FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Diplomová práce

Nelineární blokové kódy pro testování číslicových obvodů

Studijní program: N0613A140028 Informační technologie
Studijní obor: Výpočetní systémy

Autor práce: **Bc. Martin Koňák**
Vedoucí práce: prof. Ing. Ondřej Novák, CSc.
Ústav informačních technologií a elektroniky

Liberec 2024

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání diplomové práce

Nelineární blokové kódy pro testování číslicových obvodů

Jméno a příjmení:	**Bc. Martin Koňák**
Osobní číslo:	M22000034
Studijní program:	N0613A140028 Informační technologie
Specializace:	Výpočetní systémy
Zadávající katedra:	Ústav informačních technologií a elektroniky
Akademický rok:	2023/2024

Zásady pro vypracování:

1. Seznamte se s výsledky výzkumu v oblasti nelineárních kompresních kódů.
2. Ověřte teoretické výsledné hodnoty parametrů kódů s více specifikovanými bity pomocí simulací v Matlabu [3].
3. Najděte strategii deterministického vytváření kódů pro čtyři a více specifikovaných bitů.
4. Zvažte hardwarovou náročnost dekompresorů využívajících navržených kódů.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	40 – 50 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	čeština

Seznam odborné literatury:

[1] O. Novák, "Nonlinear Compression Block Codes Search Strategy," *2022 25th Euromicro Conference on Digital System Design (DSD)*, Maspalomas, Spain, 2022, pp. 665-670, doi: 10.1109/DSD57027.2022.00094

[2] Jan Schmidt and Petr Fišer: "Nonlinear codes for test patterns compression: the old school way," 14^{th}international Workshop on Boolean Problems, Bremen, DE, 2020

[3] https://www.mathworks.com/products/matlab.html

Vedoucí práce:	prof. Ing. Ondřej Novák, CSc. Ústav informačních technologií a elektroniky
Datum zadání práce:	20. října 2023
Předpokládaný termín odevzdání:	14. května 2024

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. RNDr. Pavel Satrapa, Ph.D.
garant studijního programu

V Liberci dne 20. října 2023

Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

8. května 2024 Bc. Martin Koňák

Poděkování

Na tomto místě bych velice rád poděkoval vedoucímu své práce prof. Ing. Ondřeji Novákovi, CSc za příležitost zabývat se tímto tématem, ochotu, vstřícnost a přínosnou spolupráci, díky které tato práce vznikla.

Velké poděkování patří také mé rodině za důvěru a podporu během celé doby studia.

V neposlední řadě bych rád poděkoval také doc. Ing. Petru Fišerovi, Ph.D, který provedl srovnávací syntézu a poskytl tak přínosná data pro dokončení poslední kapitoly této práce.

Nelineární blokové kódy pro testování číslicových obvodů

Abstrakt

Téma nelineárních blokových kódů a jejich využití pro testování logických obvodů přináší možnosti inovace v této oblasti. Testování logických obvodů je v současnosti realizováno s využitím lineárních blokových kódů. Nelineární blokové kódy nabízejí novou perspektivu z hlediska testování například v možném přínosu vyšší efektivity. Vytváření těchto kódů a uzpůsobení k testování vychází z počtu specifikovaných bitů testovaného obvodu. Pro každý rozdílný počet existují návrhy deterministických přístupů pro realizaci takových kódů. V současné době je možné tyto přístupy teoreticky ověřit navázáním na dostupné výsledky a přiblížit celé téma k možnosti realizace.

Klíčová slova: Nelineární blokový kód, Lineární blokový kód, Testování logických obvodů, Dekompresor, Hardwarová realizace, Syntéza logického obvodu

Nonlinear Block Codes Used in Digital Circuit Testing

Abstract

The topic of nonlinear block codes and their use for testing logic circuits brings opportunities for innovation in this area. Logic circuit testing is currently implemented using linear block codes. Non-linear block codes offer a new perspective in terms of testing, for example, in the potential benefit of increased efficiency. The creation of these codes and the adaptation to testing is based on the number of specified bits of the circuit under test. For each different count, there are proposals for deterministic approaches to implement such codes. Currently, it is possible to verify these approaches theoretically by relating them to available results and bringing the whole topic closer to the possibility of implementation.

Keywords: Nonlinear block code, Linear block code, Logic circuit testing, Decompressor, Hardware implementation, Logic circuit synthesis

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

LBC	Linear block code
NBC	Noninear block code
BIST	Built-in self test
LFSR	Linear feedback shift register
CCN	Clique Cover Number

Úvod

Práce se zabývá problematikou využití nelineárních blokových kódů (NBC) pro testování číslicových obvodů. Jedná se o relativně krátce zkoumané téma, které má v současné době stále převážně teoretickou podstatu. Hlavní myšlenkou přínosu tohoto tématu svému oboru je efektivnější testování číslicových obvodů, kdy by mělo být možné dosáhnout vyššího počtu testovacích vzorků dopravených do testovaného obvodu při zachování menšího počtu informačních bitů a nižší hardwarové náročnosti.

Rešerši problematiky lze sepsat na základě několika dostupných literárních zdrojů, které toto téma popisují a navrhují konkrétní metody a přístupy jak lze takové blokové kódy vytvářet. Existuje již několik navržených přístupů, které je možné ověřit a potvrdit zda jsou pro toto téma přínosné a na základě kterých je možné realizovat další postup v rozvíjení tématu.

Na dostupné zdroje k tomuto tématu tato diplomová práce přímo navazuje. Ve své práci budu čerpat převážně z publikací vydaných profesorem Novákem, který je zároveň vedoucím práce. Diplomová práce navazuje na některá z témat jeho publikací a rozšiřuje konkrétní oblasti NBC, zde bych měl přispět vlastní částí práce pro rozšíření tématu jako celku.

Jak vyplývá ze zadání, klíčovým tématem je ověření navržených parametrů přístupů pro vytváření NBC pro tři specifikované bity, dále navržení přístupu pro vytváření NBC pro čtyři specifikované bity a následné zvážení hardwarové náročnosti ověřených či navržených parametrů. Dle těchto bodů lze vytyčit základní kapitoly, které budu v této práci popisovat.

Způsob ověření bude proveden sadou testovacích skriptů v prostředí MATLAB, kdy budou prováděny triviální testy navržených kódů a ověřování parametrů formou experimentů. Odvození nových parametrů či přístupů bude provedeno taktéž sadou experimentů doplněných o adekvátní testovací skripty.

Výstupem této práce by tak mělo být ověření navržených přístupů a zobecnění nastavených parametrů pro NBC se třemi specifikovanými bity. Déle navržení deterministického přístupu a dalšího možného postupu pro návrh parametrů pro NBC se čtyřmi a více specifikovanými bity. a následně možná hardwarová realizace a popis složitosti hardwarového zapojení komponent k realizaci navrhovaných NBC.

1 Blokové kódy a testování logických obvodů

Následující kapitola popisuje teoretické poznatky řešené problematiky. Je použita forma rešerše dostupných literárních zdrojů, které jsou k dispozici převážně v anglickém jazyce. Výstupem této sekce by měl být ucelený přehled aktuálních informací o tomto tématu, který poskytne dostatečný teoretický základ znalostí pro zpracování vlastní části práce.

Rešerše je rozdělena na několik klíčových témat, které jsou vzhledem k tématu relevantní. Nejprve je nutné popsat systematiku testování logických obvodů, dále popsat způsob jejich testování pomocí lineárních blokových kódů, a následné srovnání s kódy nelineárními.

Nelineární blokové kódy budou dále popsány na základě aktuálně dostupných zdrojů a rozděleny dle podtémat na prvotní přístupy, volby dekompresorů a nejaktuálnějších deterministických přístupů pro jejich návrh. Rešerše bude v tomto bodě sestavena chronologicky dle dostupných publikací, aby byl zřejmý vývoj této oblasti a možnosti dalšího navázání na toto téma.

1.1 Testování logických obvodů

Téma testování logických obvodu popisuje Novák ve svých publikacích [1] a [2]. Zpracovává rešerše teoretických základů celého tématu a navazuje na ně vlastními experimenty. Z počátku je třeba popsat jakým způsobem funguje testování logických obvodů, jaké jsou možné přístupy řešení takového problému a jak jsou do celého procesu zapojeny lineární či nelineární blokové kódy.

Jednou z hlavních popisovaných částí jsou kódy využívané ke kompresi testovacích vzorů a následné dekompresi po transferu na testovaný obvod. Testovací vzory obsažené v takových kódech obsahují značné množství redundantních dat a proto jsou vhodné ke kompresi, jelikož je snížené riziko ztráty informace. Skutečnost, kdy je potřeba pouze několik bitů nesoucích logickou informaci popisují autoři v publikaci [3]. V průběhu testování je proud bitů transformován na n-bitové testovací vzorky, které jsou následně aplikovány na funkční vstupy logického obvodu [4].

Požadavkem na dekompresor, je sestavení r-tice z vygenerovaných testovacích vzorků a nastavení jejich bitů na libovolné hodnoty [5]. Dekompresor je poté charakterizován počtem n skenovacích řetězců (šířka dekomprimovaných testovacích

vektorů), dále počtem informačních bitů i (hodnota šířky vstupu) a počtem r specifikovaných bitů [2].

Tato charakteristika je následně využita pro další popis testování pomocí blokových kódů. S využitím teoretických znalostí o lineárních kódech popisuje [2] možnosti návrhu vestavěného samo-testovacího zařízení (BIST) nebo vhodného konkrétního dekompresoru. Schéma BIST provádí testovací sadu, která kompletně otestuje veškeré podobvody s počtem vstupu r nebo nižším pomocí 2^i testovacích sad. Oproti tomu dekompresor transformuje i-bitový testovací vzorek na širší n-bitový výstupní vzorek, který garantuje že libovolná r-tice výstupního vzoru může být nastavená na libovolnou hodnotu beze změny informace. Pro konkrétnější představu poskytuje autor zdroj [6], který tento rozdíl popisuje detailněji.

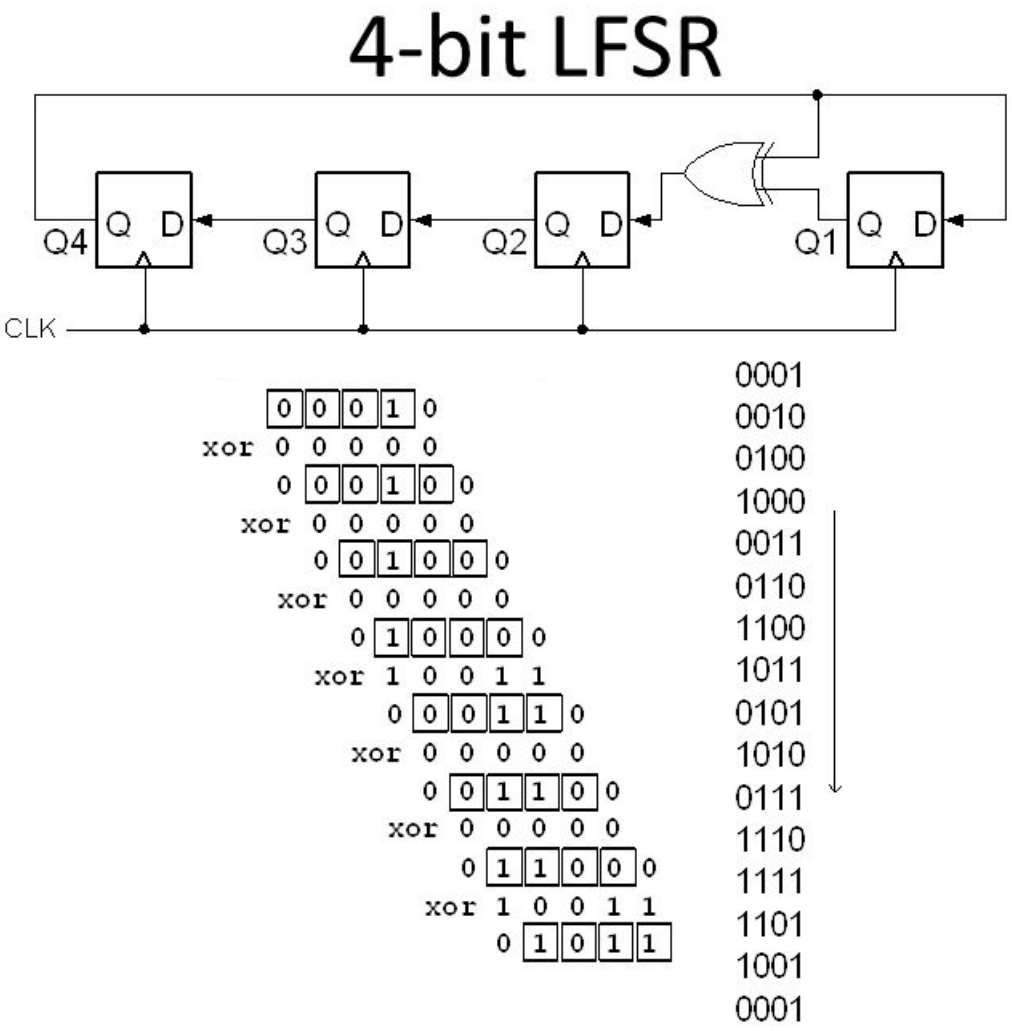
Při využití dekompresoru je nutné nalezení minimálního počtu vstupů a maximálního poštu výstupů, které garantují pokrytí n testovacími vzorky na r specifikovaných bitech. Tyto kódy s hodnotami n, r a i popisuje [2] jako (n, i, r) Kód. Vhodnými kódy pro dekompresi jsou takové, které mají vyšší hodnotu minimální vzdálenosti kódového slova d_{min}. Parametr r je poté shodný s hodnotou d_{min}*-1* [7]. Nevýhodou využití LBC v této metodě je limit výše hodnoty n, která by zajistila kompletní pokrytí specifikovaných bitů r. Autor [2] uvádí že historie vývoje a hledání vhodných LBC je již dlouholetá a nepředpokládá nalezení vhodnějších LBC pro pokrytí vyššího počtu specifikovaných bitů r při požadavku vyšší počty testovacích vzorků n.

Při návrhu, který využívá více skenovacích řetězců (multi-scan-chain) musí být zajištěna možnost nastavení různých logických hodnot na jednotlivé testovací vzorky v rámci jednoho hodinového taktu [8]. Vzájemnou nezávislost testovacích vzorků napříč tímto schématem zajišťuje metoda XPAND, která využívá dekompresor sestavený z hradel typu XOR a zaručuje pokrytí počtu r specifikovaných bitů roven 2 a více [5]. Dosavadní výzkum prokázal, že kódy s parametrem *r=3* jsou vhodné pro testování částečně oddělených hardwarových jader a publikace [2] je shledává velice přínosnými z hlediska dalších možností výzkumů.

1.1.1 Testování obvodů s využitím lineárních blokových kódů

Na základě teorie LBC vycházející z publikací [9] a [10] popisuje Novák [1] ve své rešerši způsob testování logických obvodů s využitím posuvného registru s lineární zpětnou vazbou (LFSR), aby bylo možné jednoznačně srovnat způsoby testování pro LBC a NBC. LFSR generuje jednotlivá kódová slova požadovaného LBC. Takový kód je opět popsán parametry n (délka kódového slova), i (délka LFSR, počet informačních bitů) a d_{min} (minimální vzdálenost kódových slov). Pro každý LBC tedy existuje duální kód, s vlastní hodnotou d_{min}. Kódy, které obsahují ireducibilní generující polynomy a mají vysokou hodnotu minimální kódové vzdálenosti jsou dle [1] vhodné pro aplikaci ve vestavěném testovacím zařízení (BIST). Pokud je LFSR definován takovým polynomem, poté generuje exhaustivní testovací sady v podobě kódu (n, $d_{min}-1$) během jednoho hodinového taktu. Parametr $d_{min}-1$ v tuto chvíli odpovídá poštu specifikovaných bitů r. Každá r-tice bitů kódového slova v průběhu testování postupně dosáhne všech možných logických hodnot a tím kompletně

protestuje všechny r-tice vstupů testovaného obvodu [1]. Možný příklad grafického znázornění zapojeni LFSR je zachycen na následujícím obrázku (Obrázek 1.1), který představuje generování všech čtyřbitových možností s využitím logické funkce XOR.

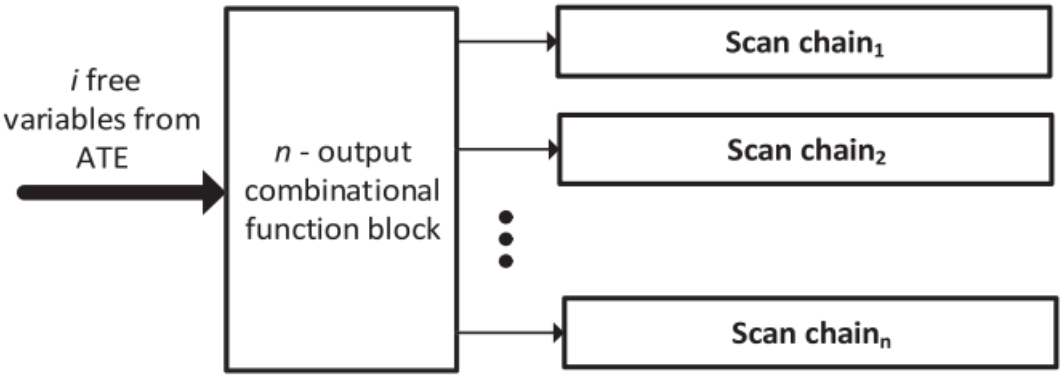


Obrázek 1.1: LFSR

Porovnání s využitím NBC je provedeno ve dvou rovinách. Při rovnání efektivity LBC a NBC pro korekci chyb je výsledek pozitivnější pro aplikaci LBC. Naopak při vytváření kompresních kódů pro testování logických obvodů, nabízejí nelineární vlastnosti více možností pro definování funkcí dekompresoru [2].

1.1.2 Dekompresory a jejich využití pro nelineární blokové kódy

Téma dekompresorů samostatně zpracovává také publikace [11]. Kompresní kódy jsou klíčové pro přenos testovacích vzorků na testovaný obvod (CUT). Testovací vzory jsou náhodně generovány Automatickým generátorem testovacích vzorů (ATPG), jsou zakódovány, uloženy, přeneseny, dekomprimovány a nahrány do paralelních skenovacích řetězců v CUT (Obrázek 1.2. Parametry kódu jsou voleny tak, aby zaručily bezchybný přenos maximálního počtu bitů v přenášeném testovacím vzoru. Přenášené bity jsou pojmenované také jako volné proměnné a jejich počet je značen symbolem i. Počet bitů přeneseného vzorku je reprezentován jako n. Počet specifikovaných bitů, které jsou podrobeny testování je pro kód charakteristický písmenem r [11].



Obrázek 1.2: Schéma dekompresoru [11]

Jak dále uvádí [11] je při optimalizaci kódů s maximálním počtem specifikovaných bitů r pro daný počet informačních bitů i nezbytné použít efektivní algoritmy pro nalezení vhodných kontrolních bitů. Publikace představuje dvě možnosti jak takové algoritmy fungují.

První algoritmus využívá náhodného hledání kombinačních funkcí s cílem pokrýt stanovený počet specifikovaných bitů r. Tento postup se omezuje na hledání vhodných pravdivostních tabulek kombinačních funkcí s vyváženým počtem jedniček a nul. Nově objevené funkce jsou zkoumány s ohledem na splnění požadovaného počtu specifikovaných bitů. Algoritmus postupně rozšiřuje množinu logických funkcí, které garantují požadovaný počet specifikovaných bitů, přičemž se zaměřuje na minimalizaci úprav již nalezených pravdivostních tabulek [11].

Druhý algoritmus využívá strukturu lineárních kontrolních bitů k minimalizaci hardwarové náročnosti dekompresoru. Pozorování nad jednoduchým NBC s i = 4 a r = 3 vedlo k identifikaci regulární struktury pravdivostních tabulek ne-lineárních kontrolních bitů. Tato struktura umožňuje minimalizaci hardwarové náročnosti dekompresoru a zároveň zachování požadovaného počtu specifikovaných bitů [11].

Výsledky experimentů provedené autory [11] ukazují, že kódování testových vzorů s vyšším počtem specifikovaných bitů je s NBC dekodérem účinnější než s LBC deko-

dérem. Tato zjištění jsou klíčová pro účinné testování moderních obvodů v průmyslových aplikacích, kde je efektivita testovacích metod zásadní pro kvalitu a spolehlivost výrobků.

1.2 Nelineární blokové kódy (NBC)

Na základě dostupné literatury je možné téma nelineárních blokových kódů popsat z několika různých směrů. Samozřejmostí je srovnání jejich podobnosti s kódy lineárními a popis jejich sestavování odvozováním. Důležité je srovnání prvotního přístupu, který byl spíše stochastický, s aktuálně dostupnými deterministickými metodami. Za porovnání stojí také metody pro generování méně rozsáhlých nelineárních blokových kódů, které vycházejí z lineárních oproti rozsáhlejším, kde se exaktní metody mohou lišit.

1.2.1 Souvislosti lineárních a nelineárních blokových kódů

Hlavní výhodu využití nelineárních blokových kódů při detekci chyb oproti lineárním shledává autor [1] ve vyšší efektivitě v průběhu komprese testovacích vzorků. Při srovnání délek obou možností je možné zaznamenat jednodušší hardwarovou implementaci, která je velice slibnou možností pro využití v kompresi a následné dekompresi testovacích vzorků [12].

Publikace [13] navrhuje způsob vytvoření nelineárních blokových kódů (NBC) přidáním rozšiřujících bitů k příslušnému lineárnímu kódu (LBC). Jedná se o jeden z prvních návrhů sestavení NBC spojením lineárního kódu s přidanými kontrolními bity, přičemž lze dosáhnout zachování počtu informačních bitů a tím také složitosti kódu, s možností přínosu nárůstu efektivity při dekompresi kódu a zvýšení počtu kódových slov n.

Dále vychází z návrhu několika funkcí, které lze přidat k LBC se třemi informačními bity a tím tento kód efektivně rozšířit, svou teorii podporuje důkazem, který je v článku taktéž obsažen. Návrh na rozšíření kódu o jednotlivé funkce popisuje následující tabulka na obrázku (Obrázek 1.3) [13]. toto rozšíření o konkrétní bity, z nichž jeden je s lineární vlastností a zbytek nelineární poskytuje vyšší efektivitu při zachování původních vlastností kódu.

Informační bity			Kontrolní bity						
a_1	a_2	a_3	$a_1 \oplus a_2 \oplus a_3$	$(a_1 \& a_2) \oplus a_3$	$(a_1 \text{ OR } a_2) \oplus a_3$	$(a_1 \& a_3) \oplus a_2$	$(a_1 \text{ OR } a_3) \oplus a_2$	$(a_2 \& a_3) \oplus a_1$	$(a_2 \text{ OR } a_3) \oplus a_1$
1	1	1	1	0	0	0	0	0	0
1	1	0	0	1	0	1	0	1	1
1	0	1	0	1	0	1	1	1	1
1	0	0	1	1	1	0	1	0	0
0	1	1	0	1	1	1	0	1	1
0	1	0	1	0	1	1	1	0	0
0	0	1	1	0	1	0	1	1	1
0	0	0	0	0	0	0	0	0	0

Obrázek 1.3: Tabulka kontrolních bitů

Důkaz obsažený v publikaci [13] vztahující se k tomuto rozšíření lze následovně volně přeložit. Lze uvažovat dvě funkce f_1 (čtvrtý řádek tabulky) a f_2 (pátý řádek tabulky). První tři řádky tabulky představují pravdivostní tabulku osmi kombinací vstupů pro tři vstupní bity. Pokud jsou hodnoty vstupních bitů a_1 a a_2 rovny 0, poté výstupní hodnoty obou funkcí budou totožné. Naopak pokud jsou tyto vstupy rovny 1, poté budou výstupní funkce rozdílné. Vstupní bit a_3 invertuje výstupní hodnoty pro obě funkce.

Na základě popsané souvislosti je vždy možné nalézt čtyři různé kombinace na výstupních hodnotách funkcí f_1 a f_2 doprovázené osmi přípustnými kombinacemi tři informačních bitů. Prokazatelná existence všech čtyř kombinací vyvozuje skutečnost, že tyto funkce tvoří pár nezávisle specifikovaných bitů.

Každá další modifikace funkce f_2 (následující řádky tabulky) je definována záměnou alespoň jednoho vstupního parametru. Všechny takto implementované páry funkcí jsou nezávislé a zachovávají tak počet specifikovaných bitů rozšířeného kódu roven dvěma. Autor tedy prokázal že je možné nalézt rozšiřující funkce ke stávajícím LBC, při zachování stejného počtu specifikovaných bitů. Rozšířením je možné podstatně zvýšit parametr n vyčerpávajícího testu nebo počet skenovacích řetězců, které jsou paralelně načítány dekompresorem.

Následně autor [13] uvádí několik příkladů, jak by byla ovlivněna délka kódů v souvislosti s růstem počtu testovacích vzorků n při využití LBC v porovnáním s výše popsaným rozšířením. Významné jsou hodnoty pro nižší počty vstupních bitů, jelikož jsou jednoznačněji prokazatelné. Příkladem jsou hodnoty *i=3* informačních bitů, *r=2* specifikovaných bitů a *n=7* v případě LBC a *n=32* za využití nelineárního rozšíření.

V následujícím příkladu [13] je zobecněný přístup pro rozšíření na nelineární kód s určitým počtem specifikovaných bitů. Pro každou trojici je možné nalézt lineární kontrolní bity, které splňují podmínky daného postupu rozšíření. Příklady takových funkcí lze opět čerpat z tabulky výše. V tomto případě vzniká kód (13, 3), který je nelineární a lze jej využít pro vyčerpávající test realizovaný třístupňovým registrem s lineární zpětnou vazbou.

Přínosným výstupem publikace [13] je fakt, že je možné LBC rozšířit tak, aby byl zachován počet informačních bitů původního kódu a zároveň vzrostla délka kódového slova. Rozšířený kód délky n je tak možné využít při vyčerpávajícím testování, kdy lze zaznamenat výrazné zvýšení efektivity. V oblasti komprese vzorků se poté navyšuje počet skenovacích řetězců, které jsou nyní načítány paralelně.

1.2.2 Prvotní přístupy generování NBC

Novák v publikaci [12] popisuje jeden z prvních přístupů k vytvoření nelineárních blokových kódů s více specifikovanými bity, které jsou vhodné pro testování logických obvodů. Tento způsob využívá opěr rozšíření LBC o sloupce kontrolních bitů. V tomto případě se experimenty pohybují v poli náhodných libovolných funkcí, které rozšiřují daný kód a zachovávají jeho vlastnosti v počtech bitů. Kontrolní bity vytvářejí pravdivostní tabulky s vyváženými počty hodnot 0 a 1, které mohou být následně aplikovány do návrhů dekompresoru. Tento náhodný způsob hledání vhodných kódů je dle [12] efektivní zejména pro NBC s nižším počtem informačních bitů. S rostoucím počtem roste poté složitost hledání kódu exponenciálně.

Téma sestavování nelineárních blokových kódů popisují také autoři Schmidt a Fišer v publikaci [14]. Jako jedni z prvních popisují suboptimální řešení pro návrh NBC, které popsali pomocí metody CNN (Clique Cover Number), ve volném překladu problém klikového pokrytí. CNN řešením je minimální počet součástí grafu, jejichž podmnožina indukuje kliku. Byla navržena metoda, která zajišťuje nalezení i informačních bitů potřebných pro pokrytí r specifikovaných bitů v rámci n výstupů.

Parametr r představuje 2^r krychlí požadavků pro každou r-tici výstupních bitů v rámci n výstupů. Každá krychle požadavků vyžaduje alespoň jeden výstupní vektor s r specifikovanými bity v rámci n výstupů. Všechny možné hodnoty všech potenciálních r-bitových výstupních vektorů odpovídají úplné množině požadavků. Krychle požadavků lze srovnat s vrcholy grafu a nalézt kompatibilitu s hranami grafů. Existují kompatibilní požadavky, což znamená, že tyto požadavky lze splnit pouze pomocí jednoho výstupního vektoru. Tyto požadavky tvoří kliku. Kompatibilní požadavky mají stejné logické hodnoty korespondující s hodnotami na výstupech. Nalezení CCN řeší konkrétní problém sestavení NBC se zadaným parametrem n a minimální hodnotou i [12] [14].

Toto byl jeden z prvních návrhů přístupu sestavení NBC, který nebyl založen na náhodné metodě. Přínos je zde zejména pro NBC s počtem specifikovaných bitů r=2, kde byly experimentální výsledky velice pozitivní [14]. Jak uvádí dále [12], tato metoda má své limity při vyšším počtu specifikovaných bitů r, kdy roste výpočetní náročnost.

1.2.3 Deterministický přístup generování kódů

V roce 2022 přichází Novák [15] s první deterministickou metodou návrhu NBC, která vychází ze znalostí rozšíření LBC a dosahuje lepších výsledků oproti náhodným metodám z hlediska výpočetní časové náročnosti. Navrhl pravidla pro kódy s různým počtem informačních bitů a počtem specifikovaných bitů *r=3*.

Metoda vychází z rozšíření pravdivostní tabulky pro tři informační bity (8 řádků a 5 sloupců) o jeden sloupec kontrolních bitů. Hodnoty v tomto sloupci musí být zvoleny tak, aby při libovolné kombinací tří sloupců bylo zachováno pokrytí všech tříbitových hodnot. Takto vzniká NBC s hodnotami *n=4*, *i=3* a *r=3*.

1.2.4 Generování kódů pro testování dvou specifikovaných bitů

S konkrétní deterministickou metodou pro generování NBC se dvěma specifikovanými bity přichází Novák [2] v roce 2023. Nyní je zřetelné, že nalezení pravidla pro tvorbu NBC s hodnotou *r=2* je relativně jednoduché. Při nalezení kódu s hodnotou *n=4* vyžaduje postup maximálně 2^{16} kroků, což není náročné na výpočet. Minimální hodnota N pro tento kód je rovna 5, což reprezentuje pět čtveřic, které pokrývají všechny možnosti na výstupech tohoto kódu. Znázorněn je v matici A.

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

Tato matice je jednou z možných podob tohoto kódu se specifikovanými hodnotami. Tento tvar zvolil autor z důvodu snadné verifikace. Vzhledem k této vlastnosti následně rozšiřuje kód zdvojením této matice a přidáním řádků s hodnotami, které chybí v celkovém pokrytí. Kód je znázorněn maticí D.

$$D = \begin{bmatrix} A & A \\ \text{Zeros}_4 & \text{Ones}_4 \\ \text{Ones}_4 & \text{Zeros}_4 \end{bmatrix}$$

Zřetězením matice a bylo dosaženo garance všech možných hodnot na všech dvojicích sloupců. Matici D lze využít pro opakované zřetězení a zajištění pokrytí dvou specifikovaných bitů. Za předpokladu že budou vždy doplněny řádky pro pokrytí chybějících hodnoty. Autor uvádí, že počet zřetězení není omezen, ovšem z hlediska efektivity není tento přístup ideálním.

Dále pro NBC se dvěma specifikovanými bity představuje [2] matice *B*, *neg(B)* a C_{16}. Matice *B* velikosti 4x4 opět obsahuje pokrytí všech požadovaných hodnot. Společně se svou negovanou verzí je zřetězena co matice C_{16}, která byla sestavena opět s doplněním chybějících hodnot.

$$C_{16} = \begin{bmatrix} neg(B) & B & B & B \\ \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 \\ \text{Zeros}_4 & \text{Ones}_4 & \text{Zeros}_4 & \text{Zeros}_4 \\ \text{Zeros}_4 & \text{Zeros}_4 & \text{Ones}_4 & \text{Zeros}_4 \\ \text{Zeros}_4 & \text{Zeros}_4 & \text{Zeros}_4 & \text{Ones}_4 \end{bmatrix}$$

Dle experimentálních výsledků autor potvrdil pokrytí pro NBC se dvěma specifikovanými bity s vyšší efektivitou něž u zřetězení matice D [2].

Na tento výzkum odkazuje Novák ve své navazující publikaci [1], kdy popisuje jednak metodu negace pro NBC s *r=2*, tak i další 2 možnosti, jak tyto kódy vytvářet. Původní metodu pojmenoval strategie negace. Následně navazuje vylepšenou strategií negace, jejíž hlavní přínos je v minimalizaci počtu řádků blokového kódu a tím snaha o snížení počtu potřebných informačních bitů. Z této metody vychází například matice C_{16}. Nejefektivnější strategií pro NBC se dvěma specifikovanými bity je metoda pojmenovaná *n*n*. Tato metoda využívá výše popsanou matici A a její zřetězení do komplexnější podoby, matice C.

$$C = \begin{bmatrix} A & A & A & A \\ A & A^1 & A^2 & A^3 \end{bmatrix}$$

Matice a vytváří kód (4, 2, 2), N=5. Horní index matice a značí stupeň modifikace a říká o kolik sloupců je matice vertikálně rotovaná. Tato sestavený blokový kód zajišťuje pokrytí dvou specifikovaných bitů na všech kombinacích dvojic sloupců. Dle experimentálních výsledků autor popisuje tuto metodu efektivnější oproti strategii negace, ovšem při využití stejných menších bloků kódu. Veškeré zmíněné metody jsou shledány efektivnějšími z hlediska testování při nižších hodnotách oproti využití LBC.

1.2.5 Generování kódů pro testování tří specifikovaných bitů

Autor [2] přichází také s konkrétní metodou generování NBC s hodnotou *r=3*. Tentokrát vychází opět z matic *B* a *neg(B)*, které mají následující tvar:

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} neg(B) = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Zřetězením těchto matic dle horizontální osy získává kód, který na čtyřech sloupcích a osmi řádcích pokrývá hodnoty pro tři specifikované bity. Matice tohoto kódu je označená symbolem F.

Následně autor provádí experimenty s maticemi *B* a *neg(B*, aby docílil rozšíření NBC a zachoval pokrytí pro *r=3*. Představuje matici K_{16}, která vznikla zřetězením

menších matic a doplněním řádku s hodnotami 1, pro zajištění kompletního pokrytí.

$$K_{16} = \begin{bmatrix} \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 \\ \text{neg(B)} & B & B & B \\ B & \text{neg(B)} & B & B \\ B & B & \text{neg(B)} & B \\ B & B & B & \text{neg(B)} \end{bmatrix}$$

Důkaz pro kompletní pokrytí požadovaných hodnot je zahrnut v publikaci [2]. Matice K_{16} je tedy NBC s hodnotami (16, 5, 3) a N=17.

Publikace [2] navazuje na možnost získání NBC se třemi specifikovanými bity rozšířením LBC obdobně jako [15]. S využitím pravidelné struktury LBC je možné přidat nelineární kontrolní bity, které nezmění hodnotu kódového slova. Například LBC s parametry *i=4* a *r=3* obsahuje čtyři kontrolní bity. Tyto bity lze doplnit pomocí pravdivostních tabulek pro všechny možné vstupní XOR funkce.Při analýze pravdivostních tabulek kontrolních bitů LBC lze odvodit následující vlastnosti kódu. Počet jedniček v každém sloupci pravdivostní tabulky je roven počtu nul. Logické hodnoty všech dvojic bitů jsou rovnoměrně distribuovány, což znamená, že každá dvojice sloupců má stejný počet různých logických hodnot. Existují různé případy výskytu čtyřčlenných řádků v pravdivostních tabulkách, což ovlivňuje distribuci logických hodnot a vlastnosti kódu. Z těchto vlastností lze odvodit principy konstrukce nelineárních kontrolních bitů, které jsou založeny na zachování rovnoměrné distribuce logických hodnot a vyvážení počtu jedniček a nul. Pro dosažení tohoto cíle je nezbytné pečlivě vybrat pravdivostní tabulky a správně je kombinovat tak, aby byly splněny všechny požadované vlastnosti kódu.

Dalším přínosným kódem je NBC (136,5,3), který je definován jako LBC s pěti informačními bity, který má pravidelnou strukturu. Každá dvojice bitů v tomto kódu má každou logickou hodnotu přesně osmkrát v rámci pravdivostní tabulky , zatímco každá trojice má každou logickou hodnotu právě na čtyřech řádcích. Délka LBC kódu je 16. Vysoký počet řádků s identickými logickými hodnotami na sloupcích LBC umožňuje větší flexibilitu při umisťování čtyřčlenných skupin do kontrolních bitů [2].

V nejaktuálnějším článku (vzhledem k této práci) představuje Novák [1] dvě plně deterministické metody pro generování NBC se třemi specifikovanými bity. Obě metody jsou založeny na sestavení rozsáhlejších kódů sestavením z menších, snadno ověřitelných. První metoda využívá zřetězení již dříve navržených menších matic a jejich negovaných variant. Vychází z obdobného principu strategie negace, jako u NBC se dvěma specifikovanými bity. Aktuálnější publikace doplňuje poznatky o matici K_{16} a přináší další důkazy o vysoké efektivitě využití této matice pro tvorbu rozsáhlejších NBC se třemi specifikovanými bity.

Druhá metoda využívá taktéž zřetězení matic a jejich modifikací, které jsou nyní prováděny pomocí sloupcových rotací. Strategie nese prozatím označení *n*(n-1)*. Dle dosavadního zhodnocení má tato metoda potenciál být nejefektivnějším způsobem generování NBC s *r=3*, a to i při vyšších hodnotách *n*. Metoda využívá matici K, která obsahuje *n* sloupců a N řádků. Zřetězení matice k je poté provedeno na třech řádcích a patnácti sloupcích. Podmínkou zřetězení je změna jednotlivých matic k po-

mocí sloupcových rotací. Autor v publikaci specifikuje konkrétní důkazy, zejména proč je potřeba, aby se na každém sloupci zřetězení vyskytoval každý tvar rotované matice pouze jednou. Zde vyvstává prostor pro další navázání na toto téma. Autor popisuje úvahy, jakých hodnot by mělo či nemohlo dosahovat n a jakým způsobem jsou rotace realizovány, aby bylo zajištěné pokrytí tří specifikovaných bitů.

Matice K, která má 16 sloupců může vhledem k rotacím nabývat šestnácti různých podob. Jako příklad je uvedená matice K_{240}, která obsahuje 240 sloupců (hodnota n) a využívá 15 různých stupňů rotací matice K.

$$K_{240} = \begin{bmatrix} K & K & K & ... & K & K & K & ... & K & K \\ K & \mathrm{K}^1 & \mathrm{K}^2 & ... & \mathrm{K}^7 & \mathrm{K}^8 & \mathrm{K}^9 & ... & \mathrm{K}^{13} & \mathrm{K}^{14} \\ K & \mathrm{K}^2 & \mathrm{K}^4 & ... & \mathrm{K}^{14} & \mathrm{K}^1 & \mathrm{K}^3 & ... & \mathrm{K}^{11} & \mathrm{K}^{13} \end{bmatrix}$$

Zde bylo prvotním cílem ověřit pokrytí tří specifikovaných bitů na všech kombinacích trojic sloupců. Autor provádí srovnání pro kritické kombinace, například sloupce na indexech 1, 17 a 33. Tyto sloupce jsou vždy prvním sloupcem matice K. V případě, že by nedocházelo k rotacím matice by tyto tři sloupce pokrývaly pouze hodnoty (000) a (111). Díky rotacím a záměnám sloupců tato trojice konkrétně pokrývá všech osm možných hodnot pro tři bity. Obdobné srovnání provádí auto pro všechny další kombinace sloupců.

V tomto bodě nabízí práce prostor pro řadu experimentů s možnostmi zřetězení matic a rozdělením hodnot rotací. Tento přístup a tvar matice K_{240} je jedním z mnoha. Je třeba ověřit, jaký tvar by měla mít matice, která bude řetězena a jakých hodnot může nabývat n. Závěrem publikace je zmínka o možné hardwarové implementaci, která pracuje s realizací obvodu dvouvstupých hradel a možného návrh dekodéru pro praktické využití těchto NBC při testování logických obvodů [1].

1.2.6 Generování kódů pro testování čtyř specifikovaných bitů

Způsob rozšíření kódů pro splnění podmínky $r>3$ lze také realizovat zřetězením maticového kódu a jeho rotacemi. Opět je nutné zaručit, aby všechny r-tice zřetězených matic byly na jednotlivých indexech rozdílné. Autor [1] navrhuje využití matice sestavené na základě pravdivostní tabulky pro čtyři specifikované bity, rozšířený o sloupec paritních bitů pro každý řádek (sudů počet 1 reprezentuje hodnota 0). Tato matice je označena symbolem K_5 a má následující tvar:

$$K_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Jelikož tato matice obsahuje 5 sloupců, je možné provést maximálně 5 sloupcových rotací, a to hodnoty 0 až 4. Navrhovaný blokový kód označený K_{25}, sestavený zřetězením rotací této matice lze navrhnout například takto (údaj horního indexu opět představuje stupeň rotace):

$$K_{25} = \begin{bmatrix} K_5 & K_5 & K_5 & K_5 & K_5 \\ K_5 & K_5^1 & K_5^2 & K_5^3 & K_5^4 \\ K_5 & K_5^2 & K_5^4 & K_5^1 & K_5^3 \\ K_5 & K_5^3 & K_5^1 & K_5^4 & K_5^2 \\ K_5 & K_5^4 & K_5^3 & K_5^2 & K_5^1 \end{bmatrix}$$

Publikace [1] dále popisuje přínos v podobě nárůstu efektivity, využitím kódů tohoto typu (tedy (25, 7, 4) pro N=76) při testování čtyř specifikovaných bitů oproti metodě využívající LBC.

s rostoucím počtem specifikovaných bitů prokazatelně roste složitost využívaných dekompresorů. Navržený postup je stále ale využitelný pro praktickou aplikaci, jelikož i takto sestavený kód zlepšuje kvalitu a efektivitu testů oproti využití LBC dekompresorů [1].

1.3 Diskuze a cíle práce

Vzhledem k situaci, kdy se jedná o relativně krátce zkoumané téma, alespoň z hlediska nejaktuálnějších přístupů, bylo rešerše provedena způsobem zpracování jednotlivých dostupných literárních zdrojů v samostatných kapitolách, které jako celek pokrývají problematiku z několika úhlů pohledu.

Práce navazuje na zdroje publikované profesorem Novákem, který je zároveň vedoucím této práce, a na jehož články práce přímo navazuje. Z tohoto důvodu je dominantním autorem v citované literatuře.

Nejvýznamnějšími zdroji jsou publikace [1] a [2], které jsou přímo navázané na zadání této práce, a proto je jim věnováno nejvíce rešeršního prostoru. Z poznatků uvedených v těchto článcích plyne jednoznačný přínos zkoumané oblasti a možné způsoby navázaní a rozšíření teoretických podkladů, které zde dostávají prostor pro realizaci.

Jak je již specifikováno v zadání práce, významným tématem je využití NBC pro testování obvodů se třemi specifikovanými bity, způsoby sestavování těchto kódů a ověření správnosti navržených přístupů. Klíčové je provést správný postup ověření a v nejlepším případě stanovit obecné podmínky pro postup vytváření takového kódu.

Relativně méně obsáhlým tématem jsou kódy pro čtyři specifikované bity, jimž se věnuje pouze jedna publikace, a které nabízejí prostor pro řadu experimentů, na základě kterých by bylo možné odvodit alespoň základní specifický deterministický přístup sestavování. Toto téma poskytuje širokou variabilitu z hlediska přístupu ověření a následného experimentování s popisem jednotlivých parametrů.

Konkrétními cíli práce je, dle bodů zadání, nalézt přístup a pravidlo pro způsob vytváření NBC pomocí rotací matic. Toto by mělo být dominantní téma ve vlastní části práce. Způsob této realizace je navržen v publikaci [1], zejména v teoretické rovině a proto je zde dostatek prostoru na navázání několika experimentů a ověření způsobu, jakým je možné rotace a zřetězení realizovat. Jedná se například o popis výsledných matic *popis n*(n-1)*, který je třeba přesně popsat a určit pravidlo pro stanovení jednotlivých hodnot.

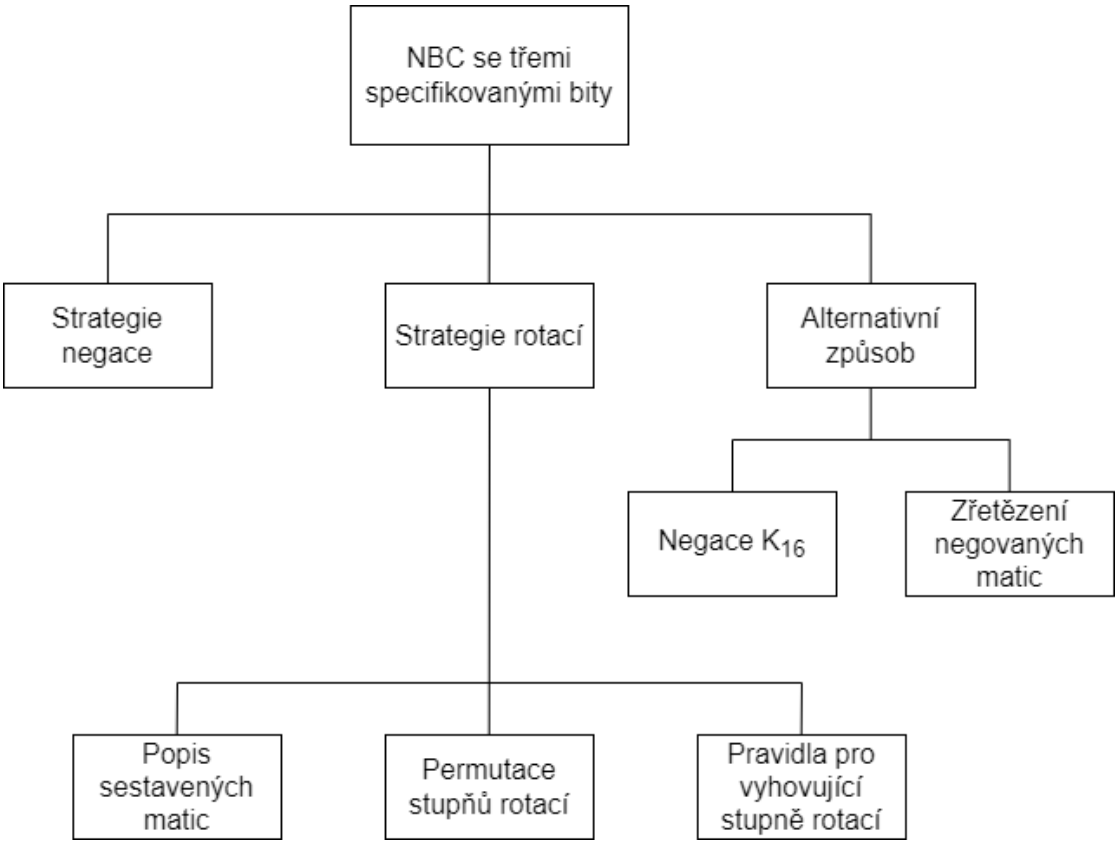
Dále je třeba ověřit parametry NBC pro čtyři specifikované bity. Publikace nabízí jedno možné řešení, které je opět třeba minimálně ověřit a popsat zda a jakým způsobem tento přístup vytváření NBC funguje. V neposlední řadě je cílem práce také přispět v rovině hardwarové realizace a navrhnout možný přístup manuálního sestavení zapojení navržených kódů.

2 Strategie vytváření blokových kódů se třemi specifikovanými bity

Na základě seznámení s problematikou a znalostmi výchozího stavu tématu nyní mohu ověřit, jakým způsobem navržené parametry pro NBC fungují. Dle aktuálně dostupných zdrojů tedy existují dva potenciálně efektivní deterministické přístupy, které poskytují návrh pro sestavení NBC, které je možné využít pro testování logických obvodů při hodnotě specifikovaných bitů $r=3$. Ověření parametrů těchto metod a sestavených kódů lze realizovat provedením triviálního testu pro tři vstupní bity.

Pomocí autorských testovacích skriptů v prostředí MATLAB, je možné otestovat jednotlivé parametry deterministického přístupu a rozhodnout, zda jsou pro využití při testování logických obvodů vyhovující. Nabízí se přístup pro porovnání každé deterministické metody zvlášť a následné porovnání efektivity a využitelnosti pro praktické použití.

Aby byl postup ověřování jednoznačný a systematický, zvolil jsem využití nástroje myšlenkové mapy (Obrázek 2.1), která schématicky popisuje problematiku ověření jednotlivých parametrů a zachycuje možnosti postupu.



Obrázek 2.1: Myšlenková mapa problematiky

Na základě této myšlenkové mapy a znalostí zpracovaných v rešerši jsem zvolil metodu sestavení tří hypotéz, které pokrývají téma testování tří specifikovaných bitů, a dle kterých určuji další postup. Každá hypotéza představuje jednu podkapitolu, které se v tomto zadání věnuji. Jejich znění je následující:

Hypotéza 1: *Lze aplikovat NBC vytvořené strategií negace také při testování obvodů se třemi specifikovanými bity.*

Hypotéza 2: *Existuje více možností realizace NBC pomocí zřetězení rotací matic.*

Hypotéza 3: *Existuje alternativní přístup sestavení NBC s využitím navržených přístupů.*

Pro postup ověření, sestavení NBC, provedení triviálního testu a zobrazení výsledku využiji sadu testovacích skriptů, které usnadní proces ověření využitelnosti kódů. Každá sada programových skriptů bude popsána schématem a slovním popisem, aby bylo jednoznačné, jaký postup sestavení testu byl zvolen a jakým způsobem bylo dosaženo konkrétních výsledků.

Na základě navrženého postupu se lze nyní zaměřit na tři konkrétní oblasti ověřování. První z nich jsou NBC vytvořené strategií negace, následně NBC sestavené rotacemi matice a v poslední části lze vyhradit prostor pro nalezení alternativního

přístupu na základě experimentů. Každou z hypotéz popisuje vždy jedna z následujících kapitol.

2.1 NBC vytvořené strategií negace

Dle dosavadních znalostí vycházím zejména z využití matic B a neg(B), které navrhují publikace předcházející této práci a mají následující tvar:

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} neg(B) = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Pro sloupcové zřetězení těchto matic lze snadno ověřit pokrytí tří specifikovaných bitů, jelikož vycházím z postupu jakým byly vytvořeny. Sestavením pravdivostní tabulky pro tři vstupní bity vzniká kód tří sloupců a osmi řádků, přidáním čtvrtého sloupce, který má tvar lichých paritních bitů, lze získat následující kód (Tabulka 2.1):

Tabulka 2.1: Vytvoření matice B

1	1	1	0
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	0
0	0	0	1

Správnou záměnou několika řádků a rozdělením na dva kódy o velikosti 4x4 jsou získány matice B a neg(B). Přínos těchto dvou matic je z hlediska provádění triviálního testu pro tři specifikované bity jednoznačný.

Splnění podmínek pro pokrytí tří specifikovaných bitů lze provést vizuální kontrolou kódu. Takto lze jednoznačně demonstrovat, jakým způsobem funguje triviální test pro ověření splnění podmínek.

Podmínkou splnění je, aby na každé kombinaci tří sloupců kódu existovalo alespoň jedno řádkové zastoupení pro každou kombinaci hodnot tří bitů. Pro tři bity existuje 8 kombinací uspořádání hodnot 0 a 1, proto musí testovaný kód obsahovat minimálně 8 řádků. Pokud kód dále obsahuje 4 sloupce, existují 4 jejich kombinace. Nejpřehlednějším způsobem pro vizuální kontrolu je vždy vynechání jednoho sloupce a kontrola, zda se na zbývajících vyskytují všechny možné kombinace hodnot pro tři bity. u tohoto kódu je možné jednoznačně určit, že podmínky pokrytí splňuje.

Jak lze dále vyjít z dostupné teorie obě matice lze uspořádat do dvou základních útvarů, které lze následovně využít k rozsáhlejší sestavě NBC. První z nich je matice

F, která je sestavená zřetězením matic B a *neg(B)* a opět představuje rozšířenou pravdivostní tabulku pro tři informační bity a druhou je matice K_{16}, která má 17 řádků a 16 sloupců:

$$F = \begin{bmatrix} B \\ \text{neg(B)} \end{bmatrix}, K_{16} = \begin{bmatrix} \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 & \text{Ones}_4 \\ \text{neg(B)} & B & B & B \\ B & \text{neg(B)} & B & B \\ B & B & \text{neg(B)} & B \\ B & B & B & \text{neg(B)} \end{bmatrix}$$

2.1.1 Triviální test negovaných matic

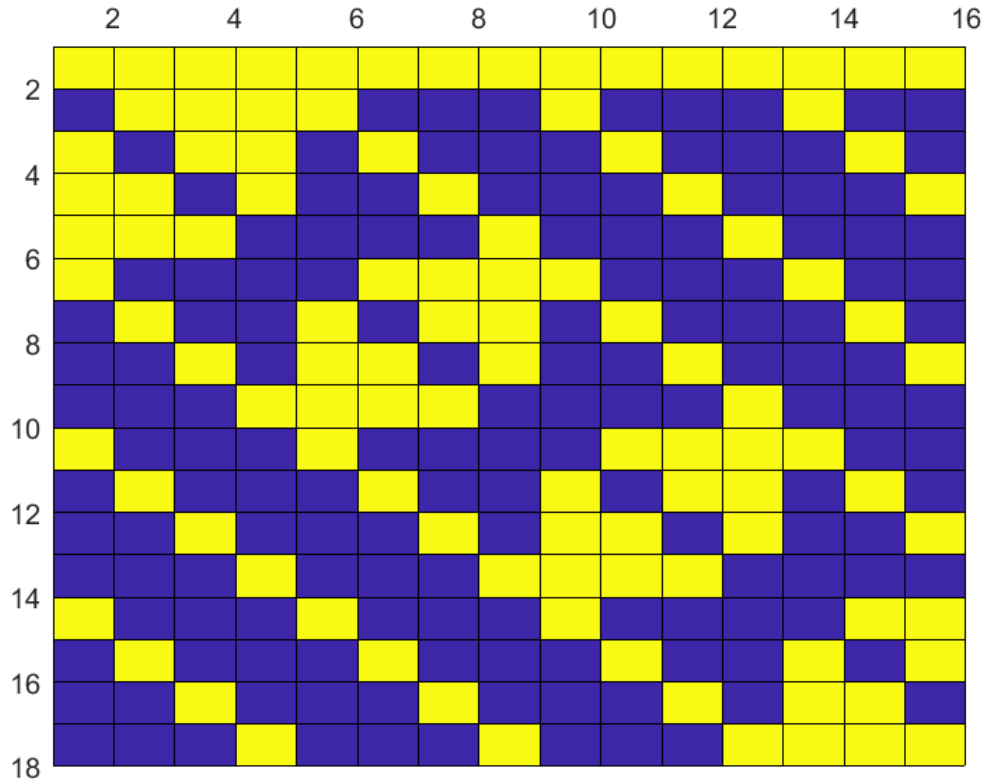
Ověření platnosti triviálního testu pro matici K_{16} jsem provedl pomocí několika skriptů v prostředí MATLAB. Zde jsem implementoval několik skriptů a ověřujících funkcí pro testování a možnost správného ověření potřebných parametrů.

Funkce *sestaveniMatice.m* dle zadaného parametru (K nebo F) vrátí proměnnou ve tvaru konkrétní matice sestaven z hodnot 0 a 1. Tuto hodnotu předává funkci *triviálníTest.m*, která následně dle zadaného parametru (2, 3 nebo 4) provede triviální test obdrženého kódu, tento skript je z hlediska rozboru důležitější.

Způsob provedení testu probíhá v několika krocích. Nejprve vytvořím sadu kombinací 1 a 0, pro počet specifikovaných bitů r, proměnná hodnota obsahuje tedy matici ve tvaru pravdivostní tabulky, v případě hodnoty 3 (pro tři bity) obsahuje tato matice 8 řádků, jeden pro každou hledanou kombinaci. Následuje vytvoření sady kombinací indexů jednotlivých sloupců. Do proměnné ukládám dle zadaného parametru buď dvojice, trojice nebo čtveřice kombinací všech indexů sloupců blokového kódu. Jedná se o kombinace, neboť nezáleží na pořadí sloupců při porovnávání, což významně taktéž dokáže snížit paměťové nároky skriptu. V následném cyklu, jsou porovnám všechny kombinace s pravdivostní tabulkou uloženou v matici, cílem je prokázat zda se každý řádek nachází v každé kombinaci sloupců kódu alespoň jednou. Pro toto ověření jsem využil speciální funkce MATLABu *all* a *ismember*, které výrazně zpřehledňují zdrojový kód skriptu a zefektivňují časovou náročnost testu.

V případě obou matic NBC kódu bylo prokázáno že triviální test s parametrem 3 splňuje požadované podmínky pro počet specifikovaných bitů *r=3*. Ověření je v tomto ohledu jednoznačné a obě matice lze využít k širšímu využití při sestavování NBC , které by mohly být využit pro testování tří specifikovaných bitů.

Matice K_{16} tedy skutečně představuje kód (16, 5, 3) pro *N=17*. graficky ji lze reprezentovat následovně (Obrázek 2.2):



Obrázek 2.2: Matice K_{16} pro kód (16, 5, 3)

2.1.2 Využití negovaných matic pro NBC

Tímto testem lze potvrdit první navrhovanou hypotézu, kdy skutečně lze aplikovat strategii negace také pro vytvoření NBC se třemi specifikovanými bity.

Konkrétní využití matic F a zejména matice K_{16} je naznačeno již ve výchozích publikacích. Díky svému tvaru je matici k možné využít v řádkovém či sloupcovém zřetězení a zaručit platnost $r{=}3$ pro vyšší hodnotu n testovacích vzorků. Stěžejním principem uplatněným při tomto postupu jsou rotace této matice, které pokrytí triviálního testu zaručují, jejich popisem a odvozením, které je hlavním tématem této kapitoly, se zabývá následující část.

2.2 NBC vytvořené strategií zřetězení a rotací matic

V následující podkapitole popisuji, jakým způsobem jsem ověřil platnost triviálního testu pro navrhovanou matici K_{240}, která představuje NBC (240, 6, 3) pro N=49, jaké jsou její hlavní výhody, jakým způsobem lze využít rotace matice k a jaké pro její zřetězení platí podmínky.

Téma rotací je možné rozdělit na tři části, nejprve je třeba ověřit, zda a jakým způsobem funguje triviální test pro matici K_{240}, dále jaké jsou možné hodnoty pro počet zřetězení matice a kolikrát lze matici rotovat a nakonec zda je možné určit obecná pravidla pro zřetězení rotací matic NBC.

2.2.1 Triviální test navržené matice K_{240}

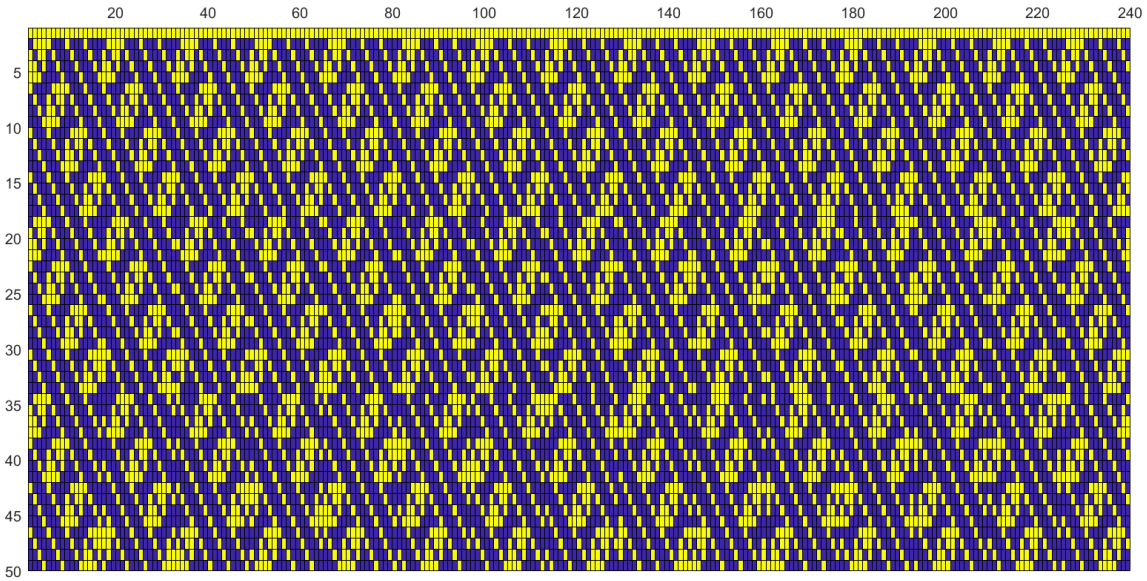
Ověření podmínek, které by měla splňovat tato matice provedu opět pomocí testovacích skriptů v prostředí MATLAB. Navržená matice K_{240}, která představuje NBC (240, 6, 3) má následující zápis:

$$K_{240} = \begin{bmatrix} K & K & K & ... & K & K & K & ... & K & K \\ K & \mathrm{K}^{1} & \mathrm{K}^{2} & ... & \mathrm{K}^{7} & \mathrm{K}^{8} & \mathrm{K}^{9} & ... & \mathrm{K}^{13} & \mathrm{K}^{14} \\ K & \mathrm{K}^{2} & \mathrm{K}^{4} & ... & \mathrm{K}^{14} & \mathrm{K}^{1} & \mathrm{K}^{3} & ... & \mathrm{K}^{11} & \mathrm{K}^{13} \end{bmatrix}$$

V tomto schématu jsou matice K, maticí K_{16} s vynecháním prvního řádku s hodnotami 1, který je zde přidán samostatně pro celou matici K_{240}. Horní index u každého zastoupení matice k představuje stupeň rotace, tj. o kolik sloupců jsou změněné indexy sloupců každé matice. Matice K^1 tedy představuje rotaci o jeden index sloupce vpravo, tedy sloupec na indexu 16 se posouvá na index 1, atd..

Toto zřetězení vychází z podmínky, která vyžaduje, aby na každém sloupci byly stupně zřetězení matice k na všech řádcích rozdílné, mimo první sloupec, ve kterém je matice k vždy v původním tvaru. Dle této podmínky se tedy nemohou vyskytovat stupně rotace na žádném řádku také více než jednou. Dodržením této podmínky dosáhneme schématu, které pokrývá různé kombinace hodnot 0 a 1 a může zajistit pokrytí triviálního testu.

Graficky je tato matice vyjádřena na následujícím obrázku (Obrázek 2.3):



Obrázek 2.3: Matice K_{240} pro kód (240, 6, 3)

K ověření této matice jsem znovu použil již výše uvedený skript *trivialniTest.m*, kterému jsem jako parametr předal matici vytvořenou zřetězením matice k a s příslušnými rotacemi za použití funkce v MATLABu *cicrshift*, která dokáže požadované rotace realizovat. Triviální test pro tři specifikované bit na tomto NBC prokázal, že matice splňuje požadované podmínky.

Nicméně tato navržená matice splňuje podmínky zejména na základě intuitivního návrhu, který vychází ze zmíněné publikace. Zde vyvstává několik hypotéz, kterými hodlám rozšířit téma této problematiky a zobecnit, jaké hodnoty zřetězení jsou validní, pro jaký počet zřetězených matic mohou splňovat podmínky a kolik řádků zřetězení je pro úspěšný test potřeba.

2.2.2 Podmínky pro rotace matice

V současné chvíli tedy známe kód, který vznikl rotacemi matice k a jejím následným zřetězením. Ovšem tento konkrétní způsob interpretace je pouze jeden z mnoha, který mohl být z hlediska počtu různých rotací zvolen. Zde vyvstává úvaha, kterou jsem se řídil během následujících experimentů, která má dvě stěžejní části, za prvé, kolik existuje různých permutací stupňů rotací, které mohou realizovat příslušný NBC a pro jaké počty sloupců zřetězení dokáže vygenerovat matice NBC, které splňuje podmínky pro tři specifikované bity.

Dále samozřejmě vzniká prostor pro úvahu jak tímto způsobem upravit předpis *n*(n-1)*, který navrhuje zmíněná publikace tak, aby odpovídal nalezeným výsledkům. Zde je klíčové správně popsat hodnotu *n*, jakým způsobem ovlivňuje výslednou matici a jakým způsobem určit pravidla pro hledané permutace stupňů rotací.

Pro matici K_{240} platí rovnost kdy po dosazení hodnoty 16 skutečně příslušným předpisem *n*(n-1)* tedy *16*(16-1)* získáme počet sloupců roven 240. Zde ovšem hodnota 16 vystupuje pouze v počtu sloupců matice K, protože stupeň zřetězení je roven

maximálně hodnotě *(n-1)* a počet zřetězených sloupců je roven 15. Z tohoto důvodu jsem si pro experimentální ověření vytvořil substituce a předpis následně upravil. Hodnotu *n-1*, která je jednoznačnější pro popis nahradím symbolem m a hodnotu n, která reprezentuje počet sloupců matice k nahradím samotným číslem 16. Upravený předpis tedy vypadá následovně: *16*m*, kdy m představuje počet sloupců zřetězení matice K. Samozřejmě je předpis nyní dosti konkrétní vzhledem k číslu 16, které vychází z matice K, proto nyní veškeré experimenty vycházejí právě z této matice.

Nyní hledám možné permutace pro hodnoty 1-14, pro 15 sloupců (rotace stupně 0 zůstává vždy v prvním sloupci). První řádek a první sloupec matice reprezentující rotace obsahuje hodnoty 0, druhý řádek od druhého sloupce obsahuje vzestupně seřazené hodnoty 1 až *m-1*. V tomto případě je m rovno 15. Třetí řádek zřetězení by měl obsahovat permutaci těchto hodnot která splňuje dvě klíčové podmínky.

Za prvé na žádném sloupci nesmí být opakující se hodnoty stupně rotace a za druhé musí hodnota stupně splňovat podmínku modula. Tato podmínka eliminuje možnosti, kdy by se v jednom sloupci měly vyskytovat stupně rotací, které nebudou schopné sestavit NBC pokrývající tři specifikované bity. Jsou to případy, kdy by se rozdíly stupně rotací v sousedních sloupcích modulované počtem sloupců matice k (tedy 16) vzájemně rovnaly. V tuto chvíli by nastal kolaps při tvorbě NBC a nedošlo by k sestavení kódu, který pokryje všechny tříbitové možnosti. Došlo by ke zřetězení ekvivalentních sloupců, které by tímto způsobem nedokázaly pokrýt konkrétní hodnoty.

Pro realizaci hledání permutací jsem opět zvolil prostředí MATLAB, ve kterém jsem vytvořil sadu funkcí, které sestaví jednotlivé permutace a ověří platnost podmínek zřetězení rotací. Na základě následujících experimentů zobecňuji pravidla pro zřetězení matice K, tak aby výsledný NBC pokrýval tři specifikované bity.

2.2.3 Popis pravidel pro rotace matice

Ve svých experimentech jsem vycházel z faktu, kdy pracuji se třemi řádkovými zřetězeními matice K_{16}. V prvním řádku blokového kódu jsou zřetězení této matice bez jakýchkoliv úprav v podobě rotací nebo negací. Ve druhém řádku jsou tyto matice rotované vždy o jeden stupeň s rostoucí tendencí. Na základě znalostí pravidel pro stupně rotací ve sloupcích nyní zjišťuji, jakým způsobem lze deterministicky určit stupně rotací pro třetí řádek tohoto blokového kódu. Experimenty jsem začal provádět na matici, která obsahovala 15 sloupcových zřetězení.

$$\begin{bmatrix} K & K & K & ... & K & K \\ K & \mathrm{K}^{1} & \mathrm{K}^{2} & ... & \mathrm{K}^{13} & \mathrm{K}^{14} \\ K & \mathrm{K}^{?} & \mathrm{K}^{?} & ... & \mathrm{K}^{?} & \mathrm{K}^{?} \end{bmatrix}$$

Jednoznačným způsobem pro nalezení hodnot rotací, které by bylo možné umístit do třetího řádku je nalezení permutace, jejíž hodnoty budou odpovídat podmínkám pro jednotlivé sloupce. Z hlediska realizace skriptem v MATLABU byla implementace následující. Do proměnné matice jsem vygeneroval možné permutace hodnot

1 až 14, následně jsem sestavil matici o třech řádcích a patnácti sloupcích s příslušnými hodnotami na známých pozicích a cyklicky přiřazoval každou permutaci hodnot na třetí řádek matice. Nutné bylo zkontrolovat několik podmínek, jednak aby se žádná hodnota na sloupci neopakovala a poté aby rozdíly hodnot mezi jednotlivými sloupci nebyly shodné po operaci *modulo 16*. Po spuštění skriptu následovalo upozornění na využití paměti, kdy MATLAB nebyl schopný časově a paměťově vyhodnotit pole s uloženými možnostmi všech permutací. Přístup jsem tedy upravil a experimentoval na zřetězení sloupců od počtu 2 vzestupně, až do dosažení paměťového limitu. Tato skutečnost přináší první dílčí závěr a to fakt, že pokud nebude deterministická metoda pro hledání NBC navržena správně, tak bude realizace velice složitá z hlediska paměťové náročnosti testovaných obvodů.

Následující experimenty byly o mnoho přínosnější. Postupným přidáváním zřetězených sloupů začalo být zřejmé, jaké hodnoty pro zřetězení a jaké permutace jsou potřeba pro dosažení optimálních výsledků. V případě zřetězení pouze tří sloupců existovala jediná možnost zřetězení a to matice:

$$\begin{bmatrix} K & K & K \\ K & \mathrm{K}^1 & \mathrm{K}^2 \\ K & \mathrm{K}^2 & \mathrm{K}^1 \end{bmatrix}$$

Z hlediska splnění podmínek a triviálního testu pokrytí tří bitů bylo vše splněno, ovšem tato matice není zdaleka tak efektivní v počtu testovacích vzorků jako matice K_{240}. Proto bylo nutné sloupce zřetězení navýšit. V případě čtyř sloupců nebyla nalezená permutace, která by podmínkám odpovídala, znázornění výsledku je zachyceno na obrázku (Obrázek 2.4).

```
K =

     0     0     0     0
     0     1     2     3
     0     0     0     0
     0     0     0     0
     0     0     0     0

TEST FAILED
>>
>>
```

Obrázek 2.4: Permutace sloupců 4

Postupným přidáváním sloupců jsem zjistil že existuje mnoho permutací, které splňují podmínky pro zřetězení při využití lichého počtu sloupcového zřetězení. Efektivní tady byly například možnosti s hodnotami 0-6 nebo 0-12. V takových případech existovalo řádkových zřetězení splňující dané podmínky hned několik. Popsány jsou na obrázcích níže (Obrázek 2.5) (Obrázek 2.6).

```
M =

     0     0     0     0     0     0     0
     0     1     2     3     4     5     6
     0     6     5     4     3     2     1
     0     5     3     1     6     4     2
     0     4     1     5     2     6     3
     0     3     6     2     5     1     4
     0     2     4     6     1     3     5

OK
TEST SUCCEDED
fx >>
```

Obrázek 2.5: Permutace sloupců 7

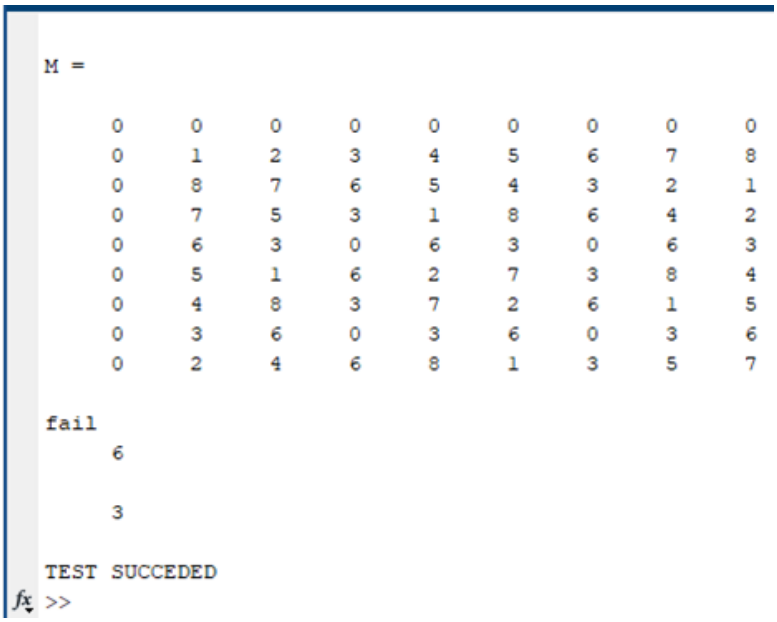
```
M =

     0     0     0     0     0     0     0     0     0     0     0     0     0
     0     1     2     3     4     5     6     7     8     9    10    11    12
     0    12    11    10     9     8     7     6     5     4     3     2     1
     0    11     9     7     5     3     1    12    10     8     6     4     2
     0    10     7     4     1    11     8     5     2    12     9     6     3
     0     9     5     1    10     6     2    11     7     3    12     8     4
     0     8     3    11     6     1     9     4    12     7     2    10     5
     0     7     1     8     2     9     3    10     4    11     5    12     6
     0     6    12     5    11     4    10     3     9     2     8     1     7
     0     5    10     2     7    12     4     9     1     6    11     3     8
     0     4     8    12     3     7    11     2     6    10     1     5     9
     0     3     6     9    12     2     5     8    11     1     4     7    10
     0     2     4     6     8    10    12     1     3     5     7     9    11

OK
TEST SUCCEDED
fx >>
```

Obrázek 2.6: Permutace sloupců 13

Zajímavou výjimkou byla hodnota pro devět sloupcových zřetězení. V tomto případě existovaly možnosti, které splňovaly podmínky triviálního testu pro pokrytí, ovšem na konkrétních řádcích, které byly svázané se soudělností s hodnotou 3 docházelo k duplikaci hodnot, které plné pokrytí nezajišťovaly. Výstup z experimentu popisuje obrázek (Obrázek 2.7).



Obrázek 2.7: Permutace sloupců 9

Tento tvar zřetězení při dosazení správných hodnot splňuje podmínky pokrytí triviálního testu pro tři specifikované bity a tento přístup z hlediska strategií rotací je doposud nejefektivnějším přístupem pro realizaci NBC se třemi specifikovanými bity.

Nicméně nejpřínosnějším poznatkem bylo nalezení zjevného pravidla vždy v posledním řádku možností rotací. poslední permutace (vzhledem k uspořádání testování od nejvyšších hodnot) vždy obsahovala stejný vzorec pro určení hodnot. Konkrétně se jedná o růst mezi sloupci o hodnotu 2 a to nejprve vzestupně přes sudé hodnoty a při vyčerpání možností přes liché. Všechny tyto možnosti jsem následně ručně připsal do požadovaného tvaru matice a podrobil triviálnímu testu pro pokrytí. Zde byla ušetřena výpočetní doba z hlediska hledání permutací a veškeré hodnoty pro zřetězení byly prokázány jako úspěšné z hlediska pokrytí tří bitů. Takto bylo možné nalézt hodnoty až do počtu zřetězení 13 sloupců z důvodů paměťové náročnosti.

Zde je nutné správně popsat vztahy mezi sloupci a stanovit přípustné hodnoty. Pravidla pro zřetězení a splnění podmínek platila pro liché počty sloupcových zřetězení, ovšem za neměnného stupně rotace v prvním sloupci. Hledané permutace tedy probíhaly na sudých hodnotách. Tento fakt je zcela logický, protože aby mohlo dojít ke zřetězení zároveň se splněním podmínek, musí být řád permutace dělitelný beze zbytku dvěma, aby bylo možné dosáhnout stejného poměru rozdělení na sudé a liché stupně rotací.

Zde je možné vycházet z poslední hodnoty rotace na prostředním řádku zřetězení, protože tato hodnota je z hlediska stupně rotací maximem. Tato hodnota bude vždy o 1 nižší než počet sloupcových zřetězení. Dle substitučního značení by hodnota m měla být sudá, stejně tedy tak jako hodnota n-1. Hodnota 16 zůstává stála,

pokud je pro zřetězení využitá matice K_{16}. Hodnota n představuje celkový počet sloupcových zřetězení.

Dle nalezeného výsledku jsem poté mohl ručně sestavit také hodnoty třetího řádku pro vyšší počty sloupců. Z důvodu nutnosti splnění podmínky *modulo 16* jsem prokázal že vyšší počet zřetězení sloupců nezaručí v žádném z případů pokrytí tři specifikovaných bitů, v každé možnosti dochází buď ke ztrátě kombinací nebo nadbytku nevyhovujících vzorků. Touto metodou tedy bylo navrženo obecné pravidlo pro dosazování hodnot do třetího řádku zřetězení matice. Tímto způsobem vznikla taktéž matice K_{240} a její teoretický návrh a podoba taktéž prokazují, že tato strategie navrhování NBC se třemi specifikovanými bity je nastavená správným způsobem.

Ve zobecnění předpisu *n*(n-1)* nebo upravené podobě *m*16* je nutné aby hodnota m bylo liché číslo s minimální hodnotou 3 a maximální hodnotou 15. Tímto lze získat finální obecnou podobu matice ve tvaru:

$$K_{n*(n\text{-}1)} = \begin{bmatrix} K & K & K & ... & K & K & K & ... & K \\ K & \mathrm{K}^{1} & \mathrm{K}^{2} & ... & ... & ... & ... & ... & \mathrm{K}^{n\text{-}2} \\ K & \mathrm{K}^{2} & \mathrm{K}^{4} & ... & \mathrm{K}^{n\text{-}2} & \mathrm{K}^{1} & \mathrm{K}^{3} & ... & \mathrm{K}^{n-3} \end{bmatrix}$$

V tomto případě pro sjednocení předpisu pro rotace matic a určení hodnoty n sloupců navrhuji následné úpravy. Zde již není třeba využívat substituce za hodnotu n. Navržený předpis *n*(n-1)* upravíme do podoby *16*(n-1)* pro liché hodnoty n. Kdy hodnota 16 představuje počet sloupců v jednotlivých zřetězeních a n počet zřetězených sloupců.

Hodnota *n-1* musí být sudé číslo z důvodu poměrového dělení řádu permutace. Číslo n je tedy liché číslo značící počet sloupcových zřetězení.

Pravidlem pro doplnění třetího řádku zřetězených matic je tedy vzestupné seřazení lichých hodnot a poté sudých. Hodnota n ovšem nesmí přesáhnout číslo 15. Hodnoty 17 výš neposkytují splnění parametrů zřetězení, neboť zde nastává kolize zbytků po dělení 16 a dochází k duplikátům indexů a kolapsům při triviálním testu.

V přesnějším zápisu lze tedy vyjádřit, že n je kongruentní 1 modulo 2 v intervalu od 3 do 15: $n \equiv 1 \pmod 2$; $n \in [3, 15]$.

2.3 Alternativní možnost sestrojení NBC

V následující části popisuji možnosti pro návrh odlišného přístupu pro nastavení parametrů generování NBC. Existuje několik možností, které je možné na základě pokusné metody ověřit a určit zda jsou pro toto téma přínosné.

Opuštěním metody rotací zbývá jen málo alternativních možností, kterými lze sestavit NBC matice. Klíčovou myšlenkou, která je podchycená v teorii tématu a také o kapitolu výš je stále nutné splnit podmínku neopakujícího se indexu sloupce ve řetězeném blokovém kódu. Přínos systému rotací matice K_{16} je v tomto ohledu jednoznačný. Nabízí se ovšem možnost nalezení jiného přístupu, kterým by bylo možné, za pomocí experimentů, dosáhnout obdobného výsledku výsledku.

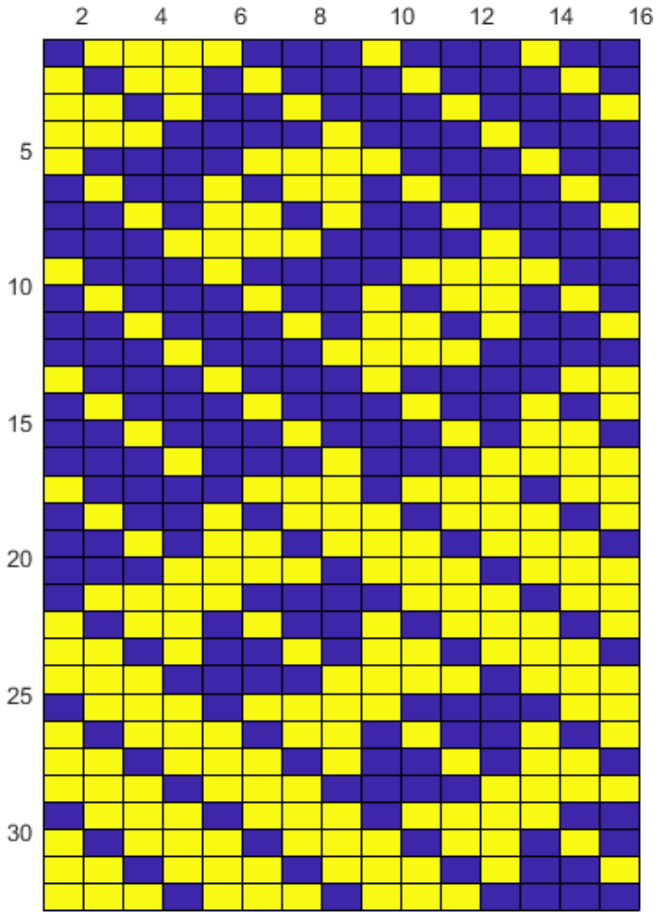
Na základě úvahy a experimentů formuluji následující myšlenku, kterou lze dojít ke konkrétnímu závěru, zda existuje efektivnější přístup pro generování NBC splňujících podmínku pro tři specifikované bity *r=3*. Jedná se o využití matice k a její negace a tyto dva kusy kódu experimentálně zřetězit a zhodnotit dosažené výsledky.

2.3.1 Negace matice K_{16}

První možností je využití matice K_{16} s přidaným řádkem s hodnotami 1 na všech pozicích, tím je zachována podmínka splnění triviálního testu. Následnou negací této matice a sloupcovým zřetězením je možné získat taktéž zajímavý kus nelineárního kódu, se kterým je možné provádět další experimenty. Vhodnou úpravou je poté odebrání řádku s hodnotami 1, jelikož matice K_{16} a její negovaná verze společně triviální test tří bitů také splňují. Následující schéma popisuje postup úvahy a výsledný tvar:

$$\begin{bmatrix} \text{Ones}_{16} \\ K \end{bmatrix} -- > \begin{bmatrix} \text{Ones}_{16} \\ K \\ K' \end{bmatrix} -- > \begin{bmatrix} K \\ K' \end{bmatrix}$$

Tímto způsobem tak vzniká NBC kód (16, 5, 3) s hodnotou N=32, pro který platí podmínky triviálního testu pro *r=3*. Tento kód lze využít jako výchozí stav pro následující experimenty a s jeho pomocí nalézt a popsat jinou možnost přístupu pro generování NBC. Matice tohoto kódu je vyobrazená na obrázku níže *(Obrázek 2.8)*.



Obrázek 2.8: K, neg(K)

V případě rotací existuje možnost zřetězení až do počtu 16 sloupců se stupni rotaci 0-15, díky vlastnosti matice k (má 16 sloupců). Experimenty ukázaly, že efektivnější zřetězení v případě negace matice k s vysokou pravděpodobností navrhnout nelze. V případě stanoveného cíle, maximalizace počtu sloupců při nižším růstu počtu řádků kódu se lze dostat pouze na hodnotu tří zřetězení, tak aby aby byla zachována efektivita kódu.

Při využití znalostí z aplikace rotací matic jsem sestavil matici 33x32, která využívá matici k a její negovanou variantu, schéma vypadá takto a obsahuje navíc řádek hodnot 1:

$$\begin{bmatrix} \text{Ones}_{16} & \text{Ones}_{16} \\ K & K \\ K & neg(K) \end{bmatrix}$$

Tato matice po přidání řádku s hodnotami 1 splňuje podmínky triviálního testu pro *r=3*, kód se tedy značí (32, 5, 3) při N=33. Takto vypadající kód zajišťuje sice vyšší efektivitu oproti využití LBC ovšem ne natolik vysokou oproti systému rotací. Jak jsem uvedl v předchozí kapitole, aby nebylo nutné ke kódu přidávat samostatné řádky, které by pokryly chybějící hodnoty triviálního testu, je vhodné pro kódy se

třemi specifikovanými bity sestavit tři řádky zřetězení matice K. Možností které by mohly navázat na předchozí návrh existuje hned několik. Intuitivními přístupy je rozšíření o negovanou matici k na dalším řádku a sloupci nebo rozšíření a následná záměna negací, zde uvádím sice neúspěšné, ale nejefektivněji vypadající modely označeny M1 a M2:

$$M1 = \begin{bmatrix} K & K & K \\ K & neg(K) & neg(K) \\ K & neg(K) & neg(K) \end{bmatrix} M2 = \begin{bmatrix} K & K & K \\ K & K & neg(K) \\ K & neg(K) & neg(K) \end{bmatrix}$$

Matice M1 nesplňuje podmínky triviálního testu z téměř poloviny (polovina kombinací tří bitů není pokrytá), ovšem matice M2 postrádá pouze 2 kombinace a to hodnoty 101 a 010. Přidáním dvou řádků do této matice jsem dosáhl pokrytí triviálního testu pro tři specifikované bity. Matice (nyní M2a) by vypadala takto:

$$M2a = \begin{bmatrix} K & K & K \\ K & K & neg(K) \\ K & neg(K) & neg(K) \\ \text{Ones}_{16} & \text{Zeros}_{16} & \text{Ones}_{16} \\ \text{Zeros}_{16} & \text{Ones}_{16} & \text{Zeros}_{16} \end{bmatrix}$$

Z pohledu efektivity tato matice není dramaticky významným přínosem pro NBC pokrývající tři specifikované bity, ovšem v případě nízkého požadovaného množství testovacích vzorků je vhodnou alternativou při požadavcích na nižší náročnost hardwarového zapojení, kterému se dále věnuji v následujících kapitolách. Lze tedy říci, že i tento způsob sestavení NBC má svůj přínos a využití.

Ačkoliv dva přidané řádky k této matici nezvyšují počet potřebných informačních bitů, pokusil jsem se dalším krokem experimentu získat tvar matice, který má pouze tři řádkové i sloupcové zřetězení matice tvarů matice K_{16}, bez potřeby ji jakkoliv dále rozšiřovat. Tato matice má aktuálně 49 řádků a 48 sloupců, reprezentuje kód (48, 5, 3) při N=49 a značí se M:

$$M = \begin{bmatrix} K & neg(K) & neg(K) \\ neg(K) & K & neg(K) \\ neg(K) & neg(K) & K \end{bmatrix}$$

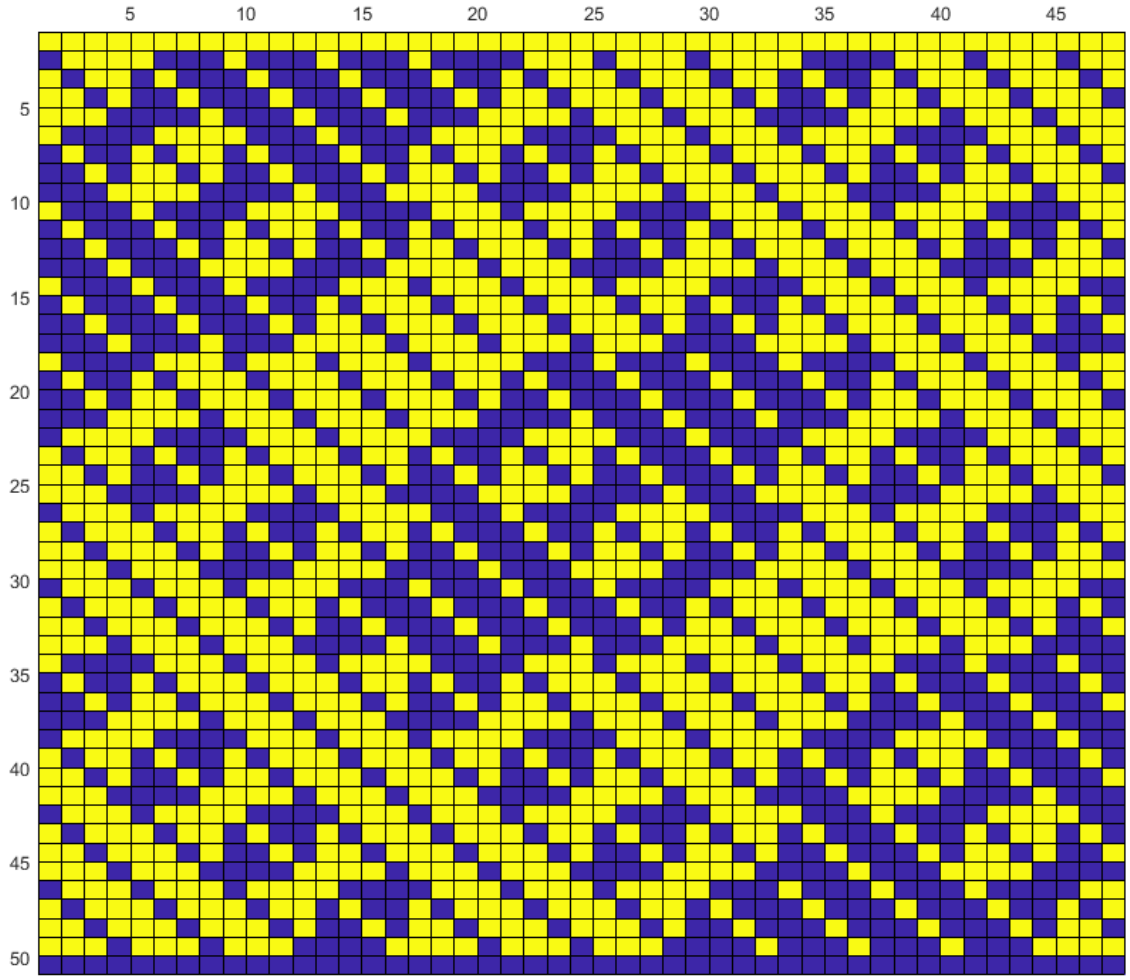
Tento experiment vychází z analogického přístupu, který byl použit ve výchozích publikacích pro vytvoření samotné matice K. Z pohledu problematiky má tato matice tedy dva hodnotné přínosy. Jednak poskytuje již zmíněnou alternativu pro realizace NBC, která by mohla zajisti nižší HW náročnost (ovšem s nižším počtem vzorků) a následně slouži toto schéma jako doplňující ověření správného přístupu sestavení matice k a celkově návrhu přístupu pro NBC se třemi specifikovanými bity.

Bohužel z hlediska splnění podmínek triviálního testu pro kódy se třemi specifikovanými bity je tato matice nedostačující. Experimentem jsem zjistil, že sada kombinací postrádá dá hodnoty *111* a také *000*. Obdobně podle strategie rotací je nutné přidat řádky pokrývající tyto hodnoty. Z estetického hlediska je řádek pokrývající hodnoty 1 přidán na první pozici a řádek s hodnotami 0 na poslední pozici.

Matice M má výsledný tento tvar:

$$M = \begin{bmatrix} \text{Ones}_{16} & \text{Ones}_{16} & \text{Ones}_{16} \\ K & neg(K) & neg(K) \\ neg(K) & K & neg(K) \\ neg(K) & neg(K) & K \\ \text{Zeros}_{16} & \text{Zeros}_{16} & \text{Zeros}_{16} \end{bmatrix}$$

A v grafické interpretaci (Obrázek 2.9) vypadá matice M následovně:



Obrázek 2.9: Matice M

Závěrem mohu shrnout, že alternativní přístup pro tvorbu NBC se třemi specifikovanými bity existuje, tím je potvrzena třetí hypotéza. Přístup je využitelný například při nižších hardwarových nárocích, ovšem za cenu nižší efektivity v podobě menšího počtu testovacích vzorků. Přístup, který by využíval pouze tvary matice K_{16} existuje, ovšem nejvyšším přínosem byl tvar matice M, který vyžívá řádky s hodnotami 1 a 0. Z hlediska hardwarové realizace je to jednodušší podání oproti matici *M2a*, kdy by návrh realizace byl kvůli střídání hodnot v posledních řádcích složitější.

3 Strategie deterministického vytváření kódů pro čtyři specifikované bity

3.1 Blokové kódy pro čtyři specifikované bity

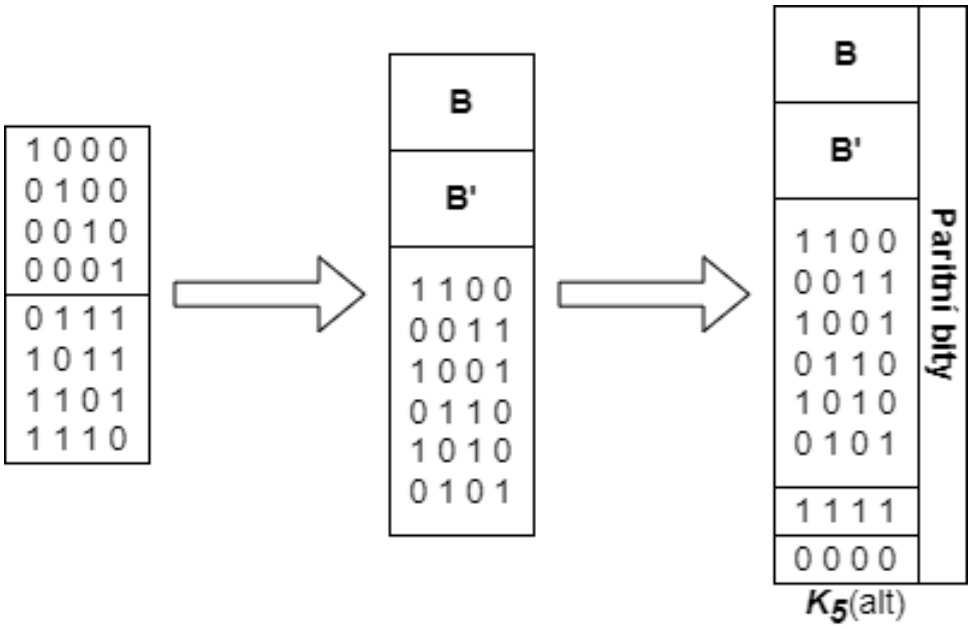
Téma NBC, které kompletně pokrývají čtyři specifikované bity je stále velice málo prozkoumané a může být obtížné stanovit konkrétní závěry na základě dostupných zdrojů. Tato kapitola pojednává o současně navrhovaných možnostech přístupu generování takových kódů a rozvíjí možnosti, jakým způsobem v této oblasti lze dále v experimentech postupovat.

3.1.1 Matice K_5

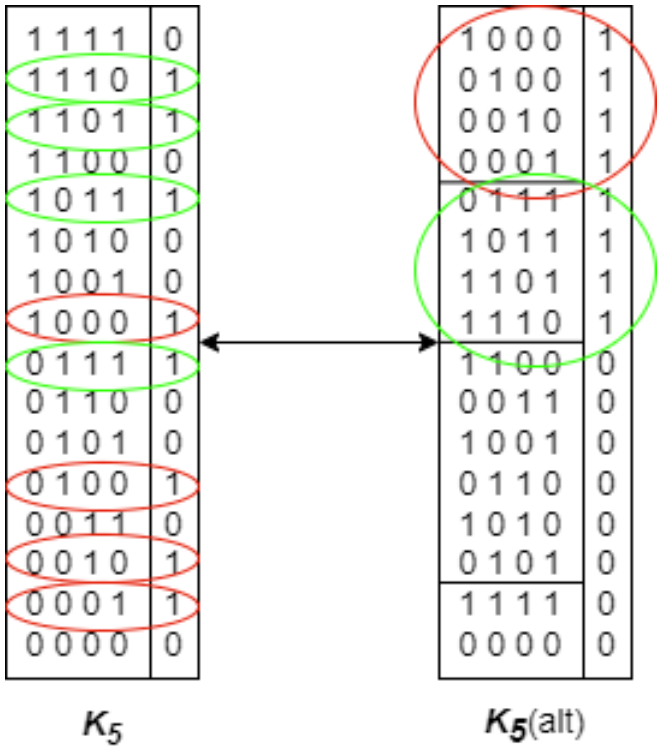
Výchozím přístupem, který je třeba ověřit je již výše zmíněná matice s označením K_5, která vznikla rozšířením pravdivostní tabulky pro čtyři bity (16 řádků) o sloupec paritních bitů. V tomto případě byla použita sudá parita pro doplnění pátého sloupce, ovšem možností je využít taktéž paritu lichou a zachovat pokrytí triviálního testu. Tímto způsobem vzniká blokový kód, který nabízí 5 kombinací, na kterých je třeba ověřit splnění podmínek čtyř specifikovaných bitů. Vzhledem k relativně menšímu rozměru matice jsem tuto kontrolu provedl vizuálně. Samozřejmě lze skutečnost prokázat opět využitím skriptu *trivialniTest.m* s parametrem pro kontrolu kombinací na čtyřech bitech. Matice K_5 tedy po kontrole skutečně podmínkám odpovídá.

$$K_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Obdobným přístupem, který je možné zmínit je sestavení této matice úpravou hodnot kódů pro tři specifikované bity a jejich rozšířením. Využitím matic B a neg(B), přidáním řádků s chybějícími hodnotami a rozšířením o stejný sloupec s paritními bity jsem dosáhl totožného výsledku. Postup je zachycen na následujících schématech (Obrázek 3.1) a (Obrázek 3.2).



Obrázek 3.1: Alternativní vytvoření matice K_5



Obrázek 3.2: Souvislosti matice K_5 a alternativy

Tuto možnost sestrojení NBC zde udávám z důvodu možného navázání při vytváření NBC pro pět specifikovaných bitů. Je to možnost, kterou lze využít při sestavení matice způsobem rozšíření o chybějící hodnoty a paritní bity.

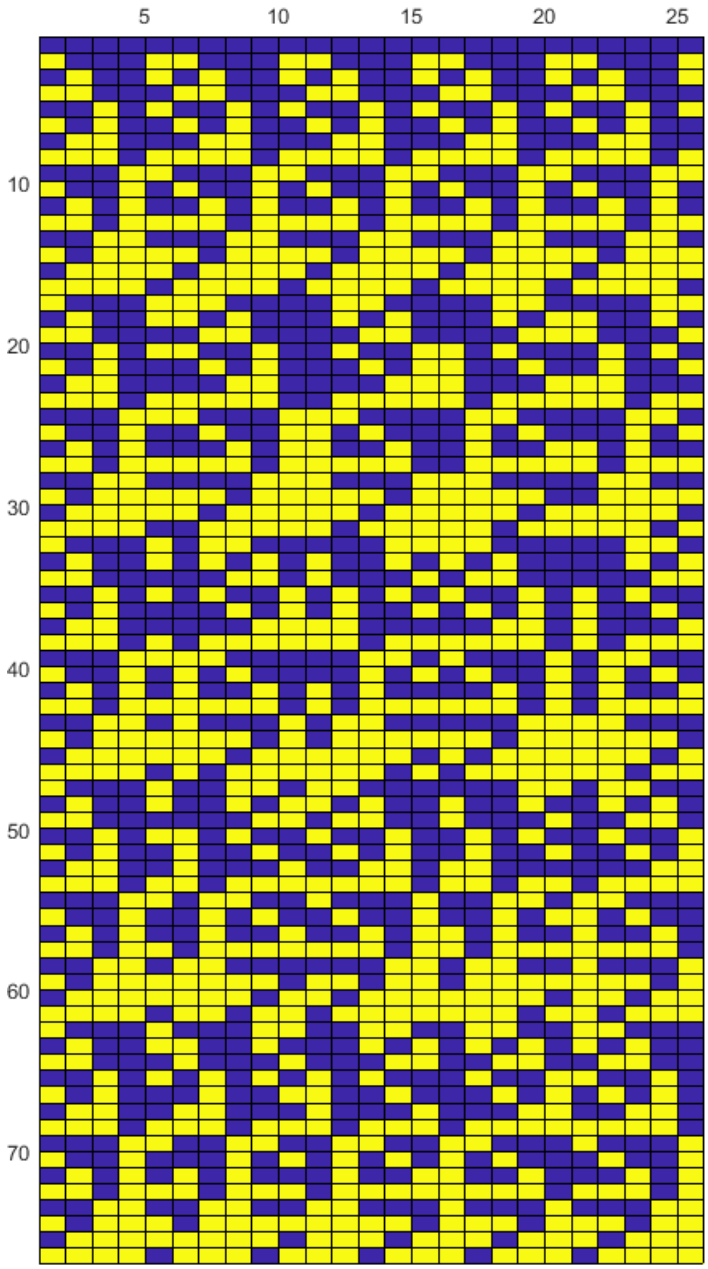
3.1.2 Triviální test matice K_{25}

Následně zkoumaným kódem je NBC, který vznikl zřetězením matice K_5 a měl by být schopný pokrýt triviální test pro čtyři specifikované bity. NBC řetězí matici K_5 v pěti řádcích a pěti sloupcích, s tím že zachovává podmínky rotací stejným způsobem jaký byl využit při řetězení matice k při *r=3*. Vzhledem ke skutečnosti, že matice K_5 má pouze pět sloupců, je možné provést pouze takový počet rotací, tedy s hodnotami 0-4. Z tohoto důvodu není možné matici sloupcově zřetězit vícekrát. Zároveň musí být zachovány podmínky, aby se na žádném sloupci nevyskytovaly shodné stupně rotací. Proto má matice, nesoucí označení K_{25} následující tvar:

$$K_{25} = \begin{bmatrix} K_5 & K_5 & K_5 & K_5 & K_5 \\ K_5 & K_5^1 & K_5^2 & K_5^3 & K_5^4 \\ K_5 & K_5^2 & K_5^4 & K_5^1 & K_5^3 \\ K_5 & K_5^3 & K_5^1 & K_5^4 & K_5^2 \\ K_5 & K_5^4 & K_5^3 & K_5^2 & K_5^1 \end{bmatrix}$$

S využitím MATLAB funkce *circshift* jsem sestavil příslušnou matici a aplikováním skriptu *trivialniTest.m*, tentokrát s parametrem *r=4* jsem prokázal, že tato matice

splňuje podmínky pro pokrytí čtyř specifikovaných bitů a reprezentuje NBC ve tvaru (25, 7, 4) při N=76. Graficky tento kód vypadá následovně (Obrázek 3.3):



Obrázek 3.3: Kód (25, 7, 4), N=76

Pro následující experimenty vyvstává zejména následující úvaha, která se zabývá počtem řádků zřetězení, které je potřeba pro pokrytí čtyř specifikovaných bitů. Navržený kód využívá pět řádků zřetězené matice K_5. Další úvahou, kterou následně popisuji je způsob vytvoření jiného NBC, který pokrývá čtyři specifikované bity a to buď s použitím původní matice k pro tři specifikované bity a nebo matice B a neg(B), které vycházejí z pravdivostní tabulky pro tři specifikované bity.

Při experimentu s využitím pouze čtyř řádkových zřetězení jsem nenalezl vhodnou kombinaci matice K_5 tak, abych dosáhl plného pokrytí čtyř specifikovaných bitů. V tuto chvíli lze tedy vycházet z předpokladu, že k pokrytí čtyř specifikovaných bitů s využitím matice K_5 je nutné ji zřetězit minimálně na pěti řádcích.

4 Hardwarová náročnost nalezených řešení

4.1 Popis navržených matic K_{16} a K_{240}

Na základě specifikovaných požadavků dle zadání je klíčový popis již navržené matice K_{240}, která je sestavena na principu rotací a zřetězení matice K_{16}. Důležitý je popis pomocí logických funkcí, zejména s využitím funkce nonekvivalence (XOR), jednak pro matici K_{16}, tak i pro celou matici K_{240}.

Na základě logického popisu lze následně nakreslit schéma hardwarového zapojení a určit počet potřebných hradel pro fyzickou realizaci. Cílem je rozlišit statický počet hradel, takový aby setrval konstantní při opakovaném využití NBC matice K_{16} a následný dynamický počet, který se mění individuálně dle hodnoty n a způsobu sestavení zřetězení.

4.1.1 Popis matice K_{16} pomocí logických výrazů

Realizaci popisu matice K_{16} jsem provedl pomocí pravdivostní tabulky se čtyřmi logickými vstupy a každý sloupec matice jsem na ni aplikoval jako předepsanou funkci. Díky tomu vzniká 16 logických výrazů, které lze následně upravit a určit pravidlo pro logický popis této matice. Matice K_{16} je ve tvaru 16 řádků a 16 sloupců, bez započtení prvního řádku obsahující pouze hodnoty 1. Prvním důvodem této úpravy je jednodušší realizace popisu, protože postačují pouze čtyři vstupní bity a druhým důvodem je vzhled matice K_{240}, který taktéž ve zřetězení tyto řádky vynechává. Z této úvahy plyne jasná možnost optimalizace, kterou popisuji v následujících podkapitolách. Tvar matice K_{16} je nyní následující:

$$K_{16} = \begin{bmatrix} neg(B) & B & B & B \\ B & neg(B) & B & B \\ B & B & neg(B) & B \\ B & B & B & neg(B) \end{bmatrix}$$

Následující obrázek (Obrázek 4.1) zachycuje postup pro logický zápis prvního sloupce matice K_{16}.

x_4	x_3	x_2	x_1	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

$$f = \overline{x}_4\overline{x}_3\overline{x}_2x_1 + \overline{x}_4\overline{x}_3x_2\overline{x}_1 + \overline{x}_4\overline{x}_3x_2x_1 + \overline{x}_4x_3\overline{x}_2\overline{x}_1 + x_4\overline{x}_3\overline{x}_2\overline{x}_1 + x_4x_3\overline{x}_2\overline{x}_1 =$$

$$\overline{x}_4\overline{x}_3(\overline{x}_2x_1 + x_2\overline{x}_1 + x_2x_1) + \overline{x}_2\overline{x}_1(\overline{x}_4x_3 + x_4\overline{x}_3 + x_4x_3) =$$

$$(\overline{x}_4 + \overline{x}_3)(x_2 + x_1) + (\overline{x}_2 + \overline{x}_1)(x_4 + x_3) =$$

$$(x_4 + x_3) \oplus (x_2 + x_1)$$

$$e_1 \qquad a_1$$

první sloupec: $e_1 \oplus a_1$

Obrázek 4.1: Logický popis prvního sloupce

Postup vyplývá z pravidel pro úpravy logických výrazů, pro čtyři sloupce představující vstupní bity je pátý sloupec s předepsanou funkcí, který kopíruje hodnoty v prvním sloupci matice K_{16}. Výraz, který je třeba upravit se skládá ze šesti součinů jednotlivých vstupů dle hodnot 1 v předepsané funkci (hodnoty 0 není třeba brát v potaz). Takto vypadá popis prvního sloupce ve tvaru součtu součinů (logických). Následovně jsem provedl několik úprav, abych dodržel požadavek na popis matice pomocí logické funkce nonekvivalence (XOR). Postupným vytýkáním a krácením jsem získal výsledný tvar, který požadovanou funkci obsahuje.

Logický výraz pro první sloupec po náležitém upravení nyní tedy vypadá takto: $(x_4 + x_3) \oplus (x_2 + x_1)$. Analogickým postupem jsem provedl takový popis pro všechny zbývající sloupce. Zde bylo jasně patrné pravidlo pro tvary jednotlivých částí logických výrazů a popis matice se stal jednoznačně intuitivní záležitostí. Pro zjednodušení a zpřehlednění jsem provedl jednoznakové substituce za jednotlivé části těchto výrazů., které jsou představeny v následující tabulce (Tabulka 4.1):

Tabulka 4.1: Substituce logických výrazů

$(x_2 + x_1)$	a	$(x_4 + x_3)$	e_1
$(x_2 + x_1')$	b	$(x_4 + x_3')$	e_2
$(x_2' + x_1)$	c	$(x_4' + x_3)$	e_3
$(x_2' + x_1')$	d	$(x_4' + x_3')$	e_4

Následně popisuji všechny sloupce pomocí těchto substitucí a vzniká 16 logických výrazů, které je možné následně zakreslit do schématu hardwarového zapojení. Následující tabulka (Tabulka 4.2) představuje zápis pro každý sloupec, značený indexem sloupce, dále následuje výraz a v posledním sloupci tabulky je symbol označující výstup, který bude přiřazen ve schématu:

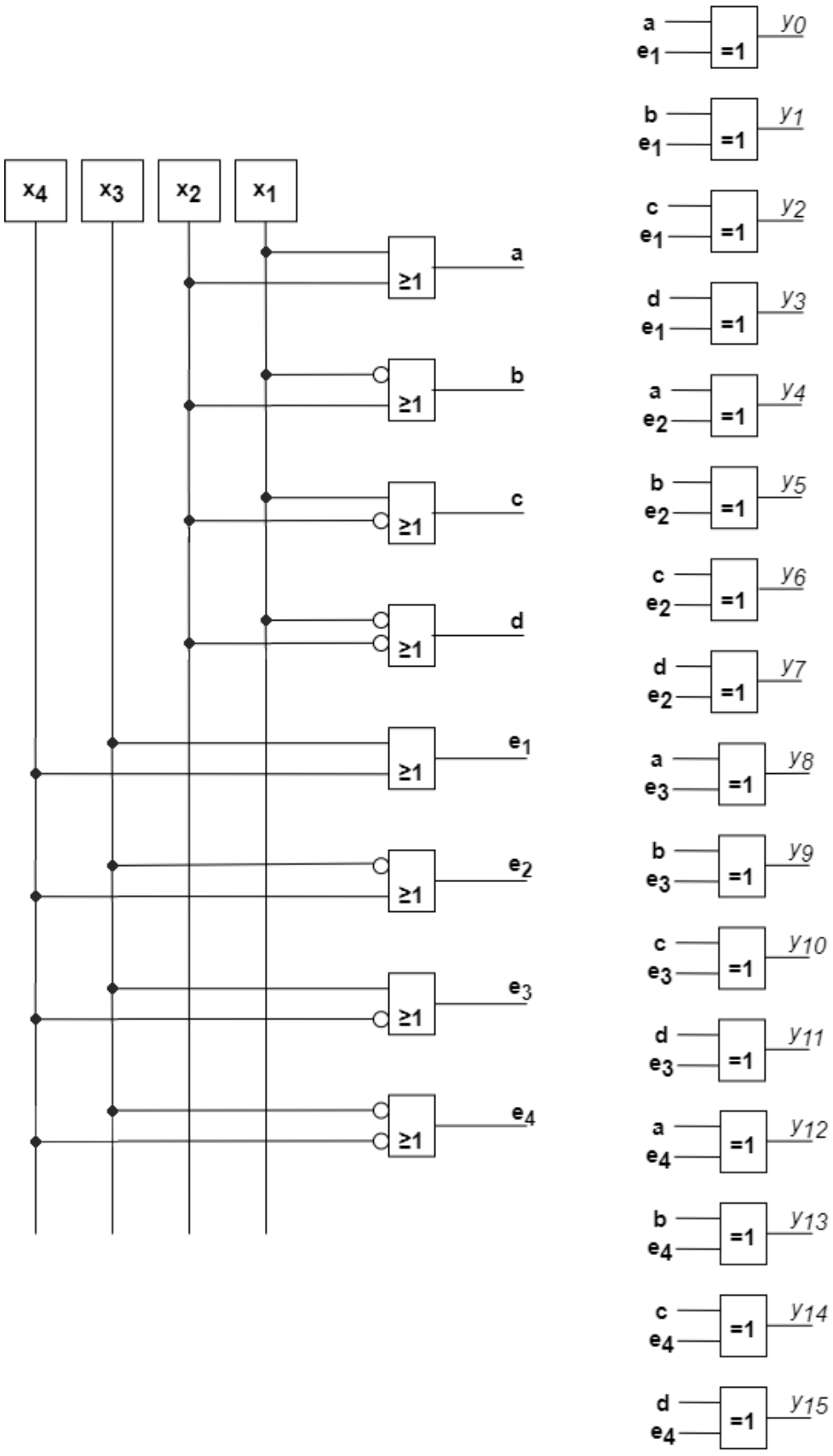
Tabulka 4.2: HW popis matice K

Sloupec	Logický výraz	Výstup
1	$a \oplus e_1$	y_0
2	$b \oplus e_1$	y_1
3	$c \oplus e_1$	y_2
4	$d \oplus e_1$	y_3
5	$a \oplus e_2$	y_4
6	$b \oplus e_2$	y_5
7	$c \oplus e_2$	y_6
8	$d \oplus e_2$	y_7
9	$a \oplus e_3$	y_8
10	$b \oplus e_3$	y_9
11	$c \oplus e_3$	y_{10}
12	$d \oplus e_3$	y_{11}
13	$a \oplus e_4$	y_{12}
14	$b \oplus e_4$	y_{13}
15	$c \oplus e_4$	y_{14}
16	$d \oplus e_4$	y_{15}

Tímto popisem je splněna podmínka zadání, aby matice byla popsána pomocí logických funkcí nonekvivalence (XOR). Z této tabulky následně vytvořím schéma zapojení těchto funkcí pomocí hradel a vytvořím zjednodušený blok, který bude tuto matici reprezentovat ve schématech, kterými budu dále popisovat matici K_{240}.

4.1.2 Schéma hardwarového zapojení matice K_{16}

Nyní je možné zakreslit schéma zapojení s využitím logických funkcí namísto hradel a sestavit tak jednoznačný hardwarový popis pro tuto matici. Schéma je zachyceno na následujícím obrázku (Obrázek 4.2):



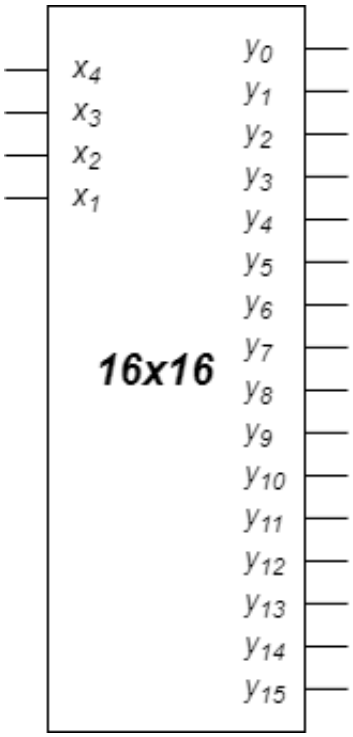
Obrázek 4.2: Schéma matice K_{16}

Pro realizaci zapojení jsem využil osm hradel typu OR (funkce logického součtu), která po správném zapojení vstupů reprezentují možné kombinace, které jsou popsány výše. Výstup každého z těchto hradel je označen symbolem, který jsem určil jako substituci v tabulce výše. Následně využiji 16 hradel typu XOR (funkce nonekvivalence), kterými zrealizuji zapojení dle předepsaných funkcí pro každý sloupec matice. Celkem toto schéma využívá 24 dvouvstupých hradel, jejichž počet zůstává při využití v zapojení nezměněn.

Vzhledem k následně prováděným syntézám a potřebám srovnání je nutné specifikovat také popis původní matice K_{16} i při zachování řádku obsahujícího hodnoty 1. Matice která by tak v původním tvaru obsahovala 17 řádků a 16 sloupců vyžaduje pro logický popis pět informačních bitů. Vzhledem k tomu že je třeba dosáhnou pouze hodnot 1 lze toto schéma rozšířit pouze o jeden další informační bit a poté každý z původních výstupů přivést do hradla typu OR společně s vodičem pátého bitu. Těchto dalších 16 hradel zajistí hodnoty 1 pro první řádek původní matice. Celkem by tak pro realizaci matice 17x16 bylo potřeba 40 dvouvstupých hradel.

Pro přehlednost nákresu jsem zvolil způsob zápisu pomocí symbolů výstupů přidaných na příslušné pozice vstupů hradel XOR. Každý výstup jednoho hradla je označen symbolem y_i obdobně jako je tomu v uvedené tabulce. Toto označení má své využití v následujícím postupu, kdy jsou tyto symboly využívány pro zpřehlednění zapojení matice K_{240}.

Pro svou jednoznačnost a stálost jsem se rozhodl toto schéma, které zůstane v dalším postupu neměnné nahradit jednoduchým blokem, který jej bude interpretovat a usnadní přehlednost následujících schémat. Zvolil jsem dva následující tvary (Obrázek 4.3), které využiji v závislosti na konkrétní potřebě.



Obrázek 4.3: Zjednodušený zápis matice K_{16}

4.1.3 Popis zřetězení matic K_{240} a jejích rotací pomocí logických výrazů

Popis zřetězené matice vyžaduje přidání dalších dvou informačních bitů do schématu. Tato matice obsahuje 49 řádků, proto je nezbytné využité celkem šesti vstupních bitů, označil jsem je x_6 a x_5. Řádkové zřetězení obsahuje třikrát matici K_{16} a její rotace a jeden řádek s hodnotami 1. Pro tyto 4 možnosti existují 4 kombinace přidaných dvou vstupů pro odlišení jednotlivých případů.

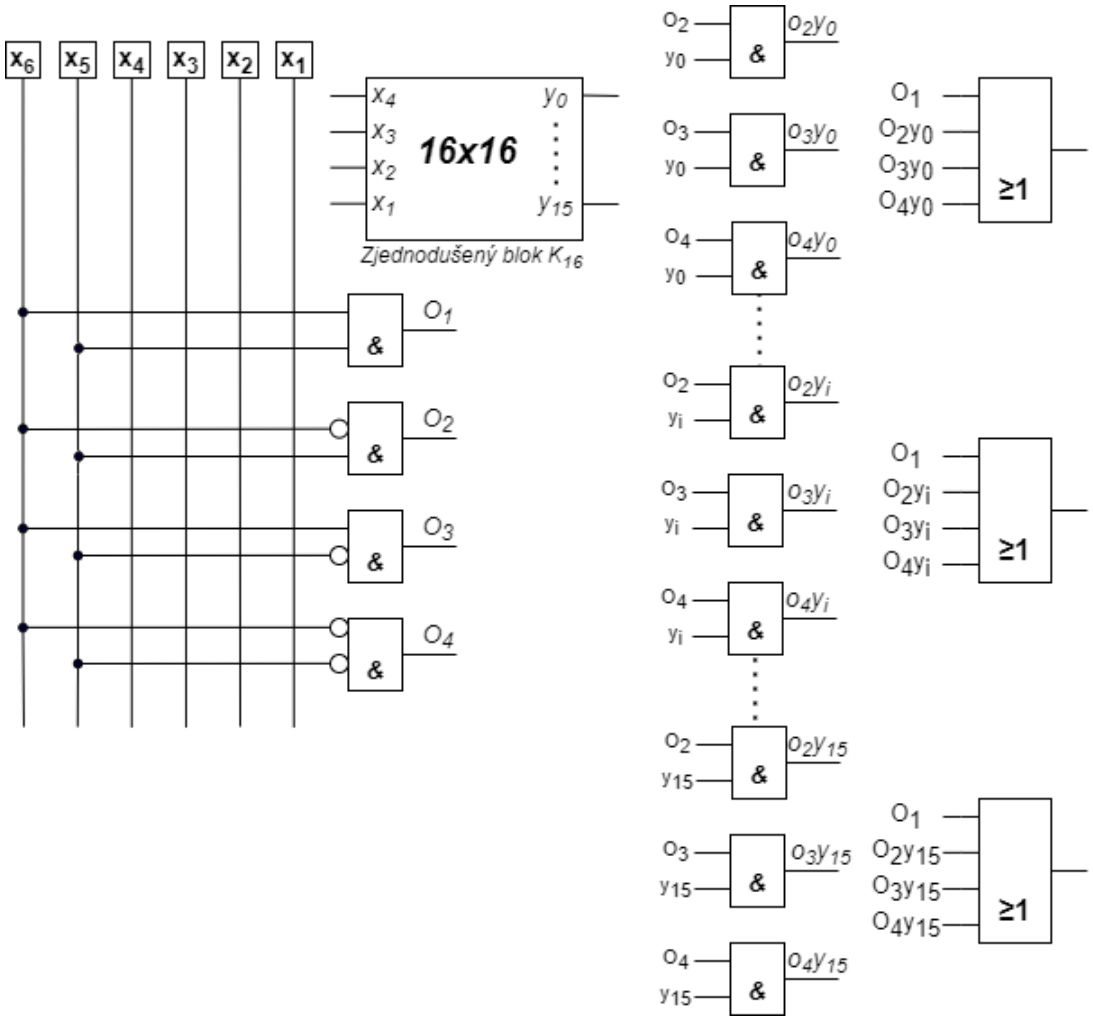
Pro popis prvního sloupce zřetězení této matice, který má tento tvar,

$$\begin{bmatrix} \mathrm{Ones}_{16} \\ \mathrm{K}_0 \\ \mathrm{K}_0 \\ \mathrm{K}_0 \end{bmatrix}$$

jsem postupoval následovně. Nejprve jsem rozšířil pravdivostní tabulku o další dva vstupní bity, v tuto chvíli bylo zřetelné, že popis každého řádku bude vycházet z výrazu k blokovému schématu matice K_{16}, jeho součtů a přidaných součinů doplněných vstupních bitů. Pro první řádek, který obsahuje hodnoty 1 je úprava zjednodušená, kdy je možné část součinu pro popis odlišných hodnot vynechat. Možné kombinace vstupů tedy jsou x_6x_5, $x_6{}'x_5$, $x_6x_5{}'$ a $x'_6x'_5$. Tyto kombinace jsem v součinovém tvaru připsal ke čtyřem shodným výrazům pro popis prvního sloupce matice K_{16} a následně sestavil logický součet těchto výrazů, který celkový první sloupec matice

K_{240} představuje. Výraz vypadá následovně $x_6x_5 + x_6{}'x_5((x_4 + x_3) \oplus (x_2 + x_1)) + x_6x_5{}'((x_4 + x_3) \oplus (x_2 + x_1)) + x_6{}'x_5'((x_4 + x_3) \oplus (x_2 + x_1))$.

Pro zakreslení do schématu jsem vytvořil obdobné substituce, podobně těm v předchozí kapitole, abych docílil přehlednějšího zápisu při popisu vícero sloupců. Jednotlivé substituce pro kombinace x_6 a x_5 jsou značeny $o_4(x_6x_5)$, o_3, o_2 a $o_1(x_6'x_5')$. Zjednodušený zápis pro první sloupec matice vypadá tedy následovně: $o_1 + o_2y_0 + o_3y_0 + o_4y_0$. Popsané schéma je zachycené taktéž graficky na následujícím obrázku (Obrázek 4.4), které znázorňuje kombinace zapojení dílčích výstupů pro realizaci prvních šestnácti sloupců (prvního sloupce zřetězení celé matice), které zatím neobsahují rotace.



Obrázek 4.4: Popis prvních 16ti sloupců matice K_{240}

Toto schéma zapojení využívá zjednodušené bloky matice K_{16}, které se třikrát opakují. Tento postup vychází z metody sdružení vytýkáním nejčastějších členů. Z tohoto důvodu bylo možné využít jeden tento blok, který poskytne realizaci jedné matice a následně správným rozdělením vodičů v rámci schématu je možné tento blok využít pro realizaci všech takových matic. Samotné schéma pro první sloupec

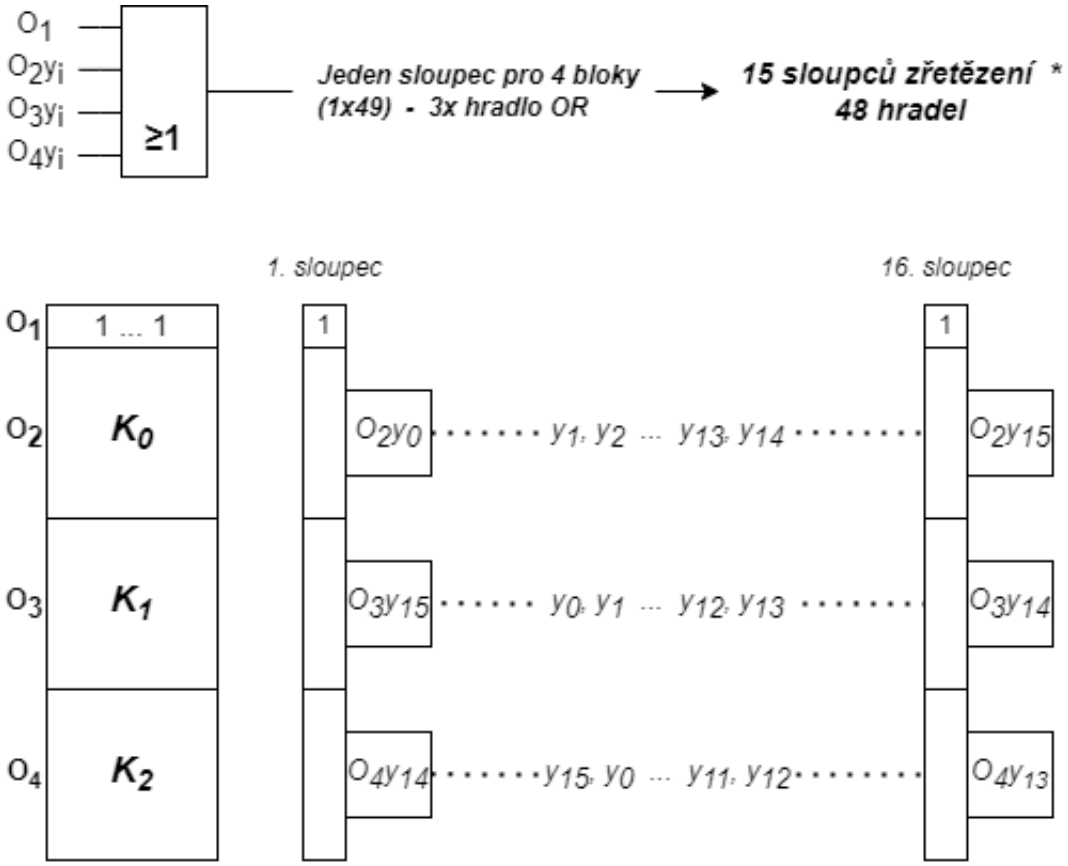
zřetězení využívá přítomnosti 24 dvouvstupcých hradel pro blok 16x16, dále 4 hradla pro kombinace bitů x_6 a x_5, následně 48 hradel pro realizaci tří možných rotací sloupců a 16 čtyřvstupých hradel (tedy 48 dvouvstupých) pro reprezentaci každého sloupce. Tato poslední hodnota je proměnlivá na základě počtu sloupců zřetězených matic.

Statický počet hradel v tomto schématu je dán součtem 24 + 4 + 48, tedy 76.

Následně jsem zpracoval úvahu a postup pro popis celé matice K_{240}, společně se stanovením počtu logických hradel, která by pro tuto realizaci byla potřebná. Vzhledem ke skutečnosti, kdy každý sloupec je popsaný logickým výrazem, který je možné dynamicky upravovat a z hlediska hardwarové realizace se jedná pouze o rozdílné zapojení vodičů mezi hradly, je úprava pro popis zbytku matice jednoznačná.

Pro první sloupec zřetězení, kdy jsou všechny tři řádky matice K_{16} se stupněm rotace 0 jsou všechny tři výrazy vždy stejné. Všechny začínají kombinacemi pro y_0 a pokračují vzestupným tempem až po y_{15}. V případě potřeby rotace bude opět 16 výrazů pro jeden sloupec zřetězení, ve kterých budou zpravidla rostoucí hodnoty y, ovšem v každém určitém výrazu budou vždy v jiném pořadí (v závislosti na stupni rotace dané matice).

Způsob zápisu výrazů a posloupnost kombinací výstupů y je popsána schématem na obrázku (Obrázek 4.5).



1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1	1 ... 1
K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0	K_0
K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8	K_9	K_{10}	K_{11}	K_{12}	K_{13}	K_{14}
K_0	K_2	K_4	K_6	K_8	K_{10}	K_{12}	K_{14}	K_1	K_3	K_5	K_7	K_9	K_{11}	K_{13}

Obrázek 4.5: Znázornění zapojení při rotacích matice K_{240}

Posledním krokem je určení počtu hradel potřebných k finální hardwarové realizaci. Vzhledem k předchozímu schématu a způsobu realizace je nyní potřeba již minimální počet přidaných hradel. Konkrétně za každý sloupec s hodnotami, až do počtu 240 je potřeba jedno čtyřvstupé hradlo pro správnou interpretaci rotací. Za každý sloupec zřetězených matice je to tedy 48 dvouvstupých hradel.

Celkový počet hradel pro hardwarovou realizaci matice K_{240} jsem vyhodnotil následovně. 76 dvouvstupých hradel je neměnný počet, nezávislý na počtu zřetězených sloupců, v případě této konkrétní matice je počet zřetězení roven 15, tedy pro správnou interpretaci rotací je potřeba 15*48, tedy 720 dvouvstupých hradel. Celkem je dle mého návrhu nutno využít 796 hradel pro hardwarovou realizaci této matice při manuálním návrhu schématu.

4.1.4 Srovnání manuálního návrhu se strojní syntézou do dvouvstupých hradel

Pro automatickou syntézu navrženého kódu byla využita metoda zaměřená převážně na využití dvouvstupých hradel, s minimálním zapojením metody LUT. Syntéza byla provedena s důrazem na minimalizaci počtu použitých hradel, čehož bylo v konečném výsledku dosaženo. To zajišťuje efektivní využití dostupného hardwaru a zároveň umožňuje snadnější optimalizaci a ladění výsledného obvodu. Tento přístup přinesl výhodu přehledné a efektivní hardwarové realizace navržené matice, která splňuje specifikace a zároveň minimalizuje nároky na zdroje.

Automatickou syntézu provedl doc. Ing. Petr Fišer, Ph.D. pro matice 17X16 a 49x240. V případě první matice využil metodu Espresso-Dso-BDS-2-gates, která zajistila nejlepší výsledek s konečným minimálním počtem 65 dvouvstupých hradel. Pro větší matici to byla metoda 2-gates, která provedla realizaci za použití 1042 dvouvstupcých hradel. Obě metody a jejich specifikace jsou k nalezení na webu Petra Fišera [16].

Při srovnání s výsledky výše provedené manuální syntézy lze říci, že dosažené výsledky jsou velice podobné. Ve srovnání menší matice využívá manuální syntéza 40 hradel oproti automatické, která návrh zrealizovala pomocí 65 dvouvstupých hradel. V případě větší matice je rozdíl mezi hodnotami 796 a 1042 také minimální. Ze srovnání lze říci, že manuální syntéza pro tyto konkrétní matice vychází o něco efektivněji, ovšem je to z důvodu relativně malého rozsahu zkoumaných kódů a stoprocentní znalosti jejich tvaru. V případě popisu rozsáhlejších kódů nebo méně specifikovaných lze očekávat že návrhy těmito metodami automatické syntézy budou výrazně efektivnější oproti manuálním.

Závěr

V úvodu práce byla zpracována rešerše aktuálních zdrojů o tématu nelineárních blokových kódů a jejich využití při testování logických obvodů. Současné zdroje jsem postupně zpracoval a chronologicky sestavil podkapitoly pro jednotlivá témata, aby bylo zřejmé jakým způsobem se problematika tohoto tématu vyvíjela. Klíčové jsou zejména nejaktuálnější poznatky v oblasti NBC pokrývající tři specifikované bity, na které tato práce přímo navazuje. Rešerše poskytla solidní přehled informací pro seznámení se s danou problematikou a utváří přehled pozadí tématu a představuje klíčové souvislosti.

Na tomto základě jsem s pomocí myšlenkové mapy zpracoval několik hypotéz pro splnění vytyčených cílů v oblasti NBC se třemi specifikovanými bity. Toto téma jsem rozdělil na tři části, ověření NBC vytvořených strategií negace, dále strategie vytváření NBC pomocí rotací matic blokového kódu a následně možná alternativní negace pro nalezení jiného přístupu. K ověření požadovaných parametrů jsem využil experimentální úvahy nad kombinacemi blokových kódů, které jsem následně realizoval pomocí testovacích skriptů v prostředí MATLAB. Takto jsem ověřil jednak parametry pro NBC vytvořené strategií negace, tak pro NBC vytvoření strategií rotací matice. Zde bylo několik velice přínosných závěrů, které ukazují, jakým způsobem je možné zřetězení matic realizovat pro dosažení vyšší efektivity, a to jak v časové složitosti nalezení kódu, tak i v oblasti samotného testování logických obvodů.

Přínosem v oblasti rotací matice a jejich zřetězení je zejména nalezení pravidla pro vytváření takových matice. Pomocí zmíněných experimentů jsem dosáhl nalezení shodného chování stupňů rotací v několika případech zřetězení a bylo tak možné určit jakým způsobem rotace realizovat. Důležité bylo taktéž popsání pravidla pro označení matice a sestavení správného výrazu pro popis indexu kódu. Alternativním přínosem bylo nalezení rozšířené strategie negace, která taktéž poskytuje vyšší efektivitu NBC oproti LBC, ovšem nedosahuje výsledků, které zajišťuje strategie rotací.

Téma NBC se čtyřmi specifikovanými bity je ve velice rané fázi svého popisu a zde bylo zapotřebí taktéž ověřit parametry, které jsou pro jejich vytváření nastavené. Přínosem bylo také ověření správnosti sestavené matice, která poté opět využívá rotace. Experimentálním způsobem jsem navrhl druhý nezávislý přístup s obdobným výsledkem.

Dle specifikací zadání bylo potřeba provést návrh a zvážení hardwarové realizace navržených řešení. Zejména pro matice K_{16} a K_{240}. Tuto realizaci jsem provedl pomocí popisu logických výrazů pro jednotlivé sloupce (předepsané funkce) každé z těchto matic. Realizaci jsem znázornil schématem zapojení jednotlivých částí, hra-

del a vodičů, s důrazem na minimalizaci počtu logických hradel a co nejvyšším počtem využití hradel typu XOR. Návrh schématu byl (a měl být) realizován pouze v teoretické rovině, k samotné realizaci by bylo potřeba učinit více testování a zvolit další způsoby ověření správnosti návrhu tohoto schématu.

Ověření návrhu jsem provedl srovnáním se strojní syntézou do dvouvstupých hradel, kterou provedl dle specifikací pan Petr Fišer. Na základě srovnání lze zhodnotit, že manuální návrh je obdobný z hlediska počtu využitích hradel, ovšem některé jeho části by bylo možné optimalizovat. Manuální syntéza je s minimálním rozdílem efektivnější z hlediska počtu dvouvstupých hradel, ovšem při realizaci složitějšího obvodu by pravděpodobně měl manuální návrh o poznání horší výsledky a automatická syntéza by dosahovala výrazně efektivnějších výsledků. V této situaci rozdíl využitých hradel není vysoký. Pro jednoduchou demonstraci a ukázku možného návrhu hardwarové realizace jsou oba přístupy ideální ukázkou.

Z toho lze vyvodit závěr, že téma NBC pro tři a čtyři specifikované bity nabízí alternativu v přístupu testování logických obvodů. Téma je velice široké a z velké části nabízí prostor pro řadu experimentů, které by jej mohly významně obohatit. Tato práce popisuje některé ze základních aspektů nelineárních blokových kódů, plynule navazuje na dosavadní dosažené výsledky a nabízí další možnosti pro navázání na toto téma a následné rozšíření. Zejména v oblasti hardwarové realizace a návrhu strategií pro vytváření NBC s více specifikovanými bity.

Seznam použité literatury

1. NOVÁK, Ondřej. Deterministic Search Strategy of Compression Codes. In: *2023 26th Euromicro Conference on Digital System Design (DSD)*. IEEE, 2023, s. 198–205.
2. NOVÁK, Ondřej. Nonlinear compression block codes: Exact and random search strategy. *Microprocessors and Microsystems*. 2023, roč. 101, s. 104877.
3. HAMZAOGLU, Ilker; PATEL, Janak H. Reducing test application time for full scan embedded cores. In: *Digest of Papers. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing (Cat. No. 99CB36352)*. IEEE, 1999, s. 260–267.
4. BAHL, Swapnil et al. Unifying scan compression. In: *2014 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*. IEEE, 2014, s. 191–196.
5. MITRA, Subhasish; KIM, Kee Sup. XPAND: An efficient test stimulus compression technique. *IEEE Transactions on Computers*. 2006, roč. 55, č. 2, s. 163–173.
6. *MinT* [online]. 2015. [cit. 2024-05-05]. Dostupné z: http://mint.sbg.ac.at/index.php.
7. SRINIVASAN, Rajagopalan; GUPTA, Sandeep K; BREUER, Melvin A. Novel test pattern generators for pseudoexhaustive testing. *IEEE Transactions on Computers*. 2000, roč. 49, č. 11, s. 1228–1240.
8. SHAH, Mihir A; PATEL, Janak H. Enhancement of the Illinois scan architecture for use with multiple scan inputs. In: *IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2004, s. 167–172.
9. DING, Cunsheng. *Designs from linear codes*. World Scientific, 2022.
10. GOLAN, P. Pseudoexhaustive test pattern generation for structured digital circuits. *Proc. FTSD9, Brno, Czechoslovakia*. 1986, s. 214–220.
11. NOVÁK, Ondřej; ROZKOVEC, Martin; PLÍVA, Jan. Decompressors using nonlinear codes. *Microprocessors and Microsystems*. 2020, roč. 76, s. 103076.
12. NOVÁK, Ondřej. Search strategy of large nonlinear block codes. In: *2021 24th Euromicro Conference on Digital System Design (DSD)*. IEEE, 2021, s. 527–534.

13. NOVÁK, Ondřej. Extended binary nonlinear codes and their application in testing and compression. In: *2017 22nd IEEE European Test Symposium (ETS)*. IEEE, 2017, s. 1–2.

14. SCHMIDT, Jan; FIŠER, Petr. Nonlinear codes for test patterns compression: the old school way. In: *Recent Findings in Boolean Techniques: Selected Papers from the 14th International Workshop on Boolean Problems*. Springer, 2021, s. 125–142.

15. NOVÁK, Ondřej. Nonlinear compression block codes search strategy. In: *2022 25th Euromicro Conference on Digital System Design (DSD)*. IEEE, 2022, s. 665–670.

16. *Doc. Ing. Petr Fišer, Ph.D.* [online]. 2024. [cit. 2024-05-07]. Dostupné z: https://users.fit.cvut.cz/~fiserp/index.php?page=research.