Mục lục

[I. Mô hình khôi phục mật khẩu VeraCrypt 2](#_Toc167571660)

[II. Các thuật toán sử dụng trong quá trình khôi phục 3](#_Toc167571661)

[2.1. Password Generator: 3](#_Toc167571662)

[2.2. Key Derivation Function: 3](#_Toc167571663)

[2.2.1. Whirlpool 3](#_Toc167571664)

[2.2.2. PBKDF2-HMAC-Whirlpool 6](#_Toc167571665)

[2.3. Giải mã khối mã hóa Serpent 7](#_Toc167571666)

[2.3.1. Thuật toán mã hóa khối Serpent 7](#_Toc167571667)

[2.3.2. Serpent ở chế độ XTS 10](#_Toc167571668)

[III. Thiết kế module 10](#_Toc167571669)

[3.1. Password Generator 10](#_Toc167571670)

[3.2. PBKDF2-HMAC-Whirlpool 10](#_Toc167571671)

[3.2.1. Module TOP\_WHIRLPOOL\_HASH 10](#_Toc167571672)

[3.1.2. Module WHIRLPOOL\_HASH 13](#_Toc167571673)

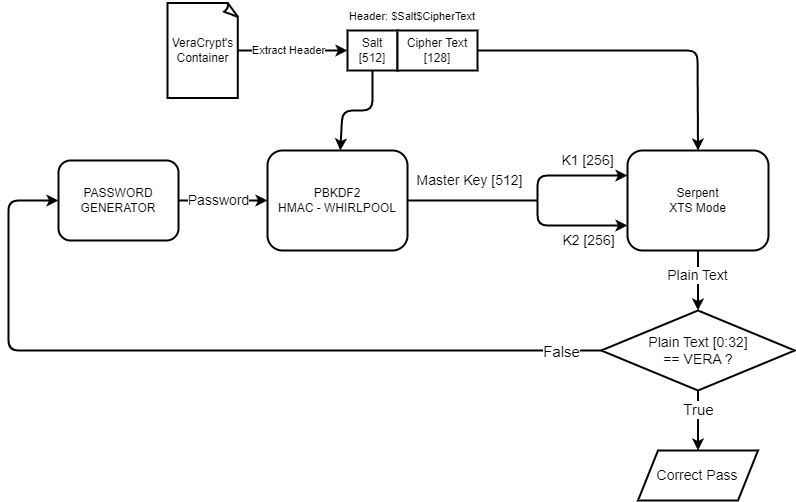
[3.1.3. Module WHIRLPOOL\_KEY\_EXPANSION 15](#_Toc167571674)

[3.3. XTS Serpent 18](#_Toc167571675)

[3.3.1. Module XTS 18](#_Toc167571676)

[3.3.2. Module Serpent 18](#_Toc167571677)

# I. Mô hình khôi phục mật khẩu VeraCrypt



Hình . Mô hình khôi phục mật khẩu

VeraCrypt là một phần mềm mã hóa mã nguồn mở, được thiết kế để bảo vệ dữ liệu bằng cách mã hóa toàn bộ ổ đĩa hoặc tạo các tệp chứa mã hóa. VeraCrypt là sự kế thừa của TrueCrypt, bổ sung thêm các tính năng và cải tiến về bảo mật. Các đặc điểm nổi bật của VeraCrypt bao gồm:

**+ Hỗ trợ thuật toán:** Hỗ trợ nhiều thuật toán mã hóa như AES, Serpent, Twofish,…

**+ Tính năng mã hóa ổ đĩa toàn phần:** Bảo vệ dữ liệu trên toàn bộ ổ đĩa hoặc phân vùng, bao gồm cả ổ đĩa hệ thống.

**+ Tạo các Container mã hóa:** Tạo các Container có thể được gắn như các ổ đĩa ảo.

**- Quy trình tạo Container của VeraCrypt:**

+ Nhận chuỗi mật khẩu mà người dùng nhập vào, đồng thời thu thập salt trên di chuyển của con trỏ chuột ( ở phần mềm ). Mật khẩu và salt sau đó được đưa vào hàm băm PBKDF2-HMAC – < Thuật toán băm > và cho ra Khóa chính. Thuật toán băm ở đây bao gồm các thuật toán SHA256, SHA512, Whirlpool,…

+ Sau khi có khóa chính, VeraCrypt sử dụng nó để tiến hành mã hóa dựa trên các thuật toán AES, Serpent, Twofish,… ở chế độ XTS, từ đó sinh ra Container (.vc file).

**- Quy trình khôi phục:** Với mật khẩu được sinh ra từ Password Gen, ta sẽ thử các mật khẩu bằng cách áp dụng mật khẩu đó lên header của Container theo quy trình: PBKDF2 – Decrypt. Nếu 4bytes đầu của chuỗi dữ liệu ta đã giải mã có kí tự “VERA”, thì có thể password đó là password chính xác

# II. Các thuật toán sử dụng trong quá trình khôi phục

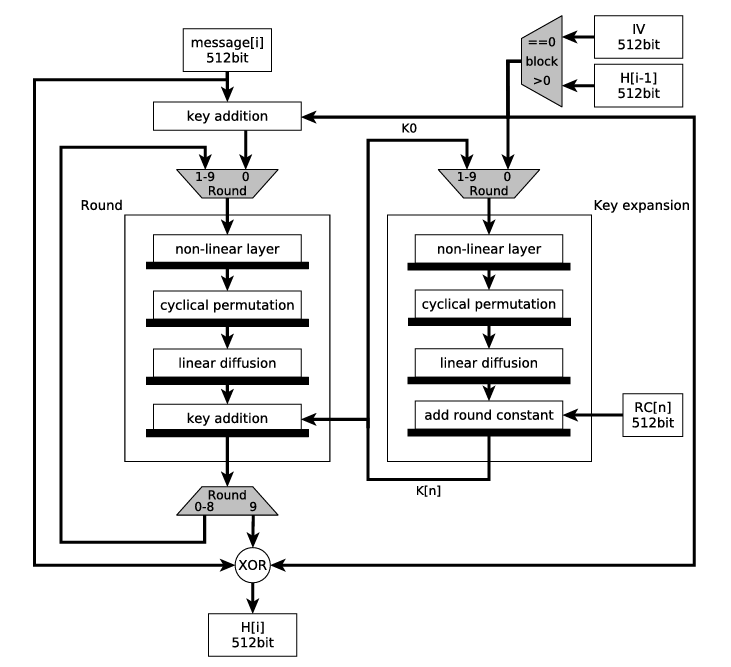
## 2.1. Password Generator:

## 2.2. Key Derivation Function:

### 2.2.1. Whirlpool

#### 2.2.1.1. Hàm băm Whirlpool

Hàm băm Whirlpool được phát triển bởi Vincent Rijment, là nhà đồng phát triển của thuật toán mã hóa khối Rijndael. Hàm băm Whirlpool dựa trên thuật toán mã hóa khối, được sử dụng trong xác thực, toàn vẹn dữ liệu. Tuy nhiên ở trường hợp này, hàm băm sẽ được dùng để tạo ra Master Key cho quá trình giải mã.

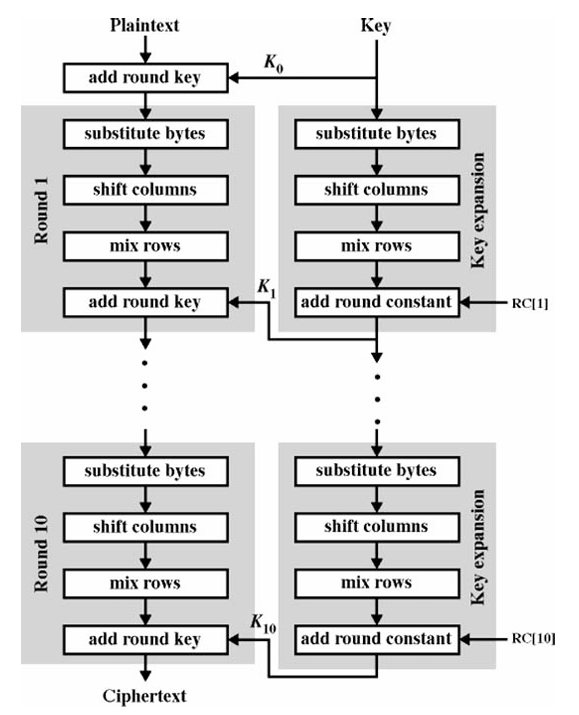


Hình . Thuật toán băm Whirlpool

Dữ liệu đầu vào của hàm băm được cố định là 512 bit, nếu dữ liệu đầu vào chưa đủ độ dài thì sẽ được padding lên sao cho đủ 512 bit. Kết quả của hàm băm cho một mã băm với độ dài 512 bit. Whirlpool bao gồm 10 vòng với 2 khối, 1 khối Key Expansion để tạo key cho từng round và 1 khối băm dữ liệu đầu vào. Ở vòng cuối (9) dữ liệu sẽ được XOR với dữ liệu ở vòng 8 và dữ liệu gốc để cho ra kết quả cuối cùng.

#### 2.2.1.2. Mô tả các bước trong thuật toán

Whirlpool bao gồm 10 vòng, với mỗi vòng gồm các bước **Add Round Key**, **Substitute Bytes**, **Shift Columns, Mix Rows.** Dữ liệu đầu vào 512 bit sẽ được chuyển thành một ma trận 8x8 bytes



Hình . Các bước xử lí trong 1 vòng

**AddRoundKey:** Được thực hiện ở bước khởi tạo đầu tiên, và ở đầu ra của các vòng.   
Dữ liệu được trộn với khóa vòng theo

**SubstituteBytes:** Chức năng SubstituteBytes là thực hiện thay thế từng byte của ma trận trạng thái bằng một giá trị đã quy định trước. Bảng quy định giá trị thay thế gọi là S-box.

**ShiftColumns:** Ở giai đoạn này, các bytes trong ma trận sẽ được dịch xuống khỏi ma trận trạng thái. Ex: phần tử ở cột j sẽ bị dịch xuống j lần

**MixRows:** Ma trận dữ liệu sau khi ShiftColumns sẽ được tiến hành nhân với một ma trận C cho trước

Ở khối Key Expansion, quá trình trên cũng diễn ra ứng với từng vòng, tuy nhiên giai đoạn AddRoundKey sẽ được thay thế thành AddRoundConstant với RC là một ma trận được định nghĩa như sau:

A close-up of numbers

Description automatically generated

**Bảng Sbox và Ma trận C**

A table of numbers and letters

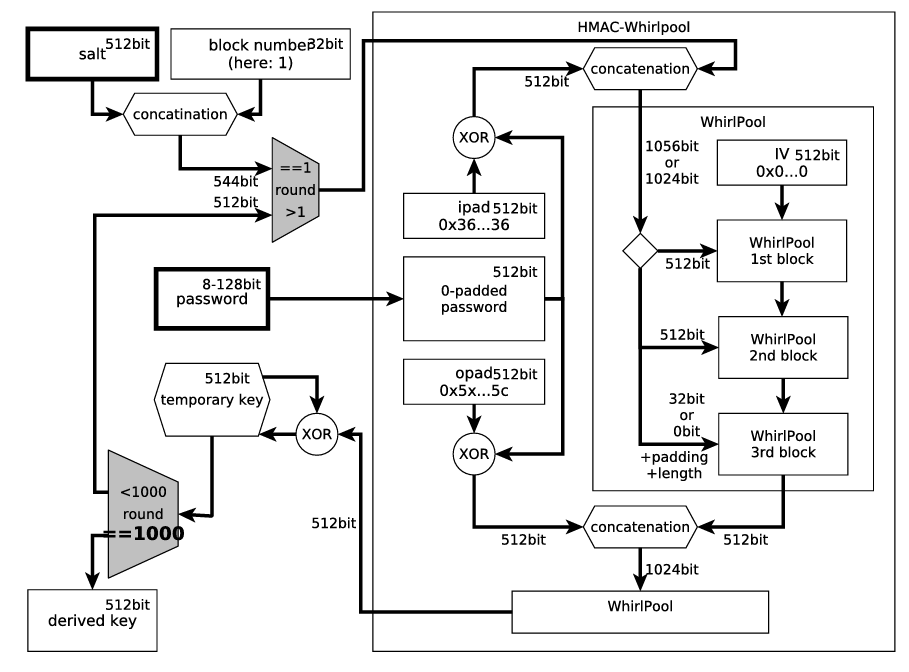
Description automatically generated

**A number of numbers on a white background

Description automatically generated**

### 2.2.2. PBKDF2-HMAC-Whirlpool

#### 2.2.2.1. PBKDF2 sử dụng HMAC-Whirlpool



*Hình . PBKDF2-HMAC-Whirlpool*

Ta sử dụng PBKDF2 để biến mật khẩu người dùng nhập vào trở thành một khóa mã hóa mạnh, không dự đoán được để ngăn chặn các cuộc tấn công Brute-force với:

+ K: Khóa chính mà ta sẽ sử dụng trong quá trình giải mã Cipher Text

+ Pwd: Password được sinh ra từ Module Pass Gen

+ C: Số lần lặp lại của quá trình

+ Klen: Độ dài K mong muốn, ở đây ta sẽ set bằng 512 bit

Khi đi kèm với HMAC, PBKDF2 sẽ sử dụng HMAC như một hàm sinh số ngẫu nhiêu, mật khẩu sẽ được PBKDF2 áp dụng HMAC nhiều lần.

## 2.3. Giải mã khối mã hóa Serpent

### 2.3.1. Thuật toán mã hóa khối Serpent

Serpent là một thuật toán mã hóa đối xứng khối, được sử dụng trong các ứng dụng mật mã và bảo mật

#### 2.3.1.1. Tổng quan thuật toán Serpent

A diagram of a program

Description automatically generated

Hình . Sơ đồ khối thuật toán của Serpent

Thuật toán nhận dữ liệu đầu vào (P) có kích thước là 128 bit với key (K) mã hóa với các kích thước 128, 192, 256 bit, ở đây ta sẽ tập trung vào Key 256 bit.

#### 2.3.1.2. Quá trình mã hóa của thuật toán Serpent

Dữ liệu đầu vào sẽ được trải qua vòng Hoán vị đầu (IP) để phân tác các bit dữ liệu đầu vào giúp cho các bước sau của thuật toán mang đến kết quả tốt hơn. Đầu ra của hoán vị đầu được kí hiệu là B0

A table of numbers with a white background

Description automatically generated

Bảng . Bảng index của bước Hoán vị đầu.

Ta có: với . Ví dụ, với thì

, từ đó ta có đầu ra B0 128 bit làm dữ liệu đầu vào cho 32 rounds tiếp theo.

Ở 32 vòng tiếp theo, Serpent sẽ thực hiện các bước:

- Mix Round Key: , với Kr được sinh ra từ khóa đầu vào K

- Sboxes: *B = Sbox(B),* dữ liệu 128 bit đầu vào B sẽ được tách ra thành 4 word 32 bit được đánh số X0, X1, X2, X3. Sau đó các bit đầu tiên của các word được ghép thành một dãy 4 bit, làm đầu vào cho khối Sbox. Tương tự với các bit còn lại của word, từ đó ta sẽ có 32 dãy 4 bit làm đầu vào cho 32 khối Sbox được đặt song song với nhau. Ứng với 32 vòng ta sẽ có 8 loại Sbox với giá trị khác nhau được xác định bởi

A diagram of a word

Description automatically generated

Hình . Quá trình Sboxes

- Biến đổi tuyến tính: Dữ liệu đầu ra của khối Sbox X0’, X1’, X2’, X3’ tiếp tục sẽ làm đầu vào cho bước biến đổi tuyến tính được miêu tả sau đây:

A diagram of a circuit

Description automatically generated

Hình . Quá trình biến đổi tuyến tính.

Quá trình trên được lặp lại 31 vòng, cho đến vòng thứ 32, đầu ra của khối Sbox sẽ không trải qua bước biến đổi tuyến tính mà sẽ Mix với Round Key 32.

Và trải qua bước hoán vị cuối để được dữ liệu đã mã hóa (C).

#### 2.3.1.3. Quá trình giải mã của thuật toán Serpent

Ngược lại với 1 giá trị Sboxes và quy tắc biến đổi LT khác, RoundKey cũng sẽ được áp dụng ở chiều ngược lại từ 32 về 1 và được biến đổi thông qua Sboxes inverse

#### 2.3.1.4. Key Schedule trong Serpent

Quá trình tạo Round Key để mã hóa từng vòng của thuật toán Serpent được diễn ra như sau.

A diagram of a computer

Description automatically generated

Hình . Key Schedule

Key đầu vào 256 bit được tách ra thành 8 word con với mỗi word 32 bit được đánh số từ . Chúng ta sau đó tạo ra 132 word còn lại theo công thức:

Với . Từ đó ta có 132 word được sinh ra, ta tiến hành đưa 4 word w(4i, 4i + 1, 4i + 2, 4i + 3) vào Sboxes, từ đó sinh ra được Subkey Ki dành cho các vòng.

#### 2.3.1.5. Các bảng giá trị

A table of numbers with a few lines

Description automatically generated with medium confidence

A grid of numbers with black text

Description automatically generated

### 2.3.2. Serpent ở chế độ XTS

XTS (XEX-based Tweaked CodeBook mode with CipherText Stealing) là một chế độ mã hóa khối, thường được sử dụng chủ yếu để mã hóa đĩa cứng và thiết bị lưu trữ.

XTS yêu cầu 2 khóa K1, K2 được tách ra từ Master Key K để có thể vận hành. Với khóa chính K1 dùng để mã hóa và giải mã các khối dữ liệu, dùng K2 để tạo giá trị Tweak. Khi tiến hành XOR các đầu ra/vào dữ liệu với Tweak sẽ tăng khả năng bảo mật và độc nhất của khối dữ liệu được mã hóa.

A diagram of a block diagram

Description automatically generated

Hình . Serpent ở XTS mode.

# III. Thiết kế module

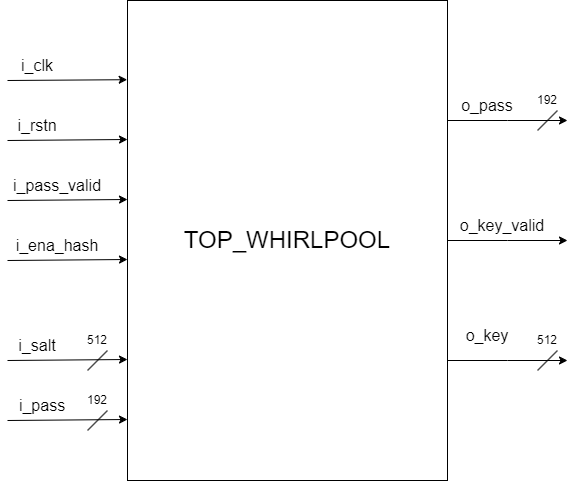
## 3.1. Password Generator

## 3.2. PBKDF2-HMAC-Whirlpool

### 3.2.1. Module TOP\_WHIRLPOOL\_HASH

#### 3.2.1.1. Mô tả In/Out

Module TOP\_WHIRLPOOL đóng vai trò như 1 lớp PBKDF2-HMAC bọc module băm Whirlpool bên trong, có tác dụng padding mật khẩu đầu vào, điều khiển số lần áp dụng hàm băm Whirlpool. Đầu vào là mật khẩu được sinh ra từ module Password Generator với 192 bit. Module sẽ cho ra chuỗi hash với mật khẩu tương ứng, chuỗi hash này sẽ được sử dụng làm khóa chính cho quá trình giải mã XTS Serpent.

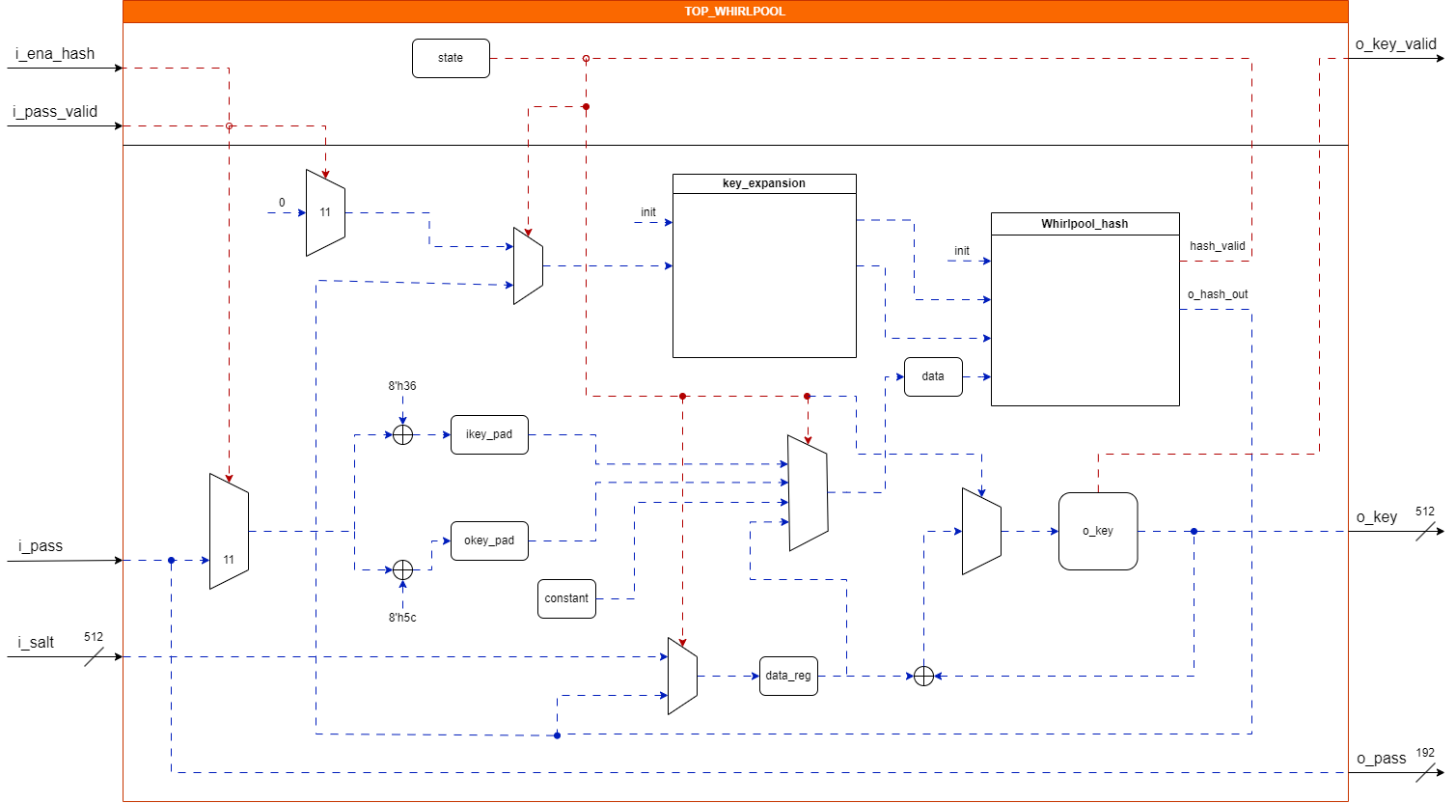


Hình . IO Module TOP\_Whirlpool

|  |  |  |  |
| --- | --- | --- | --- |
| Tín hiệu | Loại | Kích thước (bit) | Mô tả |
| i\_clk | Input | 1 | Xung clock sườn dương |
| i\_rstn | Input | 1 | Tín hiệu reset sườn âm |
| i\_pass\_valid | Input | 1 | Tín hiệu thông báo từ module PassGen để báo răng password valid |
| i\_ena\_hash | Input | 1 | Tín hiệu cho phép băm từ module TOP |
| i\_salt | Input | 512 | Dữ liệu muối được extract từ header của VeraCrypt’s Volume |
| i\_pass | Input | 192 | Mật khẩu từ PassGen |
| o\_pass | Output | 192 | Mật khẩu từ PassGen được chuyển tiếp ra khỏi module |
| o\_key\_valid | Output | 1 | Tín hiệu thông báo key đã hợp lệ |
| o\_key | Output | 512 | Master Key cho XTS Serpent |

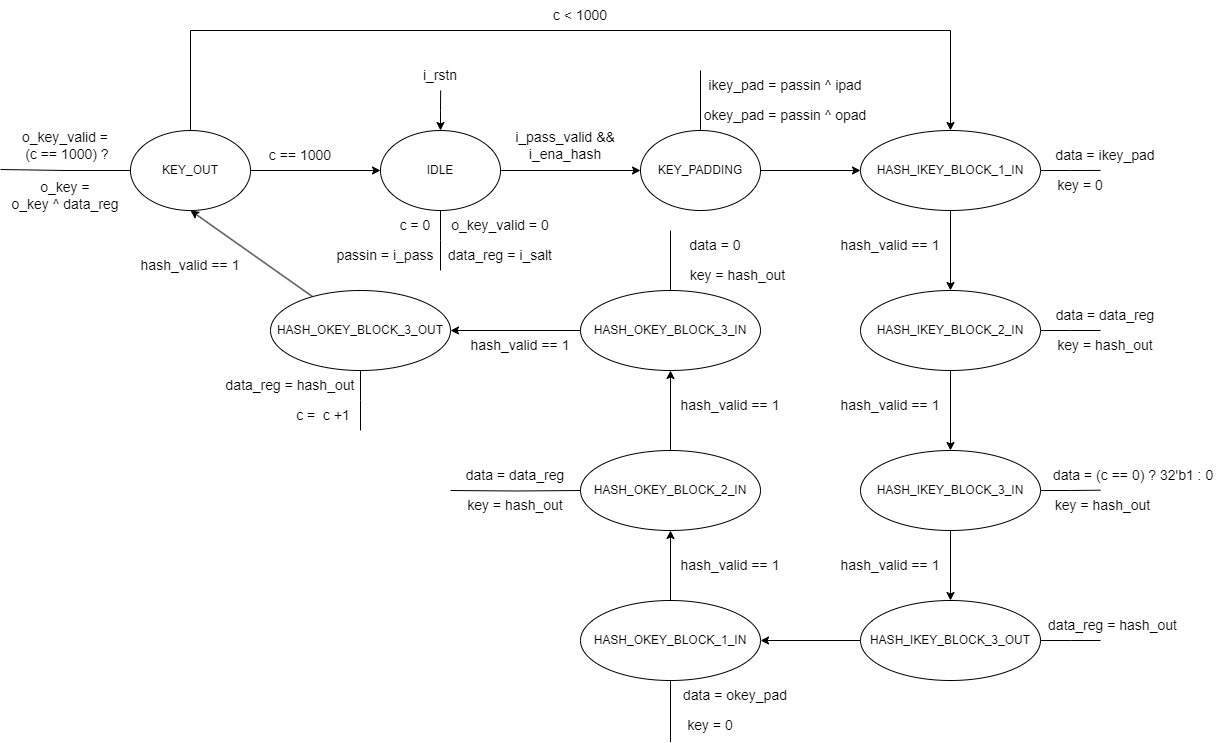
Bảng . I/O của khối module TOP\_Whirlpool

#### 3.2.1.2. Control Path & Data Path



Hình . Control Path & DataPath của module TOP\_Whirlpool

#### 3.2.1.3. FSM



Hình . Máy trạng thái của module TOP\_WHIRLPOOL\_HASH

**Mô tả FSM:**

**IDLE:** Chờ tín hiệu i\_pass\_valid và i\_ena\_hash để bắt đầu. Chuẩn bị dữ liệu mật khẩu và salt đầu vào.

**KEY\_PADDING:** Thực hiện padding cho khóa bên trong (ikey\_pad) và khóa bên ngoài (okey\_pad) bằng cách XOR với ipad và opad.

**HASH\_IKEY\_BLOCK\_IN:** Bắt đầu băm khối đầu tiên với ikey\_pad

**HASH\_IKEY\_BLOCK\_OUT:** Lưu trữ kết quả băm khối đầu tiên vào data\_reg.

**HASH\_OKEY\_BLOCK\_IN:** Bắt đầu băm khối thứ 2 với okey\_pad và dữ liệu của data\_reg ( kết quả băm của khối i\_key )

**HASH\_OKEY\_BLOCK\_OUT:** Lưu trữ kết quả băm khối thứ hai vào data\_reg

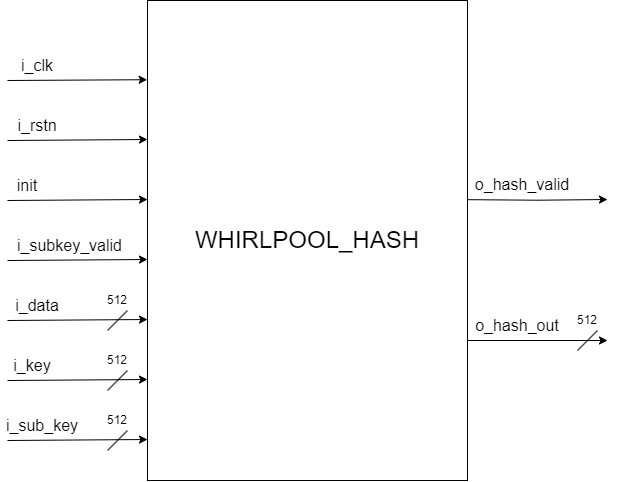
**KEY\_OUT:** XOR kết quả băm của khối thứ hai vào kết quả trước đó

Nếu đủ 1000 vòng lặp, báo hiệu kết quả hợp lệ (o\_key\_valid bật); nếu không, quay lại băm khối tiếp theo.

### 3.1.2. Module WHIRLPOOL\_HASH

#### 3.1.2.1. Mô tả In/Out

Là mạch tổ hợp tính toán mã băm cho từng round với đầu vào là i\_message hoặc là giá trị hash của round trước đó

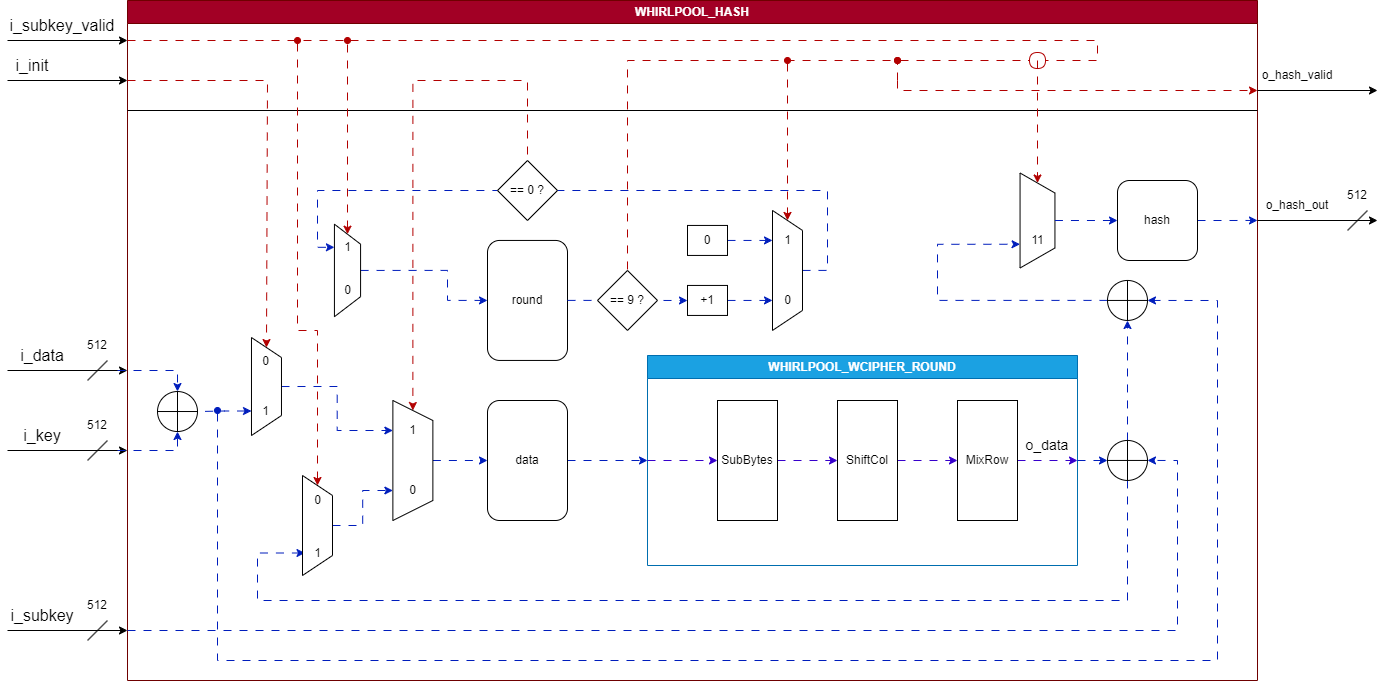


Hình . IO Module WHIRLPOOL\_HASH

|  |  |  |  |
| --- | --- | --- | --- |
| Tín hiệu | Loại | Kích thước (bit) | Mô tả |
| i\_clk | Input | 1 | Xung clock sườn dương |
| i\_rstn | Input | 1 | Tín hiệu reset sườn âm |
| init | Input | 1 | Tín hiệu bắt đầu quá trình mới |
| i\_key | Input | 512 | Key đầu vào cho Key Expansion |
| i\_subkey\_valid | Input | 1 | Tín hiệu báo hiệu key hợp lệ từ Key Expansion |
| i\_sub\_key | Input | 512 | Round Key được sinh ra từ Key Expansion |
| i\_data | Input | 1 | Dữ liệu cần băm |
| o\_hash | Output | 512 | Chuỗi mã băm đầu ra |
| o\_hash\_valid | Output | 1 | Tín hiệu báo hiệu tính xong mã băm |

Bảng . I/O của module WHIRLPOOL\_HASH

#### 3.1.2.2. Control Path & Data Path



Hình . Control Path & Data Path của module WHIRLPOOL\_HASH

**Mô tả:**

**Control Path:**

- Nhận các tín hiệu: init - tín hiệu cho phép bắt đầu quy trình băm mới, tín hiệu i\_subkey\_valid – tín hiệu báo round key hợp lệ từ key\_expansion

- Sau khi nhận được tín hiệu init, quá trình băm bắt đầu, tín hiệu init ở mức cao giúp cho data ( dữ liệu băm ) được cập nhật giá trị khởi tạo.

- Khi nhận được tín hiệu i\_subkey\_valid, data sẽ được cập nhật thành dữ liệu băm của vòng mới, quá trình trên tiếp diễn ở 10 vòng. Khi đến vòng thứ 10, control path sẽ gởi tín hiệu về data path báo hiệu rằng quá trình băm đã hoàn tất, tải dữ liệu từ data ra output. Và chuyển tín hiệu o\_hash\_valid – báo hiệu mã băm hợp lệ lên mức cao.

**Data path:**

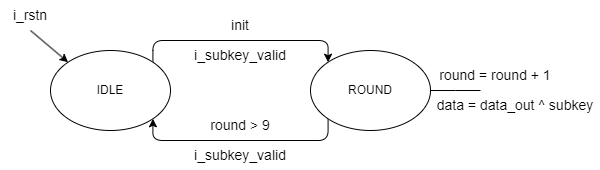
- Nhận i\_data – dữ liệu cần băm, i\_key – khóa khởi tạo, i\_subkey – khóa vòng.

- Khi có tín hiệu init từ control path, data sẽ được cập nhật giá trị:

- Data sau đó sẽ được đưa qua khối WCIPHER\_ROUND để xử lí theo thuật toán whirlpool dể sinh ra o\_data. Sau khi nhận được tín hiệu i\_subkey\_valid, o\_data sẽ được XOR với i\_subkey và cập nhật vào data.

- Khi nhận được tín hiệu kết thúc ( round == 10 ) data sẽ được XOR 1 lần nữa với dữ liệu ban đầu và được chuyển đến o\_hash\_out.

#### 3.1.2.3. FSM

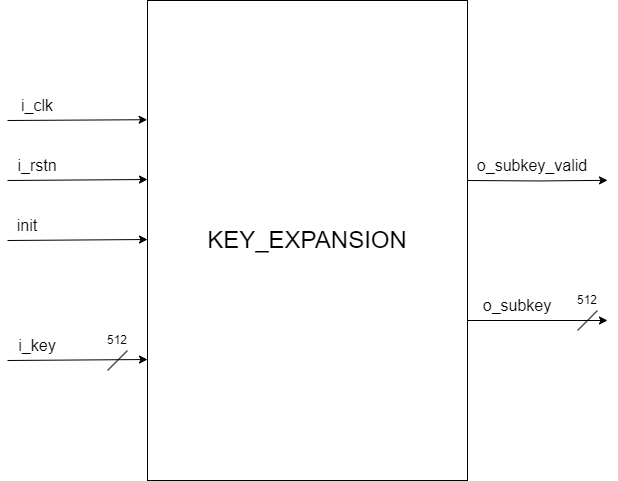


Hình . FSM của module WHIRLPOOL\_HASH

### 3.1.3. Module WHIRLPOOL\_KEY\_EXPANSION

#### 3.1.3.1. Mô tả In/Out

Là mạch tổ hợp tính round key để cung cấp cho module WHIRLPOOL\_HASH

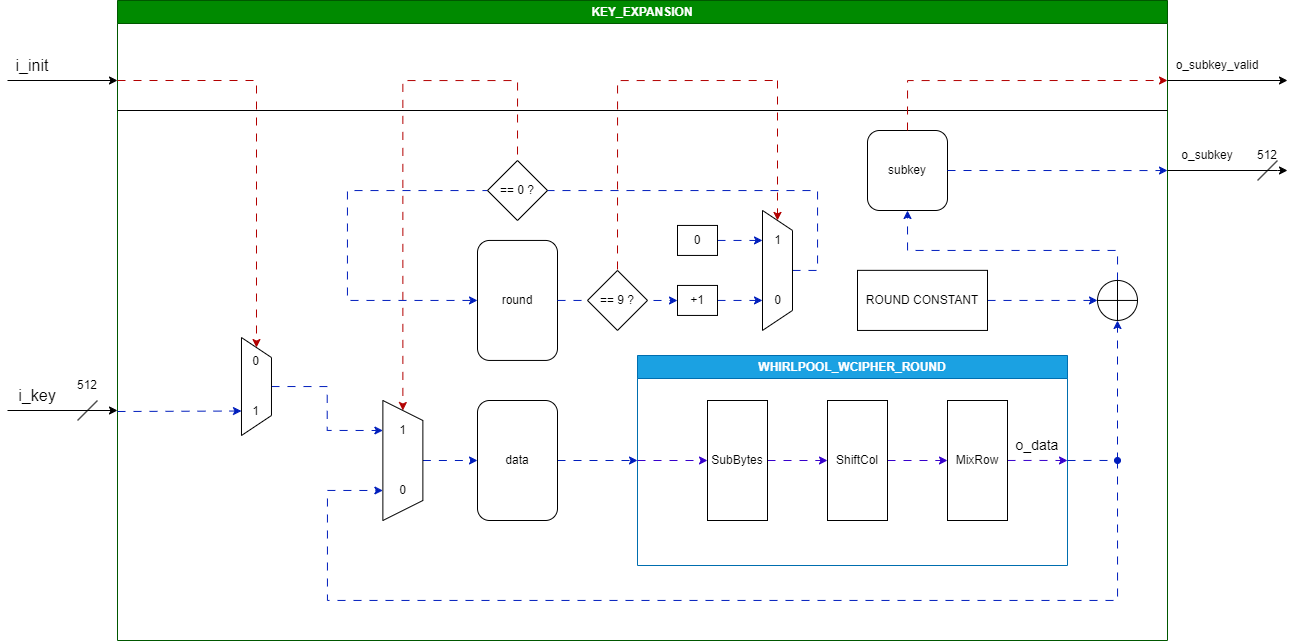


Hình . IO Module WHIRLPOOL\_KEY\_EXPANSION

|  |  |  |  |
| --- | --- | --- | --- |
| Tín hiệu | Loại | Kích thước (bit) | Mô tả |
| i\_clk | Input | 1 | Xung clock sườn dương |
| i\_rstn | Input | 1 | Tín hiệu reset sườn âm |
| init | Input | 1 | Tín hiệu bắt đầu quá trình mới |
| i\_key | Input | 512 | Key đầu vào |
| o\_subkey\_valid | Input | 1 | Tín hiệu báo hiệu round key hợp lệ |
| o\_subkey | Input | 512 | Round Key dữ liệu đầu ra |

Bảng . I/O của module WHIRLPOOL\_KEY\_EXPANSION

#### 3.1.3.2. Control Path & Data Path



Hình . Control Path & DataPath của module Key\_Expansion

**Mô tả:**

**Control Path:**

- Nhận các tín hiệu: init - tín hiệu cho phép bắt đầu quy trình tạo khóa mới

- Sau khi nhận được tín hiệu init, quá trình tạo khóa bắt đầu, tín hiệu init ở mức cao giúp cho data ( mã khóa ) được cập nhật giá trị i\_key khởi tạo

- Quá trình tạo khóa vòng sẽ được diễn ra 10 vòng liên tục theo xung clock và tín hiệu o\_subkey\_valid – báo hiệu round key hợp lệ sẽ được chuyển lên mức cao khi round key mới được sinh ra.

**Data Path:**

- Nhận i\_key – khóa khởi tạo làm đầu vào.

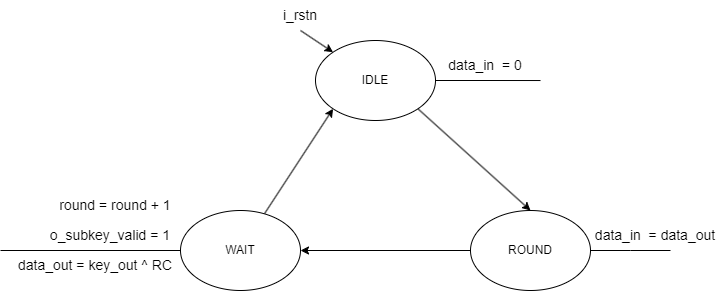
- Khi có tín hiệu init từ control path, data sẽ được cập nhật bằng i\_key

- Data sẽ được đưa qua khối WCIPHER tương tự như WHIRLPOOL\_HASH để sinh ra data\_out là thành phần của round key. , Với RC ( Round Constant ) là một mảng thanh ghi 512bit với giá trị mỗi vòng như sau.

|  |  |
| --- | --- |
| Round | Round Constant ( HEX ) |
| 1 | 1823c6e887b8014f |
| 2 | 36a6d2f5796f9152 |
| 3 | 60bc9b8ea30c7b35 |
| 4 | 1de0d7c22e4bfe57 |
| 5 | 157737e59ff04ada |
| 6 | 58c9290ab1a06b85 |
| 7 | bd5d10f4cb3e0567 |
| 8 | e427418ba77d95d8 |
| 9 | fbee7c66dd17479e |
| 10 | ca2dbf07ad5a8333 |

*Bảng . Giá trị của Round Constant*

#### 3.1.3.3. FSM



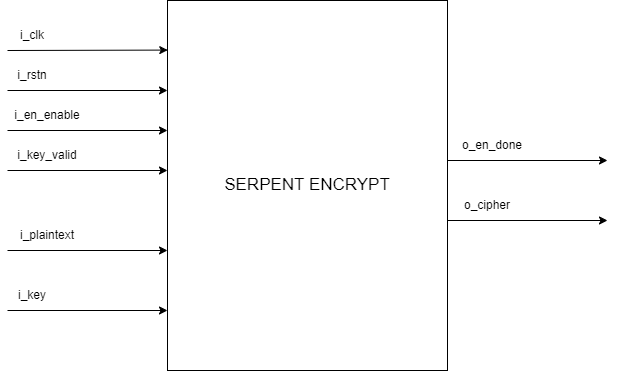
Hình . FSM của Key\_Expansion

## 3.3. XTS Serpent

### 3.3.1. Module XTS

### 3.3.2. Module Serpent encrypt

#### 3.3.2.1. Mô tả In/Out

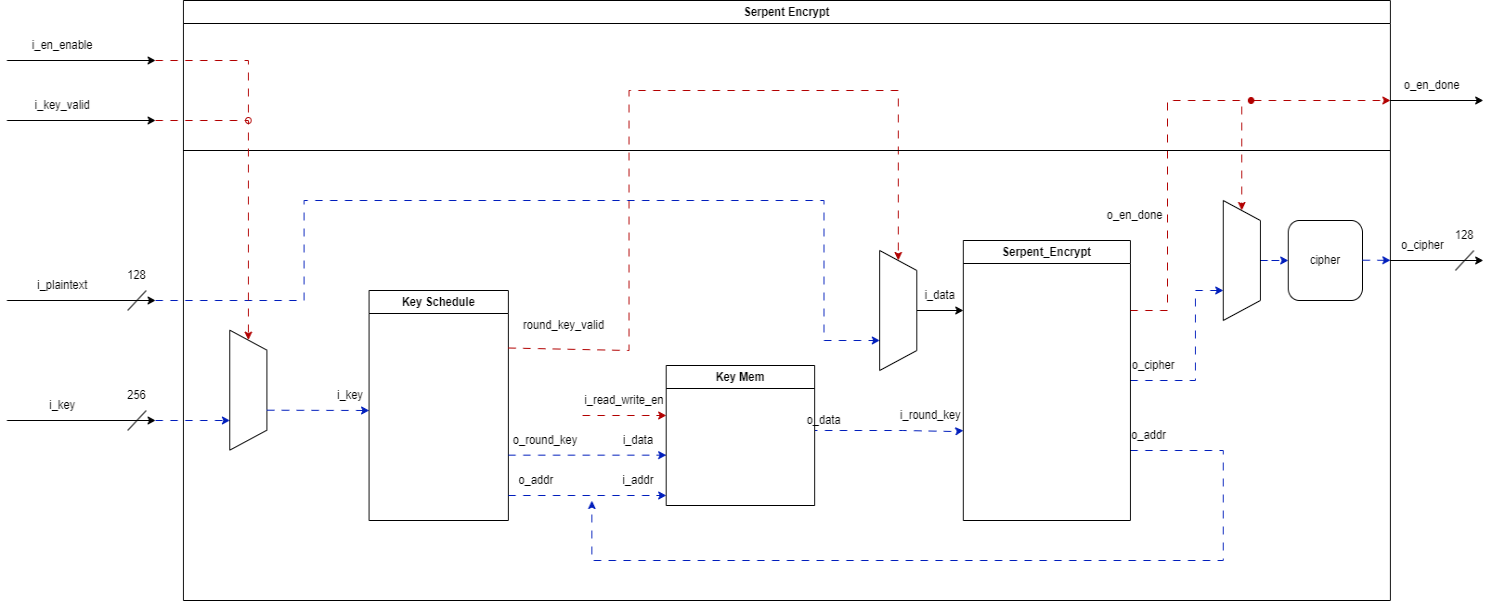


Hình . IO Module SERPENT\_ENCRYPT

|  |  |  |  |
| --- | --- | --- | --- |
| Tín hiệu | Loại | Kích thước (bit) | Mô tả |
| i\_clk | Input | 1 | Xung clock sườn dương |
| i\_rstn | Input | 1 | Tín hiệu reset sườn âm |
| i\_en\_enable | Input | 1 | Tín hiệu cho phép mã hóa từ XTS |
| i\_key\_valid | Input | 1 | Tín hiệu báo hiệu key hợp lệ |
| i\_plain\_text | Input | 128 | Dữ liệu cần mã hóa |
| i\_key | Input | 256 | Key đầu vào |
| o\_en\_done | Input | 1 | Tín hiệu báo quá trình mã hóa hoàn tất |
| o\_cipher | Input | 128 | Dữ liệu đã mã hóa |

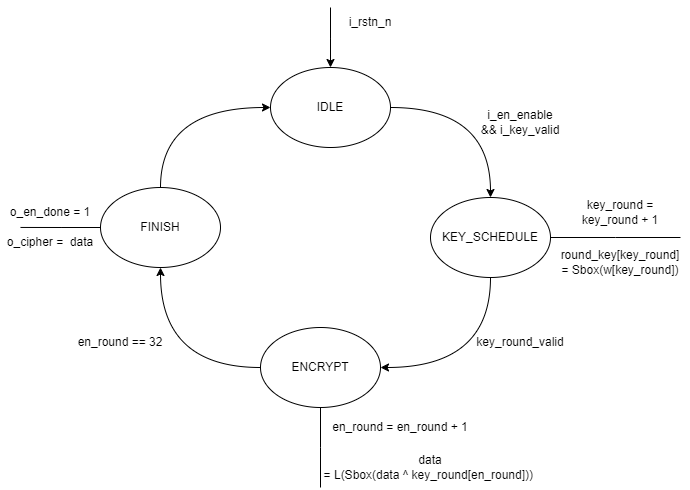
Bảng . Mô tả I/O của module SERPENT\_ENCRYPT

#### 3.3.2.2. Control Path & Data Path



Hình . Control Path & Data Path của SERPENT\_ENCRYPT

#### **3.3.2.3. FSM**



Hình . FSM của SERPENT\_ENCRYPT