



Final Project PSD  
Kelompok PA01

# Dual-Lock Logic with Caesar and Hill Cipher



[https://github.com/nadizzati/FinproPSD\\_Kelompok1](https://github.com/nadizzati/FinproPSD_Kelompok1)

# Anggota Kelompok PA01



Eugenia Huwaida  
2406421384



Kayla Joanna  
2406487014



Nadia Izzati  
2406487033



Safina Amarani  
2406415665

# Background

## Mengapa ini Penting?

### Urgensi Keamanan Digital

Di era informasi, sistem enkripsi yang efisien dan efektif sangat dibutuhkan untuk melindungi data sensitif dari akses tidak berwenang.

### Tantangan Komputasi

Proses enkripsi dan dekripsi menuntut beban komputasi yang tinggi, terutama pada sistem hardware dengan sumber daya terbatas.

### Tujuan Proyek

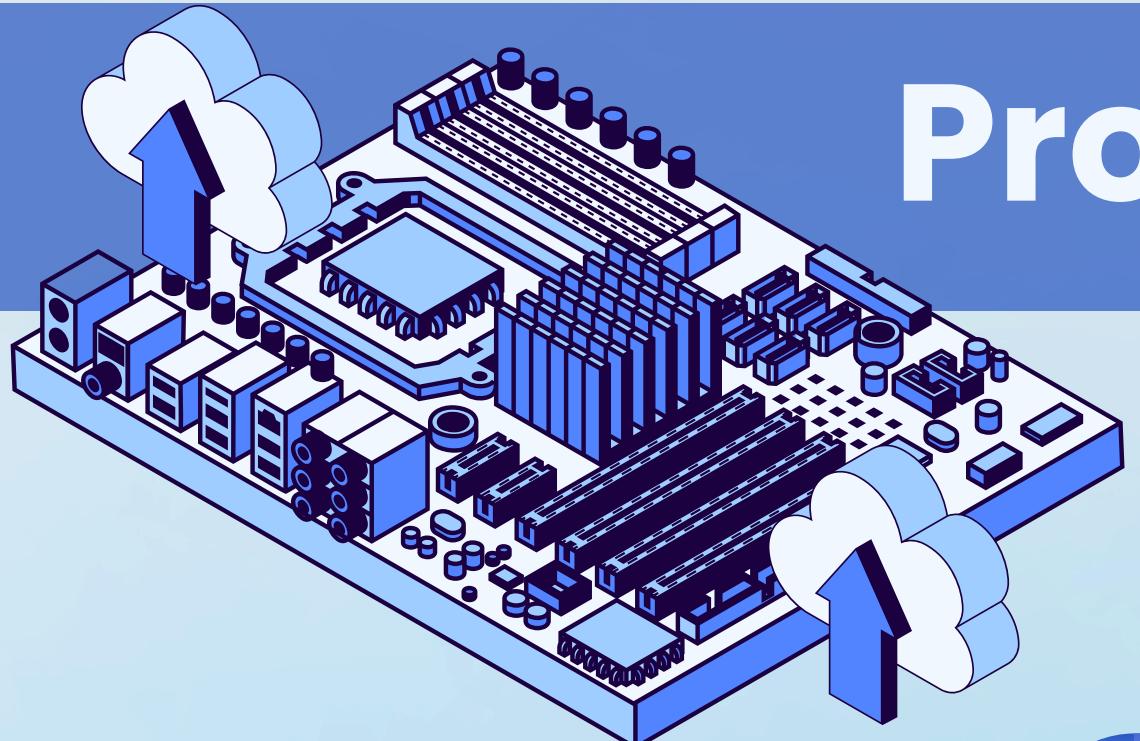
Menciptakan sistem yang mampu mengoptimalkan proses enkripsi, memastikan efisiensi tinggi, tetapi tetap mempertahankan tingkat keamanan yang kuat.

## Solusi

Solusi ini menggabungkan **Caesar Cipher** (untuk kemudahan dan kecepatan) dan **Hill Cipher** (untuk tingkat keamanan yang lebih tinggi melalui matriks kunci) guna mengamankan data. Implementasi kedua algoritma diakselerasi menggunakan **FPGA (Field-Programmable Gate Array)**. Pemanfaatan FPGA memungkinkan eksekusi algoritma secara paralel sehingga menghasilkan kecepatan dan efisiensi yang jauh lebih tinggi dibandingkan jika diterapkan menggunakan software biasa.



# Project Description



## Tujuan Utama

Mengembangkan arsitektur **keamanan data** berbasis algoritma hibrida yang mengintegrasikan metode **substitusi Caesar Cipher dan transformasi linear Hill Cipher** untuk menciptakan sistem pengamanan berlapis.

## Caesar Cipher

Proses pengamanan dimulai dengan Caesar Cipher sebagai lapisan awal yang melakukan **shifting** sesuai parameter input.

## Hill Cipher

Output dari caesar cipher kemudian diproses lebih lanjut menggunakan **operasi matrix-key** pada Hill Cipher untuk meningkatkan kompleksitas.

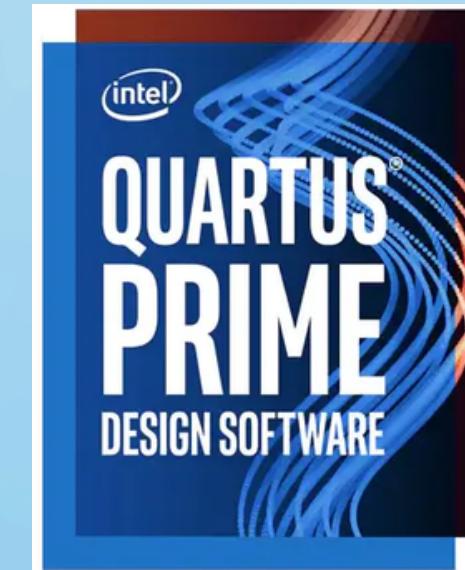
## Decryption

Proses dekripsi menerapkan **prinsip reverse sequence**, di mana pemrosesan diawali dengan dekripsi Hill Cipher dan diakhiri oleh dekripsi Caesar Cipher. Metode ini **menjamin integritas data** saat mengembalikan cipher text menjadi plain text original.

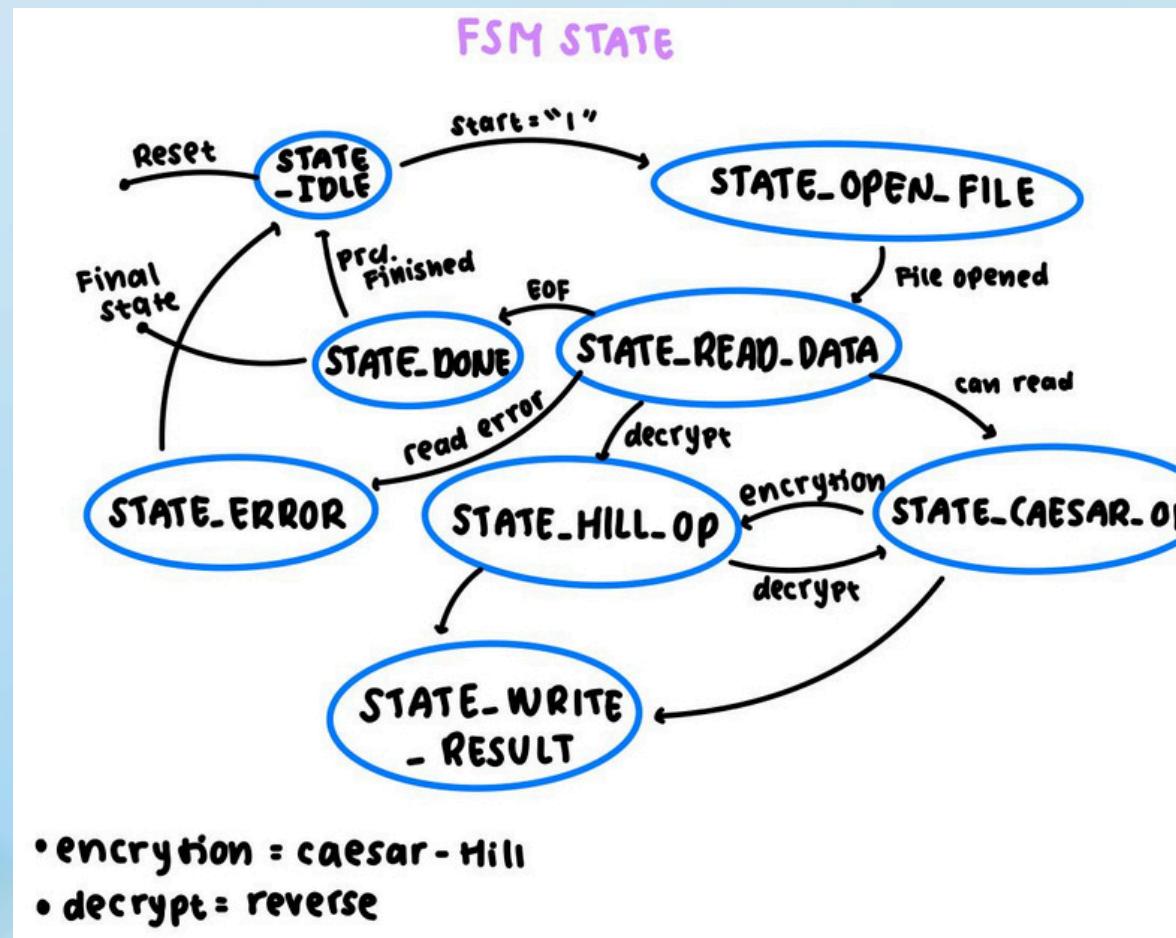
# Objective

- Pemenuhan syarat nilai Praktikum Perancangan Sistem Digital
- Mendesain serta membangun sistem enkripsi–dekripsi ganda dengan metode Caesar Cipher dan Hill Cipher
- Mengoptimalkan keamanan data melalui penggunaan perangkat keras berbasis FPGA

# Tools



# FSM DIAGRAM



Sistem ini memanfaatkan FSM untuk mengendalikan proses enkripsi dan dekripsi menggunakan Caesar Cipher dan Hill Cipher. Operasi dilakukan pada sebuah file input dan menghasilkan file output sesuai mode yang dipilih. Eksekusi dimulai pada STATE\_IDLE, di mana sistem menunggu sinyal start. Mode operasi kemudian menentukan apakah sistem menjalankan enkripsi (mode = '0') atau dekripsi (mode = '1').

Setelah mode ditetapkan, FSM berpindah ke STATE\_OPEN\_FILE untuk membuka file input dan menyiapkan file output. Jika proses pembukaan file berhasil, sistem melanjutkan ke STATE\_READ\_DATA untuk membaca setiap karakter secara bertahap.

Pada proses enkripsi, data melewati STATE\_CAESAR\_OP terlebih dahulu, kemudian STATE\_HILL\_OP, sebelum akhirnya ditulis ke file output melalui STATE\_WRITE\_RESULT. Sebaliknya, pada proses dekripsi, urutan operasi dimulai dari STATE\_HILL\_OP, dilanjutkan dengan STATE\_CAESAR\_OP, lalu hasilnya langsung ditulis ke output.

Ketika seluruh karakter telah diproses, sistem memasuki STATE\_DONE untuk menutup file dengan aman dan mengaktifkan sinyal done. Jika terjadi kegagalan pada salah satu tahap, FSM akan beralih ke STATE\_ERROR, mengaktifkan sinyal error, lalu kembali ke STATE\_IDLE. Dengan pendekatan ini, FSM memastikan alur kerja yang terstruktur, konsisten, dan modular.

# Caesar Cipher

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity caesarCipher is
    port (
        char_input      : in  std_logic_vector(7 downto 0);
        enc_dec_mode   : in  std_logic; -- '0' encrypt, '1' decrypt
        shift_amount    : in  integer range 0 to 25;
        char_output     : out std_logic_vector(7 downto 0)
    );
end entity caesarCipher;

architecture Behavioral of caesarCipher is
begin
    process(char_input, enc_dec_mode, shift_amount)
        variable ascii_value : integer range 0 to 255;
        variable shift_value : integer range -25 to 25;
    begin
        -- input -> integer
        ascii_value := to_integer(unsigned(char_input));

        -- determine shift direction based on mode
        if enc_dec_mode = '0' then -- encrypt
            shift_value := shift_amount;
        else -- decrypt
            shift_value := -shift_amount;
        end if;

        -- uppercase letters (A-Z)
        if (ascii_value >= 65 and ascii_value <= 90) then
            ascii_value := ((ascii_value - 65 + shift_value + 26) mod 26) + 65;

        -- lowercase letters (a-z)
        elsif (ascii_value >= 97 and ascii_value <= 122) then
            ascii_value := ((ascii_value - 97 + shift_value + 26) mod 26) + 97;
        end if;

        char_output <= std_logic_vector(to_unsigned(ascii_value, 8));
    end process;
end architecture Behavioral;
```

Sistem ini menggunakan tipe state fsm\_state untuk mengatur alur kerja, dan pada STATE\_CAESAR\_OP modul caesarCipher dijalankan untuk melakukan enkripsi atau dekripsi karakter ASCII. Untuk menerima input 8-bit, mode operasi, dan nilai pergeseran, lalu mengonversi karakter ke integer untuk menentukan arah pergeseran. Jika karakter berada dalam rentang huruf besar atau kecil, pergeseran dihitung menggunakan operasi modular agar tetap berada dalam alfabet. Setelah itu, hasil dikonversi kembali menjadi std\_logic\_vector dan dikirim sebagai output sebelum FSM melanjutkan ke tahap pemrosesan berikutnya.

# Hill Cipher



## Entity Specification

Entity hillCipher dirancang untuk mengeksekusi algoritma kriptografi berbasis matriks 2x2. Interface sistem menerima input dan menghasilkan output bertipe std\_logic\_vector 8-bit, yang merepresentasikan standar pengkodean karakter ASCII.

## Behavioral Architecture

Struktur internal mendefinisikan matrix key dan determinan invers sebagai konstanta tetap. Operasi aritmatika dikendalikan oleh fungsi mod26 untuk memastikan hasil komputasi matrix tetap berada dalam rentang index alfabet (0–25).

## Transformasi Data

Konversi tipe data input menjadi integer untuk kalkulasi. Dari signal, sistem akan melakukan perkalian matrix key untuk enkripsi atau perkalian determinan invers untuk dekripsi sebelum mengonversi hasil kembali ke format vektor.

## Format Karakter

Sistem menerapkan logika penanganan offset untuk mempertahankan konsistensi format huruf besar dan huruf kecil. Selain itu, karakter non-alfabet akan diteruskan secara transparan tanpa mengalami proses transformasi.

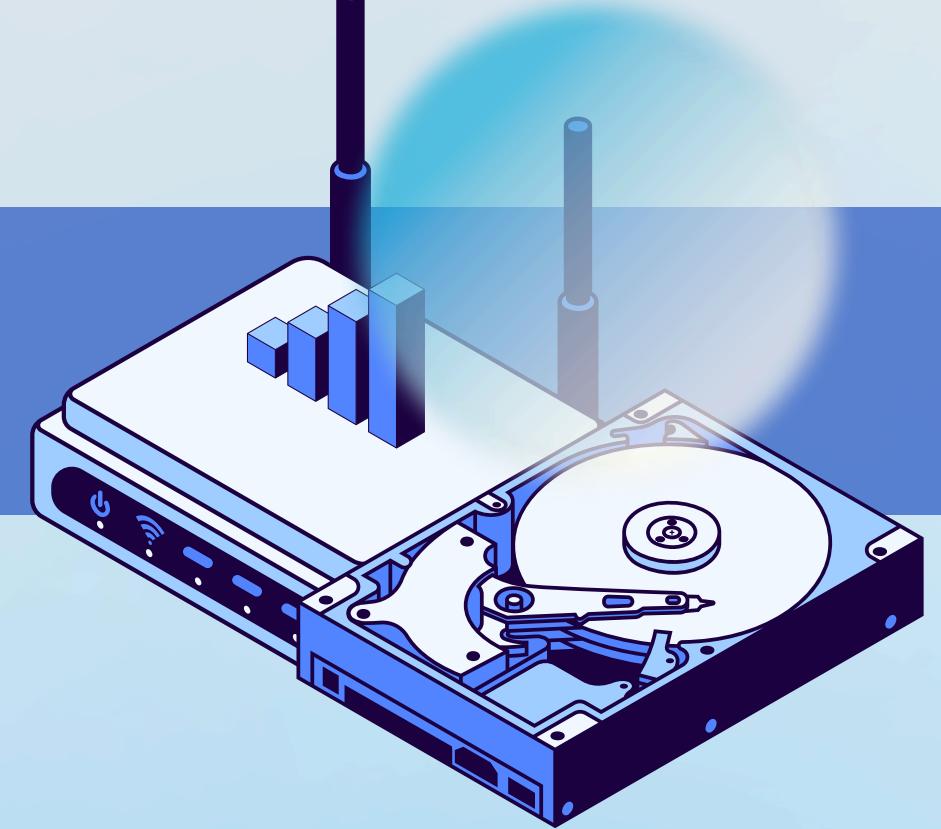
# Hill Cipher (Snippet Code)

```
entity hillCipher is
  Port (
    data_in : in STD_LOGIC_VECTOR(7 downto 0);
    operation_mode : in STD_LOGIC;
    data_out : out STD_LOGIC_VECTOR(7 downto 0)
  );
end hillCipher;

architecture Behavioral of hillCipher is
  constant ENCRYPT_KEY : integer := 5;
  constant DECRYPT_KEY : integer := 21;

  function mod26(val : integer) return integer is
    variable result : integer;

begin
  result := val mod 26;
  while result < 0 loop
    result := result + 26;
  end loop;
  return result;
end function;
```



# Testbench

## Input & Enkripsi

File input.txt digunakan sebagai masukan yang berisi teks uji coba "hello". Proses enkripsi dimulai dengan algoritma Caesar Cipher yang bekerja pada teks input. Hasil output dari tahap pertama ini (setelah diproses Caesar Cipher) kemudian disimpan sebagai file sementara dengan nama phase1.txt.

## Behavioral Architecture

Setelah hasil sementara didapatkan, proses enkripsi dilanjutkan dengan Hill Cipher, yang mengambil input dari phase1.txt. Hasil akhir dari keseluruhan proses enkripsi ini, yang merupakan output dari Hill Cipher, kemudian disimpan ke dalam file final yang bernama encryptOutput.txt.

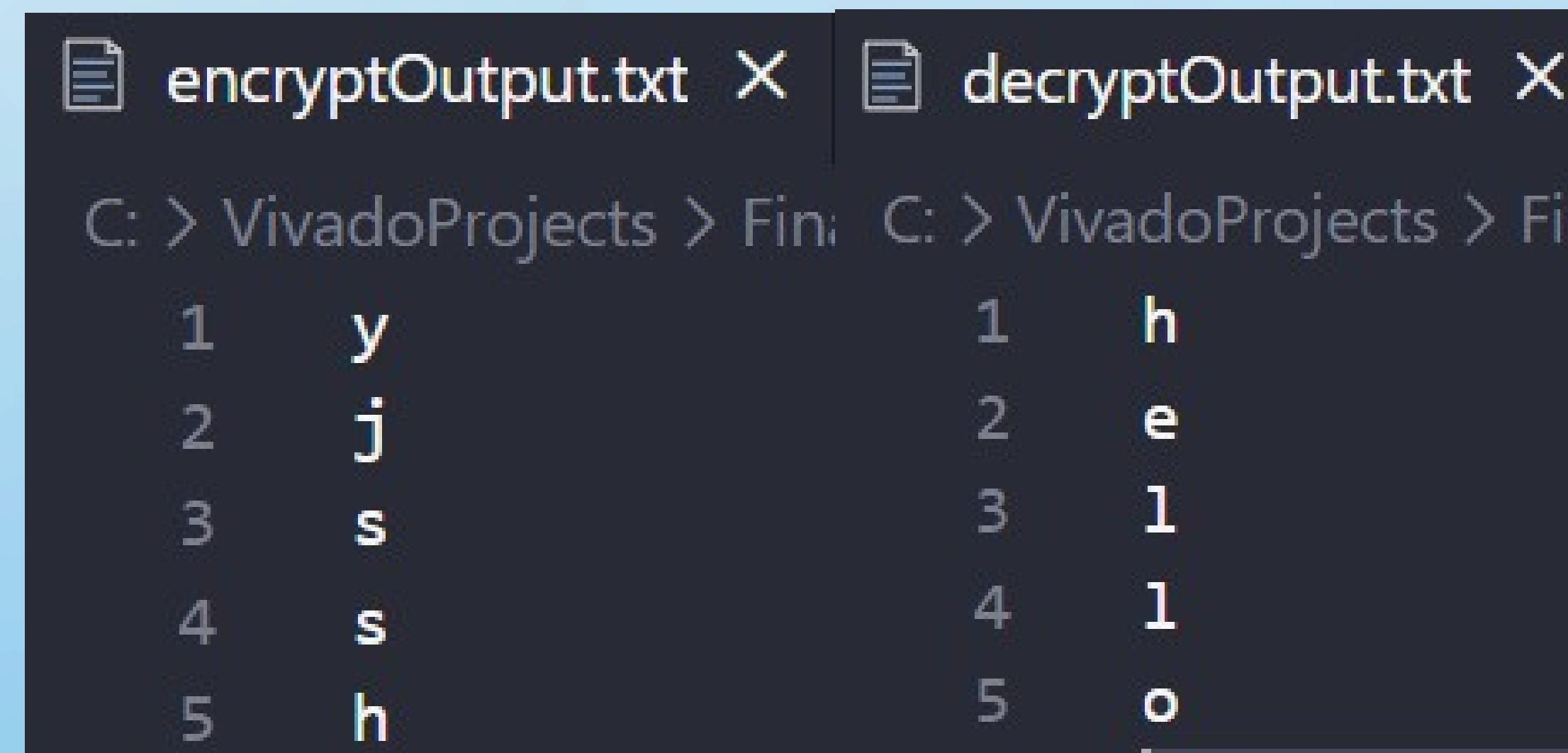
## Output & Validasi

Untuk memverifikasi kebenaran sistem, file encryptOutput.txt diproses ulang melalui dekripsi. Proses ini harus dilakukan secara terbalik: dimulai dengan dekripsi Hill Cipher, dilanjutkan dengan dekripsi Caesar Cipher. Hasil akhir dari proses dekripsi ini kemudian disimpan ke dalam file decryptOutput.txt.

## Proses Dekripsi & Verifikasi

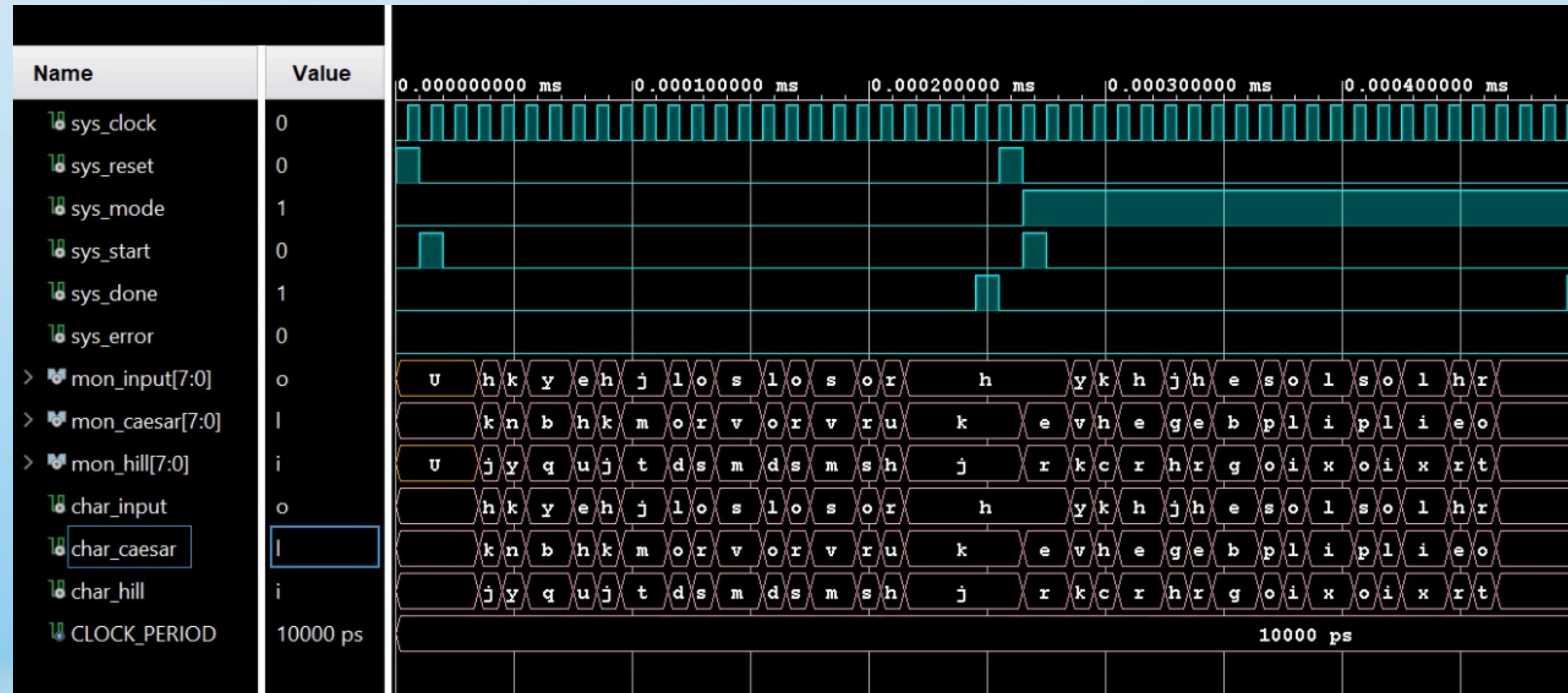
Langkah validasi akhir adalah memastikan isi file decryptOutput.txt (hasil dekripsi) sepenuhnya identik dengan isi file input.txt (teks "hello" awal). Kesesuaian ini membuktikan bahwa seluruh sistem enkripsi dan dekripsi telah bekerja dengan benar, memproses data tanpa adanya error atau kehilangan data.

# File Output

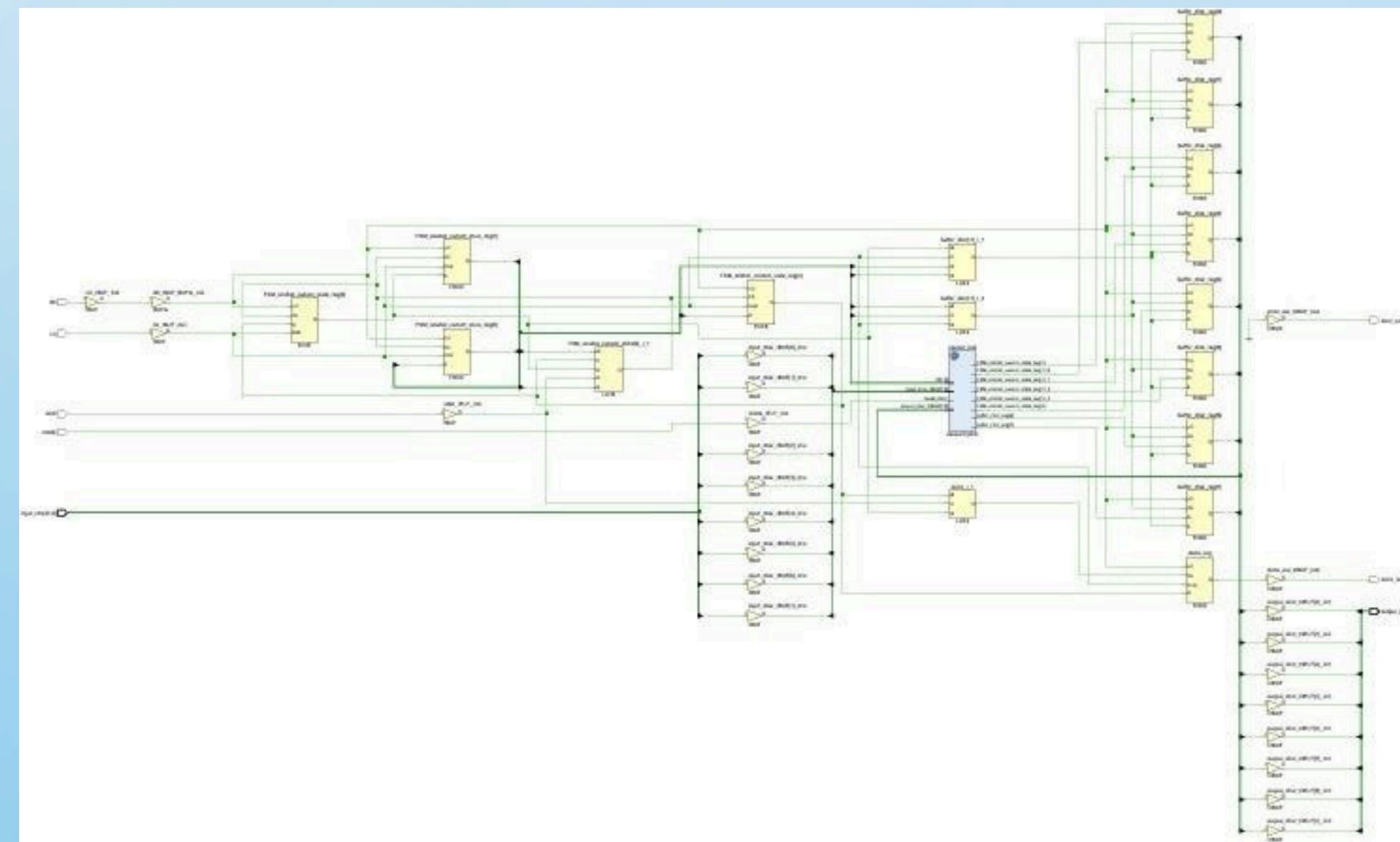


```
encryptOutput.txt X decryptOutput.txt X
C: > VivadoProjects > Fin: C: > VivadoProjects > Fin
1 y 1 h
2 j 2 e
3 s 3 l
4 s 4 l
5 h 5 o
```

# Wave



# Synthesized Result



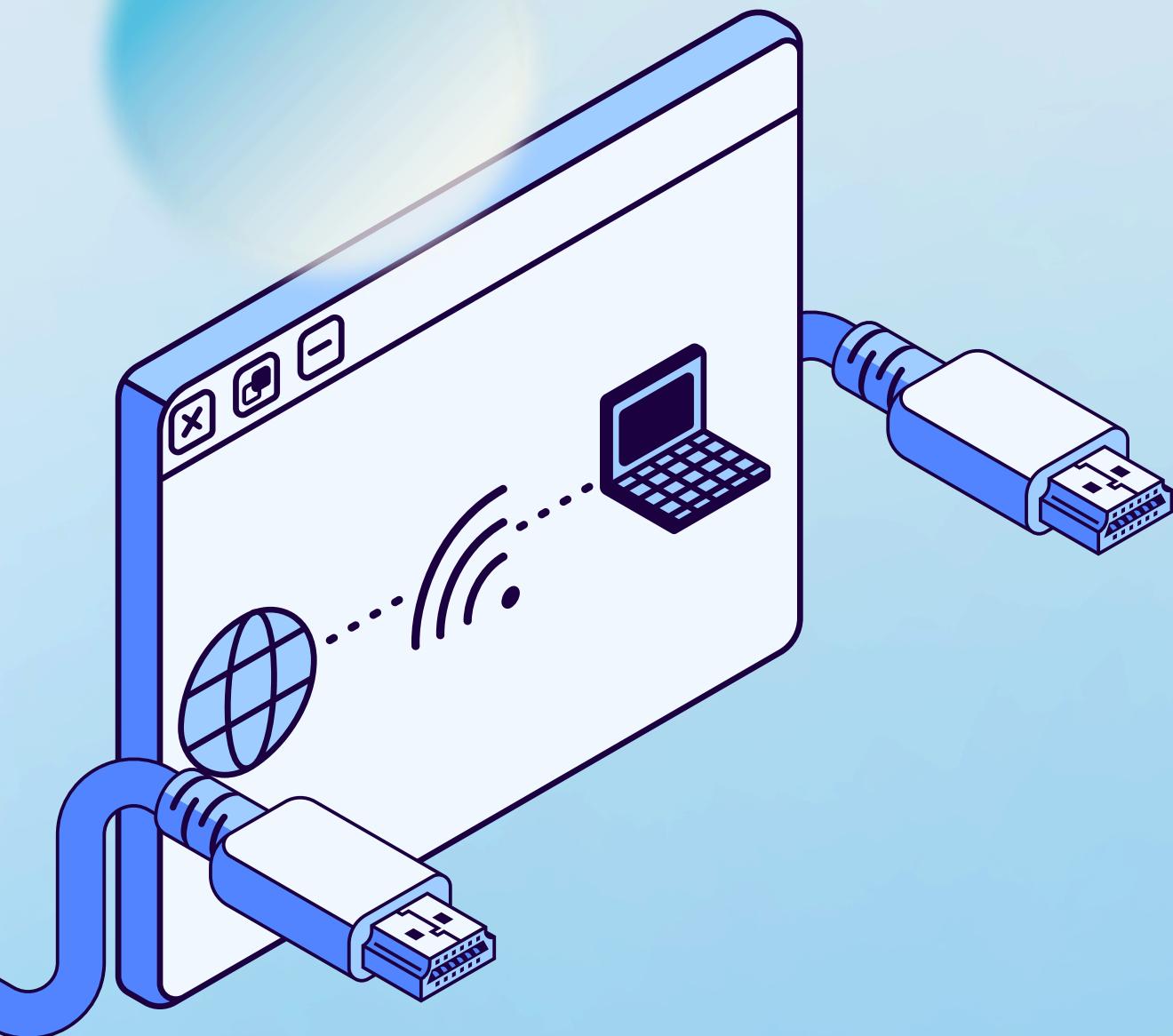
# Tcl Console

```
Note: Reading Encrypted Output
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output Char: y
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output Char: j
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output Char: s
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output Char: s
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output Char: h
Time: 255 ns Iteration: 0 Process: /MainTB/test_proc
Note: Output matches expected data|
```

# Tcl Console

```
Note: DECRYPTION TEST SCENARIO
Time: 265 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Input written to file: yjssh
Time: 265 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Reading Decrypted Output
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output Char: h
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output Char: e
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output Char: l
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output Char: l
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output Char: o
Time: 505 ns  Iteration: 0  Process: /MainTB/test_proc
Note: Output matches expected data
```

# Kesimpulan



- Sistem enkripsi dan dekripsi ganda menggunakan Caesar Cipher dan Hill Cipher berhasil diimplementasikan pada FPGA.
- Mekanisme enkripsi dua tahap (Caesar lalu Hill) dan dekripsi terbalik diterapkan dengan benar.
- Proses enkripsi bekerja sesuai harapan dan menghasilkan output yang benar.
- Alur Finite State Machine (FSM) dan mekanisme proses dua tahap telah diterapkan dengan benar.
- Diyakini bahwa sebagian besar kode dan fungsionalitas dasar sistem telah bekerja dengan baik dan stabil.
- Sistem menunjukkan potensi besar dalam menerapkan enkripsi yang lebih kuat melalui dua tahap pengolahan data, menawarkan lapisan keamanan tambahan.



Kelompok PA01

# Thank You!

