1}\right)$ i $\left(x_{u2}, x_{u3} \in A^{(j)}_{p1,k1+1}\right)$ [albo: $\left(x_{u1}, x_{u2} \in A^{(j)}_{p1,k1}\right)$ i $\left(x_{u3} \in A^{(j)}_{p1,k1+1}\right)$], a następnie $\left(x_{u2} \in A^{(j)}_{p2,k2}\right)$ i $\left(x_{u3} \in A^{(j)}_{p2,k2+1}\right)$ [albo, odpowiednio: $\left(x_{u1} \in A^{(j)}_{p2,k2}\right)$ i $\left(x_{u2} \in A^{(j)}_{p2,k2+1}\right)$].

Z (4.2.11) wynika, że $H^i_{s_i+p1}\, X = H^i_{s_i+p2}\, X = 1$.

Ad b) Z (4.2.11) wynika, że

$H^i_{j1}(X) = H^i_{j2}(X) = 1$.

Niech p - liczba porządkowa podziału zbioru $A^{(j)}$, w wyniku którego $x_{u2} \in A^{(j)}_{p,k}$ i $x_{u3} \in A^{(j)}_{p,k+1}$.

Wówczas $H^i_{s_i+p}(X) = 1$.

Ad c) Z (4.2.11) wynika, że $H^i_{j1}(X) = H^i_{j2}(X) = H^i_{j3}(X) = 1$.

W każdym z rozważanych przypadków układ $|H^i$ odwzorowuje wejściowe słowo niekodowe $X \in C_{3/n_{i-1}}$ w wyjściowe słowo niekodowe ze zbioru C_{3/n_i}.

Q.E.D.

Klasy uszkodzeń występujących w układach $|H^i$, $i \in \overline{\{1,t-1\}}$

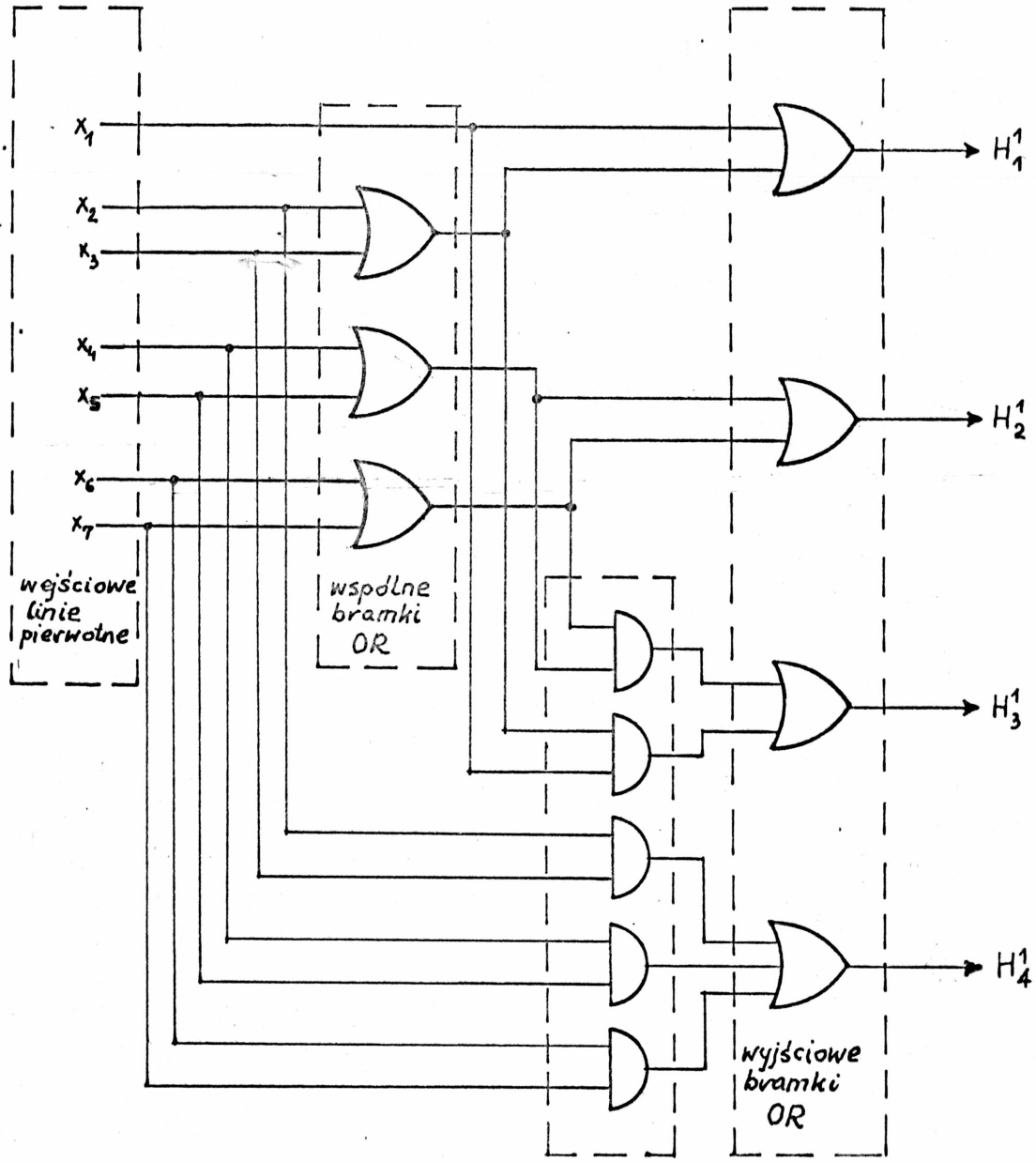
Każdy układ $|H^i$, $i \in \overline{\{1,t-1\}}$, zaprojektowany wg algorytmu 4.2.1 ma strukturę taką jak układ na rys. 4.2.2, opisany funkcjami:

$$H^1_1 = x_1 + (x_2+x_3)\ ,$$

$$H^1_2 = (x_4+x_5) + (x_6+x_7)\ ,$$

$$H^1_3 = x_1 \cdot (x_2+x_3) + (x_4+x_5) \cdot (x_6+x_7)\ ,$$

$$H^1_4 = x_2 \cdot x_3 + x_4 \cdot x_5 + x_6 \cdot x_7\ .$$



Rys. 4.2.2. Schemat ST kodowo rozłącznego translatora kodu 2/7 w kod 2/4 (układ H^1 w STC kodu 2/7)

W układzie takim wyróżniamy następujące klasy uszkodzeń pojedynczych:

1) uszkodzenia typu s/z, $z = \{0,1\}$, bramek OR występujących w podukładach realizujących funkcje H_j^i, $j \in \{\overline{1,s_i}\}$,
2) uszkodzenia typu s/0 wejściowych i wyjściowych linii bramek AND oraz wejściowych linii bramek OR wyjściowych, realizujących funkcje H_j^i, $j \in \{\overline{s_i+1, n_i}\}$,
3) uszkodzenia typu s/1 wejściowych linii bramek AND,

4) uszkodzenia typu s/z, z = {0,1} , wejściowych linii pierwotnych,

5) uszkodzenia typu s/1 wyjściowych linii bramek AND oraz wejściowych i wyjściowych linii bramek OR wyjściowych, realizujących funkcje H^i_j, $j \in \{\overline{s_i+1, n_i}\}$.

Ponieważ uszkodzenia z klas 4 i 5 są równoważne pewnym jednokierunkowym uszkodzeniom podwójnym z klas 1 - 3, więc na mocy własności W. 2.3.3 są one wykrywane przez testy, które wykrywają wszystkie uszkodzenia z klas 1 - 3.

Zbiór testów dla uszkodzeń układu $|H^i$, $i \in \{\overline{1,t-1}\}$

W zbiorze $C_{2/n_{i-1}}$ wyróżniamy dwa podzbiory rozłączne $\mathbb{T}^1_i$, $\mathbb{T}^2_i$, które spełniają warunki:

zbiór $\mathbb{T}^1_i$:

a) każda zmienna wejściowa co najmniej jeden raz przyjmuje wartość 1,

b) w każdym słowie $X \in \mathbb{T}^1_i$ wartość 1 przyjmują zmienne należące do różnych zbiorów $A^{(j)}$,

c) odwzorowanie H^i: $\mathbb{T}^1_i \longrightarrow C_{2/s_i} \times C_{0/r_i}$ jest surjekcją;

zbiór $\mathbb{T}^2_i$:

- każdy z $n_i - s_i$ iloczynów występujących w funkcjach H^i_j, $j \in \{\overline{s_i+1, n_i}\}$, co najmniej jeden raz przyjmuje wartość 1.

Zbiór $\mathbb{T}^2_i$, taki, że $|\mathbb{T}^2_i| = n_i - s_i$, tworzą np. słowa kodu $2/n_{i-1}$ o postaci ogólnej

$$(H^{i-1}_u, H^{i-1}_{u+1}), \text{ gdzie } H^{i-1}_u \in A^{(j)}_{p,k} \text{ i } H^{i-1}_{u+1} \in A^{(j)}_{p,k+1},$$

przy czym pary niepustych zbiorów $A^{(j)}_{p,k}$, $A^{(j)}_{p,k+1}$ wyczerpują wszystkie $n_i - s_i$ par podzbiorów zbioru $A^{(j)}$, otrzymanych w wyniku wszystkich r_i podziałów zbioru $A^{(j)}$.

Minimalne liczności tych zbiorów wynoszą:

$$|\mathbb{T}^1_i| = \max\left\{ \binom{s_i}{2}, \left\lceil \frac{n_i - 1}{2} \right\rceil \right\},$$

$$|\mathbb{T}^2_i| = n_i - s_i .$$

Lemat 4.2.3

Zbiór $\mathbb{T}_i = \mathbb{T}^1_i \cup \mathbb{T}^2_i$ wystarcza do wykrycia uszkodzeń ze

F_p^i układu $|H^i$ zaprojektowanego wg algorytmu 4.2.1 (kroki 1 - 8).

Dowód

Z przeprowadzonej analizy klas uszkodzeń występujących w układach $|H^i$, $i \in \overline{(1,t-1)}$, wynika, że wystarczy wykazać, że wszystkie uszkodzenia należące do klas 1 - 3 są wykrywane przez testy ze zbioru $\mathbb{T}_i$.

Wprowadzamy oznaczenia:

$$\{a_j, a_k\} = \left\{X \in C_{2/n_{i-1}} \mid (\exists! x_u \in A^{(j)}) \wedge (\exists! x_v \in A^{(k)}) \wedge (j \neq k) \wedge (x_u = x_v = 1)\right\},$$

$$\{a_j', a_j''\} = \left\{X \in C_{2/n_{i-1}} \mid (\exists x_u, x_v \in A^{(j)}) \wedge (u \neq v) \wedge (x_u = x_v = 1)\right\}.$$

1) Załóżmy, że uszkodzenie typu s/z należące do klasy 1 wystąpiło w połączeniu, które realizuje funkcję zależną, m. in. od zmiennej $x_u \in A^{(j)}$.
   a) Jeżeli z=0, to testem dla tego uszkodzenia jest słowo $X \in \mathbb{T}_i^1$ takie, że $X \in \{a_j, a_k\}$ i $X = (x_u, x_v)$.
   b) Jeżeli z=1, to testem dla tego uszkodzenia jest słowo $X \in \mathbb{T}_i^2$ takie, że $X \in \{a_k', a_k''\}$.
2) Każde uszkodzenie należące do klasy 2 jest testowane przez słowo $X \in \mathbb{T}_i^2$, dla którego bramka AND, której to uszkodzenie dotyczy, przyjmuje wartość 1,
3) Załóżmy, że wystąpiło uszkodzenie należące do klasy 3, i że w nieuszkodzonej wejściowej linii uszkodzonej bramki AND realizowana jest funkcja uzależniona od zmiennej $x_u \in A^{(j)}$. Testem dla tego uszkodzenia jest słowo $X \in \mathbb{T}_i^1$ takie, że $X \in \{a_j, a_k\}$ i $X = (x_u, x_v)$.

Q.E.D.

Wnioski z lematu 4.2.3

1) na mocy własności W. 2.3.3 zbiór $\mathbb{T}_i$ jest wystarczającym zbiorem testów również dla uszkodzeń ze zbioru F_u^i,
2) istnieje wiele zbiorów testów $\mathbb{T}_i$ takich, że $|\mathbb{T}_i| = \max\left\{\binom{s_i}{2}, \lceil n_{i-1}/2 \rceil\right\} + n_{i-1} - s_i$,
3) dla ustalonego n_{i-1} można wyznaczyć takie s_i, że zbiór $\mathbb{T}_i$ jest najmniej liczny,

4) ponieważ $\mathbb{T}_1 \subset C_{2/n}$, więc układ $\mathbb{H}^1$ zaprojektowany wg algorytmu 4.2.1 jest samotestowalny dla uszkodzeń ze zbioru F_p^1.

Twierdzenie 4.2.4

Układ $\mathbb{H}$ zaprojektowany wg algorytmu 4.2.1 jest układem kontrolnym kodu 2/n samotestowalnym dla uszkodzeń ze zbioru F_u.

Dowód

Zakładamy, że układ $\mathbb{H}$ zaprojektowano stosując algorytm 4.2.1 - zatem spełnione są wszystkie warunki i ograniczenia wyszczególnione w algorytmie 4.2.1. Ponieważ rozważany układ $\mathbb{H}$ jest bezinwersyjny, ma strukturę jak układ z rys. 2.5.1 i spełnia założenia 1 - 3 tw. 2.5.3, wystarczy wykazać, że są spełnione warunki 1 - 3 wyszczególnione w tw. 2.5.3.

Z wniosku 4 z lematu 4.2.3 wynika, że układ $\mathbb{H}^1$ jest samotestowalny dla uszkodzeń ze zbioru F_p^1. Z lematu 4.2.2 wynika, że układ $\mathbb{H}^1$ jest translatorem kodu 2/n w kod $2/n_1$. Warunek 1 tw. 2.5.3 jest więc spełniony.

Czy są spełnione warunki 2 i 3 tw. 2.5.3 sprawdzimy osobno dla dwóch przypadków: 1) $i \in \{\overline{1,t-1}\}$, 2) $i=t$.

1) $i \in \{\overline{1,t-1}\}$

Z lematu 4.2.2 i tw. 2.5.2 wynika, że warunek 2 jest spełniony dla każdego $i \in \{\overline{1,t-1}\}$. Z lematu 4.2.3 wynika, że zbiór $\mathbb{T}_i$, $i \in \{\overline{1,t-1}\}$, wystarcza do wykrycia wszystkich uszkodzeń ze zbioru F_p^i. Wystarczy zatem wykazać, że dla każdego $i \in \{\overline{1,t-1}\}$, $\mathbb{T}_i \subset \mathbb{Z}_{i-1}$.

Ze struktury zbioru $\mathbb{Z}_i$ określonej w lemacie 4.2.1 natychmiastowo wynika, że do zbioru $\mathbb{Z}_{i-1}$ należą słowa kodu $2/n_{i-1}$ spełniające warunki a, b, c nakładane na zbiór $\mathbb{T}_i^1$.

Ponieważ z lematu 4.2.1 wynika, że do zbioru $\mathbb{Z}_{i-1}^{\emptyset}$ nie należy żadne słowo kodu $2/n_{i-1}$ ani w postaci $(H_u^{i-1}, H_{u+1}^{i-1})$, $u \in \{\overline{1,n_{i-1}-1}\}$ (istotne gdy $r_{i-1} \geq 1$), ani w postaci $(H_j^{i-1}, H_{s_{i-1}+1}^{i-1})$, $j \in \{\overline{1,s_{i-1}}\}$, $l \in \{\overline{1,r_{i-1}}\}$ (istotne, gdy $r_{i-1} > 1$), wynika stąd, że $\mathbb{T}_i^2 \subset \mathbb{Z}_{i-1}$.

2) i=t

Układ $|H^t$ - STC kodu 2/4 - realizuje funkcje (4.2.13). Z lematu 4.2.2 i tw. 2.5.2 wynika, że $\overline{\mathbb{Z}}_{H_{t-1}} = \overline{\mathbb{Z}}^{-}_{t-1} \cup \cup \overline{\mathbb{Z}}^{+}_{t-1}$, a więc jest spełniony warunek 2. Zbadamy, czy zbiór $\mathbb{Z}_{t-1}$ zawiera wystarczający zbiór testów dla uszkodzeń układu $|H^t$ ze zbioru F_p^t.

Rozważymy dwa przypadki: a) $r_{t-1}=1$, b) $r_{t-1}=2$.

Ad a) $(r_{t-1}=1) \Rightarrow ((s_{t-1}=3) \wedge (v_{t-1}=2 \vee v_{t-1}=3))$.

Z lematu 4.2.1' wynika, że:

$(v_{t-1}=2) \Rightarrow (\overline{\mathbb{Z}}^{\emptyset}_{t-1} = \{(1001)\})$,

$(v_{t-1}=3) \Rightarrow (\overline{\mathbb{Z}}^{\emptyset}_{t-1} = \emptyset)$.

Ad b) $(r_{t-1}=2) \Rightarrow (s_{t-1}=2) \Rightarrow (\mathbb{Z}^{\emptyset}_{t-1} = \{(0011)\})$.

Wystarczającymi zbiorami testów dla uszkodzeń ze zbioru F_u^t są zbiory: w przypadku a - $\mathbb{T}_t = \{1100, 1010, 0101, 0011\}$, w przypadku b - $\mathbb{T}_t = \{1010, 1001, 0101, 0110\}$. Oczywiście $\mathbb{T}_t \subset \mathbb{Z}_{t-1}$.

Twierdzenie 4.2.5

Jeżeli układ $|H^{i-1}$, $i \in \{\overline{2,t-1}\}$ został zaprojektowany wg algorytmu 4.2.1, to dla każdego $n_{i-1} \geq 5$ możliwe jest zaprojektowanie układu ^{i} wg algorytmu 4.2.1 kroki 2 - 8 .

Dowód

Jeżeli układ $|H^{i-1}$ zaprojektowano wg algorytmu 4.2.1, to parametry układu $|H^{i-1}$: s_{i-1}, r_{i-1}, n_{i-1}, b_{i-1}, c_{i-1} spełniają warunki W1 - W5. Rozważmy układ $|H^i$, dla którego przyjęto parametr $s_i = \lceil n_{i-1}/2 \rceil$ (oczywiście $s_i \geq 3$), a pozostałe parametry przyjmują następujące wartości:

$$r_i=1 \left(r_i \leq \left\lceil \frac{n_i}{2} \right\rceil\right), n_i = \left\lceil \frac{n_{i-1}}{2} \right\rceil + 1 \quad (n_i < n_{i-1} \text{ i } n_i \geq 4),$$

$b_i \in \{\overline{1,2}\}$ i $c_i=2$.

Parametry układu $|H^i$ spełniają zatem warunki W1 - W4.

Przypuśćmy, że $r_i=1$, $b_i=1$ i $r_{i-1} > 2$. Ponieważ

$r_{i-1} \leq \left\lceil \frac{n_{i-1}}{2} \right\rceil$ (warunek W3), $s_i = \left\lceil \frac{n_{i-1}}{2} \right\rceil$ i $v_i \in \{\overline{s_i-1, s_i}\}$, więc jeżeli $v_i = s_i-1$, to $r_{i-1} = v_i+1$, i tym samym warunek W5 też jest spełniony.

Q.E.D.

Wniosek: ponieważ układ $|H^1$ nie wymaga spełnienia warunku W5, więc również dla każdego $n \geq 5$ algorytm 4.2.1 umożliwia zaprojektowanie takiego układu.
Zatem dla każdego kodu 2/n, $n \geq 5$, możliwe jest zaprojektowanie STC kodu 2/n wg algorytmu 4.2.1.

4.2.3. Wyznaczanie parametrów STC kodów 2/n

Ocenę złożoności układu $|H^i$, $i \in \{\overline{1,t-1}\}$, dokonuje się na podstawie zależności podanych w tabeli 4.2.2.

Tabela 4.2.2. Bilans liczby bramek i liczby wejść bramek układu $|H^i$

	Typ bramki		Łącznie
	OR	AND	
Liczba bramek Br_i	$n_{i-1}-s_i+r_i$	$n_{i-1}-s_i$	$2\cdot(n_{i-1}-s_i)+r_i$
Liczba wejść bramek We_i	$3\cdot(n_{i-1}-s_i)$	$2\cdot(n_{i-1}-s_i)$	$5\cdot(n_{i-1}-s_i)$

$n_{i-1}-s_i = \sum_{j=1}^{s_i} \left(n^{(j)}-1\right)$ bramek OR realizuje funkcje H^i_j, $j \in \{\overline{1,s_i}\}$, a pozostałe r_i bramek OR, to wyjściowe bramki OR w układach realizujących funkcje $H^i_{s_i+j}$, $j \in \{\overline{1,r_i}\}$.
Liczba poziomów $L_i = \left\lceil \log_2 \frac{n_{i-1}}{s_i} \right\rceil + 1$.

Wnioski:
1) złożoność układu $|H^i$, $i \in \{\overline{1,t-1}\}$, liniowo zależy od $(n_{i-1}-s_i)$,
2) im większe s_i tym lepsze są parametry Br_i, We_i, L_i układu $|H^i$.

Układ $|H^t$ - STC kodu 2/4 ma parametry: $Br_t=6$, $We_t=12$, $L_t=2$.
Parametry układu $|H$ wyznacza się z zależności:

$$Br = \sum_{i=1}^{t} Br_i, \quad We = \sum_{i=1}^{t} We_i, \quad L_i = \sum_{i=1}^{t} L_i,$$

$$|T| = \max\left\{\binom{s_1}{2}, \left\lceil\frac{n}{2}\right\rceil\right\} + n-s_1 \quad \text{(z lematu 4.2.2)}.$$

4.2.4. Zasady doboru optymalnej struktury STC

Algorytm 4.2.1 dopuszcza sporą dowolność, jeżeli chodzi o dobór parametrów układów $|H^i$ - s_i oraz n_i.
W związku z tym układy $|H$ tego samego kodu 2/n mogą się różnić nie tylko strukturą pojedynczego układu $|H^i$, lecz również liczbą bloków - t.
Np. dla kodu 2/10 podajemy kilka rozwiązań alternatywnych dopuszczalnych algorytmem 4.2.1:

$$2/10 \xrightarrow{(8,1)} 2/9 \xrightarrow{(7,1)} 2/8 \xrightarrow{(6,1)} \ldots \xrightarrow{(3,1)} 2/4 \longrightarrow 1/2$$

$$2/10 \xrightarrow{(6,1)} 2/7 \xrightarrow{(2,2)} 2/4 \longrightarrow 1/2$$

$$2/10 \xrightarrow{\substack{(5,1)\\(4,2)}} 2/6 \xrightarrow{(3,1)} 2/4 \longrightarrow 1/2$$

$$2/10 \xrightarrow{\substack{(2,3)\\(3,2)}} 2/5 \xrightarrow{(3,1)} 2/4 \longrightarrow 1/2$$

Pary liczb umieszczone nad strzałkami oznaczają parametry s_i, r_i danego układu $|H^i$.
W związku z tym zbadano złożoność rozwiązań alternatywnych układu $|H$ dla wyróżnionej grupy kodów 2/n. Najmniejszej liczby bramek wymagają układy o parametrach zamieszczonych w tabeli 4.2.3.

Tabela 4.2.3. Parametry struktury STC o minimalnej złożoności wybranych kodów 2/n

n	Struktura układu
5,6	$2/n \xrightarrow{(3,1)} 2/4 \longrightarrow 1/2$
7,8	$2/n \xrightarrow{(2,2)} 2/4 \longrightarrow 1/2$
9,10	$2/n \xrightarrow{(5,1)} 2/6 \xrightarrow{(3,1)} 2/4 \longrightarrow 1/2$
11,12	$2/n \xrightarrow{(6,1)} 2/7 \xrightarrow{(2,2)} 2/4 \longrightarrow 1/2$
13,14	$2/n \xrightarrow{(4,2)} 2/6 \xrightarrow{(2,2)} 2/4 \longrightarrow 1/2$
15,16	$2/n \xrightarrow{(5,2)} 2/7 \xrightarrow{(2,2)} 2/4 \longrightarrow 1/2$

4.2.5. Przykład

Stosując algorytm 4.2.1 zaprojektować STC kodu 2/13. Parametry s_i, r_i układu przyjmujemy na podstawie tabeli 4.2.3.

Projektowanie układu IH^1

1. $i=1$, $n_0=13$, $\{x_1, x_2, \ldots, x_{13}\}$ - zbiór zmiennych wejściowych.
2. $s_1=4$, $r_1=2$, $n_1=6$.
3. $b_1=3$, $c_1=4$, $v_1=1$;
   $A^{(1)} = \{x_1, x_2, x_3\}$, $A^{(2)} = \{x_4, x_5, x_6\}$, $A^{(3)} = \{x_7, x_8, x_9\}$,
   $A^{(4)} = \{x_{10}, x_{11}, x_{12}, x_{13}\}$.
4. $H_1^1 = x_1 + (x_2 \underline{+} x_3)$, $H_2^1 = x_4 + (x_5 \underline{+} x_6)$, $H_3^1 = x_7 + (x_8 \underline{+} x_9)$,
   $H_4^1 = (x_{10} \underline{+} x_{11}) + (x_{12} \underline{+} x_{13})$.
5. $H_5^1 = x_1 \cdot (x_2 \underline{+} x_3) + x_4 \cdot (x_5 \underline{+} x_6) + x_7 \cdot (x_8 \underline{+} x_9) +$
   $+ (x_{10} \underline{+} x_{11}) \cdot (x_{12} \underline{+} x_{13})$,
   $H_6^1 = x_2 \cdot x_3 + x_5 \cdot x_6 + x_8 \cdot x_9 + x_{10} \cdot x_{11} + x_{12} \cdot x_{13}$.
6, 7. - kroków tych nie wykonuje się.
8. Wspólne bramki OR stosowane w podukładach realizujących funkcje H_j^1, $j \in \{\overline{1, 5}\}$, podkreślono.

9. Ponieważ $n_1=6 > 4$ więc:
   a) $i=2$,
   b) $n_{i-1}=6$, $\{H_1^1, H_2^1, \ldots, H_6^1\}$ - nowy zbiór zmiennych wejściowych,
   c) przechodzimy do kroku 2.

Projektowanie układu IH^2

2. $s_2=3$, $r_2=1$, $n_2=4$.
3. $b_2=c_2=2$, $v_2=3$;
   $A^{(1)}=\{H_1^1, H_2^1\}$, $A^{(2)}=\{H_3^1, H_4^1\}$, $A^{(3)}=\{H_5^1, H_6^1\}$.
4. $H_1^2 = H_1^1+H_2^1$, $H_2^2 = H_3^1+H_4^1$, $H_3^2 = H_5^1+H_6^1$.
5. $H_4^2 = H_1^1 \cdot H_2^1+H_3^1 \cdot H_4^1+H_5^1 \cdot H_6^1$.
6. Nie wykonuje się.
7. Ponieważ $i > 1$ i $r_1 > 1$ modyfikujemy funkcje H_j^2, $j \in \{\overline{1,4}\}$:
   $H_1^2 = H_1^1+H_2^1$, $H_2^2 = H_3^1+H_5^1$, $H_3^2 = H_4^1+H_6^1$,
   $H_4^2 = H_1^1 \cdot H_2^1+H_3^1 \cdot H_5^1+H_4^1 \cdot H_6^1$.
8. Nie ma możliwości stosowania wspólnych bramek.
9. Nie wykonuje się.

Projektowanie układu IH^3

10. $H_3^1 = (H_1^2+H_3^2) \cdot (H_2^2+H_4^2)$,
    $H_3^2 = H_1^2 \cdot H_3^2+H_2^2 \cdot H_4^2$.

Funkcje otrzymane po wykonaniu kroków 4, 5 - dla układu IH^1, kroku 7 - dla układu IH^2, kroku 10 - dla układu IH^3 są funkcjami STC kodu 2/13.

4.2.6. Porównanie parametrów STC kodów 2/n zaprojektowanych różnymi metodami

W tabeli 4.2.4 zamieszczono parametry STC (o minimalnej liczbie bramek) niektórych kodów 2/n zaprojektowanych wg algorytmu 4.2.1 (w tabeli oznaczono gwiazdką "*") oraz z zastosowaniem najlepszych metod znanych z literatury, głównie z [35]; je-

dynie dla kodu 2/5 podajemy również wyniki otrzymane w [44]. Ponieważ w [35] nie podano ogólnych zależności pozwalających wyznaczyć liczbę bramek i połączeń, parametry układów podano jedynie dla wybranych kodów 2/n.

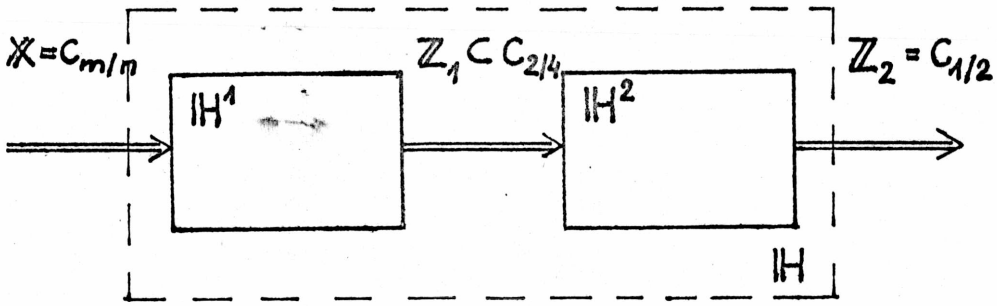
Tabela 4.2.4. Zestawienie parametrów STC kodów 2/n zaprojektowanych różnymi metodami

	Liczba bramek		Liczba wejść bramek		Liczba poziomów		Liczba testów		
n	R	[35]	R	[35]	R	[35]	R	[35]	
5	11	17 12	22	35 30	4	5 2	5	7 10	wg [44]
6	13	19	27	42	4	5	6	9	
7	18	22	36	49	5	5	9	11	
8	20	25	42	56	5	5	10	12	
9	22	36	47	79	6	8	14	16	
16	38	63	81	152	7	8	21	32	

Dla wyszczególnionych w tabeli 4.2.4 wybranych przykładowo kodów 2/n, STC zaprojektowane wg algorytmu 4.2.1 mają wszystkie cztery parametry (wyszczególnione w tabeli 4.2.4) lepsze niż analogiczne układy otrzymane wg metod z [35].
Jedynie układ dla kodu 2/5 ma większą liczbę poziomów niż układ otrzymany w [44]; jednakże pozostałe parametry posiada lepsze.

4.3. Kody m/n, $m \geq 3$, $(2.m+1) \leq n \leq 4.m$

STC kodów m/n takich, że $m \geq 3$ i $(2.m+1) \leq n \leq 4.m$, mają strukturę jak na rys. 2.5.1, która w tym przypadku upraszcza się do postaci przedstawionej na rys. 4.3.1.



Rys.4.3.1. Schemat STC kodów m/n takich, że $m \geq 3$ i $(2.m+1) \leq n \leq 4.m$

Układ $\mathbb{H}^1$ jest samotestowalnym kodowo-rozłącznym translatorem kodu m/n w kod 2/4. Układ $\mathbb{H}^2$ jest STC kodu 2/4, zaprojektowanym wg metody z [3] .

4.3.1. Podział zbioru słów kodu m/n na podzbiory

Zbiór zmiennych wejściowych $\{x_1, x_2, \ldots, x_n\}$ dzielimy na dwa niepuste podzbiory rozłączne $A = \{x_1, x_2, \ldots, x_{n_a}\}$, $B = \{x_{n_a+1}, \ldots, x_n\}$ o licznościach:

$$n_a = \left\lfloor \frac{n}{2} \right\rfloor, \quad n_b = \left\lceil \frac{n}{2} \right\rceil. \tag{4.3.1}$$

Prawdziwe są nierówności: $0 \leq n_b - n_a \leq 1$, $n_a \geq 3$, $n_b \geq 4$.

Z kolei zbiory A,B dzielimy na niepuste podzbiory rozłączne $A_1 = \{x_1, \ldots, x_{n_{a1}}\}$, $A_2 = \{x_{n_{a1}}, \ldots, x_{n_a}\}$, $B_1 = \{x_{n_a+1}, \ldots, x_{n_a+n_{b1}}\}$, $B_2 = \{x_{n_a+n_{b1}+1}, \ldots, x_{n_b}\}$ o licznościach:

$$n_{a1} = \left\lfloor \frac{n_a}{2} \right\rfloor, \; n_{a2} = \left\lceil \frac{n_a}{2} \right\rceil, \; n_{b1} = \left\lfloor \frac{n_b}{2} \right\rfloor, \; n_{b2} = \left\lceil \frac{n_b}{2} \right\rceil. \tag{4.3.2}$$

Ponadto przyjmujemy:

$$k_a = \left\lfloor \frac{m}{2} \right\rfloor, \quad k_b = \left\lceil \frac{m}{2} \right\rceil, \tag{4.3.3}$$

gdzie: $k_a \geqslant 1$, $k_b \geqslant 2$.

Zbiór $C_{m/n}$ można przedstawić jako sumę zbiorów słów kodów stałowagowych złożonych

$$C_{m/n} = \bigcup_{i=0}^{m} C_{i/n_a} \times C_{(m-i)/n_b} .$$

Zbiór $C_{m/n}$ dzielimy następnie na trzy podzbiory rozłączne:

1) $$C_{AB} = C_{k_a/n_a} \times C_{k_b/n_b},$$

2) $$C_A = \bigcup_{i=0}^{k_a-1} C_{i/n_a} \times C_{(m-i)/n_b} ,$$

3) $$C_B = \bigcup_{s=0}^{k_b-1} C_{(m-1)/n_a} \times C_{s/n_b} .$$

Dla uproszczenia zapisu wprowadzamy oznaczenie

$$D_V^i = C_{i/n_v} \times C_{(m-i)/n_{\bar{v}}}, \quad V = A,B , \quad i \in \{\overline{0,k_v-1}\},$$

gdzie: jeżeli V = A, to v = a, $\bar{v}$ = b,
jeżeli V = B, to v = b, $\bar{v}$ = a.

W szczególności jeżeli V = A, to $D_A^i = C_{i/n_a} \times C_{(m-i)/n_b}$,
jeżeli V = B, to $D_B^i = C_{(m-i)/n_a} \times C_{i/n_b}$.

Zbiory C_A, C_B można teraz przedstawić w postaci

$$C_V = \bigcup_{i=0}^{k_v-1} D_V^i, \quad V = \{A,B\}. \tag{4.3.4}$$

Z kolei, po uwzględnieniu podziału zbiorów A,B na podzbiory A_1, A_2,B_1,B_2, każdy ze zbiorów D_V^i ($V = \{A,B\}$, $i \in \{\overline{0,k_v-1}\}$) można teraz przedstawić w postaci:

$$D_V^i = \bigcup_{j=\underline{j}(i)}^{\bar{j}(i)} C_{i/n_v} \times C_{j/n_{\bar{v}1}} \times C_{1/n_{\bar{v}2}} \qquad (4.3.5)$$

gdzie: $1 = m-i-j$,

$$\underline{j}(i) = \max\{0, m-i-n_{\bar{v}2}\}, \qquad (4.3.6)$$

$$\bar{j}(i) = \min\{n_{\bar{v}1}, m-i\}. \qquad (4.3.7)$$

Dla zbioru $C_{i/n_v} \times C_{j/n_{v1}} \times C_{1/n_{v2}}$ będziemy w dalszym ciągu stosować wyłącznie skrócony zapis - $\{i/j,1\}$.
Zbiory D_V^i i C_V można teraz przedstawić w postaci:

$$D_V^i = \bigcup_{j=\underline{j}(i)}^{\bar{j}(i)} \{i/j,1\}, \quad V = \{A,B\}, \qquad (4.3.8)$$

$$C_V = \bigcup_{i=0}^{k_v-1} \bigcup_{j=\underline{j}(i)}^{\bar{j}(i)} \{i/j,1\}, \quad V = \{A,B\}. \qquad (4.3.9)$$

4.3.2. Ogólna postać funkcji realizowanej przez układ $|H^1$

Dla każdego słowa kodu m/n ze zbioru $\{i/j,1\} \subset C_V$ $(V=\{A,B\})$ funkcja progowa $T(v \geq k_v)$ oraz iloczyn funkcji progowych $T(v \geq i).T(\bar{v}_1 \geq j).T(\bar{v}_2 \geq 1)$ przyjmują równocześnie wartość 1. Natomiast dla każdego słowa kodu m/n ze zbioru C_{AB} funkcje progowe $T(a \geq k_a), T(b \geq k_b)$ równocześnie przyjmują wartość 1.
W związku z powyższym proponuje się następującą postać ogólną funkcji $H^1 = \{H_1^1, H_2^1, H_3^1, H_4^1\}$ realizowanej przez układ $|H^1$:

$$\left.\begin{aligned}
H_1^1 &= T(a \geq k_a) + \sum_{\{i/j,1\} \subset E_{A1}} T(a \geq i).T(b_1 \geq j).T(b_2 \geq 1), \\
H_2^1 &= T(b \geq k_b) + \sum_{\{i/j,1\} \subset E_{B2}} T(b \geq i).T(a_1 \geq j).T(a_2 \geq 1), \\
H_u^1 &= \sum_{\{i/j,1\} \subset E_{Au}} T(a \geq i).T(b_1 \geq j).T(b_2 \geq 1) + \sum_{\{i/j,1\} \subset E_{Bu}} \\
&\quad T(b \geq i).T(a_1 \geq j).T(a_2 \geq 1), \quad u = \{3,4\},
\end{aligned}\right\} \qquad (4.3.10)$$

gdzie E_{A1}, E_{A3}, E_{A4} oraz E_{B2}, E_{B3}, E_{B4} są rozłącznymi podzbiorami zbiorów C_A, C_B takimi, że:

$$(\{i/j,1\} \subset \{E_{A1} \cup E_{B2}\}) \Longleftrightarrow (i < k_v - 1), \tag{4.3.11}$$

$$\left(\bigcup_{w \in \{1,3,4\}} E_{Aw}\right) \left(\bigcup_{w \in \{2,3,4\}} E_{Bw}\right) = C_A \cup C_B. \tag{4.3.12}$$

Ograniczenie (4.3.11) jest podyktowane tym, że jeżeli $\{k_v-1/j,1\} \subset E_{Vw}$ (w = 1 - jeżeli V = A, w = 2 - jeżeli V = B), to żadne słowo kodu (m+1)/n, które powstało z dowolnego słowa kodu m/n ze zbioru $\{k_v-1/j,1\}$ wskutek wystąpienia błędu $0 \rightarrow 1$ na pozycji ze zbioru V, nie jest wykrywane przez układ $\|H^1$.

Główny problem projektowania układu $\|H^1$ polega teraz na podaniu takiej metody podziału zbiorów C_V (V = {A,B}) na podzbiory E_{Vw}, aby układ $\|H^1$ realizujący funkcje (4.3.10) był kodowo-rozłącznym translatorem kodu m/n w kod 2/4 samotestowalnym dla uszkodzeń pojedynczych typu s/z (ze zbioru F_p^1). Warunki wystarczające, nakładane na zbiory E_{Vw}, sformułowano w postaci twierdzeń 4.3.1 i 4.3.2. W twierdzeniach tych skorzystamy z następującej własności:

W.4.3.1)

a) $(d_B^0 = 1) \Longleftrightarrow (n_a < n_b \wedge m = n_a)$,

b) $(d_A^0 = 2) \Longleftrightarrow \left((n_a < n_b \wedge m = n_a) \vee (n_a = n_b \wedge m = n_a - 1)\right)$,

c) $(d_B^0 = 2) \Longleftrightarrow (n_a \leq n_b \wedge m = n_a - 1)$,

d) $(d_B^1 = 2) \Longleftrightarrow (n_a < n_b \wedge m = n_a)$,

e) w pozostałych przypadkach $d_V^i \geq 3$,

gdzie

$$d_V^i = \min\{n_{\bar{V}1}, m-i\} - \max\{0, m-i-n_{\bar{V}2}\} + 1 \tag{4.3.13}$$

oznacza liczność zbioru D_V^i, V = {A,B}, $i \in \{\overline{0,k_v-1}\}$.

Dowód - wynika z wykonania następujących kroków:

1) wzór (4.3.13) przekształcić do postaci

$$d_V^i = (n_{\bar{V}1}+1) + \min\{0, m-i-n_{\bar{V}1}\} - \max\{0, m-i-n_{\bar{V}2}\}, \qquad (4.3.14)$$

$$i \in \{\overline{0, k_V-1}\},$$

2) korzystając z tego, że:

i) $n_{b2} \leq m < n_b$ (wynika z założenia, że $m \geq 3$ i $(2.m+1) \leq n \leq 4.m$, oraz z (4.3.1) i (4.3.2)),

ii) $0 \leq n_{\bar{V}2} - n_{\bar{V}1} \leq 1$, $V = \{a,b\}$ (wynika z (4.3.2)),

uprościć wzór (4.3.14) rozważając następujące przypadki:

a) $m \leq i+n_{\bar{V}1} - d_V^i = m+1-i$,

b) $m = i+n_{\bar{V}2} - d_V^i = n_{\bar{V}1}+1$,

c) $m > i+n_{\bar{V}2} - d_V^i = n_{\bar{V}} + i + 1 - m$;

3) zbadać jakie wartości przyjmuje d_V^i ($V = \{A,B\}$) dla różnych parametrów m, i, $n_{\bar{V}1}$, $n_{\bar{V}}$.

Niech $\{c/d\}$ będzie skróconą wersją zapisu zbioru $C_{c/n_a} \times C_{d/n_b}$. Dla zbiorów $\{i/j,l\}$, $\{i'/j',l'\}$, $\{c/d\}$ wprowadzamy relację "$\leq$".

Definicja 4.3.1

$$(\{i/j,l\} \leq \{i'/j',l'\}) \Longleftrightarrow (i \leq i' \wedge j \leq j' \wedge l \leq l');$$

$$(\{i/j,l\} \leq \{c/d\}) \Longleftrightarrow \begin{cases} (i \leq c \wedge j + l \leq d) - \text{jeżeli } \{i/j,l\} \subset E_{Au}, \\ (i \leq d \wedge j + l \leq c) - \text{jeżeli } \{i/j,l\} \subset E_{Bu}. \end{cases}$$

Twierdzenie 4.3.1

Jeżeli dla każdego $\{i/j,l\} \subset E_{Vt}$, $V = \{A,B\}$, $t = \{1,4\}$, są spełnione warunki:

1) jeżeli $k_a=1$, to:

a) $(j \geq 1) \Longrightarrow (\{0/j-1,1+1\} \subset \{C_V \setminus E_{Vt}\} \vee \{0/j+1,1-1\} \subset \{C_V \setminus E_{Vt}\})$,

b) $(1 \geq 1) \Longrightarrow (\{0/j-1,1+1\} \subset \{C_V \setminus E_{Vt}\} \vee \{0/j+1,1-1\} \subset \{C_V \setminus E_{Vt}\})$;

2) jeżeli $k_v \geq 2$ oraz $i \in \{\overline{0,k_v-2}\}$, to:

a) $(i \geq 1) \Longrightarrow (\{i-1/j+1,1\} \subset \{C_V \setminus E_{Vt}\} \vee \{i-1/j,1+1\} \subset \{C_V \setminus E_{Vt}\})$,

b) $(j \geq 1) \Longrightarrow (\{i+1/j-1,1\} \subset \{C_V \setminus E_{Vt}\} \vee \{i/j-1,1+1\} \subset \{C_V \setminus E_{Vt}\})$,

c) $(1 \geq 1) \Longrightarrow (\{i+1/j,1-1\} \subset \{C_V \setminus E_{Vt}\} \vee \{i/j+1,1-1\} \subset \{C_V \setminus E_{Vt}\})$;

3) jeżeli $k_v \geq 2$ oraz $i = k_v-1$, to

$(\{k_v-2/j+1,1\} \subset \{C_V \setminus E_{Vt}\} \vee \{k_v-2/j,1+1\} \subset \{C_V \setminus E_{Vt}\})$,

to każdy nieredundancyjny bezinwersyjny układ $\mathbb{H}^1$, który realizuje funkcje (4.3.10) jest samotestowalny dla uszkodzeń pojedynczych ze zbioru F_p^1.

Dowód

Załóżmy, że układ $\mathbb{H}^1$ realizuje funkcje (4.3.10), jest nieredundancyjny i bezinwersyjny, oraz że są spełnione warunki 1-3 tw. 4.3.1. Na mocy twierdzeń 2.4.1 i 2.4.2 wystarczy wykazać, że są spełnione trzy warunki określone w tw. 2.4.1. Warunek 1 jest spełniony, gdyż układ $\mathbb{H}^1$ z założenia jest nieredundancyjny. Wykażemy, że są spełnione pozostałe dwa warunki.

Wprowadzamy oznaczenia:

$$M_{Vu} = \{m_k \mid (m_k \leftrightarrow X_k) \wedge (X_k \in E_{Vu})\}, \quad V = \{A,B\}, \quad u \in \{\overline{1,4}\},$$

$$M_V = \bigcup_{u \in \{\overline{1,4}\}} M_{Vu}, \quad V = \{A,B\},$$

$$M_{AB} = \{m_k \mid m_k \leftrightarrow X_k \wedge X_k \in C_{AB}\},$$

$$M_{1a} = \{m_k \mid m_k \leftrightarrow X_k \wedge X_k \in C_{k_a/n_a} \times C_{0/n_b}\},$$

$$M_{2b} = \{m_k \mid m_k \leftrightarrow X_k \wedge X_k \in C_{0/n_a} \times C_{k_b/n_b}\}.$$

Zbiory implikantów występujące w tw. 2.4.1 mają postać:

$$M' = \{m_k \mid m_k \leftrightarrow X_k \wedge X_k \in C_{m/n}\},$$

$$M'_1 = M_{AB} \cup M_{A1},$$

$$M'_2 = M_{AB} \cup M_{B2},$$

$$M'_w = M_{Aw} \cup M_{Bw}, \quad w = 3,4,$$

$$M_1 = M_{1a} \cup M_{A1},$$

$$M_2 = M_{2b} \cup M_{B2},$$

$$M_w = M'_w, \quad w = 3,4.$$

Ponieważ:

- $(\forall m_k \in M_{1a})(\exists X_1 \in C_{AB} \mid m_k \subset m_1 \wedge \sim(\exists m_p \in M_{1a} \mid m_p \neq m_k \wedge m_p \subset m_1))$,
- analogiczną własność ma zbiór M_{2b},
- jest spełniony warunek (4.3.11) oraz $C_{AB} \cup C_A \cup C_B = C_{m/n}$, więc jest spełniony war.2 tw. 2.4.1.

W zbiorach M_u, $u \in \{\overline{1,4}\}$, wyróżniamy trzy grupy implikantów, które są iloczynami: $k_a (k_a < m$, należą do zbioru $M_{1a})$, k_b $(k_b < m$, należą do zbioru $M_{2b})$ oraz m (w ogólnym przypadku występują we wszystkich czterech zbiorach M_u) zmiennych wejściowych. Kolejno dla każdej grupy implikantów wykażemy, że jest spełniony war.3 tw. 4.3.1.

Ponieważ parametru i dotyczy ograniczenie (4.3.11), więc $(\forall m_k \in M_{1a})(\exists X_1 \in \{C_A \setminus E_{A1}\} \mid m_k \subset m_1)$. Stąd wynika, że

$(\forall m_k \in M_{1a})(\forall m_k(x_r) \in m_k(A))(\exists m_1 \in \{M \setminus M_1\} \mid m_k(x_r) \subset m_1)$, tzn. dla

implikantów ze zbioru M_{1a} war.3 tw. 2.4.1 jest spełniony. Podobnie jest w przypadku implikantów ze zbioru M_{2b}.
Rozpatrzymy teraz implikanty, które są iloczynami m zmiennych. Każdemu zbiorowi $\{i/j,1\} \subset E_{Vu}$, $V = \{A,B\}$, $u \in \{\overline{1,4}\}$, odpowiada zbiór implikantów $M_V(i,j,1) = \{m_k \mid (m_k \leftrightarrow X_k) \wedge (X_k \in \{i/j,1\}) \wedge \wedge (\{i/j,1\} \subset E_V)\}$; $M_V(i,j,1) \subset M_{Vu}$.
Zbiorom $\{i-1/j,1\}$ (jeżeli $i \geq 1$), $\{i/j-1,1\}$ (jeżeli $j \geq 1$) oraz $\{i/j,1-1\}$ (jeżeli $1 \geq 1$) odpowiadają zbiory dzielników implikantów:

$$M_V^1(i-1,j,1) = \{m_k(x_r) \mid m_k \in M_V(i,j,1) \wedge x_r \in V\},$$

$$M_V^2(i,j-1,1) = \{m_k(x_r) \mid m_k \in M_V(i,j,1) \wedge x_r \in \bar{V}1\},$$

$$M_V^3(i,j,1-1) = \{m_k(x_r) \mid m_k \in M_V(i,j,1) \wedge x_r \in \bar{V}2\}.$$

Oczywiście, jeżeli któryś z parametrów i,j,1 jest równy zero, to odpowiedni zbiór dzielników implikantów jest pusty.
Odrębnie rozpatrzymy dwa przypadki: 1) $k_a=1$, 2)$k_v \geq 2$, $v = \{a,b\}$, które, jak wynika z (4.3.3), wyczerpują wszystkie możliwości.

$k_a = 1$

Jeżeli $k_a=1$, to: $i=0$, $t = \{3,4\}$, $M_V^1 = \emptyset$.
Warunek 1a tw. 4.3.1 można zapisać w postaci

$$\left(\forall M_A(0,j,1) \mid \{0/j,1\} \subset E_{At}, t=\{3,4\}\right)\left((j \geq 1) \Rightarrow \right. \qquad (4.3.15)$$

$$\left. \Rightarrow \left(M_A(0,j-1,1+1) \subset M_{A\bar{t}}\right) \vee \left(M_A(0,j+1,1-1) \subset M_{A\bar{t}}\right)\right),$$

gdzie t=3 i t=4 lub odwrotnie.
Z (4.3.15) wynika, że

$$\left(\forall m_k(x_r) \in M_V^2(0,j-1,1)\right)\left(\exists m_p \in M_{A\bar{t}} \mid m_k(x_r) \subset m_p\right).$$

Podobnie jest w przypadku 1b. Oznacza to, że w tym przypadku jest spełniony war. 3 tw. 2.4.1.

$k_v \geq 2$

Z podobnych przesłanek jak w przypadku $k_a=1$ wynika, że dla implikantów ze zbiorów M_V (i,j,1), V = {A,B}, i $\in \{\overline{0,k_v-2}\}$, war.3 tw.2.4.1 jest spełniony. Rozważymy zatem zbiór $M_V(k_v-1,j,1)$. Warunek 3 tw. 4.3.1 zapisujemy w nowej postaci

$$(k_v \geq 2) \Rightarrow \Big(\big(\forall M_V(k_v-1,j,1) \,|\, \{k_v-1/j,1\} \subset E_{Vt}, V=\{A,B\}, t \in \{\overline{1,4}\}\big)$$

$$\big(\big(M_V(k_v-2,j+1,1) \subset \{M_V \setminus M_{Vt}\}\big) \vee \big(M_V(k_v-2,j,1+1) \subset \{M_V \setminus M_{Vt}\}\big)\big)\Big) \quad (4.3.16)$$

Ponadto zachodzą relacje:

$$\{k_v-1/j-1,1\} \leq \{k_a/k_b\}, \quad (4.3.17')$$

$$\{k_v-1/j,1-1\} \leq \{k_a/k_b\}. \quad (4.3.17")$$

Ponieważ występuje ograniczenie (4.3.11), więc:

$$(\forall j \in \{\underline{j}(k_v-1), \overline{j}(k_v-1)\}) \; M_V(k_v-1,j,1) \subset \{M_{V3} \setminus M_{V4}\}, \quad (4.3.18)$$

$$\{M_V \setminus M_{Vt}\} = M_{V\bar{t}}, \; t = \{3,4\}. \quad (4.3.19)$$

Warunek (4.3.16) oraz relacje (4.3.17') i (4.3.17") zapisujemy w postaci równoważnej:

$$\big(\forall m_k(x) \in M_V^1(k_v-1,j-1,1)\big)\big(\exists m_p \in M_{AB} \,|\, m_k(x) \subset m_p\big), \quad (4.3.20)$$

$$\big(\forall m_k(x) \in \{M_V^2(k_v-1,j-1,1) \cup M_V^3(k_v-1,j,1-1)\}\big)\big(\exists m_p \in M_{AB} \,|\, m_k(x) \subset m_p\big). \quad (4.3.21)$$

Ponieważ $M_{AB} \subset \{M' \setminus M_V\}$, więc z (4.3.20) i (4.3.21) wynika, że war.3 tw. 2.4.1 i w tym przypadku jest spełniony

Q.E.D.

Twierdzenie 4.3.2

Jeżeli dla każdego $\{i/j,1\} \subset E_{Vt}$, V = {A,B}, t $\in \{\overline{1,4}\}$, są spełnione warunki:

1) $(i < k_v-1) \Longrightarrow (\{i+1/j-1,1\} \subset \{C_V \setminus E_{Vt}\} \vee \{i+1/j,1-1\} \subset \{C_V \setminus E_{Vt}\})$,

2) $(j < n_{\overline{v}1}) \Rightarrow (\{i-1/j+1,1\} \subset \{C_V \smallsetminus E_{Vt}\} \vee \{i/j+1,1-1\} \subset \{C_V \smallsetminus E_{Vt}\})$,

3) $(1 < n_{\overline{v}2}) \Rightarrow (\{i-1/j,1+1\} \subset \{C_V \smallsetminus E_{Vt}\} \vee \{i/j-1,1+1\} \subset \{C_V \smallsetminus E_{Vt}\})$,

to układ $\|H^1$, który realizuje funkcje (4.3.10) jest kodowo-rozłącznym translatorem kodu m/n w kod 2/4.

Dowód

Załóżmy, że warunki 1-3 są spełnione. Wykażemy, że układ $\|H^1$, który realizuje funkcje (4.3.10) spełnia warunki (korzystamy z def. 2.1.3):

1) $(\forall X \in C_{m/n})(H^1(X) \in C_{2/4})$,

2) $(\forall X \in C_{m/n})(H^1(X) \in C_{2/4})$.

Ad.1) Funkcje (4.3.10) spełniają warunki:

i) $(\forall X \in \{C_{AB} \cup C_{A1} \cup C_{B2}\})(H^1(X) = (1100) \in C_{2/4})$,

ii) $(\forall X \in C_{A3})(H^1(X) = (1001) \in C_{2/4})$,

iii) $(\forall X \in C_{A4})(H^1(X) = (1001) \in C_{2/4})$, (4.3.22)

iv) $(\forall X \in C_{B3})(H^1(X) = (0110) \in C_{2/4})$,

v) $(\forall X \in C_{B4})(H^1(X) = (0101) \in C_{2/4})$.

Ponieważ ponadto jest spełniony warunek (4.3.11), więc warunek 1 jest spełniony.

Ad.2) Ponieważ funkcje (4.3.10) są monotoniczne i są spełnione warunki:

$$(\forall X' \in \bigcup_{i=0}^{m-2} C_{i/n})(\exists X'' \in C_{(m-1)/n} | X' < X''),$$

$$(\forall X' \in \bigcup_{i=m+2}^{n} C_{i/n})(\exists X'' \in C_{(m+1)/n} \mid X'' < X'),$$

wystarczy wykazać, że:

i) $(\forall X \in C_{(m-1)/n})\left(H_1^1(X) \in \{C_{0/4} \cup C_{1/4}\}\right)$,

ii) $(\forall X \in C_{(m+1)/n})\left(H_1^1(X) \in \{C_{3/4} \cup C_{4/4}\}\right)$.

Ad.i) Spośród funkcji H_u^1, $u \in \{\overline{1,4}\}$, iloczyny stopnia niższego od m występują jedynie w funkcjach H_1^1, H_2^1 w członach odpowiadających funkcjom progowym $T(a \geqslant k_a)$ - w funkcji H_1^1, $T(b \geqslant k_b)$ - w funkcji H_2^1. Jeżeli $X \in C_{(m-1)/n}$, to tylko dla jednej z tych funkcji zachodzi $H_u^1(X) = 1$, $u = 1,2$, dla pozostałych natomiast, zachodzi $H_w^1(X) = 0$, $w \in \{\overline{1,4}\}$; $w \neq u$. Warunek i) jest więc spełniony.

Ad.ii) Załóżmy, że słowo $X \in C_{(m+1)/n}$ powstało wskutek wystąpienia błędu $0 \rightarrow 1$ ze słowa: a) $X' \in \{c/d\}$, b) $X' \in \{i/j,1\}$, $\{i/j,1\} \subset E_{Vt}$, $V = \{A,B\}$, $t \in \{\overline{1,4}\}$.

Ad.a) Jeżeli $X' \in \{c/d\}$ to $H_1^1(X') = H_2^1(X') = 1$. Jeżeli błąd wystąpił na pozycji ze zbioru A, to $(\exists X'' \in C_A \mid X'' < X)$, a z (4.3.11) wynika, że $X'' \in \{E_{A3} \cup E_{A4}\}$. Zatem $H_1^1(X) = H_2^1(X) = H_u^1(X) = 1$, $u = 3$ lub 4. Podobnie jest jeżeli błąd wystąpił na pozycji ze zbioru B.

Ad.b) Jeżeli $V = A$, to $H_1^1(X') = H_t^1(X') = 1$, $t \in \{2,3,4\}$; jeżeli $V = B$, to $H_2^1(X') = H_t^1(X') = 1$, $t \in \{1,3,4\}$. Jeżeli błąd wystąpił na pozycji ze zbioru V, to:

- jeżeli $i < k_v - 1$, to z war. 1 tw. 4.3.2 wynika, że $(\exists X'' \in \{C_V \setminus E_{Vt}\} \mid X'' < X)$. Jeżeli $V = A$, to z (4.3.11) wynika, że: $X'' \in E_{Au}$, $u \neq t$ i $u \neq 1$. Tym samym $H_1^1(X) = H_t^1(X) = H_u^1(X) = 1$. Podobnie jest, gdy $V = B$;
- jeżeli $i = k_v - 1$, to $(\exists X'' \in C_{AB} \mid X'' < X')$ i wtedy: $H_1^1(X) = H_2^1(X) = H_t^1(X) = 1$.

Jeżeli $j = n_{\overline{V1}}$, to błąd nie może wystąpić na pozycji ze zbioru V1. Załóżmy, że $j < n_{\overline{V1}}$. Ponieważ jest spełniony war.2 tw. 4.3.2, więc $(\exists X'' \in \{C_V \setminus E_{Vt}\} \mid X'' < X)$. Jeżeli $V = A$,

to z (4.3.11) wynika, że $X'' \in E_{Vu}$, $u \neq 1$ i $u \neq t$. Zatem $H_1^1(X) = H_t^1(X) = H_u^1(X) = 1$. Podobnie jest dla V = B. Jeżeli błąd wystąpił na pozycji ze zbioru $\overline{V2}$ dowód jest analogiczny jak dla zbioru $\overline{V1}$.

W każdym z przypadków a oraz b jest zatem spełniony warunek $\left(X \in C_{(m+1)/n}\right) \Rightarrow \left(H^1(X) \in C_{3/4}\right)$.

Q.E.D.

4.3.3. Algorytm projektowania STC

W bieżącym podrozdziale przedstawiamy kolejno: - algorytm 4.3.1 - podziału zbioru C_V ($V = \{A,B\}$) na podzbiory $E_{Vt}(t \in \{\overline{1,4}\})$, który wykorzystuje się w algorytmie 4.3.2 - projektowania STC kodów m/n, oraz tw. 4.3.3, w którym wykazano, że każdy układ zaprojektowany wg alg. 4.3.2 jest STC.

Algorytm 4.3.1 - podziału zbioru C_V na podzbiory $E_{Vt}(t \in \{\overline{1,4}\})$.

Dane wejściowe tego algorytmu otrzymuje się w trakcie wykonywania alg. 4.3.2.

1. Jeżeli V = A i $k_a = 1$ wykonać krok 2. W przeciwnym razie przejść do kroku 3.

2. Przyjąć postać zbiorów E_{Vt}:

$$E_{A1} = E_{A2} = \emptyset,$$

$$E_{A3} = \begin{cases} \{0/1,2\} & \text{dla } n = 7,8; \\ \{\{0/0,3\},\{0/2,1\}\} & \text{dla } n = \{\overline{9,12}\}; \end{cases}$$

$$E_{A4} = \begin{cases} \{0/2,1\} & \text{dla } n = 7,8; \\ \{0/1,2\} & \text{dla } n = 9,10; \\ \{\{0/1,2\},\{0/3,0\}\} & \text{dla } n = 11,12. \end{cases}$$

Koniec algorytmu.

3. Jeżeli V = B oraz $d_B^0 = \min\{n_{a1}, m\} - \max\{0, m-n_{a2}\} = 1$ wykonać krok 4. W przeciwnym razie przejść do kroku 5.

4. Utworzyć zbiory:

$$E_{B1} = E_{B2} = \emptyset,$$

$$E_{B3} = \bigcup_{i} D_V^i, \quad i \text{ - nieparzyste, } i \in \{\overline{1,k_v-1}\},$$

$$E_{B4} = \bigcup_{i} D_V^i, \quad i \text{ - parzyste, } i \in \{\overline{0,k_v-1}\}.$$

Koniec algorytmu.

5. Kolejno dla każdego $i \in \{\overline{0,k_v-1}\}$ wyznaczyć $d_V^i = \min\{n_{\overline{v}1}, m-i\} - \max\{0, m-i-n_{\overline{v}2}\}$, a następnie ze zbioru D_V^i utworzyć zbiory D_{Vu}^i ($u \in \{\overline{1,3}\}$) wykonując kolejno kroki a - d:

a) Utworzyć zbiory:

$$D_{V1}^i = \bigcup_{j} \{i/j,1\}, \quad j = \underline{j}+2.p, \quad p \in \left\{\overline{0, \left\lfloor \frac{d_V^i-1}{2} \right\rfloor}\right\},$$

$$D_{V2}^i = \bigcup_{j} \{i/j,1\}, \quad j = \underline{j}+4.p+1, p \in \left\{\overline{0, \left\lfloor \frac{d_V^i-2}{4} \right\rfloor}\right\}.$$

Jeżeli $d_V^i < 4$ przyjąć $D_{V3}^i = \emptyset$ i przejść do kroku b. W przeciwnym razie utworzyć zbiór

$$D_{V3}^i = \bigcup_{j} \{i/j,1\}, \quad j = \underline{j}+4.p+3, \quad p \in \left\{\overline{0, \left\lfloor \frac{d_V^i-4}{4} \right\rfloor}\right\}.$$

b) Jeżeli $i = k_v-1$ przyjąć $D_{V2}^i = \{D_{V2}^i \cup D_{V3}^i\}$.

c) Jeżeli $i \in \{\overline{0,k_v-2}\}$ oraz d_V^i jest nieparzyste, to zbiór $\{i/\overline{j}-1, \underline{1}+1\} \subset D_{Vp}^i$ przenieść do zbioru D_{Vp}^i, a zbiór $\{i/\overline{j}, \underline{1}\} \subset D_{V1}^i$ przenieść do zbioru D_{Vp}^i, gdzie jeżeli $p = 2$, to $\overline{p} = 3$, a jeżeli $p = 3$ to $\overline{p} = 2$.

d) Jeżeli $\{i/j,1\} \subset D_{V3}^i$ to przyjąć $D_{V2}^i = D_{V3}^i$ oraz $D_{V3}^i = D_{V2}^i$.

6. Utworzyć zbiory:

$$E_{Vw} = \left(\bigcup_{i} D_{V1}^i\right) \cup \left(\bigcup_{j} D_{V2}^j\right), \quad i,j \in \{\overline{0,k_v-2}\},$$

i - parzyste, j - nieparzyste,

$$E_{Vw} = \emptyset,$$

gdzie: jeżeli $V = A$, to $w = 1$, $\overline{w} = 2$,

jeżeli V = B, to w = 2, $\overline{w}$ = 1;

$$E_{V3} = \left(\bigcup_i D^i_{V1}\right) \cup \left(\bigcup_i D^j_{V2}\right) \cup D^{k_v-1}_{Vp},$$

$i,j \in \{\overline{0,k_v-2}\}$, i - nieparzyste, j - parzyste,

$$E_{V4} = \left(\bigcup_{i=0}^{k_v-2} D^i_{V3}\right) \cup D^{k_v-1}_{Vp},$$

gdzie: p = 1 i $\bar{p}$ = 2 - jeżeli $d_V^{k_v-1}$ jest parzyste i $D^{k_v-2}_{V2} \subset E_{V3}$,

p = 2 i $\bar{p}$ = 1 - jeżeli $d_V^{k_v-1}$ jest nieparzyste lub $D^{k_v-2}_{V1} \subset E_{V3}$.

Koniec algorytmu.

Algorytm 4.3.2 - projektowania STC kodów m/n, $m \geq 3$, $2.m+1 \leq n \leq 4.m$

Dane wejściowe algorytmu stanowią: m,n, zbiór zmiennych wejściowych $\{x_1, x_2, \ldots, x_n\}$.

1. Zbiór zmiennych wejściowych $\{x_1, x_2, \ldots, x_n\}$ podzielić na podzbiory rozłączne $A = \{x_1, x_2, \ldots, x_{n_a}\}$, $B = \{x_{n_a+1}, \ldots, x_n\}$ o licznościach $n_a = \lfloor \frac{n}{2} \rfloor$, $n_b = \lceil \frac{n}{2} \rceil$.

2. Przyjąć: $k_a = \lfloor m/2 \rfloor$, $k_b = \lceil m/2 \rceil$, $n_{a1} = \lfloor n_a/2 \rfloor$ $n_{a2} = \lceil n_a/2 \rceil$,

$n_{b1} = \lfloor n_b/2 \rfloor$, $n_{b2} = \lceil n_b/2 \rceil$.

oraz utworzyć zbiory: $A_1 = \{x_1, \ldots, x_{n_{a1}}\}$, $A_2 = \{x_{n_{a1}}, \ldots, x_{n_a}\}$,

$B_1 = \{x_{n_a+1}, \ldots, x_{n_a+n_{b1}}\}$, $B_2^{a1} = \{x_{n_a+n_{b1}+1}, \ldots x_n\}$.

3. Zbiór $C_V (V = \{A,B\})$ podzielić na podzbiory

$$C_V = \bigcup_{i=0}^{k_v-1} D^i_V,$$

gdzie: $$D^i_V = \bigcup_{j=\underline{j}(i)}^{\overline{j}(i)} \{i/j,1\},$$

$$\underline{j}(i) = \max\{0, m-i-n_{\overline{V}2}\},$$

$$\overline{j}(i) = \min\{n_{\overline{V}1}, m-i\}.$$

4. Utworzyć zbiory E_{Vu}, $V = \{A,B\}$, $u \in \{\overline{1,4}\}$, stosując algorytm 4.3.1 kolejno dla V = A i V = B.

5. Wyznaczyć funkcje układu $\mathbb{H}^1$:

$$\left.\begin{aligned} H_1^1 &= T(a \geq k_a) + \overline{\sum_{\{i/j,l\} \subset E_{A1}}} T(a \geq i).T(b_1 \geq j).T(b_2 \geq l), \\ H_2^1 &= T(b \geq k_b + \overline{\sum_{\{i/j,l\} \subset E_{B2}}} T(b \geq i).T(a_1 \geq j).T(a_2 \geq l), \\ H_u^1 &= \overline{\sum_{\{i/j,l\} \subset E_{Au}}} T(a \geq i).T(b_1 \geq j).T(b_2 \geq l) + \\ &\quad + \overline{\sum_{\{i/j,l\} \subset E_{Bu}}} T(b \geq i).T(a_1 \geq j).T(a_2 \geq l), \quad u=3,4. \end{aligned}\right\} \quad (4.3.10)$$

6. Stosując algorytm 3.3.2 każdą funkcję H_u^1 ($u \in \{\overline{1,4}\}$) zapisać w postaci uwzględniającej wielopoziomową strukturę układów progowych.

7. W realizacji układowej funkcji H_u^1, $u \in \{\overline{1,4}\}$, zastosować wspólne podukłady.

8. Układ $\mathbb{H}^2$ zrealizować dla następujących funkcji:

$$\left.\begin{aligned} H_1^2 &= (H_1^1 + H_3^1).(H_2^1 + H_4^1), \\ H_2^2 &= H_1^1 . H_3^1 + H_2^1 . H_4^1. \end{aligned}\right\} \quad (4.3.23)$$

Koniec algorytmu.

W dowodzie tw. 4.3.3 skorzystamy z następujących własności zbiorów D_V^i - W.4.3.2 i W.4.3.3, oraz zbiorów D_{Vp}^i i E_{Vs} (otrzymanych po wykonaniu kroków 5 i 6 alg. 4.3.1) - W.4.3.4 ÷ W.4.3.6 .

W.4.3.2) Jeżeli $k_v \geq 2$ oraz $i \in \{\overline{1,k_v-1}\}$ ($v = \{a,b\}$), to:

$$\underline{j}(i-1) - \underline{j}(i) = 1 \text{ dla } m \geq i+n_{\overline{v}2},$$

$$\underline{j}(i-1) = \underline{j}(i) \text{ dla } m \leq i-1+n_{\overline{v}2},$$

gdzie: $\underline{j}(i) = \max\{0, m-i-n_{\overline{v}2}\}$,

$\underline{j}(i-1) = \max\{0, m-(i-1)-n_{\overline{v}2}\}$.

Dowód - wynika ze sprawdzenia przypadków: a) $m \leq i-1+n_{\bar{v}2}$,
b) $m \geq i+n_{\bar{v}2}$.

W.4.3.3) Jeżeli $k_v \geq 2$ oraz $i \in \{\overline{1,k_v-1}\}$ $(v = \{a,b\})$, to:

$$\bar{j}(i-1)-\bar{j}(i) = 1 \text{ dla } m \leq i-1+n_{\bar{v}1},$$

$$\bar{j}(i) = \bar{j}(i-1) \text{ dla } m \geq i+n_{\bar{v}1},$$

gdzie: $\bar{j}(i) = \min\{n_{\bar{v}1}, m-i\}$,

$\bar{j}(i-1) = \min\{n_{\bar{v}1}, m-(i-1)\}$.

Dowód - wynika ze sprawdzenia przypadków: a) $m \leq i-1+n_{\bar{v}1}$,
b) $m \geq i+n_{\bar{v}1}$.

Z własności W.4.3.2 i W.4.3.3 wynikają wnioski:

1) między parametrem j, $j \in \{\overline{\underline{j}(i), \bar{j}(i)}\}$, a parametrami $\underline{j}(i-1), \bar{j}(i-1)$ mogą zachodzić relacje:

a) $j = \underline{j}(i) < \underline{j}(i-1) = j+1$;

b) $j = \underline{j}(i) = \underline{j}(i-1)$;

c) $\underline{j}(i-1) < j < \bar{j}(i-1)$, gdzie $\underline{j}(i) < j < \bar{j}(i)$;

d) $j = \bar{j}(i) = \bar{j}(i-1)$;

2) między parametrem j, $j \in \{\overline{\underline{j}(i), \bar{j}(i)}\}$, a parametrami $\underline{j}(i+1), \bar{j}(i+1)$ mogą zachodzić relacje:

a) $j = \underline{j}(i) = \underline{j}(i+1) + 1$;

b) $j = \underline{j}(i) = \underline{j}(i+1)$;

c) $\underline{j}(i+1) < j < \bar{j}(i+1)$, gdzie $\underline{j}(i) < j < \bar{j}(i)$;

d) $j = \bar{j}(i) = \bar{j}(i+1)$;

e) $j = \bar{j}(i) = \bar{j}(i+1) + 1;$

W.4.3.4) $(\forall D^i_{Vp},\ V = \{A,B\},\ p \in \{\overline{1,3}\},\ i \in \{\overline{0,k_v-1}\})$

$(\forall \{i/j,l\} \subset D^i_{Vp})(\{i/j-1,l+1\} \not\subset D^i_{Vp} \wedge$

$\wedge \{i/j+1,l-1\} \not\subset D^i_{Vp}).$

W.4.3.5) $(\forall E_{Vs},\ V = \{A,B\},\ s \in \{\overline{1,4}\})\ (\exists D^i_{Vp}, D^i_{Vr}, i \in \{\overline{0,k_v-1}\},$

$p,r \in \{\overline{1,3}\},\ p \neq r\ |(D'_{Vp} \neq \emptyset) \wedge (D'_{Vr} \neq \emptyset) \wedge$

$\wedge (\{D^l_{Vp} \cup D^l_{Vr}\} \subset E_{Vs})).$

W.4.3.6) $(\forall i \in \{\overline{0,k_v-1}\},\ v = \{a,b\})\ (\{i-1/\underline{j},\bar{1}\} \subset E_{Vr} \wedge$

$\wedge \{i/\underline{j},\bar{1}\} \subset E_{Vs},\ r \neq s) \wedge (\{i-1/j,1\} \subset E_{Vt} \wedge$

$\wedge \{i/j,1\} \subset E_{Vw},\ t \neq w).$

Dowody własności W.4.3.4–W.4.3.6 pomijamy.

Twierdzenie 4.3.3

Układ $|H$ zaprojektowany wg algorytmu 4.3.2 jest układem kontrolnym kodu m/n samotestowalnym dla wszystkich uszkodzeń jednokierunkowych (ze zbioru F_u).

Dowód

Układ $|H$ zaprojektowany wg algorytmu 4.3.2 spełnia założenia tw. 2.5.3. Zatem wystarczy wykazać, że spełnione są trzy warunki podane w tw. 2.5.3 (warunki te kolejno cytujemy).

1a) Układ $|H^1$ jest samotestowalny dla uszkodzeń ze zbioru F^1_p – wykażemy, że zbiory E_{Vt} otrzymane w wyniku podziału zbiorów C_V ($V = \{A,B\}$) wg algorytmu 4.3.1 spełniają warunki określone w tw. 4.3.1.

Rozważymy następujące przypadki:

i) $k_a=1$, ii) $d^0_B=1$, iii) $k_v \geq 2$ i $d^0_V \geq 2$.

Ad.i) Ze sprawdzenia wszystkich zbiorów E_{At} otrzymanych po wykonaniu kroku 2 alg. 4.3.1 wynika, że warunek 1 tw. 4.3.1 jest spełniony.

Ad.ii) Wykażemy, że zbiory E_{Bt} otrzymane po wykonaniu kroku 4 alg. 4.3.1 spełniają warunki 2 i 3 tw. 4.3.1. Jeżeli $d_B^o = 1$, zbiory D_B^i mają własności:

1) $(\forall i \in \{\overline{0,k_v-1}\})(\bar{j}(i) = n_{a2})$,

2) $(\forall i \in \{\overline{1,k_v-2}\})(\underline{j}(i) = \underline{j}(i-1)-1)$,

3) $(\forall i \in \{\overline{0,k_v-2}\})(D_V^i \subset E_{Bw} \Rightarrow D_B^{i+1} \subset E_{B\bar{w}})$,

4) $(\forall i \in \{\overline{1,k_v-1}\})(D_V^i \subset E_{Bw} \Rightarrow D_B^{i-1} \subset E_{B\bar{w}})$,

gdzie: ($w=3$ i $\bar{w}=4$) lub ($w=4$ i $\bar{w}=3$).

Wynika z nich, że dla każdego $\{i/j,1\} \subset E_{Bt}$ ($i \in \{\overline{0,k_b-1}\}$, $t = \{3,4\}$) są spełnione warunki 2 i 3 tw. 4.3.1, gdyż:

$$(i \geq 1) \Rightarrow \Big(\big[(j = \underline{j}(i)) \Rightarrow (\{i-1/j+1,1\} \subset \{C_B \setminus E_{Bt}\}) \big] \wedge$$
$$\wedge \big[(j \neq \underline{j}(i)) \Rightarrow ((\{i-1/j+1,1\} \subset \{C_B \setminus E_{Bt}\}) \wedge$$
$$\wedge (\{i-1/j,1+1\} \subset \{C_B \setminus E_{Bt}\}) \big] \Big),$$

$$(j \geq 1) \Rightarrow (\{i+1/j-1,1\} \subset \{C_B \setminus E_{Bt}\}),$$

$$(1 \geq 1) \Rightarrow (\{i+1/j,1-1\} \subset \{C_B \setminus E_{Bt}\}).$$

Ad.iii) Załóżmy, że $k_V \geq 2$, $d_V^o \geq 2$, $\{i/j,1\} \subset D_{Vu}^i$, $D_{Vu}^i \subset E_{Vt}$. Wykażemy, że zbiory E_{Vt} ($V = \{A,B\}$, $t \in \{\overline{1,4}\}$) otrzymane w krokach 5 i 6 algorytmu 4.3.1 spełniają warunki 2 i 3 tw. 4.3.1. Z własności W.4.3.1 wynika, że jeżeli $d_V^o \geq 2$, to: $(\forall V, V = \{A,B\})(\forall i \in \{\overline{0,k_v-1}\})(d_V^i \geq 2)$.

Warunek 2 - zakładamy, że $i \in \{\overline{0.k_v-2}\}$.

a) Załóżmy, że $i \geq 1$. Skorzystamy z wniosku 1 z własności W.4.3.2 i W.4.3.3. Jeżeli między parametrem j a parametrami $\underline{j}(i-1)$, $\bar{j}(i-1)$ zachodzą relacje a,b

lub d, to z własności W.4.3.6 wynika, że war. 2a jest spełniony. Jeżeli zachodzi relacja c, to z własności W.4.3.4 wynika, że $\{i-1/j,1+1\} \subset D_{Vp}^{i-1}$ i $\{i-1/j+1,1\} \subset D_{Vw}^{i-1}$ $(p \neq r)$, a z własności W.4.3.5 wynika, że $D_{Vp}^{i-1} \subset \{C_V \setminus E_{Vt}\}$ lub $D_{Vr}^{i-1} \subset \{C_V \setminus E_{Vt}\}$.

b) Załóżmy, że $j \geq 1$.
Jeżeli $\underline{j}(i) < j \leq \overline{j}(i)$, to z własności W.4.3.4 i W 4.3.5 wynika, że $\{i/j-1,1+1\} \subset \{C_V \setminus E_{Vt}\}$.
Jeżeli $j = \underline{j}(i)$, to z wniosku 2 z własności W.4.3.2 i W.4.3.3 wynika, że $j = \underline{j}(i+1)+1$ lub $j = \underline{j}(i+1)$. Ponieważ $j \geq 1$, więc $j = \underline{j}(i+1)+1$. Z własności W.4.3.6 wynika, że $\{i+1/j-1,1\} \subset \{C_V \setminus E_{Vt}\}$.

c) Dowód jest analogiczny jak dla warunku 2b.

Warunek 3 - ponieważ $k_V - 2 \geq 0$, więc dowód jest analogiczny jak dla warunku 2a.

1b) Układ H^1 jest translatorem kodu m/n w kod 2/4
- wykażemy, że zbiory E_{Vt} otrzymane w wyniku podziału zbiorów C_V $(V = \{A,B\})$ wg algorytmu 4.3.1 spełniają warunki określone w tw. 4.3.2.
Rozważymy następujące przypadki:
i) $k_a = 1$, ii) $d_B^0 = 1$, iii) $k_V \geq 2$ i $d_V^0 \geq 2$.

Ad.i) Ze sprawdzenia wszystkich zbiorów E_{At} otrzymanych po wykonaniu kroku 2 alg. 4.3.1 wynika, że warunki określone w tw. 4.3.2 są spełnione.

Ad.ii) Skorzystamy z własności 1-4 zbiorów D_B^i wyszczególnionych w dowodzie tego twierdzenia dla przypadku 1a - ii). Z własności tych wynika, że dla każdego $\{i/j,1\} \subset E_{Bt}$ i $\in \{\overline{0,k_b-1}\}$, $t = \{3,4\}$ są spełnione warunki 1-3 tw. 4.3.2, gdyż:

$$(i < k_b-1) \Rightarrow (\{i+1/j-1,1\} \subset \{C_B \setminus E_{Bt}\} \wedge \{i+1/j,1\} \subset \{C_B \setminus E_{Bt}'\})$$

$$(j < n_{a1}) \Rightarrow (\{i-1/j+1,1\} \subset \{C_B \setminus E_{Bt}\}),$$

$$(j < n_{a2}) \Rightarrow (\{i-1/j,1+1\} \subset \{C_B \setminus E_{Bt}\}).$$

Ad.iii) Załóżmy, że $k_V \geq 2$, $d_V^o \geq 2$, $\{i/j,1\} \subset D_{Vu}^i$, $D_{\overline{Vu}}^i \subset E_{Vt}$. Wykażemy, że zbiory E_{Vt} ($V = \{A,B\}$, $t \in \{1,4\}$) otrzymane po wykonaniu kroków 5 i 6 algorytmu 4.3.1 spełniają warunki 1-3 tw. 4.3.2.

Warunek 1

Załóżmy, że $i < k_V - 1$. Skorzystamy z wniosku 2 z własności W.4.3.2 i W.4.3.3. Jeżeli między parametrem j a parametrami $\underline{j}(i+1), \overline{j}(i+1)$ zachodzą relacje a,b,d lub e, to z własności W.4.3.6 wynika, że war. 1 jest spełniony. Jeżeli zachodzi relacja c, to z własności W.4.3.4 wynika, że $\{i+1/j-1,1\} \subset D_{Vp}^{i+1}$ i $\{i-1/j,1+1\} \subset D_{Vr}^{i+1}$ ($p \neq r$), a z własności W.4.3.5 wynika, że $D_{Vp}^{i+1} \subset \{C_V \setminus E_{Vt}\}$ lub $D_r^{i+1} \subset \{C_V \setminus E_{Vt}\}$.

Warunek 2

Załóżmy, że $j < n_{\overline{V}1}$.
Jeżeli $\underline{j}(i) \leq j < \overline{j}(i)$, to z własności W.4.3.4 i W.4.3.5 wynika, że $\{i/j+1,1-1\} \subset \{C_V \setminus E_{Vt}\}$.
Załóżmy, że $j = \overline{j}(i)$. Z wniosku 1 z własności W.4.3.2 i W.4.3.3 wynika wtedy, że $j = \overline{j}(i-1)$, co stanowi sprzeczność z założeniem, że $j < n_{\overline{V}1}$.
Zatem przypadek taki nie występuje.

Warunek 3 - dowód jest analogiczny jak dla warunku 2.

2) Układ IH^2 jest układem kontrolnym układu IH^1.

Ponieważ układ IH^1 spełnia warunki określone w tw. 2.5.2, więc $\overline{\mathbb{Z}}_1 = \overline{\mathbb{Z}}_1^- \cup \overline{\mathbb{Z}}_1^+$. Ponieważ $\overline{\mathbb{Z}}_1 = \{C_{0/4} \cup C_{1/4} \cup C_{3/4} \cup C_{4/4}\}$, a układ IH^2 realizujący funkcje (4.3.23) jest STC kodu 2/4, więc stąd wynika, że war. 2 tw. 2.5.3 też jest spełniony.

3) Zbiór $\mathbb{Z}_1$ zawiera wszystkie testy wystarczające do wykrycia uszkodzeń ze zbioru F_p^2.

(Zbiór $\mathbb{T} = \{(1001),(0110),(1010),(0101)\}$, zawiera wszystkie testy wystarczające do wykrycia uszkodzeń układu IH^2 ze zbioru F_p^2. Ze względu na to, że $(\forall V, V = \{A,B\})(\forall u \in \{3,4\})(E_{Vu} \neq \emptyset)$, z (4.3.22) wynika, że $\mathbb{Z}_1 = \{C_{2/4} \setminus \{(0011)\}\}$. Ponieważ $\mathbb{T}_2 \subset \mathbb{Z}_1$, więc war. 3 tw. 2.5.3 również jest spełniony.

Q.E.D.

4.3.4. P r z y k ł a d y

Przykład 4.3.1

Stosując algorytm 4.3.2 zaprojektować STC kodu 3/8.

1. $n_a = n_b = 4$; $A = \{x_1, x_2, x_3, x_4\}$, $B = \{x_5, x_6, x_7, x_8\}$.

2. $k_a = 1$, $k_b = 2$, $n_{a1} = n_{a2} = n_{b1} = n_{b2} = 2$;

$A_1 = \{x_1, x_2\}$, $A_2 = \{x_3, x_4\}$, $B_1 = \{x_5, x_6\}$, $B_2 = \{x_7, x_8\}$.

3. i) V = A

$$C_A = D_A^0 = C_{0/4} \times C_{3/4},$$

$$D_A^0 = \begin{Bmatrix} \{0/1,2\} \\ \{0/2,1\} \end{Bmatrix}.$$

ii) V = B

$$C_B = \{D_B^0 \cup D_B^1\} = \{C_{3/4} \times C_{0/4} \cup C_{2/4} \times C_{1/4}\},$$

$$D_B^0 = \begin{Bmatrix} \{0/1,2\} \\ \{0/2,1\} \end{Bmatrix}, \quad D_B^1 = \begin{Bmatrix} \{1/0,2\} \\ \{1/1,1\} \\ \{1/2,0\} \end{Bmatrix}.$$

4. Wykonujemy algorytm 4.3.1

V = A

1) Ponieważ V = A i $k_a = 1$ wykonujemy krok 2.

2) $E_{A1} = E_{A2} = \emptyset$, $E_{A3} = \{0/1,2\}$, $E_{A4} = \{0/2,1\}$.

V = B

1) Ponieważ V = B przechodzimy do kroku 3.

3) Ponieważ $d_B^0 = 1$ przechodzimy do kroku 5.

5) a) $D_{B1}^0 = \{0/1,2\}$, $D_{B2}^0 = \{0/2,1\}$, $D_{B3}^0 = \emptyset$.

$$D_{B1}^1 = \begin{Bmatrix} \{1/0,2\} \\ \{1/2,0\} \end{Bmatrix}, \quad D_{B2}^1 = \{1/1,1\}, \quad D_{B3}^1 = \emptyset.$$

Kroki b,c,d nie wprowadzają żadnych zmian.

6) $E_{B1} = \emptyset$, $E_{B2} = \{0/1,2\}$,

$E_{B3} = \begin{Bmatrix}\{0/2,1\}\\ \{1/1,1\}\end{Bmatrix}$, $E_{B4} = \begin{Bmatrix}\{1/0,2\}\\ \{1/2,0\}\end{Bmatrix}$.

5. $H_1^1 = T(a \geq 1)$,

$H_2^1 = T(b \geq 2) + T(a_1 \geq 1).T(a_2 \geq 2)$,

$H_3^1 = T(b_1 \geq 1).T(b_2 \geq 2)+T(a_1 \geq 2).T(a_2 \geq 1)+$

$+ T(b \geq 1).T(a_1 \geq 1).T(a_2 \geq 1)$,

$H_4^1 = T(b_1 \geq 2).T(b_2 \geq 1)+T(b_2 \geq 2).T(b \geq 1)+$

$+T(a_1 \geq 2).T(b \geq 1)$.

6,7. $H_1^1 = \underset{a}{(x_1+x_2)} + \underset{b}{(x_3+x_4)}$,

$H_2^1 = \underset{c}{(x_5+x_6)}+\underset{d}{(x_5+x_6)}+\underset{e}{(x_7+x_8)}+\underset{f}{(x_7x_8)}+\underset{a}{(x_1+x_2)}\underset{g}{(x_3x_4)}$,

$H_3^1 = \underset{c}{(x_5+x_6)}\underset{f}{(x_7x_8)}+\underset{h}{(x_1x_2)}\underset{b}{(x_3+x_4)}+ \underset{d}{(x_5+x_6)}+\underset{i}{(x_7+x_8)}$

$\underset{a}{(x_1+x_2)}\underset{b}{(x_3+x_4)}$,

$H_4^1 = \underset{c}{(x_5x_6)}\underset{e}{(x_7+x_8)}+\underset{g}{(x_3x_4)}\ \underset{i}{(x_5+x_6)+(x_7+x_8)} +$

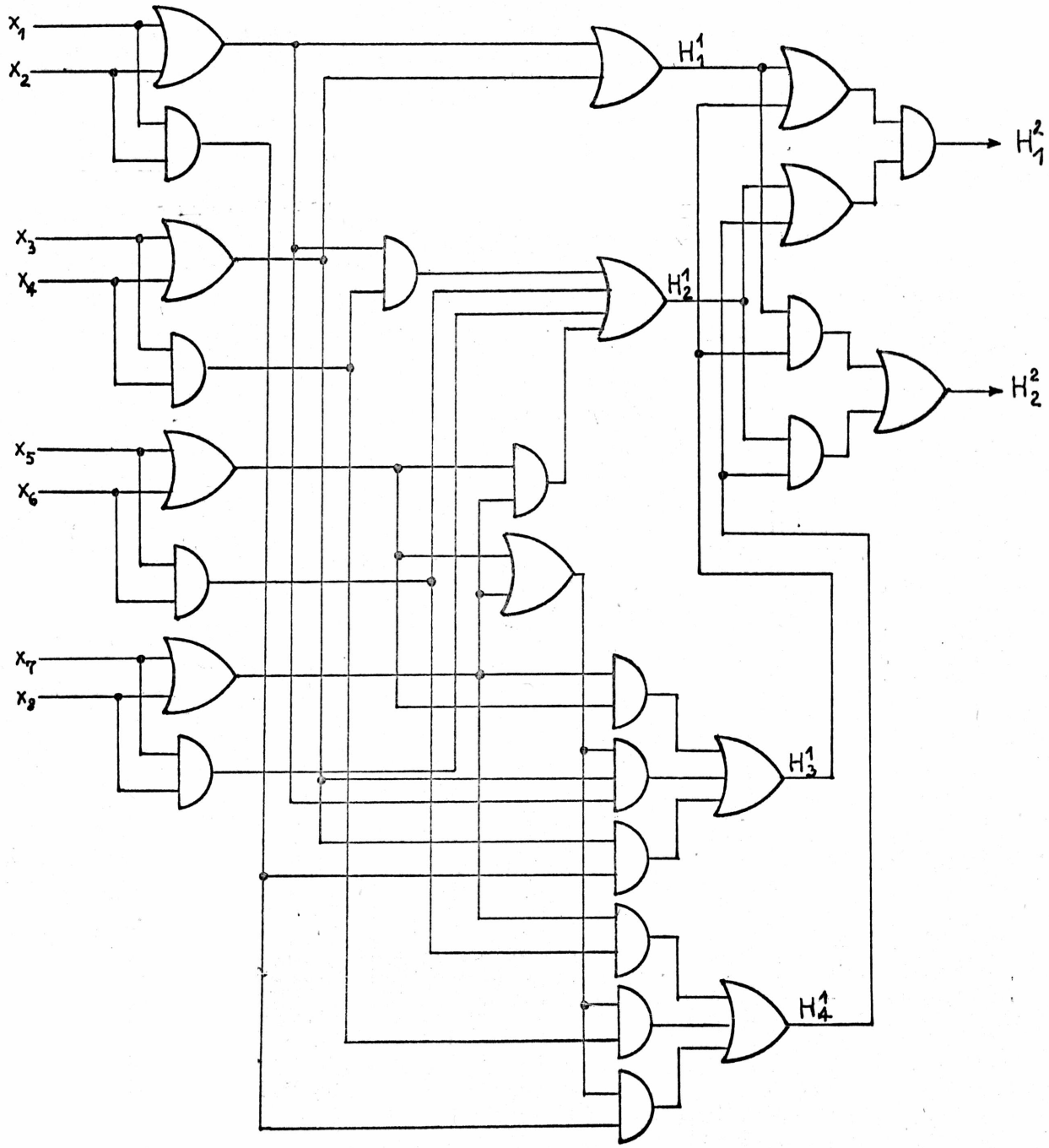
$+\underset{h}{(x_1x_2)}\ \underset{i}{(x_5+x_6)+(x_7+x_8)}$.

Bramki i podukłady stosowane jako wspólne podkreślono i oznaczono literami.

8. $H_1^2 = (H_1^1 + H_3^1)(H_2^1 + H_4^1)$,

$H_2^2 = H_1^1 \cdot H_3^1 + H_2^1 \cdot H_4^1$.

Schemat tak zaprojektowanego układu przedstawia rys.4.3.2.



Rys.4.3.2. Schemat STC kodu 3/8.

Przykład 4.3.2

Stosując algorytm 4.3.2 zaprojektować STC kodu 4/9.

$n_a = 4,\ n_b = 5;$

$A = \{x_1, x_2, x_3, x_4\},\ B = \{x_5, x_6, x_7, x_8, x_9\};$

$k_a = k_b = 2,\ n_{a1} = n_{a2} = n_{b1} = 2,\ n_{b2} = 3;$

$A_1 = \{x_1, x_2\},\ A_2 = \{x_3, x_4\},\ B_1 = \{x_5, x_6\},\ B_2 = \{x_7, x_8, x_9\};$

Podajemy tylko funkcje układu $\mathbb{H}^1$ w postaci ogólnej.

$$H_1^1 = T(a \geq 2) + T(b_1 \geq 1) \cdot T(b_2 \geq 3),$$

$$H_2^1 = T(b \geq 2),$$

$$H_3^1 = T(b_1 \geq 2) \cdot T(b_2 \geq 2) + T(a \geq 1) \cdot T(b_1 \geq 1) \cdot T(b_2 \geq 2) +$$
$$+ T(a_1 \geq 1) \cdot T(a_2 \geq 2) \cdot T(b \geq 1) + T(a_1 \geq 2) \cdot T(a_2 \geq 1) \cdot T(b \geq 1),$$

$$H_4^1 = T(a \geq 1) \cdot T(b_2 \geq 3) + T(a \geq 1) \cdot T(b_1 \geq 2) \cdot T(b_2 \geq 1) +$$
$$+ T(a_1 \geq 2) \cdot T(a_2 \geq 2).$$

4.3.5. Wyznaczanie parametrów STC

Rezultaty zamieszczone w tym podrozdziale dotyczą STC (układów $\mathbb{H}$) zaprojektowanych wg algorytmu 4.3.2, w których (zresztą zgodnie z algorytmem) w maksymalnym stopniu wykorzystano możliwość stosowania wspólnych podukładów. Oszacowania liczby bramek i liczby wejść bramek oraz zbiory testów podajemy kolejno dla układów $\mathbb{H}^1$ i $\mathbb{H}^2$. Wzory na ww. parametry układu oraz na liczbę poziomów układu $\mathbb{H}$ podano na końcu tego podrozdziału.

Układ IH^1

W układzie IH^1 realizującym funkcje (4.3.10) wyróżniamy następujące typy podukładów:

1) układy progowe realizujące funkcje $T(v \geq k_v)$, $v = \{a,b\}$,
2) układy realizujące funkcje $T(v \geq i).T(\overline{v}_1 \geq j).T(\overline{v}_2 \geq 1)$, $v = \{a,b\}$, $i \in \{\overline{0,k_v-1}\}$, $j \in \{\underline{j}(i), \overline{j}(i)\}$, $1=m-i-j$,
3) bramki OR realizujące sumy wyjściowe w funkcjach H_u^1, $u \in \{\overline{1,4}\}$.

Dla każdego z tych typów podukładów podamy teraz wzory, z których wyznacza się liczbę bramek i liczbę wejść bramek oraz zbiory testów wystarczające do wykrycia wszystkich uszkodzeń pojedynczych typu s/z ($z = \{0,1\}$) wraz z oceną ich liczności.

Ad.1. A. Liczba bramek i liczba wejść bramek

$$Br_1 = \sum_{v=\{a,b\}} Br(T_{k_v}^{n_v}), \tag{4.3.24}$$

$$We_1 = \sum_{v=\{a,b\}} We(T_{k_v}^{n_v}). \tag{4.3.25}$$

Parametry $Br(T_{k_v}^{n_v})$, $We(T_{k_v}^{n_v})$ wyznacza się ze wzorów podanych w podrozdziale 3.4 i/lub z tabeli 3.5.1.

B. Zbiór testów

W podrozdz. 3.4 określono następujący zbiór testów wystarczających do wykrycia wszystkich uszkodzeń pojedynczych typu s/z ($z = \{0,1\}$) w układzie progowym realizującym funkcję $T(v \geq k_v)$:

1) zbiory $\mathbb{T}_{k_v}$, $\mathbb{T}_{k_v-1}$ liczące n_v słów kodów k_v/n oraz $(k_v-1)/n_v$, i takich że na kolejnych k_v (lub k_v-1) pozycjach każdego słowa cyklicznie występują jedynki (pozycje o indeksach 1 i n_v uważa się za kolejne),
2) zbiór $\mathbb{T}_{c_v}$ obejmujący c_v dodatkowych słów kodu k_v/n_v, których sposób wyznaczania omówiono w podrozdz. 3.4, a przykładowe zbiory takich testów podano w tabeli 3.4.1.

Niech $X = (X_a X_b)$ oznacza słowo utworzone przez złożenie słów X_a, X_b.
W zbiorze $C_{m/n}$ wyróżniamy trzy podzbiory.

Zbiór $\mathbb{T}^1$, który spełnia warunki:

1) $(\forall X_a \in \{\mathbb{T}_{k_a} \cup \mathbb{T}_{c_a}\})(\exists X \in \mathbb{T}^1 | X = (X_a X_r))$,

2) $(\forall X_b \in \{\mathbb{T}_{k_b} \cup \mathbb{T}_{c_b}\})(\exists X \in \mathbb{T}^1 | X = (X_s X_b))$,

oczywiście $X_r \in C_{k_b/n_b}$, $X_s \in C_{k_a/n_a}$.

3) $|\mathbb{T}^1| = \max \{|\mathbb{T}_{k_b}| + |\mathbb{T}_{c_b}|, v = \{a,b\}\}$.

Uwaga: Z zasady wyznaczania parametrów k_v, n_v oraz danych zamieszczonych w p.3.4 wynika, że

$$|\mathbb{T}^1| = |\mathbb{T}_{k_b}| + |\mathbb{T}_{c_b}| ;$$

przyjmujemy zatem, że

$$|\mathbb{T}^1| = |\mathbb{T}(T_{k_b}^{n_b})| = n_b + c_b. \qquad (4.3.26)$$

Zbiór $\mathbb{T}^2$, który spełnia warunki:

a) $(\forall X_a \in \mathbb{T}_{k_a-1})(\exists X \in \mathbb{T}^2 | X = (X_a X_r))$; oczywiście

$X_r \in C_{(k_b+1)/n_b}$;

2) $(\forall \{k_a-1/j,1\} \subset E_A,\ j \in \{\underline{j}(k_a-1), \bar{j}(k_a-1)\})(\{k_a-1/j,1\} \cap \mathbb{T}^2 \neq \emptyset)$; (4.3.27)

3) $|\mathbb{T}^2| = n_a$.

Zbiór $\mathbb{T}^3$, który spełnia warunki:

1) $(\forall X_b \in \mathbb{T}_{k_b-1})(\exists X \in \mathbb{T}^3 | X = (X_s X_b)$; oczywiście $X_s \in C_{(k_a+1)/n_a}$;

2) $(\forall \{k_b-1/j,1\} \subset E_B, j \in \{\underline{j}(k_b-1), \bar{j}(k_b-1)\})(\{k_b-1/j,1\} \cap \mathbb{T}^3\} \neq \emptyset)$;

3) $|\mathbb{T}^3| = n_b$.

Zbiór $\bigcup_{i=1}^{3} \mathbb{T}^i$ jest wystarczającym zbiorem testów dla uszkodzeń s/z ($z=\{0,1\}$) układów progowych realizujących funkcje $T(v \geq k_v)$, $v = \{a,b\}$.

Ad.2.) Ponieważ dla realizacji tych układów i układów progowych realizujących funkcje $T(v \geq k_v)$, $v = \{a,b\}$, stosuje się wspólne podukłady, uwzględnimy wyłącznie te podukłady, które nie występują w ww. układach progowych.
Korzystamy z przedstawienia funkcji $T(v \geq k_v)$ w postaci

$$T(v \geq k_v) = \sum_{p=0}^{k_v} T(v_1 \geq p).T(v_2 \geq k_v - p), \quad v=\{a,b\}. \tag{4.3.29}$$

W układach progowych realizujących funkcje (4.3.29) nie występują następujące bramki:

a) bramki AND realizujące iloczyny

$$T(v \geq i).T(\bar{v}_1 \geq j).T(\bar{v}_2 \geq 1), \quad v = \{a,b\}; \tag{4.3.30}$$

b) bramki OR (realizujące sumy wyjściowe oraz bramki AND (realizujące iloczyny funkcji $T(v_1 \geq r), T(v_2 \geq i-r)$) w podukładach realizujących funkcje:

$$T(v \geq i) = \sum_{r=0}^{i} T(v_1 \geq r).T(v_2 \geq i-r), \quad i \in \{\overline{1, k_v - 1}\}; \tag{4.3.31}$$

c) bramki AND realizujące funkcje progowe:

$$T(\bar{v}_1 \geq j) = T(\bar{v}_{1,1} \geq j_1).T(\bar{v}_{1,2} \geq j_2),$$

$$T(\bar{v}_2 \geq 1) = T(\bar{v}_{2,1} \geq l_1).T(\bar{v}_{2,2} \geq l_2),$$

gdzie: $j_1 = \left\lfloor \frac{j}{2} \right\rfloor$, $j_2 = \left\lceil \frac{j}{2} \right\rceil$, $l_1 = \left\lfloor \frac{1}{2} \right\rfloor$, $l_2 = \left\lceil \frac{1}{2} \right\rceil$,

a funkcja $T(\bar{v}_{1,1} \geq j_1)$ jest iloczynem $\left\lfloor \frac{n_{\bar{v}1}}{2} \right\rfloor$ zmiennych wejściowych o najmniejszych indeksach i należących do zbioru $\bar{V}1$; podobnie określa się pozostałe trzy funkcje.

Rozpatrywane tutaj funkcje $T(\bar{v}_1 \geq j), T(\bar{v}_2 \geq 1)$ występują w iloczynie (4.3.30) wtedy, gdy odpowiednio $k_{\bar{v}} < j \leq n_{\bar{v}1}$, $k_{\bar{v}} < 1 \leq n_{\bar{v}2}$.

Ad.a)

A. Liczba bramek i liczba wejść bramek

W zależności od tego, czy któryś z parametrów i,j,l przyjmuje wartość 0 bramka AND jest 3-wejściowa (i,j,l≠0), 2--wejściowa (dokładnie jeden z parametrów i,j,l jest równy 0) lub nie występuje (dwa z parametrów i,j,l są równe 0). Liczba bramek i liczba wejść tych bramek podlegają następującym oszacowaniom:

$$Br_2 < \sum_{v=\{a,b\}} \sum_{i=0}^{k_v-1} d_v^i, \tag{4.3.32}$$

$$We_2 < 3.Br_2 - (d_A^0 + d_B^0). \tag{4.3.33}$$

B. Zbiór testów

Tworzymy zbiór

$$\mathbb{T}^4 = \{X \in C_{m/n} | (\forall \{i/j,l\} \subset \{E_A \cup E_B\},\ i \in$$

$$\in \{\overline{0,k_v-2}\},\ j \in \{\underline{j}(i), \bar{j}(i)\})(\exists! X \in \{i/j,l\})$$

taki, że w każdym słowie $X \in \mathbb{T}^4$ jedynki występują na pierwszych i,j,l pozycjach odpowiednio ze zbiorów V, $\bar{V}1$, $\bar{V}2$.

$$|\mathbb{T}^4| = t_a + t_b, \tag{4.3.34}$$

gdzie: jeżeli $k_a=1$ to $t_a=0$

jeżeli $k_v \geq 2$ to $t_v = \sum_{i=0}^{k_v-2} d_v^i$, $v = \{a,b\}$.

Wszystkie uszkodzenia typu s/0 rozpatrywanych bramek AND są wykrywane przez testy ze zbioru $\bigcup_{i=2}^{4} \mathbb{T}^i$.

Z tw.4.3.3 (w którym wykazano m.in., że układ zaprojektowany wg algorytmu 4.3.2 spełnia warunki określone w tw. 4.3.1) oraz z tw. 4.3.1 wynika, że zbiór $\bigcup_{i=1}^{4} \mathbb{T}^i$ wystarcza do wykrycia wszystkich uszkodzeń typu s/1 rozważanych bramek AND.

Ad.b)

A.Liczba bramek i liczba wejść bramek

Bramki OR:

$$Br_3 = m-2, \tag{4.3.35}$$

$$We_3 = \sum_{v=\{a,b\}} \left(\frac{1}{2}.k_v.(k_v+1)-1\right). \tag{4.3.36}$$

Bramki AND:

$$Br_4 = Br_{4a} + Br_{4b}, \tag{4.3.37}$$

gdzie $Br_{4v} = \begin{cases} 0, & \text{jeżeli } k_v<3, \\ \frac{1}{2}.k_v.(k_v-3)+1, & \text{jeżeli } k_v\geq 3; \end{cases}$

$$We_4 = 2.Br_4. \tag{4.3.38}$$

B.Zbiór testów

Niech $\{r,i-r/j,1\}$ oznacza podzbiór zbioru $\{i/j,1\}$ taki, że na pozycjach ze zbiorów V1, V2 występuje odpowiednio po r oraz i-r jedynek. Tworzymy zbiory $\mathbb{T}^5, \mathbb{T}^6$ o następujących własnościach:

1a) $\mathbb{T}^5 = \mathbb{T}^5_a \cup \mathbb{T}^5_b$, 4.3.39

gdzie: i) $(k_v=1 \vee i=0) \Rightarrow (\mathbb{T}^5_v=\emptyset)$,

ii) $(k_v\geq 2 \wedge i\geq 1) \Rightarrow (\mathbb{T}^5_v =$

$$= \{X \in C_{m/n} \mid (\forall \{i/j,1\} \subset E_{Vt},\ t \in \overline{\{1,4\}},$$

$$i \in \overline{\{1,k_v-1\}})(\exists! X \in \{r,i-r/j,1\}, r \in \overline{\{0,i-1\}}).$$

1b) Niech $\{i/j,1\} \subset E_{Vt}$.

$$\mathbb{T}^6 = \bigcup_{v=\{a,b\}} \bigcup_{i=0}^{k_v-2} \mathbb{T}^6_{v_i}, \tag{4.3.40}$$

gdzie każdy zbiór $\mathbb{T}^6_{v_i}$ tworzymy następująco:

i) $(k_v<3 \vee i<2) \Rightarrow (\mathbb{T}^6_{v_i} = \emptyset)$,

ii) $(k_v \geq 3 \wedge \{i-1/j+1,1\} \subset \{E_V \setminus E_{Vt}\}) \Rightarrow$

$$\left(T^6_{v_i} = \left\{X \in C_{m/n} \mid (\forall r \in \{\overline{0,i-2}\})(\exists! X \in \right.\right.$$

$$\left.\left. \in \{r,i-r-1/j+1,1\})\right\}\right),$$

iii) $(k_v \geq 3 \wedge \{i-1/j+1,1\} \subset E_{Vt}) \Rightarrow \left(T^6_{v_i} = \{X \in \right.$

$$\left. \in C_{m/n} \mid (\forall r \in \{\overline{0,i-2}\})(\exists! X \in \{r,i-r-1/j,1+1\})\right) .$$

2) W każdym słowie $X \in T^5$ (lub T^6) jedynki występują na pierwszych: r pozycjach ze zbioru V1, i-r lub i-r-1 (odpowiednio - jeżeli $X \in T^5$ lub $X \in T^6$) pozycjach ze zbioru V2, j+1 pozycjach ze zbioru $\overline{V1}$, l pozycjach ze zbioru $\overline{V2}$.

3) $$\left|T^5 \cup T^6\right| = \sum_{v=\{a,b\}} (k_v-1)^2 . \qquad (4.3.41)$$

Większość uszkodzeń typu s/0 (s/1) rozpatrywanych tutaj bramek OR i AND jest wykonywana przez testy ze zbioru T^5 (T^6). Pozostałe uszkodzenia typu s/0 wykrywają testy X takie, że $X \in \{i,0/j,1\} \in T^4$ (jeżeli $i \in \{\overline{1,k_v-2}\}$) oraz $X \in \{\{i,0/j,1\} \wedge \{T^2 \cup T^3\}\}$ (jeżeli $i = k_v-1$), a typu s/1 - testy X takie, że $X \in \left(\{\{i-1,0/j+1,1\} \cup \right.$ $\left. \cup \{i-1,0/j,1+1\}\} \wedge \{T^2 \cup T^3\}\right)$.

Ad.c)

A. Liczba bramek i liczba wejść bramek

$$Br_5 = \sum_{v=a,b} \sum_{u=1,2} Br_{5v_u}, \qquad (4.3.42)$$

gdzie: $$Br_{5v_u} = \begin{cases} 0, & \text{jeżeli } k_v > n_{v_u}, \\ n_{v_u}-k_v, & \text{jeżeli } k_v < n_{v_u}; \end{cases} \qquad (4.3.43)$$

$$We_5 = 2 \cdot Br_5 .$$

B. Zbiór testów

Załóżmy, że $\{i/j,1\} \subset E_{Vt}$.

Każde uszkodzenie typu s/0 bramki realizującej funkcję $T(\bar{v}_1 \geq j)$ lub $T(\bar{v}_2 \geq 1)$ jest wykrywane przez test X taki, że $X \in \{i/j,1\} \cap \{\mathbb{T}^2 \cup \mathbb{T}^3 \cup \mathbb{T}^4\}$.

Niech $\{i/(j_1, j_2-1),1\}$ oznacza podzbiór zbioru $\{i/j,1\}$ taki, że na pierwszych $\lfloor \frac{n\bar{v}1}{2} \rfloor$ pozycjach ze zbioru $\bar{V}1$ występuje j_1 jedynek, a na pozostałych $\lceil \frac{n\bar{v}1}{2} \rceil$ pozycjach j_2-1 jedynki.

Uszkodzenie typu s/1 wejścia bramki AND, w którym występuje sygnał $T(\bar{v}_{1,2} \geq j_2)$ jest wykrywane przez test X taki, że

i) $(\{i+1/(j_1, j_2-1),1\} \subset \{E_V \setminus E_{Vt}\}) \Rightarrow$

$$\Rightarrow (X \in \{i+1/(j_1, j_2-1),1\} \cap \{\bigcup_{r=1}^{4} \mathbb{T}^r\}),$$

ii) $(\{i+1/(j_1, j_2-1),1\} \subset E_{Vt}) \Rightarrow$

$$\Rightarrow (X \in \{i/(j_1, j_2-1),1+1\} \cap \{\bigcup_{r=2}^{3} \mathbb{T}^r\}).$$

Uszkodzenie typu s/1 wejścia bramki AND, w którym występuje sygnał $T(\bar{v}_{1,1} \geq j_1)$ jest wykrywalne przez test X należący do nowo tworzonego zbioru $\mathbb{T}^7$ takiego, że:

i) $(\{i+1/(j_1-1, j_2),1\} \subset \{E_V \setminus E_{Vt}\}) \Rightarrow$

$$\Rightarrow (\exists! X \in \{\{i+1/(j_1-1, j_2),1\} \cap \mathbb{T}^7),$$

ii) $(\{i+1/(j_1-1, j_2),1\} \subset E_{Vt}) \Rightarrow$

$$\Rightarrow (\exists! X \in \{\{i/(j_1-1, j_2),1+1\} \cap \mathbb{T}^7).$$

Analogicznie wyznacza się testy dla bramek AND realizujących funkcję $T(v_2 \geq 1)$. Warunki nakładane na test X wynikają z twierdzenia 4.3.1.

$$|\mathbb{T}^7| = Br_5. \qquad (4.3.44)$$

Ad.3)

Jeżeli wyjściowe bramki OR układów progowych $T(v \geq k_v)$ uzupełnimy o dodatkowe wejścia, do których dołączymy wyjścia AND-ów realizujących iloczyn (4.3.30) (takich, że $\{i/j,1\} \subset \{E_{A1} \cup E_{B2}\}$), to liczby dodatkowych bramek i wejść bramek wynoszą:

$$Br_6 = 2, \tag{4.3.45}$$

$$We_6 = Br_2. \tag{4.3.46}$$

Pomijamy dowód, że testy ze zbioru $\bigcup_{r=1}^{4} \mathbb{T}^r$ wystarczają do wykrycia wszystkich uszkodzeń typu s/z tych bramek.

Układ $\mathbb{H}^2$

Liczba bramek i liczba wejść bramek układu $\mathbb{H}^2$ wynoszą:

$$Br_r = 6, \tag{4.3.47}$$

$$We_r = 12. \tag{4.3.48}$$

Pomijamy dowód, że testy ze zbioru $\bigcup_{r=1}^{4} \mathbb{T}^r$ wystarczają do wykrycia wszystkich uszkodzeń typu s/z tych bramek.

Parametry układu $\mathbb{H}$ - STC kodu m/n - podsumowanie

Oszacowania liczby bramek, liczby wejść bramek, wyrażają się ogólnymi wzorami

$Br < \sum_{i=1}^{7} Br_i$, $We < \sum_{i=1}^{7} Br_i$, które po uproszczeniu przyjmują postać:

$$Br < Br_1 + Br_2 + Br_4 + Br_5 + m + 6, \tag{4.3.49}$$

gdzie: Br_p, $p = \{1,2,4,5\}$ wyznacza się ze wzorów (4.3.24), (4.3.32), (4.3.37), (4.3.42).

$$We < We_1 + We_3 + 4.Br_2 + 2.Br_4 + 2.Br_5 - (d_A^0 + d_B^0) + 12, \tag{4.3.50}$$

gdzie: We_1, We_3 wyznacza się ze wzorów (4.3.25), (4.3.36), a d_A^0, d_B^0 - ze wzoru (4.3.13).

Zbiór $\mathbb{T}$ testów wystarczających do wykrycia wszystkich uszko-

dzeń typu s/z (z={0,1}) występujących w STC jest sumą zbiorów

$$\mathbb{T} = \bigcup_{i=1}^{7} \mathbb{T}^i.$$

Po dokonaniu uproszczeń wystarczającą liczbę testów określa wzór

$$|\mathbb{T}| = n + n_b + c_b + (k_a-1)^2 + (4_b-1)^2 + |\mathbb{T}^4| + Br_5, \qquad (4.3.51)$$

gdzie: c_b wyznacza się w sposób podany w podrozdz.3.4 (np. z tabeli 3.4.1), $|\mathbb{T}^4|$ określa wzór (4.3.34), a Br_5 - wzór (4.3.42).

Uwaga: pomimo, że istnieją przesłanki ku temu, pomijamy problem dalszej minimalizacji wystarczającego zbioru testów.

Liczba poziomów bramek L

Ze wzorów i danych zamieszczonych w podrozdziałach 3.4 i 3.5 wynika, że jeżeli $i \leq \lfloor \frac{n}{2} \rfloor$, to $0 \leq L(T_i^n) - L(T_i^{n-1}) \leq 1$ oraz $0 \leq L(T_i^n) - L(T_{i-1}^n) \leq 1$ (jeżeli $i > \lfloor \frac{n}{2} \rfloor$, to po podstawieniu $n-i, n-(i-1)$ zamiast $i, i-1$, prawdziwe są podobne nierówności). W związku z tym maksymalna liczba poziomów bramek w układzie wynosi

$$L = L\left(T(b \geq k_b-1)\right)+4, \qquad (4.3.52)$$

gdzie $L\left(T(b \geq k_b-1)\right)$ jest liczbą poziomów najbardziej złożonego układu progowego występującego w podukładzie realizującym iloczyn $T(b \geq k_b-1).T(a_1 \geq j).T(a_2 \geq k_a-j+1)$.
Liczbę tę można wyznaczyć ze wzorów podanych w podrozdziale 3.4 lub z tabeli 3.5.1.
Na cztery pozostałe poziomy składają się: bramka AND realizująca ww. iloczyn, wyjściowa bramka OR układu IH^1 i dwa poziomy układu IH^2.

4.3.6. Porównanie parametrów STC zaprojektowanych różnymi metodami

W tabeli 4.3.1 podano parametry STC wybranych kodów m/n zaprojektowanych wg algorytmu 4.3.2 (parametry tych układów wyznaczono ze wzorów (4.3.49)-(4.3.52)) i wg algorytmów Marouf'a-Friedmana [35]. Ponieważ w [35] nie podano wzorów pozwalających oszacować liczbę bramek i połączeń, parametry STC podano dla tych kodów, dla których w [35] zamieszczono wartości liczbowe.

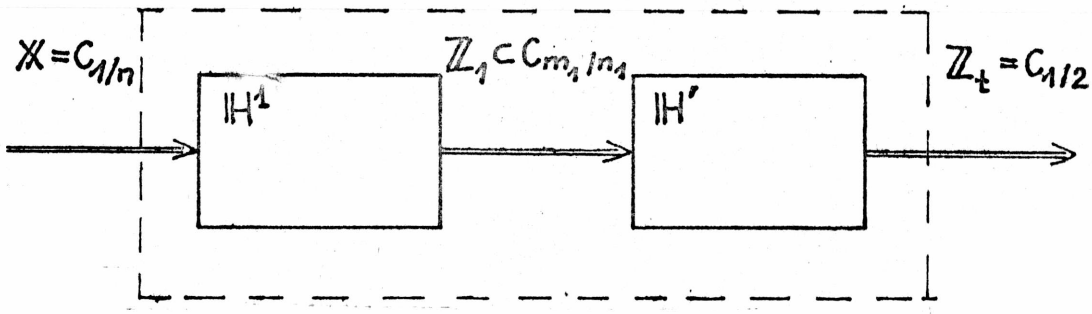
Tabela 4.3.1. Zestawienie parametrów STC kodów m/n zaprojektowanych wg algorytmu 4.3.2 (*) i wg algorytmów z [35]

Kod	Liczba bramek		Liczba wejść bramek		Liczba poziomów		Liczba testów	
	*	[35]	*	[35]	*	[35]	*	[35]
3/7	23	33	50	76	6	7	13	15
3/8	27	40	61	96	6	7	15	20
3/10	34	96	78	203	7	7	19	34
3/12	41	128	95	280	7	7	23	46
4/9	33	61	78	128	7	7	19	31
4/10	38	87	92	228	7	7	21	40
5/11	46	113	112	372	8	7	27	63
5/14	63	260	161	930	9	7	32	166
6/24	156	3471	396	17714	10	7	72	2922
10/21	152	3094	398	134230	11	7	82	2042

Z danych zamieszczonych w tabeli 4.3.1 wynika, że dla wszystkich wyróżnionych kodów m/n STC zaprojektowane wg algorytmu 4.3.2 są (niekiedy bardzo znacznie) prostsze i łatwiej testowalne niż analogiczne układy zaprojektowane wg metod z [35] (najlepszych z dotychczas znanych autorowi metod). Jedynie liczba poziomów STC kodów m/n takich, że $n \geq 11$, jest większa niż w układach z [35]. Można przypuszczać, że podobne rezultaty otrzymuje się dla pozostałych kodów m/n takich, że $m \geq 3$ i $(2.m+1) \leq n \leq 4.m$.

4.4. Kody 1/n, n ≥ 7

STC kodów 1/n, zarówno te znane z literatury [3], jak i projektowane wg metody podanej w tym podrozdziale, mają strukturę taką jak na rys. 4.4.1.



Rys. 4.4.1. Ogólny schemat STC kodu 1/n

Układ H^1 jest samotestowalnym kodowo-rozłącznym translatorem kodu 1/n w kod m_1/n_1 i składa się z jednego poziomu n_1 bramek OR o łącznej liczbie wejść $m_1.n$ (na podstawie założenia przyjętego w podrozdz. 1.2 pomijamy możliwość wystąpienia ograniczeń liczby wejść bramek). Układ H' jest STC kodu m_1/n_1. Na wyjściu układu H^1 występuje zbiór Z_1 n słów kodu m_1/n_1 taki, że $T' \subset Z_1$, gdzie T' oznacza zbiór testów wystarczający do wykrycia wszystkich uszkodzeń pojedynczych typu s/z układu H'. Jeżeli układ H' zrealizuje się jako bezinwersyjny, to nietrudno zauważyć, że są spełnione wszystkie warunki określone w tw. 2.5.3 (konieczne i wystarczające, aby układ H był STC kodu 1/n).

Strukturę STC kodów 1/n przedstawioną na rys. 4.4.1 podano w [3]. Jako kod m_1/n_1 stosowano kod $m_1/2.m_1$ (dla kodów n/2.m w [3] podano metodę projektowania STC) taki, że

$$\binom{2.(m_1-1)}{m_1-1} < n \leq \binom{2.m_1}{m_1} . \qquad (4.4.1)$$

Ponieważ w [48] podano metodę projektowania STC kodów m/2.m o najmniejszej złożoności, za najlepsze rozwiązanie STC kodów 1/n znane dotychczas przyjmujemy STC o strukturze z [3] i układzie H' zaprojektowanym wg metody z [48].

Algorytmy podane w podrozdz. 4.2 i 4.3 pozwalają zaprojektować STC wyróżnionych klas kodów m/n ($m \geq 2$) o mniejszych rozmiarach i mniejszej liczbie testów niż układy otrzymane z zastosowaniem metod znanych z literatury. Stworzyło to przesłanki dla zastosowania kodów 2/n ($n \geq 5$) i m/n ($m \geq 3$ i $(2.m+1) \leq n \leq 4.m$) jako kodu m_1/n_1 w STC kodu 1/n o strukturze z rys.4.4.1. Istotne rezultaty otrzymano dla $n \geq 7$.

Metoda 4.4.1 - projektowania STC kodów 1/n, $n \geq 7$

1. Dobrać kod m_1/n_1, który spełnia warunki:

a) $m_1 \geq 2$,

b) $2.m_1 < n_1$,

c) $\binom{n_1-1}{n_1} < n \leq \binom{n_1}{m_1}$ i $\binom{n_1}{m_1-1} < n$,

d) $|T'| \leq n$, gdzie T' jest zbiorem testów STC kodu m_1/n_1.

2. Utworzyć zbiór $Z_1 \subset C_{m_1/n_1}$ taki, że $T' \subset Z_1$.

3. Zaprojektować układ H^1.

4. Zaprojektować układ H' stosując algorytm 4.2.1 - jeżeli $m_1=2$, algorytm 4.3.2 - jeżeli $m_1 \geq 3$.

Uwaga: dla $n \geq 22$ istnieje więcej niż jeden kod m_1/n_1 spełniający warunki 1a-1d (4.4.2)-(4.4.5). Pozwala to minimalizować wybrany parametr STC kodu 1/n: liczbę bramek, liczbę połączeń, lub liczbę poziomów, albo wybrać rozwiązanie pośrednie.

Parametry STC kodów 1/n wyznacza się ze wzorów:

$$Br = n_1 + Br', \tag{4.4.2}$$

$$We = m_1.n + We', \tag{4.4.3}$$

$$L = 1 + L', \tag{4.4.4}$$

gdzie Br', We', L' są parametrami układu H', które wyznacza się ze wzorów i tabel podanych w podrozdz. 4.2 i 4.3. Dla STC kodów m/2.m o minimalnej złożoności zaprojektowanych wg metody z [48] parametry te wynoszą:

$$Br' = \begin{cases} 14 - \text{dla kodu } 3/6, \\ 26 - \text{dla kodu } 4/8, \\ 42 - \text{dla kodu } 5/10, \end{cases}$$

$$We' = 2\cdot(Br'+m-2),$$

$$L' = m_1+1.$$

Do wykrycia wszystkich uszkodzeń typu s/z (z = {0,1}) STC kodu 1/n w każdym z omówionych tutaj przypadków potrzeba i wystarcza wszystkie n słów kodu 1/n.

Tabela 4.4.1. Zestawienie parametrów STC kodów 1/n zaprojektowanych wg nowej metody (*) i wg metody z [3] i [48]

n	*				[3,48]			
	m_1/n_1	Br	We	L	m/2.m	Br	We	L
7	2/5	16	36	5	3/6	20	51	4
10	2/5	16	42	5			60	
11	2/6	19	49	5			63	
15	2/6	19	57	5			75	
16	2/7	25	68	6			78	
20	2/7	25	76	6			90	
45	2/10	34	142	8	4/8	34	236	5
	3/8	35	196	7				
46	2/11	40	153	8			240	
	3/8	35	199	7				
70	2/13	46	212	8			336	
	3/9	40	279	8				
71	2/13	46	214	8	5/10	52	342	6
	3/9	40	282	8				
100	2/15	55	286	9			590	
	3/10	44	378	8				

Ze wzorów (4.4.2) i (4.4.3) oraz z danych zamieszczonych w tab. 4.4.1 wynikają następujące wnioski:

1) dla każdego $n \in \{\overline{7,100}\}$ STC kodu 1/n zaprojektowane wg metody z [3] i [48] wnoszą mniejsze opóźnienie,

2) dla każdego $n \in \{\{\overline{7,15}\} \cup \{\overline{21,45}\} \cup \{\overline{71,100}\}\}$ istnieje STC kodu 1/n zaprojektowany wg nowej metody przedstawionej w tym podrozdz., który charakteryzuje się mniejszą liczbą bramek i wejść bramek,

3) dla każdego $n \in \{\overline{7,100}\}$ wszystkie STC kodu 1/n zaprojektowane wg nowej metody charakteryzują się mniejszą liczbą wejść bramek,

4) zastosowanie kodów $2/n_1$ zawsze najbardziej minimalizuje liczbę wejść bramek.

4.5. Kody dualne

Kodem dualnym do kodu m/n jest kod (n-m)/n.

W [35] podano prostą procedurę projektowania STC kodów dualnych, do kodów m/n takich, że $2.m < n$. Procedurę tę podajemy w formie dostosowanej do naszych potrzeb. Rozważamy kody dualne (n-m)/m do następujących klas kodów m/n:

1) $m=1$ i $n \geq 7$,
2) $m=2$ i $n \geq 5$,
3) $m \geq 3$ i $(2.m+1) \leq n \leq 4.m$.

Procedura 4.5.1

1. Dla kodu (n-m)/n dualnego do pierwszego kodu m/n należącego do jednej z trzech ww. klas kodów zaprojektować STC kodu m/n (układ $\mathbb{H}$) stosując odpowiednią z metod podanych w podrozdziałach 4.2-4.4.

2. Wygenerować układ $\mathbb{H}^d$ dualny do układu $\mathbb{H}$ poprzez zastąpienie każdej bramki AND (OR) bramką OR (AND).

4.6. Uwagi końcowe

STC kodów m/n zaprojektowane wg metod podanych w tym rozdziale różnią się od analogicznych układów zaprojektowanych wg metod znanych z literatury następującymi cechami:

1) ogólna struktura STC (dobór kodów kolejnych translatorów);

2) postać minimalna funkcji translatorów (w funkcjach translatorów kodu m/n w kod $1/\binom{n}{m}$ - w STC z [3], 1/Z - w STC z [35], występują wyłącznie iloczyny m zmiennych. Natomiast w funkcjach translatorów kodu m/n ($m \geq 2$) w kod $2/n_1$ oraz kodu $2/n_i$ w kod $2/n_{i+1}$ ($i \in \{\overline{1, t-1}\}$) występujących w STC zaprojektowanych wg metod podanych w tym rozdziale występują iloczyny stopnia niższego od m_{i-1} (przyjmujemy, że $m_0 = m$));

3) zastosowanie wielopoziomowych układów progowych (zaprojektowanych wg nowej metody podanej w rozdz.3).

Z tabel 4.2.4, 4.3.1 i 4.4.1, w których zamieszczono parametry STC kodów m/n zaprojektowanych metodami znanymi z literatury i wg nowych metod przedstawionych w tym rozdziale wynika, że dla wyróżnionych klas kodów tzn.:

a) kodów 1/n ($n \geq 7$),
b) kodów 2/n ($n \geq 5$),
c) kodów m/n ($m \geq 3$ i $(2.m+1) \leq n \leq 4.m$),

STC zaprojektowane odpowiednio wg metody 4.4.1 oraz algorytmów 4.2.1 i 4.3.2 charakteryzują się:

1) mniejszą złożonością mierzoną liczbą bramek (z wyjątkiem niektórych kodów 1/n) i liczbą wejść bramek (z czego wynika, że teza pracy została potwierdzona a cel pracy osiągnięty),

2) lepszą testowalnością mierzoną liczbą testów koniecznych do wykrycia wszystkich uszkodzeń jednokierunkowych (z wyjątkiem STC kodów 1/n, dla których liczba testów nie zmieniła się),

3) na ogół nieco dłuższym czasem propagacji mierzonym liczbą poziomów bramek (z wyjątkiem kodów 2/n oraz 3/7 i 3/8).

Z punktu widzenia dotychczasowych zastosowań kodów m/n (ich przegląd podano w podrozdziale 1.1) najważniejsze są następujące klasy tych kodów:

1) kody m/2.m - z racji największej pojemności kodu przy danej długości słowa i relatywnie prostych STC (w odniesieniu do innych kodów m/n),

2) kody 1/n - z racji ich szerokiego rozpowszechnienia nie tylko w systemach cyfrowych o podwyższonej niezawodności (samosprawdzalnych, bezpiecznych (ang. fail-safe) lub tolerujących uszkodzenia), lecz również w systemach i układach cyfrowych ogólnego zastosowania (np. w dekoderach),

3) kod 2/5 oraz kod złożony z kodów 1/2 i 1/5 - do kodowania cyfr dziesiętnych.

Dzięki wynikom otrzymanym w tym rozdziale, tam gdzie z racji małej złożoności STC znajdowały zastosowanie wyłącznie kody m/2.m, szersze zastosowanie mogą znaleźć kody m/(2.m+1) oraz (m-1)/2.m o pojemnościach pośrednich w stosunku do kodów z klasy m/2.m i mniejszych rozmiarach STC niż STC kodów m/2.m. Z wyników zamieszczonych w podrozdz. 4.5 wynika duża przydatność STC kodów 2/n do projektowania STC kodów 1/n. Ponadto z badań prowadzonych przez autora [40] wynika możliwość zastosowania STC wszystkich klas kodów m/n do projektowania STC kodów Bergera.

Na zakończenie podamy jeszcze kilka uwag nt. możliwości rozszerzania stosowalności algorytmów 4.2.1 i 4.3.2.

1) Algorytm 4.2.1 pozwala zaprojektować dla każdego kodu 2/n, $n \geq 5$, taki STC, że podzbiór n słów kodu 2/n, w których cyklicznie na kolejnych dwóch pozycjach występują jedynki, jest wystarczającym zbiorem testów dla wszystkich uszkodzeń jednokierunkowych układu (można wykazać, że dalsze zmniejszenie liczby testów nie jest możliwe, gdyż konieczne jest aby każda zmienna wejściowa w co najmniej dwóch słowach kodowych przyjmowała wartość 1). Układ taki projektuje się stosując algorytm 4.2.1 dla następujących parametrów:

$$n_i = \left\lceil {}^{n_{i-1}}/_{2} \right\rceil + 1, \quad i \in \{\overline{1, t-1}\},$$

gdzie: t - wymagana liczba stopni układu.

2) Pominięcie kroku 6 (i ewentualnie kroku 7) algorytmu 4.3.2 pozwala zmniejszyć liczbę poziomów STC. W skrajnym przypadku zastosowania realizacji dwupoziomowej AND-OR układu H^1 liczba poziomów jest równa 4. Możliwe do zastosowania są również rozwiązania pośrednie. Oczywiście w każdym przypadku odbywa się to kosztem mniejszych możliwości stosowania wspólnych podukładów, wskutek czego pogarszają się pozostałe trzy parametry układu. Samotestowalność takiego układu wynika z twierdzeń 4.3.1 i 4.3.3.

3) Zastosowanie translacji kodu m/n w kod $2/n_1$ taki, że n_1 5 stwarza przesłanki rozszerzenia algorytmu 4.3.2 na kody m/n takie, że $m \geq 3$ i $n > 4.m$.

5. OCENA NIEZAWODNOŚCI SAMOTESTOWALNYCH UKŁADÓW KONTROLNYCH KODÓW m/n

5.1. Przegląd metod oceny niezawodności układów cyfrowych

Celem analizy niezawodnościowej STC jest ilościowa ocena efektywności projektowania i własności różnych wersji układu kontrolnego tego samego kodu.

Najpowszechniej stosowaną miarą niezawodności układów cyfrowych jest niezawodność funkcjonalna definiowana jako prawdopodobieństwo, że w układzie nie ma uszkodzeń [1]. Jednakże układ cyfrowy może funkcjonować poprawnie (tzn. generuje poprawny sygnał wyjściowy) nawet jeżeli występują w nim uszkodzenia. Tym samym miara ta stanowi na ogół zbyt pesymistyczne oszacowanie niezawodności układu cyfrowego. Jeżeli znane są prawdopodobieństwa występowania uszkodzeń w układzie i prawdopodobieństwa wystąpienia wektorów wejściowych, to możliwe jest wyznaczanie prawdopodobieństwa, że sygnał wyjściowy układu jest poprawny. Ogólne zależności pozwalające wyznaczyć to prawdopodobieństwo podano po raz pierwszy w [1]. W [38] prawdopodobieństwo to nazwano niezawodnością sygnału (ang. signal reliability) i podano dwie metody jego wyznaczania, udoskonalone następnie w [30] i [31]. W przeciwieństwie do niezawodności funkcjonalnej, niezawodność sygnału wyjściowego układu zależy od struktury wewnętrznej połączeń układu, natury możliwych uszkodzeń i ich prawdopodobieństw wystąpienia. Tym samym dostarcza ona bardziej adekwatnego porównania niezawodności różnych realizacji układowych.

W [50] po raz pierwszy podano metody oceny niezawodności układów samosprawdzalnych. W pracy tej zdefiniowano trzy różne miary niezawodności sygnału dla układów samosprawdzalnych i podano procedury ich wyznaczania. Jednakże miary te nie uwzględniają specyfiki działania układów kontrolnych, a wymienione procedury są efektywne jedynie w przypadku najprostszych układów (nie więcej niż 10 bramek).

Wyniki z [50] rozszerzono w [32] na przypadek STC oraz podano znacznie efektywniejsze metody wyznaczania miar niezawodności sygnału. Jednakże ze względu na zbyt dużą złożoność wzorów analitycznych, z których wyznacza się wyszczególnione miary nie-

zawodności, metody te są mało przydatne w zastosowaniu do układów o liczbie wejść $n \geqslant 5$ i takich, które w znacznym stopniu wykorzystują wspólne podukłady. Ponadto wszystkie miary niezawodności układów samosprawdzalnych podane w [32,50] są niezależne od czasu, co utrudnia właściwą ocenę jakości układu pracującego przez dłuższy okres czasu (np. rzędu 1000h). W związku z tym w dalszej części tego rozdziału zaproponowano model niezawodnościowy STC uwzględniający parametr czasu (p. 5.2), zdefiniowano miary niezawodności STC (p.5.3), podano ogólne zasady ich wyznaczania (p.5.4). Wnioski natury ogólniejszej wynikające z załączonych przykładów zastosowań (p.5.5) zamieszczono w p.5.6. Należy nadmienić, że większość rezultatów podanych w tym rozdziale nadaje się, po dokonaniu nieznacznych modyfikacji, do oceny niezawodności STC dowolnych kodów detekcyjnych a nie tylko kodów m/n.

5.2. Model niezawodnościowy STC

Rozważamy układ $\mathbb{H}$ zdefiniowany w podrozdziale 2.1. Zagadnienie poprawności działania układu $\mathbb{H}$ (kryteria poprawnej i błędnej pracy układu $\mathbb{H}$ sformułowano również w podrozdz. 2.1), rozpatrywać będziemy jedynie w dyskretnych chwilach czasu t_l, $l = 1,2,3,\ldots$, wyznaczanych pojawieniem się kolejnych słów na wejściu układu $\mathbb{H}$. Zakładamy, że opóźnienie wnoszone przez układ $\mathbb{H}$ jest pomijalnie małe.
Dla każdej chwili t_l, $l = 1,2,3,\ldots$, i dla każdej pary $[X_i(t_l), f^{*}(t_l)]$ wyjście $Z(t_l)$ układu $\mathbb{H}$ jest jednoznacznie określone

$$Z(t_l) = H[X_i(t_l), f^{*}(t_l)]$$

i oczywiście $Z(t_l) \in \mathbb{Z}_0$.
Parę (X_i, f^{*}), gdzie $X_i \in \mathbb{X}_0$, $f^{*} \in F^{*}$ (f^{*} jest stanem niezawodnościowym układu $\mathbb{H}$) nazywamy <u>stanem niezawodnościowo-funkcjonalnym</u> (stanem n-f) układu $\mathbb{H}$. Natomiast zbiór

$$\mathbb{S} = \mathbb{X}_0 \times F^{*} = \bigcup_{X_i \in \mathbb{X}_0} \bigcup_{f^{*} \in F^{*}} (X_i, f^{*})$$

nazywamy przestrzenią stanów niezawodnościowo-funkcjonalnych układu $\mathbb{H}$.

W przestrzeni $\mathbb{S}$ można wyróżnić pięć podzbiorów rozłącznych:

1) $\mathbb{S}_r = \{(X_i, f^{\mathbb{H}}) \in \mathbb{S} \mid (X_i \in \mathbb{X}) \wedge (f^{\mathbb{H}} = \varphi)\}$ - słowo kodowe na wejściu nieuszkodzonego układu $\mathbb{H}$,

2) $\mathbb{S}_{m_c} = \{(X_i, f^{\mathbb{H}}) \in \mathbb{S} \mid (X_i \in \mathbb{X}) \wedge (f^{\mathbb{H}} \in F) \wedge (H(X_i, f^{\mathbb{H}}) \in \mathbb{Z})\}$ - maskowanie uszkodzenia układu H przez wejściowe słowo kodowe,

3) $\mathbb{S}_{d_c} = \{(X_i, f^{\mathbb{H}}) \in \mathbb{S} \mid (X_i \in \mathbb{X}) \wedge (f^{\mathbb{H}} \in F) \wedge (H(X_i, f^{\mathbb{H}}) \in \overline{\mathbb{Z}})\}$ - wykrycie uszkodzenia układu H przez wejściowe słowo kodowe,

4) $\mathbb{S}_{d_n} = \{(X_i, f^{\mathbb{H}}) \in \mathbb{S} \mid (X_i \in \mathbb{X}) \wedge (f^{\mathbb{H}} \in F) \wedge (H(X_i, f^{\mathbb{H}}) \in \overline{\mathbb{Z}})\}$ - wykrycie uszkodzenia układu H przez wejściowe słowo niekodowe,

5) $\mathbb{S}_{u_n} = \{(X_i, f^{\mathbb{H}}) \in \mathbb{S} \mid (X_i \in \mathbb{X}) \wedge (f^{\mathbb{H}} \in F) \wedge (H(X_i, f^{\mathbb{H}}) \in \mathbb{Z})\}$ - maskowanie wejściowego słowa niekodowego przez uszkodzenie układu $\mathbb{H}$.

Oczywiście $\mathbb{S}_r \cup \mathbb{S}_{m_c} \cup \mathbb{S}_{d_c} \cup \mathbb{S}_{d_n} \cup \mathbb{S}_{u_n} = \mathbb{S}$.

Podstawę dla probabilistycznego opisu działania układu $\mathbb{H}$ stanowią następujące zjawiska losowe:

1) strumień wektorów wejściowych $X_i(t_l)$, $l = 1,2,3,\ldots$ generowanych przez kontrolowany układ $\mathbb{G}$,

2) strumień stanów niezawodnościowych układu $\mathbb{H}$ - $f^{\mathbb{H}}(t_l)$, $l = 1,2,3,\ldots$ (w dalszym ciągu będzie mowa tylko o strumieniu uszkodzeń),

3) strumień wektorów wyjściowych $Z(t_l)$, $l = 1,2,3,\ldots$ generowanych przez układ kontrolny $\mathbb{H}$, będący pochodną tamtych strumieni, określony jednoznacznie przez funkcję logiczną H.

Zbiór $\mathbb{S}$ stanowi zatem przestrzeń zdarzeń elementarnych, a proces funkcjonowania układu $\mathbb{H}$ jest procesem stochastycznym opartym na dyskretnej i skończonej przestrzeni stanów n-f - $\mathbb{S}$. Powyższa analiza umożliwia sformułowanie następujących definicji.

Definicja 5.2.1

Układ $\mathbb{H}$ funkcjonuje niezawodnie w chwili t_l, $l = 1,2,3,\ldots$, jeżeli jest spełniony warunek

$$[X_i(t_1) \in \bar{\mathbb{X}}] \Rightarrow (H[X_i(t_1), f^{*}(t_1)] \in \bar{\mathbb{Z}}). \qquad (5.2.1)$$

Oczywiście $[X_i(t_1), f^{*}(t_1)] \in \{ \mathbb{S} \setminus \mathbb{S}_{u_n} \}$.

Definicja 5.2.2

Układ H funkcjonuje zawodnie w chwili t_1, $1 = 1,2,3,\ldots$, jeżeli jest spełniony warunek

$$[X_i(t_1) \in \bar{\mathbb{X}}] \Rightarrow (H[X_i(t_1), f^{*}(t_1)] \in \mathbb{Z}). \qquad (5.2.2)$$

Oczywiście $[X_i(t_1), f^{*}(t_1)] \in \mathbb{S}_{u_n}$.

Definicja 5.2.3

Układ H funkcjonuje niezawodnie w przedziale czasu $[0, t]$, jeżeli dla każdego $t_1 \in [0,t]$, $1 = 1,2,3,\ldots$, jest spełniony warunek

$$\left(X_i(t_1) \in \bar{\mathbb{X}}\right) \Rightarrow \left(H[X_i(t_1), f^{*}(t_1)] \in \bar{\mathbb{Z}}\right). \qquad (5.2.3)$$

Definicja 5.2.4

Układ H funkcjonuje zawodnie w przedziale czasu $[0,t]$, jeżeli jest spełniony warunek

$$(\exists X_i(t_1) \in \bar{\mathbb{X}},\ t_1 \in [0,t] \mid H[X_i(t_1), f^{*}(t_1)] \in \mathbb{Z}). \qquad (5.2.4)$$

Z def. 5.2.3 wynika, że układ H może być w przedziale czasu $[0,t]$ wielokrotnie uszkodzony i naprawiony, a mimo to jego funkcjonowanie uważamy za niezawodne, o ile jest spełniony warunek (5.2.3). Natomiast z def. 5.2.4 wynika, że jeżeli chociaż jedno słowo niekodowe, które wystąpi w przedziale czasu $[0,t]$, nie zostanie wykryte przez układ H, to jego funkcjonowanie w tym przedziale czasu uważamy za zawodne.

Przyjmujemy następujące założenia wstępne:

Z.5.2.1) Układ H jest samotestowalny dla każdego uszkodzenia ze zbioru F,

Z.5.2.2) Odnowa jest natychmiastowa, tzn. czas przeznaczony na odnowę po wystąpieniu sygnału alarmowego z układu H,

jest równy zero,

Z.5.2.3) Do zbioru F należą tylko uszkodzenia trwałe,

Z.5.2.4) Rozkład prawdopodobieństwa s_j wystąpienia każdego uszkodzenia $f_j \in F$ jest zadany funkcją zależną od czasu $s_j(t) = 1 - \exp(-\lambda_j \cdot t)$, gdzie λ_j jest intensywnością uszkodzeń.

5.3. Miary niezawodności STC

Wyróżniamy trzy grupy miar niezawodności STC.

A. Miary zależne wyłącznie od złożoności układu

Definicja 5.3.1

Niezawodność funkcjonalna FR(t) układu kontrolnego w chwili t jest to prawdopodobieństwo, że w chwili t w układzie nie ma uszkodzenia

$$FR(t) = Pr\{f^{\mathbb{M}}(t) = \varphi\}. \tag{5.3.1}$$

Definicja 5.3.2

Niezawodność funkcjonalna $R_F(t)$ układu kontrolnego w przedziale czasu $[0,t]$ jest to prawdopodobieństwo, że w przedziale czasu $[0,t]$ w układzie nie wystąpi żadne uszkodzenie

$$R_F(t) = Pr\{(\forall \tau \in [0,t]) f^{\mathbb{M}}(\tau) = \varphi\}. \tag{5.3.2}$$

Powyższe definicje są adaptacjami znanych z [1] (def. 5.3.1) oraz z [30] (def. 5.3.2) definicji sformułowanych dla dowolnych układów cyfrowych.

Niezawodność funkcjonalna układu rośnie, jeżeli rozmiary układu maleją. Miara ta nie uwzględnia jednak następujących faktów:

1) obecność uszkodzenia w układzie nie jest równoznaczna z błędnym działaniem układu - uszkodzenie może być maskowane dla pewnej liczby wektorów wejściowych,

2) ponieważ rozpatrywany układ jest samotestowalny, więc uszkodzenie zanim przeszkodzi wykryciu słowa niekodowego na wejściu, może zostać wcześniej wykryte i usunięte.

B. Miary charakteryzujące własności struktury logicznej układu

Definicja 5.3.3

Średnia maskowalność uszkodzeń dla wejść kodowych - S_{M_c} - jest to prawdopodobieństwo, że wyjście układu kontrolnego należy do przestrzeni kodowej, gdy układ jest uszkodzony, a wejście jest słowem kodowym

$$S_{M_c} = \Pr\{H(X_i,f_j) \in \mathbb{Z} \mid (f_j \in F) \wedge (X_i \in \mathbb{X})\}. \qquad (5.3.3)$$

Definicja 5.3.4

Średnia wykrywalność uszkodzeń dla wejść kodowych - S_{D_c} - jest to prawdopodobieństwo, że wyjście układu kontrolnego należy do przestrzeni niekodowej, gdy układ jest uszkodzony a wejście jest słowem kodowym

$$S_{D_c} = \Pr\{H(X_i,f_j) \in \overline{\mathbb{Z}} \mid (f_j \in F) \wedge (X_i \in \mathbb{X})\}. \qquad (5.3.4)$$

Definicja 5.3.5

Średnia wykrywalność uszkodzeń dla wejść niekodowych - S_{D_n} - jest to prawdopodobieństwo, że wyjście układu kontrolnego należy do przestrzeni niekodowej, gdy układ jest uszkodzony, a wejście jest słowem niekodowym

$$S_{D_n} = \Pr\{H(X_i,f_j) \in \overline{\mathbb{Z}} \mid (f_j \in F) \wedge (X_i \in \mathbb{X})\}. \qquad (5.3.5)$$

Definicja 5.3.6

Średnia niewykrywalność wejść niekodowych wskutek uszkodzeń wewnętrznych - S_{U_n} - jest to prawdopodobieństwo, że wyjście układu kontrolnego należy do przestrzeni kodowej, gdy układ jest uszkodzony, a wejście jest słowem niekodowym

$$S_{U_n} = \Pr\{H(X_i,f_j) \in \mathbb{Z} \mid (f_j \in F) \wedge (X_i \in \mathbb{X})\}. \qquad (5.3.6)$$

Miary określone definicjami 5.3.3-5.3.6 są analogiczne do miar zdefiniowanych w [32], z tym, że w powyższym przypadku nie zależą one od prawdopodobieństw wystąpienia uszkodzeń $s_j(t)$. Miary te charakteryzują w zasadzie wyłącznie strukturę logiczną układu z uszkodzeniami pod względem jej potencjalnych możliwości maskowania i wykrywalności uszkodzeń układu, oraz wykrywalności i niewykrywalności wejściowych słów niekodowych. Zależą one (jak się to dokładniej okaże w p. 5.4) jednakże również od charakterystyk układu kontrolowanego $\mathbb{G}$, w następujący sposób:

1) rozkład prawdopodobieństwa uszkodzeń układu $\mathbb{G}$ rzutuje na parametry w_c i w_n (odpowiednio prawdopodobieństwo wystąpienia słowa kodowego lub niekodowego na wyjściu układu $\mathbb{G}$ a wejściu układu $\mathbb{H}$),

2) w zależności od typu układu $\mathbb{G}$ zmienia się rozkład prawdopodobieństw warunkowych $Pr\{X_i | \overline{\mathbb{X}}\}$, $Pr\{X_i | \overline{\overline{\mathbb{X}}}\}$, który z kolei rzutuje na maskowalność i wykrywalność uszkodzeń układu $\mathbb{H}$, oraz na maskowalność wejściowych słów niekodowych przez uszkodzenia wewnętrzne układu $\mathbb{H}$.

Za najważniejszą z tych miar należy uznać S_{U_n}, która charakteryzuje podatność danej struktury logicznej układu na zawodne działanie, spowodowane obecnością nie wykrytego uszkodzenia i wystąpieniem wejściowego słowa niekodowego, Wadą tej miary jest to, że nie uwzględnia wpływu rozmiarów układu, a ściślej - wpływu liczby możliwych uszkodzeń (układ większy psuje się częściej) na średnią liczbę nie wykrytych wejściowych słów niekodowych.

W kolejnym p.C zdefiniowano miary niezawodności STC posiadające zalety miar zdefiniowanych dotychczas, a nie mające ich niedostatków.

C. Miary globalne niezawodności STC

Definicja 5.3.7

Niezawodność sygnału wyjściowego SR(t) układu kontrolnego w chwili t jest to prawdopodobieństwo, że jeżeli w chwili t na wejściu układu wystąpi słowo niekodowe, to zostanie wykryte przez układ

$$SR(t) = Pr\{(X_i(t) \in \overline{\mathbb{X}}) \Rightarrow (H[X_i(t), f^*(t)] \in \overline{\mathbb{Z}})\}. \qquad (5.3.7)$$

Definicja 5.3.8

Niezawodność sygnału wyjściowego $R_S(t)$ układu kontrolnego w przedziale czasu $[0,t]$ jest to prawdopodobieństwo, że każde wejściowe słowo niekodowe, które wystąpi w przedziale czasu $[0,t]$ zostanie wykryte przez układ

$$R_S(t) = Pr\left\{\left((X_i(t_1) \in \overline{X}) \wedge (t_1 \in [0,t])\right) \Rightarrow \left(H[X_i(t_1), f^*(1)] \in \overline{Z}\right)\right\}. \tag{5.3.8}$$

Powyższe definicje są dostosowanymi do specyfiki działania STC adaptacjami znanych z [30,38] (def.5.3.7) oraz z [38] (def. 5.3.8) definicji sformułowanych dla dowolnych układów cyfrowych. Miara $R_S(t)$ stanowi podstawę do wyznaczania tzw. czasu T_m (ang. mission time), wprowadzonego w [8] do oceny niezawodności systemów tolerujących uszkodzenia.

Definicja 5.3.9

Czas T_m jest to czas, po upływie którego niezawodność sygnału wyjściowego STC spada do pewnego poziomu minimalnego $R_{Smin} = R_S(T_m)$.

Czas T_m umożliwia ocenę ile razy dłużej dany układ prcuje na poziomie niezawodności sygnału nie niższym od R_{Smin}, niż inny układ realizujący tą samą funkcję.

Między miarami FR(t), $R_F(t)$, SR(t) i $R_S(t)$ w rozpatrywanym tutaj przypadku dla każdego t zachodzi relacja [30]

$$SR(t) \geqslant R_S(t) \geqslant R_F(t) \geqslant FR(t).$$

W następnym podrozdziale (p.5.4) wykażemy, że miary podane w p.C umożliwiają globalną ocenę niezawodności STC, gdyż uwzględniają zarówno wpływ rozmiarów układu (mierzonych liczbą uszkodzeń) jak i własności jego struktury logicznej (głównie zdolność maskowania wejściowych słów niekodowych przez uszkodzenia wewnętrzne układu).

5.4. Metody wyznaczania miar niezawodności STC

Oprócz obowiązujących nadal założeń Z.5.2.1-Z.5.2.4 wprowadzamy ponadto następujące założenia:

Z.5.4.1) w układzie $|H$ występują dwa parami niezależne strumienie zdarzeń losowych: pojedynczy strumień wektorów wejściowych i pojedynczy strumień uszkodzeń,

Z.5.4.2) strumień uszkodzeń trwałych jest złożonym procesem Poissona określonym przez zbiór intensywności uszkodzeń przyjmujących stałe wartości $\{\lambda_j\}$,

Z.5.4.3) strumień wektorów wejściowych jest stacjonarnym procesem dwumianowym, określonym przez parę prawdopodobieństw: $w_c = Pr\{X_i \in \mathbb{X}\}, w_n = Pr\{X_i \in \overline{\mathbb{X}}\}$; $w_c + w_n = 1$,

Z.5.4.4) $(\forall X_i \in \mathbb{X})\ Pr\{X_i\} = \frac{w_c}{|\mathbb{X}|}$,

Z.5.4.5) $(\forall X_i \in \overline{\mathbb{X}})\ Pr\{X_i\} = \frac{w_n}{|\overline{\mathbb{X}}|}$,

Z.5.4.6) czas między pojawieniem się dwóch kolejnych wektorów wejściowych jest pomijalnie mały w stosunku do czasu między dwoma kolejnymi uszkodzeniami,

Z.5.4.7) czas obecności uszkodzenia trwałego w układzie jest pomijalnie mały w stosunku do czasu między uszkodzeniami.

A. Niezawodność funkcjonalną określa wzór:

$$FR(t) = R_F(t) = \exp\left(-\sum_{f_j \in F} \lambda_j . t\right), \qquad (5.4.1)$$

a jeżeli $(\forall f_j \in F)\ (\lambda_j = \lambda)$

$$FR(t) = R_F(t) = \exp\left(-|F| \cdot \lambda \cdot t\right). \qquad (5.4.2)$$

B. Z założenia Z.5.4.1 wynika, że

$$Pr\{(X_i, f_j)\} = Pr\{X_i\} \cdot Pr\{f_j\}. \qquad (5.4.3)$$

Zatem:

$$S_{M_c} = \sum_{f_j \in F} \left(\Pr\{f_j \mid F\} \cdot \left(\sum_{X_i \in \mathbb{X}} \Pr\{X_i \mid \mathbb{X}\} \cdot \operatorname{sgn}\{H(X_i, f_j) \in \mathbb{Z}\} \right) \right), \quad (5.4.4)$$

$$S_{D_c} = \sum_{f_j \in F} \left(\Pr\{f_j \mid F\} \cdot \left(\sum_{X_i \in \mathbb{X}} \Pr\{X_i \mid \mathbb{X}\} \cdot \operatorname{sgn}\{H(X_i, f_j) \in \overline{\mathbb{Z}}\} \right) \right), \quad (5.4.5)$$

$$S_{D_n} = \sum_{f_j \in F} \left(\Pr\{f_j \mid F\} \cdot \left(\sum_{X_i \in \overline{\mathbb{X}}} \Pr\{X_i \mid \overline{\mathbb{X}}\} \cdot \operatorname{sgn}\{H(X_i, f_j) \in \overline{\mathbb{Z}}\} \right) \right), \quad (5.4.6)$$

$$S_{U_n} = \sum_{f_j \in F} \left(\Pr\{f_j \mid F\} \cdot \left(\sum_{X_i \in \overline{\mathbb{X}}} \Pr\{X_i \mid \overline{\mathbb{X}}\} \cdot \operatorname{sgn}\{H(X_i, f_j) \in \mathbb{Z}\} \right) \right), \quad (5.4.7)$$

gdzie $\operatorname{sgn}\{\rtimes\} \stackrel{df}{=} \begin{cases} 0, & \text{jeżeli relacja "}\rtimes\text{" jest fałszywa,} \\ 1, & \text{jeżeli relacja "}\rtimes\text{" jest prawdziwa.} \end{cases}$

Z założeń Z.5.4.5 i Z.5.4.6 wynika, że

$$(\forall X_i \in \mathbb{X}) \left(\Pr\{X_i \mid \mathbb{X}\} = \frac{1}{|\mathbb{X}|} \right) \quad (5.4.8)$$

i

$$(\forall X_i \in \overline{\mathbb{X}}) \left(\Pr\{X_i \mid \overline{\mathbb{X}}\} = \frac{1}{|\overline{\mathbb{X}}|} \right) . \quad (5.4.9)$$

Jeżeli przyjmiemy, że $(\forall f_j \in F)\ (\lambda_j = \lambda)$ to

$$(\forall f_j \in F) \left(\Pr\{f_j \mid F\} = \frac{1}{|F|} \right) . \quad (5.4.10)$$

Wzory (5.4.4)-(5.4.7) upraszczają się wtedy do postaci:

$$S_{M_c} = \frac{1}{|F| \cdot |\mathbb{X}|} \left(\sum_{f_j \in F} \sum_{X_i \in \mathbb{X}} \operatorname{sgn} H(X_i, f_j) \in \mathbb{Z} \right), \quad (5.4.11)$$

$$S_{D_c} = \frac{1}{|F| \cdot |\mathbb{X}|} \left(\sum_{f_j \in F} \sum_{X_i \in \mathbb{X}} \operatorname{sgn} H(X_i, f_j) \in \overline{\mathbb{Z}} \right), \quad (5.4.12)$$

$$S_{D_n} = \frac{1}{|F| \cdot |\overline{\mathbb{X}}|} \left(\sum_{f_j \in F} \sum_{X_i \in \overline{\mathbb{X}}} \operatorname{sgn} H(X_i, f_j) \in \overline{\mathbb{Z}} \right), \quad (5.4.13)$$

$$S_{U_n} = \frac{1}{|F| \cdot |\overline{\mathbb{X}}|} \left(\sum_{f_j \in F} \sum_{X_i \in \overline{\mathbb{X}}} \operatorname{sgn} H(X_i, f_j) \in \mathbb{Z} \right), \quad (5.4.14)$$

gdzie: 1) dla kodów stałowagowych m/n $|\mathbb{X}| = \binom{n}{m}$, $|\overline{\mathbb{X}}| = 2^n - \binom{n}{m}$,

2) jeżeli F zawiera uszkodzenia typu s/z, z = {0,1}, we wszystkich połączeniach, to |F| = 2·(liczba połączeń).

Autorowi nie są znane efektywne metody, które można zastosować do wyznaczania miar S_{Mc}, S_{Dc}, S_{Dn}, S_{Un} w przypadku układów kontrolnych o dowolnej strukturze logicznej, w szczególności o większej liczbie wejść i dużej liczbie wspólnych podukładów. Do czasu opracowania takich metod jedynie możliwe jest komputerowe wyznaczenie tych miar z odpowiednich wzorów (5.4.4)-(5.4.7) lub (5.4.11)-(5.4.14).

C. 1) Niezawodność sygnału wyjściowego SR(t) w chwili t określa wzór:

$$SR(t) = FR(t) + \prod_{j=1}^{|F|} s_j(t)\cdot(w_c + w_n \cdot S_{Dn}^j), \tag{5.4.15}$$

gdzie $S_{D_n}^j = \frac{1}{|\overline{\mathbb{X}}|}\cdot\sum_{X_i \in \overline{\mathbb{X}}} \operatorname{sgn}\{H(X_i, f_j) \notin \mathbb{Z}\}$. (5.4.16)

Jeżeli $(\forall f_j \in F)\ (\lambda_j = \lambda)$ to $s_j(t) = s(t) = \exp(-\lambda\cdot t)$ i wzór (5.4.15) przyjmuje postać

$$SR(t) = FR(t) + [s(t)]^{|F|} \cdot (w_c + w_n \cdot S_{Dn}). \tag{5.4.17}$$

Jeżeli przyjmiemy, że $[s(t)]^{|F|} \approx 1-\exp(-|F|\cdot\lambda\cdot t)$, i skorzystamy z tożsamości $w_c + w_n \cdot S_{Dn} = 1 - w_n \cdot S_{Un}$, to po przekształceniach otrzymujemy wzór

$$SR(t) = 1 - w_n \cdot S_{U_n} \cdot [1 - \exp(-|F|\cdot\lambda\cdot t)]. \tag{5.4.18}$$

2) Po uwzględnieniu założeń Z.5.4.2, Z.5.4.6 i Z.5.4.7 otrzymujemy następujący wzór na niezawodność sygnału wyjściowego $R_s(t)$ w przedziale czasu [0,t]:

$$R_s(t) = \exp(-w_n \cdot t \cdot \sum_{f_j \in F} \lambda_j \cdot S_{Un}^j), \tag{5.4.19}$$

$$\text{gdzie } S^j_{U_n} = \frac{1}{|\mathbb{X}|} \cdot \sum_{X_i \in \mathbb{X}} \operatorname{sgn}\{H(X_i, f_j) \in \mathbb{Z}\} . \qquad (5.4.20)$$

Jeżeli $(\forall f_j \in F)\ (\lambda_j = \lambda)$, to wzór (5.4.19) upraszcza się do postaci

$$R_S(t) = \exp -(\lambda \cdot F \cdot S_{U_n} \cdot w_n \cdot t) . \qquad (5.4.21)$$

3) Czas T_m wyznacza się ze wzoru

$$T_m = - \frac{\ln R_{Smin}}{w_n \cdot \sum_{f_j \in F} (\lambda_j \cdot S^j_{U_n})} \qquad (5.4.22)$$

lub gdy $(\forall f_j \in F)\ (\lambda_j = \lambda)$, to

$$T_m = - \frac{\ln R_{Smin}}{\lambda \cdot |F| \cdot S_{U_n} \cdot w_n} . \qquad (5.4.23)$$

5.5. Przykłady

Przykład 5.5.1

Wyznaczyć miary niezawodności dwóch STC kodów 2/5;

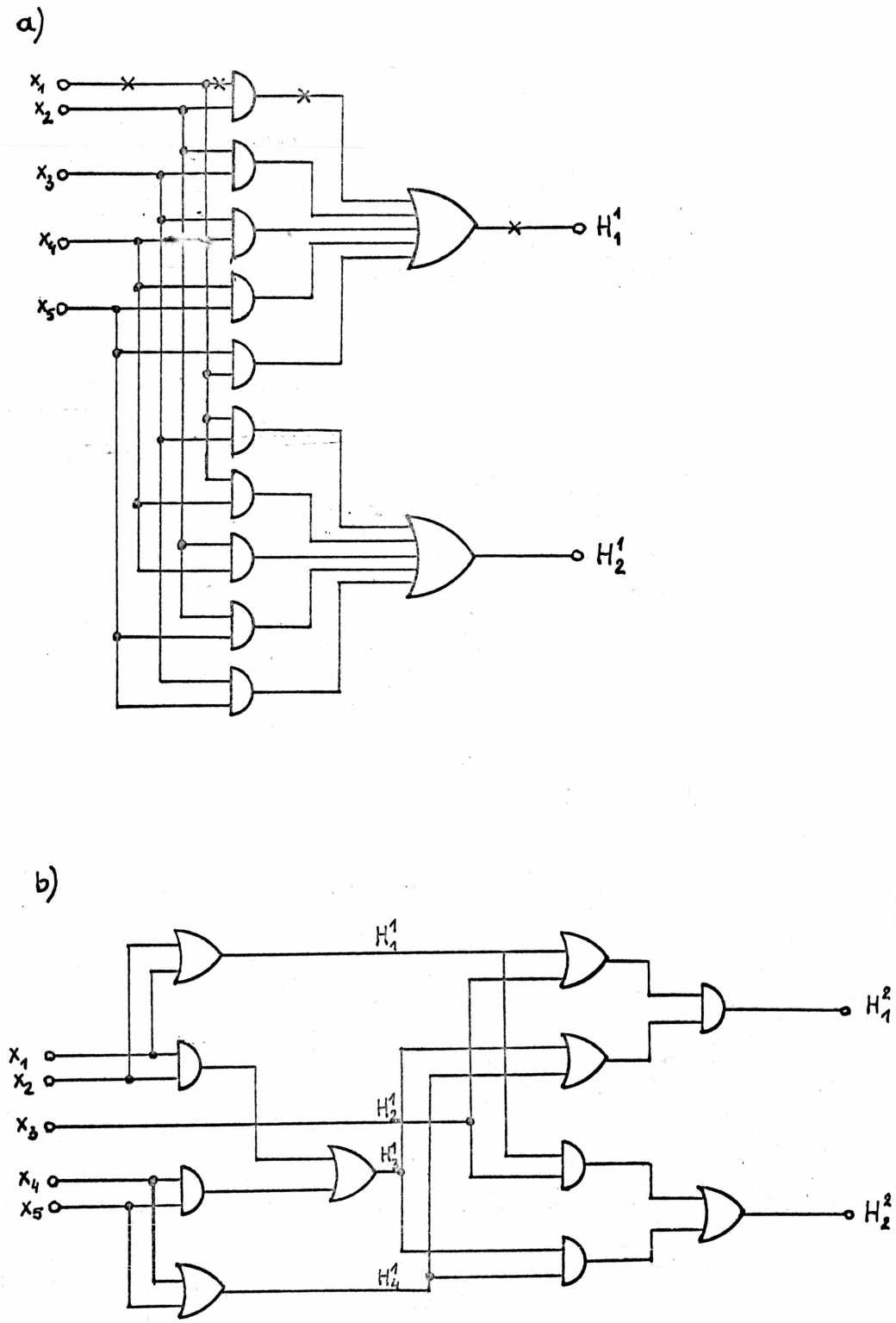
a) układu A zaprojektowanego wg metody z [44],

b) układu B zaprojektowanego wg algorytmu 4.2.1, przy następujących założeniach:

1) uszkodzenia typu s/z $z=\{0,1\}$ występują w pierwotnych liniach wejściowych i wyjściowych układu oraz w połączeniach wewnętrznych (rozróżnia się uszkodzenia przed i za punktem rozgałęzienia linii połączeń),
2) intensywność uszkodzeń wynosi λ dla każdego uszkodzenia,
3) prawdopodobieństwa wystąpienia wszystkich słów kodowych są jednakowe,
4) prawdopodobieństwa wystąpienia wszystkich słów niekodowych są jednakowe.

Schematy logiczne układów A i B przedstawia rys. 5.5.1.

Parametry i miary niezawodności obydwu układów zestawiono w tabeli 5.5.1.



Rys. 5.5.1. Schematy logiczne STC kodu 2/5 zaprojektowane a) wg metody z [44], b) wg algorytmu 4.2.1

f_1, f_2, f_3, f_4 oznaczają przykładowe miejsca typowych uszkodzeń uwzględnianych przy ocenie niezawodności

Tabela 5.5.1

Symbol parametru lub miary	Układ A	Układ B
$\|F\|$	74	64
$FR(t)=R_F(t)$	$e^{-74\cdot\lambda\cdot t}$	$e^{-64\cdot\lambda\cdot t}$
S_{Mc}	0,74	0,68
S_{Dc}	0,26	0,32
S_{Dn}	0,88	0,8
S_{Un}	0,12	0,2
$SR(t)$	$1-0,12\cdot w_n\cdot(1-e^{-74\cdot\lambda\cdot t})$	$1-0,2\cdot w_n\cdot(1-e^{-64\cdot\lambda\cdot t})$
$R_s(t)$	$e^{-8,88\cdot\lambda\cdot w_n\cdot t}$	$e^{-12,8\cdot\lambda\cdot w_n\cdot t}$
T_m	$-\frac{\ln R_S}{8,88\cdot w_n\cdot\lambda}$	$-\frac{\ln R_S}{12,8\cdot w_n\cdot\lambda}$

Z danych zamieszczonych w tabeli 5.5.1 wynikają następujące wnioski:

1) układ B jest zawsze bardziej niezawodny od układu A, ponieważ jest mniejszy (występuje w nim mniej uszkodzeń) w sensie klasycznej definicji niezawodności $(FR(t), R_F(t))$, która nie uwzględnia własności struktury logicznej układu ,
2) uszkodzenia układu B są łatwiej wykrywalne przez słowa kodowe w czasie pracy układu, niż uszkodzenia układu A $(S_{Dc}(B) > S_{Dc}(A))$
3) średnia niewykrywalność wejść niekodowych wskutek uszkodzeń (S_{Un}) jest wyższa w układzie B niż w układzie A,
4) układ A, chociaż jest większy, ma zawsze większą niezawodność sygnału wyjściowego $(SR(t), R_S(t))$ niż układ B. Wynika to z mniejszej podatności (scharakteryzowanej przez S_{Un}) struktury logicznej układu A do błędnego działania wskutek uszkodzeń wewnętrznych,
5) czas T_m układu A jest zawsze 1,4413 raza dłuższy od czasu T_m układu B.

Przykład 5.5.2

Wyznaczyć miary niezawodności dwóch STC kodu 3/8:

a) układu A zaprojektowanego wg algorytmu Marouf'a-Friedman'a [35],

b) układu zaprojektowanego wg algorytmu 4.3.2, przy tych samych założeniach co w przykładzie 5.5.1.

Wybrane parametry i miary niezawodności obydwu układów zamieszczono w tabeli 5.5.2.

Tabela 5.5.2

Symbol parametru lub miary	Układ A	Układ B
$\|F\|$	236	150
$FR(t)=R_F(t)$	$e^{-236\cdot\lambda\cdot t}$	$e^{-150\cdot\lambda\cdot t}$
S_{Un}	0,047	0,056
$SR(t)$	$1-0,047\cdot w_n\cdot(1-e^{-236\cdot\lambda\cdot t})$	$1-0,056\cdot w_n\cdot(1-e^{-150\cdot\lambda\cdot t})$
$R_S(t)$	$e^{-11,09\cdot\lambda\cdot w_n\cdot t}$	$e^{-8,4\cdot\lambda\cdot w_n\cdot t}$
T_m	$-\frac{\ln R_S}{11,09\cdot w_n\cdot\lambda}$	$-\frac{\ln R_S}{8,4\cdot w_n\cdot\lambda}$

Z danych zamieszczonych w tabeli 5.5.2 wynikają następujące wnioski:

1) średnia niewykrywalność wejść niekodowych wskutek uszkodzeń (S_{Un}) jest nieco wyższa w układzie B niż w układzie A,

2) układ B, który jest mniejszy, jest bardziej niezawodny od układu A, zarówno w sensie klasycznej definicji niezawodności $(FR(t), R_F(t))$ jak i ze względu na większą niezawodność sygnału wyjściowego $(SR(t), R_S(t))$. Zadecydowała o tym znacznie większa (o ponad 50%) liczba uszkodzeń układu A niż układu B, której wpływ został tylko częściowo skompensowany przez nieco lepszy parametr S_{Un} układu B,

3) czas T_m układu B jest zawsze 1,32 raza dłuższy od czasu T_m układu A.

5.6. Uwagi końcowe

Mniejsze rozmiary układu nie przesądzają o jego bardziej niezawodnym działaniu, scharakteryzowanym przez niezawodność sygnału wyjściowego.
Fakt ten stwierdzony np. w [30] dla zwykłych układów kombinacyjnych znalazł również potwierdzenie w przypadku STC kodów m/n. Świadczy to o tym, że wpływ występowania w układzie o większych rozmiarach większej liczby uszkodzeń może zostać skompensowany przez większą zdolność struktury logicznej układu do maskowania wpływu uszkodzeń wewnętrznych na jego błędne działanie. Jednakże przy relatywnie dużym wzroście rozmiarów układu (przykład 5.5.2) przede wszystkim one decydują o wyższej niezawodności sygnału wyjściowego. Pozwala to przypuszczać, że większość STC kodów m/n zaprojektowanych wg metod podanych w rozdziale 4 działa bardziej niezawodnie niż analogiczne układy zaprojektowane wg metod znanych z literatury. Jednakże dokładne zbadanie tego problemu wykracza poza zakres tej pracy i powinno być przedmiotem dalszych szczegółowych opracowań.

6. ZAKOŃCZENIE

W przedstawionej pracy podjęto próbę opracowania nowych metod projektowania samotestowalnych układów kontrolnych STC kodów stałowagowych m/n.

Dla następujących klas kodów m/n:

i) kody 1/n, $n \geq 7$,

ii) kody 2/n, $n \geq 5$,

iii) kody m/n, $m \geq 3$, $(2 \cdot m + 1) \leq n \leq 4 \cdot m$,

iv) kodów dualnych do nich, tzn. kodów (n-m)/n,

gdzie n i m spełniają warunki określone dla klas i) - iii), metody te umożliwiają zaprojektowanie STC o mniejszej złożoności mierzonej liczbą bramek i łączną liczbą wejść bramek niż analogiczne układy otrzymane wg znanych metod.

Ich wyróżniającą cechą jest translacja kodu m/n w kod m_1/n_1 ($n_1 < n$), gdzie: i) $n_1 \geq 5$ i $m_1 \geq 2$ gdy m=1, ii) $m_1=2$ gdy m=2, iii) $m_1=2$ i $n_1=4$ gdy $m \geq 3$, a następnie na kod 1/2 (w przypadku kodów 1/n i 2/n translacja taka może być wielostopniowa, z zachowaniem w.w. ograniczeń na parametry kolejnych translatorów). Zastosowanie translacji kodu m/n w kod $2/n_1$ (i ewentualnie w inne kody $2/n_i$) pozwoliło podać wektorową funkcję logiczną realizowaną przez każdy taki translator w postaci sum iloczynów, w których występują iloczyny mniejszej liczby zmiennych wejściowych niż wynosi waga kodu wejść np. m (w dotychczasowych rozwiązaniach STC kodów m/n było to niemożliwe, gdyż układ nie mógłby wtedy wykryć wszystkich słów niekodowych). W połączeniu z zastosowaniem w tych translatorach układów progowych (o wszystkich wagach wejść równych 1) wielopoziomowych i o wspólnych podukładach (dla układów tych podano nowy algorytm projektowania, bardziej efektywny od znanych dotychczas metod podanych w [3, 59]), przyczyniło się to w decydujący sposób nie tylko do zmniejszenia rozmiarów STC, lecz również poprawy ich testowalności. Ponieważ w pracy Marouf'a i Friedman'a [35], z której osiągnięciami porównywano większość otrzymanych rezultatów, nie podano ogólnych zależności pozwalających oszacować rozmiary układu, porównania dokonano dla układów kontrolnych tych kodów, dla których oszacowanie podano w [35]. Z porównania tego wynika, że te same układy zaprojektowane wg przedstawionych tutaj metod dla wszystkich wyszczególnionych w [35] kodów (dla których $m \geq 2$) wymagają mniejszej liczby bramek

i wejść bramek (oszczędności wahają się od ok. $\frac{1}{5}$ dla kodów 2/n, do ok. 30 razy dla kodu 10/21). Istnieją przesłanki, że podobne wyniki uzyskuje się dla innych kodów, których dotyczą metody z [35]. Również dla większości kodów 1/n otrzymano mniejsze układy (szczególnie korzystnie zmalała liczba połączeń), niż układy o strukturze znanej z [3] i układzie kontrolnym kodu $m_1/2m_1$ zaprojektowanym wg metody z [48]. Ponadto w przypadku STC kodów m/n, takich, że $m \geqslant 2$, dla wszystkich w.w. kodów wymagają one mniejszej liczby testów koniecznych do wykrycia wszystkich uszkodzeń pojedynczych typu s/z, $z = \{0,1\}$, a dla dużej liczby kodów wnoszą mniejsze opóźnienie (mierzone liczbą poziomów bramek) niż układy zaprojektowane wg metody z [35].

Parametry układów progowych i STC kodów m/n takie jak: liczba bramek, liczba wejść bramek, liczba poziomów i minimalna liczba testów, wyznaczono z podanych wzorów.

Na podkreślenie, zdaniem autora, zasługuje fakt, że podane metody projektowania (dla kodów 1/n metoda heurystyczna, dla pozostałych kodów m/n metody algorytmiczne) są intuicyjnie proste i mało czasochłonne.

Sformułowano również warunki wystarczające samotestowalności dla uszkodzeń jednokierunkowych dowolnego bezinwersyjnego układu kombinacyjnego wykorzystującego kody stałowagowe oraz wykazano kilka własności kombinacyjnych układów kontrolnych. Rezultaty te wykorzystano w dowodach poprawności metod projektowania STC kodów m/n.

Rozważano ponadto możliwość oceny niezawodności STC. W oparciu o przyjęty model niezawodnościowy STC zaproponowano miary niezawodności uwzględniające własności strukturalne tych układów i podano dla nich ogólne zależności. Wprowadzenie tych miar ma na celu zarówno poznanie parametrów niezawodnościowych układu, jak i porównanie układów otrzymanych różnymi metodami. Ze względu na brak efektywnych metod wyznaczania tych miar ograniczono się do porównania niezawodności kilku STC kodów m/n otrzymanych różnymi metodami. Najważniejszym wnioskiem wypływającym z tej analizy jest potwierdzenie znanego skądinąd np. 30 faktu, że mniejsze rozmiary układu nie przesądzają o jego większej niezawodności. Istnieją przesłanki, że dotyczy to co najwyżej układów, które nie różnią się rozmiarami więcej niż o ok. 50%.

Kontynuacja badań powinna dotyczyć następujących zagadnień bezpoś-

rednio związanych z poruszoną w pracy tematyką:

- rozszerzenie wyników otrzymanych w pracy na STC kodów m/n takich, że $m \geqslant 3$ i $n > 4 \cdot m$,
- metody projektowania STC innych kodów nieuporządkowanych, np. kodów Bergera, z zastosowaniem STC kodów stałowagowych,
- metody projektowania samosprawdzalnych układów sekwencyjnych synchronicznych i asynchronicznych z zastosowaniem kodów nieuporządkowanych,
- metody projektowania samosprawdzalnych układów cyfrowych wykorzystujących kody stałowagowe niekompletne,
- efektywne metody wyznaczania miar niezawodności STC.

mgr inż. Stanisław Piestrak

Raport wpłynął do Redakcji
IV. 1982

A B S T R A C T

"DESIGN METHODS OF SELF-TESTING CHECKERS FOR m-out-of-n CODES"

In this dissertation new design methods of self-testing checkers for m-out-of-n codes are presented. The following three cases of these codes are considered:
1/ 1-out-of-n codes, $n \geq 7$,
2/ 2-out-of-n codes, $n \geq 5$,
3/ m-out-of-n codes, $m \geq 3$ and $(2m+1) \leq n \leq 4m$.
The basic idea in designing these checkers is to first translate the m-out-of-n code in the m_1-out-of-n_1 code $(n_1 < n)$, where: 1/ $n_1 \geq 5$ and $m_1 \geq 2$ if $m=1$, 2/ $m_1=2$ if $m=2$, 3/ $m_1=2$ and $n_1=4$ if $m \geq 3$, and then translating the m_1-out-of-n code into 1-out-of-n code. In the cases of (1 or 2)-out-of-n codes more than one translator is generally used. In the realization of these translators the multilevel majority detection circuits using shared logic are used. Efficient algorithm of designing such circuits is also given. They require slightly fewer gates and considerably fewer logic levels than those designed using Reddy's method. The proposed checkers are self-testing for unidirectional faults. They require fewer number of gates having fewer number of inputs and are generally easier testable than any other known until now.
General rules of estimation of reliability measures of self-testing checkers are also given.

7. LITERATURA

[1] Amarel S., Brzozowski J.A., "Theoretical considerations on reliability properties of recursive trangular switch networks", w: "Redundancy Techniques for Computing Systems, Wilcox and Mann, Ed., Washington, D.C., Spartan 1962.

[2] Anderson D.[illegible], "Design of self-checking digital networks using coding techniques", Ph.D., dissert., Univ. of Illinois, Urbana, Oct. 1971.

[3] Anderson D.A., Metze G., "Design of totally self-checking check circuits for m-out-of-n codes", IEEE Trans. Comput., vol, C-22, No 3, March 1973, ss. 263-269.

[4] Bennetts R.G., "A review of fault-tolerance and its application to digital systems containing VLSI components", Proc. 2nd Conf., Fault-Tolerant Systems and Diagnostics 79, Brno 1979, ss. 1 - 12.

[5] Berger J.M.,"A note on error detection codes for assymmetric channels", Inform. Control, vol. 4, 1961, ss. 68 - 73.

[6] Betancourt R., "Derivation of minimum test sets for unate logic circuits", IEEE Trans. Comput., vol. C-20, No 11, Nov. 1971, ss. 1264 - 1269.

[7] Bose B., Rao T.R.N., "Unidirectional error codes for shift register memories", Dig. Pap. FTC-10, Kyoto, Japan, Oct. 1 - 3, 1980, ss. 26 - 28.

[8] Bouricius W.G., et. al., "Reliability modeling for fault-tolerant computers", IEEE Trans. Comput., vol. C-20, No 11, Nov. 1971, ss. 1306 - 1311.

[9] Breuer M.A., Friedman A.D., "Diagnosis and Reliable Design of Digital Systems", Computer Science Press, Inc., Woodland Hills, California, 1976.

[10] Carter W.C., Scheider P.R., "Design of dynamically-checked computers", Information Processing 68, Proc. IFIP Conf., Edinburgh, Scotland, August 1968, ss. 878 - 883.

[11] Chang H.Y.-P., Dorr R.C., Senese D.J., "The design of microprogrammed self-checking processor of an Electronic Switching System", IEEE Trans. Comput., vol. C-22, No 5, May 1973, ss. 489 - 500.

[12] Clary J.B., Sacane R.A., "Self-testing computers", Computer, Oct. 1979, ss. 49 - 59.

[13] Cook R.W. et al., "Design of a self-checking microprogram control", IEEE Trans. Comput., vol. C-22, No 3, March 1973, ss. 255 - 262.

[14] Crouzet Y., "Conception de circuits a large échelle d'intégration totalement autotestables", Docteur Ingenieur Thèse, Institut National Polytechnique de Toulouse, Toulouse, Nov. 1978.

[15] Crouzet Y., Landrault Ch., "Design of self-checking MOS-LSI circuits. Application to a four-bit microprocessor", IEEE Trans. Comput., vol. C-29, No 6, June 1980, ss. 532 - 537.

[16] Crouzet Y., Landrault C., "Design specifications of a self--checking detection processor", Dig. Pap. FTC-10, Kyoto, Japan, Oct. 1-3, 1980, ss. 275 - 277.

[17] Dandapani R., "Derivation of minimal test sets for monotonic circuits", IEEE Trans. Comput., vol. C-22, No 7, July 1973, ss. 657 - 661.

[18] David R., "A totally self-checking 1-out-of-3 checker", IEEE Trans. Comput., vol. C-27, No 6, June 1978, ss. 570 - 572.

[19] David R., Thevenod-Fosse P., "Design of totally self-checking asynchronous modular circuits", J. Des. Autom. Fault-Tolerant Comput., No 2, 1978, ss. 271 - 287.

[20] Diaz M., "Design of totally self-checking and fail-safe sequential machines", Dig. Pap. FTC-4, Urbana, Illinois, June 1974, s. 3-19 - 3-24.

[21] Diaz M., Azema P., Ayache J.M., "Unified design of self-checking and fail-safe combinational circuits and sequential machines", IEEE Trans. Comput., vol. C-28, No 3, March 1979, ss. 276 - 281.

[22] Diaz M., de Souza J.M., "Design of self-checking microprogrammed controls", Dig. Pap. FTC-5, Paris, France, June 1975, ss. 137 - 142.

[23] Etiemble D., "Multivalued I^2L circuits for TSC checkers", IEEE Trans. Comput., No 1, C-29, No 6, June 1980, ss. 537 - - 540.

[24] Freiman C.V., "Optimal error detection codes for completely assymmetric binary channels", Inform. Control, vol. 5, 1962, ss. 64 - 71.

[25] Gaitanis N., Halatsis C., Sigala M., "Fail-safe counters", Digit. Process., vol. 5, No 1-2, 1979, ss. 43 - 57.

[26] Galiay J., Crouzet Y., Vergniault M., "Physical versus logical fault models in MOS LSI circuits: Impact on their testability", IEEE Trans. Comput. vol. C-29, No 6, June 1980, ss. 527 - 531.

[27] Geffroy J.-C., "Test en ligne et sécurité de systèmes logiques", Docteur d'Etat Thèse, Univ. Paul Sabatier de Toulouse, Toulouse, Nov. 1976.

[28] Geffroy J.-C., Diaz M., "Unified approach to the study of self-checking systems", Digit., Process., vol. 3, No 4, 1977, ss. 289 - 306.

[29] Kohavi Z., "Switching and Finite Automata Theory", 2nd ed., McGraw-Hill Book Co., N.Y., 1978.

[30] Koren I., "Analysis of the signal reliability measure and an evaluation procedure", IEEE Trans. Comput., vol. C-28, No 3, March 1979, ss. 244 - 249.

[31] Koren I., Sadeh E., "A new approach to the evaluation of the reliability of digital systems", IEEE Trans. Comput., vol. C-29, No 3, March 1980, ss. 261 - 267.

[32] Kwek K.H., Tohma Y., "Signal reliability evaluation of self--checking networks", Dig. Pap. FTC-10, Kyoto, Japan, Oct. 1-3, 1980, ss. 257 - 262.

[33] Lin S., "An introduction to error-correcting codes", Prentice-Hall, Inc., Englewood Cliffs, N.Y., 1970.

[34] Maki G.K., "A self-checking microprocessor design", J. Des. Autom. Fault-Tolerant Comput., vol. 3, No 1, Jan. 1978, ss. 15-27.

[35] Marouf M.A., Friedman A.D., "Efficient design of self-checking checker for any m-out-of-n code", IEEE Trans. Comput., vol. C-27, No 6, June 1978, ss. 482 - 490.

[36] McDonald J.C., "Testing for high reliability: a case study", Computer, Feb. 1976, ss. 18 - 21.

[37] Nanya T., Tohma Y., "Design of self-checking asynchronous sequential circuits", Dig. Pap., FTC-10, Kyoto, Japan, Oct. 1-3, 1980, ss. 278 - 281.

[38] Ogus R.C., "The probability of a correct output from a combinational circuit", IEEE Trans. Comput., vol. C-24, No 5, May 1975, ss. 534 - 544.

[39] Parhami B., Avižienis A., "Detection of storage errors in mass memories using low-cost arithmetic error codes", IEEE Trans. Comput., C-27, No 4, April 1978, ss. 302 - 308.

[40] Piestrak S., "On the structure of Berger codes", w przygotowaniu.

[41] Pradhan D.K., "A new class of error-correcting/detecting codes for fault-tolerant computer applications", IEEE Trans. Comput., vol. C-29, No 6, June 1980, ss. 471 - 481.

[42] Pradhan D.K., Stiffler J.J., "Error-correcting codes and self--checking circuits", Computer, March 1980, ss. 27 - 37.

[43] Rao T.R.N., "Error coding for arithmetic processors", Academic Press, New York, N.Y., 1974.

[44] Reddy S.M., "A note on self-checking checkers", IEEE Trans. Comput., vol. C-23, No 10, Oct. 1974, ss. 1100 - 1102.

[45] Rennels D.A., "Architectures for fault-tolerant spececraft computers, Proc. IEEE, vol. 66, No 10, Oct. 1978, ss. 1255 - -1268.

[46] Sapozhnikov V.V., Sapozhnikov Vl.V., "Sintez połnostiusamokontrolirujuszczichsa asinchronnych avtomatov", Avtom. Telemech., No 1, 1979, ss. 154 - 166.

[47] Smith J.E., "The design of totally self-checking combinational circuits", Coordinated Science Laboratory, Rep. R-737, Univ. of Illinois, August 1976.

[48] Smith J.E., "The design of totally self-checking check circuits for a class of unordered codes", J. Des. Autom. Fault--Tolerant Comput., vol. 2, No 4, Oct. 1977, ss. 321 - 342.

[49] Souza J.M. de Paz E.P., Landrault C., "A research oriented microcomputer with built-in auto-diagnostics", Dig. Pap. FTC-6, Pittsburgh, PA. June 1976, ss. 3-8.

[50] Sum E.K.S., Avižienis A., "A probabilistic model for the evaluation of signal reliability of self-checking logic circuits", Dig. Pap. FTC-6, Pittsburgh, Pennsylvania, June 21--23, 1976, ss. 83 - 87.

[51] Tasar O., Tasar V., "A study of intermittent faults in digital computers", AFIPS Conf. Proc., vol. 46, 1977 NCC, ss. 807 - 811.

[52] Tohma Y. et al., "Realization of fail-safe sequential machines by using a k-out-of-n code", IEEE Trans. Comput., vol. C-20, No 11, Nov. 1971, ss. 1270 - 1275.

[53] Toy W.N., "Modular LSI control logic design with error detection", IEEE Trans. Comput., vol. C-20, No 2, Feb. 1971, ss. 161 - 166.

[54] Wakerly J.F., "Partially self-checking circuits and their use in performing logical operations", IEEE Trans. Comput., vol. C-23, No 7, July 1974, ss. 658 - 666.

[55] Wakerly J.F., "Detection of unidirectional multiple errors using low-cost arithmetic codes", IEEE Trans. Comput., vol. C-24, No 2, Feb. 1975, ss. 210 - 212.

[56] Wakerly J.F., "Error detecting codes, self-checking circuits and applications", American Elsevier, New York, 1978.

[57] Wang S.L., Avižienis A., "The design of totally self-checking circuits using programmable logic arrays", Dig. Pap. FTC-9, Madison, Wisconsin, USA, June 20-22, 1979, ss. 173 - 180.

[58] Yang Y.W., Lin S.M., Chen T.C., "Threshold I^2L totally self--checking circuits", Dig. Pap. FTC-10, Kyoto, Japan, Oct. 1-3, 1980, ss. 284 - 287.

[59] Reddy S.M., Wilson J.R., "Easily testable cellular realizations for the (exactly p)-out-of-n and (p or more)-out-of-n logic functions", IEEE Trans. Comput. vol. C-23, No. 1, Jan. 1974, ss. 98-100.

Niniejszy raport otrzymują:

1.	Biblioteka Główna Politechniki Wrocławskiej	1
2.	OINT - Biblioteka Instytutu Cybernetyki Technicznej -	1
3.	Asystent Dyrektora d/s Kształcenia Kadry Naukowej	1
4.	Promotor -	1
5.	Recenzenci -	2
6.	Autor -	1

Razem: 7 egz.