Jakub HYLA[1 2] Wojciech SUŁEK[1]

Silesian University of Technology (1), TKH Technology Poland Sp. z.o.o. (2)

doi: 10.15199/48.2024.03.44

Niebinarne kodowanie LDPC dla systemów Internetu rzeczy

Streszczenie. *Artykuł dotyczy implementacji niebinarnego kodera LDPC dla urządzeń IoT. W artykule zaproponowano wykorzystanie efektywnego algorytmu kodowania LDPC (Low Density Parity Check) w urządzeniach o mocno ograniczonych zasobach – pamięci oraz mocy obliczeniowej. Wskazano algorytm kodujący uogólniony do kodów niebinarnych nad ciałem $GF(2^q)$ oraz przedstawiono wyniki eksperymentalne implementacji w układzie typowego mikrokontrolera. W publikacji pokazano porównanie zależności czasowych dla kodów nad różnymi rzędami ciał GF. Pokazano wpływ wyboru kodu na potencjalne zużycie energii.*

Abstract. *The article concerns the implementation of a non-binary LDPC encoder for IoT devices. The article proposes the use of an effective LDPC (Low Density Parity Check) coding algorithm in devices with very limited resources – memory and computing power. A coding algorithm generalized to non-binary codes over the $GF(2^q)$ is indicated and experimental results of implementation in a typical microcontroller system are presented. The publication shows a comparison of time dependencies for codes over different orders of GF fields. The impact of code selection on potential energy consumption is also shown.* ***(Non-Binary LDPC coding for Internet of Things)***

Słowa kluczowe: IoT, LDPC, QC-LDPC, kodowanie
Keywords: IoT, LDPC, QC-LDPC, coding

Wprowadzenie

Współczesny postęp technologiczny ma ścisły związek z wysokiej jakości transmisją danych w systemach teleinformatycznych, w tym systemach kategoryzowanych jako Internet rzeczy. Oczekuje się przesyłania coraz większej liczby informacji w coraz krótszym czasie, przy minimalnym zużyciu energii [18]. To z kolei rodzi problemy związane z błędami w przesyle danych, które są nieuniknione, ale mogą być wykrywane i korygowane odpowiednimi technikami [11].

W celu eliminacji wspomnianych błędów, stosuje się techniki kodowania kanałowego. Te metody umożliwiają skuteczne wykrywanie oraz korekcję błędów występujących w danych po stronie odbiorcy informacji. Kodowanie kanałowe wymaga dodawania nadmiarowych bitów kontrolnych do danych, co zwykle odbywa się w blokach o jednakowej długości [17].

Kody LDPC (Low Density Parity Check) stanowią obecnie jedną z najskuteczniejszych metod kodowania blokowego i charakteryzują się znakomitymi zdolnościami korekcyjnymi. Ich historia sięga roku 1962 [13], ale z różnych powodów nie były szeroko wykorzystywane w tamtych czasach [6]. Dopiero w 1999 roku zyskały popularność i obecnie znajdują zastosowanie w różnych systemach, takich jak DVB-S2, DVB-T2 [4], WiFi [1], WiMAX[10] i sieci komórkowe 5G[15, 2, 14, 19].

Projekt badawczy, z którym związany jest niniejszy artykuł, ma na celu pokazanie, że kody LDPC mogą być skutecznie stosowane także w systemach o ograniczonych zasobach sprzętowych, takich jak urządzenia z obszaru Internetu Rzeczy (IoT). W ramach tego projektu stworzono wydajną implementację kodera LDPC dla mikrokontrolerów, a także dekodera, co przedstawiono w publikacjach [9],[7]. Omówiono wyzwania związane ze złożonością obliczeniową kodera LDPC, proponując zmiany w sposobie zapisu macierzy kontrolnych kodów, co znalazło zastosowanie w urządzeniach typu IoT opartych na mikrokontrolerach o ograniczonych możliwościach obliczeniowych [3, 16]. Efektywna implementacja algorytmu kodowania może przynieść znaczne korzyści, szczególnie w przypadku urządzeń IoT zasilanych bateryjne, co pozwala na oszczędność energii [8].

Kontynuacja prac dotyczy wykorzystania niebinarnych (NB) kodów LDPC (Low-Density Parity-Check) w kontekście kodowania bloków informacyjnych. Autorzy zwrócili uwagę na istotne aspekty związane z wydajnością implementacji algorytmu kodowania, sugerując, że skuteczne wdrożenie może przynieść znaczne korzyści. Szczególnie podkreślono potencjalne korzyści dla urządzeń Internetu Rzeczy (IoT) zasilanych bateryjnie, gdzie efektywne kodowanie może istotnie przyczynić się do oszczędności energii. Wnioski z badań wskazują na obiecujące perspektywy zastosowania kodów NB-LDPC w praktyce, zarówno pod względem efektywności energetycznej, jak i ogólnej wydajności systemów komunikacyjnych. W artykule omówiono wyniki związane ze zużyciem pamięci, czasem potrzebnym do zakodowania bloku informacyjnego oraz zużyciem energii.

Podstawowe definicje

Kody LDPC to klasa kodów korekcyjnych, liniowych kodów blokowych, które cechują się niskim stopniem zagęszczenia macierzy kontroli parzystości. Znaczna większość elementów macierzy kodu jest zerami. W związku z tym możliwe jest wykorzystanie efektywnych iteracyjnych algorytmów dekodowania, a jednocześnie kody te charakteryzują się dużymi możliwościami korekcyjnymi [12].

Macierz kontroli parzystości, lub też macierz kontrolna, jest to macierz, która opisuje zależności między bitami informacyjnymi a bitami parzystości (nadmiarowymi) w kodzie. Macierz ta definiuje konkretny kod i w przypadku kodów LDPC jest macierzą rzadką. Wartości różne od zera w macierzy parzystości wskazują, które bity informacyjne są powiązane z którymi bitami parzystości w danym wektorze kodowym.

Graf Tannera, znany również jako graf wiadomości (message-passing graph) lub graf dekodowania, jest strukturą używaną w kontekście dekodowania kodów korekcyjnych, w tym kodów LDPC (Low-Density Parity-Check) oraz kodów BCH (Bose-Chaudhuri-Hocquenghem). Graf Tannera jest używany do reprezentacji i analizy procesu dekodowania w algorytmach korzystających z iteracyjnych propagacji wiadomości (BP – Belief Propagation).

Stopień wierzchołka oznacza liczbę krawędzi incydentnych do danego wierzchołka w grafie, który reprezentuje kod LDPC. Rozkład stopni wierzchołków jest istotnym parametrem w analizie i projektowaniu kodów LDPC, ponieważ wpływa z jednej strony na ich własności korekcyjne, a z drugiej strony – złożoność obliczeniową.

Sprawność kodu LDPC określa stosunek liczby bitów informacyjnych do ogólnej liczby bitów w wektorze kodowym. Jest to jeden z podstawowych parametrów, który można dostosować, aby uzyskać kod LDPC o określonych możliwościach korekcyjnych.

Pojemność informacyjna kanału odnosi się do maksy-

malnej liczby informacji, jaką można przesłać przez dany kanał komunikacyjny przy założeniu pewnego poziomu zakłóceń. Kody LDPC osiągają możliwości zbliżone do tej pojemności, co oznacza, że są one w stanie przesyłać dane z dużą wydajnością przy minimalnych błędach [17].

Operacje arytmetyczne w ciele Galois

Algorytm kodowania wymaga wykonywania operacji dodawania i mnożenia w ciele $GF(2^q)$. W niniejszej sekcji, zostaną omówione te operacje, w kontekście ich implementacji w układzie mikrokontrolera.

×	0	1
0	0	0
1	0	1

Rys. 1. Mnożenie w GF(2)

Ciało Galois o najniższym możliwym rzędzie, $GF(2)$, nazywane jest również ciałem binarnym [5] i składa się z dwóch elementów: 0 i 1. Operacje dodawania i mnożenia w ciele GF(2) są zdefiniowane w sposób - dodawanie jest tożsame do operacji OR , zgodnie z odpowiadającym fragmentem macierzy dodawania z Rys. 4, mnożenie jest tożsame do operacji AND, zgodnie z Rys. 1.

Ciało GF(2) jest używane w różnych dziedzinach, takich jak informatyka, elektronika, kryptografia i teoria kodowania korekcyjnego, ze względu na swoją prostotę i możliwość bezpośredniej reprezentacji danych binarnych.

×	1	2	3
1	1	2	3
2	2	3	1
3	3	1	2

Rys. 2. Mnożenie w GF(2^2)

Ciało $GF(2^q)$ nazywane jest rozszerzonym ciałem Galois, gdzie q jest liczbą całkowitą większą od 1. To ciało składa się z 2^q elementów, a każdy element może być reprezentowany przez wielomian $P(x)$ stopnia $q-1$ ze współczynnikami 0 lub 1. Operacje dodawania i mnożenia w $GF(2^q)$ są nieco bardziej skomplikowane niż w przypadku ciała $GF(2)$. Ciała rozszerzone $GF(2^q)$ są używane w zaawansowanych aplikacjach, takich jak kodowanie korekcyjne Reeda-Solomona, np. w zapisie danych na płytach CD i DVD, a także w algorytmach kryptograficznych.

Operacje dodawania w ciele $GF(2^q)$ dla $q = 1, 2, \ldots$ odbywają się na q-bitowych wartościach binarnych przy użyciu operacji bitowego XOR. Dla $q = 4$ zaprezentowano tablicę dodawania na rys. 4, natomiast dla niższych rzędów niż $q = 4$, należy wykorzystać fragment tej tablicy (odpowiednią lewą górną jej część).

Elementy ciała $GF(2^2)$ można reprezentować przez liczby dziesiętne z zakresu $0 \ldots 3$, lub też słowa 2-bitowe. Mnożenie można także przedstawić w formie tabeli, jak na Rys. 2. Kolumnę i wiersz z wartościami zerowymi pominięto dla czytelności tabeli, ponieważ mnożenie przez zero zawsze daje wynik równy zero.

Elementy ciała $GF(2^3)$ można reprezentować przez liczby z zakresu $0 \ldots 7$, co odpowiada słowom 3-bitowym. Mnożenie w $GF(2^3)$ można zdefiniować jako mnożenie wielomianów, po którym następuje redukcja iloczynu modulo $P(x)$. Co ważne z punktu widzenia implementacji, operację

×	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7
2	2	4	6	3	1	7	5
3	3	6	5	7	4	1	2
4	4	3	7	6	2	5	1
5	5	1	4	2	7	3	6
6	6	7	1	5	3	2	4
7	7	5	2	1	6	4	3

Rys. 3. Mnożenie w GF(2^3)

mnożenia można przedstawić w postaci tablicy, jak na rys. 3.

Mnożenie w ciele GF to operacja, która może interpretowana jako dzielenie z redukcją (modulo), tzn. dzielenie przy użyciu wielomianu redukującego jako dzielnika.

+	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	0	3	2	5	4	7	6	9	8	B	A	D	C	F	E
2	2	3	0	1	6	7	4	5	A	B	8	9	E	F	C	D
3	3	2	1	0	7	6	5	4	B	A	9	8	F	E	D	C
4	4	5	6	7	0	1	2	3	C	D	E	F	8	9	A	B
5	5	4	7	6	1	0	3	2	D	C	F	E	9	8	B	A
6	6	7	4	5	2	3	0	1	E	F	C	D	A	B	8	9
7	7	6	5	4	3	2	1	0	F	E	D	C	B	A	9	8
8	8	9	A	B	C	D	E	F	0	1	2	3	4	5	6	7
9	9	8	B	A	D	C	F	E	1	0	3	2	5	4	7	6
A	A	B	8	9	E	F	C	D	2	3	0	1	6	7	4	5
B	B	A	9	8	F	E	D	C	3	2	1	0	7	6	5	4
C	C	D	E	F	8	9	A	B	4	5	6	7	0	1	2	3
D	D	C	F	E	9	8	B	A	5	4	7	6	1	0	3	2
E	E	F	C	D	A	B	8	9	6	7	4	5	2	3	0	1
F	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0

Rys. 4. Dodawanie w GF(2^4)

Wartości w ciele $GF(2^4)$ to czterobitowe wartości, które obejmują zakres dziesiętny [0..15]. Dodawanie odbywa się na tych czterobitowych wartościach za pomocą operacji logicznego XOR. Na przykład: 5 + 6 = (0101) + (0110) = (0011) = 3 (zaznaczone na rys. 4).

Każde zwiększenie stopnia wielomianu o 1 prowadzi do dwukrotnego zwiększenia każdego z wymiarów tablicy dodawania i tablicy mnożenia, co skutkuje czterokrotnie większym rozmiarem w pamięci. Operacja dodawania może być wykonana za pomocą funkcji XOR bitów w reprezentacji wielomianowej, zatem jest obliczana na bieżąco w czasie trwania algorytmu, bez korzystania z tablicy operacji. Obliczenie wyniku mnożenia w ciele Galois wyższych rzędów wymagałoby bardziej złożonych obliczeń. W związku z tym, jeśli pojemność pamięci programu na to pozwala, lepszym rozwiązaniem może być wykorzystanie tablicy wyników mnożenia, co zostało wykorzystane w niniejszych pracach.

×	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	2	4	6	8	A	C	E	3	1	7	5	B	9	F	D
3	3	6	5	C	F	A	9	B	8	D	E	7	4	1	2
4	4	8	C	3	7	B	F	6	2	E	A	5	1	D	9
5	5	A	F	7	2	D	8	E	B	4	1	9	C	3	6
6	6	C	A	B	D	7	1	5	3	9	F	E	8	2	4
7	7	E	9	F	8	1	6	D	A	3	4	2	5	C	B
8	8	3	B	6	E	5	D	C	4	F	7	A	2	9	1
9	9	1	8	2	B	3	A	4	D	5	C	6	F	7	E
A	A	7	D	E	4	9	3	F	5	8	2	1	B	6	C
B	B	5	E	A	1	F	4	7	C	2	9	D	6	8	3
C	C	B	7	5	9	E	2	A	6	1	D	F	3	4	8
D	D	9	4	1	C	8	5	2	F	B	6	3	E	A	7
E	E	F	1	D	3	2	C	9	7	6	8	4	A	B	5
F	F	D	2	9	6	4	B	1	E	C	3	8	7	5	A

Rys. 5. Mnożenie w GF(2^4)

Autorzy zdecydowali się skorzystać z gotowych tablic mnożących dla branych pod uwagę przypadków, a eksperymenty przeprowadzono dla ciał od $GF(2)$ do $GF(2^6)$.

Model systemu IoT

W ogólnej architekturze rozpatrywanego w artykule systemu IoT (Rys. 6) można wyróżnić różne kategorie (warstwy) urządzeń. W warstwie urządzeń końcowych IoT, istotną klasą są urządzenia zbierające dane pomiarowe w postaci surowej lub przetworzonej, z wykorzystaniem na przykład filtrów cyfrowych. Mogą to być czujniki parametrów środowiskowych, pogodowych, poziomu wody, wilgotności gleby, itp. W niniejszym artykule rozpatrywana jest komunikacja pomiędzy tymi urządzeniami a centralnymi węzłami – bramkami. Bramki agregują dane z wielu podłączonych do nich urządzeń – ich zadaniem jest zbieranie danych od urządzeń IoT i przekazywanie ich do chmury. Komunikacja ta może odbywać się z wykorzystaniem różnych rozwiązań – protokołów sieciowych, jak również różnych mediów – bezprzewodowych, przewodowych, światłowodowych. W założeniu urządzenia końcowe mogą być zasilane bateryjnie, natomiast bramka jest zasilana z sieci energetycznej. W związku z tym, szczególnego znaczenia nabiera efektywność energetyczna łącza w kierunku w górę (od urządzenia końcowego do bramki). Zagadnienia implementacji kodowania korekcyjnego mają tu istotne znaczenie z dwóch powodów: 1) im lepsze własności korekcyjne zastosowanego kodu, tym niższa może być moc nadawanego sygnału, 2) najlepsze własności korekcyjne są osiągane dla kodów o stosunkowo dużej złożoności obliczeniowej implementacji, co wymaga obciążenia układu obliczeniowego urządzenia, także istotnym zużyciem energii. Warto zastosować jak najlepsze kody korekcyjne, które zapewniają znakomite własności korekcyjne, w szczególności dla krótkich bloków, gdyż dane transmitowane w sieciach IoT mają często charakter krótkich bloków. Jednocześnie czas kodowania powinien być możliwie jak najkrótszy dla zapewnienia jak najkrótszego czasu w trybie aktywnym układu procesora.

Biorąc to wszystko pod uwagę, wydaje się, że niebinarne kody LDPC, pomimo ich dużej złożoności algorytmu dekodowania, mogą mieć potencjał zastosowania w łączu w górę w tego typu sieci, ponieważ znane są ich znakomite własności w szczególności dla krótkich i średnich rozmiarów bloków, a złożoność operacji kodowania może zostać utrzymana na poziomie konkurencyjnym do podobnych kodów binarnych. W niniejszym artykule przedstawiamy wyniki implementacji kodera kodów niebinarnych w układzie mikrokontrolera, typowej jednostce centralnej urządzenia końcowego. Przedstawiamy algorytm, implementację oraz eksperymentalnie określone czasy kodowania dla różnych rzędów ciała Galois $GF(q)$. Przeprowadzamy także analizę rozmiaru pamięci RAM wymaganej do pomieszczenia podstawowych zmiennych algorytmu.

Algorytm kodowania niebinarnego LDPC

Wykorzystano efektywny algorytm Richardsona–Urbanke [17], uogólniając go dla kodów nad niebinarnymi ciałami $GF(2^q)$. Wykorzystywana jest macierz kontrolna w postaci prawie-dolnotrójkątnej:

$$\mathbf{H} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{T} \\ \mathbf{C} & \mathbf{D} & \mathbf{E} \end{bmatrix}, \tag{1}$$

gdzie $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, $\mathbf{E}$, $\mathbf{T}$ to podmacierze $\mathbf{H}$ nad ciałem$GF(2^q)$, przy czym $\mathbf{T}$ jest macierzą dolnotrójkątną. Można wykazać, że zakodowanie słowa informacyjnego $\mathbf{u}$ w słowo kodowe $\mathbf{c} = [\mathbf{u}, \mathbf{p}_1, \mathbf{p}_2]$ można wyrazić z wykorzystaniem dwóch członów bloku parzystości:

$$\mathbf{p}_1^T = \mathbf{\Phi}^{-1}(\mathbf{E}\mathbf{T}^{-1}\mathbf{A}\mathbf{u}^T + \mathbf{C}\mathbf{u}^T), \tag{2}$$

$$\mathbf{p}_2^T = \mathbf{T}^{-1}(\mathbf{A}\mathbf{u}^T + \mathbf{B}\mathbf{p}_1^T) \tag{3}$$

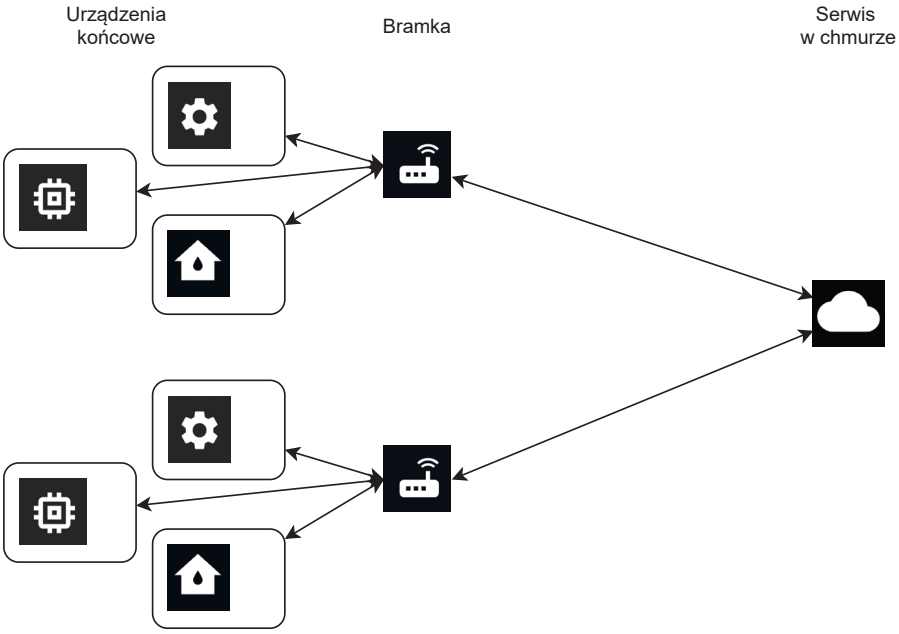
Operacje elementarne, na które można rozłożyć (2)–(3), to mnożenie przez macierz rzadką, dodawanie macierzy rzadkich oraz tzw. operacja wstawiania wstecz [9], do której można sprowadzić mnożenie przez $\mathbf{T}^{-1}$. Jedynie $\mathbf{\Phi} = \mathbf{E}\mathbf{T}^{-1}\mathbf{B} + \mathbf{D}$ jest w ogólności macierzą gęstą. Efektywnie dekodowane kody, które opracowano na cele przedstawionych prac, cechują się tym, że $\mathbf{\Phi}$ jest macierzą jednostkową, $\mathbf{\Phi} = \mathbf{I}$.

Implementacja kodowania NB-LDPC w urządzeniach IoT

Rozwiązanie oparte o binarne kody LDPC, wraz z algorytmem kodowania i jego implementacją, zostało opisane w [9]. Przedstawiono efektywne kodowanie z wykorzystaniem kodów LDPC oraz QC-LDPC, w ciele $GF(2)$. Prace przedstawione w niniejszym artykule są kontynuacją związaną z implementacją kodera dla kodów niebinarnych, definiowanych nad ciałami $GF(2^2)$–$GF(2^6)$.

W implementacji algorytmu wykorzystano reprezentację wektorową współczynników wielomianów reprezentujących elementy ciała. Mnożenie elementów ciała jest wykonywane poprzez indeksację tablicy wartości definiujących mnożenie (rys. 2,3,5 itd.), dla rzędów od 2^2 do 2^6. Macierz kontroli parzystości jest zapisana w pamięci w postaci szeregu indeksów elementów niezerowych oraz szeregu wartości tych elementów, w ciele GF. Należy zauważyć, że zwiększanie rzędu ciała, przy zachowaniu wielkości bloku kodowego (wyrażonego w bitach) oraz sprawności kodu, daje proporcjonalnie mniejsze macierze kontroli parzystości.

Prace eksperymentalne przeprowadzono uwzględniając dwa kryteria optymalizacji, tj. zajętość pamięci RAM przez struktury danych algorytmu oraz czas kodowania. Starano



Rys. 6. Uproszczony model komunikacji z wykorzystaniem kodowania korekcyjnego LDPC w systemie IoT o topologii gwiazdy

się wskazać rząd ciała, który daje najlepszy kompromis pomiędzy złożonością (definiowaną przez dwa wymienione kryteria) oraz możliwościami korekcyjnymi (które rosną wraz z rzędem ciała). Badania skoncentrowano na stosunkowo krótkich blokach danych, nie większych niż 1000 bitów, co jest uzasadnione dla typowo krótkich wiadomości transmitowanych w sieciach IoT.

W artykule podjęto analizę wpływu rzędu ciała Galois na liczbę przeprowadzanych obliczeń oraz zużycie energii, wyrażone skalą zużycia baterii, przy poczynionych założeniach co do sprzętu oraz trybów pracy (aktywny i uśpienia). Przeprowadzono również ocenę wpływu kodów niebinarnych na czas kodowania w warunkach ograniczonych zasobów obliczeniowych. Przedstawione zostaną zalety i wady tychże kodów w kontekście zastosowania w systemach Internetu rzeczy.

Szacowanie zużycia energii oraz dobór baterii

Dobór odpowiedniej baterii dla urządzeń IoT stanowi istotne wyzwanie, mając kluczowy wpływ na zapewnienie deklarowanego czasu pracy zgodnie z określonymi parametrami. Długość działania urządzenia zasilanego bateryjnie jest determinowana przez szereg czynników, takich jak:

- napięcie nominalne,
- pojemność baterii,
- maksymalny, chwilowy i średni prąd pobierany z baterii,
- temperatura baterii w trakcie eksploatacji.

To zagadnienie jest złożone, a jego analiza utrudniona przez wielość parametrów oraz wzajemne zależności między nimi. Na przykład urządzenie monitorujące, które jest wykorzystywane w różnych warunkach pogodowych, np. pełni funkcję monitorowania zapełnienia pojemników na śmieci, w Polsce w roku 2022 funkcjonowałoby w zakresie temperatur od minimalnej na poziomie -18,6 stopni Celsjusza do maksymalnej na poziomie 38,3 stopni Celsjusza.

Temperatura ma bezpośredni wpływ na pojemność baterii i napięcie, istnieje także zależność pomiędzy pobieranym prądem a napięciem oraz pojemnością. Dodatkowo, utrudnieniem w dokładnej ocenie jest pozostała pojemność baterii, wynikająca z charakterystyki napięciowej w trakcie procesu rozładowywania. W praktyce, szacuje się czas pracy na baterii dla stałych warunków oraz przeprowadza testy dla konkretnych punktów pracy, mające na celu symulację określonego scenariusza w skróconym okresie czasu.

Urządzenia będące węzłami IoT, jeśli są zasilane bateryjnie, w celu zwiększenia żywotności baterii powinny działać w schemacie opartym o dwa tryby pracy, zmieniane periodycznie:

- Tryb uśpienia – urządzenie zużywa minimalną liczbę energii, konieczną do podtrzymania zawartości pamięci oraz ewentualnych innych energooszczędnych peryferiów.
- Tryb aktywny – pomiaru i wysyłania danych – urządzenie jest wybudzone i w możliwie krótkim czasie agreguje i wysyła dane do węzła sieci i/lub infrastruktury chmurowej.

W takim schemacie działania, szybkość zużycia baterii podlega wpływowi następujących parametrów urządzenia:

- częstotliwość pomiarów i wysyłania informacji,
- czas pracy w stanie aktywnym,
- prąd zużywany w trybie uśpienia,
- prąd zużywany w trybie aktywnym.

Aby przeprowadzić pomiar i komunikację, należy ustalić oczekiwaną liczbę aktywacji. Przykładowo, urządzenie może dokonywać pomiarów i przesyłać dane co godzinę, co wymaga jednokrotnego wybudzenia w każdej godzinie, a w pozostałej części godziny może pozostawać w trybie uśpienia.

Szacunkowa metoda określenia średniego prądu pobieranego przez układ

Szacunkowe pomiary czasu kodowania zostały wykonane z wykorzystaniem oscyloskopu oraz analizatora stanów logicznych, poprzez pobudzanie wybranych portów wyjściowych jednostki w kluczowych momentach realizacji algorytmu i monitorowaniu zmian stanów logicznych na wyjściach. Przy użyciu przyrządów pomiarowych oraz odpowiednich sond i przystawek można mierzyć zależności czasowe w układzie. Taka metoda może być używana do obserwacji i analizy aktywności układu przy różnych warunkach obciążenia. Możliwe jest śledzenie zmian sygnałów, co umożliwia identyfikację i pomiar czasu trwania poszczególnych trybów (uśpienia / aktywne), a stąd – szacowania średniej wartości pobieranego prądu. Oczywiście realizacja protokołu

		Liczba elementów niezerowych	Rozmiar w pamięci	Rozmiar w pamięci macierzy mnożącej	
Macierz	Rząd ciała Galois		indeks - 2 BAJTY wartość - 2 BAJTY	wartość - 1 BAJT	Suma
HN1	$GF(2)$	2880	11520	4	11524
$R = 0.5$	$GF(2^2)$	1272	5088	16	5104
$N_b = 960$	$GF(2^3)$	816	3264	64	3328
	$\mathbf{GF(2^4)}$	**576**	**2304**	**256**	**2560**
	$GF(2^5)$	440	1760	1024	2784
	$GF(2^6)$	352	1408	4096	5504
	$GF(2^7)$	284	1136	16384	17520
	$GF(2^8)$	240	960	65536	66496
HN2	$GF(2)$	1440	5760	4	5764
$R = 0.5$	$GF(2^2)$	636	2544	16	2560
$N_b = 480$	$GF(2^3)$	404	1616	64	1680
	$\mathbf{GF(2^4)}$	**288**	**1152**	**256**	**1408**
	$GF(2^5)$	220	880	1024	1904
	$GF(2^6)$	176	704	4096	4800
	$GF(2^7)$	142	568	16384	16952
	$GF(2^8)$	120	480	65536	66016
HN3	$GF(2)$	1920	7680	4	7684
$R = 0.75$	$GF(2^2)$	848	3392	16	3408
$N_b = 640$	$GF(2^3)$	540	2160	64	2224
	$\mathbf{GF(2^4)}$	**384**	**1536**	**256**	**1792**
	$GF(2^5)$	292	1168	1024	2192
	$GF(2^6)$	236	944	4096	5040
	$GF(2^7)$	196	784	16384	17168
	$GF(2^8)$	160	640	65536	66176
HN4	$GF(2)$	960	3840	4	3844
$R = 0.75$	$GF(2^2)$	424	1696	16	1712
$N_b \approx 320$	$GF(2^3)$	270	1080	64	1144
	$\mathbf{GF(2^4)}$	**192**	**768**	**256**	**1024**
	$GF(2^5)$	148	592	1024	1616
	$GF(2^6)$	118	472	4096	4568
	$GF(2^7)$	98	392	16384	16776
	$GF(2^8)$	80	320	65536	65856

Tablica 1. Rozmiar macierzy w zależności od rzędu wielomianu ciała Galois.

komunikacji wymaga wykonania szeregu operacji, nie tylko kodowania kanałowego, jednakże w niniejszym artykule skupiono się na analizie czasu operacji kodowania i wynikających z tego konsekwencji.

Celem badania było oszacowanie średniego prądu pobieranego przez koder przy periodycznych zmianach trybu (aktywny / uśpienia), a w konsekwencji szacowanie czasu, w którym układ może być zasilany bez wymiany baterii, przy założonych warunkach (mikrokontroler, bateria, okres pomiędzy transmisjami wiadomości). Metoda pomiarowa umożliwia przeprowadzanie eksperymentów przy użyciu całego spektrum sprzętu, jednak wybów mikrokontrolera nie powinien mieć zasadniczego wpływu na wnioski: porównanie kodów o różnych rzędach ciała GF.

Mikrokontroler	STM32L476RGT6
Zestaw ewaluacyjny	Nucleo L476RZ
Urządzenia pomiarowe	Analizator stanów logicznych Saleae Oscyloskop cyfrowy
Zasilanie	Bateria 3.6V Fanso ER26500M

Tablica 2. Wykaz sprzętu wykorzystanego w eksperymentach

Wyniki

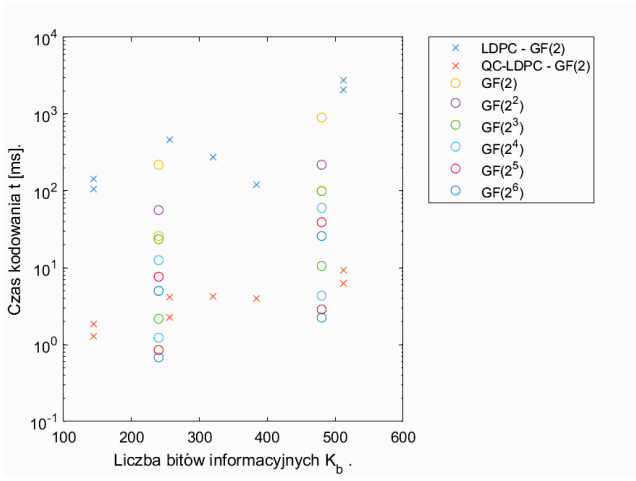
Badania eksperymentalne wykonano z wykorzystaniem sprzętu wymienionego w tab. 2.

Eksperymenty przeprowadzono dla 4 różnych szeregów kodów, oznaczanych dalej HN1...HN4, których parametry podano w tab. 1: rozmiar bloku (N_b – wyrażony w bitach) oraz sprawność kodu (R). W każdym szeregu zawarte są kody o takich samych rozmiarach bloków N_b (lub bardzo zbliżonych, w szczególności dla $q = 7$) i sprawności R, ale skonstruowane nad ciałami Galois o różnych rzędach: $GF(2) \dots GF(2^8)$. Wykorzystano macierze kontrolne kodów binarnych regularnych oraz niebinarnych semiregularnych. Większa sprawność kodu wymaga mniejszej liczby obliczeń przy kodowaniu, natomiast możliwości korekcji błędów są wtedy mniejsze. Długość bloku kodowego w ciele $GF(2^q)$ wynosi dla poszczególnych kodów $N = N_b/q$, a co za tym idzie – liczba elementów niezerowych w macierzy maleje wraz ze wzrostem q.

Każdy element niezerowy jest zapisany w pamięci w reprezentacji macierzy rzadkiej, zawierającej indeks elementu (2 bajty) oraz wartość elementu (2 bajty). Biorąc

Macierz	Rząd	Wykorzystanie baterii w skali roku	Prąd średni [μA]
HN1	$GF(2)$	17,34%	69,49
	$GF(2^2)$	4,22%	33,55
	$GF(2^3)$	1,91%	27,24
	$GF(2^4)$	1,16%	25,17
	$GF(2^5)$	0,75%	24,06
	$GF(2^6)$	0,50%	23,37
HN2	$GF(2)$	4,22%	33,56
	$GF(2^2)$	1,08%	24,97
	$GF(2^3)$	0,45%	23,24
	$GF(2^4)$	0,24%	22,66
	$GF(2^5)$	0,15%	22,40
	$GF(2^6)$	0,10%	22,26
HN3	$GF(2)$	1,93%	27,28
	$GF(2^2)$	0,50%	23,37
	$GF(2^3)$	0,20%	22,56
	$GF(2^4)$	0,08%	22,23
	$GF(2^5)$	0,06%	22,15
	$GF(2^6)$	0,04%	22,12
HN4	$GF(2)$	0,18%	23,37
	$GF(2^2)$	0,10%	22,26
	$GF(2^3)$	0,04%	22,11
	$GF(2^4)$	0,02%	22,06
	$GF(2^5)$	0,02%	22,04
	$GF(2^6)$	0,01%	22,04

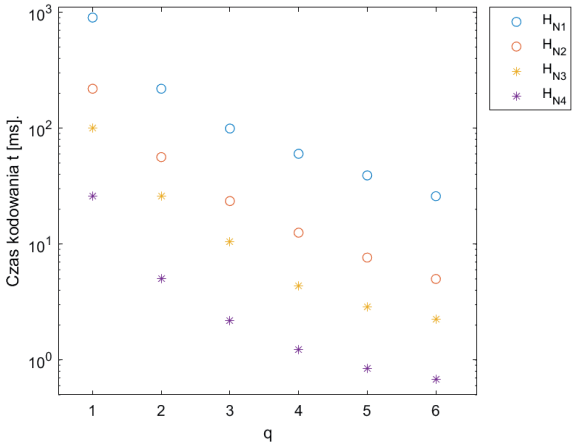
Tablica 3. Wykorzystanie baterii w skali roku dla algorytmów kodujących w ciele Galois rzędu wyższych rzędów $GF(2)$ - $GF(2^6)$. Dane dotyczą samego procesu kodowania przy założeniu jednej wiadomości na godzinę.



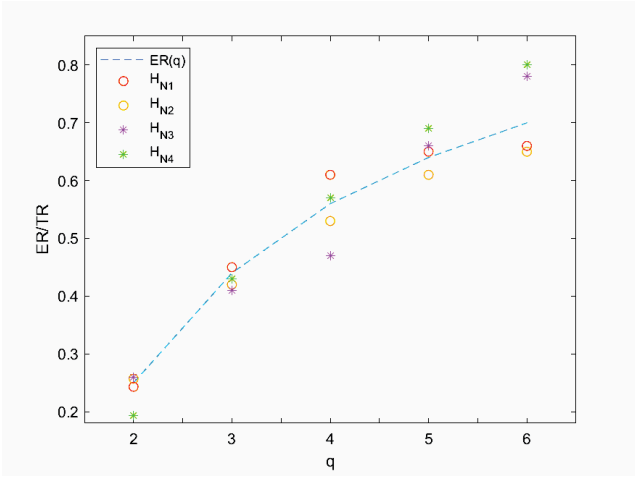
Rys. 7. Wykres zależności czasu kodowania od wielkości macierzy kontroli parzystości dla różnych schematów kodowania - binarne LDPC, QC-LDPC oraz niebinarne LDPC

to pod uwagę, określono i sprawdzono eksperymentalnie rozmiar pamięci potrzebny na zapisanie macierzy kontrolnej, co przedstawiono w tab. 1.

W tabeli podano także rozmiary tablicy wykorzystywanej w operacji mnożenia, która rośnie wraz ze wzrostem q.



Rys. 8. Wykres zależności czasu kodowania od rzędu q ciała Galois kodera dla różnych macierzy kontroli parzystości kodów niebinarnych.



Rys. 9. Teoretyczne i eksperymentalne wyniki stosunków czasów kodowania.

Po zsumowaniu rozmiarów pamięci wymaganej do zapisania macierzy kontrolnej oraz tablicy mnożącej, uzyskano wartości przedstawione w ostatniej kolumnie tabeli. Przedstawione wyniki pozwalają na sformułowanie interesującego wniosku. Otóż zauważono, że można określić taki rząd ciała, dla którego łączna wymagana pamięć ma minimalną wartość. Dla wszystkich szeregów kodów, minimum to przypada na $q = 4$, a zatem wykonane eksperymenty wskazują, że implementacja kodera w mikrokontrolerze będzie się cechowała najmniejszą wymaganą pamięcią dla kodów na ciałem $GF(2^4)$.

Drugą konsekwencją faktu, że rozmiar macierzy $\mathbf{H}$ zmniejsza się wraz ze zwiększaniem rzędu ciała GF, jest zmniejszenie liczby operacji w algorytmie kodującym. Proporcjonalnie mniej jest operacji, ale nad proporcjonalnie wyższym rzędem ciała, co jednakże w implementacji w mikrokontrolerze, z mnożeniem tablicowym, pozwala na znaczące zmniejszenie czasu kodowania. Wyniki dotyczące eksperymentalnych pomiarów czasu przedstawiono na rys. 7 oraz rys. 8, przy czym spektrum badanych kodów obejmuje kilka dodatkowych, oprócz HN1...HN4, badano także kody o innych rozmiarach macierzy.

Poczyniono także spostrzeżenie, że istnieje związek pomiędzy stosunkiem liczby elementów dwóch macierzy o kolejnych wartościach rzędu ciała q, $q + 1$, oznaczonym $ER(q)$, a stosunkiem określonych eksperymentalnie czasów

kodowania $TR(q)$ dla tych samych rzędów. Wspomniane wartości zdefiniowano we wzorach (4) oraz (5), a na rys. 9 zobrazowano związek pomiędzy tymi wartościami: linia przerywana obrazuje wyliczone $ER(q)$, natomiast eksperymentalne wyniki stosunków czasów $TR(q)$ są zaprezentowane wykresami punktowymi.

Niech M_q oraz N_q oznaczają – odpowiednio – liczbę wierszy i kolumn kodu nad ciałem $GF(2^q)$, więc dla określonego szeregu kodów o stałej długości słowa informacyjnego i kodowego N_b wyrażonego w bitach łatwo pokazać, że: $M_{q+1}/M_q = N_{q+1}/N_q = q/(q+1)$. Stąd stosunek liczby elementów w macierzach $GF(2^q)$ oraz $GF(2^{q+1})$ wynosi:

$$ER(q) = \frac{M_{q+1}}{M_q} \cdot \frac{N_{q+1}}{N_q} = (\frac{q}{q+1})^2 \tag{4}$$

Punkty na wykresie na rys. 9 oznaczają stosunek czasu kodowania zgodnie ze wzorem 5:

$$TR(q) = \frac{t_{q+1}}{t_q} \tag{5}$$

gdzie t_q jest czasem kodowania kodu nad ciałem $GF(2^q)$.

Eksperymentalnie określono i porównano czasy kodowania w odniesieniu do rozmiaru macierzy kontroli parzystości, a tym samym do rozmiaru słowa informacyjnego. Zauważono zależność związaną ze zmniejszaniem się czasu kodowania wraz ze zmniejszaniem się wielkości macierzy kontroli parzystości. Wyniki czasowe zostały zaprezentowane dla różnych macierzy na rys. 7. Poddane analizie zostały macierze z poprzedniej publikacji autorów [9] oraz porównane z kodami i kodowaniem niebinarnym. Krzyżykami zaznaczone zostały wyniki wcześniejsze [9]: w kolorze niebieskim – wyniki dla różnych macierzy kontroli parzystości dla ciała $GF(2)$ i efektywnego algorytmu kodującego LDPC; kolorem czerwonym – wyniki dla różnych macierzy kontroli parzystości dla ciała $GF(2)$ i efektywnego algorytmu kodującego QC-LDPC. Okręgami zostały oznaczone wyniki dla kolejnych rzędów ciała $GF(2^q)$ dla różnych rozmiarów macierzy. Rząd 6 ciała, $GF(2^6)$ pozwala na uzyskanie najlepszych wyników czasowych i zbliża się do poziomu algorytmu efektywnego kodowania binarnego QC-LDPC. Jednocześnie, kod niebinarny oferuje większe możliwości korekcyjne, co przekłada się na potencjał dodatkowej oszczędności energii w transmisji.

Zużycie energii

W celu zobrazowania poziomu energii zużywanej przez układ kodera implementowanego w mikrokontrolerze oraz porównania poszczególnych kodów nad różnymi rzędami ciała, przeprowadzono szacunki zużycia energii i czasu pracy baterii przy założeniu stałej długości ramki danych i przechodzeniu w stan aktywności co określony czas – wyniki w tej publikacji przedstawiono dla aktywności z okresem jednej godziny. Założono wartości prądów i napięć takie jak w środowisku eksperymentalnym (wymienionym w tab. 2):

- W trybie aktywnym urządzenie zużywa stały prąd na poziomie 190mA, a w trybie uśpienia – 22μA.
- Napięcie baterii to 3.6, natomiast napięcie zasilania układu mikrokontrolera to 3.3V

Energię można wyrazić wzorem:

$$E = U \cdot Q = U \cdot I \cdot t \tag{6}$$

gdzie E to energia elektryczna wyrażona w [J], U to napięcie elektryczne wyrażone w [V], Q to ładunek elektryczny wyrażony w [C], I to natężenie prądu wyrażone w [A] oraz t to czas [s].

Zgromadzone eksperymentalnie dane są sprowadzane do jednostek podstawowych i korzystając ze wzoru 6 określana jest energia. Należy także określić pojemność baterii: wybrano baterię firmy Fanso ER26500M, o napięciu znamionowym 3.6V oraz pojemności 2200mAh, wyrażona w $[As]$ wynosi:

$$E_{bat}[J] = 3.6[V] \cdot 2.2[A] \cdot 3600[s] \tag{7}$$

Z 7 energia baterii wynosi $E_{bat} = 28512[J]$. Jest to energia zgromadzona w baterii i dostępna do wykorzystania. Urządzenie działające w trybie uśpienia z poborem prądu na poziomie 22μA w skali roku pobiera około 2300J energii z baterii co stanowi około 8% jej nominalnej pojemności. Przedstawione rozważania są uproszczone i nie zakładają wpływu prądu obciążenia i temperatury na pojemność oraz efektów starzenia baterii.

Obliczenia dotyczą przypadku, gdy urządzenie wysyła dane raz na godzinę, co daje 8760 wiadomości rocznie. W zależności od wybranej macierzy kontroli parzystości oraz stopnia wielomianu ciała Galois, poziom wykorzystania baterii w skali jednego roku może osiągać od około 0,01% do nawet ponad 50%. Widoczna jest zależność wprost proporcjonalna do czasu kodowania. Zmniejszenie macierzy $\mathbf{H}$ powoduje znaczne zmniejszenie liczby elementarnych operacji wymaganych do zakodowania danych.

Przeliczono wykorzystanie baterii na periodyczne kodowanie wiadomości, a wyniki zgromadzono w formie prądu uśrednionego (łącznie w trybach aktywnym i uśpienia) oraz zużycia baterii w perspektywie rocznej, w tabeli 3. Wpływ kodera na zużycie baterii można zmniejszyć nawet kilkudziesięciokrotnie wykorzystując kody niebinarne wyższych rzędów. Rozwiązanie to przynosi wymierny zysk energetyczny. Należy pamiętać, że czas kodowania jest ułamkiem całej procedury pomiarowo-transmisyjnej na która mogą składać się: wykonanie pomiaru, opracowanie wyniku, przygotowanie wiadomości, zakodowanie wiadomości, wysłanie wiadomości. Nie zmienia to faktu, że należy dążyć do możliwie jak najkrótszego czasu kodowania, aby nie wprowadzać dodatkowych opóźnień w transmisji danych oraz ograniczyć zużycie energii.

Wnioski

W artykule zaprezentowano implementację programową nowoczesnych metod kodowania korekcyjnego NB-LDPC oraz wyniki implementacji w typowych układach wbudowanych – mikrokontrolerach. Przeprowadzone badania otwierają perspektywy wykorzystania tych metod w komercyjnych rozwiązaniach Internetu Rzeczy, co powinno pozwolić na ograniczenie zużycia energii w realizacji protokołów najniższych warstw komunikacyjnych. Zaproponowane algorytmy poszerzają aktualną wiedzę dotyczącą metod kodowania w systemach o ograniczonych mocach obliczeniowych, przy minimalizacji opóźnienia i wymaganej pamięci. Mogą być efektywnie zastosowane w systemach zasilanych bateriami. Zaprezentowano potencjał kodów niebinarnych, które po stronie kodera (nadajnika) mogą mieć lepsze parametry czasowe niż kody binarne, jednocześnie zapewniając lepsze możliwości korekcyjne. Problemem może pozostać kwestia zwiększonej złożoności obliczeniowej dekodera, lecz w systemach z centralnym węzłem o dużych możliwościach

obliczeniowych może to być ograniczeniem nieistotnym, a zaprezentowany potencjał kodów NB wskazuje na dalsze możliwe kierunki badań i rozwoju w tej tematyce.

***Autorzy**:*
- Jakub Hyla
TKH Technology Poland Sp. z.o.o.
email: jakubhyla93@gmail.com,
- Ph.D. Wojciech Sułek
Institute of Automatic Control and Robotics, Electrinics and Telefomunication, Silesian University of Technology ul. Akademicka 2a, 44-100 Gliwice, Poland,

This research was co-financed by the Ministry of Education and Science of Poland under grant No. DWD/3/7/2019.

LITERATURA

[1] Ieee 802.11-2016. ieee standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks–specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications, 2016.

[2] Salima Belhadj and Moulay Lakhdar Abdelmounaim. On error correction performance of ldpc and polar codes for the 5g machine type communications. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pages 1–4, 2021.

[3] Vishal A. Dubal and Y. Srinivasa Rao. A low-power high-performance sensor node for internet of things. In *IEEE Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 607–612, Madurai, India, June 2018.

[4] ETSI Standard: EN 302 307 v1.1.1, *Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications*, 2005.

[5] Marc P. C. Fossorier, Miodrag Mihaljevic, and Hideki Imai. Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation. *IEEE Transactions on Communications*, 47, No. 5:673–680, May 1999.

[6] Robert G. Gallager. Low-Density Parity-Check Codes. *IRE Transactions on Information Theory*, IT-8:21–28, January 1962.

[7] Jakub Hyla and Wojciech Sułek. Dekoder ldpc implementowany w mikrokontrolerze dla systemów internetu rzeczy. *Przedląd Elektrotechniczny*, (04/2023):133, 2023.

[8] Jakub Hyla and Wojciech Sułek. Energy-efficient raptor-like ldpc coding scheme design and implementation for iot communication systems. *Energies*, 16(12), 2023.

[9] Jakub Hyla, Wojciech Sułek, Weronika Izydorczyk, Leszek Dziczkowski, and Wojciech Filipowski. Efficient ldpc encoder design for iot-type devices. *Applied Sciences*, 12(5), 2022.

[10] IEEE Standard: IEEE P802.11n=D10. Draft IEEE Standard for Local Metropolitan Networks – Specific requirements. Part 11: Wireless LAN Medium Access Control (MAC), and Physical Layer (PHY) specifications: Enhancements for Higher Throughput, March 2006.

[11] Shu Lin and Daniel J. Costello, Jr. *Error Control Coding: Fundamentals and Applications, 2nd Edition*. Prentice-Hall, Inc., Upper Saddle River, New Jersey 07458, 2004.

[12] David J. C. MacKay. Good Error-Correcting Codes Based on Very Sparse Matrices. *IEEE Trans. Inf. Theory*, 45:399–431, March 1999.

[13] David J. C. MacKay and Radford M. Neal. Near Shannon Limit Performance of Low Density Parity Check Codes. *Electronics Letters*, 32:1645–1646, August 1996.

[14] Jérémy Nadal and Amer Baghdadi. Parallel and flexible 5g ldpc decoder architecture targeting fpga. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 29(6):1141–1151, 2021.

[15] Vladimir L. Petrović, Dragomir M. El Mezeni, and Andreja Radošević. Flexible 5g new radio ldpc encoder optimized for high hardware usage efficiency. *Electronics*, 10(9), 2021.

[16] C Rajasekaran and K Raguvaran. Microcontroller based reconfigurable iot node. In *IEEE 4th International Conference on Frontiers of Signal Processing*, pages 12–16, Poitiers, France, September 2018.

[17] Thomas J. Richardson and Rüdiger L. Urbanke. Efficient Encoding of Low-Density Parity-Check Codes. *IEEE Trans. Inf. Theory*, 47:638–656, February 2001.

[18] S. Narasimha Swamy and Salomon Raju Kota. An empirical study on system level aspects of internet of things (iot). *IEEE Access*, 8:188082–188134, 2020.

[19] Chance Tarver, Matthew Tonnemacher, Hao Chen, Jianzhong Zhang, and Joseph R. Cavallaro. Gpu-based, ldpc decoding for 5g and beyond. *IEEE Open Journal of Circuits and Systems*, 2:278–290, 2021.