Kacper Sionkowski
Projekt

Temat: Jednostka arytmetyczna z detekcją błędów.

1. Wprowadzenie:

Celem projektu było zaimplementowanie jednostki arytmetycznej będącej wyposażoną w obwód wykrywający błędy wykonywanych operacji. Dostępne operacje to dodawanie i mnożenie na dwóch bitach, zgodnie z założonym planem projektu jednostka została zaimplementowana w oparciu o załączony artykuł naukowy.

2. Schemat ogólny.

Poniższy rysunek przedstawia implementowany układ, A i B to dwu bitowe słowa wejściowe a $|A|_3$ i $|B|_3$ to dwu bitowe kody resztkowe użyte do weryfikacji obliczeń. Przy takich założeniach i dozwolonych operacjach jednostki arytmetycznej tj. 0 – dodawanie i 1 – mnożenie, największe słowo wyjściowe może być reprezentowane na trzech bitach a jej reszta na dwóch (przy założeniu dzielenia modulo 3).

modulo of $m = 3$ is used for residue codes in this thesis.

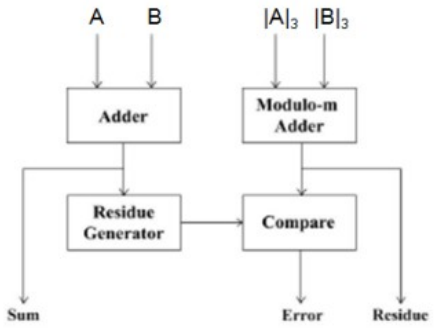
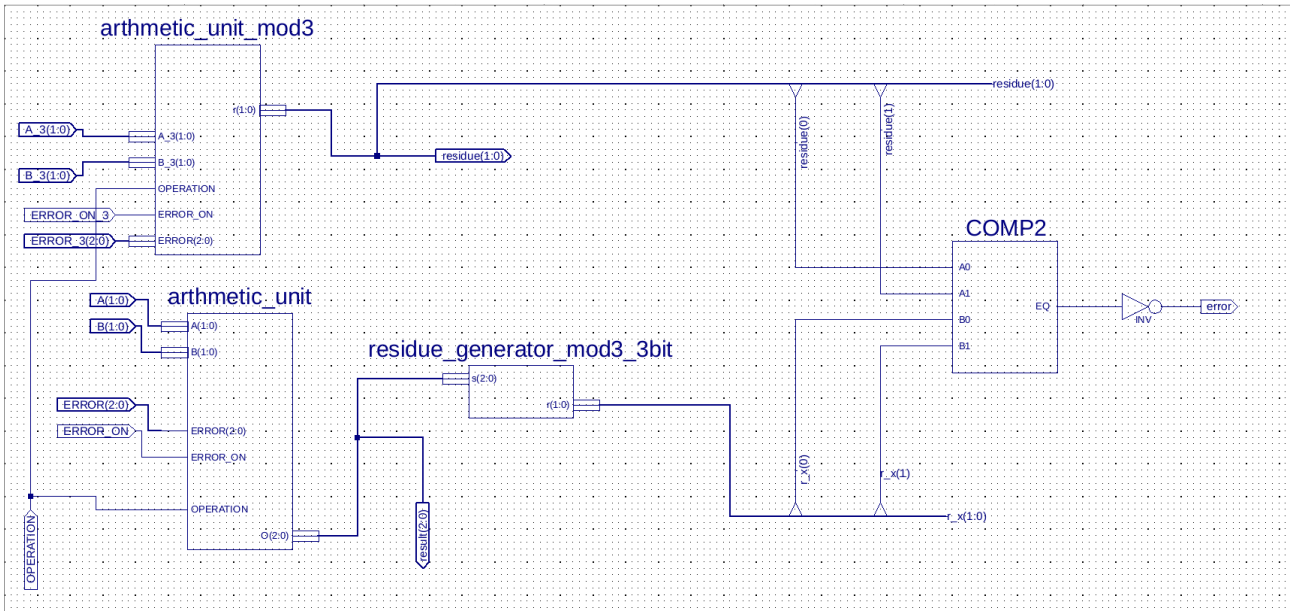


Figure 4: Residue code adder (or any arithmetic operation) [25].

Rysunek 1: Konceptualny schemat projektu.

W faktycznym schemacie projektu zastąpione zostały bloki Adder i Modulo-m Adder blokami jednostki arytmetycznej.

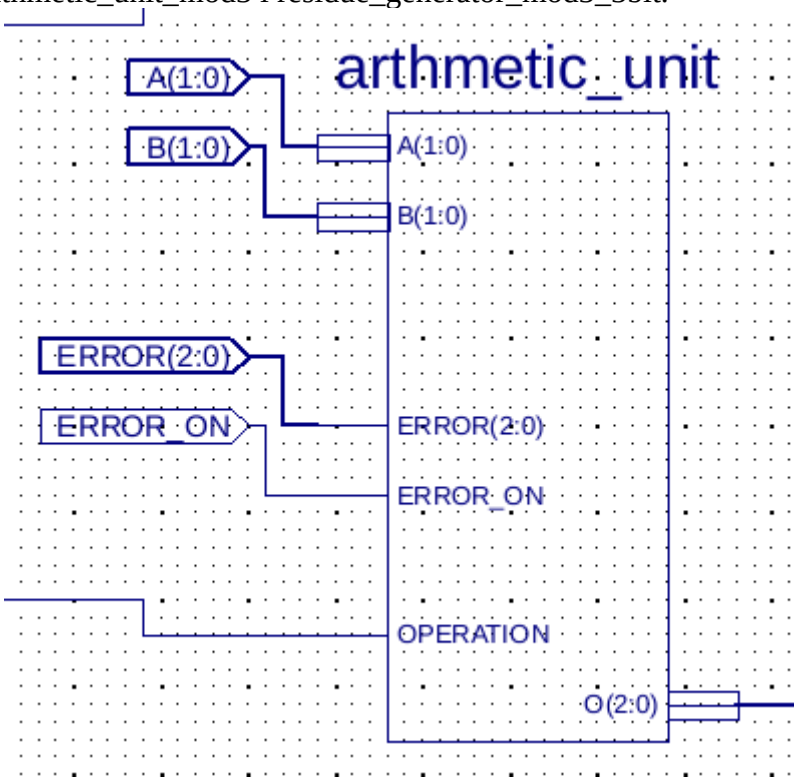


Rysunek 2. Faktyczny schemat projektu.

Wejścia takie jak A i B czy A_3 są zgodne ze schematem ogólnym, dodatkowymi wejściami są ERROR i ERROR_ON które pozwalają na wprowadzenie błędu obliczeń do schematu i tym samym zweryfikowanie działania układu.

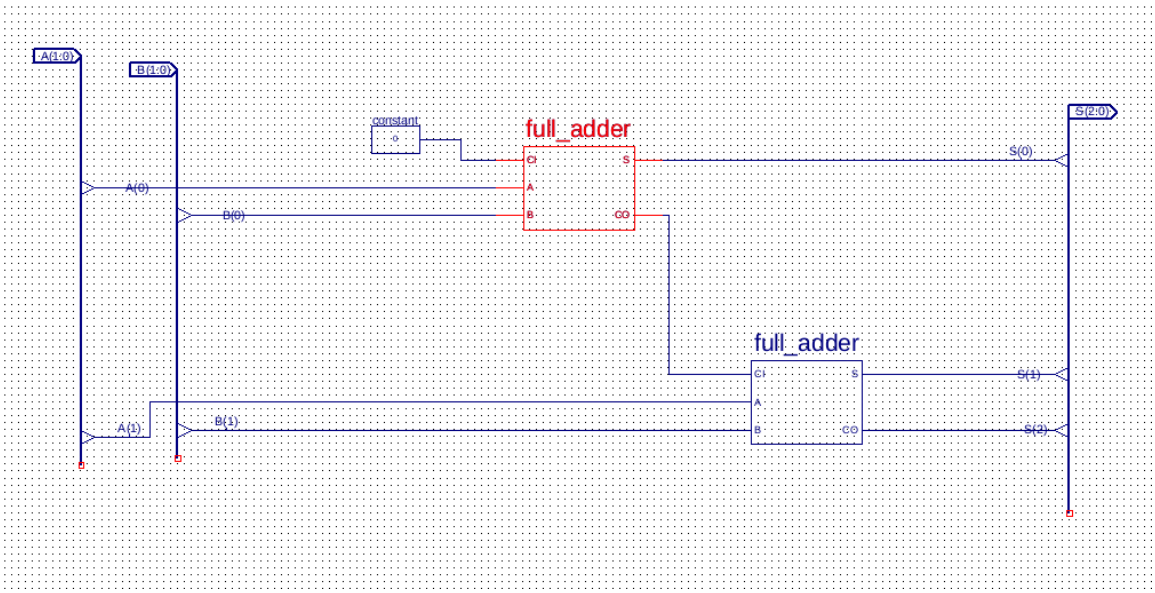
3. Opis poszczególnych elementów schematu:

Głównymi elementami pozwalającymi na poprawne funkcjonowanie układu są bloki arithmetic_unit, arithmetic_unit_mod3 i residue_generator_mod3_3bit.

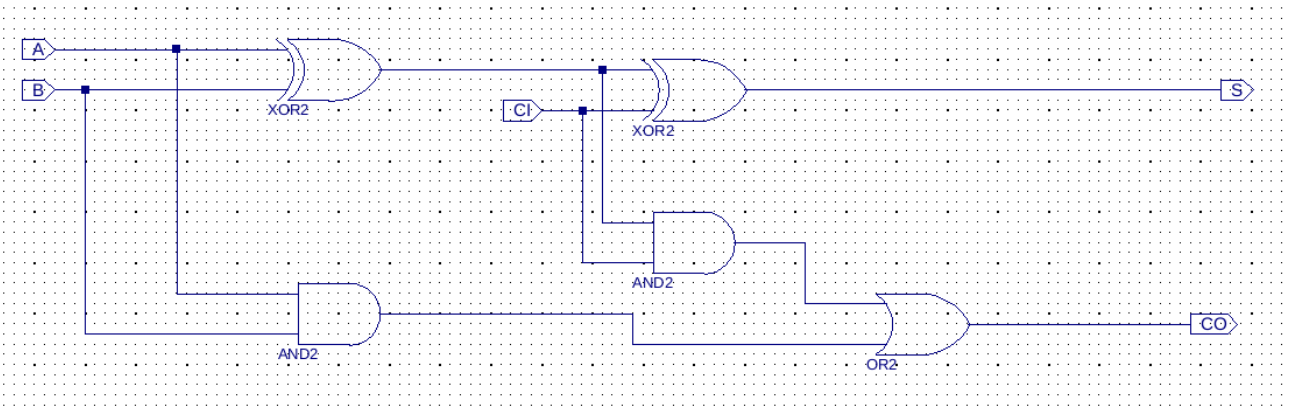


Rysunek 3. Jednostka arytmetyczna wraz z wejściami/wyjściami.

Sama jednostka arytmetyczna działa w oparciu o blok adder i multiplier. Jednostka sumująca została zaimplementowana w oparciu o proste sumatory pełne na które składają się dwie bramki XOR, dwie bramki and i jedna bramka or.

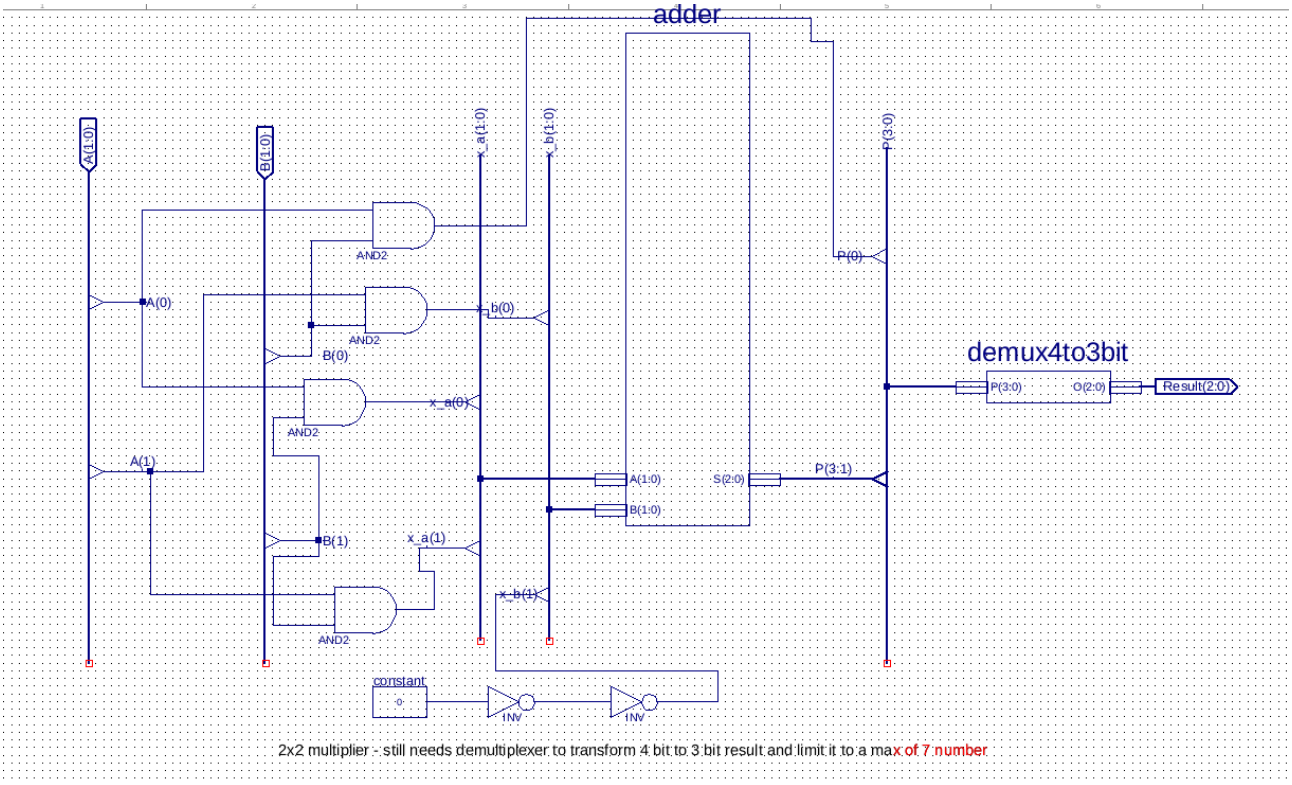


Rysunek 4. Schemat jednostki sumującej.



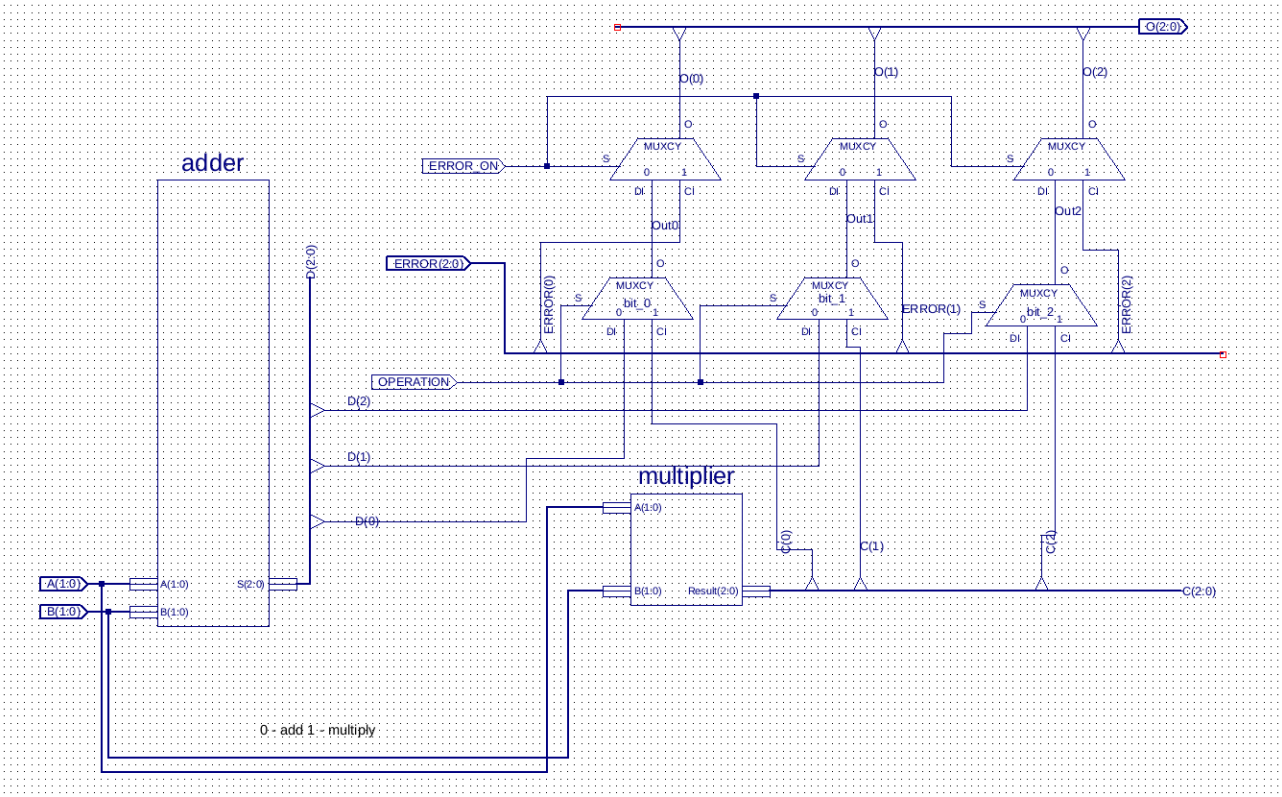
Rysunek 5. Schemat sumatora pełnego.

Jednostka mnożąca natomiast zbudowana została jako suma wyników mnożenia pomiędzy poszczególnymi bitami słowa wejściowego. Wynik mnożenia został przeskalowany tak aby można go było zapisać na 3 bitach tj. maksymalny wynik z mnożenia dwu bitowych słów nie może przekroczyć 7. Modyfikacja otrzymanego wyniku została zaimplementowana w oparciu o syntezę na siatkach Karnaugh.



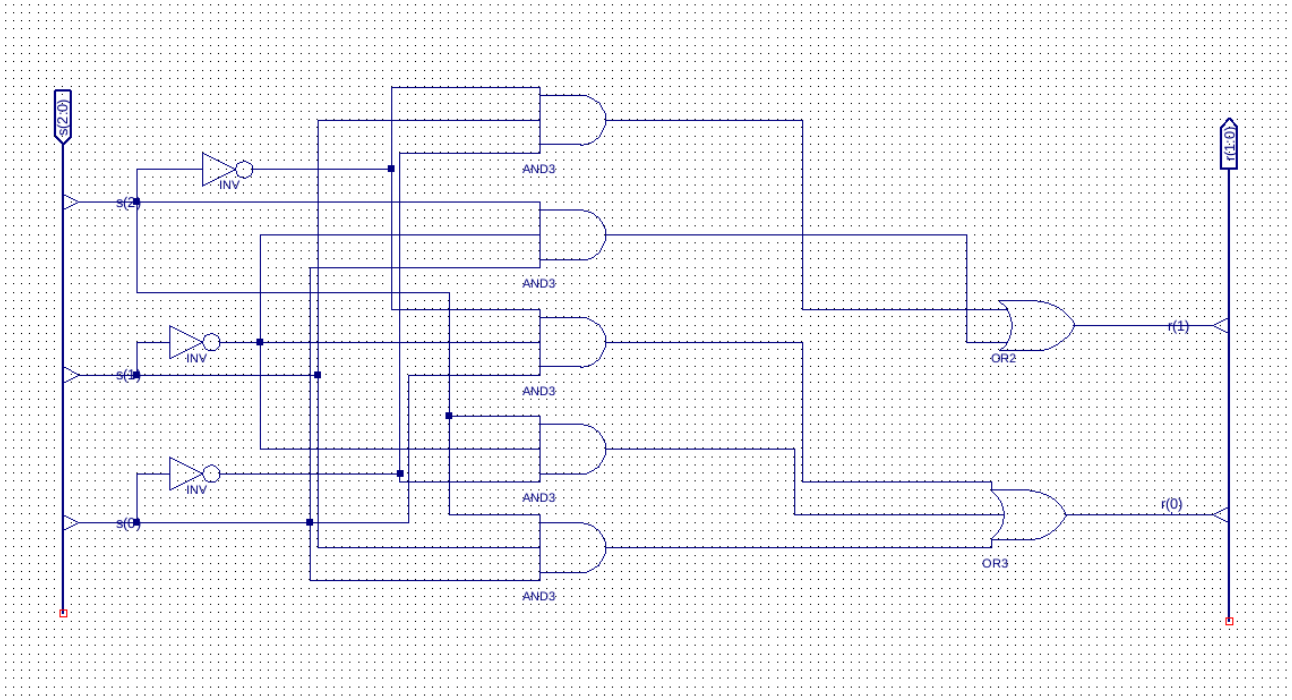
Rysunek 6. Schemat jednostki mnożącej.

Z wykorzystaniem powyżej opisanych bloków stworzony został schemat implementujący prostą jednostkę arytmetyczną sterowaną bitem OPERATION i pozwalającą na wybór między operacjami mnożenia/dodawania jak i również wprowadzenia błędu do układu za pomocą wejść ERROR i ERROR_ON.



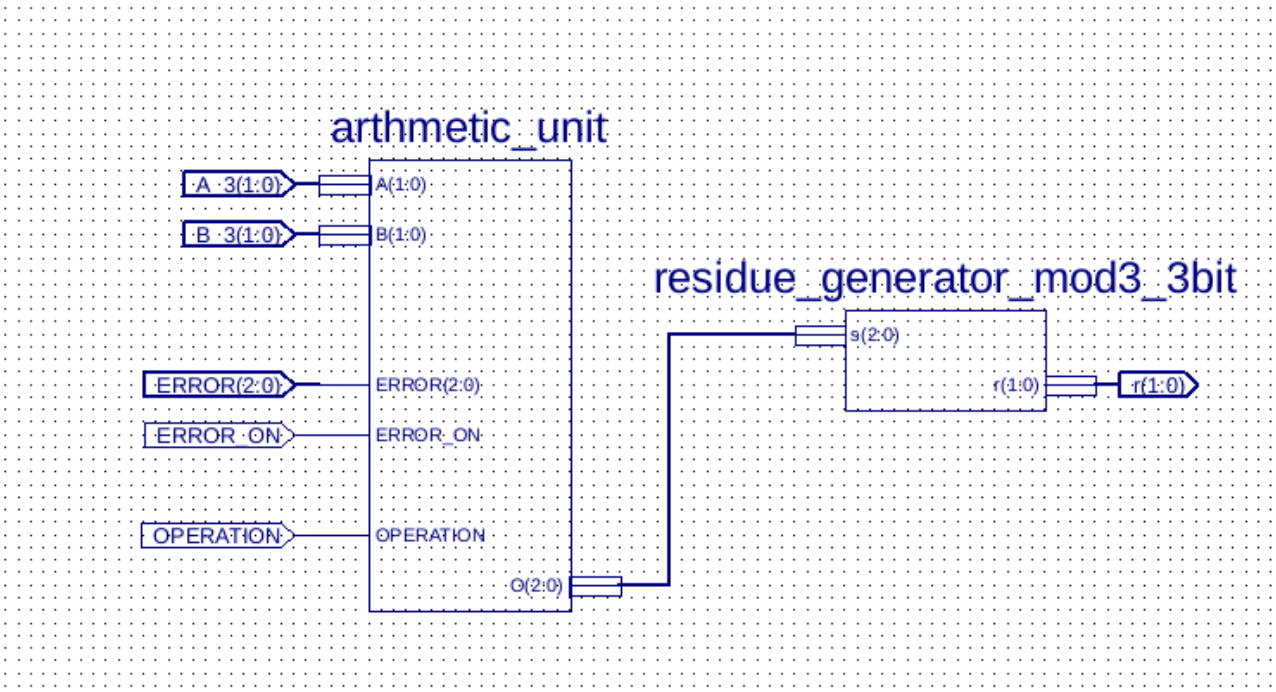
Rysunek 7. Schemat jednostki arytmetycznej.

Ostatnim istotnym elementem projektu jest residue_generator_mod3_3bit, który został stworzony w oparciu o tabelę prawdy, dla każdego 3 bitowego słowa wejściowego generowany jest kod resztkowy na 2 bitach który na koniec schematu porównywany jest w celu uzyskania kodu błędu.



Rysunek 8. Schemat generator kodów resztkowych modulo 3.

Sama jednostka arytmetyczna modulo 3 działa analogicznie do samej jednostki arytmetycznej, z otrzymanego wyniku generuje kod resztkowy aby go porównać. Różnica między obydwoma układami modulo 3 i zwykłej jednostki arytmetycznej polega na wprowadzaniu innych słów wejściowych – oczekiwane kody resztkowe. Zatem jednostka arytmetyczna mod 3 jest niezależna od właściwej jednostki arytmetycznej.



Rysunek 9. Schemat jednostki arytmetycznej kodów resztkowych.

4. Podsumowanie i wnioski:

Projekt pozwolił na skuteczne zaimplementowanie jednostki arytmetycznej, która wykonuje podstawowe operacje arytmetyczne - mnożenie i dodawanie, a także sprawdza ich poprawność. Układ skutecznie identyfikuje błędy obliczeń, co potwierdza efektywność założonych technik detekcji. Dodatkowe metody optymalizacji, takie jak synteza na siatkach Karnaugh, pozwoliły na efektywne skalowanie wyników mnożenia.