Automatyzacja procesu tworzenia sprzętowego narzędzia służącego do rozwiązywania zagadnienia logarytmu dyskretnego na krzywych eliptycznych

Autor:
Piotr Majkowski

Pod opieką:
prof. Zbigniew Kotulski

Politechnika Warszawska – Wydział Elektroniki i Technik Informacyjnych Instytut Telekomunikacji

Opis prezentacji

- **TŁO**
- **CEL**
- **WYMAGANIA**
- **METODA**
- **OSIĄGNIĘCIA**
- **WYZWANIA**

TŁO

System rozpraszania obliczeń z zastosowaniem w rozwiązywaniu zagadnienia logarytmu dyskretnego na krzywych eliptycznych

Piotr Majkowski
Praca inżynierska pod opieką: prof Zbigniewa Kotulskiego

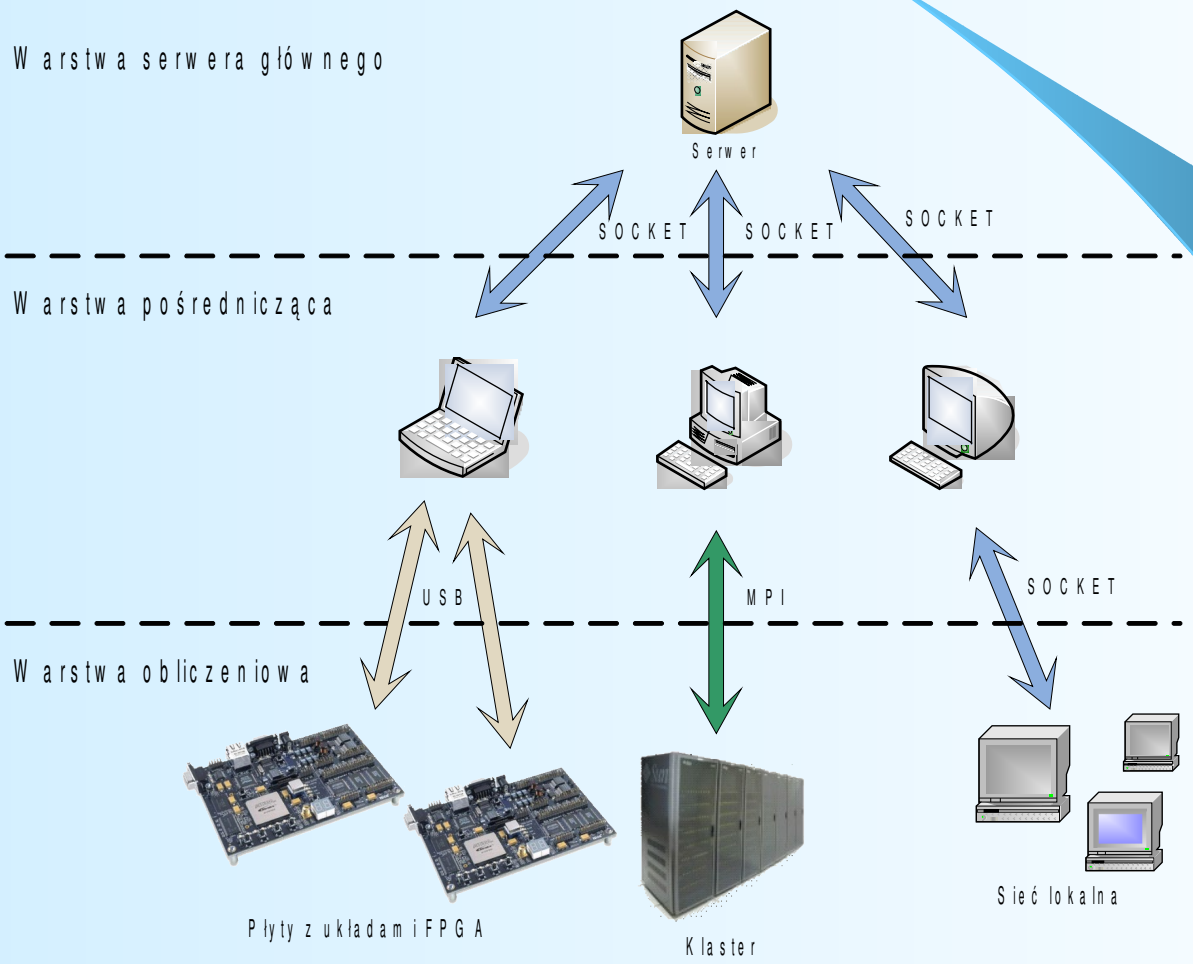
Cechy stworzonego systemu

- Niezależność od przeprowadzanych obliczeń.
- Niezależność od technologii przesyłania danych.
- Niezależność od środowiska systemowego.
- Skalowalność.
- Umożliwienie dynamicznych zmian liczby klientów obliczeniowych niezauważalnie dla algorytmu obliczeniowego

Wyniki implementacji

- Implementacja systemu w języku C
- Biblioteka krzywych eliptycznych oparta na MIRACL
- Obliczenia przeprowadzono:
  - Sieć lokalna (Windows i Unix / Socket i MPI).
  - Internet (Windows i Linux / Socket).
  - Klaster w ICM (Linux / MPI)

Heterogeniczny system obliczeniowy



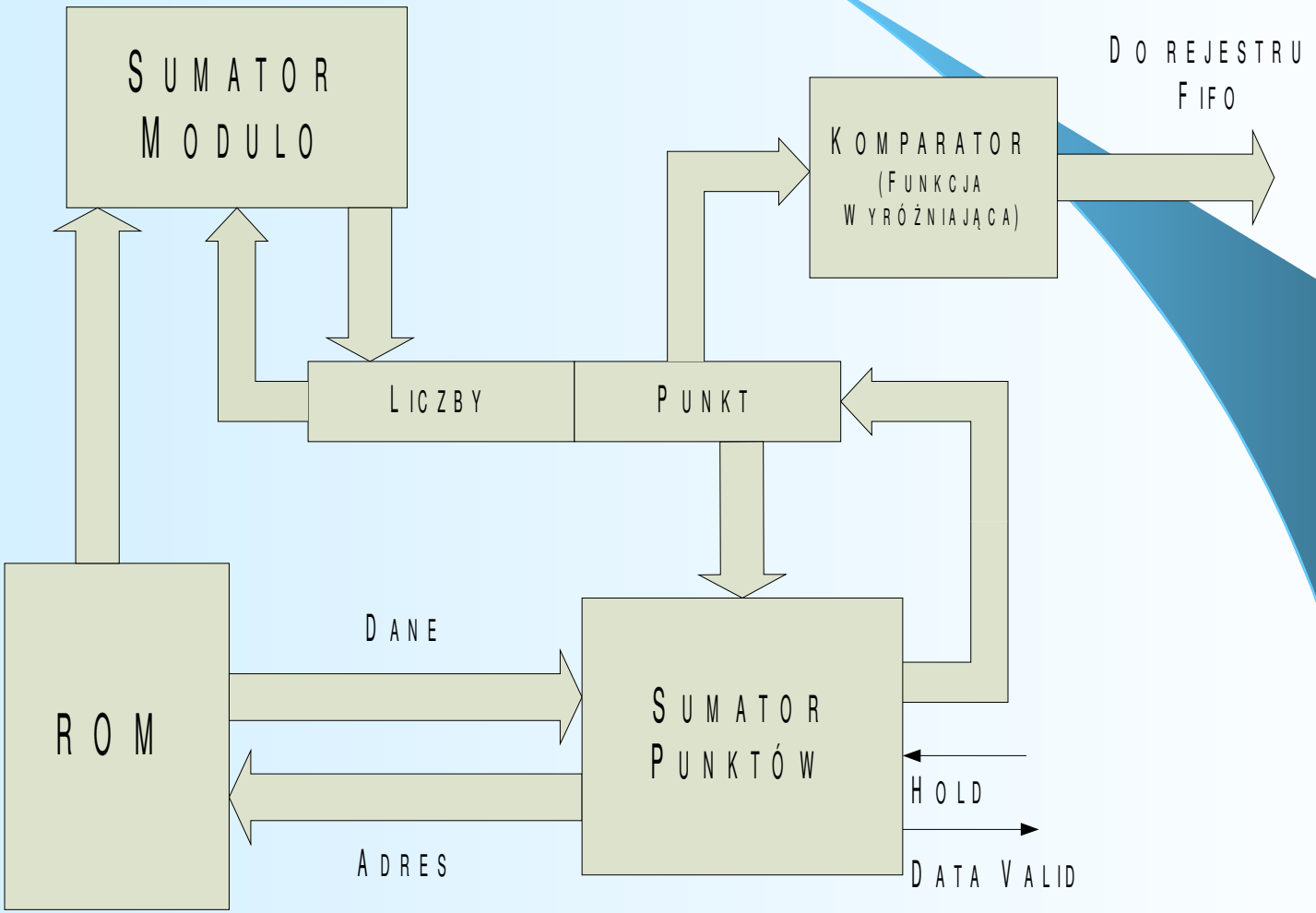
Wyzwanie

Certicom ECC Challenge

http://www.certicom.com

10 lat temu krzywą ECC2-89 złamano za pośrednictwem 70 komputerów w ok. 16 dni

Układ HardRho



Wyniki implementacji

Układ EP2S60F1020C4 z rodziny Stratix II (Altera)

Wykorzystane komórki logiczne:		
	9216	27593
Całkowite zużycie zasobów:		
	20%	58%
Częstotliwość zegara:		
	138 MHz	116 MHz
Efektywność obliczeń:		
	12.5 mln iter/sec	31,6 mln iter/sec

Porównanie z Certicom

Stacja Alpha 500 Mhz => 187 tysięcy iter/sec

Hard Rho 116 MHz => 31600 tysięcy iter/sec

Stosunek: HardRho/Alpha = 169

Szacowana długość obliczeń to około 8 dni.

RUC 2007

Realizacja jednostki wspomagającej kryptoanalizę szyfrów opartych na krzywych eliptycznych w strukturach reprogramowalnych

ENIGMA 2007

System sprzętowo - programowy do rozproszonej kryptoanalizy szyfrów opartych na krzywych eliptycznych

Piotr Majkowski, Tomasz Wojciechowski, Maciej Wojtyński, Mariusz Rawski

CEL

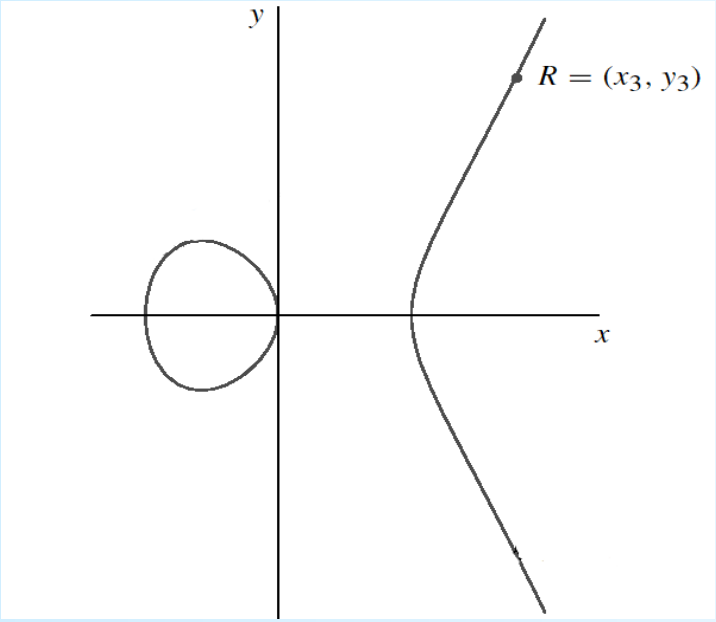
- Generacja kodu w VHDL układu HardRho dla dowolnego ciała
- Wykorzystanie szybkiej arytmetyki w ciele $GF(2^n)$ z zastosowaniem baz normalnych
- Uogólnienie rozwiązanie dla baz gausowskich
- Zapewnienie metody wymiany danych z systemami opartymi nie na ONB

Krzywe eliptyczne nad $GF(2^n)$

Krzywa eliptyczna ***E*** nad ciałem $GF(2^n)$ jest zdefiniowana przez następujące równanie:

$$y^2 + xy = x^3 + ax^2 + b$$

gdzie a, b $GF(2^n)$.



Ciało $GF(2^n)$ – ang. *Galois Field* - elementami ciała są binarne, n wymiarowe wektory współrzędnych w ustalonej bazie.

Bazy

Baza potęgowa

$$(1, \alpha, \alpha^2, \alpha^3, \alpha^4, \ldots, \alpha^{n-1})$$

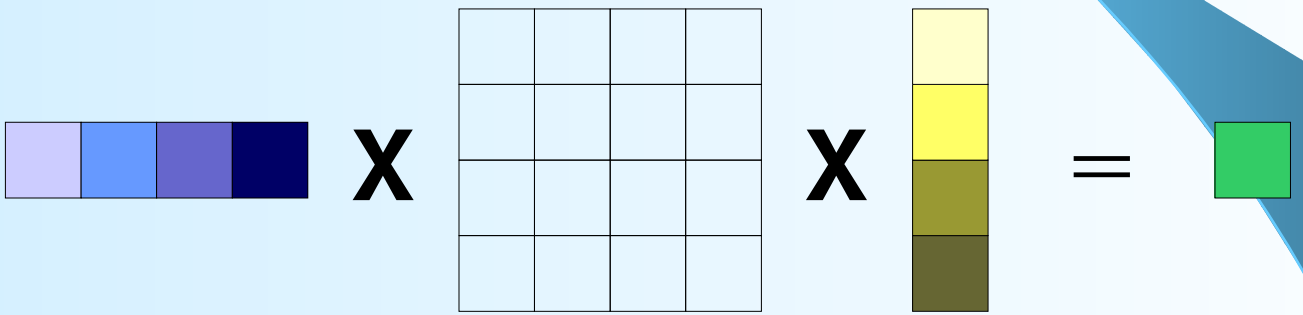
Baza normalna

$$(\beta, \beta^2, \beta^4, \beta^8, \ldots, \beta^{2^{n-1}})$$

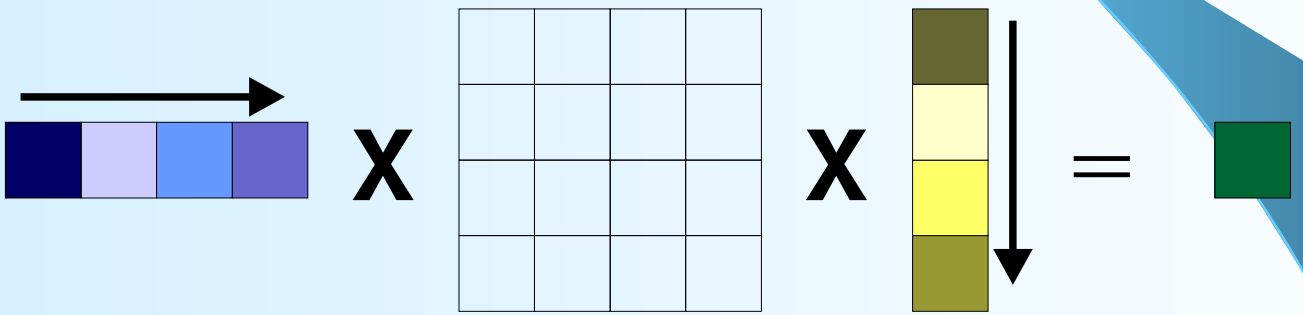
Operacje w ONB

- Dodawanie → XOR po wszystkich współrzędnych
- Podnoszenie do kwadratu → Cykliczna rotacja
- Mnożenie → Za pomocą macierzy mnożenia

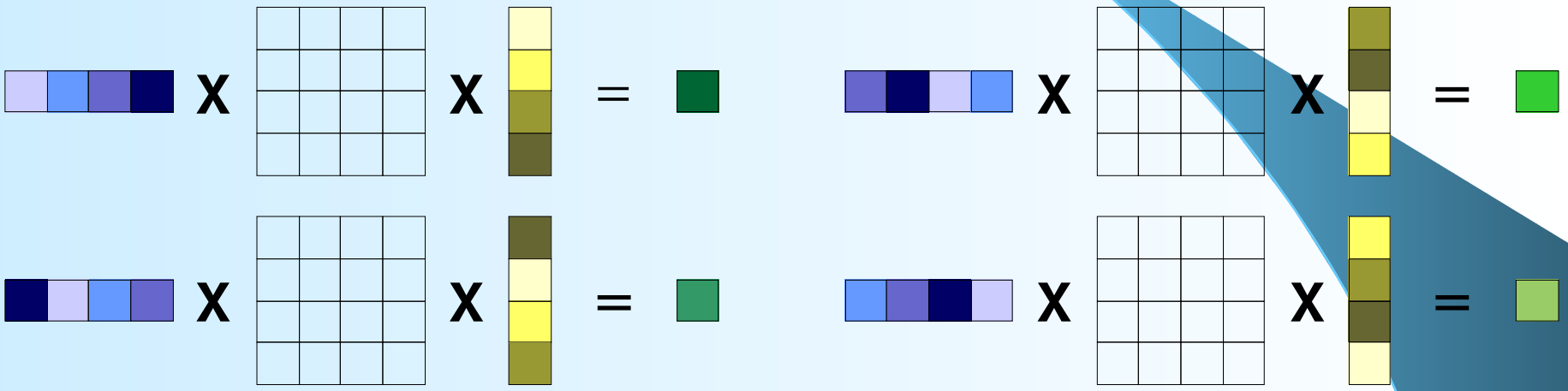
Mnożenie w ONB



Mnożenie w ONB architektura szeregowa



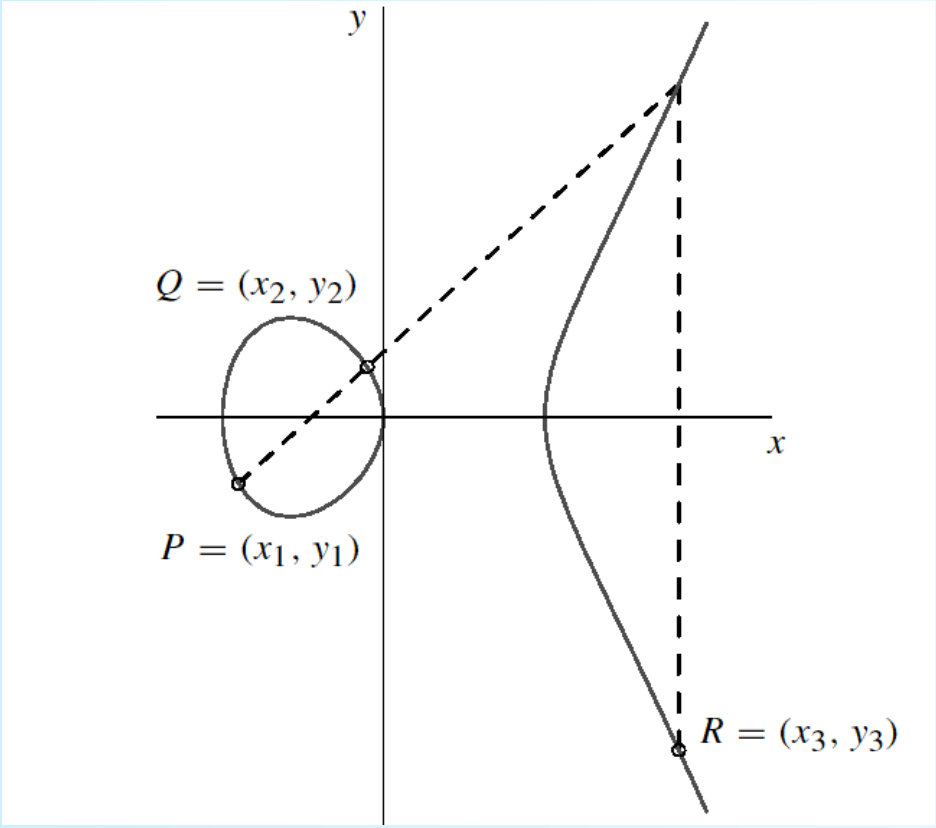
Mnożenie w ONB architektura równoległa



Biblioteka ciała skończonego

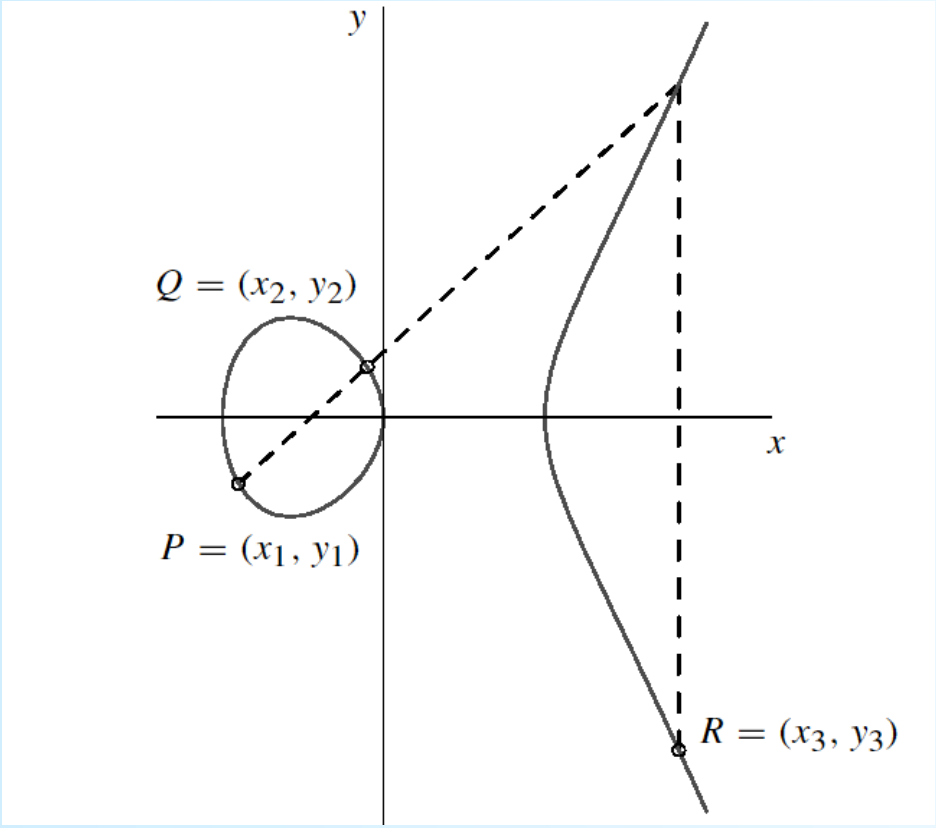
- Sprawdzanie występowania bazy dla (m, T)
- Obliczanie macierzy mnożenia
- Arytmetyka w ciele Gailos z wykorzystaniem baz normalnych
- Obliczanie wielomianu pierwotnego
- Obliczanie macierzy konwersji
- Import/Eksport pomiędzy bazami

Dodawanie punktów na krzywej eliptycznej



$$
\begin{aligned}
U_0 &= X_0 \cdot Z_1^2 \\
S_0 &= Y_0 \cdot Z_1^3 \\
U_1 &= X_1 \cdot Z_0^2 \\
W &= U_0 + U_1 \\
S_1 &= Y_1 \cdot Z_0^3 \\
R &= S_0 + S_1 \\
L &= Z_0 \cdot W \\
V &= R \cdot X_1 + L \cdot Y_1 \\
Z_2 &= L \cdot Z_1 \\
T &= R + Z_2 \\
X_2 &= a \cdot Z_2^2 + T \cdot R + W^3 \\
Y_2 &= T \cdot X_2 + V \cdot L^2
\end{aligned}
$$

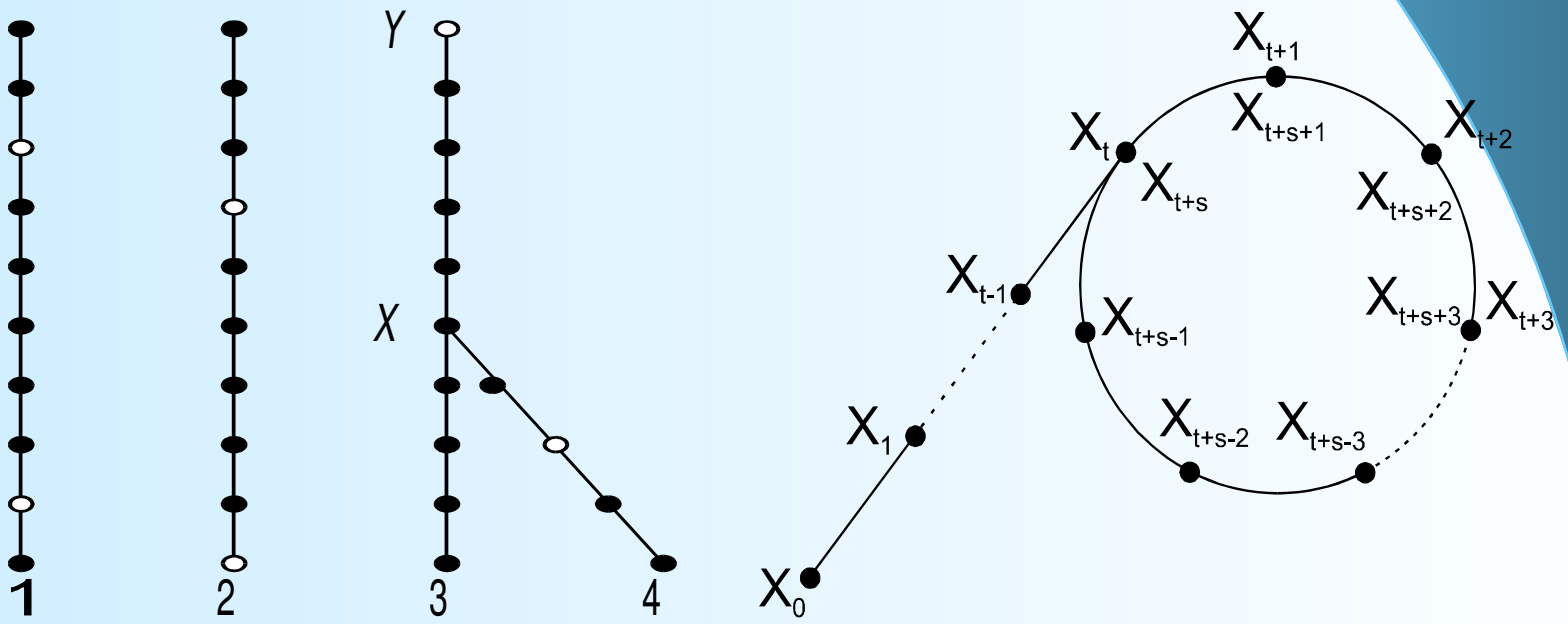
Dodawanie punktów na krzywej eliptycznej



$$U_0 = X_0 \cdot Z_1^2$$
$$S_0 = Y_0 \cdot Z_1^3$$
$$U_1 = X_1 \cdot Z_0^2$$
$$W = U_0 + U_1$$
$$S_1 = Y_1 \cdot Z_0^3$$
$$R = S_0 + S_1$$
$$L = Z_0 \cdot W$$
$$V = R \cdot X_1 + L \cdot Y_1$$
$$Z_2 = L \cdot Z_1$$
$$T = R + Z_2$$
$$X_2 = a \cdot Z_2^2 + T \cdot R + W^3$$
$$Y_2 = T \cdot X_2 + V \cdot L^2$$

Algorytm rho Pollarda

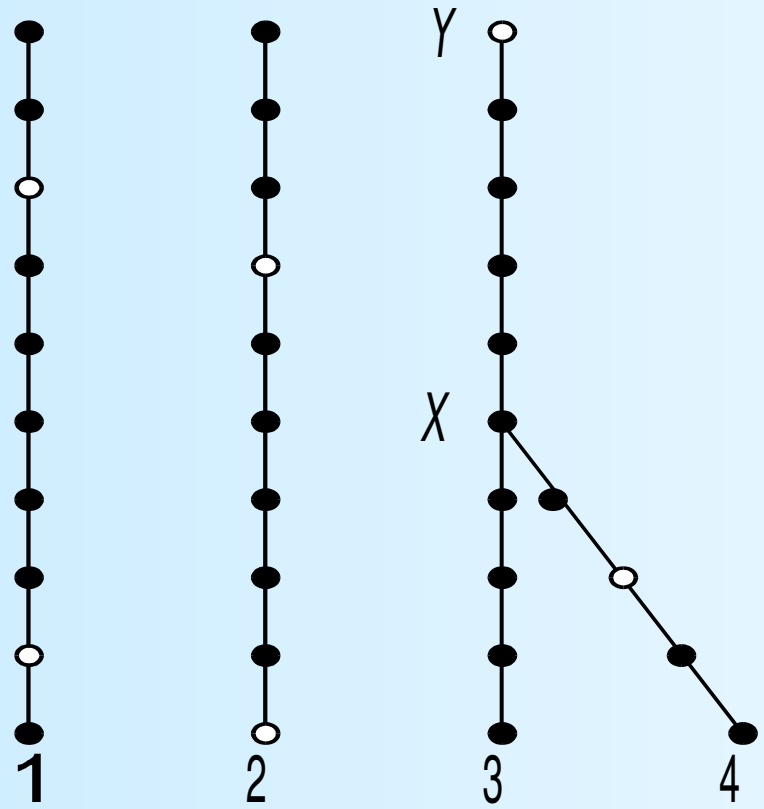
- Najlepszy znany obecnie algorytm rozwiązywania ECDLP
- Algorytm probabilistyczny oparty na błądzeniu przypadkowym
- Istnieje wersja równoległa
- Zapotrzebowanie na pamięć można ograniczać



Algorytm rho - komponenty

- Funkcja błądzenia przypadkowego
  - Bieżący punkt X
  - Funkcja pozycjonująca *H()*
- Tablice *a[], b[], R[]*
- Kryterium wyróżniające

Równoległy algorytm rho



- Wspólne dla procesorów:
  - funkcja błądzenia
  - tablice *a[16]*, *b[16]*, *R[16]*
  - kryterium wyróżnialności
- Unikalne dla procesorów:
  - punkt startowy

Algorytm rho

Każdy z procesorów wykonuje
a) Wybierz losowo c' oraz d' z $[0, n-1]$
b) Oblicz $X' = c'P + d'Q$

c) Wykonuj aż do napotkania kolizji

* Jeśli X' spełnia warunek wyróżnienia to wyślij (c', d', X') do serwera

* Oblicz $j = H(X')$

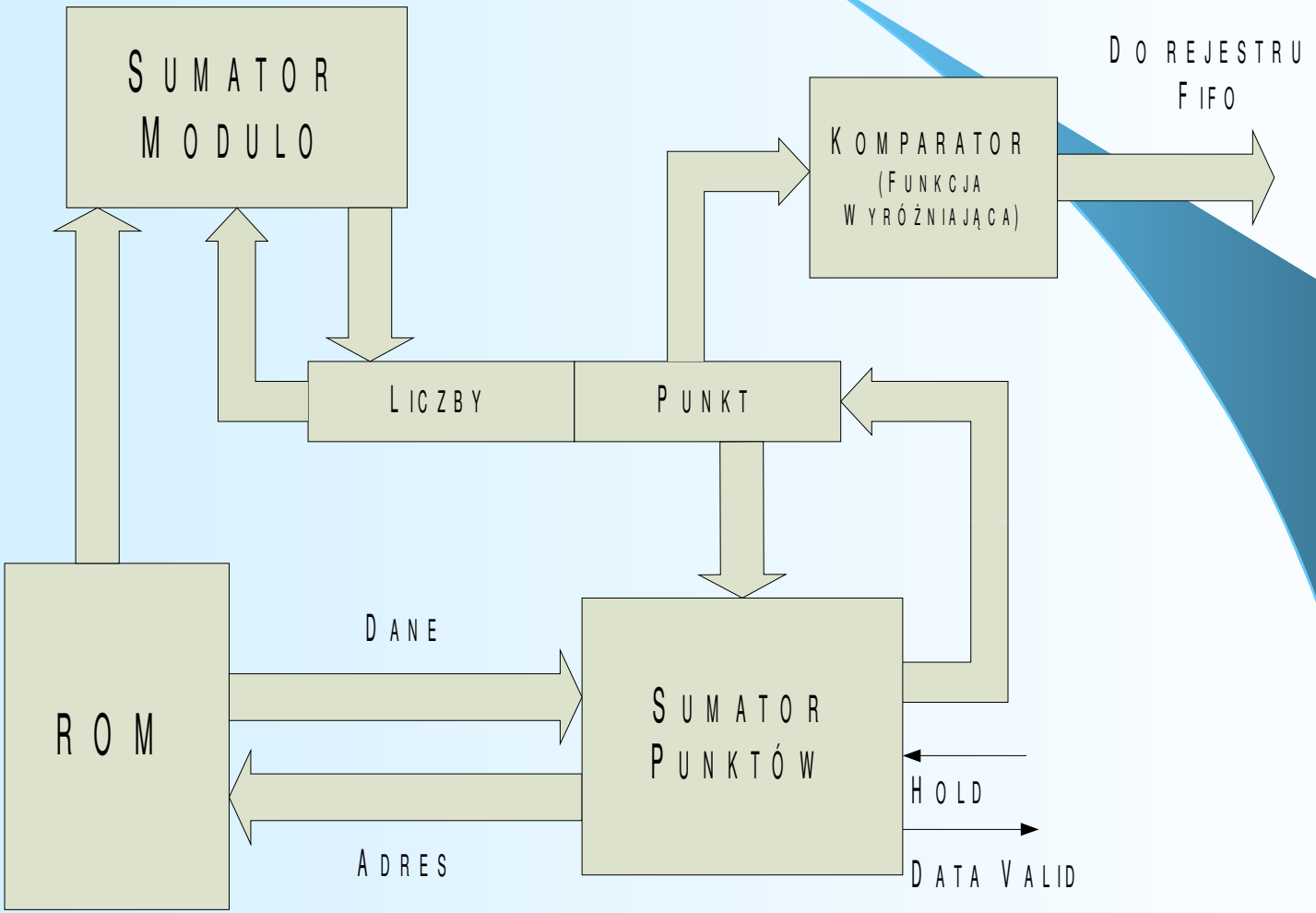
* $X = X + R_j$

* $c = c + a_j \pmod n$; $d = d + b_j \pmod n$

Biblioteka krzywych i rho

- Przeprowadzanie operacji dodawania na krzywych eliptycznych
- Równoległa wersja algorytmu rho Pollarda

Układ HardRho



Generator kodu VHDL

- Generacja „równań mnożenia” w oparciu o znalezioną macierz mnożenia
- Generacja sumatora punktów na krzywej
- Generacja pełnego układu HardRho

Plan pracy (1)

I. Wstęp

I.1. Cel pracy i wymagania

I.2. Opis pracy

Plan pracy (2)

II. Ciało skończone charakterystyki II
- II.1. Wstęp
- II.2. Bazy potęgowe
- II.3. Bazy normalne
- II.4. Istnienie i występowanie baz normalnych
- II.5. Mnożenie w bazach normalnych
- II.6. Konwersje pomiędzy bazami

Plan pracy (3)

III. Krzywe eliptyczne
- III.1. Wstęp
- III.2. Definicja krzywej eliptycznej
- III.3. Postać normalna krzywej
- III.4. Działania grupowe
- III.5. Reprezentacja punktów
- III.6. Obliczanie wielokrotności punktu
- III.7. Rząd krzywej
- III.8. Generacja silnych krzywych
- III.9. Zagadnienie *ECDLP*

Plan pracy (4)

IV. Algorytm rho Pollarda

IV.1. Główna idea algorytmu
IV.2. Wersja sekwencyjna
IV.3. Wersja równoległa

Plan pracy (5)

V. Biblioteka ciała skończonego

V.1. mathUtils
V.2. polyInGNB
V.3. gnb
V.4. curves
V.3. rho

Plan pracy (6)

V. Generator kodu VHDL
* Opis funkcji generujących
* Wyniki implementacji
* Zakres stosowalności rozwiązania

VI. Podsumowanie
* Plany na przyszłość

Plany na przyszłość

- Prace na zwielokrotnieniem
- Przeprowadzenie obliczeń
- Podłączenie układów FPGA do Systemu Rozproszonych Obliczeń

Dziękuję za uwagę