(12) BẢN MÔ TẢ SÁNG CHẾ THUỘC BẰNG ĐỘC QUYỀN SÁNG CHẾ

(19) **Cộng hòa xã hội chủ nghĩa Việt Nam (VN)**
CỤC SỞ HỮU TRÍ TUỆ

(11) **1-0052135**

(51)[2021.01] **H04N 19/70**; H04N 19/137; H04N 19/176; H04N 19/91; H04N 19/60; H04N 19/132; H04N 19/18

(13) **B**

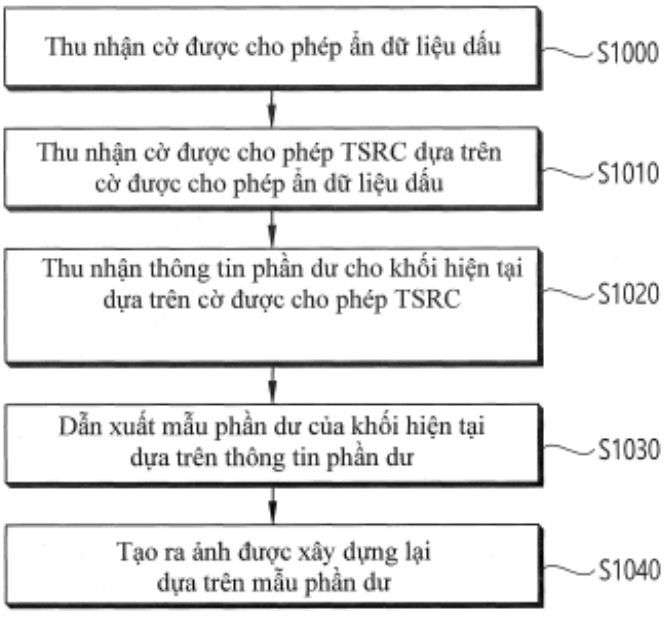
(21) 1-2022-06975
(22) 30/03/2021
(86) PCT/KR2021/003892 30/03/2021
(87) WO2021/201549 07/10/2021
(30) 63/003,221 31/03/2020 US
(45) 27/10/2025 451
(43) 26/12/2022 417A
(73) LG ELECTRONICS INC. (KR)
128, Yeoui-daero, Yeongdeungpo-gu Seoul 07336, Korea
(72) CHOI, Jungah (KR); YOO, Sunmi (KR); NAM, Junghak (KR); HEO, Jin (KR); CHOI, Jangwon (KR); KIM, Seunghwan (KR).
(74) Công ty Luật TNHH T&G (TGVN)

(54) PHƯƠNG PHÁP GIẢI MÃ HÌNH ẢNH, PHƯƠNG PHÁP MÃ HÓA HÌNH ẢNH, PHƯƠNG TIỆN LƯU TRỮ ĐỌC ĐƯỢC BỞI MÁY TÍNH KHÔNG CHUYỂN TIẾP VÀ PHƯƠNG PHÁP TRUYỀN DỮ LIỆU CHO THÔNG TIN HÌNH ẢNH

(21) 1-2022-06975

(57) Sáng chế đề cập đến phương pháp giải mã hình ảnh, phương pháp mã hóa hình ảnh, phương tiện lưu trữ đọc được bởi máy tính không chuyển tiếp và phương pháp truyền dữ liệu cho thông tin hình ảnh. Phương pháp giải mã hình ảnh được thực hiện bởi thiết bị giải mã, theo sáng chế, bao gồm các bước: thu cờ cho phép ẩn dữ liệu dấu; thu cờ cho phép lập mã phần dư bỏ qua biến đổi (transform skip residual coding, TSRC) trên cơ sở của cờ cho phép ẩn dữ liệu dấu; thu thông tin phần dư cho khối hiện tại, trên cơ sở của cờ cho phép TSRC; dẫn xuất mẫu phần dư của khối hiện tại, trên cơ sở của thông tin phần dư; và tạo ra ảnh được xây dựng lại trên cơ sở của mẫu phần dư, trong đó cờ cho phép ẩn dữ liệu dấu là cờ chỉ ra xem việc ẩn dữ liệu dấu có được cho phép hay không, cờ cho phép TSRC là cờ chỉ ra xem TSRC có được cho phép hay không, và cờ cho phép TSRC được thu trên cơ sở của cờ cho phép ẩn dữ liệu dấu có giá trị là không.

FIG. 10



Lĩnh vực kỹ thuật được đề cập

Sáng chế đề cập đến công nghệ lập mã hình ảnh, và cụ thể hơn, đề cập đến phương pháp và thiết bị giải mã hình ảnh, trong đó thông tin cờ lập mã thể hiện xem lập mã phần dư bỏ qua biến đổi (transform skip residual coding, TSRC) có được cho phép trong dữ liệu phần dư lập mã của khối hiện tại trong hệ thống lập mã hình ảnh hay không.

Tình trạng kỹ thuật của sáng chế

Hiện tại, nhu cầu về các hình ảnh độ phân giải cao, chất lượng cao, chẳng hạn như các hình ảnh độ phân giải cao (High Definition, HD) và các hình ảnh độ phân giải cực cao (Ultra High Definition, UHD), đang gia tăng trong các lĩnh vực khác nhau. Do dữ liệu hình ảnh có độ phân giải cao và chất lượng cao, lượng thông tin hoặc các bit cần được truyền tăng so với dữ liệu hình ảnh có trước. Do đó, khi dữ liệu hình ảnh được truyền sử dụng phương tiện như là đường băng rộng có dây/không dây thông thường hoặc dữ liệu hình ảnh được lưu trữ sử dụng phương tiện lưu trữ sẵn có, chi phí truyền và chi phí lưu trữ của chúng tăng.

Theo đó, có đòi hỏi về công nghệ nén hình ảnh hiệu quả cao để truyền, lưu trữ, và tái tạo một cách hiệu quả thông tin của các ảnh độ phân giải cao và chất lượng cao.

Bản chất kỹ thuật của sáng chế

Sáng chế đề xuất phương pháp và thiết bị để cải thiện hiệu quả lập mã hình ảnh.

Sáng chế cũng đề xuất phương pháp và thiết bị để cải thiện hiệu quả lập mã phần dư.

Theo một phương án của sáng chế, phương pháp giải mã hình ảnh được thực hiện bởi thiết bị giải mã được đề xuất. Phương pháp gồm các bước sau: thu nhận cờ được cho phép ẩn dữ liệu dấu; thu nhận cờ được cho phép lập mã phần dư bỏ qua biến đổi (transform skip residual coding, TSRC) dựa trên cờ được cho phép ẩn dữ liệu dấu;

thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC; dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư; và tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư, trong đó cờ được cho phép ẩn dữ liệu dấu là cờ về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không, trong đó cờ được cho phép TSRC là cờ về việc liệu TSRC có được cho phép hay không, và trong đó cờ được cho phép TSRC được thu nhận dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0.

Theo một phương án khác của sáng chế, thiết bị giải mã thực hiện việc giải mã hình ảnh được đề xuất. Thiết bị giải mã gồm: bộ giải mã entrôpi được tạo cấu hình để thu nhận cờ được cho phép ẩn dữ liệu dấu, để thu nhận cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) dựa trên cờ được cho phép ẩn dữ liệu dấu, để thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC; bộ xử lý phần dư được tạo cấu hình để dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư; và bộ cộng được tạo cấu hình để tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư, trong đó cờ được cho phép ẩn dữ liệu dấu là cờ về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không, trong đó cờ được cho phép TSRC là cờ về việc liệu TSRC có được cho phép hay không, và trong đó cờ được cho phép TSRC được thu nhận dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0.

Theo một phương án khác nữa của sáng chế, phương pháp mã hóa viđeo được thực hiện bởi thiết bị mã hóa được đề xuất. Phương pháp gồm các bước sau: mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không; mã hóa cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được cho phép hay không dựa trên cờ được cho phép ẩn dữ liệu dấu; mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC; và tạo ra luồng bit gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC và thông tin phần dư, trong đó cờ được cho phép TSRC được mã hóa dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0.

Theo một phương án khác nữa của sáng chế, thiết bị mã hóa viđeo được đề xuất. Thiết bị mã hóa gồm bộ mã hóa entrôpi được tạo cấu hình để mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không, để mã hóa cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được cho phép hay không dựa trên cờ được cho phép ẩn dữ liệu dấu, để mã hóa thông tin phần

dư cho khối hiện tại dựa trên cờ được cho phép TSRC, để tạo ra luồng bit gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC và thông tin phần dư, trong đó cờ được cho phép TSRC được mã hóa dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0.

Theo một phương án khác nữa của sáng chế, phương tiện lưu trữ đọc được bởi máy tính không chuyển tiếp lưu trữ luồng bit gồm thông tin hình ảnh khiến thực hiện phương pháp giải mã hình ảnh được đề xuất. Trong phương tiện lưu trữ đọc được bởi máy tính không chuyển tiếp, phương pháp giải mã hình ảnh gồm: thu nhận cờ được cho phép ẩn dữ liệu dấu; thu nhận cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) dựa trên cờ được cho phép ẩn dữ liệu dấu; thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC; dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư; và tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư, trong đó cờ được cho phép ẩn dữ liệu dấu là cờ về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không, trong đó cờ được cho phép TSRC là cờ về việc liệu TSRC có được cho phép hay không, và trong đó cờ được cho phép TSRC được thu nhận dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0.

Theo sáng chế, hiệu quả lập mã phần dư có thể được nâng cao.

Theo sáng chế, cờ được cho phép TSRC có thể được báo hiệu khi việc ẩn dữ liệu dấu không được cho phép bằng việc thiết đặt mối quan hệ báo hiệu giữa cờ được cho phép ẩn dữ liệu dấu và cờ được cho phép TSRC, qua đó, khi cú pháp RRC được lập mã cho khối bỏ qua biến đổi bởi vì TSRC không được cho phép, việc ẩn dữ liệu dấu không được sử dụng để cải thiện hiệu quả lập mã, và hiệu quả lập mã phần dư tổng thể có thể được cải thiện thông qua việc giảm lượng các bit được lập mã.

Theo sáng chế, mối quan hệ báo hiệu giữa cờ được cho phép lượng tử hóa phụ thuộc và cờ được cho phép TSRC được thiết lập, và nếu việc lượng tử hóa phụ thuộc là không được cho phép, cờ được cho phép TSRC có thể được báo hiệu, và qua đó, nếu TSRC là không được cho phép và cú pháp RRC được lập mã cho khối bỏ qua biến đổi, việc lượng tử hóa phụ thuộc là sẽ không được sử dụng, do đó hiệu quả lập mã được cải thiện, và hiệu quả lập mã phần dư tổng có thể được cải thiện thông qua việc giảm của lượng các bit được lập mã.

Theo sáng chế, mối quan hệ báo hiệu giữa cờ được cho phép bỏ qua biến đổi và cờ được cho phép TSRC được thiết lập, và nếu việc bỏ qua biến đổi được cho phép, cờ được cho phép TSRC có thể được báo hiệu, và qua đó, hiệu quả lập mã phần dư tổng thể có thể được cải thiện thông qua việc giảm lượng các bit được lập mã.

Mô tả vắn tắt các hình vẽ

Fig.1 minh họa sơ lược ví dụ của thiết bị lập mã viđeo/hình ảnh mà cho nó các phương án của sáng chế có thể áp dụng.

Fig.2 là sơ đồ sơ lược minh họa cấu hình của thiết bị mã hóa viđeo/hình ảnh mà cho nó (các) phương án của sáng chế có thể được áp dụng.

Fig.3 là sơ đồ sơ lược minh họa cấu hình của thiết bị giải mã hình ảnh/viđeo mà với nó (các) phương án của sáng chế có thể áp dụng.

Fig.4 thể hiện ví dụ lập mã số học nhị phân thích ứng ngữ cảnh (CABAC) cho việc mã hóa phần tử cú pháp.

Fig.5 là sơ đồ thể hiện các hệ số biến đổi ví dụ trong khối 4x4.

Fig.6 minh họa ví dụ các bộ lượng tử hóa vô hướng được sử dụng trong việc lượng tử hóa phụ thuộc.

Fig.7 minh họa ví dụ việc chuyển tiếp trạng thái và việc lựa chọn bộ lượng tử hóa cho việc lượng tử hóa phụ thuộc.

Fig.8 minh họa sơ lược phương pháp mã hóa hình ảnh được thực hiện bởi thiết bị mã hóa theo phần bộc lộ này.

Fig.9 minh họa sơ lược thiết bị mã hóa để thực hiện phương pháp mã hóa hình ảnh theo phần bộc lộ này.

Fig.10 minh họa sơ lược phương pháp giải mã hình ảnh được thực hiện bởi thiết bị giải mã theo phần bộc lộ này.

Fig.11 minh họa sơ lược thiết bị giải mã để thực hiện phương pháp giải mã hình ảnh theo sáng chế.

Fig.12 minh họa sơ đồ cấu trúc của hệ thống phát luồng các nội dung với nó sáng chế được áp dụng.

Mô tả chi tiết sáng chế

Sáng chế có thể được sửa đổi theo các dạng khác nhau, và các phương án cụ thể của sáng chế sẽ được mô tả và được minh họa trên các hình vẽ. Tuy nhiên, các phương án này không nhằm để giới hạn sáng chế. Các thuật ngữ được sử dụng trong phần mô tả sau đây được sử dụng chỉ đơn thuần để mô tả các phương án cụ thể, chứ không nhằm giới hạn sáng chế. Sự trình bày dạng số ít sẽ gồm sự trình bày dạng số nhiều, miễn là nó được đọc khác nhau một cách rõ ràng. Các từ ngữ như "gồm" và "có" được nhằm để chỉ ra rằng các dấu hiệu, các số lượng, các bước, các hoạt động, các phần tử, các thành phần, hoặc các tổ hợp của chúng, mà được sử dụng trong phần mô tả sau đây, là tồn tại, và vì thế, nên được hiểu rằng khả năng tồn tại hoặc bổ sung của một hoặc nhiều dấu hiệu, số lượng, bước, hoạt động, phần tử, thành phần khác, hoặc những tổ hợp của chúng, là không bị loại trừ.

Trong khi đó, các phần tử trên các hình vẽ mà được mô tả theo sáng chế là được vẽ độc lập để thuận tiện cho việc mô tả các chức năng cụ thể khác nhau, và không có nghĩa rằng các phần tử này là được thực hiện bởi phần cứng độc lập hoặc phần mềm độc lập. Ví dụ, hai hoặc nhiều phần tử trong số các phần tử là có thể được kết hợp để tạo thành một phần tử, hoặc một phần tử có thể được phân vùng thành nhiều phần tử. Các phương án mà trong đó các phần tử được kết hợp và/hoặc được phân vùng là thuộc sáng chế chứ không nằm ngoài phạm vi của sáng chế.

Ở dưới đây, các phương án của sáng chế sẽ được mô tả chi tiết với sự tham khảo đến các hình vẽ kèm theo. Bên cạnh đó, các số chỉ dẫn giống nhau được dùng để chỉ các thành phần giống nhau trong suốt các hình vẽ, và các phần mô tả giống nhau của các thành phần giống nhau này sẽ được lược bỏ.

Fig.1 minh họa sơ lược ví dụ của thiết bị lập mã viđeo/hình ảnh mà cho nó các phương án của sáng chế có thể áp dụng.

Tham khảo đến Fig.1, hệ thống lập mã viđeo/hình ảnh có thể gồm thiết bị thứ nhất (thiết bị nguồn) và thiết bị thứ hai (thiết bị nhận). Thiết bị nguồn có thể chuyển dữ liệu hoặc thông tin viđeo/hình ảnh được mã hóa dưới dạng tệp hoặc phát luồng tới thiết bị nhận thông qua màng hoặc phương tiện lưu trữ kỹ thuật số.

Thiết bị nguồn có thể gồm nguồn viđeo, thiết bị mã hóa, và bộ truyền. Thiết bị nhận có thể gồm bộ nhận, thiết bị giải mã, và bộ kết xuất. Thiết bị mã hóa có thể được gọi là thiết bị mã hóa viđeo/hình ảnh, và thiết bị giải mã có thể được gọi là thiết bị giải mã viđeo/hình ảnh. Bộ truyền có thể được gồm trong thiết bị mã hóa. Bộ nhận có thể được gồm trong thiết bị giải mã. Bộ kết xuất có thể gồm bộ phận hiển thị, bộ phận hiển thị có thể được tạo cấu hình là thiết bị riêng biệt hoặc thành phần ngoài.

Nguồn viđeo có thể giành được viđeo/hình ảnh thông qua quy trình chụp, tổng hợp, hoặc tạo ra viđeo/hình ảnh. Nguồn viđeo có thể gồm thiết bị chụp viđeo/hình ảnh và/hoặc thiết bị tạo ra viđeo/hình ảnh. Thiết bị chụp viđeo/hình ảnh có thể gồm, ví dụ, một hoặc nhiều camêra, kho lưu trữ viđeo/hình ảnh gồm các viđeo/hình ảnh đã được chụp trước đó, và tương tự. Thiết bị tạo ra viđeo/hình ảnh có thể gồm, ví dụ, máy tính, máy tính bảng, và điện thoại thông minh, và có thể tạo ra các viđeo/hình ảnh (theo cách điện). Ví dụ, viđeo/hình ảnh ảo có thể được tạo ra thông qua máy tính hoặc tương tự. Trong trường hợp này, quy trình chụp viđeo/hình ảnh có thể được thay thế bởi quy trình tạo ra dữ liệu liên quan.

Thiết bị mã hóa có thể mã hóa hình ảnh/hình ảnh đầu vào. Thiết bị mã hóa có thể thực hiện một loạt các thủ tục như là dự đoán, biến đổi, và lượng tử hóa cho hiệu quả nén và lập mã. Dữ liệu được mã hóa (thông tin viđeo/hình ảnh được mã hóa) có thể được xuất ra trong dạng luồng bit.

Bộ truyền có thể truyền hình ảnh/thông tin hình ảnh được mã hóa hoặc dữ liệu được xuất trong dạng luồng bit đến bộ nhận của thiết bị nhận thông qua phương tiện lưu trữ kỹ thuật số hoặc mạng trong dạng tệp hoặc phát luồng. Phương tiện lưu trữ kỹ thuật số có thể gồm các phương tiện lưu trữ khác nhau như là USB, SD, CD, DVD, Blu-ray, HDD, SSD, và tương tự. Bộ truyền có thể gồm phần tử để tạo ra tệp phương tiện thông qua định dạng tệp được xác định trước và có thể gồm phần tử để truyền thông qua mạng phát rộng/truyền thông. Bộ nhận có thể nhận/trích xuất luồng bit và truyền luồng bit nhận được đến thiết bị giải mã.

Thiết bị giải mã có thể giải mã viđeo/hình ảnh bằng cách thực hiện một loạt các thủ tục như là khử lượng tử hóa, biến đổi ngược, và dự đoán tương ứng với hoạt động của thiết bị mã hóa.

Bộ kết xuất có thể kết xuất viđeo/hình ảnh được giải mã. Viđeo/hình ảnh được kết xuất có thể được hiển thị qua bộ phận hiển thị.

Sáng chế liên quan đến việc lập mã viđeo/hình ảnh. Ví dụ, các phương pháp/các phương án được bộc lộ trong sáng chế có thể được áp dụng cho phương pháp được bộc lộ trong tiêu chuẩn lập mã viđeo vạn năng (Versatile Video Coding, VVC), tiêu chuẩn lập mã viđeo thiết yếu (Essential Video Coding, EVC), tiêu chuẩn AOMedia viđeo 1 (AOMedia Video 1, AV1), thế hệ thứ 2 của tiêu chuẩn lập mã auđio viđeo (2nd Generation of Audio Video Coding Standard, AVS2), hoặc tiêu chuẩn lập mã viđeo/hình ảnh thế hệ tiếp theo (ví dụ, H.267 hoặc H.268, v.v.).

Sáng chế đề xuất các phương án khác nhau của việc lập mã viđeo/hình ảnh, và các phương án ở trên có thể được thực hiện trong sự tổ hợp với nhau trừ khi được chỉ rõ theo cách khác.

Trong sáng chế, viđeo có thể đề cập đến một loạt các hình ảnh theo thời gian. Ảnh đề cập chung đến đơn vị biểu diễn một hình ảnh trong khung thời gian cụ thể, và lát/phiến/ảnh con là đơn vị cấu thành một phần của ảnh trong việc lập mã. Lát/phiến/ảnh con có thể gồm một hoặc nhiều đơn vị cây lập mã (Coding Tree Unit, CTU). Một ảnh có thể gồm một hoặc nhiều lát/phiến/ảnh con. Một ảnh có thể gồm một hoặc nhiều nhóm phiến. Một nhóm phiến có thể gồm một hoặc nhiều phiến. Một viên gạch có thể biểu diễn vùng hình chữ nhật của các hàng CTU nằm trong phiến trong ảnh. Phiến có thể được phân vùng thành nhiều viên gạch, mà mỗi viên trong số đó bao gồm một hoặc nhiều hàng CTU nằm trong một phiến. Phiến không được phân vùng thành nhiều viên gạch cũng có thể được gọi là viên gạch. Việc quét viên gạch là việc sắp thứ tự theo trình tự cụ thể của các CTU phân vùng ảnh, trong đó các CTU được sắp thứ tự liên tiếp trong việc quét mành CTU trong viên gạch, các viên gạch nằm trong phiến được sắp thứ tự liên tiếp trong việc quét mành của các viên gạch của phiến, và các phiến trong ảnh được sắp thứ tự liên tiếp trong việc quét mành của các phiến của ảnh. Ngoài ra, ảnh con có thể thể hiện vùng hình chữ nhật của một hoặc nhiều lát trong ảnh. Nghĩa là, ảnh con chứa một hoặc nhiều lát mà che phủ chung vùng chữ nhật của ảnh. Phiến là vùng hình chữ nhật của các CTU nằm trong cột phiến cụ thể và hàng phiến cụ thể trong ảnh. Cột phiến là vùng hình chữ nhật của các CTU có chiều cao bằng với chiều cao của ảnh và chiều rộng được chỉ rõ bởi các phần tử cú pháp trong tập thông số ảnh. Hàng phiến là

vùng hình chữ nhật của các CTU có chiều cao được chỉ rõ bởi các phần tử cú pháp trong tập thông số ảnh và chiều rộng bằng với chiều rộng của ảnh. Việc quét phiến là việc sắp thứ tự theo trình tự cụ thể của các CTU phân vùng ảnh mà trong đó các CTU được sắp thứ tự liên tiếp trong việc quét mành CTU trong phiến, trong khi các phiến trong ảnh được sắp thứ tự liên tiếp trong việc quét mành của các phiến của ảnh. Lát gồm số nguyên các viên gạch của ảnh có thể được gồm theo cách dành riêng trong đơn vị NAL đơn lẻ. Lát có thể gồm hoặc là số các phiến hoàn chỉnh hoặc chỉ là chuỗi liên tiếp của các viên gạch hoàn chỉnh của một phiến. Các nhóm phiến và các lát có thể được sử dụng thay thế cho nhau trong phần bộc lộ này. Ví dụ, trong sáng chế, phần đầu nhóm phiến/nhóm lát có thể được gọi là lát/phần đầu lát.

Điểm ảnh (pixel) hoặc pel có thể có nghĩa là đơn vị nhỏ nhất cấu thành một ảnh (hoặc hình ảnh). Ngoài ra, 'mẫu' có thể được sử dụng như là thuật ngữ tương ứng với điểm ảnh. Mẫu có thể biểu diễn chung cho điểm ảnh hoặc giá trị của điểm ảnh, và có thể chỉ biểu diễn điểm ảnh/giá trị điểm ảnh của thành phần độ chói hoặc chỉ điểm ảnh/giá trị điểm ảnh của thành phần sắc độ.

Đơn vị có thể biểu diễn đơn vị cơ bản của việc xử lý hình ảnh. Đơn vị có thể gồm ít nhất một thành phần trong số vùng cụ thể của ảnh và thông tin liên quan đến vùng. Một đơn vị có thể gồm một khối độ chói và hai khối sắc độ (ví dụ cb, cr). Đơn vị có thể được sử dụng theo kiểu thay thế cho nhau theo nghĩa như là khối hoặc khu vực trong một số trường hợp. Trong trường hợp chung, các khối M×N có thể gồm các mẫu (hoặc các mảng mẫu) hoặc tập hợp (hoặc các mảng) của các hệ số biến đổi của M cột và N hàng.

Trong sáng chế, thuật ngữ "A hoặc B" có thể có nghĩa là "chỉ A", "chỉ B", hoặc "cả A và B". Nói cách khác, trong sáng chế, "A hoặc B" có thể được diễn dịch là "A và/hoặc B". Ví dụ, thuật ngữ "A, B hoặc C" ở đây có nghĩa là "chỉ A", "chỉ B", "chỉ C", hoặc "bất kỳ và tổ hợp bất kỳ của A, B và C".

Dấu gạch chéo (/) hoặc dấu phẩy được sử dụng trong sáng chế có nghĩa là "và/hoặc". Ví dụ, "A/B" có nghĩa là "A và/hoặc B". Theo đó, "A/B" có nghĩa là "chỉ A", "chỉ B", hoặc "cả A và B". Ví dụ, "A, B, C" có thể có nghĩa là "A, B hoặc C".

Trong sáng chế, "ít nhất một thành phần trong số A và B" có thể chỉ "chỉ A", "chỉ B", hoặc "cả A và B". Ngoài ra, trong sáng chế, sự trình bày "ít nhất một thành phần trong số A hoặc B" hoặc "ít nhất một thành phần trong số A và/hoặc B" có thể được diễn giải giống với "ít nhất một thành phần trong số A và B".

Ngoài ra, trong sáng chế, "ít nhất một thành phần trong số A, B và C" có thể có nghĩa là "chỉ A", "chỉ B", "chỉ C", hoặc "tổ hợp bất kỳ của A, B và C". Ngoài ra, "ít nhất một thành phần trong số A, B hoặc C" hoặc "ít nhất một thành phần trong số A, B và/hoặc C" có thể có nghĩa là "ít nhất một thành phần trong số A, B và C".

Ngoài ra, dấu ngoặc đơn được sử dụng trong sáng chế có thể có nghĩa là "ví dụ". Cụ thể là, khi "việc dự đoán (việc dự đoán nội)" được chỉ ra, thì "việc dự đoán nội" có thể được đề xuất là ví dụ về "việc dự đoán". Nói cách khác, "việc dự đoán" trong sáng chế không được giới hạn ở "việc dự đoán nội", và "việc dự đoán nội" có thể được đề xuất là ví dụ của "việc dự đoán". Ngoài ra, ngay cả khi "việc dự đoán (tức là, việc dự đoán nội)" được chỉ ra, thì "việc dự đoán nội" có thể được đề xuất là ví dụ về "việc dự đoán".

Trong phần mô tả này, các dấu hiệu kỹ thuật mà được mô tả riêng lẻ trong một hình vẽ có thể được thực hiện một cách riêng lẻ hoặc có thể được thực hiện cùng lúc.

Các hình vẽ sau đây được tạo nên để diễn giải ví dụ cụ thể của bản mô tả này. Do các tên gọi của các thiết bị cụ thể được mô tả trong các hình vẽ hoặc các tên gọi của các tín hiệu/các thông điệp/các trường cụ thể được giới thiệu theo cách làm ví dụ, các dấu hiệu kỹ thuật của bản mô tả này không được giới hạn ở các tên gọi cụ thể được sử dụng trong các hình vẽ dưới đây.

Fig.2 là sơ đồ sơ lược minh họa cấu hình của thiết bị mã hóa viđeo/hình ảnh mà cho nó (các) phương án của sáng chế có thể được áp dụng. Ở dưới đây, thiết bị mã hóa viđeo có thể gồm thiết bị mã hóa hình ảnh.

Tham khảo đến Fig.2, thiết bị mã hóa 200 gồm bộ phân vùng hình ảnh 210, bộ dự đoán 220, bộ xử lý phần dư 230, và bộ mã hóa entrôpi 240, bộ cộng 250, bộ lọc 260, và bộ nhớ 270. Bộ dự đoán 220 có thể gồm bộ dự đoán liên 221 và bộ dự đoán nội 222. Bộ xử lý phần dư 230 có thể gồm bộ biến đổi 232, bộ lượng tử hóa 233, bộ khử lượng tử hóa 234, và bộ biến đổi ngược 235. Bộ xử lý phần dư 230 có thể còn gồm bộ trừ 231.

Bộ cộng 250 có thể được gọi là bộ xây dựng lại hoặc bộ tạo khối được xây dựng lại. Bộ phân vùng hình ảnh 210, bộ dự đoán 220, bộ xử lý phần dư 230, bộ mã hóa entrôpi 240, bộ cộng 250, và bộ lọc 260 có thể được tạo cấu hình bởi ít nhất một thành phần phần cứng (ví dụ, bộ chip bộ mã hóa hoặc bộ xử lý) theo một phương án. Bên cạnh đó, bộ nhớ 270 có thể gồm bộ đệm ảnh được giải mã (Decoded Picture Buffer, DPB) hoặc có thể được tạo cấu hình bởi phương tiện lưu trữ kỹ thuật số. Thành phần phần cứng có thể còn gồm bộ nhớ 270 như là thành phần trong/ngoài.

Bộ phân vùng hình ảnh 210 có thể phân vùng hình ảnh đầu vào (hoặc ảnh hoặc khung) được nhập vào thiết bị mã hóa 200 thành một hoặc nhiều bộ xử lý. Ví dụ, bộ xử lý có thể được gọi là đơn vị lập mã (Coding Unit, CU). Trong trường hợp này, đơn vị lập mã có thể có thể được phân vùng theo cách đệ quy theo cấu trúc cây tứ phân cây nhị phân cây tam phân (Quad-Tree Binary-Tree Ternary-Tree, QTBTTT) từ đơn vị cây lập mã (Coding Tree Unit, CTU) hoặc đơn vị lập mã lớn nhất (Largest Coding Unit, LCU). Ví dụ, một đơn vị lập mã có thể được phân vùng thành nhiều đơn vị lập mã có độ sâu sâu hơn dựa trên cấu trúc cây tứ phân, cấu trúc cây nhị phân, và/hoặc cấu trúc tam phân. Trong trường hợp này, ví dụ, cấu trúc cây tứ phân có thể được áp dụng trước tiên và cấu trúc cây nhị phân và/hoặc cấu trúc tam phân có thể được áp dụng sau. Theo cách thay thế, cấu trúc cây nhị phân có thể được áp dụng trước tiên. Thủ tục lập mã theo sáng chế có thể được thực hiện dựa trên đơn vị lập mã cuối cùng mà không được phân vùng tiếp nữa. Trong trường hợp này, đơn vị lập mã lớn nhất có thể được dùng làm đơn vị lập mã cuối cùng dựa trên hiệu quả lập mã theo các đặc tính của ảnh, hoặc nếu cần thiết, đơn vị lập mã có thể được phân vùng theo cách đệ quy thành các đơn vị lập mã có chiều sâu sâu hơn nếu cần, và đơn vị lập mã có kích cỡ tối ưu có thể được sử dụng làm đơn vị lập mã cuối cùng. Ở đây, thủ tục lập mã có thể gồm thủ tục dự đoán, biến đổi, và xây dựng lại, vốn sẽ được mô tả sau. Theo một ví dụ khác, bộ xử lý có thể còn gồm đơn vị dự đoán (Prediction Unit, PU) hoặc đơn vị biến đổi (Transform Unit, TU). Trong trường hợp này, đơn vị dự đoán và đơn vị biến đổi có thể được chia tách và được phân vùng từ đơn vị lập mã cuối cùng được đề cập ở trên. Đơn vị dự đoán có thể là đơn vị của mẫu dự đoán, và đơn vị biến đổi có thể là đơn vị để dẫn xuất hệ số biến đổi và/hoặc đơn vị để dẫn xuất tín hiệu dư từ hệ số biến đổi.

Đơn vị có thể được sử dụng theo kiểu thay thế cho nhau theo nghĩa như là khối hoặc khu vực trong một số trường hợp. Trong trường hợp chung, khối M×N có thể biểu diễn tập hợp của các mẫu hoặc các hệ số biến đổi được tạo thành từ M cột và N hàng. Mẫu có thể biểu diễn chung cho điểm ảnh hoặc giá trị của điểm ảnh, có thể biểu diễn chỉ điểm ảnh/giá trị điểm ảnh của thành phần độ chói hoặc chỉ biểu diễn điểm ảnh/giá trị điểm ảnh của thành phần sắc độ. Mẫu có thể được sử dụng theo nghĩa tương ứng với một ảnh (hoặc hình ảnh) hoặc điểm ảnh hoặc pel.

Trong thiết bị mã hóa 200, tín hiệu dự đoán (khối được dự đoán, mảng mẫu dự đoán) được xuất ra từ bộ dự đoán liên 221 hoặc bộ dự đoán nội 222 được trừ khỏi tín hiệu hình ảnh đầu vào (khối gốc, mảng mẫu gốc) để tạo ra khối phần dư tín hiệu dư, mảng mẫu phần dư), và tín hiệu dư được tạo ra được truyền đến bộ biến đổi 232. Trong trường hợp này, như được thể hiện, đơn vị để trừ tín hiệu dự đoán (khối được dự đoán, mảng mẫu dự đoán) khỏi tín hiệu hình ảnh đầu vào (khối gốc, mảng mẫu gốc) trong bộ mã hóa 200 có thể được gọi là bộ trừ 231. Bộ dự đoán có thể thực hiện việc dự đoán trên khối cần được xử lý (ở dưới đây được gọi là khối hiện tại) và tạo ra khối được dự đoán gồm các mẫu dự đoán cho khối hiện tại. Bộ dự đoán có thể xác định liệu việc dự đoán nội hay việc dự đoán liên được áp dụng trên cơ sở khối hiện tại hay trên CU. Như được mô tả ở dưới trong phần mô tả của mỗi chế độ dự đoán, bộ dự đoán có thể tạo ra thông tin khác nhau liên quan đến việc dự đoán, như thông tin chế độ dự đoán, và truyền thông tin được tạo ra đến bộ mã hóa entrôpi 240. Thông tin về việc dự đoán có thể được mã hóa trong bộ mã hóa entrôpi 240 và được xuất ra trong dạng luồng bit.

Bộ dự đoán nội 222 có thể dự đoán khối hiện tại bằng cách tham chiếu đến các mẫu trong ảnh hiện tại. Các mẫu được tham chiếu có thể được đặt trong vùng lân cận của khối hiện tại hoặc có thể được đặt tách ra theo chế độ dự đoán. Trong việc dự đoán nội, các chế độ dự đoán có thể gồm nhiều chế độ không có hướng và nhiều chế độ có hướng. Chế độ không có hướng có thể gồm, ví dụ, chế độ DC và chế độ phẳng. Chế độ có hướng có thể gồm, ví dụ, 33 chế độ dự đoán có hướng hoặc 65 chế độ dự đoán có hướng theo mức độ chi tiết của hướng dự đoán. Tuy nhiên, đây chỉ đơn thuần là ví dụ, nhiều hoặc ít chế độ dự đoán định hướng hơn có thể được sử dụng phụ thuộc vào việc thiết lập. Bộ dự đoán nội 222 có thể xác định chế độ dự đoán được áp dụng cho khối hiện tại bằng cách sử dụng chế độ dự đoán được áp dụng cho khối lân cận.

Bộ dự đoán liên 221 có thể dẫn xuất khối được dự đoán cho khối hiện tại dựa trên khối tham chiếu (mảng mẫu tham chiếu) được chỉ rõ bởi vectơ chuyển động trên ảnh tham chiếu. Ở đây, để làm giảm lượng thông tin chuyển động được truyền trong chế độ dự đoán liên, thì thông tin chuyển động có thể được dự đoán trong các đơn vị của các khối, các khối con, hoặc các mẫu dựa trên sự tương liên của thông tin chuyển động giữa khối lân cận và khối hiện tại. Thông tin chuyển động có thể gồm vectơ chuyển động và chỉ số ảnh tham chiếu. Thông tin chuyển động có thể còn gồm thông tin hướng dự đoán liên (việc dự đoán L0, việc dự đoán L1, việc dự đoán Bi, v.v.). Trong trường hợp của việc dự đoán liên, khối lân cận có thể gồm khối lân cận theo không gian có mặt trong ảnh hiện tại và khối lân cận theo thời gian có mặt trong ảnh tham chiếu. Ảnh tham chiếu gồm khối tham chiếu và ảnh tham chiếu gồm khối lân cận theo thời gian có thể là giống nhau hoặc khác nhau. Khối lân cận theo thời gian có thể được gọi là khối tham chiếu được đặt cùng một chỗ, CU được đặt cùng một chỗ (co-located CU, colCU), và tương tự, và ảnh tham chiếu gồm khối lân cận theo thời gian có thể được gọi là ảnh được đặt cùng một chỗ (colocated Picture, colPic). Ví dụ, bộ dự đoán liên 221 có thể tạo cấu hình danh sách ứng viên thông tin chuyển động dựa trên các khối lân cận và tạo ra thông tin đang chỉ ra ứng viên nào được sử dụng để dẫn xuất vectơ chuyển động và/hoặc chỉ số ảnh tham chiếu của khối hiện tại. Việc dự đoán liên có thể được thực hiện dựa trên các chế độ dự đoán khác nhau. Ví dụ, trong trường hợp của chế độ bỏ qua và chế độ hợp nhất, bộ dự đoán liên 221 có thể sử dụng thông tin chuyển động của khối lân cận làm thông tin chuyển động của khối hiện tại. Trong chế độ bỏ qua, không giống với chế độ hợp nhất, tín hiệu dư có thể không được truyền. Trong trường hợp của chế độ dự đoán vectơ chuyển động (Motion Vector Prediction, MVP), vectơ chuyển động của khối lân cận có thể được sử dụng làm bộ dự đoán vectơ chuyển động và vectơ chuyển động của khối hiện tại có thể được chỉ ra bằng cách báo hiệu sự chênh lệch vectơ chuyển động.

Bộ dự đoán 220 có thể tạo ra tín hiệu dự đoán dựa trên các phương pháp dự đoán khác nhau được mô tả dưới đây. Ví dụ, bộ dự đoán có thể không chỉ áp dụng việc dự đoán nội hoặc việc dự đoán liên để dự đoán một khối mà cũng có thể áp dụng một cách đồng thời cả việc dự đoán nội và việc dự đoán liên. Việc này có thể được gọi là việc dự đoán nội và liên được kết hợp (inter and intra prediction, CIIP). Bên cạnh đó, bộ dự đoán có thể dựa trên chế độ dự đoán sao chép khối nội (intra block copy, IBC) hoặc chế độ

bảng màu cho việc dự đoán của khối. Chế độ dự đoán IBC hoặc chế độ bảng màu có thể được sử dụng cho việc mã hóa hình ảnh/viđeo nội dung của trò chơi hoặc tương tự, ví dụ, lập mã nội dung màn (Screen Content Coding, SCC). IBC về cơ bản thực hiện việc dự đoán trong ảnh hiện tại mà có thể được thực hiện tương tự như việc dự đoán liên mà trong đó khối tham chiếu được dẫn xuất trong ảnh hiện tại. Nghĩa là, IBC có thể sử dụng ít nhất một kỹ thuật trong số các kỹ thuật dự đoán liên được mô tả trong sáng chế. Chế độ bảng màu có thể được coi như là, ví dụ, việc lập mã trong ảnh hoặc việc dự đoán nội. Khi chế độ bảng màu được áp dụng, thì giá trị mẫu nằm trong ảnh có thể được báo hiệu dựa trên thông tin về bảng của bảng màu và chỉ số bảng màu.

Tín hiệu dự đoán được tạo ra bởi bộ dự đoán (gồm bộ dự đoán liên 221 và/hoặc bộ dự đoán nội 222) có thể được sử dụng để tạo ra tín hiệu được xây dựng lại hoặc để tạo ra tín hiệu dư. Bộ biến đổi 232 có thể tạo ra các hệ số biến đổi bằng cách áp dụng kỹ thuật biến đổi cho tín hiệu dư. Ví dụ, kỹ thuật biến đổi có thể gồm ít nhất một kỹ thuật trong số biến đổi côsin rời rạc (Discrete Cosine Transform, DCT), biến đổi sin rời rạc (Discrete Sine Transform, DST), biến đổi Karhunen-loève (Karhunen-Loève Transform, KLT), biến đổi dựa trên đồ thị (Graph-Based Transform, GBT), hoặc biến đổi phi tuyến tùy theo điều kiện (Conditionally Non-Linear Transform, CNT). Ở đây, GBT nghĩa là việc biến đổi thu nhận được từ đồ thị khi thông tin quan hệ giữa các điểm ảnh được biểu diễn bởi đồ thị. CNT đề cập đến việc biến đổi được tạo ra dựa trên tín hiệu dự đoán được tạo ra nhờ sử dụng tất cả các điểm ảnh được xây dựng lại trước đó. Bên cạnh đó, quy trình biến đổi có thể được áp dụng cho các khối điểm ảnh hình vuông có cùng kích cỡ hoặc có thể được áp dụng cho các khối có kích cỡ khác hình vuông.

Bộ lượng tử hóa 233 có thể lượng tử hóa các hệ số biến đổi và truyền chúng đến bộ mã hóa entrôpi 240 và bộ mã hóa entrôpi 240 có thể mã hóa tín hiệu được lượng tử hóa (thông tin về các hệ số biến đổi được lượng tử hóa) và xuất ra luồng bit. Thông tin về các hệ số biến đổi được lượng tử hóa có thể được gọi là thông tin phần dư. Bộ lượng tử hóa 233 có thể sắp xếp lại các hệ số biến đổi được lượng tử hóa loại khối thành dạng vectơ một chiều dựa trên thứ tự quét hệ số và tạo ra thông tin về các hệ số biến đổi được lượng tử hóa dựa trên các hệ số biến đổi được lượng tử hóa trong dạng vectơ một chiều. Thông tin về các hệ số biến đổi có thể được tạo ra. Bộ mã hóa entrôpi 240 có thể thực hiện các phương pháp mã hóa khác nhau, ví dụ, như Golomb hàm số mũ, lập mã chiều

dài biến đổi thích ứng ngữ cảnh (Context-Adaptive Variable Length Coding, CAVLC), lập mã số học nhị phân thích ứng ngữ cảnh (Context-Adaptive Binary Arithmetic Coding, CABAC), và tương tự. Bộ mã hóa entrôpi 240 có thể mã hóa thông tin cần thiết cho việc xây dựng lại viđeo/hình ảnh khác với các hệ số biến đổi được lượng tử hóa (ví dụ, các giá trị của các phần tử cú pháp, v.v.) cùng nhau hoặc một cách riêng biệt. Thông tin được mã hóa (ví dụ, thông tin viđeo/hình ảnh được mã hóa) có thể được truyền hoặc được lưu trữ trong các đơn vị lớp trừu tượng mạng (Network Abstraction Layer, NAL) trong dạng luồng bit. Thông tin viđeo/hình ảnh có thể còn gồm thông tin về các tập thông số khác nhau như tập thông số thích ứng (adaptation parameter set, APS), tập thông số ảnh (picture parameter set, PPS), tập thông số trình tự (sequence parameter set, SPS), hoặc tập thông số viđeo (video parameter set, VPS). Bên cạnh đó, thông tin viđeo/hình ảnh có thể còn gồm thông tin ràng buộc chung. Trong sáng chế, thông tin và/hoặc các phần tử cú pháp được truyền/được báo hiệu từ thiết bị mã hóa đến thiết bị giải mã có thể được gồm trong thông tin viđeo/ảnh. Thông tin viđeo/hình ảnh có thể được mã hóa thông qua thủ tục mã hóa được mô tả ở trên và được gồm trong luồng bit. Luồng bit có thể được truyền qua mạng hoặc có thể được lưu trữ trong phương tiện lưu trữ kỹ thuật số. Mạng có thể gồm mạng phát rộng và/hoặc mạng truyền thông, và phương tiện lưu trữ kỹ thuật số có thể gồm các phương tiện lưu trữ khác nhau như USB, SD, CD, DVD, Blu-ray, HDD, SSD, và tương tự. Bộ truyền (không được thể hiện) truyền tín hiệu được xuất ra từ bộ mã hóa entrôpi 240 và/hoặc đơn vị lưu trữ (không được thể hiện) lưu trữ tín hiệu có thể được gồm dưới dạng thành phần trong/ngoài của thiết bị mã hóa 200, và theo cách thay thế, bộ truyền có thể được gồm trong bộ mã hóa entrôpi 240.

Các hệ số biến đổi được lượng tử hóa mà được xuất ra từ bộ lượng tử hóa 233 có thể được sử dụng để tạo ra tín hiệu dự đoán. Ví dụ, tín hiệu dư (khối phần dư hoặc các mẫu phần dư) có thể được xây dựng lại bằng cách áp dụng việc khử lượng tử hóa và biến đổi ngược cho các hệ số biến đổi được lượng tử hóa qua bộ khử lượng tử hóa 234 và bộ biến đổi ngược 235. Bộ cộng 250 cộng tín hiệu dư được xây dựng lại vào tín hiệu dự đoán được xuất ra từ bộ dự đoán liên 221 hoặc bộ dự đoán nội 222 để tạo ra tín hiệu được xây dựng lại (ảnh được xây dựng lại, khối được xây dựng lại, mảng mẫu được xây dựng lại). Nếu không có phần dư cho khối cần được xử lý, như trường hợp mà trong đó chế độ bỏ qua được áp dụng, thì khối được dự đoán có thể được sử dụng làm khối được

xây dựng lại. Bộ cộng 250 có thể được gọi là bộ xây dựng lại hoặc bộ tạo khối được xây dựng lại. Tín hiệu được xây dựng lại mà được tạo ra có thể được sử dụng cho việc dự đoán nội của khối tiếp theo cần được xử lý trong ảnh hiện tại và có thể được sử dụng cho việc dự đoán liên của ảnh tiếp theo qua việc lọc như được mô tả dưới đây.

Trong khi đó, việc ánh xạ độ chói với việc định cỡ sắc độ (luma mapping with chroma scaling, LMCS) có thể được áp dụng trong suốt quá trình mã hóa và/hoặc xây dựng lại ảnh.

Bộ lọc 260 có thể cải thiện chất lượng hình ảnh chủ quan/khách quan bằng cách áp dụng việc lọc cho tín hiệu được xây dựng lại. Ví dụ, bộ lọc 260 có thể tạo ra ảnh được xây dựng lại mà được sửa đổi bằng cách áp dụng các phương pháp lọc khác nhau cho ảnh được xây dựng lại và lưu trữ ảnh được xây dựng lại mà sửa được đổi trong bộ nhớ 270, cụ thể là, DPB của bộ nhớ 270. Các phương pháp lọc khác nhau có thể gồm, ví dụ, lọc khử khối, dịch chuyển thích ứng mẫu, bộ lọc vòng lặp thích ứng, bộ lọc song phương, và tương tự. Bộ lọc 260 có thể tạo ra thông tin khác nhau liên quan tới việc lọc và truyền thông tin được tạo ra đến bộ mã hóa entrôpi 240 như sẽ được mô tả sau trong phần mô tả của mỗi phương pháp lọc. Thông tin liên quan đến việc lọc có thể được mã hóa bởi bộ mã hóa entrôpi 240 và được xuất ra trong dạng luồng bit.

Ảnh được xây dựng lại mà sửa được đổi sẽ được truyền đến bộ nhớ 270 có thể được sử dụng làm ảnh tham chiếu trong bộ dự đoán liên 221. Khi việc dự đoán liên được áp dụng thông qua thiết bị mã hóa, thì sự không khớp của việc dự đoán giữa thiết bị mã hóa 200 và thiết bị giải mã 300 có thể được tránh được và hiệu quả mã hóa có thể được cải thiện.

DPB của bộ nhớ 270 DPB có thể lưu trữ ảnh được xây dựng lại mà sửa được đổi để sử dụng làm ảnh tham chiếu trong bộ dự đoán liên 221. Bộ nhớ 270 có thể lưu trữ thông tin chuyển động của khối mà thông tin chuyển động trong ảnh hiện tại được dẫn xuất (hoặc được mã hóa) từ đó và/hoặc thông tin chuyển động của các khối trong ảnh hiện vừa được xây dựng lại. Thông tin chuyển động được lưu trữ có thể được truyền đến bộ dự đoán liên 221 và được sử dụng như là thông tin chuyển động của khối lân cận theo không gian hoặc thông tin chuyển động của khối lân cận theo thời gian. Bộ nhớ 270 có thể lưu trữ các mẫu được xây dựng lại của các khối được xây dựng lại trong ảnh hiện tại và có thể truyền các mẫu được xây dựng lại đến bộ dự đoán nội 222.

Fig.3 là sơ đồ sơ lược minh họa cấu hình của thiết bị giải mã hình ảnh/viđeo mà với nó (các) phương án của sáng chế có thể áp dụng.

Tham khảo đến Fig.3, thiết bị giải mã 300 có thể gồm bộ giải mã entrôpi 310, bộ xử lý phần dư 320, bộ dự đoán 330, bộ cộng 340, bộ lọc 350, và bộ nhớ 360. Bộ dự đoán 330 có thể gồm bộ dự đoán liên 331 và bộ dự đoán nội 332. Bộ xử lý phần dư 320 có thể gồm bộ khử lượng tử hóa 321 và bộ biến đổi ngược 322. Bộ giải mã entrôpi 310, bộ xử lý phần dư 320, bộ dự đoán 330, bộ cộng 340, và bộ lọc 350 có thể được tạo cấu hình bởi thành phần phần cứng (ví dụ, bộ chip hoặc bộ xử lý bộ giải mã) theo một phương án. Bên cạnh đó, bộ nhớ 360 có thể gồm bộ đệm ảnh được giải mã (DPB) hoặc có thể được tạo cấu hình bởi phương tiện lưu trữ kỹ thuật số. Thành phần phần cứng có thể còn gồm bộ nhớ 360 như là thành phần trong/ngoài.

Khi luồng bit gồm thông tin viđeo/hình ảnh được nhập vào, thì thiết bị giải mã 300 có thể xây dựng lại hình ảnh tương ứng với quy trình mà trong đó thông tin viđeo/hình ảnh được xử lý trong thiết bị mã hóa trên Fig.2. Ví dụ, thiết bị giải mã 300 có thể dẫn xuất các đơn vị/các khối dựa trên thông tin liên quan đến việc phân vùng khối thu nhận được từ luồng bit. Thiết bị giải mã 300 có thể thực hiện việc giải mã nhờ sử dụng bộ xử lý được áp dụng trong thiết bị mã hóa. Vì thế, bộ xử lý để giải mã có thể là đơn vị lập mã, ví dụ, và đơn vị lập mã có thể được phân vùng theo cấu trúc cây tứ phân, cấu trúc cây nhị phân và/hoặc cấu trúc cây tam phân từ đơn vị cây lập mã hoặc đơn vị lập mã lớn nhất. Một hoặc nhiều đơn vị biến đổi có thể được dẫn xuất từ đơn vị lập mã. Tín hiệu hình ảnh được xây dựng lại mà được giải mã và được xuất ra thông qua thiết bị giải mã 300 có thể được xây dựng lại thông qua thiết bị xây dựng lại.

Thiết bị giải mã 300 có thể nhận tín hiệu được xuất ra từ thiết bị mã hóa trên Fig.2 trong dạng luồng bit, và tín hiệu nhận được có thể được giải mã thông qua bộ giải mã entrôpi 310. Ví dụ, bộ giải mã entrôpi 310 có thể phân tích cú pháp luồng bit để dẫn xuất thông tin (ví dụ, thông tin viđeo/hình ảnh) cần thiết cho việc xây dựng lại hình ảnh (hoặc xây dựng lại ảnh). Thông tin viđeo/hình ảnh có thể còn gồm thông tin về các tập thông số khác nhau như tập thông số thích ứng (APS), tập thông số ảnh (PPS), tập thông số trình tự (SPS), hoặc tập thông số viđeo (VPS). Bên cạnh đó, thông tin viđeo/hình ảnh có thể còn gồm thông tin ràng buộc chung. Thiết bị giải mã còn có thể giải mã ảnh dựa trên thông tin về tập thông số và/hoặc thông tin ràng buộc chung. Thông tin được báo

hiệu/nhận được và/hoặc các phần tử cú pháp được mô tả sau trong sáng chế có thể được giải mã có thể giải mã thủ tục giải mã và thu nhận được từ luồng bit. Ví dụ, bộ giải mã entrôpi 310 giải mã thông tin trong luồng bit dựa trên phương pháp lập mã như là lập mã Golomb hàm mũ, CAVLC, hoặc CABAC, và xuất ra ra các phần tử cú pháp cần thiết cho việc xây dựng lại hình ảnh và các giá trị được lượng tử hóa của các hệ số biến đổi cho phần dư. Cụ thể hơn là, phương pháp giải mã entrôpi CABAC có thể nhận ngăn tương ứng với từng phần tử cú pháp trong luồng bit, xác định mô hình ngữ cảnh nhờ sử dụng thông tin phần tử cú pháp mục tiêu giải mã, thông tin giải mã của khối mục tiêu giải mã hoặc thông tin về ký hiệu/ngăn được giải mã trong giai đoạn trước đó, và thực hiện việc giải mã số học trên ngăn bằng cách dự đoán khả năng xuất hiện của ngăn theo mô hình ngữ cảnh được xác định, và tạo ra ký hiệu tương ứng với giá trị của mỗi phần tử cú pháp. Trong trường hợp này, phương pháp giải mã entrôpi CABAC có thể cập nhật mô hình ngữ cảnh bằng cách sử dụng thông tin của ký hiệu/ngăn được giải mã cho mô hình ngữ cảnh của ký hiệu/ngăn tiếp theo sau khi xác định mô hình ngữ cảnh. Thông tin liên quan đến việc dự đoán trong số thông tin được giải mã bởi bộ giải mã entrôpi 310 có thể được cung cấp cho bộ dự đoán (bộ dự đoán liên 332 và bộ dự đoán nội 331), và giá trị dư mà việc giải mã entrôpi được thực hiện trên đó trong bộ giải mã entrôpi 310, nghĩa là, các hệ số biến đổi được lượng tử hóa và thông tin thông số liên quan, có thể được nhập vào trong bộ xử lý phần dư 320. Bộ xử lý phần dư 320 có thể dẫn xuất tín hiệu dư (khối phần dư, các mẫu phần dư, mảng mẫu phần dư). Bên cạnh đó, thông tin về việc lọc trong số thông tin được giải mã bởi bộ giải mã entrôpi 310 có thể được cung cấp cho bộ lọc 350. Trong lúc đó, bộ nhận (không được thể hiện) để nhận tín hiệu được xuất ra từ thiết bị mã hóa có thể còn được tạo cấu hình làm phần tử trong/ngoài của thiết bị giải mã 300, hoặc bộ nhận có thể là thành phần của bộ giải mã entrôpi 310. Trong lúc đó, thiết bị giải mã theo sáng chế có thể được gọi là thiết bị giải mã viđeo/hình ảnh/ảnh, và thiết bị giải mã có thể được phân loại thành bộ giải mã thông tin (bộ giải mã thông tin viđeo/hình ảnh/ảnh) và bộ giải mã mẫu (bộ giải mã mẫu viđeo/hình ảnh/ảnh). Bộ giải mã thông tin có thể gồm bộ giải mã entrôpi 310, và bộ giải mã mẫu có thể gồm ít nhất một thành phần trong số bộ khử lượng tử hóa 321, bộ biến đổi ngược 322, bộ cộng 340, bộ lọc 350, bộ nhớ 360, bộ dự đoán liên 332, và bộ dự đoán nội 331.

Bộ khử lượng tử hóa 321 có thể khử lượng tử hóa các hệ số biến đổi được lượng tử hóa và xuất ra các hệ số biến đổi. Bộ khử lượng tử hóa 321 có thể sắp xếp lại các hệ số biến đổi được lượng tử hóa trong dạng khối hai chiều. Trong trường hợp này, việc sắp xếp lại có thể được thực hiện dựa trên thứ tự quét hệ số được thực hiện trong thiết bị mã hóa. Bộ khử lượng tử hóa 321 có thể thực hiện việc khử lượng tử hóa trên các hệ số biến đổi được lượng tử hóa bằng cách sử dụng thông số lượng tử hóa (ví dụ, thông tin kích cỡ bước lượng tử hóa) và thu nhận được các hệ số biến đổi.

Bộ biến đổi ngược 322 biến đổi ngược các hệ số biến đổi để thu nhận tín hiệu dư (khối phần dư, mảng mẫu phần dư).

Bộ dự đoán có thể thực hiện việc dự đoán trên khối hiện tại và tạo ra khối được dự đoán gồm các mẫu dự đoán cho khối hiện tại. Bộ dự đoán có thể xác định liệu việc dự đoán nội hay việc dự đoán liên được áp dụng cho khối hiện tại dựa trên thông tin về việc dự đoán được xuất ra từ bộ giải mã entrôpi 310 và có thể xác định chế độ dự đoán nội/liên cụ thể.

Bộ dự đoán 320 có thể tạo ra tín hiệu dự đoán dựa trên các phương pháp dự đoán khác nhau được mô tả dưới đây. Ví dụ, bộ dự đoán có thể không chỉ áp dụng việc dự đoán nội hoặc dự đoán liên để dự đoán một khối mà còn có thể áp dụng một cách đồng thời việc dự đoán nội và dự đoán liên. Việc này có thể được gọi là việc dự đoán nội và liên được kết hợp (CIIP). Bên cạnh đó, bộ dự đoán có thể dựa trên chế độ dự đoán sao chép khối nội (IBC) hoặc chế độ bảng màu cho việc dự đoán của khối. Chế độ dự đoán IBC hoặc chế độ bảng màu có thể được sử dụng cho việc mã hóa hình ảnh/viđeo nội dung của trò chơi hoặc tương tự, ví dụ, lập mã nội dung màn (SCC). IBC về cơ bản thực hiện việc dự đoán trong ảnh hiện tại mà có thể được thực hiện tương tự như việc dự đoán liên mà trong đó khối tham chiếu được dẫn xuất trong ảnh hiện tại. Nghĩa là, IBC có thể sử dụng ít nhất một kỹ thuật trong số các kỹ thuật dự đoán liên được mô tả trong sáng chế. Chế độ bảng màu có thể được coi như là, ví dụ, việc lập mã trong ảnh hoặc việc dự đoán nội. Khi chế độ bảng màu được áp dụng, thì giá trị mẫu nằm trong ảnh có thể được báo hiệu dựa trên thông tin về bảng của bảng màu và chỉ số bảng màu.

Bộ dự đoán nội 331 có thể dự đoán khối hiện tại bằng cách tham chiếu đến các mẫu trong ảnh hiện tại. Các mẫu được tham chiếu có thể được đặt trong vùng lân cận của khối hiện tại hoặc có thể được đặt tách ra theo chế độ dự đoán. Trong việc dự đoán

nội, các chế độ dự đoán có thể gồm nhiều chế độ không có hướng và nhiều chế độ có hướng. Bộ dự đoán nội 331 có thể xác định chế độ dự đoán được áp dụng cho khối hiện tại bằng cách sử dụng chế độ dự đoán được áp dụng cho khối lân cận.

Bộ dự đoán liên 332 có thể dẫn xuất khối được dự đoán cho khối hiện tại dựa trên khối tham chiếu (mảng mẫu tham chiếu) được chỉ rõ bởi vectơ chuyển động trên ảnh tham chiếu. Trong trường hợp này, để làm giảm lượng thông tin chuyển động được truyền trong chế độ dự đoán liên, thì thông tin chuyển động có thể được dự đoán trong các đơn vị của các khối, các khối con, hoặc các mẫu dựa trên sự tương liên của thông tin chuyển động giữa khối lân cận và khối hiện tại. Thông tin chuyển động có thể gồm vectơ chuyển động và chỉ số ảnh tham chiếu. Thông tin chuyển động có thể còn gồm thông tin hướng dự đoán liên (việc dự đoán L0, việc dự đoán L1, việc dự đoán Bi, v.v.). Trong trường hợp của việc dự đoán liên, khối lân cận có thể gồm khối lân cận theo không gian có mặt trong ảnh hiện tại và khối lân cận theo thời gian có mặt trong ảnh tham chiếu. Ví dụ, bộ dự đoán liên 332 có thể tạo cấu hình danh sách ứng viên thông tin chuyển động dựa trên các khối lân cận và dẫn xuất vectơ chuyển động của khối hiện tại và/hoặc chỉ số ảnh tham chiếu dựa trên thông tin chọn ứng viên nhận được. Việc dự đoán liên có thể được thực hiện dựa trên các chế độ dự đoán khác nhau, và thông tin về việc dự đoán có thể gồm thông tin chỉ ra chế độ dự đoán liên cho khối hiện tại.

Bộ cộng 340 có thể tạo ra tín hiệu được xây dựng lại (ảnh được xây dựng lại, khối được xây dựng lại, mảng mẫu được xây dựng lại) bằng cách cộng tín hiệu dư thu nhận được vào tín hiệu dự đoán (khối được dự đoán, mảng mẫu được dự đoán) được xuất ra từ bộ dự đoán (gồm bộ dự đoán liên 332 và/hoặc bộ dự đoán nội 331). Nếu không có phần dư cho khối cần được xử lý, như khi chế độ bỏ qua được áp dụng, thì khối được dự đoán có thể được sử dụng làm khối được xây dựng lại.

Bộ cộng 340 có thể được gọi là bộ xây dựng lại hoặc bộ tạo ra khối được xây dựng lại. Tín hiệu được xây dựng lại mà được tạo ra có thể được sử dụng cho việc dự đoán nội của khối tiếp theo cần được xử lý trong hình ảnh hiện tại, có thể được xuất ra qua việc lọc như được mô tả dưới đây, hoặc có thể được sử dụng cho việc dự đoán liên của ảnh tiếp theo.

Trong khi đó, việc ánh xạ độ chói với việc định cỡ sắc độ (LMCS) có thể được áp dụng trong quy trình giải mã ảnh.

Bộ lọc 350 có thể cải thiện chất lượng hình ảnh chủ quan/khách quan bằng cách áp dụng việc lọc cho tín hiệu được xây dựng lại. Ví dụ, bộ lọc 350 có thể tạo ra ảnh được xây dựng lại mà được sửa đổi bằng cách áp dụng các phương pháp lọc khác nhau cho ảnh được xây dựng lại và lưu trữ ảnh được xây dựng lại mà sửa được đổi trong bộ nhớ 360, cụ thể là, DPB của bộ nhớ 360. Các phương pháp lọc khác nhau có thể gồm, ví dụ, lọc khử khối, dịch chuyển thích ứng mẫu, bộ lọc vòng lặp thích ứng, bộ lọc song phương, và tương tự.

Ảnh được xây dựng lại (được sửa đổi) mà được lưu trữ trong DPB của bộ nhớ 360 có thể được sử dụng làm ảnh tham chiếu trong bộ dự đoán liên 332. Bộ nhớ 360 có thể lưu trữ thông tin chuyển động của khối mà thông tin chuyển động trong ảnh hiện tại được dẫn xuất (hoặc được giải mã) từ đó và/hoặc thông tin chuyển động của các khối trong ảnh hiện đã được xây dựng lại. Thông tin chuyển động được lưu trữ có thể được truyền đến bộ dự đoán liên 260 để được tận dụng làm thông tin chuyển động của khối lân cận theo không gian hoặc thông tin chuyển động của khối lân cận theo thời gian. Bộ nhớ 360 có thể lưu trữ các mẫu được xây dựng lại của các khối được xây dựng lại trong ảnh hiện tại và truyền các mẫu được xây dựng lại đến bộ dự đoán nội 331.

Trong sáng chế, các phương án được mô tả trên bộ lọc 260, bộ dự đoán liên 221, và bộ dự đoán nội 222 của thiết bị mã hóa 200 có thể được áp dụng một cách đồng đều hoặc một cách tương ứng với bộ lọc 350, bộ dự đoán liên 332, và bộ dự đoán nội 331 của thiết bị giải mã 300. Điều tương tự cũng có thể được áp dụng cho đơn vị 332 và bộ dự đoán nội 331.

Trong sáng chế, ít nhất một hoạt động trong số việc lượng tử hóa/việc lượng tử hóa ngược và/hoặc việc biến đổi/việc biến đổi ngược có thể được lược bỏ. Khi việc lượng tử hóa/việc lượng tử hóa ngược được lược bỏ, thì các hệ số biến đổi được lượng tử hóa có thể được gọi là các hệ số biến đổi. Khi việc biến đổi/biến đổi ngược được lược bỏ, thì các hệ số biến đổi có thể được gọi là các hệ số hoặc các hệ số dư, hoặc có thể vẫn được gọi là các hệ số biến đổi để trình bày đồng nhất.

Trong sáng chế, hệ số biến đổi được lượng tử hóa và hệ số biến đổi có thể được gọi là hệ số biến đổi và hệ số biến đổi được định cỡ, một cách tương ứng. Trong trường hợp này, thông tin phần dư có thể gồm thông tin về hệ số biến đổi (các hệ số biến đổi), và thông tin về hệ số biến đổi (các hệ số biến đổi) có thể được báo hiệu thông qua cú

pháp lập mã phần dư. Các hệ số biến đổi có thể được dẫn xuất dựa trên thông tin phần dư (hoặc thông tin về hệ số biến đổi (các hệ số biến đổi)), và các hệ số biến đổi được định cỡ có thể được dẫn xuất bởi việc biến đổi ngược (định cỡ) trên các hệ số biến đổi. Các mẫu phần dư có thể được dẫn xuất dựa trên việc biến đổi ngược (biến đổi) của các hệ số biến đổi được định cỡ. Điều này cũng có thể được áp dụng/được trình bày trong các phần khác của sáng chế.

Như được mô tả ở trên, thiết bị mã hóa có thể thực hiện các phương pháp mã hóa khác nhau như là Golomb hàm số mũ, lập mã chiều dài biến đổi thích ứng ngữ cảnh (CAVLC), và lập mã số học nhị phân thích ứng ngữ cảnh (CABAC). Ngoài ra, thiết bị giải mã có thể giải mã thông tin trong luồng bit dựa trên phương pháp lập mã như là lập mã Golomb hàm mũ, CAVLC hoặc CABAC, và xuất ra giá trị của phần tử cú pháp cần thiết cho việc xây dựng lại hình ảnh và các giá trị được lượng tử hóa của các hệ số biến đổi liên quan đến các phần dư.

Ví dụ, các phương pháp lập mã được mô tả ở trên có thể được thực hiện như được mô tả sau đây.

Fig.4 thể hiện ví dụ lập mã số học nhị phân thích ứng ngữ cảnh (CABAC) cho việc mã hóa phần tử cú pháp. Ví dụ, trong quy trình mã hóa CABAC, khi tín hiệu đầu vào là phần tử cú pháp, thay vì giá trị nhị phân, thiết bị mã hóa có thể chuyển đổi tín hiệu đầu vào thành giá trị nhị phân bằng việc nhị phân hóa giá trị của tín hiệu đầu vào. Ngoài ra, khi tín hiệu đầu vào đã là giá trị nhị phân (nghĩa là, khi giá trị của tín hiệu đầu vào là giá trị nhị phân), việc nhị phân hóa có thể không được thực hiện và có thể được đi vòng qua. Ở đây, mỗi số nhị phân 0 hoặc 1 cấu thành giá trị nhị phân có thể được gọi là ngăn (bin). Ví dụ, nếu chuỗi nhị phân sau việc nhị phân hóa là 110, mỗi trong số 1, 1, và 0 được gọi một ngăn. (Các) ngăn cho một phần tử cú pháp có thể chỉ ra giá trị của phần tử cú pháp.

Sau đó, các ngăn được nhị phân hóa của các phần tử cú pháp có thể được nhập vào công cụ lập mã thông thường hoặc công cụ lập mã đi vòng qua. Công cụ lập mã thông thường của thiết bị mã hóa có thể phân phối mô hình ngữ cảnh phản ánh giá trị xác suất tới ngăn tương ứng, và có thể mã hóa ngăn tương ứng dựa trên mô hình ngữ cảnh được phân phối. Công cụ lập mã thông thường của thiết bị mã hóa có thể cập nhật mô hình ngữ cảnh cho mỗi ngăn sau khi thực hiện việc mã hóa trên mỗi ngăn. Ngăn

được mã hóa như được mô tả ở trên có thể được gọi là ngăn được lập mã ngữ cảnh.

Cùng lúc đó, khi các ngăn được nhị phân hóa của các phần tử cú pháp được nhập vào công cụ lập mã đi vòng qua, chúng có thể được lập mã như sau. Ví dụ, công cụ lập mã đi vòng qua của thiết bị mã hóa lược bỏ thủ tục ước tính xác suất đối với ngăn đầu vào và thủ tục cập nhật mô hình xác suất được áp dụng cho ngăn sau khi mã hóa. Khi việc mã hóa đi vòng qua được áp dụng, thiết bị mã hóa có thể mã hóa ngăn đầu vào bằng việc áp dụng sự phân bố xác suất đồng nhất thay vì phân phối mô hình ngữ cảnh, nhờ đó cải thiện tốc độ mã hóa. Ngăn được mã hóa như được mô tả ở trên có thể được gọi là ngăn đi vòng qua.

Việc giải mã entrôpi có thể biểu diễn quy trình thực hiện cùng quy trình như việc mã hóa entrôpi được mô tả ở trên theo thứ tự đảo ngược.

Ví dụ, khi phần tử cú pháp được giải mã dựa trên mô hình ngữ cảnh, thiết bị giải mã có thể nhận ngăn tương ứng với phần tử cú pháp thông qua luồng bit, xác định mô hình ngữ cảnh sử dụng phần tử cú pháp và thông tin giải mã của khối mục tiêu giải mã hoặc khối lân cận hoặc thông tin của ký hiệu/ngăn được giải mã trong giai đoạn trước đó, dự đoán xác suất xảy ra của ngăn nhận được theo mô hình ngữ cảnh được xác định, và thực hiện việc giải mã số học trên ngăn để dẫn xuất giá trị của phần tử cú pháp. Sau đó, mô hình ngữ cảnh của ngăn mà được giải mã tiếp theo có thể được cập nhật với mô hình ngữ cảnh được xác định.

Ngoài ra, ví dụ, khi phần tử cú pháp được giải mã đi vòng qua, thiết bị giải mã có thể nhận ngăn tương ứng với phần tử cú pháp thông qua luồng bit, và giải mã ngăn đầu vào bằng việc áp dụng sự phân bố xác suất đồng nhất. Trong trường hợp này, thủ tục của thiết bị giải mã để dẫn xuất mô hình ngữ cảnh của các phần tử cú pháp và thủ tục để cập nhật mô hình ngữ cảnh được áp dụng cho ngăn sau khi giải mã có thể được lược bỏ.

Như được mô tả ở trên, các mẫu phần dư có thể được dẫn xuất làm các hệ số biến đổi được lượng tử hóa thông qua các quy trình biến đổi và lượng tử hóa. Các hệ số biến đổi được lượng tử hóa cũng có thể được gọi là các hệ số biến đổi. Trong trường hợp này, các hệ số biến đổi trong khối có thể được báo hiệu trong dạng của thông tin phần dư. Thông tin phần dư có thể gồm cú pháp lập mã phần dư. Nghĩa là, thiết bị mã

hóa có thể tạo cấu hình cú pháp lập mã phần dư với thông tin phần dư, mã hóa cú pháp lập mã phần dư, và xuất ra nó dưới dạng luồng bit, và thiết bị giải mã có thể giải mã cú pháp lập mã phần dư từ luồng bit và dẫn xuất các hệ số biến đổi (được lượng tử hóa) phần dư. Cú pháp lập mã phần dư có thể gồm các phần tử cú pháp trình bày xem việc biến đổi đã được áp dụng cho khối tương ứng hay chưa, vị trí của hệ số biến đổi hiệu quả cuối cùng trong khối, xem hệ số biến đổi hiệu quả có tồn tại trong khối con hay không, kích thước/dấu của hệ số biến đổi hiệu quả, và tương tự, như sẽ được mô tả sau đây.

Ví dụ, các phần tử cú pháp liên quan tới việc mã hóa/giải mã dữ liệu phần dư có thể được trình bày như được thể hiện trong bảng sau đây.

[Bảng 1]

transform_unit(x0, y0, tbWidth, tbHeight, treeType, subTuIndex, chType) {	**Descriptor**
if(IntraSubPartitionsSplitType != ISP_NO_SPLIT && treeType == SINGLE_TREE && subTuIndex == NumIntraSubPartitions − 1) {	
xC = CbPosX[chType][x0][y0]	
yC = CbPosY[chType][x0][y0]	
wC = CbWidth[chType][x0][y0] / SubWidthC	
hC = CbHeight[chType][x0][y0] / SubHeightC	
} else {	
xC = x0	
yC = y0	
wC = tbWidth / SubWidthC	
hC = tbHeight / SubHeightC	
}	
chromaAvailable = treeType != DUAL_TREE_LUMA && sps_chroma_format_idc != 0 && (IntraSubPartitionsSplitType == ISP_NO_SPLIT || (IntraSubPartitionsSplitType != ISP_NO_SPLIT && subTuIndex == NumIntraSubPartitions − 1))	
if((treeType == SINGLE_TREE || treeType == DUAL_TREE_CHROMA) && sps_chroma_format_idc != 0 && ((IntraSubPartitionsSplitType == ISP_NO_SPLIT && !(cu_sbt_flag && ((subTuIndex == 0 && cu_sbt_pos_flag) || (subTuIndex == 1 && !cu_sbt_pos_flag)))) || (IntraSubPartitionsSplitType != ISP_NO_SPLIT && (subTuIndex == NumIntraSubPartitions − 1)))) {	
tu_cb_coded_flag[xC][yC]	ae(v)
tu_cr_coded_flag[xC][yC]	ae(v)
}	

if(treeType == SINGLE_TREE || treeType == DUAL_TREE_LUMA) {	
if((IntraSubPartitionsSplitType == ISP_NO_SPLIT && !(cu_sbt_flag && ((subTuIndex == 0 && cu_sbt_pos_flag) || (subTuIndex == 1 && !cu_sbt_pos_flag))) && ((CuPredMode[chType][x0][y0] == MODE_INTRA && !cu_act_enabled_flag[x0][y0]) || (chromaAvailable && (tu_cb_coded_flag[xC][yC] || tu_cr_coded_flag[xC][yC])) || CbWidth[chType][x0][y0] > MaxTbSizeY || CbHeight[chType][x0][y0] > MaxTbSizeY)) || (IntraSubPartitionsSplitType != ISP_NO_SPLIT && (subTuIndex < NumIntraSubPartitions − 1 || !InferTuCbfLuma)))	
tu_y_coded_flag[x0][y0]	ae(v)
if(IntraSubPartitionsSplitType != ISP_NO_SPLIT)	
InferTuCbfLuma = InferTuCbfLuma && !tu_y_coded_flag[x0][y0]	
}	
if((CbWidth[chType][x0][y0] > 64 || CbHeight[chType][x0][y0] > 64 || tu_y_coded_flag[x0][y0] || (chromaAvailable && (tu_cb_coded_flag[xC][yC] || tu_cr_coded_flag[xC][yC])) && treeType != DUAL_TREE_CHROMA && pps_cu_qp_delta_enabled_flag && !IsCuQpDeltaCoded) {	
cu_qp_delta_abs	ae(v)
if(cu_qp_delta_abs)	
cu_qp_delta_sign_flag	ae(v)
}	
if((CbWidth[chType][x0][y0] > 64 || CbHeight[chType][x0][y0] > 64 || (chromaAvailable && (tu_cb_coded_flag[xC][yC] || tu_cr_coded_flag[xC][yC]))) && treeType != DUAL_TREE_LUMA && sh_cu_chroma_qp_offset_enabled_flag && !IsCuChromaQpOffsetCoded) {	
cu_chroma_qp_offset_flag	ae(v)
if(cu_chroma_qp_offset_flag && pps_chroma_qp_offset_list_len_minus1 > 0)	
cu_chroma_qp_offset_idx	ae(v)
}	

if(sps_joint_cbcr_enabled_flag && ((CuPredMode[chType][x0][y0] == MODE_INTRA && (tu_cb_coded_flag[xC][yC] || tu_cr_coded_flag[xC][yC])) || (tu_cb_coded_flag[xC][yC] && tu_cr_coded_flag[xC][yC])) && chromaAvailable)	
tu_joint_cbcr_residual_flag[xC][yC]	ae(v)
if(tu_y_coded_flag[x0][y0] && treeType != DUAL_TREE_CHROMA) {	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0][0] && tbWidth <= MaxTsSize && tbHeight <= MaxTsSize && (IntraSubPartitionsSplitType == ISP_NO_SPLIT) && !cu_sbt_flag)	
transform_skip_flag[x0][y0][0]	ae(v)
if(!transform_skip_flag[x0][y0][0] || sh_ts_residual_coding_disabled_flag)	
residual_coding(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
else	
residual_ts_coding(x0, y0, Log2(tbWidth), Log2(tbHeight), 0)	
}	
if(tu_cb_coded_flag[xC][yC] && treeType != DUAL_TREE_LUMA) {	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0][1] && wC <= MaxTsSize && hC <= MaxTsSize && !cu_sbt_flag)	
transform_skip_flag[xC][yC][1]	ae(v)
if(!transform_skip_flag[xC][yC][1] || sh_ts_residual_coding_disabled_flag)	
residual_coding(xC, yC, Log2(wC), Log2(hC), 1)	
else	
residual_ts_coding(xC, yC, Log2(wC), Log2(hC), 1)	
}	
if(tu_cr_coded_flag[xC][yC] && treeType != DUAL_TREE_LUMA && !(tu_cb_coded_flag[xC][yC] && tu_joint_cbcr_residual_flag[xC][yC])) {	
if(sps_transform_skip_enabled_flag && !BdpcmFlag[x0][y0][2] && wC <= MaxTsSize && hC <= MaxTsSize && !cu_sbt_flag)	
transform_skip_flag[xC][yC][2]	ae(v)
if(!transform_skip_flag[xC][yC][2] || sh_ts_residual_coding_disabled_flag)	
residual_coding(xC, yC, Log2(wC), Log2(hC), 2)	
else	
residual_ts_coding(xC, yC, Log2(wC), Log2(hC), 2)	
}	
}	

transform_skip_flag chỉ ra xem việc biến đổi có được bỏ qua trong khối được kết hợp hay không. transform_skip_flag có thể là phần tử cú pháp của cờ bỏ qua biến đổi. Khối được kết hợp có thể là khối lập mã (coding block, CB) hoặc khối biến đổi (transform block, TB). Liên quan đến các thủ tục biến đổi (và lượng tử hóa) và lập mã phần dư, CB và TB có thể được sử dụng thay thế cho nhau. Ví dụ, như được mô tả ở trên, các mẫu phần dư có thể được dẫn xuất cho CB, và các hệ số biến đổi (được lượng tử hóa) có thể được dẫn xuất thông qua biến đổi và lượng tử hóa cho các mẫu phần dư,

và thông qua thủ tục lập mã phần dư, thông tin (ví dụ, các phần tử cú pháp) chỉ ra một cách hiệu quả vị trí, độ lớn, dấu, v.v. của các hệ số biến đổi (được lượng tử hóa) có thể được tạo ra và được báo hiệu. Các hệ số biến đổi được lượng tử hóa có thể được gọi đơn giản là các hệ số biến đổi. Nói chung, khi CB là không lớn hơn TB tối đa, kích thước của CB có thể là giống như kích thước của TB, và trong trường hợp này, khối mục tiêu cần được biến đổi (và được lượng tử hóa) và phần dư được lập mã có thể được gọi là CB hoặc TB. Cùng lúc đó, khi CB là lớn hơn TB tối đa, khối mục tiêu cần được biến đổi (và được lượng tử hóa) và phần dư được lập mã có thể được gọi là TB. Sau đây, sẽ được mô tả rằng các phần tử cú pháp liên quan đến việc lập mã phần dư được báo hiệu trong các đơn vị của các khối biến đổi (TB) nhưng đây là ví dụ và TB có thể được sử dụng thay thế lẫn nhau với các khối lập mã (các CB như được mô tả ở trên.

Cùng lúc đó, các phần tử cú pháp mà được báo hiệu sau cờ bỏ qua biến đổi được báo hiệu có thể là giống như các phần tử cú pháp được bộc lộ trong Bảng 2 và/hoặc Bảng 3 dưới đây, và các phần mô tả chi tiết về các phần tử cú pháp được mô tả dưới đây.

[Bảng 2]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth == 5 && log2TbHeight < 6)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth < 6 && log2TbHeight == 5)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	

do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock−−	
}	
lastScanPos−−	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth == 2 || log2TbWidth == 3) && log2TbWidth == log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((lastSubBlock > 0 || lastScanPos > 0) && cIdx == 0)	
MtsDcOnly = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i−−) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
inferSbDcSigCoeffFlag = 0	
if(i < lastSubBlock && i > 0) {	
sb_coded_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	

if(sb_coded_flag[xS][yS] && (xS > 3 || yS > 3) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = firstPosMode0	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(sb_coded_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != Last SignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	

AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag [n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n] [1]	
if(sh_dep_quant_used_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
firstPosMode1 = n − 1	
}	
for(n = firstPosMode0; n > firstPosMode1; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n] [0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n] [1]	
if(abs_level_gtx_flag[n][1])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder[n]	
}	
for(n = firstPosMode1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n] [0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n] [1]	
if(sb_coded_flag[xS][yS])	
dec_abs_level[n]	ae(v)
if(AbsLevel[xC][yC] > 0) {	
if(lastSigScanPosSb = = −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
if(sh_dep_quant_used_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(sh_dep_quant_used_flag || !sh_sign_data_hiding_used_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	

for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(sh_dep_quant_used_flag) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
} else {	
sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	

if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

[Bảng 3]

residual_ts_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	
inferSbCbf = 1	
RemCcbs = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
for(i =0; i <= lastSubBlock; i++) {	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH][i][1]	
if(i != lastSubBlock || !inferSbCbf)	
sb_coded_flag[xS][yS]	ae(v)
if(sb_coded_flag[xS][yS] && i < lastSubBlock)	
inferSbCbf = 0	
/* First scan pass */	
inferSbSigCoeffFlag = 1	
lastScanPosPass1 = −1	
for(n = 0; n <= numSbCoeff − 1 && RemCcbs >= 4; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
lastScanPosPass1 = n	

if(sb_coded_flag[xS][yS] && (n != numSbCoeff − 1 || !inferSbSigCoeffFlag)) {	
sig_coeff_flag[xC][yC]	ae(v)
RemCcbs− −	
if(sig_coeff_flag[xC][yC])	
inferSbSigCoeffFlag = 0	
}	
CoeffSignLevel[xC][yC] = 0	
if(sig_coeff_flag[xC][yC]) {	
coeff_sign_flag[n]	ae(v)
RemCcbs− −	
CoeffSignLevel[xC][yC] = (coeff_sign_flag[n] > 0 ? −1 : 1)	
abs_level_gtx_flag[n][0]	ae(v)
RemCcbs− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
RemCcbs− −	
}	
}	
AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0]	
}	
/* Greater than X scan pass (numGtXFlags=5) */	
lastScanPosPass2 = −1	
for(n = 0; n <= numSbCoeff − 1 && RemCcbs >= 4; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
AbsLevelPass2[xC][yC] = AbsLevelPass1[xC][yC]	
for(j = 1; j < 5; j++) {	
if(abs_level_gtx_flag[n][j − 1]) {	
abs_level_gtx_flag[n][j]	ae(v)
RemCcbs− −	
}	
AbsLevelPass2[xC][yC] += 2 * abs_level_gtx_flag[n][j]	
}	
lastScanPosPass2 = n	
}	

/* remainder scan pass */	
for(n = 0; n <= numSbCoeff − 1; n++) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((n <= lastScanPosPass2 && AbsLevelPass2[xC][yC] >= 10) || (n > lastScanPosPass2 && n <= lastScanPosPass1 && AbsLevelPass1[xC][yC] >= 2) || (n > lastScanPosPass1 && sb_coded_flag[xS][yS]))	
abs_remainder[n]	ae(v)
if(n <= lastScanPosPass2)	
AbsLevel[xC][yC] = AbsLevelPass2[xC][yC] + 2 * abs_remainder[n]	
else if(n <= lastScanPosPass1)	
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] + 2 * abs_remainder[n]	
else { /* bypass */	
AbsLevel[xC][yC] = abs_remainder[n]	
if(abs_remainder[n])	
coeff_sign_flag[n]	ae(v)
}	
if(BdpcmFlag[x0][y0][cIdx] == 0 && n <= lastScanPosPass1) {	
absLeftCoeff = xC > 0 ? AbsLevel[xC − 1][yC]) : 0	
absAboveCoeff = yC > 0 ? AbsLevel[xC][yC − 1]) : 0	
predCoeff = Max(absLeftCoeff, absAboveCoeff)	
if(AbsLevel[xC][yC] == 1 && predCoeff > 0)	
AbsLevel[xC][yC] = predCoeff	
else if(AbsLevel[xC][yC] > 0 && AbsLevel[xC][yC] <= predCoeff)	
AbsLevel[xC][yC]− −	
}	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (1 − 2 * coeff_sign_flag[n]) * AbsLevel[xC][yC]	
}	
}	
}	

Theo phương án này, như được thể hiện trong Bảng 1, việc lập mã phần dư có thể được chia theo giá trị của phần tử cú pháp transform_skip_flag của cờ bỏ qua biến đổi. Nghĩa là, phần tử cú pháp khác có thể được sử dụng cho việc lập mã phần dư dựa trên giá trị của cờ bỏ qua biến đổi (dựa trên liệu việc biến đổi có được bỏ qua hay không). Việc lập mã phần dư được sử dụng khi việc bỏ qua biến đổi không được áp dụng (tức là, khi việc biến đổi được áp dụng) có thể được gọi là lập mã phần dư thông thường (regular residual coding, RRC), và việc lập mã phần dư được sử dụng khi việc bỏ qua biến đổi được áp dụng (tức là, khi việc biến đổi không được áp dụng) có thể được gọi là lập mã phần dư bỏ qua biến đổi (TSRC). Ngoài ra, việc lập mã phần dư thông thường

có thể được gọi là việc lập mã phần dư chung. Ngoài ra, việc lập mã phần dư thông thường có thể được gọi là cấu trúc cú pháp lập mã phần dư thông thường, và việc bỏ qua biến đổi việc lập mã phần dư có thể được gọi là cấu trúc cú pháp lập mã phần dư bỏ qua biến đổi. Bảng 2 ở trên có thể thể hiện phần tử cú pháp của việc lập mã phần dư khi giá trị của transform_skip_flag là 0, tức là, khi việc biến đổi được áp dụng, và Bảng 3 ở trên có thể thể hiện phần tử cú pháp của việc lập mã phần dư khi giá trị của transform_skip_flag là 1, tức là, khi việc biến đổi không được áp dụng.

Cụ thể là, ví dụ, cờ bỏ qua biến đổi chỉ ra xem có bỏ qua việc biến đổi của khối biến đổi hay không có thể được phân tích cú pháp, và việc liệu cờ bỏ qua biến đổi có phải là 1 hay không có thể được xác định. Nếu giá trị của cờ bỏ qua biến đổi là 0, như được thể hiện trong Bảng 2, các phần tử cú pháp last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, sb_coded_flag, sig_coeff_flag, abs_level_gtx_flag, par_level_flag, abs_remainder, coeff_sign_flag và/hoặc dec_abs_level cho hệ số phần dư của khối biến đổi có thể được phân tích cú pháp, và hệ số phần dư có thể được dẫn xuất dựa trên các phần tử cú pháp. Trong trường hợp này, các phần tử cú pháp có thể được phân tích cú pháp tuần tự, và thứ tự phân tích cú pháp có thể được thay đổi. Ngoài ra, abs_level_gtx_flag có thể thể hiện abs_level_gt1_flag và/hoặc abs_level_gt3_flag. Ví dụ, abs_level_gtx_flag[n][0] có thể là ví dụ của cờ mức hệ số biến đổi thứ nhất (abs_level_gt1_flag), và abs_level_gtx_flag[n][1] có thể là ví dụ của cờ mức hệ số biến đổi thứ hai (abs_level_gt3_flag).

Tham khảo Bảng 2 ở trên, last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, sb_coded_flag, sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag, abs_remainder, coeff_sign_flag, và/hoặc dec_abs_level có thể được mã hóa/được giải mã. Cùng lúc đó, sb_coded_flag có thể được biểu diễn là coded_sub_block_flag.

Trong một phương án, thiết bị mã hóa có thể mã hóa (x, y) thông tin vị trí của hệ số biến đổi khác không cuối cùng trong khối biến đổi dựa trên các phần tử cú pháp last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Cụ thể hơn, last_sig_coeff_x_prefix biểu diễn tiền tố của vị trí cột của hệ số đáng kể cuối cùng theo thứ tự quét trong khối biến đổi,

last_sig_coeff_y_prefix biểu diễn tiền tố của vị trí hàng của hệ số đáng kể cuối cùng theo thứ tự quét trong khối biến đổi, last_sig_coeff_x_suffix biểu diễn hậu tố của vị trí cột của hệ số đáng kể cuối cùng theo thứ tự quét trong khối biến đổi, và last_sig_coeff_y_suffix biểu diễn hậu tố của vị trí hàng của hệ số đáng kể cuối cùng theo thứ tự quét trong khối biến đổi. Ở đây, hệ số đáng kể có thể biểu diễn hệ số khác không. Ngoài ra, thứ tự quét có thể là thứ tự quét chéo bên phải. Mặt khác, thứ tự quét có thể là thứ tự quét ngang hoặc thứ tự quét dọc. Thứ tự quét có thể được xác định dựa trên việc liệu việc dự đoán liên/nội có được áp dụng cho khối mục tiêu (CB hoặc CB gồm TB) và/hoặc chế độ dự đoán liên/nội cụ thể hay không.

Sau đó, thiết bị mã hóa có thể chia khối biến đổi thành các khối con 4x4, và sau đó chỉ ra xem có hệ số khác không trong khối con hiện tại hay không sử dụng phần tử cú pháp 1 bit coded_sub_block_flag cho mỗi khối con 4x4.

Nếu giá trị của coded_sub_block_flag là 0, không còn thông tin cần được truyền, và do đó, thiết bị mã hóa có thể kết thúc quy trình mã hóa trên khối con hiện tại. Ngược lại, nếu giá trị của coded_sub_block_flag là 1, thiết bị mã hóa có thể tiếp tục thực hiện quy trình mã hóa trên sig_coeff_flag. Do khối con gồm hệ số khác không cuối cùng không yêu cầu việc mã hóa cho coded_sub_block_flag và khối con gồm thông tin DC của khối biến đổi có xác suất cao gồm hệ số khác không, coded_sub_block_flag có thể không được lập mã và giá trị của nó có thể được giả định là 1.

Nếu giá trị của coded_sub_block_flag là 1 và do đó nó được xác định rằng hệ số khác không tồn tại trong khối con hiện tại, thiết bị mã hóa có thể mã hóa sig_coeff_flag có giá trị nhị phân theo thứ tự quét đảo ngược. Thiết bị mã hóa có thể mã hóa phần tử cú pháp 1 bit sig_coeff_flag cho mỗi hệ số biến đổi theo thứ tự quét. Nếu giá trị của hệ số biến đổi ở vị trí quét hiện tại không phải là 0, giá trị của sig_coeff_flag có thể là 1. Ở đây, trong trường hợp của khối con gồm hệ số khác không cuối cùng, sig_coeff_flag không cần phải được mã hóa cho hệ số khác không cuối cùng, nên quy trình lập mã cho khối con có thể được lược bỏ. Việc lập mã thông tin mức có thể được thực hiện chỉ khi sig_coeff_flag là 1, và bốn phần tử cú pháp có thể được sử dụng trong quy trình mã hóa thông tin mức. Cụ thể hơn, mỗi sig_coeff_flag[xC][yC] có thể chỉ ra xem mức (giá trị) của hệ số biến đổi tương ứng tại mỗi vị trí hệ số biến đổi (xC, yC) trong TB hiện tại có khác không hay không. Trong một phương án, sig_coeff_flag có thể tương ứng với ví

dụ của phần tử cú pháp của cờ hệ số đáng kể chỉ ra xem hệ số biến đổi được lượng tử hóa có phải là hệ số đáng kể khác không hay không.

Giá trị mức còn lại sau việc mã hóa cho sig_coeff_flag có thể được dẫn xuất như được thể hiện trong phương trình sau đây. Nghĩa là, phần tử cú pháp remAbsLevel chỉ ra giá trị mức cần được mã hóa có thể được dẫn xuất từ phương trình sau đây.

[Phương trình 1]

$$remAbsLevel = |coeff| - 1$$

Ở đây, coeff có nghĩa là giá trị hệ số biến đổi thực.

Ngoài ra, abs_level_gt1_flag có thể chỉ ra xem remAbsLevel' của vị trí quét tương ứng (n) có lớn hơn 1 hay không. Ví dụ, khi giá trị của abs_level_gt1_flag là 0, giá trị tuyệt đối của hệ số biến đổi của vị trí tương ứng có thể là 1. Ngoài ra, khi giá trị của abs_level_gt1_flag là 1, remAbsLevel chỉ ra giá trị mức cần được mã hóa sau có thể được cập nhật như được thể hiện trong phương trình sau đây.

[Phương trình 2]

$$remAbsLevel = remAbsLevel - 1$$

Ngoài ra, giá trị hệ số ít đáng kể nhất (least significant coefficient, LSB) của remAbsLevel được mô tả trong Phương trình 2 được mô tả ở trên có thể được mã hóa như trong Phương trình 3 dưới đây thông qua par_level_flag.

[Phương trình 3]

$$par\_level\_flag = |coeff| \,\&\, 1$$

Ở đây, par_level_flag[n] có thể chỉ ra tính chẵn lẻ của mức hệ số biến đổi (giá trị) tại vị trí quét n.

Giá trị mức hệ số biến đổi remAbsLevel mà cần được mã hóa sau khi thực hiện việc mã hóa par_level_flag có thể được cập nhật như được thể hiện dưới đây trong phương trình sau đây.

[Phương trình 4]

$$remAbsLevel = remAbsLevel >> 1$$

abs_level_gt3_flag có thể chỉ ra xem remAbsLevel' của vị trí quét tương ứng (n) có lớn hơn 3 hay không. Việc mã hóa cho abs_remainder có thể được thực hiện chỉ trong trường hợp trong đó rem_abs_gt3_flag là bằng 1. Mối quan hệ giữa giá trị hệ số biến đổi thực coeff và mỗi phần tử cú pháp có thể là như được thể hiện dưới đây trong phương trình sau đây.

[Phương trình 5]

$$|coeff| = sig\_coeff\_flag + abs\_level\_gt1\_flag + par\_level\_flag + 2 * (abs\_level\_gt3\_flag + abs\_remainder)$$

Ngoài ra, bảng sau đây chỉ ra các ví dụ liên quan đến Phương trình 5 được mô tả ở trên.

[Bảng 4]

|coeff[n]|	sig_coeff_flag[n]	abs_level_gtX_flag[n][0]	par_level_flag[n]	abs_level_gtX_flag[n][1]	abs_remainder[n]
0	0				
1	1	0			
2	1	1	0	0	
3	1	1	1	0	
4	1	1	0	1	0
5	1	1	1	1	0
6	1	1	0	1	1
7	1	1	1	1	1
8	1	1	0	1	2
9	1	1	1	1	2
10	1	1	0	1	3
11	1	1	1	1	3
...	...	...	...		

Ở đây, | coeff | chỉ ra mức hệ số biến đổi (giá trị) và có thể cũng chỉ ra dưới dạng AbsLevel cho hệ số biến đổi. Ngoài ra, dấu của mỗi hệ số có thể được mã hóa bằng việc sử dụng coeff_sign_flag, mà là ký hiệu 1 bit.

Ngoài ra, nếu giá trị của cờ bỏ qua biến đổi là 1, như được thể hiện trong Bảng 3, các phần tử cú pháp sb_coded_flag, sig_coeff_flag, coeff_sign_flag, abs_level_gtx_flag, par_level_flag và/hoặc abs_remainder cho hệ số phần dư của khối biến đổi có thể được phân tích cú pháp, và hệ số phần dư có thể được dẫn xuất dựa trên các phần tử cú pháp. Trong trường hợp này, các phần tử cú pháp có thể được phân tích cú pháp tuần tự, và thứ tự phân tích cú pháp có thể được thay đổi. Ngoài ra, abs_level_gtx_flag có thể thể hiện abs_level_gt1_flag, abs_level_gt3_flag, abs_level_gt5_flag, abs_level_gt7_flag, và/hoặc abs_level_gt9_flag. Ví dụ, abs_level_gtx_flag[n][j] có thể là cờ chỉ ra xem giá trị tuyệt đối hoặc mức (giá trị) của hệ số biến đổi tại vị trí quét n có lớn hơn (j<<1)+1 hay không. Điều kiện (j<<1)+1 có

thể được thay thế tùy chọn với ngưỡng cụ thể chẳng hạn như ngưỡng thứ nhất, ngưỡng thứ hai, hoặc tương tự.

Cùng lúc đó, CABAC cung cấp hiệu quả cao, nhưng nhược điểm có hiệu quả thông lượng kém. Điều này được gây ra bởi công cụ lập mã thông thường của CABAC. Việc mã hóa thông thường (nghĩa là, mã hóa thông qua công cụ lập mã thông thường của CABAC) thể hiện sự phụ thuộc dữ liệu cao do nó sử dụng khoảng và trạng thái xác suất được cập nhật thông qua việc lập mã của ngăn trước đó, và nó có thể mất nhiều thời gian để đọc khoảng xác suất và xác định trạng thái hiện tại. Vấn đề thông lượng của CABAC có thể được giải quyết bằng việc giới hạn số lượng của các ngăn được lập mã ngữ cảnh. Ví dụ, như được thể hiện trong Bảng 2 được mô tả ở trên, tổng của các ngăn được sử dụng để trình bày sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag có thể được giới hạn ở số lượng của các ngăn phụ thuộc vào kích thước của khối tương ứng. Ngoài ra, ví dụ, như được thể hiện trong Bảng 3 được mô tả ở trên, tổng của các ngăn được sử dụng để trình bày sig_coeff_flag, coeff_sign_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag abs_level_gt5_flag, abs_level_gt7_flag, abs_level_gt9_flag có thể được giới hạn ở số lượng của các ngăn phụ thuộc vào kích thước của khối tương ứng. Ví dụ, nếu khối tương ứng khối có kích thước 4x4, tổng của các ngăn cho sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag hoặc sig_coeff_flag, coeff_sign_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag abs_level_gt5_flag, abs_level_gt7_flag, abs_level_gt9_flag có thể được giới hạn ở 32 (hoặc ví dụ 28), và nếu khối tương ứng khối có kích thước 2x2, tổng của các ngăn cho sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag có thể được giới hạn ở 8 (hoặc ví dụ 7). Số lượng giới hạn của các ngăn có thể được biểu diễn bởi remBinsPass1 hoặc RemCcbs. Hoặc, ví dụ, đối với thông lượng CABAC cao hơn, số lượng của các ngăn được lập mã ngữ cảnh có thể được giới hạn cho khối (CB hoặc TB) gồm CG mục tiêu lập mã. Nói cách khác, số lượng của các ngăn được lập mã ngữ cảnh có thể được giới hạn về các đơn vị của các khối (CB hoặc TB). Ví dụ, khi kích thước của khối hiện tại là 16x16, số lượng của các ngăn được lập mã ngữ cảnh cho khối hiện tại có thể được giới hạn ở 1,75 lần số lượng của các điểm ảnh của khối hiện tại, nghĩa là, 448, bất kể CG hiện tại.

Trong trường hợp này, nếu tất cả các ngăn được lập mã ngữ cảnh mà số lượng

của chúng được giới hạn được sử dụng khi phần tử ngữ cảnh được lập mã, thiết bị mã hóa có thể nhị phân hóa các hệ số còn lại thông qua phương pháp nhị phân hóa hệ số như được mô tả dưới đây, thay vì sử dụng việc lập mã ngữ cảnh, và có thể thực hiện việc mã hóa đi vòng qua. Nói cách khác, ví dụ, nếu số lượng của các ngăn được lập mã ngữ cảnh mà được lập mã cho 4x4 CG là 32 (hoặc ví dụ 28), hoặc nếu số lượng của các ngăn được lập mã ngữ cảnh mà được lập mã cho 2x2 CG là 8 (hoặc ví dụ 7), sig_coeff_flag, abs_level_gt1_flag, par_level_flag, abs_level_gt3_flag mà được lập mã với ngăn được lập mã ngữ cảnh có thể không còn được lập mã, và có thể được lập mã một cách trực tiếp thành dec_abs_level. Hoặc, ví dụ, khi số lượng của các ngăn được lập mã ngữ cảnh được lập mã cho khối 4x4 bằng 1,75 lần số lượng của các điểm ảnh của toàn bộ khối, nghĩa là, khi được giới hạn ở 28, sig_coeff_flag, abs_level_gt1_flag, par_level_flag, và abs_level_gt3_flag được lập mã dưới dạng các ngăn được lập mã ngữ cảnh có thể không được lập mã nữa, và có thể được lập mã trực tiếp dưới dạng dec_abs_level như được thể hiện trong Bảng 5 dưới đây.

[Bảng 5]

|coeff[n]|	dec_abs_level[n]
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
...	...

Giá trị |coeff| có thể được dẫn xuất dựa trên dec_abs_level. Trong trường hợp này, giá trị hệ số biến đổi, nghĩa là, |coeff|, có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 6]

$$|coeff| = dec\_abs\_level$$

Ngoài ra, coeff_sign_flag có thể chỉ ra dấu của mức hệ số biến đổi ở vị trí quét tương ứng n. Nghĩa là, coeff_sign_flag có thể chỉ ra dấu của hệ số biến đổi ở vị trí quét tương ứng n.

Fig.5 thể hiện ví dụ của các hệ số biến đổi trong khối 4x4.

Khối 4x4 của Fig.5 thể hiện ví dụ của các hệ số được lượng tử hóa. Khối của Fig.5 có thể là khối biến đổi 4x4 hoặc khối con 4x4 của khối biến đổi 8x8, 16x16, 32x32, hoặc 64x64. Khối 4x4 của Fig.5 có thể biểu diễn khối độ chói hoặc khối sắc độ.

Cùng lúc đó, như được mô tả ở trên, khi tín hiệu đầu vào không phải là giá trị nhị phân mà là phần tử cú pháp, thiết bị mã hóa có thể biến đổi tín hiệu đầu vào thành giá trị nhị phân bằng việc nhị phân hóa giá trị của tín hiệu đầu vào. Ngoài ra, thiết bị giải mã có thể giải mã phần tử cú pháp để dẫn xuất giá trị được nhị phân hóa (ví dụ, ngăn được nhị phân hóa) của các phần tử cú pháp, và có thể khử nhị phân hóa giá trị được nhị phân hóa để dẫn xuất giá trị của phần tử cú pháp. Quy trình nhị phân hóa có thể được thực hiện như quy trình nhị phân hóa rice bị cắt cụt (truncated rice, TR), quy trình nhị phân hóa Exp-Golomb thứ tự thứ k (Egk), Exp-Golomb thứ tự thứ k giới hạn (EGk giới hạn), quy trình nhị phân hóa độ dài cố định (fixed-length, FL), hoặc tương tự. Ngoài ra, quy trình khử nhị phân hóa có thể biểu diễn quy trình được thực hiện dựa trên quy trình nhị phân hóa TR, quy trình nhị phân hóa EGk, hoặc quy trình nhị phân hóa FL để dẫn xuất giá trị của phần tử cú pháp.

Ví dụ, quy trình nhị phân hóa TR có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa TR có thể là cMax và cRiceParam cho phần tử cú pháp và yêu cầu cho việc nhị phân hóa TR. Ngoài ra, đầu ra của quy trình nhị phân hóa TR có thể là việc nhị phân hóa TR cho symbolVal mà là giá trị tương ứng với chuỗi ngăn.

Cụ thể là, ví dụ, trong sự xuất hiện của chuỗi ngăn hậu tố cho phần tử cú pháp, chuỗi ngăn TR cho phần tử cú pháp có thể là sự móc nối của chuỗi ngăn tiền tố và chuỗi ngăn hậu tố, và trong sự vắng mặt của chuỗi ngăn hậu tố, chuỗi ngăn TR cho phần tử cú pháp có thể là chuỗi ngăn tiền tố. Ví dụ, chuỗi ngăn tiền tố có thể được dẫn xuất như được mô tả dưới đây.

Giá trị tiền tố của symbolVal cho phần tử cú pháp có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 7]

$$prefixVal = symbolVal >> cRiceParam$$

Ở đây, prefixVal có thể biểu thị giá trị tiền tố của symbolVal. Tiền tố (nghĩa là, chuỗi ngăn tiền tố) của chuỗi ngăn TR của các phần tử cú pháp có thể được dẫn xuất như được mô tả dưới đây.

Ví dụ, nếu prefixVal là nhỏ hơn cMax >> cRiceParam, chuỗi ngăn tiền tố có thể là chuỗi bit có độ dài prefixVal+1, được lập chỉ số bởi binIdx. Nghĩa là, nếu prefixVal là nhỏ hơn cMax >> cRiceParam, chuỗi ngăn tiền tố có thể là chuỗi bit số lượng các bit của nó là prefixVal+1, được chỉ ra bởi binIdx. Ngăn cho binIdx nhỏ hơn prefixVal có thể là bằng với 1. Ngoài ra, ngăn cho cùng binIdx như prefixVal có thể là bằng với 0.

Ví dụ, chuỗi ngăn được dẫn xuất thông qua việc nhị phân hóa một ngôi cho prefixVal có thể là như được thể hiện trong bảng dưới đây.

[Bảng 6]

prefixVal	Bin string					
0	0					
1	1	0				
2	1	1	0			
3	1	1	1	0		
4	1	1	1	1	0	
5	1	1	1	1	1	0
...						
binIdx	0	1	2	3	4	5

Cùng lúc đó, nếu prefixVal là không nhỏ hơn cMax >> cRiceParam, chuỗi ngăn tiền tố có thể là chuỗi bit trong đó độ dài là cMax >> cRiceParam và tất cả các bit là 1.

Ngoài ra, nếu cMax là lớn hơn symbolVal và nếu cRiceParam là lớn hơn 0, ngăn chuỗi ngăn hậu tố của chuỗi ngăn TR có thể có mặt. Ví dụ, chuỗi ngăn hậu tố có thể được dẫn xuất như được mô tả dưới đây.

Giá trị hậu tố của symbolVal cho phần tử cú pháp có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 8]

$$suffixVal = symbolVal - ((prefixVal) << cRiceParam)$$

Ở đây, suffixVal có thể biểu thị giá trị hậu tố của symbolVal.

Hậu tố của chuỗi ngăn TR (nghĩa là, chuỗi ngăn hậu tố) có thể được dẫn xuất dựa trên quy trình nhị phân hóa FL cho suffixVal mà giá trị cMax của nó là (1 << cRiceParam)-1.

Cùng lúc đó, nếu giá trị của thông số đầu vào, nghĩa là, cRiceParam, là 0, việc nhị phân hóa TR có thể là việc nhị phân hóa một ngôi được cắt cụt chính xác, và có thể luôn sử dụng cùng giá trị cMax như giá trị lớn nhất khả dụng của phần tử cú pháp cần được giải mã.

Ngoài ra, ví dụ, quy trình nhị phân hóa EGk có thể được thực hiện như sau. Phần tử cú pháp được lập mã với ue(v) có thể là phần tử cú pháp được tiến hành việc lập mã Exp-Golomb.

Ví dụ, quy trình nhị phân hóa Exp-Golomb thứ tự thứ 0 (EG0) có thể được thực hiện như sau.

Quy trình phân tích cú pháp cho phần tử cú pháp có thể bắt đầu bằng việc đọc bit gồm bit khác không thứ nhất bắt đầu ở vị trí hiện tại của luồng bit và đếm số lượng của các bit dẫn bằng với 0. Quy trình có thể được biểu diễn như được thể hiện trong bảng dưới đây.

[Bảng 7]

```
leadingZeroBits = −1
for( b = 0; !b; leadingZeroBits++ )
    b = read_bits( 1 )
```

Ngoài ra, biến 'codeNum' có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 9]

$$codeNum = 2^{leadingZeroBits} - 1 + read\_bits(leadingZeroBits)$$

Ở đây, giá trị được trả lại từ read_bits(leadingZeroBits), tức là, giá trị được chỉ ra bởi read_bits(leadingZeroBits), có thể được diễn giải như sự biểu diễn nhị phân của số nguyên không được gán dấu cho bit đáng kể nhất được ghi trước tiên.

Cấu trúc của mã Exp-Golomb trong đó chuỗi bit được chia thành bit "tiền tố" và bit "hậu tố" có thể được biểu diễn như được thể hiện trong bảng dưới đây.

[Bảng 8]

Bit string form	Range of codeNum
1	0
0 1 x_0	1..2
0 0 1 x_1 x_0	3..6
0 0 0 1 x_2 x_1 x_0	7..14
0 0 0 0 1 x_3 x_2 x_1 x_0	15..30
0 0 0 0 0 1 x_4 x_3 x_2 x_1 x_0	31..62
...	...

Bit "tiền tố" có thể là bit được phân tích cú pháp như được mô tả ở trên để tính toán leadingZeroBits, và có thể được biểu diễn bởi 0 hoặc 1 của chuỗi bit trong Bảng 8. Nghĩa là, chuỗi bit được bộc lộ bởi 0 hoặc 1 trong Bảng 8 ở trên có thể biểu diễn chuỗi bit tiền tố. Bit "hậu tố" có thể là bit được phân tích cú pháp trong việc tính toán của codeNum, và có thể được biểu diễn bởi xi trong Bảng 8 ở trên. Nghĩa là, chuỗi bit được bộc lộ dưới dạng xi trong Bảng 8 ở trên có thể thể hiện chuỗi bit hậu tố. Ở đây, i có thể là giá trị trong khoảng LeadingZeroBits-1. Ngoài ra, mỗi xi có thể bằng với 0 hoặc 1.

Chuỗi bit được gán cho codeNum có thể là như được thể hiện trong bảng sau đây.

[Bảng 9]

Bit string	codeNum
1	0
0 1 0	1
0 1 1	2
0 0 1 0 0	3
0 0 1 0 1	4
0 0 1 1 0	5
0 0 1 1 1	6
0 0 0 1 0 0 0	7
0 0 0 1 0 0 1	8
0 0 0 1 0 1 0	9
...	...

Nếu bộ mô tả của các phần tử cú pháp là ue(v), tức là, nếu phần tử cú pháp được lập mã với ue(v), giá trị của phần tử cú pháp có thể là bằng với codeNum.

Ngoài ra, ví dụ, quy trình nhị phân hóa EGk có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa EGk có thể là yêu cầu cho việc nhị phân hóa EGk. Ngoài ra, đầu ra của quy trình nhị phân hóa EGk có thể là việc nhị phân hóa EGk cho symbolVal, nghĩa là, giá trị tương ứng với chuỗi ngăn.

Chuỗi bit của quy trình nhị phân hóa EGk cho symbolVal có thể được dẫn xuất như sau.

[Bảng 10]

```
absV = Abs( symbolVal )
stopLoop = 0
do
    if( absV >= ( 1 << k ) ) {
        put( 1 )
        absV = absV − ( 1 << k )
        k++
    } else {
        put( 0 )
        while( k− − )
            put( ( absV >> k ) & 1 )
        stopLoop = 1
    }
while( !stopLoop )
```

Tham khảo Bảng 10 ở trên, giá trị nhị phân X có thể được bổ sung vào một đầu của chuỗi ngăn thông qua mỗi lần gọi của put(X). Ở đây, X có thể là 0 hoặc 1.

Ngoài ra, ví dụ, quy trình nhị phân hóa EGk giới hạn có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa EGk giới hạn có thể là yêu cầu cho việc nhị phân hóa EGk giới hạn, thông số rice riceParam, log2TransformRange như biến biểu diễn logarit nhị phân của giá trị lớn nhất, và maxPreExtLen như biến biểu diễn độ dài mở rộng tiền tố lớn nhất. Ngoài ra, đầu ra của quy trình nhị phân hóa EGk giới hạn có thể là việc nhị phân hóa EGk giới hạn cho symbolVal như giá trị tương ứng với chuỗi trống.

Chuỗi bit của quy trình nhị phân hóa EGk giới hạn cho symbolVal có thể được dẫn xuất như sau.

[Bảng 11]

```
codeValue = symbolVal >> riceParam
PrefixExtensionLength = 0
while( ( PrefixExtensionLength < maxPrefixExtensionLength ) &&
          ( codeValue > ( ( 2 << PrefixExtensionLength ) − 2 ) ) ) {
    PrefixExtensionLength++
    put( 1 )
}
if( PrefixExtensionLength == maxPrefixExtensionLength )
    escapeLength = log2TransformRange
else {
    escapeLength = PrefixExtensionLength + riceParam
    put( 0 )
}
symbolVal = symbolVal − ( ( ( 1 << PrefixExtensionLength ) − 1 ) << riceParam )
while( ( escapeLength− − ) > 0 )
    put( ( symbolVal >> escapeLength ) & 1 )
```

Ngoài ra, ví dụ, quy trình nhị phân hóa FL có thể được thực hiện như sau.

Đầu vào của FL quy trình nhị phân hóa có thể là yêu cầu cho việc nhị phân hóa FL và cMax cho phần tử cú pháp. Ngoài ra, đầu ra của FL quy trình nhị phân hóa có thể là việc nhị phân hóa FL cho symbolVal như giá trị tương ứng với chuỗi ngăn.

Việc nhị phân hóa FL có thể được tạo cấu hình bằng việc sử dụng chuỗi bit số lượng các bit của nó có độ dài cố định của symbolVal. Ở đây, bit có độ dài cố định có thể là chuỗi bit số nguyên không được gán dấu. Nghĩa là, chuỗi bit cho symbolVal như giá trị ký hiệu có thể được dẫn xuất thông qua việc nhị phân hóa FL, và độ dài bit (nghĩa là, số lượng của các bit) của chuỗi bit có thể có độ dài cố định.

Ví dụ, độ dài cố định có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 10]

fixedLength = Ceil(Log2(cMax + 1))

Việc tạo chỉ số của các ngăn cho việc nhị phân hóa FL có thể là phương pháp sử dụng giá trị mà tăng theo thứ tự từ bit đáng kể nhất đến bit ít đáng kể nhất. Ví dụ, chỉ số ngăn liên quan đến bit đáng kể nhất có thể là binIdx = 0.

Cùng lúc đó, ví dụ, quy trình nhị phân hóa cho phần tử cú pháp abs_remainder trong thông tin phần dư có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa cho abs_remainder có thể là yêu cầu cho việc nhị phân hóa của phần tử cú pháp abs_remainder[n], thành phần màu sắc cIdx, và vị trí độ chói (x0, y0). Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói trên cùng bên trái của ảnh.

Đầu ra của quy trình nhị phân hóa cho abs_remainder có thể là việc nhị phân hóa của abs_remainder (nghĩa là, chuỗi ngăn được nhị phân hóa của abs_remainder). Các chuỗi ngăn sẵn có cho abs_remainder có thể được dẫn xuất thông qua quy trình nhị phân hóa.

Thông số rice cRiceParam cho abs_remainder[n] có thể được dẫn xuất thông qua quy trình dẫn xuất thông số rice được thực hiện bằng việc nhập thành phần màu sắc cIdx và vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth, mà là logarit

nhị phân của độ rộng của khối biến đổi, và log2TbHeight, mà là logarit nhị phân của độ cao của khối biến đổi. Phần mô tả chi tiết của quy trình dẫn xuất thông số rice sẽ được mô tả sau.

Ngoài ra, ví dụ, cMax cho abs_remainder[n] cần được lập mã hiện tại có thể được dẫn xuất dựa trên thông số rice cRiceParam. cMax có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 11]

$$cMax = 6 << cRiceParam$$

Cùng lúc đó, việc nhị phân hóa cho abs_remainder, tức là, chuỗi ngăn cho abs_remainder, có thể là sự móc nối của chuỗi ngăn tiền tố và chuỗi ngăn hậu tố trong sự có mặt của chuỗi ngăn hậu tố. Ngoài ra, trong sự vắng mặt của chuỗi ngăn hậu tố, chuỗi ngăn cho abs_remainder có thể là chuỗi ngăn tiền tố.

Ví dụ, chuỗi ngăn tiền tố có thể được dẫn xuất như được mô tả dưới đây.

Giá trị tiền tố prefixVal của abs_remainder[n] có thể được dẫn xuất như được thể hiện trong phương trình sau.

[Phương trình 12]

$$prefixVal = Min(cMax, abs\_remainder[n])$$

Tiền tố của chuỗi ngăn (nghĩa là, chuỗi ngăn tiền tố) của abs_remainder[n] có thể được dẫn xuất thông qua quy trình nhị phân hóa TR cho prefixVal, trong đó cMax và cRiceParam được sử dụng làm đầu vào.

Nếu chuỗi ngăn tiền tố là giống hệt với chuỗi bit trong đó tất cả các bit là 1 và độ dài bit là 6, chuỗi ngăn hậu tố của chuỗi ngăn của abs_remainder[n] có thể tồn tại, và có thể được dẫn xuất như được mô tả dưới đây.

Quy trình dẫn xuất thông số rice cho dec_abs_level[n] có thể là như sau.

Đầu vào của quy trình dẫn xuất thông số rice có thể là chỉ số thành phần màu sắc cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như logarit nhị phân của độ rộng của khối biến đổi, và log2TbHeight như logarit nhị phân của độ cao của khối biến đổi. Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái

của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói phía trên bên trái của ảnh. Ngoài ra, đầu ra của quy trình dẫn xuất thông số rice có thể là thông số rice cRiceParam.

Ví dụ, biến locSumAbs có thể được dẫn xuất tương tự với mã giả được bộc lộ trong bảng dưới đây, dựa trên mảng AbsLevel[x][y] cho khối biến đổi có chỉ số thành phần đã biết cIdx và vị trí độ chói trên cùng bên trái (x0, y0).

[Bảng 12]

```
locSumAbs = 0
if( xC < (1 << log2TbWidth) − 1 ) {
   locSumAbs += AbsLevel[ xC + 1 ][ yC ]
   if( xC < (1 << log2TbWidth) − 2 )
            locSumAbs += AbsLevel[ xC + 2 ][ yC ]
   if( yC < (1 << log2TbHeight) − 1 )
            locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ]                    (1532)
}
if( yC < (1 << log2TbHeight) − 1 ) {
   locSumAbs += AbsLevel[ xC ][ yC + 1 ]
   if( yC < (1 << log2TbHeight) − 2 )
            locSumAbs += AbsLevel[ xC ][ yC + 2 ]
}
locSumAbs = Clip3( 0, 31, locSumAbs − baseLevel * 5 )
```

Sau đó, dựa trên biến đã cho locSumAbs, thông số rice cRiceParam có thể được dẫn xuất như được thể hiện trong bảng dưới đây.

[Bảng 13]

locSumAbs	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
cRiceParam	0	0	0	0	0	0	0	1	1	1	1	1	1	1	2	2
locSumAbs	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
cRiceParam	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3

Ngoài ra, ví dụ, trong quy trình dẫn xuất thông số rice cho abs_remainder[n], baseLevel có thể được thiết đặt là 4.

Mặt khác, ví dụ, thông số rice cRiceParam có thể được xác định dựa trên việc liệu việc bỏ qua biến đổi có được áp dụng cho khối hiện tại hay không. Nghĩa là, nếu việc biến đổi không được áp dụng cho TB hiện tại gồm CG hiện tại, nói cách khác, nếu việc bỏ qua biến đổi được áp dụng cho TB hiện tại gồm CG hiện tại, thông số rice cRiceParam có thể được dẫn xuất để là 1.

Ngoài ra, giá trị hậu tố suffixVal của abs_remainder có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 13]

suffixVal = abs_remainder[n] − cMax

Chuỗi ngăn hậu tố của chuỗi ngăn của abs_remainder có thể được dẫn xuất thông qua quy trình nhị phân hóa EGk giới hạn cho suffixVal trong đó k được thiết đặt là cRiceParam+1, riceParam được thiết đặt là cRiceParam, và log2TransformRange được thiết đặt là 15, và maxPreExtLen được thiết đặt là 11.

Cùng lúc đó, ví dụ, quy trình nhị phân hóa cho phần tử cú pháp dec_abs_level trong thông tin phần dư có thể được thực hiện như sau.

Đầu vào của quy trình nhị phân hóa cho dec_abs_level có thể là yêu cầu cho việc nhị phân hóa của phần tử cú pháp dec_abs_level[n], thành phần màu sắc cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như logarit nhị phân của độ rộng của khối biến đổi, và log2TbHeight như logarit nhị phân của độ cao của khối biến đổi. Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói phía trên bên trái của ảnh.

Đầu ra của quy trình nhị phân hóa cho dec_abs_level có thể là việc nhị phân hóa của dec_abs_level (nghĩa là, chuỗi ngăn được nhị phân hóa của dec_abs_level). Các chuỗi ngăn sẵn có cho dec_abs_level có thể được dẫn xuất thông qua quy trình nhị phân hóa.

Thông số rice cRiceParam cho dec_abs_level[n] có thể được dẫn xuất thông qua quy trình dẫn xuất thông số rice được thực hiện với đầu vào của thành phần màu sắc cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như logarit nhị phân của độ rộng của khối biến đổi, và log2TbHeight như logarit nhị phân của độ cao của khối biến đổi. Quy trình dẫn xuất thông số rice sẽ được mô tả dưới đây một cách chi tiết.

Ngoài ra, ví dụ, cMax cho dec_abs_level[n] có thể được dẫn xuất dựa trên thông số rice cRiceParam. cMax có thể được dẫn xuất như được thể hiện trong bảng dưới đây.

[Phương trình 14]

cMax = 6 << cRiceParam

Cùng lúc đó, việc nhị phân hóa cho dec_abs_level[n], tức là, chuỗi ngăn cho

dec_abs_level[n], có thể là sự móc nối của chuỗi ngăn tiền tố và chuỗi ngăn hậu tố trong sự có mặt của chuỗi ngăn hậu tố. Ngoài ra, trong sự vắng mặt của chuỗi ngăn hậu tố, chuỗi ngăn cho dec_abs_level[n] có thể là chuỗi ngăn tiền tố.

Ví dụ, chuỗi ngăn tiền tố có thể được dẫn xuất như được mô tả dưới đây.

Giá trị tiền tố prefixVal của dec_abs_level[n] có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 15]

prefixVal = Min(cMax, dec_abs_level[n])

Tiền tố của chuỗi ngăn (nghĩa là, chuỗi ngăn tiền tố) của dec_abs_level[n] có thể được dẫn xuất thông qua quy trình nhị phân hóa TR cho prefixVal, trong đó cMax và cRiceParam được sử dụng làm đầu vào.

Nếu chuỗi ngăn tiền tố là giống hệt với chuỗi bit trong đó tất cả các bit là 1 và độ dài bit là 6, chuỗi ngăn hậu tố của chuỗi ngăn của dec_abs_level[n] có thể tồn tại, và có thể được dẫn xuất như được mô tả dưới đây.

Quy trình dẫn xuất thông số rice cho dec_abs_level[n] có thể là như sau.

Đầu vào của quy trình dẫn xuất thông số rice có thể là chỉ số thành phần màu sắc cIdx, vị trí độ chói (x0, y0), vị trí quét hệ số hiện tại (xC, yC), log2TbWidth như logarit nhị phân của độ rộng của khối biến đổi, và log2TbHeight như logarit nhị phân của độ cao của khối biến đổi. Vị trí độ chói (x0, y0) có thể chỉ ra mẫu trên cùng bên trái của khối biến đổi độ chói hiện tại dựa trên mẫu độ chói phía trên bên trái của ảnh. Ngoài ra, đầu ra của quy trình dẫn xuất thông số rice có thể là thông số rice cRiceParam.

Ví dụ, biến locSumAbs có thể được dẫn xuất tương tự với mã giả được bộc lộ trong bảng dưới đây, dựa trên mảng AbsLevel[x][y] cho khối biến đổi có chỉ số thành phần đã biết cIdx và vị trí độ chói trên cùng bên trái (x0, y0).

[Bảng 14]

```
locSumAbs = 0
if( xC < (1 << log2TbWidth) − 1 ) {
   locSumAbs += AbsLevel[ xC + 1 ][ yC ]
   if( xC < (1 << log2TbWidth) − 2 )
            locSumAbs += AbsLevel[ xC + 2 ][ yC ]
   if( yC < (1 << log2TbHeight) − 1 )
            locSumAbs += AbsLevel[ xC + 1 ][ yC + 1 ]                    (1532)
}
if( yC < (1 << log2TbHeight) − 1 ) {
   locSumAbs += AbsLevel[ xC ][ yC + 1 ]
   if( yC < (1 << log2TbHeight) − 2 )
            locSumAbs += AbsLevel[ xC ][ yC + 2 ]
}
locSumAbs = Clip3( 0, 31, locSumAbs − baseLevel * 5 )
```

Sau đó, dựa trên biến đã cho locSumAbs, thông số rice cRiceParam có thể được dẫn xuất như được thể hiện trong bảng dưới đây.

[Bảng 15]

locSumAbs	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
cRiceParam	0	0	0	0	0	0	0	1	1	1	1	1	1	1	2	2
locSumAbs	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
cRiceParam	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3

Ngoài ra, ví dụ, trong quy trình dẫn xuất thông số rice cho dec_abs_level[n], baseLevel có thể được thiết đặt là 0, và ZeroPos[n] có thể được dẫn xuất như sau.

[Phương trình 16]

$$ZeroPos[n] = (QState < 2 ? 1 : 2) << cRiceParam$$

Ngoài ra, giá trị hậu tố suffixVal của dec_abs_level[n] có thể được dẫn xuất như được thể hiện trong phương trình sau đây.

[Phương trình 17]

$$suffixVal = dec\_abs\_level[n] - cMax$$

Chuỗi ngăn hậu tố của chuỗi ngăn của dec_abs_level[n] có thể được dẫn xuất thông qua quy trình nhị phân hóa EGk giới hạn cho suffixVal trong đó k được thiết đặt là cRiceParam+1, truncSuffixLen được thiết đặt là 15, và maxPreExtLen được thiết đặt là 11.

Cùng lúc đó, RRC và TSRC có thể có các sự khác biệt sau đây.

- Ví dụ, thông số rice cRiceParam của phần tử cú pháp abs_remainder[] và

dec_abs_level[] trong RRC có thể được dẫn xuất dựa trên locSumAbs, bảng tìm kiếm và/hoặc baseLevel như được mô tả ở trên, nhưng thông số rice cRiceParam của phần tử cú pháp abs_remainder[] trong TSRC có thể được dẫn xuất là 1. Nghĩa là, ví dụ, khi việc bỏ qua biến đổi được áp dụng cho khối hiện tại (ví dụ, TB hiện tại), thông số Rice cRiceParam cho abs_remainder[] của TSRC cho khối hiện tại có thể được dẫn xuất là 1.

- Ngoài ra, ví dụ, tham khảo Bảng 3 và Bảng 4, trong RRC, abs_level_gtx_flag[n][0] và/hoặc abs_level_gtx_flag[n][1] có thể được báo hiệu, nhưng trong TSRC, abs_level_gtx_flag[n][0], abs_level_gtx_flag[n][1], abs_level_gtx_flag[n][2], abs_level_gtx_flag[n][3], và abs_level_gtx_flag[n][4] có thể được báo hiệu. Ở đây, abs_level_gtx_flag[n][0] có thể được thể hiện dưới dạng abs_level_gt1_flag hoặc cờ mức hệ số thứ nhất, abs_level_gtx_flag[n][1] có thể được thể hiện dưới dạng abs_level_gt3_flag hoặc cờ mức hệ số thứ hai, abs_level_gtx_flag[n][2] có thể được thể hiện dưới dạng abs_level_gt5_flag hoặc cờ mức hệ số thứ ba, abs_level_gtx_flag[n][3] có thể được thể hiện dưới dạng abs_level_gt7_flag hoặc cờ mức hệ số thứ tư, và abs_level_gtx_flag[n][4] có thể được thể hiện dưới dạng abs_level_gt9_flag hoặc cờ mức hệ số thứ năm. Cụ thể là, cờ mức hệ số thứ nhất có thể là cờ về việc liệu mức hệ số có lớn hơn ngưỡng thứ nhất (ví dụ, 1) hay không, cờ mức hệ số thứ hai có thể là cờ về việc liệu mức hệ số có lớn hơn ngưỡng thứ hai (ví dụ, 3) hay không, cờ mức hệ số thứ ba có thể là cờ về việc liệu mức hệ số có lớn hơn ngưỡng thứ ba (ví dụ, 5) hay không, cờ mức hệ số thứ tư có thể là cờ về việc liệu mức hệ số có lớn hơn ngưỡng thứ tư (ví dụ, 7) hay không, cờ mức hệ số thứ năm có thể là cờ về việc liệu mức hệ số có lớn hơn ngưỡng thứ năm (ví dụ, 9) hay không. Như được mô tả ở trên, trong TSRC, được so sánh với RRC, abs_level_gtx_flag[n][0], abs_level_gtx_flag[n][1], và abs_level_gtx_flag[n][2], abs_level_gtx_flag[n][3], abs_level_gtx_flag[n][4] có thể được gồm thêm.

- Ngoài ra, ví dụ, trong RRC, phần tử cú pháp coeff_sign_flag có thể được lập mã đi vòng qua, nhưng trong TSRC, phần tử cú pháp coeff_sign_flag có thể được lập mã đi vòng qua hoặc được lập mã ngữ cảnh.

Ngoài ra, đối với quy trình lượng tử hóa mẫu phần dư, việc lượng tử hóa phụ thuộc có thể được đề xuất. Việc lượng tử hóa phụ thuộc có thể thể hiện phương pháp phụ thuộc vào giá trị của hệ số biến đổi (giá trị của mức hệ số biến đổi) trong đó tập giá

trị được xây dựng lại được cho phép cho hệ số biến đổi hiện tại đứng trước hệ số biến đổi hiện tại trong thứ tự xây dựng lại. Nghĩa là, ví dụ, việc lượng tử hóa phụ thuộc có thể đạt được bằng (a) xác định hai bộ lượng tử hóa vô hướng có các mức xây dựng lại khác nhau, và (b) xác định quy trình chuyển tiếp giữa các bộ lượng tử hóa vô hướng. Việc lượng tử hóa phụ thuộc có thể có hiệu quả ở chỗ vectơ được xây dựng lại được cho phép được tập trung hơn trong không gian vectơ N chiều so sánh với việc lượng tử hóa vô hướng độc lập hiện có. Ở đây, N có thể thể hiện số lượng của các hệ số biến đổi của khối biến đổi.

Fig.6 minh họa ví dụ các bộ lượng tử hóa vô hướng được sử dụng trong việc lượng tử hóa phụ thuộc. Tham chiếu đến Fig.6, vị trí của các mức được xây dựng lại được cho phép có thể được chỉ định bởi kích thước bước lượng tử hóa △. Tham chiếu đến Fig.6, các bộ lượng tử hóa vô hướng có thể được biểu diễn là Q0 và Q1. Bộ lượng tử hóa vô hướng đang được sử dụng có thể được dẫn xuất mà không được báo hiệu tường minh từ luồng bit. Ví dụ, bộ lượng tử hóa đang được sử dụng cho hệ số biến đổi hiện tại có thể được xác định bởi các tính chẵn lẻ của mức hệ số biến đổi trước hệ số biến đổi hiện tại trong thứ tự lập mã/xây dựng lại.

Fig.7 minh họa ví dụ việc chuyển tiếp trạng thái và việc lựa chọn bộ lượng tử hóa cho việc lượng tử hóa phụ thuộc.

Tham chiếu đến Fig.7, việc chuyển tiếp giữa hai bộ lượng tử hóa vô hướng Q0 và Q1 có thể đạt được thông qua máy trạng thái có bốn trạng thái. Bốn trạng thái có thể có bốn giá trị khác nhau (0, 1, 2, và 3). Trong thứ tự lập mã/được xây dựng lại, trạng thái cho hệ số biến đổi hiện tại có thể được xác định bởi các tính chẵn lẻ của mức hệ số biến đổi trước hệ số biến đổi hiện tại.

Ví dụ, trong trường hợp mà quy trình khử lượng tử hóa cho khối biến đổi bắt đầu, trạng thái cho việc lượng tử hóa phụ thuộc có thể được tạo cấu hình là 0. Sau đó, các hệ số biến đổi của khối biến đổi có thể được xây dựng lại theo thứ tự quét (nghĩa là, cùng thứ tự như thứ tự của việc giải mã entrôpi). Ví dụ, sau khi hệ số biến đổi hiện tại được xây dựng lại, như được minh họa trên Fig.7, trạng thái cho việc lượng tử hóa phụ thuộc có thể được cập nhật. Trong thứ tự quét, quy trình khử lượng tử hóa cho hệ số biến đổi đang được xây dựng lại sau khi hệ số biến đổi hiện tại được xây dựng lại có thể được thực hiện dựa trên trạng thái được cập nhật. Trên Fig.7, k có thể thể hiện giá trị của hệ

số biến đổi, nghĩa là, giá trị của mức giá trị hệ số biến đổi. Ví dụ, nếu k (giá trị của hệ số biến đổi hiện tại) & 1 là 0 trong trạng thái trong đó trạng thái hiện tại là 0, trạng thái có thể được cập nhật thành 0, trong khi nếu k&1 là 1, trạng thái có thể được cập nhật thành 2. Ngoài ra, ví dụ, nếu k&1 là 0 trong trạng thái trong đó trạng thái hiện tại là 1, trạng thái có thể được cập nhật thành 2, trong khi nếu k&1 là 1, trạng thái có thể được cập nhật thành 0. Ngoài ra, ví dụ, nếu k&1 là 0 trong trạng thái trong đó trạng thái hiện tại là 2, trạng thái có thể được cập nhật thành 1, trong khi nếu k&1 là 1, trạng thái có thể được cập nhật thành 3. Ngoài ra, ví dụ, nếu k&1 là 0 trong trạng thái trong đó trạng thái hiện tại là 3, trạng thái có thể được cập nhật thành 3, trong khi nếu k&1 là 1, trạng thái có thể được cập nhật thành 1. Tham chiếu đến Fig.7, nếu trạng thái là 0 hoặc 1, bộ lượng tử hóa vô hướng được sử dụng trong quy trình khử lượng tử hóa có thể là Q0, và nếu trạng thái là 2 hoặc 3, bộ lượng tử hóa vô hướng được sử dụng trong quy trình khử lượng tử hóa có thể là Q1. Hệ số biến đổi có thể được khử lượng tử hóa bởi bộ lượng tử hóa vô hướng cho trạng thái hiện tại dựa trên thông số lượng tử hóa cho mức được xây dựng lại của hệ số biến đổi.

Cùng lúc đó, sáng chế đề xuất các phương án liên quan đến việc lập mã dữ liệu phần dư. Các phương án được giải thích trong sáng chế có thể được kết hợp với nhau. Trong phương pháp lập mã dữ liệu phần dư như được mô tả ở trên, việc lập mã phần dư thông thường (RRC) và việc lập mã phần dư bỏ qua biến đổi (TSRC) có thể có mặt.

Trong số hai phương pháp như được mô tả ở trên, phương pháp lập mã dữ liệu phần dư cho khối hiện tại có thể được xác định dựa trên các giá trị của transform_skip_flag và sh_ts_residual_coding_disabled_flag như được minh họa trong Bảng 1. Ở đây, phần tử cú pháp sh_ts_residual_coding_disabled_flag có thể thể hiện xem liệu TSRC có được cho phép hay không. Theo đó, nếu slice_ts_residual_coding_disabled_flag thể hiện rằng TSRC là không được cho phép ngay cả trong trường hợp mà transform_skip_flag thể hiện việc bỏ qua biến đổi, các phần tử cú pháp theo RRC có thể được báo hiệu cho khối bỏ qua biến đổi. Nghĩa là, nếu giá trị của transform_skip_flag là 0, hoặc nếu giá trị của slice_ts_residual_coding_disabled_flag là 1, RRC có thể được sử dụng, trong khi ngược lại, TSRC có thể được sử dụng.

Mặc dù hiệu quả lập mã cao có thể được thu nhận trong các ứng dụng cụ thể (ví

dụ, lập mã không tổn hao và tương tự) bằng việc sử dụng slice_ts_residual_coding_disabled_flag, trong tiêu chuẩn lập mã viđeo/hình ảnh hiện tồn tại, các giới hạn về trường hợp mà việc lượng tử hóa phụ thuộc và slice_ts_residual_coding_disabled_flag được sử dụng cùng nhau đã không được đề xuất. Nghĩa là, việc lượng tử hóa phụ thuộc có thể được hoạt hóa ở mức cao (ví dụ, cú pháp tập thông số trình tự (sequence parameter set, SPS) / cú pháp tập thông số viđeo (video parameter set, VPS) / cú pháp tập thông số giải mã (decoding parameter set, DPS) / cú pháp phần đầu ảnh / cú pháp phần đầu lát) hoặc ở mức thấp (CU/TU), và nếu slice_ts_residual_coding_disabled_flag là 1, các giá trị phụ thuộc vào trạng thái của việc lượng tử hóa phụ thuộc trong RRC có thể thực hiện thao tác không cần thiết (nghĩa là, thao tác theo việc lượng tử hóa phụ thuộc) để làm suy giảm hiệu quả lập mã, hoặc sự hao hụt ngoài ý muốn của hiệu quả lập mã có thể xảy ra do cấu hình sai trong thiết bị mã hóa. Theo đó, phương án này đề xuất các sơ đồ để tạo cấu hình các sự phụ thuộc/các giới hạn giữa hai công nghệ của việc lượng tử hóa phụ thuộc và việc lập mã phần dư (nghĩa là, việc lập mã của các mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại thông qua RRC) trong trường hợp mà slice_ts_residual_coding_disabled_flag = 1, mà được sử dụng cùng nhau để ngăn ngừa hao hụt lập mã ngoài ý muốn hoặc trục trặc không xảy ra.

Theo một phương án, sáng chế đề xuất phương pháp trong đó slice_ts_residual_coding_disabled_flag là phụ thuộc vào ph_dep_quant_enabled_flag. Ví dụ, các phần tử cú pháp được đề xuất trong phương án này có thể là trong bảng sau đây.

[Bảng 16]

slice_header( ) {	Descriptor
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
(...)	
if(!ph_dep_quant_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(pic_scaling_list_enabled_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	uc(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	uc(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

Theo phương án này, slice_ts_residual_coding_disabled_flag có thể được báo hiệu trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0. Ở đây, ph_dep_quant_enabled_flag có thể thể hiện xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Ví dụ, nếu giá trị của ph_dep_quant_enabled_flag là 1, có thể thể hiện rằng việc lượng tử hóa phụ thuộc được cho phép, trong khi nếu giá trị của ph_dep_quant_enabled_flag là 0, có thể thể hiện rằng việc lượng tử hóa phụ thuộc là không được cho phép.

Theo đó, theo phương án này, slice_ts_residual_coding_disabled_flag có thể được báo hiệu chỉ trong trường hợp rằng việc lượng tử hóa phụ thuộc là không được cho phép, và trong trường hợp mà việc lượng tử hóa phụ thuộc được cho phép, và do đó slice_ts_residual_coding_disabled_flag không được báo hiệu, slice_ts_residual_coding_disabled_flag có thể được suy ra là 0. Cùng lúc đó, ph_dep_quant_enabled_flag và slice_ts_residual_coding_disabled_flag có thể được báo hiệu cho cú pháp phần đầu ảnh và/hoặc cú pháp phần đầu lát, hoặc có thể được báo hiệu cho cú pháp mức cao (high level syntax, HLS) khác (ví dụ, cú pháp SPS / cú pháp VPS / cú pháp DPS) mà không phải là cú pháp phần đầu ảnh và cú pháp phần đầu lát hoặc tại mức thấp (CU/TU). Nếu ph_dep_quant_enabled_flag được báo hiệu cho cú pháp

ngoại trừ cú pháp phần đầu ảnh, nó có thể được gọi bằng tên khác. Ví dụ, ph_dep_quant_enabled_flag có thể được biểu diễn là sh_dep_quant_enabled_flag, sh_dep_quant_used_flag, hoặc sps_dep_quant_enabled_flag.

Ngoài ra, sáng chế đề xuất phương án khác để tạo cấu hình các sự phụ thuộc/các giới hạn giữa việc lượng tử hóa phụ thuộc và việc lập mã phần dư (nghĩa là, việc lập mã của các mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại thông qua RRC) trong trường hợp mà slice_ts_residual_coding_disabled_flag = 1. Ví dụ, phương án này đề xuất sơ đồ để tạo ra trạng thái của việc lượng tử hóa phụ thuộc không được sử dụng trong việc lập mã giá trị mức của hệ số biến đổi trong trường hợp mà giá trị của slice_ts_residual_coding_disabled_flag là 1 nhằm ngăn ngừa hao hụt lập mã ngoài ý muốn hoặc trục trặc không xảy ra thông qua việc sử dụng của việc lượng tử hóa phụ thuộc và việc lập mã phần dư (nghĩa là, việc lập mã của các mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại thông qua RRC) trong trường hợp mà slice_ts_residual_coding_disabled_flag = 1 cùng nhau. Cú pháp lập mã phần dư theo phương án này có thể là như trong bảng sau đây.

[Bảng 17]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth == 5 && log2TbHeight < 6)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth < 6 && log2TbHeight == 5)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	

```
do {
    if( lastScanPos == 0 ) {
        lastScanPos = numSbCoeff
        lastSubBlock--
    }
    lastScanPos--
    xS = DiagScanOrder[ log2TbWidth - log2SbW ][ log2TbHeight - log2SbH ]
                        [ lastSubBlock ][ 0 ]
    yS = DiagScanOrder[ log2TbWidth - log2SbW ][ log2TbHeight - log2SbH ]
                        [ lastSubBlock ][ 1 ]
    xC = ( xS << log2SbW ) + DiagScanOrder[ log2SbW ][ log2SbH ][ lastScanPos ][ 0 ]
    yC = ( yS << log2SbH ) + DiagScanOrder[ log2SbW ][ log2SbH ][ lastScanPos ][ 1 ]
} while( ( xC != LastSignificantCoeffX ) || ( yC != LastSignificantCoeffY ) )
if( lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 &&
    !transform_skip_flag[ x0 ][ y0 ][ cIdx ] && lastScanPos > 0 )
    LfnstDcOnly = 0
if( ( lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2 ) ||
    ( lastScanPos > 7 && ( log2TbWidth == 2 || log2TbWidth == 3 ) &&
    log2TbWidth == log2TbHeight ) )
    LfnstZeroOutSigCoeffFlag = 0
if( ( lastSubBlock > 0 || lastScanPos > 0 ) && cIdx == 0 )
    MtsDcOnly = 0
QState = 0
for( i = lastSubBlock; i >= 0; i-- ) {
    startQStateSb = QState
    xS = DiagScanOrder[ log2TbWidth - log2SbW ][ log2TbHeight - log2SbH ]
                        [ i ][ 0 ]
    yS = DiagScanOrder[ log2TbWidth - log2SbW ][ log2TbHeight - log2SbH ]
                        [ i ][ 1 ]
```

inferSbDcSigCoeffFlag = 0	
if(i < lastSubBlock && i > 0) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
if(coded_sub_block_flag[xS][yS] && (xS > 3 || yS > 3) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = firstPosMode0	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != LastSignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	

AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n][1]	
if(ph_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
firstPosMode1 = n − 1	
}	
for(n = firstPosMode0; n > firstPosMode1; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(abs_level_gtx_flag[n][1])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder[n]	
}	
for(n = firstPosMode1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS])	
dec_abs_level[n]	ae(v)
if(AbsLevel[xC][yC] > 0) {	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
if(ph_dep_quant_enabled_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(ph_dep_quant_enabled_flag || !pic_sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	

for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(ph_dep_quant_enabled_flag && !slice_ts_residual_coding_disabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	

} else {	
sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

Tham khảo Bảng 17 như được mô tả ở trên, trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của slice_ts_residual_coding_disabled_flag là 0, Qstate có thể được dẫn xuất, và giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất dựa trên Qstate. Ví dụ, tham khảo Bảng 17, mức hệ số biến đổi

TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi, và (QState > 1 ? 1 : 0) có thể thể hiện 1 nếu giá trị của trạng thái QState là lớn hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 2 hoặc 3, và có thể thể hiện 0 nếu giá trị của trạng thái Qstate là bằng với hoặc nhỏ hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 0 hoặc 1.

Ngoài ra, tham khảo Bảng 17 như được mô tả ở trên, nếu giá trị của slice_ts_residual_coding_disabled_flag là 1, giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất mà không sử dụng Qstate. Ví dụ, tham khảo Bảng 17, mức hệ số biến đổi TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, và coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi.

Ngoài ra, theo phương án này, nếu giá trị của slice_ts_residual_coding_disabled_flag là 1, trạng thái của việc lượng tử hóa phụ thuộc có thể không được sử dụng trong việc lập mã giá trị mức của hệ số biến đổi, và việc cập nhật trạng thái có thể cũng không được thực hiện. Ví dụ, cú pháp lập mã phần dư theo phương án này có thể là như trong bảng sau đây.

[Bảng 18]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth == 5 && log2TbHeight < 6)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth < 6 && log2TbHeight == 5)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	

do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock--	
}	
lastScanPos--	
xS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH] [lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH] [lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth == 2 || log2TbWidth == 3) && log2TbWidth == log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((lastSubBlock > 0 || lastScanPos > 0) && cIdx == 0)	
MtsDcOnly = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i--) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH] [i][0]	
yS = DiagScanOrder[log2TbWidth - log2SbW][log2TbHeight - log2SbH] [i][1]	

inferSbDcSigCoeffFlag = 0	
if(i < lastSubBlock && i > 0) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
if(coded_sub_block_flag[xS][yS] && (xS > 3 || yS > 3) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = firstPosMode0	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != LastSignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	

AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag [n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n] [1]	
if(ph_dep_quant_enabled_flag && !slice_ts_residual_coding_disabled_flag)	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
firstPosMode1 = n − 1	
}	
for(n = firstPosMode0; n > firstPosMode1; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n] [0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n] [1]	
if(abs_level_gtx_flag[n][1])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder [n]	
}	
for(n = firstPosMode1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n] [0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n] [1]	
if(coded_sub_block_flag[xS][yS])	
dec_abs_level[n]	ac(v)
if(AbsLevel[xC][yC] > 0) {	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
if(ph_dep_quant_enabled_flag && !slice_ts_residual_coding_disabled_flag)	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(ph_dep_quant_enabled_flag || !pic_sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	

for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(ph_dep_quant_enabled_flag && !slice_ts_residual_coding_disabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	

} else {	
sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

Tham khảo Bảng 18 như được mô tả ở trên, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của slice_ts_residual_coding_disabled_flag là 0, Qstate có thể được cập nhật. Ví dụ, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của slice_ts_residual_coding_disabled_flag là 0, QState có thể được cập

nhật như QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1] hoặc QStateTransTable[QState][AbsLevel[xC][yC] & 1]. Ngoài ra, nếu giá trị của slice_ts_residual_coding_disabled_flag là 1, quy trình cập nhật Qstate có thể không được thực hiện.

Ngoài ra, tham khảo Bảng 18 như được mô tả ở trên, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của slice_ts_residual_coding_disabled_flag là 0, giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất dựa trên QState. Ví dụ, tham khảo Bảng 18, mức hệ số biến đổi TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi, và (QState > 1 ? 1 : 0) có thể thể hiện 1 nếu giá trị của trạng thái QState là lớn hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 2 hoặc 3, và có thể thể hiện 0 nếu giá trị của trạng thái Qstate là bằng với hoặc nhỏ hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 0 hoặc 1.

Ngoài ra, tham khảo Bảng 18 như được mô tả ở trên, nếu giá trị của slice_ts_residual_coding_disabled_flag là 1, giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất mà không sử dụng Qstate. Ví dụ, tham khảo Bảng 18, mức hệ số biến đổi TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, và coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi.

Ngoài ra, sáng chế đề xuất phương án khác để tạo cấu hình các sự phụ thuộc/các giới hạn giữa việc lượng tử hóa phụ thuộc và việc lập mã phần dư (nghĩa là, việc lập mã của các mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại thông qua RRC) trong trường hợp mà slice_ts_residual_coding_disabled_flag = 1. Ví dụ, phương án này đề xuất sơ đồ để bổ sung các giới hạn sử dụng transform_skip_flag trong quy trình dẫn xuất giá trị của hệ số biến đổi (mức hệ số biến đổi) một cách phụ thuộc vào việc cập nhật trạng thái hoặc trạng thái của việc lượng tử hóa phụ thuộc trong RRC. Nghĩa là, phương án này đề xuất sơ đồ để tạo ra quy trình dẫn xuất giá trị của hệ số biến đổi (mức hệ số

biến đổi) không được sử dụng một cách phụ thuộc vào việc cập nhật trạng thái và/hoặc trạng thái của việc lượng tử hóa phụ thuộc trong RRC dựa trên transform_skip_flag. Cú pháp lập mã phần dư theo phương án này có thể là như trong Bảng sau đây.

[Bảng 19]

residual_coding(x0, y0, log2TbWidth, log2TbHeight, cIdx) {	Descriptor
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth == 5 && log2TbHeight < 6)	
log2ZoTbWidth = 4	
else	
log2ZoTbWidth = Min(log2TbWidth, 5)	
if(sps_mts_enabled_flag && cu_sbt_flag && cIdx == 0 && log2TbWidth < 6 && log2TbHeight == 5)	
log2ZoTbHeight = 4	
else	
log2ZoTbHeight = Min(log2TbHeight, 5)	
if(log2TbWidth > 0)	
last_sig_coeff_x_prefix	ae(v)
if(log2TbHeight > 0)	
last_sig_coeff_y_prefix	ae(v)
if(last_sig_coeff_x_prefix > 3)	
last_sig_coeff_x_suffix	ae(v)
if(last_sig_coeff_y_prefix > 3)	
last_sig_coeff_y_suffix	ae(v)
log2TbWidth = log2ZoTbWidth	
log2TbHeight = log2ZoTbHeight	
remBinsPass1 = ((1 << (log2TbWidth + log2TbHeight)) * 7) >> 2	
log2SbW = (Min(log2TbWidth, log2TbHeight) < 2 ? 1 : 2)	
log2SbH = log2SbW	
if(log2TbWidth + log2TbHeight > 3)	
if(log2TbWidth < 2) {	
log2SbW = log2TbWidth	
log2SbH = 4 − log2SbW	
} else if(log2TbHeight < 2) {	
log2SbH = log2TbHeight	
log2SbW = 4 − log2SbH	
}	
numSbCoeff = 1 << (log2SbW + log2SbH)	
lastScanPos = numSbCoeff	
lastSubBlock = (1 << (log2TbWidth + log2TbHeight − (log2SbW + log2SbH))) − 1	

do {	
if(lastScanPos == 0) {	
lastScanPos = numSbCoeff	
lastSubBlock−−	
}	
lastScanPos−−	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [lastSubBlock][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [lastSubBlock][1]	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][lastScanPos][1]	
} while((xC != LastSignificantCoeffX) || (yC != LastSignificantCoeffY))	
if(lastSubBlock == 0 && log2TbWidth >= 2 && log2TbHeight >= 2 && !transform_skip_flag[x0][y0][cIdx] && lastScanPos > 0)	
LfnstDcOnly = 0	
if((lastSubBlock > 0 && log2TbWidth >= 2 && log2TbHeight >= 2) || (lastScanPos > 7 && (log2TbWidth == 2 || log2TbWidth == 3) && log2TbWidth == log2TbHeight))	
LfnstZeroOutSigCoeffFlag = 0	
if((lastSubBlock > 0 || lastScanPos > 0) && cIdx == 0)	
MtsDcOnly = 0	
QState = 0	
for(i = lastSubBlock; i >= 0; i−−) {	
startQStateSb = QState	
xS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [i][0]	
yS = DiagScanOrder[log2TbWidth − log2SbW][log2TbHeight − log2SbH] [i][1]	

inferSbDcSigCoeffFlag = 0	
if(i < lastSubBlock && i > 0) {	
coded_sub_block_flag[xS][yS]	ae(v)
inferSbDcSigCoeffFlag = 1	
}	
if(coded_sub_block_flag[xS][yS] && (xS > 3 || yS > 3) && cIdx == 0)	
MtsZeroOutSigCoeffFlag = 0	
firstSigScanPosSb = numSbCoeff	
lastSigScanPosSb = −1	
firstPosMode0 = (i == lastSubBlock ? lastScanPos : numSbCoeff − 1)	
firstPosMode1 = firstPosMode0	
for(n = firstPosMode0; n >= 0 && remBinsPass1 >= 4; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS] && (n > 0 || !inferSbDcSigCoeffFlag) && (xC != LastSignificantCoeffX || yC != LastSignificantCoeffY)) {	
sig_coeff_flag[xC][yC]	ae(v)
remBinsPass1− −	
if(sig_coeff_flag[xC][yC])	
inferSbDcSigCoeffFlag = 0	
}	
if(sig_coeff_flag[xC][yC]) {	
abs_level_gtx_flag[n][0]	ae(v)
remBinsPass1− −	
if(abs_level_gtx_flag[n][0]) {	
par_level_flag[n]	ae(v)
remBinsPass1− −	
abs_level_gtx_flag[n][1]	ae(v)
remBinsPass1− −	
}	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	

AbsLevelPass1[xC][yC] = sig_coeff_flag[xC][yC] + par_level_flag[n] + abs_level_gtx_flag[n][0] + 2 * abs_level_gtx_flag[n][1]	
if(ph_dep_quant_enabled_flag && !transform_skip_flag[x0][y0][cIdx])	
QState = QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1]	
firstPosMode1 = n − 1	
}	
for(n = firstPosMode0; n > firstPosMode1; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(abs_level_gtx_flag[n][1])	
abs_remainder[n]	ae(v)
AbsLevel[xC][yC] = AbsLevelPass1[xC][yC] +2 * abs_remainder[n]	
}	
for(n = firstPosMode1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(coded_sub_block_flag[xS][yS])	
dec_abs_level[n]	ac(v)
if(AbsLevel[xC][yC] > 0) {	
if(lastSigScanPosSb == −1)	
lastSigScanPosSb = n	
firstSigScanPosSb = n	
}	
if(ph_dep_quant_enabled_flag && !transform_skip_flag[x0][y0][cIdx])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	
}	
if(ph_dep_quant_enabled_flag || !pic_sign_data_hiding_enabled_flag)	
signHidden = 0	
else	
signHidden = (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	

for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHidden || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)
}	
if(ph_dep_quant_enabled_flag && !transform_skip_flag[x0][y0][cIdx]) {	
QState = startQStateSb	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0)	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][AbsLevel[xC][yC] & 1]	

} else {	
sumAbsLevel = 0	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if(AbsLevel[xC][yC] > 0) {	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if((n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1))	
TransCoeffLevel[x0][y0][cIdx][xC][yC] = −TransCoeffLevel[x0][y0][cIdx][xC][yC]	
}	
}	
}	
}	
}	
}	

Tham khảo Bảng 19 như được mô tả ở trên, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của transform_skip_flag là 0, Qstate có thể được cập nhật. Ví dụ, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của transform_skip_flag là 0, Qstate có thể được cập nhật như

QStateTransTable[QState][AbsLevelPass1[xC][yC] & 1] hoặc QStateTransTable[QState][AbsLevel[xC][yC] & 1]. Ngoài ra, nếu giá trị của transform_skip_flag là 1, quy trình cập nhật Qstate có thể không được thực hiện.

Ngoài ra, tham khảo Bảng 19 như được mô tả ở trên, nếu giá trị của ph_dep_quant_enabled_flag là 1, và giá trị của transform_skip_flag là 0, Qstate có thể được dẫn xuất, và giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất dựa trên QState. Ví dụ, tham khảo Bảng 19, mức hệ số biến đổi TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là (2 * AbsLevel[xC][yC] − (QState > 1 ? 1 : 0)) * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi, và (QState > 1 ? 1 : 0) có thể thể hiện 1 nếu giá trị của trạng thái QState là lớn hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 2 hoặc 3, và có thể thể hiện 0 nếu giá trị của trạng thái Qstate là bằng với hoặc nhỏ hơn 1, nghĩa là, nếu giá trị của trạng thái Qstate là 0 hoặc 1.

Ngoài ra, tham khảo Bảng 19 như được mô tả ở trên, nếu giá trị của transform_skip_flag là 1, giá trị của hệ số biến đổi (mức hệ số biến đổi) có thể được dẫn xuất mà không sử dụng Qstate. Theo đó, trong trường hợp mà dữ liệu phần dư theo RRC được lập mã cho khối bỏ qua biến đổi, giá trị của hệ số biến đổi có thể được dẫn xuất mà không sử dụng Qstate. Ví dụ, tham khảo Bảng 19, mức hệ số biến đổi TransCoeffLevel[x0][y0][cIdx][xC][yC] có thể được dẫn xuất là AbsLevel[xC][yC] * (1 − 2 * coeff_sign_flag[n]). Ở đây, AbsLevel[xC][yC] có thể là giá trị tuyệt đối của hệ số biến đổi được dẫn xuất dựa trên các phần tử cú pháp của hệ số biến đổi, và coeff_sign_flag[n] có thể là phần tử cú pháp của cờ dấu thể hiện dấu của hệ số biến đổi.

Ngoài ra, sáng chế đề xuất các phương án khác nhau liên quan đến việc báo hiệu của phần tử cú pháp được mô tả ở trên sh_ts_residual_coding_disabled_flag.

Ví dụ, như được mô tả ở trên, sh_ts_residual_coding_disabled_flag là phần tử cú pháp xác định xem TSRC có không được cho phép hay không, và trong trường hợp mà khối bỏ qua biến đổi không được sử dụng, có thể không cần thiết phải được báo hiệu. Nghĩa là, chỉ trong trường hợp mà phần tử cú pháp về việc liệu khối bỏ qua biến đổi có được sử dụng hay không thể hiện rằng khối bỏ qua biến đổi được sử dụng, có thể là đáng

kể để thực hiện việc báo hiệu của sh_ts_residual_coding_disabled_flag.

Theo đó, sáng chế đề xuất phương án báo hiệu sh_ts_residual_coding_disabled_flag chỉ trong trường hợp mà sps_transform_skip_enabled_flag là 1. Cú pháp theo phương án này là như trong bảng sau đây.

[Bảng 20]

slice_header( ) {	Descriptor
(...)	
if(sps_transform_skip_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 20, nếu sps_transform_skip_enabled_flag là 1, sh_ts_residual_coding_disabled_flag có thể được báo hiệu, trong khi nếu sps_transform_skip_enabled_flag là 0, sh_ts_residual_coding_disabled_flag có thể không được báo hiệu. Ở đây, ví dụ, sps_transform_skip_enabled_flag có thể thể hiện xem khối bỏ qua biến đổi có được sử dụng hay không. Nghĩa là, ví dụ, sps_transform_skip_enabled_flag có thể thể hiện xem việc bỏ qua biến đổi có được cho phép hay không. Ví dụ, nếu giá trị của sps_transform_skip_enabled_flag là 1, sps_transform_skip_enabled_flag có thể thể hiện rằng cờ bỏ qua biến đổi (transform_skip_flag) có thể có mặt trong cú pháp đơn vị biến đổi, trong khi nếu giá trị của sps_transform_skip_enabled_flag là 0, sps_transform_skip_enabled_flag có thể thể hiện rằng cờ bỏ qua biến đổi không có mặt trong cú pháp đơn vị biến đổi. Cùng lúc đó, nếu sh_ts_residual_coding_disabled_flag không được báo hiệu, có thể được suy ra rằng sh_ts_residual_coding_disabled_flag là 0. Ngoài ra, sps_transform_skip_enabled_flag được mô tả ở trên có thể được báo hiệu trong SPS, hoặc có thể được báo hiệu trong các cú pháp mức cao khác (VPS, PPS, cú pháp phần đầu ảnh, và cú pháp phần đầu lát) không phải là SPS, hoặc các cú pháp mức thấp (cú pháp dữ liệu lát, cú pháp đơn vị lập mã, và cú pháp đơn vị biến đổi). Ngoài ra, nó có thể được báo hiệu trước slice_ts_residual_coding_disabled_flag.

Ngoài ra, sáng chế đề xuất phương án trong đó các phương án được mô tả ở trên được kết hợp liên quan tới việc báo hiệu của sh_ts_residual_coding_disabled_flag. Ví

dụ, như trong bảng sau đây, phương án để báo hiệu sh_ts_residual_coding_disabled_flag có thể được đề xuất.

[Bảng 21]

slice_header( ) {	Descriptor
(...)	
if(!ph_dep_quant_enabled_flag || sps_transform_skip_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 21, trong trường hợp mà sps_transform_skip_enabled_flag là 1, hoặc ph_dep_quant_enabled_flag là 0, sh_ts_residual_coding_disabled_flag có thể được báo hiệu, và nếu không, sh_ts_residual_coding_disabled_flag có thể không được báo hiệu. Cùng lúc đó, trong trường hợp mà sh_ts_residual_coding_disabled_flag không được báo hiệu, sh_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Ngoài ra, ví dụ, phương án báo hiệu sh_ts_residual_coding_disabled_flag như trong bảng sau đây có thể được đề xuất.

[Bảng 22]

picture_header_structure( ) {	Descriptor
(...)	
ph_dep_quant_enabled_flag	
if(!ph_dep_quant_enabled_flag && sps_transform_skip_enabled_flag)	
ph_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 22, sh_ts_residual_coding_disabled_flag có thể được báo hiệu tới phần đầu ảnh. sh_ts_residual_coding_disabled_flag có thể được biểu diễn như ph_ts_residual_coding_disabled_flag. Ngoài ra, tham khảo Bảng 22, ph_dep_quant_enabled_flag có thể được báo hiệu tới phần đầu ảnh.

Ví dụ, tham khảo Bảng 22, trong trường hợp mà ph_dep_quant_enabled_flag là 0, và sps_transform_skip_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể được báo hiệu, và nếu không, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Cùng lúc đó, trong trường hợp mà ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Trong tiêu chuẩn lập mã viđeo/hình ảnh hiện có liên quan tới các phần tử cú pháp được mô tả trong các phương án của sáng chế, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và sh_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu lát. Liên quan tới điều này, sáng chế đề xuất phương án báo hiệu hai phần tử cú pháp trong cùng cú pháp mức cao hoặc cú pháp mức thấp.

Ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và sh_ts_residual_coding_disabled_flag được báo hiệu trong cú pháp phần đầu ảnh. Trong trường hợp này, sh_ts_residual_coding_disabled_flag có thể được gọi là ph_ts_residual_coding_disabled_flag.

Ngoài ra, ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và sh_ts_residual_coding_disabled_flag được báo hiệu trong cú pháp phần đầu lát. Trong trường hợp này, ph_dep_quant_enabled_flag có thể được gọi là sh_dep_quant_enabled_flag, sh_dep_quant_used_flag, hoặc slice_dep_quant_enabled_flag.

Ngoài ra, ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong cùng HLS, nhưng ph_ts_residual_coding_disabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0. Ví dụ, ví dụ trong đó cả ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong cú pháp phần đầu ảnh có thể là như trong bảng sau đây.

[Bảng 23]

picture_header_structure( ) {	**Descriptor**
(...)	
ph_dep_quant_enabled_flag	
if(!ph_dep_quant_enabled_flag)	
ph_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 23, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và nếu giá trị của ph_dep_quant_enabled_flag là 0, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh,

trong khi nếu giá trị của ph_dep_quant_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Ví dụ, nếu ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Cùng lúc đó, phương án được mô tả ở trên là ví dụ, và ví dụ có thể được đề xuất, trong đó ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong các cú pháp mức cao khác (VPS, SPS, PPS, và cú pháp phần đầu lát) hoặc các cú pháp mức thấp (cú pháp dữ liệu lát, cú pháp đơn vị lập mã, và cú pháp đơn vị biến đổi) thay vì cú pháp phần đầu ảnh.

Ngoài ra, ví dụ, phương án có thể được đề xuất, trong đó cả ph_ts_residual_coding_disabled_flag và ph_dep_quant_enabled_flag được báo hiệu trong cùng HLS, nhưng ph_dep_quant_enabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của ph_ts_residual_coding_disabled_flag là 0.

[Bảng 24]

picture_header_structure( ) {	**Descriptor**
(...)	
ph_ts_residual_coding_disabled_flag	
if(!ph_ts_residual_coding_disabled_flag)	
ph_dep_quant_enabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 24, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và nếu giá trị của ph_ts_residual_coding_disabled_flag là 0, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, trong khi nếu giá trị của ph_ts_residual_coding_disabled_flag là 1, ph_dep_quant_enabled_flag có thể không được báo hiệu. Ví dụ, nếu ph_dep_quant_enabled_flag không được báo hiệu, ph_dep_quant_enabled_flag có thể được suy ra là 0.

Cùng lúc đó, phương án được mô tả ở trên là ví dụ, và ví dụ có thể được đề xuất, trong đó ph_ts_residual_coding_disabled_flag và ph_dep_quant_enabled_flag được báo hiệu trong các cú pháp mức cao khác (VPS, SPS, PPS, và cú pháp phần đầu lát) hoặc các cú pháp mức thấp (cú pháp dữ liệu lát, cú pháp đơn vị lập mã, và cú pháp đơn

vị biến đổi) thay vì cú pháp phần đầu ảnh.

Ngoài ra, ví dụ, phương án trong đó các phương án được mô tả ở trên được kết hợp với nhau có thể được đề xuất. Ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong cùng HLS, nhưng ph_ts_residual_coding_disabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0, hoặc giá trị của sps_transform_skip_enabled_flag là 1.

[Bảng 25]

picture_header_structure( ) {	**Descriptor**
(...)	
ph_dep_quant_enabled_flag	
if(!ph_dep_quant_enabled_flag || sps_transform_skip_enabled_flag)	
ph_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 25, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0, hoặc giá trị của sps_transform_skip_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và nếu không, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Ở đây, ví dụ, sps_transform_skip_enabled_flag có thể thể hiện xem khối bỏ qua biến đổi có được sử dụng hay không. Nghĩa là, ví dụ, sps_transform_skip_enabled_flag có thể thể hiện xem việc bỏ qua biến đổi có được cho phép hay không. Ví dụ, nếu giá trị của sps_transform_skip_enabled_flag là 1, sps_transform_skip_enabled_flag có thể thể hiện rằng cờ bỏ qua biến đổi (transform_skip_flag) có thể có mặt trong cú pháp đơn vị biến đổi, trong khi nếu giá trị của sps_transform_skip_enabled_flag là 0, sps_transform_skip_enabled_flag có thể thể hiện rằng cờ bỏ qua biến đổi không có mặt trong cú pháp đơn vị biến đổi. Ví dụ, nếu ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Ngoài ra, ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong cùng HLS (ví dụ, cú pháp phần đầu lát hoặc tương tự), nhưng

ph_ts_residual_coding_disabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0, và giá trị của sps_transform_skip_enabled_flag là 1.

[Bảng 26]

picture_header_structure( ) {	Descriptor
(...)	
ph_dep_quant_enabled_flag	
if(!ph_dep_quant_enabled_flag && sps_transform_skip_enabled_flag)	
ph_ts_residual_coding_disabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 26, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và trong trường hợp mà giá trị của ph_dep_quant_enabled_flag là 0, và giá trị của sps_transform_skip_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và nếu không, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Ví dụ, nếu ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Ngoài ra, ví dụ, phương án có thể được đề xuất, trong đó cả ph_dep_quant_enabled_flag và ph_ts_residual_coding_disabled_flag được báo hiệu trong cùng HLS, nhưng ph_ts_residual_coding_disabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của sps_transform_skip_enabled_flag là 1, và ph_dep_quant_enabled_flag được báo hiệu chỉ trong trường hợp mà giá trị của ph_ts_residual_coding_disabled_flag là 0.

[Bảng 27]

picture_header_structure( ) {	Descriptor
(...)	
if(sps_transform_skip_enabled_flag)	
ph_ts_residual_coding_disabled_flag	
if(!ph_ts_residual_coding_disabled_flag)	
ph_dep_quant_enabled_flag	u(1)
(...)	
}	

Tham khảo Bảng 27, nếu giá trị của sps_transform_skip_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh,

trong khi nếu giá trị của ph_ts_residual_coding_disabled_flag là 0, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Ví dụ, nếu giá trị của sps_transform_skip_enabled_flag là 0, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Ví dụ, nếu ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0. Ngoài ra, ví dụ, nếu giá trị của ph_ts_residual_coding_disabled_flag là 1, ph_dep_quant_enabled_flag có thể không được báo hiệu. Ví dụ, nếu ph_dep_quant_enabled_flag không được báo hiệu, ph_dep_quant_enabled_flag có thể được suy ra là 0,

Cùng lúc đó, như được mô tả ở trên, thông tin (các phần tử cú pháp) trong bảng cú pháp được bộc lộ trong sáng chế có thể được gồm trong thông tin hình ảnh/viđeo, được tạo cấu hình/được mã hóa bởi thiết bị mã hóa, và được chuyển tới thiết bị giải mã dưới dạng luồng bit. Thiết bị giải mã có thể phân tích cú pháp/giải mã thông tin (các phần tử cú pháp) trong bảng cú pháp tương ứng. Thiết bị giải mã có thể thực hiện thủ tục xây dựng lại khối/hình ảnh/viđeo dựa trên thông tin được giải mã.

Ngoài ra, sáng chế đề xuất các phương án khác nhau liên quan đến việc báo hiệu của phần tử cú pháp được mô tả ở trên sh_ts_residual_coding_disabled_flag.

Ví dụ, hiệu quả lập mã cao có thể được thu nhận trong ứng dụng cụ thể (ví dụ, lập mã không tổn hao, v.v.) bằng việc sử dụng slice_ts_residual_coding_disabled_flag như được mô tả ở trên, nhưng trong tiêu chuẩn lập mã viđeo/hình ảnh hiện có, ràng buộc trong trường hợp trong đó slice_ts_residual_coding_disabled_flag được sử dụng cùng nhau cho việc ẩn dữ liệu dấu (sign data hiding, SDH) không được đề xuất.

Ở đây, phương pháp ẩn dữ liệu dấu có thể là như sau. Trong việc dẫn xuất hệ số biến đổi, dấu của hệ số biến đổi có thể được dẫn xuất dựa trên cờ dấu 1-bit (phần tử cú pháp được mô tả ở trên coeff_sign_flag). Theo đó, SDH có thể chỉ ra kỹ thuật để bỏ qua việc báo hiệu tường minh của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất trong khối con/nhóm hệ số (CG) để cải thiện hiệu quả lập mã. Ở đây, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất dựa trên tổng của các mức tuyệt đối (nghĩa là, các giá trị tuyệt đối) của các hệ số biến đổi đáng kể trong khối con/nhóm hệ số tương ứng. Nghĩa là, dấu của hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất dựa trên tổng của các mức tuyệt đối của các hệ số biến đổi đáng

kể trong khối con/nhóm hệ số tương ứng. Trong khi đó, hệ số biến đổi đáng kể có thể có nghĩa là hệ số biến đổi khác không mà giá trị (tuyệt đối) của nó là khác 0. Ví dụ, khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là chẵn, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là 1, và khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là số lẻ, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là 0. Nói cách khác, ví dụ, khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là chẵn, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị âm, và khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là số lẻ, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị dương. Hoặc, ví dụ, khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là chẵn, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là 0, và khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là số lẻ, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là 1. Nói cách khác, ví dụ, khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là chẵn, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị dương, và khi tổng của các mức tuyệt đối cho các hệ số biến đổi đáng kể là số lẻ, giá trị của coeff_sign_flag cho hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị âm.

Ví dụ, SDH trong cú pháp phần dư có thể là như được thể hiện trong bảng dưới đây.

[Bảng 28]

}	
signHiddenFlag = sh_sign_data_hiding_used_flag && (lastSigScanPosSb − firstSigScanPosSb > 3 ? 1 : 0)	
for(n = numSbCoeff − 1; n >= 0; n− −) {	
xC = (xS << log2SbW) + DiagScanOrder[log2SbW][log2SbH][n][0]	
yC = (yS << log2SbH) + DiagScanOrder[log2SbW][log2SbH][n][1]	
if((AbsLevel[xC][yC] > 0) && (!signHiddenFlag || (n != firstSigScanPosSb)))	
coeff_sign_flag[n]	ae(v)

Tham khảo Bảng 28, biến signHiddenFlag có thể chỉ ra xem SDH có được áp dụng hay không. Biến signHiddenFlag có thể được gọi là signHidden. Ví dụ, khi giá trị

của biến signHiddenFlag là 0, biến signHiddenFlag có thể chỉ ra rằng SDH không được áp dụng, và khi giá trị của biến signHiddenFlag là 1, biến signHiddenFlag có thể chỉ ra rằng SDH được áp dụng. Ví dụ, giá trị của biến signHiddenFlag có thể được thiết đặt dựa trên thông tin cờ được báo hiệu (ví dụ, sh_sign_data_hiding_used_flag hoặc pic_sign_data_hiding_enabled_flag hoặc sps_sign_data_hiding_enabled_flag). Ngoài ra, ví dụ, giá trị của biến signHiddenFlag có thể được thiết đặt dựa trên lastSigScanPosSb và firstSigScanPosSb. Ở đây, lastSigScanPosSb có thể chỉ ra vị trí của hệ số biến đổi đáng kể cuối cùng được tìm kiếm trong khối con/nhóm hệ số tương ứng theo thứ tự quét, và firstSigScanPosSb có thể chỉ ra vị trí của hệ số biến đổi đáng kể thứ nhất được tìm kiếm trong khối con/nhóm hệ số tương ứng theo thứ tự quét. Nói chung, lastSigScanPosSb có thể được nằm trong vùng thành phần tần số cao tương đối so với firstSigScanPosSb. Theo đó, khi lastSigScanPosSb - firstSigScanPosSb là lớn hơn ngưỡng được xác định trước, giá trị signHidden có thể được dẫn xuất là 1 (nghĩa là, SDH được áp dụng), mặt khác, giá trị signHidden có thể được dẫn xuất là 0 (nghĩa là, SDH không được áp dụng). Ở đây, ví dụ, tham khảo Bảng 28, ngưỡng có thể được thiết đặt là 3.

Trong khi đó, khi việc ẩn dữ liệu dấu được kích hoạt trong cú pháp mức cao (VPS, SPS, PPS, cú pháp phần đầu lát, v.v.) hoặc cú pháp mức thấp (cú pháp dữ liệu lát, cú pháp đơn vị lập mã, cú pháp đơn vị biến đổi, v.v.), và slice_ts_residual_coding_disabled_flag là 1, quy trình ẩn dữ liệu dấu của RRC có thể được sử dụng trong lập mã không tổn hao. Theo đó, việc lập mã không tổn hao có thể trở thành bất khả thi do các thiết đặt không chính xác trong thiết bị mã hóa. Mặt khác, khi lập mã tổn hao (nghĩa là, phương pháp lập mã không thể đảo ngược) khác với lập mã không tổn hao được áp dụng, và tín hiệu phần dư mà cho nó việc bỏ qua biến đổi được áp dụng được lập mã với RRC và BDPCM được áp dụng cùng lúc, mặc dù khoảng trong đó giá trị phần dư trở thành 0 xảy ra thường xuyên hơn so với trong trường hợp thông thường do sự chênh lệch giữa các phần dư trong BDPCM, tổn hao lập mã có thể xảy ra bởi vì SDH được thực hiện theo điều kiện ứng dụng SDH. Cụ thể là, ví dụ, khi các hệ số biến đổi đáng kể (dữ liệu phần dư khác không) tồn tại ở các vị trí 0 và 15 trong CG, một cách tương ứng, và giá trị của các hệ số biến đổi ở các vị trí còn lại trong CG là 0, SDH có thể được áp dụng cho CG theo điều kiện ứng dụng SDH được mô tả ở trên,

và do đó, dữ liệu dấu (nghĩa là, việc lập mã của cờ dấu) cho hệ số biến đổi đáng kể thứ nhất của CG có thể được bỏ qua. Theo đó, trong trường hợp này, tính chẵn lẻ của chỉ hai dữ liệu phần dư của CG có thể được điều chỉnh trong bước lượng tử hóa để bỏ qua dữ liệu dấu, và thay vì trường hợp trong đó SDH không được áp dụng, nhiều tổn hao lập mã hơn có thể xảy ra. Trường hợp như vậy có thể xảy ra ngay cả trong các khối mà cho chúng BDPCM không được áp dụng, nhưng do các đặc tính của BDPCM, mức được hạ thấp thông qua sự chênh lệch với phần dư lân cận, sao cho các trường hợp bất lợi có thể xảy ra thường xuyên hơn trong việc áp dụng SDH.

Do đó, để ngăn ngừa SDH và việc lập mã phần dư của slice_ts_residual_coding_disabled_flag = 1 (nghĩa là, việc lập mã các mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại với RRC) không được sử dụng cùng nhau để gây ra sự cố hoặc tổn hao lập mã ngoài ý muốn, sáng chế đề xuất phương án thiết đặt sự phụ thuộc/sự ràng buộc giữa hai công nghệ.

Ví dụ, sáng chế đề xuất phương pháp trong đó slice_ts_residual_coding_disabled_flag là phụ thuộc vào pic_sign_data_hiding_enabled_flag. Cú pháp lập mã phần dư theo phương án này có thể được thể hiện như trong bảng sau đây.

[Bảng 29]

slice_header( ) {	Descriptor
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
(…)	
if(!pic_sign_data_hiding_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(pic_scaling_list_enabled_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

Ở đây, slice_ts_residual_coding_disabled_flag có thể được báo hiệu như cú pháp phần đầu lát, hoặc cú pháp mức cao (High Level Syntax, HLS) ngoài cú pháp phần đầu lát (ví dụ, cú pháp SPS/cú pháp VPS/cú pháp DPS, v.v.), hoặc mức thấp (CU/TU). Ngoài ra, pic_sign_data_hiding_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, hoặc cú pháp mức cao (high-level syntax, HLS) khác khác với cú pháp phần đầu ảnh (ví dụ, cú pháp SPS/cú pháp VPS/cú pháp DPS, v.v.), hoặc mức thấp (CU/TU). Ví dụ, khi pic_sign_data_hiding_enabled_flag được báo hiệu trong cú pháp khác với cú pháp phần đầu ảnh, nó có thể được gọi bằng tên khác. Ví dụ, pic_sign_data_hiding_enabled_flag có thể được thể hiện bởi sps_sign_data_hiding_enabled_flag.

Ngoài ra, sps_sign_data_hiding_enabled_flag có thể là cờ chỉ ra xem việc ẩn dữ liệu dấu có được cho phép hay không. Nghĩa là, ví dụ, sps_sign_data_hiding_enabled_flag có thể chỉ ra xem việc ẩn dữ liệu dấu có được cho phép hay không. Ví dụ, khi giá trị của sps_sign_data_hiding_enabled_flag là 1, sps_sign_data_hiding_enabled_flag có thể chỉ ra rằng việc ẩn dữ liệu dấu được cho phép, và khi giá trị của sps_sign_data_hiding_enabled_flag là 0, sps_sign_data_hiding_enabled_flag có thể chỉ ra rằng việc ẩn dữ liệu dấu không được

cho phép.

Theo Bảng 29 bộc lộ phương án này, slice_ts_residual_coding_disabled_flag có thể được báo hiệu chỉ khi việc ẩn dữ liệu dấu không được cho phép. Ngoài ra, khi việc ẩn dữ liệu dấu được cho phép, slice_ts_residual_coding_disabled_flag có thể không được báo hiệu, giá trị của slice_ts_residual_coding_disabled_flag có thể được suy ra là 0 (lập mã mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại với cú pháp TSRC) hoặc 1 (lập mã mẫu phần dư của khối bỏ qua biến đổi trong lát hiện tại với cú pháp RRC).

Ngoài ra, sáng chế đề xuất phương án trong đó các phương án được mô tả ở trên được kết hợp liên quan tới việc báo hiệu của sh_ts_residual_coding_disabled_flag. Ví dụ, phương án báo hiệu sh_ts_residual_coding_disabled_flag có thể được đề xuất như được thể hiện trong bảng dưới đây.

[Bảng 30]

slice_header( ) {	**Descriptor**
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
(…)	
if(sps_transform_skip_enabled_flag && !ph_dep_quant_enabled_flag && !pic_sign_data_hiding_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(pic_scaling_list_enabled_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

Tham khảo Bảng 30, khi sps_transform_skip_enabled_flag là 1, ph_dep_quant_enabled_flag là 0, và pic_sign_data_hiding_enabled_flag là 0,

sh_ts_residual_coding_disabled_disable_flag có thể được báo hiệu, và mặt khác, sh_ts_residual_coding_disabled_disable_flag có thể không được báo hiệu. Mặt khác, khi sh_ts_residual_coding_disabled_flag không được báo hiệu, sh_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Trong khi đó, phương án của Bảng 31 là ví dụ, và ví dụ trong đó ph_dep_quant_enabled_flag, pic_sign_data_hiding_enabled_flag, và ph_ts_residual_coding_disabled_flag đều được báo hiệu trong cùng HLS (ví dụ, cú pháp phần đầu lát, v.v.) có thể được đề xuất.

Mặt khác, ví dụ, phương án báo hiệu sh_ts_residual_coding_disabled_flag như được thể hiện trong bảng dưới đây có thể được đề xuất.

[Bảng 31]

slice_header( ) {	Descriptor
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
(...)	
if(!ph_dep_quant_enabled_flag && !pic_sign_data_hiding_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(pic_scaling_list_enabled_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

Tham khảo Bảng 31, khi ph_dep_quant_enabled_flag là 0 và pic_sign_data_hiding_enabled_flag là 0, sh_ts_residual_coding_disabled_flag có thể được báo hiệu, và mặt khác, sh_ts_residual_coding_disabled_flag có thể không được báo hiệu. Mặt khác, khi sh_ts_residual_coding_disabled_flag không được báo hiệu, sh_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Trong khi đó, phương án của Bảng 31 là ví dụ, và ví dụ trong đó ph_dep_quant_enabled_flag, pic_sign_data_hiding_enabled_flag, và ph_ts_residual_coding_disabled_flag đều được báo hiệu trong cùng HLS (ví dụ, cú pháp phần đầu lát, v.v.) có thể được đề xuất.

Mặt khác, ví dụ, phương án báo hiệu sh_ts_residual_coding_disabled_flag như được thể hiện trong bảng dưới đây có thể được đề xuất.

[Bảng 32]

slice_header( ) {	Descriptor
picture_header_in_slice_header_flag	u(1)
if(picture_header_in_slice_header_flag)	
picture_header_structure( )	
(…)	
if(!ph_dep_quant_enabled_flag || !pic_sign_data_hiding_enabled_flag)	
slice_ts_residual_coding_disabled_flag	u(1)
if(ph_lmcs_enabled_flag)	
slice_lmcs_enabled_flag	u(1)
if(pic_scaling_list_enabled_flag)	
slice_scaling_list_present_flag	u(1)
if(NumEntryPoints > 0) {	
offset_len_minus1	ue(v)
for(i = 0; i < NumEntryPoints; i++)	
entry_point_offset_minus1[i]	u(v)
}	
if(slice_header_extension_present_flag) {	
slice_header_extension_length	ue(v)
for(i = 0; i < slice_header_extension_length; i++)	
slice_header_extension_data_byte[i]	u(8)
}	
byte_alignment( )	
}	

Tham khảo Bảng 32, khi ph_dep_quant_enabled_flag là 0 hoặc pic_sign_data_hiding_enabled_flag là 0, sh_ts_residual_coding_disabled_flag có thể được báo hiệu, và mặt khác, sh_ts_residual_coding_disabled_flag có thể không được báo hiệu. Mặt khác, khi sh_ts_residual_coding_disabled_flag không được báo hiệu, sh_ts_residual_coding_disabled_flag có thể được suy ra là 0.

Ngoài ra, sáng chế đề xuất phương án trong đó các phần tử cú pháp ph_dep_quant_enabled_flag, pic_sign_data_hiding_enabled_flag và slice_ts_residual_coding_disabled_flag được mô tả ở trên được báo hiệu trong cùng cú pháp mức cao hoặc cú pháp mức thấp.

Ví dụ, phương án trong đó ph_dep_quant_enabled_flag, pic_sign_data_hiding_enabled_flag, và slice_ts_residual_coding_disabled_flag đều được báo hiệu trong cú pháp phần đầu ảnh có thể được đề xuất như được thể hiện trong bảng dưới đây.

[Bảng 33]

picture_header_structure( ) {	**Descriptor**
(...)	
ph_ts_residual_coding_disabled_flag	u(1)
if(!ph_ts_residual_coding_disabled_flag){	
if(sps_dep_quant_enabled_flag)	
ph_dep_quant_enabled_flag	u(1)
if(sps_sign_data_hiding_enabled_flag && !ph_dep_quant_enabled_flag)	
pic_sign_data_hiding_enabled_flag	u(1)
}	
(...)	
}	

Trong trường hợp này, slice_ts_residual_coding_disabled_flag có thể được gọi là ph_ts_residual_coding_disabled_flag.

Tham khảo Bảng 33, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh, và khi giá trị của ph_ts_residual_coding_disabled_flag là 0, nếu giá trị của sps_dep_quant_enabled_flag là 1, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Ngoài ra, khi giá trị của ph_ts_residual_coding_disabled_flag là 0, nếu giá trị của sps_sign_data_hiding_enabled_flag là 1 và ph_dep_quant_enabled_flag là 0, pic_sign_data_hiding_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Trong khi đó, ví dụ, khi giá trị của ph_ts_residual_coding_disabled_flag là 1, ph_dep_quant_enabled_flag và pic_sign_data_hiding_enabled_flag có thể không được báo hiệu.

Mặt khác, ví dụ, phương án trong đó ph_dep_quant_enabled_flag, pic_sign_data_hiding_enabled_flag, và slice_ts_residual_coding_disabled_flag đều được báo hiệu trong cú pháp phần đầu ảnh như được thể hiện trong bảng dưới đây có thể được đề xuất.

[Bảng 34]

picture_header_structure( ) {	Descriptor
(...)	
if(sps_transform_skip_enabled_flag)	
ph_ts_residual_coding_disabled_flag	u(1)
if(!ph_ts_residual_coding_disabled_flag){	
if (sps_dep_quant_enabled_flag)	
ph_dep_quant_enabled_flag	u(1)
if (sps_sign_data_hiding_enabled_flag && !ph_dep_quant_enabled_flag)	
pic_sign_data_hiding_enabled_flag	u(1)
}	
(...)	
}	

Tham khảo Bảng 34, khi giá trị của sps_transform_skip_enabled_flag là 1, ph_ts_residual_coding_disabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Ngoài ra, ví dụ, khi giá trị của sps_transform_skip_enabled_flag là 0, ph_ts_residual_coding_disabled_flag có thể không được báo hiệu. Khi ph_ts_residual_coding_disabled_flag không được báo hiệu, ph_ts_residual_coding_disabled_flag có thể được suy ra là 0 trong thiết bị giải mã.

Ngoài ra, tham khảo Bảng 34, khi giá trị của ph_ts_residual_coding_disabled_flag là 0 và giá trị của sps_dep_quant_enabled_flag là 1, ph_dep_quant_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Ngoài ra, khi giá trị của ph_ts_residual_coding_disabled_flag là 0, nếu giá trị của sps_sign_data_hiding_enabled_flag là 1 và ph_dep_quant_enabled_flag là 0, pic_sign_data_hiding_enabled_flag có thể được báo hiệu trong cú pháp phần đầu ảnh. Trong khi đó, ví dụ, khi giá trị của ph_ts_residual_coding_disabled_flag là 1, ph_dep_quant_enabled_flag và pic_sign_data_hiding_enabled_flag có thể không được báo hiệu. Ngoài ra, ví dụ, khi ph_dep_quant_enabled_flag không được báo hiệu, ph_dep_quant_enabled_flag có thể được suy ra là 0 trong thiết bị giải mã. Ngoài ra, ví dụ, khi pic_sign_data_hiding_enabled_flag không được báo hiệu, pic_sign_data_hiding_enabled_flag có thể được suy ra là 0 trong thiết bị giải mã.

Trong khi đó, các phương án được mô tả ở trên là các ví dụ, ví dụ trong đó ph_ts_residual_coding_disabled_flag, ph_dep_quant_enabled_flag và pic_sign_data_hiding_enabled_flag được báo hiệu trong cú pháp mức cao (VPS, SPS, PPS, cú pháp phần đầu lát, v.v.) hoặc cú pháp mức thấp (cú pháp dữ liệu lát, cú pháp đơn vị lập mã, v.v.) khác với cú pháp phần đầu ảnh có thể được đề xuất.

Trong khi đó, như được mô tả ở trên, thông tin (phần tử cú pháp) trong bảng cú pháp được bộc lộ trong sáng chế có thể được gồm trong thông tin hình ảnh/viđeo, và có thể được tạo cấu hình/được mã hóa trong thiết bị mã hóa và được truyền tới thiết bị giải mã dưới dạng luồng bit. Thiết bị giải mã có thể phân tích cú pháp/giải mã thông tin (phần tử cú pháp) trong bảng cú pháp tương ứng. Thiết bị giải mã có thể thực hiện quy trình xây dựng lại khối/hình ảnh/viđeo dựa trên thông tin được giải mã.

Fig.8 minh họa sơ lược phương pháp mã hóa hình ảnh được thực hiện bởi thiết bị mã hóa theo phần bộc lộ này. Phương pháp được bộc lộ trong Fig.8 có thể được thực hiện bởi thiết bị mã hóa được bộc lộ trong Fig.2. Cụ thể là, ví dụ, S800 đến S830 của Fig.8 có thể được thực hiện bằng bộ mã hóa entrôpi của thiết bị mã hóa. Ngoài ra, mặc dù không được minh họa, quy trình dẫn xuất mẫu dự đoán có thể được thực hiện bởi bộ dự đoán của thiết bị mã hóa, quy trình dẫn xuất mẫu phần dư cho khối hiện tại dựa trên mẫu gốc và mẫu dự đoán cho khối hiện tại có thể được thực hiện bởi bộ trừ của thiết bị mã hóa, và quy trình tạo ra mẫu được xây dựng lại và ảnh được xây dựng lại cho khối hiện tại dựa trên mẫu phần dư và mẫu dự đoán cho khối hiện tại có thể được thực hiện bởi bộ cộng của thiết bị mã hóa.

Thiết bị mã hóa mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không (S800). Thiết bị mã hóa có thể mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không. Thông tin hình ảnh có thể gồm cờ được cho phép ẩn dữ liệu dấu. Ví dụ, thiết bị mã hóa có thể xác định xem việc ẩn dữ liệu dấu có được cho phép cho các khối của các ảnh trong trình tự hay không, và có thể mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể là cờ về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem việc ẩn dữ liệu dấu có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem việc ẩn dữ liệu dấu có được cho phép cho các khối của các ảnh trong trình tự hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem cờ được sử dụng ẩn dữ liệu dấu thể hiện xem việc ẩn dữ liệu dấu có được sử dụng cho lát hiện tại hay không có thể có mặt hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có giá trị là 1 có thể thể hiện rằng việc ẩn dữ liệu dấu được cho phép, và cờ được cho phép ẩn dữ liệu dấu có giá trị

là 0 có thể thể hiện rằng việc ẩn dữ liệu dấu không được cho phép. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có giá trị là 1 có thể thể hiện rằng cờ được sử dụng ẩn dữ liệu dấu có thể có mặt, và cờ được cho phép ẩn dữ liệu dấu có giá trị là 0 có thể thể hiện rằng cờ được sử dụng ẩn dữ liệu dấu là không có mặt. Ngoài ra, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể được báo hiệu trong cú pháp SPS. Mặt khác, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể được báo hiệu trong cú pháp phần đầu ảnh hoặc cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép ẩn dữ liệu dấu có thể là sps_sign_data_hiding_enabled_flag.

Thiết bị mã hóa mã hóa cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được cho phép hay không dựa trên cờ được cho phép ẩn dữ liệu dấu (S810). Thông tin hình ảnh có thể gồm cờ được cho phép TSRC.

Ví dụ, thiết bị mã hóa có thể mã hóa cờ được cho phép TSRC dựa trên cờ được cho phép ẩn dữ liệu dấu. Ví dụ, cờ được cho phép TSRC có thể được mã hóa dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), cờ được cho phép TSRC có thể được mã hóa. Nói cách khác, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), cờ được cho phép TSRC có thể được báo hiệu. Ngoài ra, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, cờ được cho phép TSRC có thể không được mã hóa, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0 trong thiết bị giải mã. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0 trong thiết bị giải mã.

Ở đây, ví dụ, cờ được cho phép TSRC có thể là cờ về việc liệu TSRC có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép TSRC có thể là cờ thể hiện xem TSRC có được cho phép cho các khối trong lát hay không. Ví dụ, cờ được cho phép TSRC có giá trị là 1 có thể thể hiện rằng TSRC là không được cho phép, và cờ được cho phép TSRC có giá trị là 0 có thể thể hiện rằng TSRC được cho phép. Ngoài ra, ví dụ, cờ được cho phép TSRC có thể được báo hiệu trong cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép TSRC có thể là sh_ts_residual_coding_disabled_flag được mô tả

ở trên.

Cùng lúc đó, ví dụ, thiết bị mã hóa có thể xác định xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không cho các khối của các ảnh trong trình tự, và có thể mã hóa cờ được cho phép lượng tử hóa phụ thuộc cho việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Thông tin hình ảnh có thể gồm cờ được cho phép lượng tử hóa phụ thuộc. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể là cờ cho việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể thể hiện xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể thể hiện xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không cho các khối của các ảnh trong trình tự. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể thể hiện xem cờ được sử dụng việc lượng tử hóa phụ thuộc thể hiện xem việc lượng tử hóa phụ thuộc có được sử dụng cho lát hiện tại hay không có thể có mặt khay không. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 1 có thể thể hiện rằng việc lượng tử hóa phụ thuộc được cho phép, và cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 0 có thể thể hiện rằng việc lượng tử hóa phụ thuộc là không được cho phép. Ngoài ra, ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể được báo hiệu trong cú pháp SPS hoặc cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép lượng tử hóa phụ thuộc có thể là sps_dep_quant_enabled_flag được mô tả ở trên. sps_dep_quant_enabled_flag có thể được gọi là sh_dep_quant_enabled_flag, sh_dep_quant_used_flag, hoặc ph_dep_quant_enabled_flag.

Ngoài ra, ví dụ, thiết bị mã hóa có thể mã hóa cờ được cho phép bỏ qua biến đổi cho việc liệu việc bỏ qua biến đổi có được cho phép hay không. Thông tin hình ảnh có thể gồm cờ được cho phép bỏ qua biến đổi. Ví dụ, thiết bị mã hóa có thể xác định xem việc bỏ qua biến đổi có được cho phép cho các khối của ảnh trong trình tự hay không, và có thể mã hóa cờ được cho phép bỏ qua biến đổi cho việc liệu việc bỏ qua biến đổi có được cho phép hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có thể là cờ cho việc liệu việc bỏ qua biến đổi có được cho phép hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có thể thể hiện xem việc bỏ qua biến đổi có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép bỏ qua biến đổi có thể thể hiện xem việc bỏ qua biến đổi có được cho phép cho các khối của các ảnh trong trình tự hay không. Ví dụ, cờ được

cho phép bỏ qua biến đổi có thể thể hiện xem cờ bỏ qua biến đổi có thể có mặt hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có giá trị là 1 có thể thể hiện rằng việc bỏ qua biến đổi được cho phép, và cờ được cho phép bỏ qua biến đổi có giá trị là 0 có thể thể hiện rằng việc bỏ qua biến đổi không được cho phép. Nghĩa là, ví dụ, cờ được cho phép bỏ qua biến đổi có giá trị là 1 có thể thể hiện rằng cờ bỏ qua biến đổi có thể có mặt, và cờ được cho phép bỏ qua biến đổi có giá trị là 0 có thể thể hiện rằng cờ bỏ qua biến đổi không có mặt. Ngoài ra, ví dụ, cờ được cho phép bỏ qua biến đổi có thể được báo hiệu tới cú pháp tập thông số trình tự (SPS). Phần tử cú pháp của cờ được cho phép bỏ qua biến đổi có thể là sps_transform_skip_enabled_flag được mô tả ở trên.

Ngoài ra, ví dụ, cờ được cho phép TSRC có thể được mã hóa dựa trên cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép lượng tử hóa phụ thuộc, và/hoặc cờ được cho phép bỏ qua biến đổi. Ví dụ, cờ được cho phép TSRC có thể được mã hóa dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0, cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 0, và cờ được cho phép bỏ qua biến đổi có giá trị là 1. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 0 (nghĩa là, cờ được cho phép lượng tử hóa phụ thuộc thể hiện rằng việc lượng tử hóa phụ thuộc không được cho phép), và giá trị của cờ được cho phép bỏ qua biến đổi là 1 (nghĩa là, cờ được cho phép bỏ qua biến đổi thể hiện rằng việc bỏ qua biến đổi được cho phép), cờ được cho phép TSRC có thể được mã hóa (hoặc được báo hiệu). Ngoài ra, ví dụ, khi giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 1, cờ được cho phép TSRC có thể không được mã hóa, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0 trong thiết bị giải mã. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 1, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0 trong thiết bị giải mã. Ngoài ra, ví dụ, khi giá trị của cờ được cho phép bỏ qua biến đổi là 0, cờ được cho phép TSRC có thể không được mã hóa, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép bỏ qua biến đổi là 0, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0.

Thiết bị mã hóa mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ được

cho phép TSRC (S820). Thiết bị mã hóa có thể mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC.

Ví dụ, thiết bị mã hóa có thể xác định cú pháp lập mã phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC. Ví dụ, thiết bị mã hóa có thể xác định cú pháp lập mã phần dư cho khối hiện tại như một trong số cú pháp lập mã phần dư thông thường (RRC) và cú pháp lập mã phần dư bỏ qua biến đổi (TSRC) dựa trên cờ được cho phép TSRC. Cú pháp RRC có thể thể hiện cú pháp theo RRC, và cú pháp TSRC có thể thể hiện cú pháp theo TSRC.

Ví dụ, dựa trên cờ được cho phép TSRC có giá trị là 1, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư thông thường (RRC). Trong trường hợp này, ví dụ, cờ bỏ qua biến đổi về việc liệu khối hiện tại có phải là khối bỏ qua biến đổi hay không có thể được mã hóa, và giá trị của cờ bỏ qua biến đổi có thể là 1. Ví dụ, thông tin hình ảnh có thể gồm cờ bỏ qua biến đổi cho khối hiện tại. Cờ bỏ qua biến đổi có thể thể hiện xem khối hiện tại có phải là khối bỏ qua biến đổi hay không. Nghĩa là, cờ bỏ qua biến đổi có thể thể hiện xem việc biến đổi đã được áp dụng cho các hệ số biến đổi của khối hiện tại hay chưa. Phần tử cú pháp biểu diễn cờ bỏ qua biến đổi có thể là transform_skip_flag như được mô tả ở trên. Ví dụ, nếu giá trị của cờ bỏ qua biến đổi là 1, cờ bỏ qua biến đổi có thể thể hiện rằng việc biến đổi đã không được áp dụng cho khối hiện tại (nghĩa là, việc biến đổi được bỏ qua), trong khi nếu giá trị của cờ bỏ qua biến đổi là 0, cờ bỏ qua biến đổi có thể thể hiện rằng việc biến đổi đã được áp dụng cho khối hiện tại. Ví dụ, nếu khối hiện tại là khối bỏ qua biến đổi, giá trị của cờ bỏ qua biến đổi cho khối hiện tại có thể là 1.

Ngoài ra, ví dụ, dựa trên cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư bỏ qua biến đổi (TSRC). Ngoài ra, ví dụ, cờ bỏ qua biến đổi về việc liệu khối hiện tại có phải là khối bỏ qua biến đổi hay không có thể được mã hóa, và dựa trên cờ bỏ qua biến đổi có giá trị là 1 và cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư bỏ qua biến đổi (TSRC). Ngoài ra, ví dụ, cờ bỏ qua biến đổi về việc liệu khối hiện tại có phải là khối bỏ qua biến đổi hay không có thể được mã hóa, và dựa trên cờ bỏ qua biến đổi có giá trị là 0 và cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác

định như cú pháp lập mã phần dư thông thường (RRC).

Sau đó, ví dụ, thiết bị mã hóa có thể mã hóa thông tin phần dư của cú pháp lập mã phần dư được xác định cho khối hiện tại. Thiết bị mã hóa có thể dẫn xuất mẫu phần dư cho khối hiện tại, và có thể mã hóa thông tin phần dư của cú pháp lập mã phần dư được xác định cho mẫu phần dư của khối hiện tại. Ví dụ, thông tin phần dư của cú pháp lập mã phần dư thông thường (RRC) có thể được mã hóa dựa trên cờ được cho phép TSRC có giá trị là 1, và thông tin phần dư của cú pháp TSRC có thể được mã hóa dựa trên cờ được cho phép TSRC có giá trị là 0. Thông tin hình ảnh có thể gồm thông tin phần dư.

Ví dụ, thiết bị mã hóa có thể xác định xem thực hiện dự đoán liên hay dự đoán nội trên khối hiện tại, và có thể xác định chế độ dự đoán liên cụ thể hoặc chế độ dự đoán nội cụ thể dựa trên chi phí RD. Theo chế độ được xác định, thiết bị mã hóa có thể dẫn xuất các mẫu dự đoán cho khối hiện tại, và có thể dẫn xuất các mẫu phần dư cho khối hiện tại bằng việc trừ các mẫu dự đoán từ các mẫu gốc cho khối hiện tại.

Sau đó, ví dụ, thiết bị mã hóa có thể dẫn xuất các hệ số biến đổi của khối hiện tại dựa trên các mẫu phần dư. Ví dụ, thiết bị mã hóa có thể xác định xem việc biến đổi có được áp dụng cho khối hiện tại hay không. Nghĩa là, thiết bị mã hóa có thể xác định xem việc biến đổi có được áp dụng cho các mẫu phần dư của khối hiện tại hay không. Thiết bị mã hóa có thể xác định xem việc biến đổi có được áp dụng cho khối hiện tại hay không xem xét đến hiệu quả lập mã. Ví dụ, thiết bị mã hóa có thể xác định rằng việc biến đổi không được áp dụng cho khối hiện tại. Khối mà cho nó việc biến đổi không được áp dụng có thể được biểu diễn là khối bỏ qua biến đổi. Nghĩa là, ví dụ, khối hiện tại có thể là khối bỏ qua biến đổi.

Nếu việc biến đổi không được áp dụng cho khối hiện tại, nghĩa là, nếu việc biến đổi không được áp dụng cho các mẫu phần dư, thiết bị mã hóa có thể dẫn xuất các mẫu phần dư được dẫn xuất như các hệ số biến đổi hiện tại. Ngoài ra, nếu việc biến đổi được áp dụng cho khối hiện tại, nghĩa là, nếu việc biến đổi được áp dụng cho các mẫu phần dư, thiết bị mã hóa có thể dẫn xuất các hệ số biến đổi bằng việc thực hiện việc biến đổi cho các mẫu phần dư. Khối hiện tại có thể gồm nhiều khối con hoặc nhóm hệ số (coefficient group, CG). Ngoài ra, kích thước của khối con của khối hiện tại có thể là kích thước 4×4 hoặc kích thước 2×2. Nghĩa là, khối con của khối hiện tại tối đa có thể

gồm 16 hệ số biến đổi khác không hoặc 4 hệ số biến đổi khác không. Ở đây, khối hiện tại có thể là khối lập mã (CB) hoặc khối biến đổi (TB). Ngoài ra, hệ số biến đổi có thể được biểu diễn là hệ số phần dư.

Cùng lúc đó, thiết bị mã hóa có thể xác định liệu việc lượng tử hóa phụ thuộc có được áp dụng cho khối hiện tại hay không. Ví dụ, nếu việc lượng tử hóa phụ thuộc được áp dụng cho khối hiện tại, thiết bị mã hóa có thể dẫn xuất các hệ số biến đổi của khối hiện tại bằng việc thực hiện quy trình lượng tử hóa phụ thuộc cho các hệ số biến đổi. Ví dụ, nếu việc lượng tử hóa phụ thuộc được áp dụng cho khối hiện tại, thiết bị mã hóa có thể cập nhật trạng thái (Qstate) cho việc lượng tử hóa phụ thuộc dựa trên mức hệ số của hệ số biến đổi ngay trước hệ số biến đổi hiện tại theo thứ tự quét, có thể dẫn xuất mức hệ số của hệ số biến đổi hiện tại dựa trên trạng thái được cập nhật và các phần tử cú pháp cho hệ số biến đổi hiện tại, và có thể dẫn xuất hệ số biến đổi hiện tại bằng việc lượng tử hóa mức hệ số được dẫn xuất. Ví dụ, hệ số biến đổi hiện tại có thể được lượng tử hóa dựa trên thông số lượng tử hóa cho mức được xây dựng lại của hệ số biến đổi hiện tại trong bộ lượng tử hóa vô hướng cho trạng thái được cập nhật.

Ví dụ, nếu cú pháp lập mã phần dư cho khối hiện tại được xác định như cú pháp RRC, thiết bị mã hóa có thể mã hóa thông tin phần dư của cú pháp RRC cho khối hiện tại. Ví dụ, thông tin phần dư của cú pháp RRC có thể gồm các phần tử cú pháp được bộc lộ trong Bảng 2 như được mô tả ở trên.

Ví dụ, thông tin phần dư của cú pháp RRC có thể gồm các phần tử cú pháp cho hệ số biến đổi của khối hiện tại. Ở đây, hệ số biến đổi có thể được biểu diễn là hệ số phần dư.

Ví dụ, các phần tử cú pháp có thể gồm các phần tử cú pháp, chẳng hạn như last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, sb_coded_flag, sig_coeff_flag, par_level_flag, abs_level_gtX_flag (ví dụ, abs_level_gtx_flag[n][0] và/hoặc abs_level_gtx_flag[n][1]), abs_remainder, dec_abs_level, và/hoặc coeff_sign_flag.

Cụ thể là, ví dụ, các phần tử cú pháp có thể gồm thông tin vị trí thể hiện vị trí của hệ số biến đổi khác không cuối cùng trong mảng hệ số phần dư của khối hiện tại. Nghĩa là, các phần tử cú pháp có thể gồm thông tin vị trí thể hiện vị trí của hệ số biến

đổi khác không cuối cùng theo thứ tự quét của khối hiện tại. Thông tin vị trí có thể gồm thông tin thể hiện tiền tố trong vị trí cột của hệ số biến đổi khác không cuối cùng, thông tin thể hiện tiền tố trong vị trí hàng của hệ số biến đổi khác không cuối cùng, thông tin thể hiện hậu tố của vị trí cột của hệ số biến đổi khác không cuối cùng, và thông tin thể hiện hậu tố trong vị trí hàng của hệ số biến đổi khác không cuối cùng. Các phần tử cú pháp cho thông tin vị trí có thể là last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Cùng lúc đó, hệ số biến đổi khác không có thể được gọi là hệ số đáng kể.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm cờ khối con được lập mã thể hiện xem khối con hiện tại của khối hiện tại gồm hệ số biến đổi khác không, cờ hệ số đáng kể thể hiện xem hệ số biến đổi của khối hiện tại là hệ số biến đổi khác không, cờ mức hệ số thứ nhất cho việc liệu mức hệ số cho hệ số biến đổi có lớn hơn giá trị ngưỡng thứ nhất hay không, cờ mức tính chẵn lẻ cho tính chẵn lẻ của mức hệ số, và/hoặc cờ mức hệ số thứ hai về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ hai hay không. Ở đây, cờ khối con được lập mã có thể là sb_coded_flag hoặc coded_sub_block_flag, cờ hệ số đáng kể có thể là sig_coeff_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag hoặc abs_level_gtx_flag, cờ mức tính chẵn lẻ có thể là par_level_flag, và cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm thông tin liên quan đến giá trị hệ số cho giá trị hệ số biến đổi của khối hiện tại. Thông tin liên quan đến giá trị hệ số có thể là abs_remainder và/hoặc dec_abs_level.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm cờ dấu thể hiện dấu của hệ số biến đổi. Cờ dấu có thể là coeff_sign_flag.

Trong khi đó, ví dụ, khi việc ẩn dữ liệu dấu được áp dụng cho khối hiện tại, cờ dấu của hệ số biến đổi đáng kể thứ nhất của nhóm hệ số (CG) hiện tại trong khối hiện tại có thể không được mã hóa và báo hiệu. Nghĩa là, ví dụ, khi việc ẩn dữ liệu dấu được áp dụng cho khối hiện tại, các phần tử cú pháp có thể không gồm cờ dấu biểu diễn dấu của hệ số biến đổi đáng kể thứ nhất. Trong khi đó, ví dụ, việc liệu việc ẩn dữ liệu dấu có được áp dụng cho khối hiện tại hay không có thể được dẫn xuất dựa trên cờ được cho phép ẩn dữ liệu dấu và/hoặc vị trí của hệ số biến đổi đáng kể thứ nhất và vị trí của hệ số

biến đổi đáng kể cuối cùng của CG hiện tại. Ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, và giá trị thu được bằng cách trừ vị trí của hệ số biến đổi đáng kể thứ nhất từ vị trí của hệ số biến đổi đáng kể cuối cùng là lớn hơn 3 (nghĩa là, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, và số lượng của các hệ số biến đổi đáng kể trong CG hiện tại là lớn hơn 3), việc ẩn dữ liệu dấu có thể được áp dụng cho CG hiện tại của khối hiện tại.

Ngoài ra, ví dụ, nếu cú pháp lập mã phần dư cho khối hiện tại được xác định như cú pháp TSRC, thiết bị mã hóa có thể mã hóa thông tin phần dư của cú pháp TSRC cho khối hiện tại. Ví dụ, thông tin phần dư của cú pháp TSRC có thể gồm các phần tử cú pháp được bộc lộ trong Bảng 3 như được mô tả ở trên.

Ví dụ, thông tin phần dư của cú pháp TSRC có thể gồm các phần tử cú pháp cho hệ số biến đổi của khối hiện tại. Ở đây, hệ số biến đổi có thể được biểu diễn là hệ số phần dư.

Ví dụ, các phần tử cú pháp có thể gồm các phần tử cú pháp được lập mã ngữ cảnh cho hệ số biến đổi và/hoặc các phần tử cú pháp được lập mã đi vòng qua. Các phần tử cú pháp có thể gồm các phần tử cú pháp, chẳng hạn như sig_coeff_flag, coeff_sign_flag, par_level_flag, abs_level_gtX_flag (ví dụ, abs_level_gtx_flag[n][0], abs_level_gtx_flag[n][1], abs_level_gtx_flag[n][2], abs_level_gtx_flag[n][3], và/hoặc abs_level_gtx_flag[n][4]), abs_remainder, và/hoặc coeff_sign_flag.

Ví dụ, các phần tử cú pháp được lập mã ngữ cảnh cho hệ số biến đổi có thể gồm cờ hệ số đáng kể thể hiện xem hệ số biến đổi là hệ số biến đổi khác không, cờ dấu thể hiện dấu cho hệ số biến đổi, cờ mức hệ số thứ nhất cho việc liệu mức hệ số cho hệ số biến đổi có lớn hơn giá trị ngưỡng thứ nhất hay không, và/hoặc cờ mức tính chẵn lẻ cho tính chẵn lẻ của mức biến đổi cho hệ số biến đổi. Ngoài ra, ví dụ, các phần tử cú pháp được lập mã ngữ cảnh có thể gồm cờ mức hệ số thứ hai cho việc liệu mức hệ số của hệ số biến đổi. Ngoài ra, ví dụ, các phần tử cú pháp được lập mã ngữ cảnh có thể gồm cờ mức hệ số thứ hai về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ hai hay không, cờ mức hệ số thứ ba về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ ba hay không, cờ mức hệ số thứ tư về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ tư hay không, và/hoặc cờ mức hệ số thứ năm về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ năm hay không.

Ở đây, cờ hệ số đáng kể có thể là sig_coeff_flag, cờ dấu có thể là ceff_sign_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag, và cờ mức tính chẵn lẻ có thể là par_level_flag. Ngoài ra, cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag, cờ mức hệ số thứ ba có thể là abs_level_gt5_flag hoặc abs_level_gtx_flag, cờ mức hệ số thứ tư có thể là abs_level_gt7_flag hoặc abs_level_gtx_flag, và cờ mức hệ số thứ năm có thể là abs_level_gt9_flag hoặc abs_level_gtx_flag.

Ngoài ra, ví dụ, các phần tử cú pháp được lập mã đi vòng qua cho hệ số biến đổi có thể gồm thông tin mức hệ số cho giá trị của hệ số biến đổi (hoặc mức hệ số) và/hoặc cờ dấu thể hiện dấu cho hệ số biến đổi. Thông tin mức hệ số có thể là abs_remainder và/hoặc dec_abs_level, và cờ dấu có thể là ceff_sign_flag.

Thiết bị mã hóa tạo ra luồng bit gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC và thông tin phần dư (S830). Ví dụ, thiết bị mã hóa có thể xuất ra thông tin hình ảnh gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC và thông tin phần dư dưới dạng luồng bit. Luồng bit có thể gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC và thông tin phần dư. Ngoài ra, luồng bit có thể còn gồm cờ được cho phép lượng tử hóa phụ thuộc và/hoặc cờ được cho phép bỏ qua biến đổi.

Cùng lúc đó, thông tin hình ảnh có thể gồm thông tin liên quan đến việc dự đoán cho khối hiện tại. Thông tin liên quan đến việc dự đoán có thể gồm thông tin chế độ dự đoán cho chế độ dự đoán liên hoặc chế độ dự đoán nội được thực hiện trên khối hiện tại.

Cùng lúc đó, luồng bit có thể được truyền tới thiết bị giải mã thông qua mạng hoặc phương tiện lưu trữ (kỹ thuật số). Ở đây, mạng có thể gồm mạng phát rộng và/hoặc mạng truyền thông, và phương tiện lưu trữ kỹ thuật số có thể gồm phương tiện lưu trữ khác nhau, chẳng hạn như USB, SD, CD, DVD, Blu-ray, HDD, và SSD.

Fig.9 minh họa sơ lược thiết bị mã hóa để thực hiện phương pháp mã hóa hình ảnh theo phần bộc lộ này. Phương pháp được bộc lộ trong Fig.8 có thể được thực hiện bởi thiết bị mã hóa được bộc lộ trong Fig.9. Cụ thể là, ví dụ, bộ mã hóa entrôpi của thiết bị mã hóa của Fig.9 có thể thực hiện từ S800 đến S830 của Fig.8. Ngoài ra, mặc dù không được minh họa, quy trình dẫn xuất mẫu dự đoán có thể được thực hiện bởi bộ dự đoán của thiết bị mã hóa, quy trình của mẫu phần dư cho khối hiện tại dựa trên mẫu gốc

và mẫu dự đoán cho khối hiện tại có thể được thực hiện bởi bộ trừ của thiết bị mã hóa, và quy trình tạo ra mẫu được xây dựng lại và ảnh được xây dựng lại cho khối hiện tại dựa trên mẫu phần dư và mẫu dự đoán cho khối hiện tại có thể được thực hiện bởi bộ cộng của thiết bị mã hóa.

Fig.10 minh họa sơ lược phương pháp giải mã hình ảnh được thực hiện bởi thiết bị giải mã theo phần bộc lộ này. Phương pháp được bộc lộ trong Fig.10 có thể được thực hiện bởi thiết bị giải mã được bộc lộ trong Fig.3. Cụ thể là, ví dụ, S1000 đến S1020 của Fig.10 có thể được thực hiện bởi bộ giải mã entrôpi của thiết bị giải mã, S1030 của Fig.10 có thể được thực hiện bởi bộ xử lý phần dư của thiết bị giải mã, và S1040 có thể được thực hiện bởi bộ cộng của thiết bị giải mã. Ngoài ra, mặc dù không được minh họa, quy trình nhận thông tin dự đoán cho khối hiện tại có thể được thực hiện bởi bộ giải mã entrôpi của thiết bị giải mã, và quy trình dẫn xuất mẫu dự đoán của khối hiện tại có thể được thực hiện bởi bộ dự đoán của thiết bị giải mã.

Thiết bị giải mã thu nhận cờ được cho phép ẩn dữ liệu dấu (S1000). Thiết bị giải mã có thể thu nhận thông tin hình ảnh gồm cờ được cho phép ẩn dữ liệu dấu thông qua luồng bit. Thông tin hình ảnh có thể gồm cờ được cho phép ẩn dữ liệu dấu. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể là cờ về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem việc ẩn dữ liệu dấu có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem việc ẩn dữ liệu dấu có được cho phép cho các khối của các ảnh trong trình tự hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể thể hiện xem cờ được sử dụng ẩn dữ liệu dấu thể hiện xem việc ẩn dữ liệu dấu có được sử dụng cho lát hiện tại hay không có thể có mặt hay không. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có giá trị là 1 có thể thể hiện rằng việc ẩn dữ liệu dấu được cho phép, và cờ được cho phép ẩn dữ liệu dấu có giá trị là 0 có thể thể hiện rằng việc ẩn dữ liệu dấu không được cho phép. Ví dụ, cờ được cho phép ẩn dữ liệu dấu có giá trị là 1 có thể thể hiện rằng cờ được sử dụng ẩn dữ liệu dấu có thể có mặt, và cờ được cho phép ẩn dữ liệu dấu có giá trị là 0 có thể thể hiện rằng cờ được sử dụng ẩn dữ liệu dấu là không có mặt. Ngoài ra, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể được báo hiệu trong cú pháp SPS. Mặt khác, ví dụ, cờ được cho phép ẩn dữ liệu dấu có thể được báo hiệu trong cú pháp phần đầu ảnh hoặc cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép ẩn dữ liệu dấu có thể là

sps_sign_data_hiding_enabled_flag.

Thiết bị giải mã thu nhận cờ được cho phép lập mã phần dư bỏ qua biến đổi (TSRC) dựa trên cờ được cho phép ẩn dữ liệu dấu (S1010). Thông tin hình ảnh có thể gồm cờ được cho phép TSRC.

Ví dụ, thiết bị giải mã có thể thu nhận cờ được cho phép TSRC dựa trên cờ được cho phép ẩn dữ liệu dấu. Ví dụ, cờ được cho phép TSRC có thể được thu nhận dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), cờ được cho phép TSRC có thể được thu nhận. Nói cách khác, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), cờ được cho phép TSRC có thể được báo hiệu. Ngoài ra, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, cờ được cho phép TSRC có thể không được thu nhận, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0.

Ở đây, ví dụ, cờ được cho phép TSRC có thể là cờ về việc liệu TSRC có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép TSRC có thể là cờ thể hiện xem TSRC có được cho phép cho các khối trong lát hay không. Ví dụ, cờ được cho phép TSRC có giá trị là 1 có thể thể hiện rằng TSRC là không được cho phép, và cờ được cho phép TSRC có giá trị là 0 có thể thể hiện rằng TSRC được cho phép. Ngoài ra, ví dụ, cờ được cho phép TSRC có thể được báo hiệu trong cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép TSRC có thể là sh_ts_residual_coding_disabled_flag được mô tả ở trên.

Cùng lúc đó, ví dụ, thiết bị giải mã có thể thu nhận cờ được cho phép lượng tử hóa phụ thuộc. Thiết bị giải mã có thể thu nhận thông tin hình ảnh gồm cờ được cho phép lượng tử hóa phụ thuộc thông qua luồng bit. Thông tin hình ảnh có thể gồm cờ được cho phép lượng tử hóa phụ thuộc. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể là cờ cho việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể thể hiện xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép lượng tử hóa

phụ thuộc có thể thể hiện xem liệu việc lượng tử hóa phụ thuộc có được cho phép hay không cho các khối của các ảnh trong trình tự. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể thể hiện xem cờ được sử dụng việc lượng tử hóa phụ thuộc thể hiện xem việc lượng tử hóa phụ thuộc có được sử dụng cho lát hiện tại hay không có thể có mặt khay không. Ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 1 có thể thể hiện rằng việc lượng tử hóa phụ thuộc được cho phép, và cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 0 có thể thể hiện rằng việc lượng tử hóa phụ thuộc là không được cho phép. Ngoài ra, ví dụ, cờ được cho phép lượng tử hóa phụ thuộc có thể được báo hiệu trong cú pháp SPS hoặc cú pháp phần đầu lát. Phần tử cú pháp của cờ được cho phép lượng tử hóa phụ thuộc có thể là sps_dep_quant_enabled_flag được mô tả ở trên. sps_dep_quant_enabled_flag có thể được gọi là sh_dep_quant_enabled_flag, sh_dep_quant_used_flag, hoặc ph_dep_quant_enabled_flag.

Ngoài ra, ví dụ, thiết bị giải mã có thể thu nhận cờ được cho phép bỏ qua biến đổi. Thiết bị giải mã có thể thông tin hình ảnh gồm cờ được cho phép bỏ qua biến đổi thông qua luồng bit. Thông tin hình ảnh có thể gồm cờ được cho phép bỏ qua biến đổi. Ở đây, khối hiện tại có thể là khối lập mã (CB) hoặc khối biến đổi (TB). Ví dụ, cờ được cho phép bỏ qua biến đổi có thể là cờ cho việc liệu việc bỏ qua biến đổi có được cho phép hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có thể thể hiện xem việc bỏ qua biến đổi có được cho phép hay không. Nghĩa là, ví dụ, cờ được cho phép bỏ qua biến đổi có thể thể hiện xem việc bỏ qua biến đổi có được cho phép cho các khối của các ảnh trong trình tự hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có thể thể hiện xem cờ bỏ qua biến đổi có thể có mặt hay không. Ví dụ, cờ được cho phép bỏ qua biến đổi có giá trị là 1 có thể thể hiện rằng việc bỏ qua biến đổi được cho phép, và cờ được cho phép bỏ qua biến đổi có giá trị là 0 có thể thể hiện rằng việc bỏ qua biến đổi không được cho phép. Nghĩa là, ví dụ, cờ được cho phép bỏ qua biến đổi có giá trị là 1 có thể thể hiện rằng cờ bỏ qua biến đổi có thể có mặt, và cờ được cho phép bỏ qua biến đổi có giá trị là 0 có thể thể hiện rằng cờ bỏ qua biến đổi không có mặt. Ngoài ra, ví dụ, cờ được cho phép bỏ qua biến đổi có thể được báo hiệu tới cú pháp tập thông số trình tự (SPS). Phần tử cú pháp của cờ được cho phép bỏ qua biến đổi có thể là sps_transform_skip_enabled_flag được mô tả ở trên.

Ngoài ra, ví dụ, cờ được cho phép TSRC có thể được thu nhận dựa trên cờ được

cho phép ẩn dữ liệu dấu, cờ được cho phép lượng tử hóa phụ thuộc, và/hoặc cờ được cho phép bỏ qua biến đổi. Ví dụ, cờ được cho phép TSRC có thể được thu nhận dựa trên cờ được cho phép ẩn dữ liệu dấu có giá trị là 0, cờ được cho phép lượng tử hóa phụ thuộc có giá trị là 0, và cờ được cho phép bỏ qua biến đổi có giá trị là 1. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 0 (nghĩa là, cờ được cho phép ẩn dữ liệu dấu thể hiện rằng việc ẩn dữ liệu dấu không được cho phép), giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 0 (nghĩa là, cờ được cho phép lượng tử hóa phụ thuộc thể hiện rằng việc lượng tử hóa phụ thuộc không được cho phép), và giá trị của cờ được cho phép bỏ qua biến đổi là 1 (nghĩa là, cờ được cho phép bỏ qua biến đổi thể hiện rằng việc bỏ qua biến đổi được cho phép), cờ được cho phép TSRC có thể được thu nhận (hoặc được báo hiệu). Ngoài ra, ví dụ, khi giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 1, cờ được cho phép TSRC có thể không được thu nhận, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép lượng tử hóa phụ thuộc là 1, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0. Ngoài ra, ví dụ, khi giá trị của cờ được cho phép bỏ qua biến đổi là 0, cờ được cho phép TSRC có thể không được thu nhận, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0. Nghĩa là, ví dụ, khi giá trị của cờ được cho phép bỏ qua biến đổi là 0, cờ được cho phép TSRC có thể không được báo hiệu, và giá trị của cờ được cho phép TSRC có thể được dẫn xuất là 0.

Thiết bị giải mã thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC (S1020). Thiết bị giải mã có thể thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC.

Ví dụ, thiết bị giải mã có thể xác định cú pháp lập mã phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC. Ví dụ, thiết bị giải mã có thể xác định cú pháp lập mã phần dư cho khối hiện tại như một trong số cú pháp lập mã phần dư thông thường (RRC) và cú pháp lập mã phần dư bỏ qua biến đổi (TSRC) dựa trên cờ được cho phép TSRC. Cú pháp RRC có thể thể hiện cú pháp theo RRC, và cú pháp TSRC có thể thể hiện cú pháp theo TSRC.

Ví dụ, dựa trên cờ được cho phép TSRC có giá trị là 1, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư thông thường (RRC).

Trong trường hợp này, ví dụ, cờ bỏ qua biến đổi về việc liệu việc bỏ qua biến đổi có được áp dụng cho khối hiện tại hay không có thể được thu nhận dựa trên cờ được cho phép bỏ qua biến đổi có giá trị là 1, và giá trị của cờ bỏ qua biến đổi có thể là 1. Ví dụ, thông tin hình ảnh có thể gồm cờ bỏ qua biến đổi cho khối hiện tại. Cờ bỏ qua biến đổi có thể thể hiện xem khối hiện tại có phải là khối bỏ qua biến đổi hay không. Nghĩa là, cờ bỏ qua biến đổi có thể thể hiện xem việc biến đổi đã được áp dụng cho các hệ số biến đổi của khối hiện tại hay chưa. Phần tử cú pháp biểu diễn cờ bỏ qua biến đổi có thể là transform_skip_flag như được mô tả ở trên. Ví dụ, nếu giá trị của cờ bỏ qua biến đổi là 1, cờ bỏ qua biến đổi có thể thể hiện rằng việc biến đổi đã không được áp dụng cho khối hiện tại (nghĩa là, việc biến đổi được bỏ qua), trong khi nếu giá trị của cờ bỏ qua biến đổi là 0, cờ bỏ qua biến đổi có thể thể hiện rằng việc biến đổi đã được áp dụng cho khối hiện tại. Ví dụ, nếu khối hiện tại là khối bỏ qua biến đổi, giá trị của cờ bỏ qua biến đổi cho khối hiện tại có thể là 1.

Ngoài ra, ví dụ, dựa trên cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư bỏ qua biến đổi (TSRC). Ngoài ra, ví dụ, cờ bỏ qua biến đổi về việc liệu việc bỏ qua biến đổi có được áp dụng cho khối hiện tại hay không có thể được thu nhận, và dựa trên cờ bỏ qua biến đổi có giá trị là 1 và cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư bỏ qua biến đổi (TSRC). Ngoài ra, ví dụ, cờ bỏ qua biến đổi về việc liệu việc bỏ qua biến đổi có được áp dụng cho khối hiện tại hay không có thể được thu nhận, và dựa trên cờ bỏ qua biến đổi có giá trị là 0 và cờ được cho phép TSRC có giá trị là 0, cú pháp lập mã phần dư cho khối hiện tại có thể được xác định như cú pháp lập mã phần dư thông thường (RRC).

Sau đó, ví dụ, thiết bị giải mã có thể thu nhận thông tin phần dư của cú pháp lập mã phần dư được xác định cho khối hiện tại. Ví dụ, thông tin phần dư của cú pháp lập mã phần dư thông thường (RRC) có thể được thu nhận dựa trên cờ được cho phép TSRC có giá trị là 1, và thông tin phần dư của cú pháp TSRC có thể được thu nhận dựa trên cờ được cho phép TSRC có giá trị là 0. Thông tin hình ảnh có thể gồm thông tin phần dư.

Ví dụ, nếu cú pháp lập mã phần dư cho khối hiện tại được xác định như cú pháp RRC, thiết bị giải mã có thể thu nhận thông tin phần dư của cú pháp RRC cho khối hiện tại. Ví dụ, thông tin phần dư của cú pháp RRC có thể gồm các phần tử cú pháp được bộc

lộ trong Bảng 2 như được mô tả ở trên.

Ví dụ, thông tin phần dư của cú pháp RRC có thể gồm các phần tử cú pháp cho hệ số biến đổi của khối hiện tại. Ở đây, hệ số biến đổi có thể được biểu diễn là hệ số phần dư.

Ví dụ, các phần tử cú pháp có thể gồm các phần tử cú pháp, chẳng hạn như last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, last_sig_coeff_y_suffix, sb_coded_flag, sig_coeff_flag, par_level_flag, abs_level_gtX_flag (ví dụ, abs_level_gtx_flag[n][0] và/hoặc abs_level_gtx_flag[n][1]), abs_remainder, dec_abs_level, và/hoặc coeff_sign_flag.

Cụ thể là, ví dụ, các phần tử cú pháp có thể gồm thông tin vị trí thể hiện vị trí của hệ số biến đổi khác không cuối cùng trong mảng hệ số phần dư của khối hiện tại. Nghĩa là, các phần tử cú pháp có thể gồm thông tin vị trí thể hiện vị trí của hệ số biến đổi khác không cuối cùng theo thứ tự quét của khối hiện tại. Thông tin vị trí có thể gồm thông tin thể hiện tiền tố trong vị trí cột của hệ số biến đổi khác không cuối cùng, thông tin thể hiện tiền tố trong vị trí hàng của hệ số biến đổi khác không cuối cùng, thông tin thể hiện hậu tố của vị trí cột của hệ số biến đổi khác không cuối cùng, và thông tin thể hiện hậu tố trong vị trí hàng của hệ số biến đổi khác không cuối cùng. Các phần tử cú pháp cho thông tin vị trí có thể là last_sig_coeff_x_prefix, last_sig_coeff_y_prefix, last_sig_coeff_x_suffix, và last_sig_coeff_y_suffix. Cùng lúc đó, hệ số biến đổi khác không có thể được gọi là hệ số đáng kể.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm cờ khối con được lập mã thể hiện xem khối con hiện tại của khối hiện tại gồm hệ số biến đổi khác không, cờ hệ số đáng kể thể hiện xem hệ số biến đổi của khối hiện tại là hệ số biến đổi khác không, cờ mức hệ số thứ nhất cho việc liệu mức hệ số cho hệ số biến đổi có lớn hơn giá trị ngưỡng thứ nhất hay không, cờ mức tính chẵn lẻ cho tính chẵn lẻ của mức hệ số, và/hoặc cờ mức hệ số thứ hai về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ hai hay không. Ở đây, cờ khối con được lập mã có thể là sb_coded_flag hoặc coded_sub_block_flag, cờ hệ số đáng kể có thể là sig_coeff_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag hoặc abs_level_gtx_flag, cờ mức tính chẵn lẻ có thể là par_level_flag, và cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm thông tin liên quan đến giá trị hệ số cho giá trị hệ số biến đổi của khối hiện tại. Thông tin liên quan đến giá trị hệ số có thể là abs_remainder và/hoặc dec_abs_level.

Ngoài ra, ví dụ, các phần tử cú pháp có thể gồm cờ dấu thể hiện dấu của hệ số biến đổi. Cờ dấu có thể là coeff_sign_flag.

Trong khi đó, ví dụ, khi việc ẩn dữ liệu dấu được áp dụng cho khối hiện tại, cờ dấu của hệ số biến đổi đáng kể thứ nhất của nhóm hệ số (CG) hiện tại trong khối hiện tại có thể không được báo hiệu. Nghĩa là, ví dụ, khi việc ẩn dữ liệu dấu được áp dụng cho khối hiện tại, các phần tử cú pháp có thể không gồm cờ dấu biểu diễn dấu của hệ số biến đổi đáng kể thứ nhất. Trong khi đó, ví dụ, việc liệu việc ẩn dữ liệu dấu có được áp dụng cho khối hiện tại hay không có thể được dẫn xuất dựa trên cờ được cho phép ẩn dữ liệu dấu và/hoặc vị trí của hệ số biến đổi đáng kể thứ nhất và vị trí của hệ số biến đổi đáng kể cuối cùng của CG hiện tại. Ví dụ, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, và giá trị thu được bằng cách trừ vị trí của hệ số biến đổi đáng kể thứ nhất từ vị trí của hệ số biến đổi đáng kể cuối cùng là lớn hơn 3 (nghĩa là, khi giá trị của cờ được cho phép ẩn dữ liệu dấu là 1, và số lượng của các hệ số biến đổi đáng kể trong CG hiện tại là lớn hơn 3), việc ẩn dữ liệu dấu có thể được áp dụng cho CG hiện tại của khối hiện tại.

Ngoài ra, ví dụ, nếu cú pháp lập mã phần dư cho khối hiện tại được xác định như cú pháp TSRC, thiết bị giải mã có thể thu nhận thông tin phần dư của cú pháp TSRC cho khối hiện tại. Ví dụ, thông tin phần dư của cú pháp TSRC có thể gồm các phần tử cú pháp được bộc lộ trong Bảng 3 như được mô tả ở trên.

Ví dụ, thông tin phần dư của cú pháp TSRC có thể gồm các phần tử cú pháp cho hệ số biến đổi của khối hiện tại. Ở đây, hệ số biến đổi có thể được biểu diễn là hệ số phần dư.

Ví dụ, các phần tử cú pháp có thể gồm các phần tử cú pháp được lập mã ngữ cảnh cho hệ số biến đổi và/hoặc các phần tử cú pháp được lập mã đi vòng qua. Các phần tử cú pháp có thể gồm các phần tử cú pháp, chẳng hạn như sig_coeff_flag, coeff_sign_flag, par_level_flag, abs_level_gtX_flag (ví dụ, abs_level_gtx_flag[n][0], abs_level_gtx_flag[n][1], abs_level_gtx_flag[n][2], abs_level_gtx_flag[n][3], và/hoặc

abs_level_gtx_flag[n][4]), abs_remainder, và/hoặc coeff_sign_flag.

Ví dụ, các phần tử cú pháp được lập mã ngữ cảnh cho hệ số biến đổi có thể gồm cờ hệ số đáng kể thể hiện xem hệ số biến đổi là hệ số biến đổi khác không, cờ dấu thể hiện dấu cho hệ số biến đổi, cờ mức hệ số thứ nhất cho việc liệu mức hệ số cho hệ số biến đổi có lớn hơn giá trị ngưỡng thứ nhất hay không, và/hoặc cờ mức tính chẵn lẻ cho tính chẵn lẻ của mức biến đổi cho hệ số biến đổi. Ngoài ra, ví dụ, các phần tử cú pháp được lập mã ngữ cảnh có thể gồm cờ mức hệ số thứ hai cho việc liệu mức hệ số của hệ số biến đổi. Ngoài ra, ví dụ, các phần tử cú pháp được lập mã ngữ cảnh có thể gồm cờ mức hệ số thứ hai về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ hai hay không, cờ mức hệ số thứ ba về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ ba hay không, cờ mức hệ số thứ tư về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ tư hay không, và/hoặc cờ mức hệ số thứ năm về việc liệu mức hệ số của hệ số biến đổi có lớn hơn giá trị ngưỡng thứ năm hay không. Ở đây, cờ hệ số đáng kể có thể là sig_coeff_flag, cờ dấu có thể là ceff_sign_flag, cờ mức hệ số thứ nhất có thể là abs_level_gt1_flag, và cờ mức tính chẵn lẻ có thể là par_level_flag. Ngoài ra, cờ mức hệ số thứ hai có thể là abs_level_gt3_flag hoặc abs_level_gtx_flag, cờ mức hệ số thứ ba có thể là abs_level_gt5_flag hoặc abs_level_gtx_flag, cờ mức hệ số thứ tư có thể là abs_level_gt7_flag hoặc abs_level_gtx_flag, và cờ mức hệ số thứ năm có thể là abs_level_gt9_flag hoặc abs_level_gtx_flag.

Ngoài ra, ví dụ, các phần tử cú pháp được lập mã đi vòng qua cho hệ số biến đổi có thể gồm thông tin mức hệ số cho giá trị của hệ số biến đổi (hoặc mức hệ số) và/hoặc cờ dấu thể hiện dấu cho hệ số biến đổi. Thông tin mức hệ số có thể là abs_remainder và/hoặc dec_abs_level, và cờ dấu có thể là ceff_sign_flag.

Thiết bị giải mã dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư (S1030). Ví dụ, thiết bị giải mã có thể dẫn xuất các hệ số biến đổi của khối hiện tại dựa trên thông tin phần dư, và có thể dẫn xuất các mẫu phần dư của khối hiện tại dựa trên các hệ số biến đổi.

Ví dụ, thiết bị giải mã có thể dẫn xuất các hệ số biến đổi của khối hiện tại dựa trên các phần tử cú pháp của thông tin phần dư. Sau đó, thiết bị giải mã có thể dẫn xuất các mẫu phần dư của khối hiện tại dựa trên các hệ số biến đổi. Như một ví dụ, nếu được

dẫn xuất rằng việc biến đổi không được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, nghĩa là, nếu giá trị của cờ bỏ qua biến đổi là 1, thiết bị giải mã có thể dẫn xuất các hệ số biến đổi như các mẫu phần dư của khối hiện tại. Ngoài ra, ví dụ, nếu được dẫn xuất rằng việc biến đổi không được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, nghĩa là, nếu giá trị của cờ bỏ qua biến đổi là 1, thiết bị giải mã có thể dẫn xuất các mẫu phần dư của khối hiện tại bằng việc khử lượng tử hóa các hệ số biến đổi. Ngoài ra, ví dụ, nếu được dẫn xuất rằng việc biến đổi được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, nghĩa là, nếu giá trị của cờ bỏ qua biến đổi là 0, thiết bị giải mã có thể dẫn xuất các mẫu phần dư của khối hiện tại bằng việc thực hiện việc biến đổi ngược của các hệ số biến đổi. Ngoài ra, ví dụ, nếu được dẫn xuất rằng việc biến đổi được áp dụng cho khối hiện tại dựa trên cờ bỏ qua biến đổi, nghĩa là, nếu giá trị của cờ bỏ qua biến đổi là 0, thiết bị giải mã có thể dẫn xuất các mẫu phần dư của khối hiện tại bằng việc khử lượng tử hóa các hệ số biến đổi và thực hiện việc biến đổi ngược của các hệ số biến đổi được khử lượng tử hóa.

Cùng lúc đó, trong trường hợp mà việc lượng tử hóa phụ thuộc được áp dụng cho khối hiện tại, thiết bị giải mã có thể dẫn xuất các mẫu phần dư của khối hiện tại bằng việc thực hiện quy trình lượng tử hóa phụ thuộc cho các hệ số biến đổi. Ví dụ, trong trường hợp mà việc lượng tử hóa phụ thuộc được áp dụng cho khối hiện tại, thiết bị giải mã có thể cập nhật trạng thái (Qstate) cho việc lượng tử hóa phụ thuộc dựa trên mức hệ số của hệ số biến đổi ngay trước hệ số biến đổi hiện tại theo thứ tự quét, có thể dẫn xuất mức hệ số của hệ số biến đổi hiện tại dựa trên trạng thái được cập nhật và các phần tử cú pháp cho hệ số biến đổi hiện tại, và có thể dẫn xuất mẫu phần dư bằng việc khử lượng tử hóa mức hệ số được dẫn xuất. Ví dụ, hệ số biến đổi hiện tại có thể được khử lượng tử hóa dựa trên thông số lượng tử hóa cho mức được xây dựng lại của hệ số biến đổi hiện tại trong bộ lượng tử hóa vô hướng cho trạng thái được cập nhật. Ở đây, mức được xây dựng lại có thể được dẫn xuất dựa trên các phần tử cú pháp cho hệ số biến đổi hiện tại.

Trong khi đó, ví dụ, khi việc ẩn dữ liệu dấu được áp dụng cho khối hiện tại, dấu của hệ số biến đổi đáng kể thứ nhất của CG hiện tại trong khối hiện tại có thể được dẫn xuất dựa trên tổng của các giá trị tuyệt đối của các hệ số biến đổi đáng kể trong CG hiện tại. Ví dụ, khi tổng của các giá trị tuyệt đối của các hệ số biến đổi đáng kể là chẵn, dấu

của hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị dương, và khi tổng của các giá trị tuyệt đối của các hệ số biến đổi đáng kể là lẻ, dấu của hệ số biến đổi đáng kể thứ nhất có thể được dẫn xuất là giá trị âm.

Thiết bị giải mã tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư (S1040). Ví dụ, thiết bị giải mã có thể tạo ra mẫu được xây dựng lại và/hoặc ảnh được xây dựng lại của khối hiện tại dựa trên mẫu phần dư. Ví dụ, thiết bị giải mã có thể dẫn xuất mẫu dự đoán bằng việc thực hiện chế độ dự đoán liên hoặc chế độ dự đoán nội trên khối hiện tại dựa trên thông tin dự đoán được nhận thông qua luồng bit, và có thể tạo ra mẫu được xây dựng lại thông qua việc bổ sung của mẫu dự đoán và mẫu phần dư với nhau.

Sau đó, nếu cần, để nâng cao chất lượng ảnh khách quan/chủ quan, thủ tục lọc trong vòng lặp, chẳng hạn như thủ tục lọc giải khối, SAO và/hoặc ALF, có thể được áp dụng cho ảnh được xây dựng lại như được mô tả ở trên.

Fig.11 minh họa sơ lược thiết bị giải mã để thực hiện phương pháp giải mã hình ảnh theo sáng chế. Phương pháp được bộc lộ trong Fig.10 có thể được thực hiện bởi thiết bị giải mã được bộc lộ trong Fig.11. Cụ thể là, ví dụ, bộ giải mã entrôpi của thiết bị giải mã của Fig.11 có thể thực hiện S1000 đến S1020 của Fig.10, bộ xử lý phần dư của thiết bị giải mã của Fig.11 có thể thực hiện S1030 của Fig.10, và bộ cộng của thiết bị giải mã của Fig.11 có thể thực hiện S1040 của Fig.10. Ngoài ra, mặc dù không được minh họa, quy trình nhận thông tin dự đoán cho khối hiện tại có thể được thực hiện bởi bộ giải mã entrôpi của thiết bị giải mã của Fig.11, và quy trình dẫn xuất mẫu dự đoán của khối hiện tại có thể được thực hiện bởi bộ dự đoán của thiết bị giải mã của Fig.11.

Theo sáng chế, hiệu quả lập mã phần dư có thể được nâng cao.

Ngoài ra, theo sáng chế, cờ được cho phép TSRC có thể được báo hiệu khi việc ẩn dữ liệu dấu không được cho phép bằng việc thiết đặt mối quan hệ báo hiệu giữa cờ được cho phép ẩn dữ liệu dấu và cờ được cho phép TSRC, qua đó, khi cú pháp RRC được lập mã cho khối bỏ qua biến đổi bởi vì TSRC không được cho phép, việc ẩn dữ liệu dấu không được sử dụng để cải thiện hiệu quả lập mã, và hiệu quả lập mã phần dư tổng thể có thể được cải thiện thông qua việc giảm lượng các bit được lập mã.

Ngoài ra, theo sáng chế, mối quan hệ báo hiệu giữa cờ được cho phép lượng tử hóa phụ thuộc và cờ được cho phép TSRC có thể được thiết lập, và nếu việc lượng tử

hóa phụ thuộc là không được cho phép, cờ được cho phép TSRC có thể được báo hiệu, và qua đó, nếu TSRC là không được cho phép và sau đó cú pháp RRC được lập mã cho khối bỏ qua biến đổi, việc lượng tử hóa phụ thuộc là sẽ không được sử dụng, do đó hiệu quả lập mã có thể được cải thiện, và hiệu quả lập mã phần dư tổng có thể được cải thiện thông qua việc giảm của lượng các bit được lập mã.

Ngoài ra, theo sáng chế, mối quan hệ báo hiệu giữa cờ được cho phép bỏ qua biến đổi và cờ được cho phép TSRC có thể được thiết lập, và nếu việc bỏ qua biến đổi được cho phép, cờ được cho phép TSRC có thể được báo hiệu, và qua đó, hiệu quả lập mã phần dư tổng thể có thể được cải thiện thông qua việc giảm lượng các bit được lập mã.

Trong phương án được mô tả ở trên, các phương pháp được mô tả dựa trên lưu đồ có một loạt các bước hoặc các khối. Sáng chế không được giới hạn ở thứ tự của các bước hoặc các khối ở trên. Một số bước hoặc khối có thể xuất hiện đồng thời hoặc theo thứ tự khác với các bước hoặc các khối khác được mô tả ở trên. Hơn nữa, người có hiểu biết trung bình trong lĩnh vực kỹ thuật tương ứng sẽ hiểu rằng các bước được thể hiện trong lưu đồ ở trên không phải là độc quyền và các bước khác nữa có thể được gồm, hoặc một hoặc nhiều bước trong các lưu đồ làm ví dụ có thể được xóa mà không ảnh hưởng tới phạm vi bảo hộ của sáng chế này.

Các phương án được mô tả trong sáng chế có thể được thực hiện bằng việc được triển khai bởi bộ xử lý, bộ vi xử lý, bộ điều khiển hoặc chip. Ví dụ, các đơn vị chức năng được thể hiện trên mỗi hình vẽ có thể được được thực hiện bằng việc được triển khai trên máy tính, bộ xử lý, bộ vi xử lý, bộ điều khiển hoặc con chip. Trong trường hợp này, thông tin cho việc thực hiện (ví dụ, thông tin về các lệnh) hoặc thuật toán có thể được lưu trữ trong phương tiện lưu trữ kỹ thuật số.

Bên cạnh đó, thiết bị giải mã và thiết bị mã hóa mà các phương án của sáng chế được áp dụng vào đó có thể được gồm trong thiết bị truyền/nhận phát rộng đa phương tiện, thiết bị đầu cuối truyền thông di động, thiết bị viđeo chiếu phim gia đình, thiết bị viđeo chiếu phim số, camêra giám sát, thiết bị trò chuyện viđeo, thiết bị truyền thông thời gian thực như là truyền thông viđeo, thiết bị phát luồng (streaming) di động, phương tiện lưu trữ, máy ghi hình cầm tay, thiết bị cung cấp dịch vụ viđeo theo yêu cầu (Video on Demand, VoD), thiết bị cung cấp dịch vụ nội dung trên nền mạng viễn thông (Over

The Top, OTT), thiết bị cung cấp dịch vụ tạo luồng Internet, thiết bị viđeo ba chiều (Three Dimensional, 3D), thiết bị viđeo dùng cho các cuộc họp tổ chức qua điện thoại, thiết bị người dùng giao thông (tức là, thiết bị người dùng trên xe, thiết bị người dùng trên máy bay, thiết bị người dùng trên tàu) và thiết bị viđeo y tế, và có thể được dùng để xử lý tín hiệu viđeo hoặc tín hiệu dữ liệu. Ví dụ, thiết bị viđeo OTT có thể gồm máy chơi trò chơi, máy đọc phát Blu-ray, tivi truy cập Internet, hệ thống rạp hát gia đình, điện thoại thông minh, máy tính bảng, đầu ghi viđeo kỹ thuật số (Digital Video Recorder, DVR), và tương tự.

Hơn thế nữa, phương pháp xử lý mà sáng chế được áp dụng vào đó có thể được tạo ra trong dạng chương trình mà cần được thực thi bởi máy tính, và có thể được lưu trữ trong phương tiện ghi đọc được bởi máy tính. Dữ liệu đa phương tiện mà có cấu trúc dữ liệu theo sáng chế thì cũng có thể được lưu trữ trên phương tiện ghi đọc được bởi máy tính. Phương tiện ghi đọc được bởi máy tính gồm tất cả các loại thiết bị lữu trữ mà dữ liệu có thể đọc được bởi hệ thống máy tính được lưu trữ trong đó. Phương tiện ghi đọc được bằng máy tính, ví dụ, có thể bao gồm đĩa Blu-ray (Blu-ray disk, BD), bus nối tiếp vạn năng (Universal Serial Bus, USB), ROM, PROM, EPROM, EEPROM, RAM, CD-ROM, băng từ, đĩa mềm, và thiết bị lưu trữ dữ liệu quang, ví dụ. Hơn thế nữa, phương tiện ghi đọc được bởi máy tính gồm phương tiện được triển khai trong dạng các sóng mang (ví dụ, truyền qua Internet). Bên cạnh đó, luồng bit được tạo ra bởi phương pháp mã hóa này có thể được lưu trữ trong phương tiện ghi đọc được bởi máy tính hoặc được truyền qua các mạng truyền thông có dây/không dây.

Bên cạnh đó, các phương án theo sáng chế là có thể được triển khai dưới dạng sản phẩm chương trình máy tính bởi các mã chương trình, và các mã chương trình này có thể được thực hiện trên máy tính bởi các phương án theo sáng chế. Các mã chương trình có thể được lưu trữ trên vật mang đọc được bởi máy tính.

Fig.12 minh họa sơ đồ cấu trúc của hệ thống phát luồng các nội dung với nó sáng chế được áp dụng.

Hệ thống phát luồng nội dung mà với nó (các) phương án của tài liệu này được áp dụng có thể gồm máy chủ mã hóa, máy chủ phát luồng, máy chủ web, bộ phận lưu trữ phương tiện, thiết bị người dùng, và thiết bị nhập vào đa phương tiện.

Máy chủ mã hóa nén nội dung được nhập vào từ các thiết bị đầu vào đa phương tiện như điện thoại thông minh, máy ảnh, máy quay viđeo, v.v. thành dữ liệu kỹ thuật số để tạo ra luồng bit và truyền luồng bit đến máy chủ phát luồng. Theo một ví dụ khác, khi thiết bị đầu vào đa phương tiện như điện thoại thông minh, máy ảnh, máy quay viđeo, v.v. trực tiếp tạo ra luồng bit, thì máy chủ mã hóa có thể được lược bỏ.

Luồng bit có thể được tạo ra bởi phương pháp mã hóa hoặc phương pháp tạo ra luồng bit mà phương án (các phương án) của sáng chế được áp dụng vào đó, và máy chủ phát luồng có thể tạm thời lưu trữ luồng bit trong quy trình truyền hoặc nhận luồng bit.

Máy chủ phát luồng truyền dữ liệu đa phương tiện đến thiết bị người dùng dựa trên yêu cầu của người dùng thông qua máy chủ web, và máy chủ web đóng vai trò là môi trường để thông báo cho người dùng về dịch vụ. Khi người dùng yêu cầu dịch vụ mong muốn từ máy chủ web, thì máy chủ web vận chuyển nó đến máy chủ phát luồng, và máy chủ phát luồng truyền dữ liệu đa phương tiện đến người dùng. Trong trường hợp này, hệ thống phát luồng nội dung có thể gồm máy chủ điều khiển riêng biệt. Trong trường hợp này, máy chủ điều khiển có tác dụng để điều khiển lệnh/phản hồi giữa các thiết bị trong hệ thống phát luồng nội dung.

Máy chủ phát luồng có thể nhận nội dung từ bộ phận lưu trữ phương tiện và/hoặc máy chủ mã hóa. Ví dụ, khi nội dung được nhận từ máy chủ mã hóa, thì nội dung có thể được nhận theo thời gian thực. Trong trường hợp này, để cung cấp dịch vụ tạo luồng trơn tru, thì máy chủ phát luồng có thể lưu trữ luồng bit trong thời gian được xác định trước.

Các ví dụ về thiết bị người dùng có thể gồm điện thoại di động, điện thoại thông minh, máy tính xách tay, thiết bị đầu cuối phát rộng kỹ thuật số, trợ lý kỹ thuật số cá nhân (Personal Digital Assistant, PDA), máy phát đa phương tiện cầm tay (Portable Multimedia Player, PMP), bộ điều hướng, PC dạng tấm, các PA dạng bảng, máy tính ultrabook, các thiết bị đeo được (ví dụ, đồng hồ thông minh, kính thông minh, bộ hiển thị gắn trên đầu), tivi kỹ thuật số, máy tính để bàn, biển báo kỹ thuật số, và tương tự. Mỗi máy chủ trong hệ thống phát luồng nội dung có thể được làm hoạt động như là máy chủ phân tán, mà trong trường hợp đó thì dữ liệu được nhận từ mỗi máy chủ có thể được phân tán.

Các điểm yêu cầu bảo hộ được mô tả trong sáng chế có thể được tổ hợp theo các cách khác nhau. Ví dụ, các dấu hiệu kỹ thuật của các điểm yêu cầu bảo hộ dạng phương pháp của sáng chế có thể được tổ hợp để được triển khai dưới dạng thiết bị, và các dấu hiệu kỹ thuật của các điểm yêu cầu bảo hộ dạng thiết bị của sáng chế có thể được tổ hợp để được triển khai dưới dạng phương pháp. Bên cạnh đó, các dấu hiệu kỹ thuật của các điểm yêu cầu bảo hộ dạng phương pháp của sáng chế và các dấu hiệu kỹ thuật của điểm yêu cầu bảo hộ dạng thiết bị có thể được tổ hợp để được triển khai dưới dạng thiết bị, và các dấu hiệu kỹ thuật của các điểm yêu cầu bảo hộ dạng phương pháp của sáng chế và các dấu hiệu kỹ thuật của điểm yêu cầu bảo hộ dạng thiết bị có thể được tổ hợp để được triển khai dưới dạng phương pháp.

YÊU CẦU BẢO HỘ

1. Phương pháp giải mã hình ảnh được thực hiện bởi thiết bị giải mã, phương pháp này bao gồm các bước:

thu nhận cờ được cho phép lượng tử hóa phụ thuộc;

thu nhận cờ bất hoạt lập mã phần dư bỏ qua biến đổi (Transform Skip Residual Coding, TSRC) dựa trên cờ được cho phép lượng tử hóa phụ thuộc;

thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ bất hoạt TSRC;

dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư; và

tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư,

trong đó cờ được cho phép lượng tử hóa phụ thuộc là cờ về việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không,

trong đó cờ bất hoạt TSRC là cờ về việc liệu TSRC có được sử dụng hay không,

trong đó cờ bất hoạt TSRC được thu nhận từ luồng bit dựa trên giá trị của cờ được cho phép lượng tử hóa phụ thuộc là bằng 0,

trong đó cờ bất hoạt TSRC có giá trị là 1 thể hiện rằng TSRC không được sử dụng, và

trong đó cờ bất hoạt TSRC có giá trị là 0 thể hiện rằng TSRC được sử dụng.

2. Phương pháp mã hóa hình ảnh được thực hiện bởi thiết bị mã hóa, phương pháp này bao gồm các bước:

mã hóa cờ được cho phép lượng tử hóa phụ thuộc về việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không;

mã hóa, dựa trên cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được sử dụng hay không;

mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ bất hoạt TSRC; và

tạo ra luồng bit gồm cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt TSRC và thông tin phần dư,

trong đó cờ bất hoạt TSRC được mã hóa vào luồng bit dựa trên giá trị của cờ được cho phép lượng tử hóa phụ thuộc là bằng 0,

trong đó cờ bất hoạt TSRC có giá trị là 1 thể hiện rằng TSRC không được sử dụng, và

trong đó cờ bất hoạt TSRC có giá trị là 0 thể hiện rằng TSRC được sử dụng.

3. Phương tiện lưu trữ đọc được bởi máy tính không chuyển tiếp để lưu trữ luồng bit được tạo ra bởi phương pháp mã hóa hình ảnh, phương pháp mã hóa hình ảnh bao gồm các bước:

mã hóa cờ được cho phép lượng tử hóa phụ thuộc về việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không;

mã hóa, dựa trên cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được sử dụng hay không;

mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ bất hoạt TSRC; và

tạo ra luồng bit gồm cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt TSRC và thông tin phần dư,

trong đó cờ bất hoạt TSRC được mã hóa vào luồng bit dựa trên giá trị của cờ được cho phép lượng tử hóa phụ thuộc là bằng 0,

trong đó cờ bất hoạt TSRC có giá trị là 1 thể hiện rằng TSRC không được sử dụng, và

trong đó cờ bất hoạt TSRC có giá trị là 0 thể hiện rằng TSRC được sử dụng.

4. Phương pháp truyền dữ liệu cho thông tin hình ảnh bao gồm các bước:

mã hóa cờ được cho phép lượng tử hóa phụ thuộc về việc liệu việc lượng tử hóa phụ thuộc có được cho phép hay không;

mã hóa, dựa trên cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt lập mã phần dư bỏ qua biến đổi (TSRC) về việc liệu TSRC có được sử dụng hay không;

mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ bất hoạt TSRC;

tạo ra luồng bit gồm cờ được cho phép lượng tử hóa phụ thuộc, cờ bất hoạt TSRC và thông tin phần dư; và

trong đó cờ bất hoạt TSRC được mã hóa vào luồng bit dựa trên giá trị của cờ được cho phép lượng tử hóa phụ thuộc là bằng 0,

trong đó cờ bất hoạt TSRC có giá trị là 1 thể hiện rằng TSRC không được sử dụng, và

trong đó cờ bất hoạt TSRC có giá trị là 0 thể hiện rằng TSRC được sử dụng.

FIG. 1

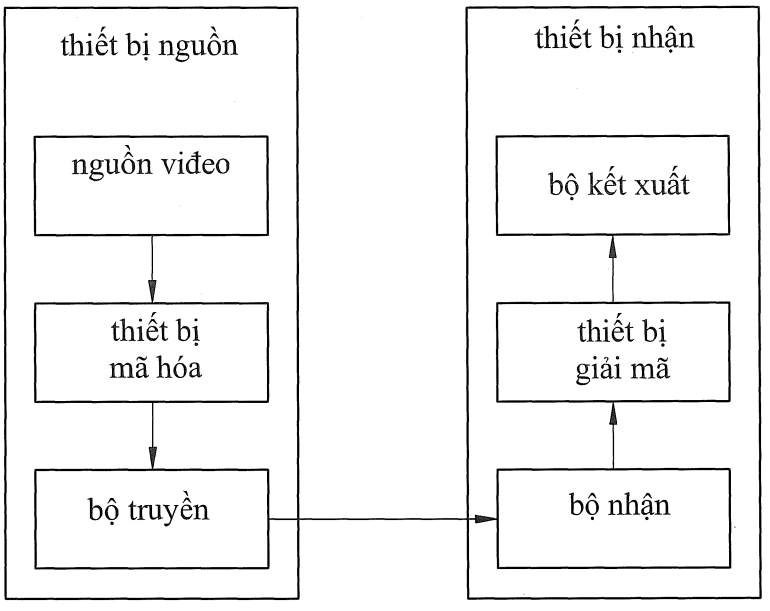

thiết bị nguồn
nguồn viđeo
thiết bị
mã hóa
bộ truyền
thiết bị nhận
bộ kết xuất
thiết bị
giải mã
bộ nhận

FIG. 2

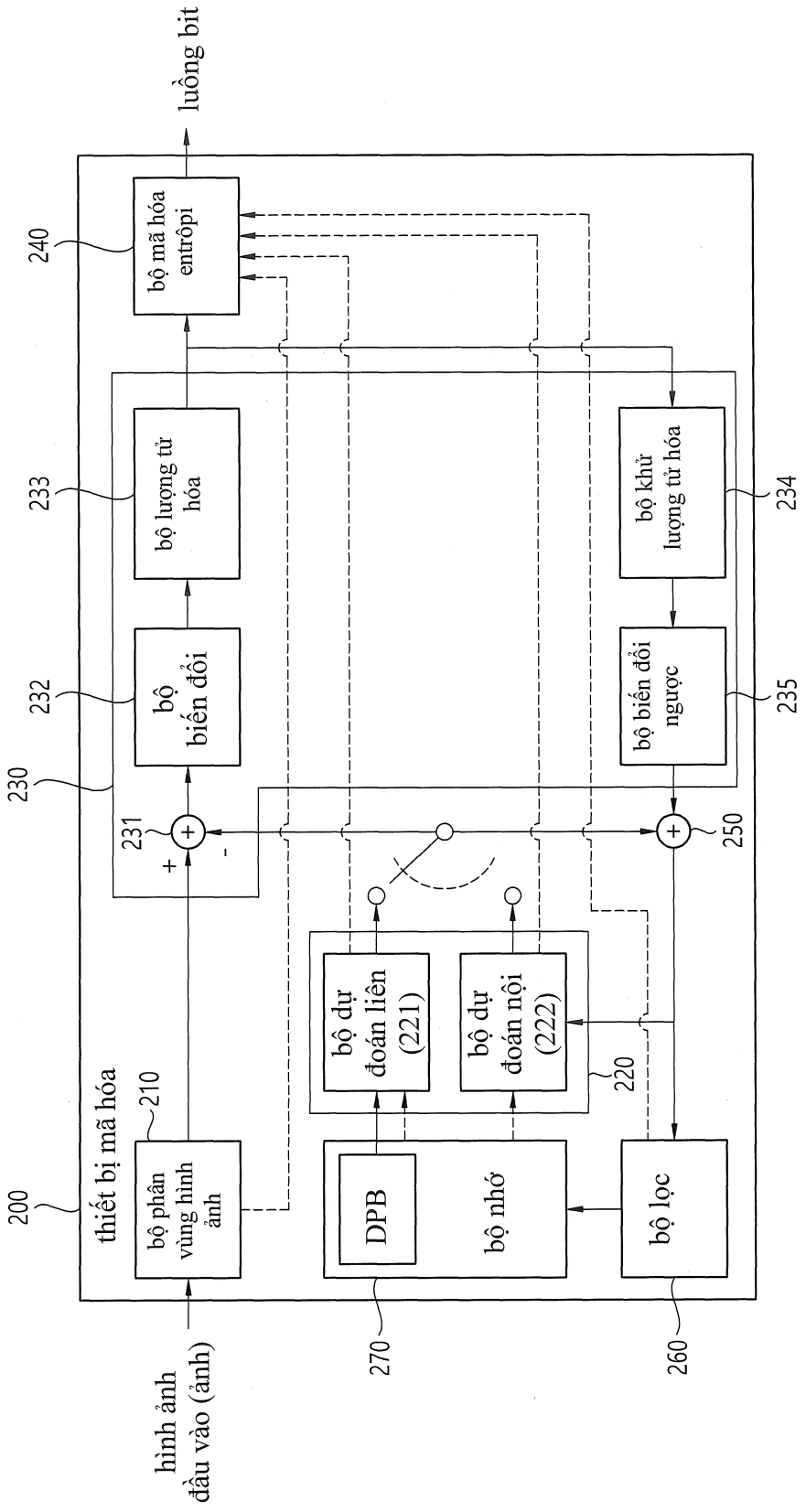

luồng bit
240
bộ mã hóa entrôpi
233
bộ lượng tử hóa
234
bộ khử lượng tử hóa
232
bộ biến đổi
235
bộ biến đổi ngược
230
231
+
-
250
bộ dự đoán liên (221)
bộ dự đoán nội (222)
220
200
thiết bị mã hóa
210
bộ phân vùng hình ảnh
DPB
bộ nhớ
bộ lọc
270
260
hình ảnh đầu vào (ảnh)

FIG. 3

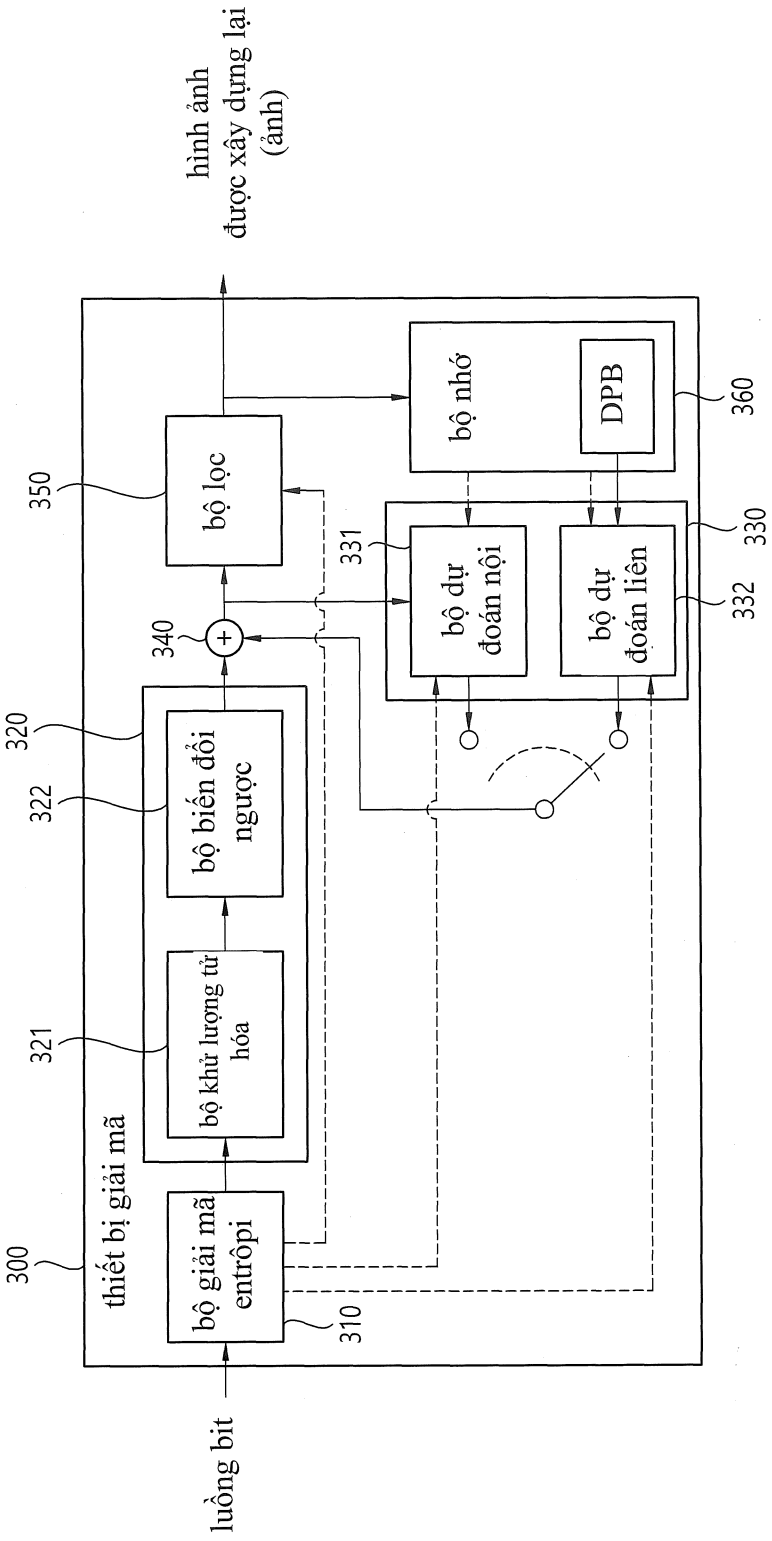

luồng bit
thiết bị giải mã
300
bộ giải mã entrôpi
310
bộ khử lượng tử hóa
321
bộ biến đổi ngược
322
320
340
bộ lọc
350
hình ảnh được xây dựng lại (ảnh)
bộ nhớ
DPB
360
bộ dự đoán nội
331
bộ dự đoán liên
332
330

FIG. 4

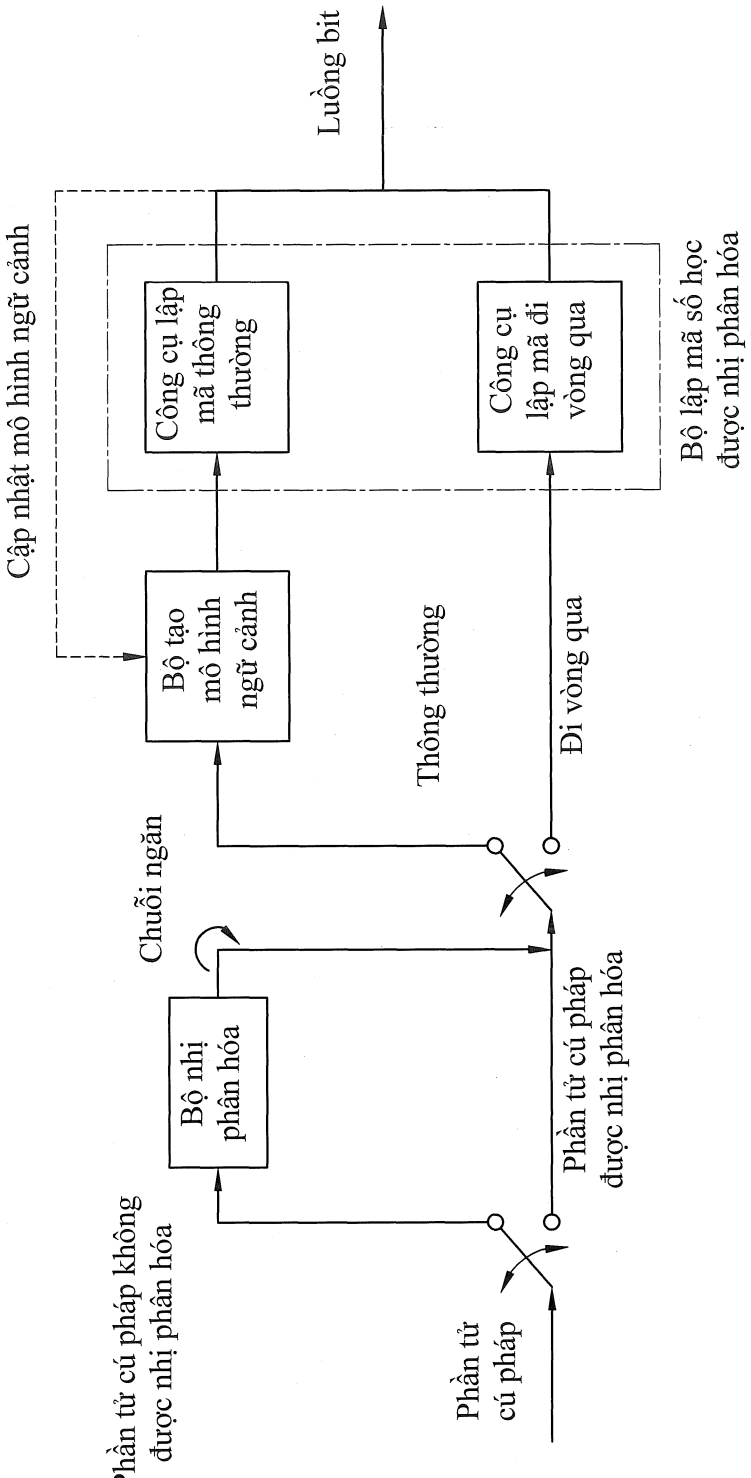

Luồng bit
Cập nhật mô hình ngữ cảnh
Công cụ lập mã thông thường
Công cụ lập mã đi vòng qua
Bộ lập mã số học được nhị phân hóa
Bộ tạo mô hình ngữ cảnh
Thông thường
Đi vòng qua
Chuỗi ngăn
Bộ nhị phân hóa
Phần tử cú pháp được nhị phân hóa
Phần tử cú pháp không được nhị phân hóa
Phần tử cú pháp

FIG. 5

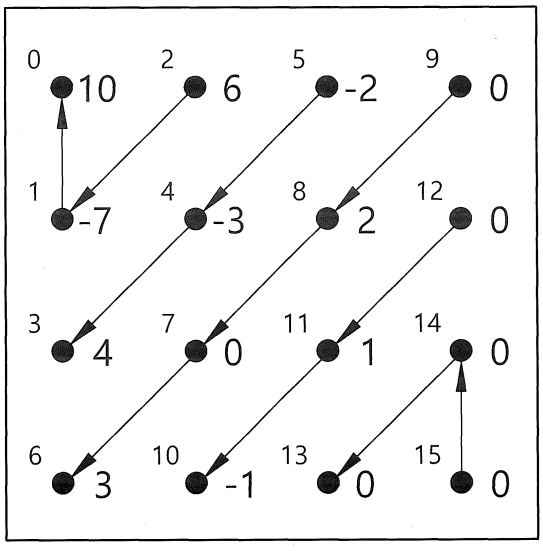


FIG. 6

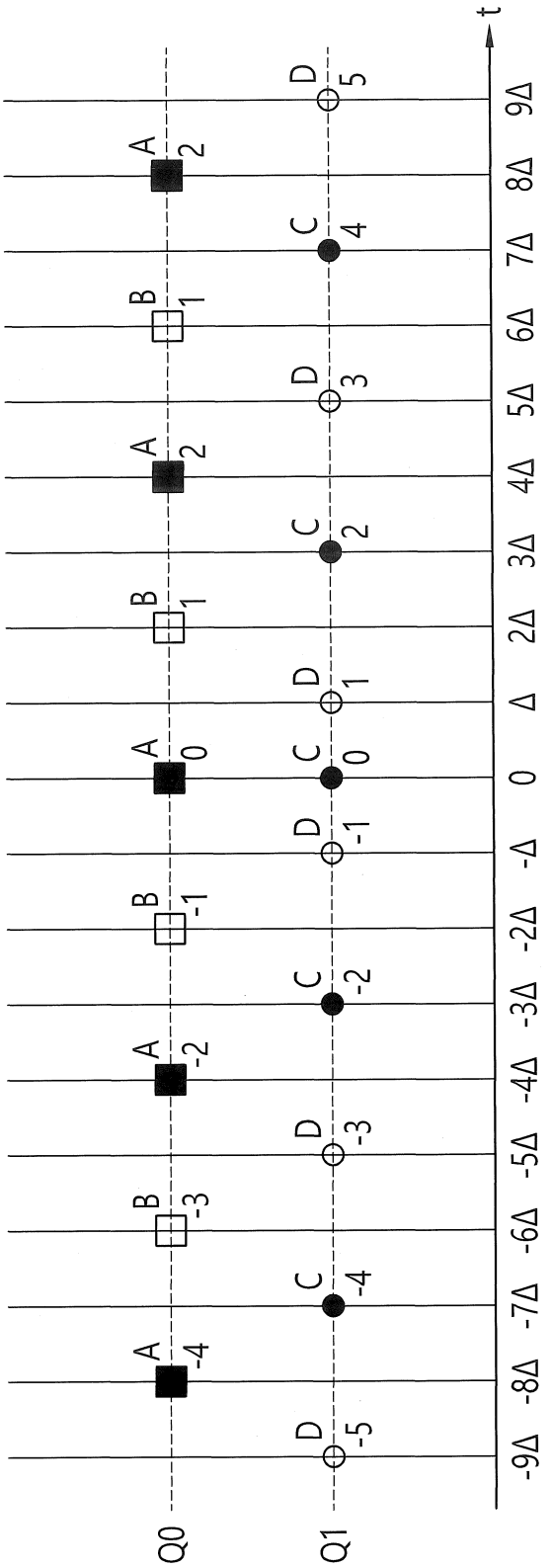


FIG. 7

Trạng thái hiện tại	trạng thái tiếp theo cho ...	
	(k & 1)==0	(k & 1)==1
0	0	2
1	2	0
2	1	3
3	3	1

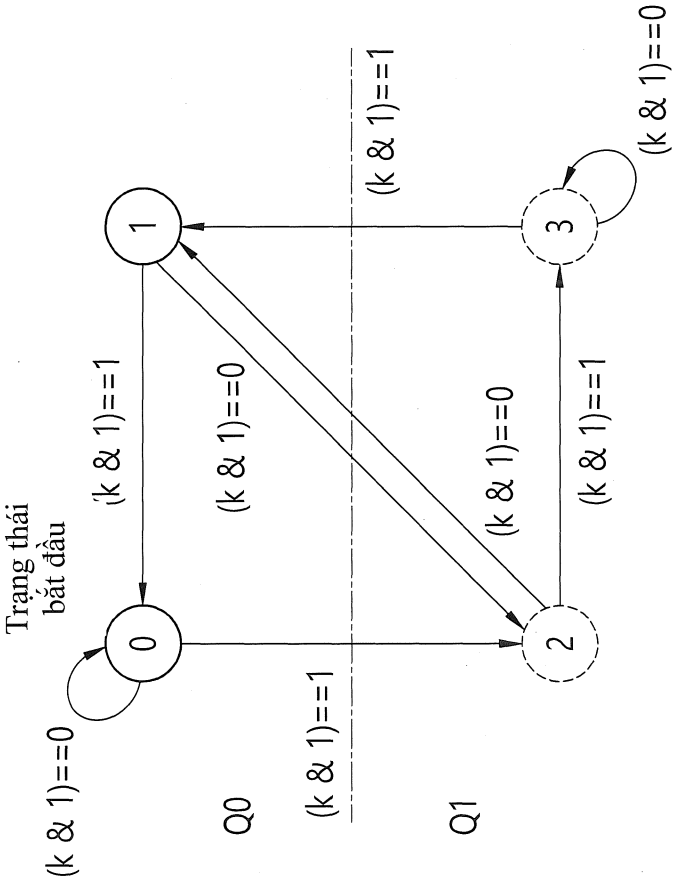


FIG. 8

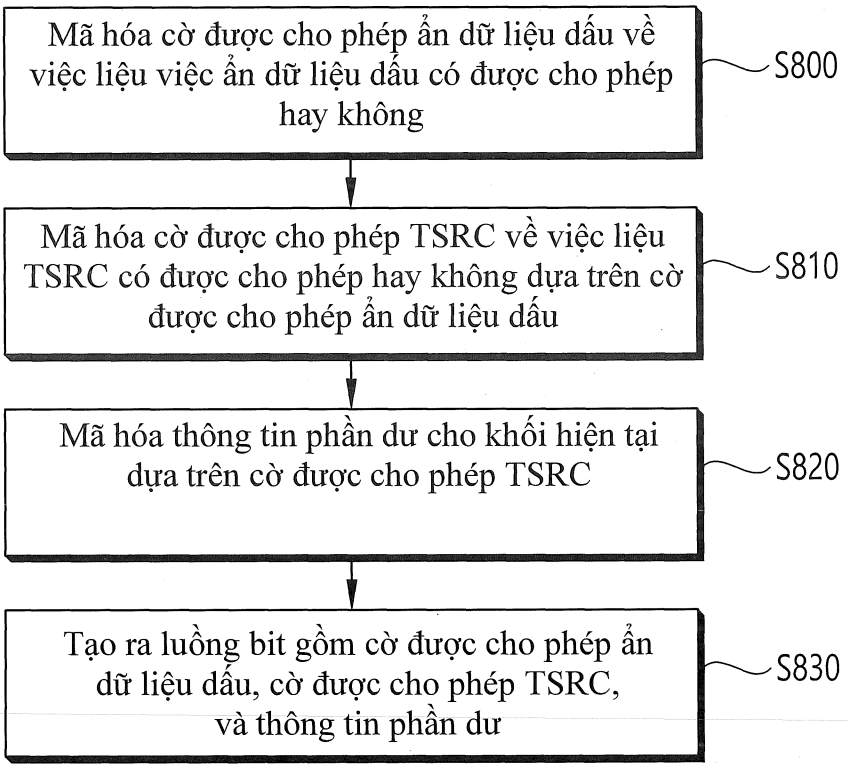

Mã hóa cờ được cho phép ẩn dữ liệu dấu về việc liệu việc ẩn dữ liệu dấu có được cho phép hay không
S800
Mã hóa cờ được cho phép TSRC về việc liệu TSRC có được cho phép hay không dựa trên cờ được cho phép ẩn dữ liệu dấu
S810
Mã hóa thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC
S820
Tạo ra luồng bit gồm cờ được cho phép ẩn dữ liệu dấu, cờ được cho phép TSRC, và thông tin phần dư
S830

FIG. 9

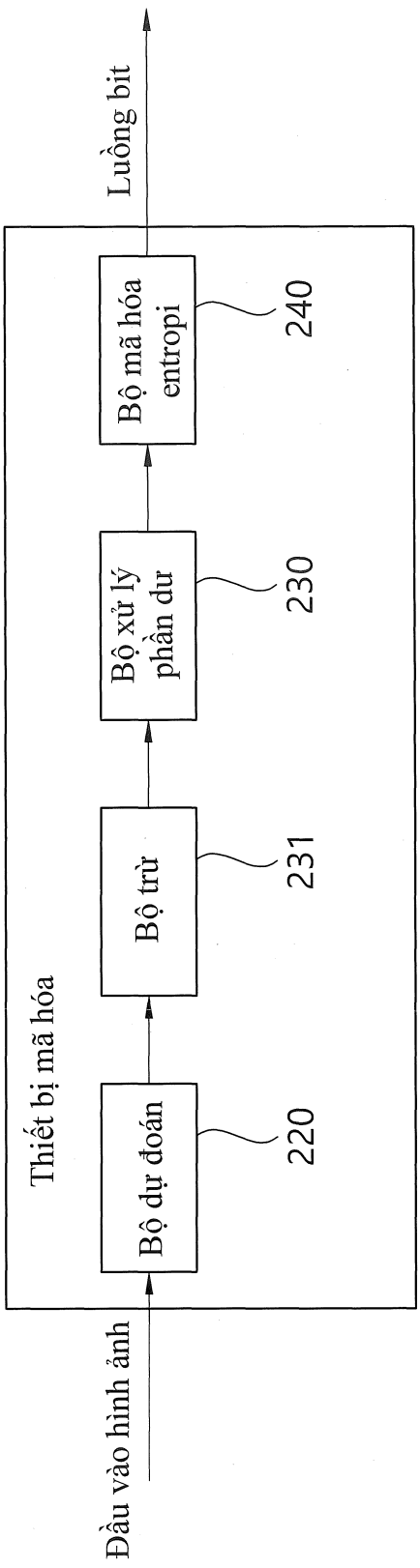

Đầu vào hình ảnh
Thiết bị mã hóa
Bộ dự đoán
220
Bộ trừ
231
Bộ xử lý phần dư
230
Bộ mã hóa entropi
240
Luồng bit

FIG. 10

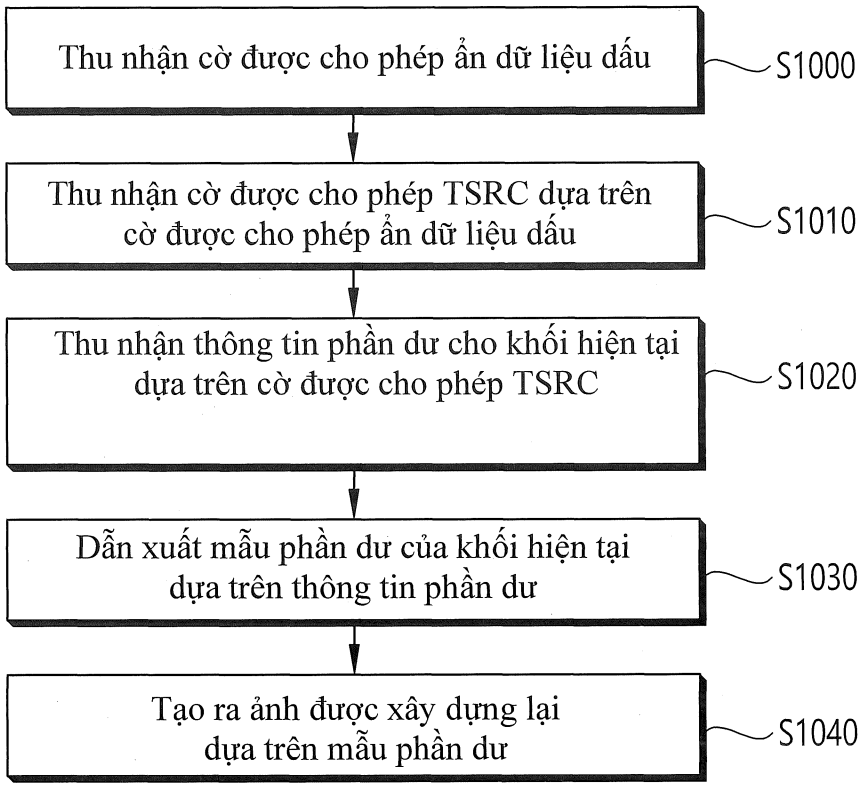

Thu nhận cờ được cho phép ẩn dữ liệu dấu
S1000
Thu nhận cờ được cho phép TSRC dựa trên cờ được cho phép ẩn dữ liệu dấu
S1010
Thu nhận thông tin phần dư cho khối hiện tại dựa trên cờ được cho phép TSRC
S1020
Dẫn xuất mẫu phần dư của khối hiện tại dựa trên thông tin phần dư
S1030
Tạo ra ảnh được xây dựng lại dựa trên mẫu phần dư
S1040

FIG. 11

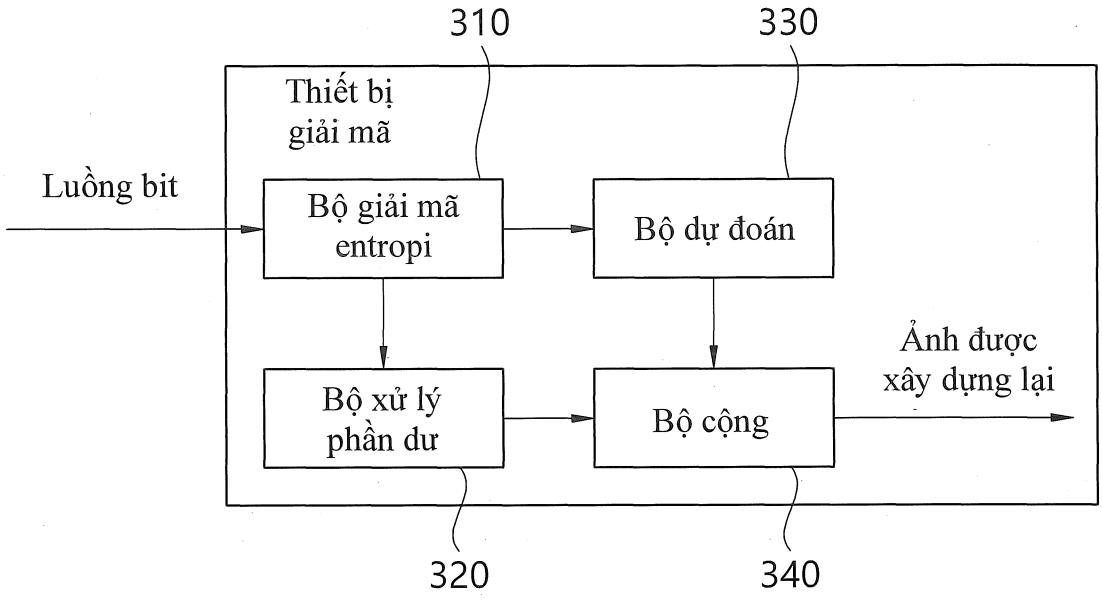
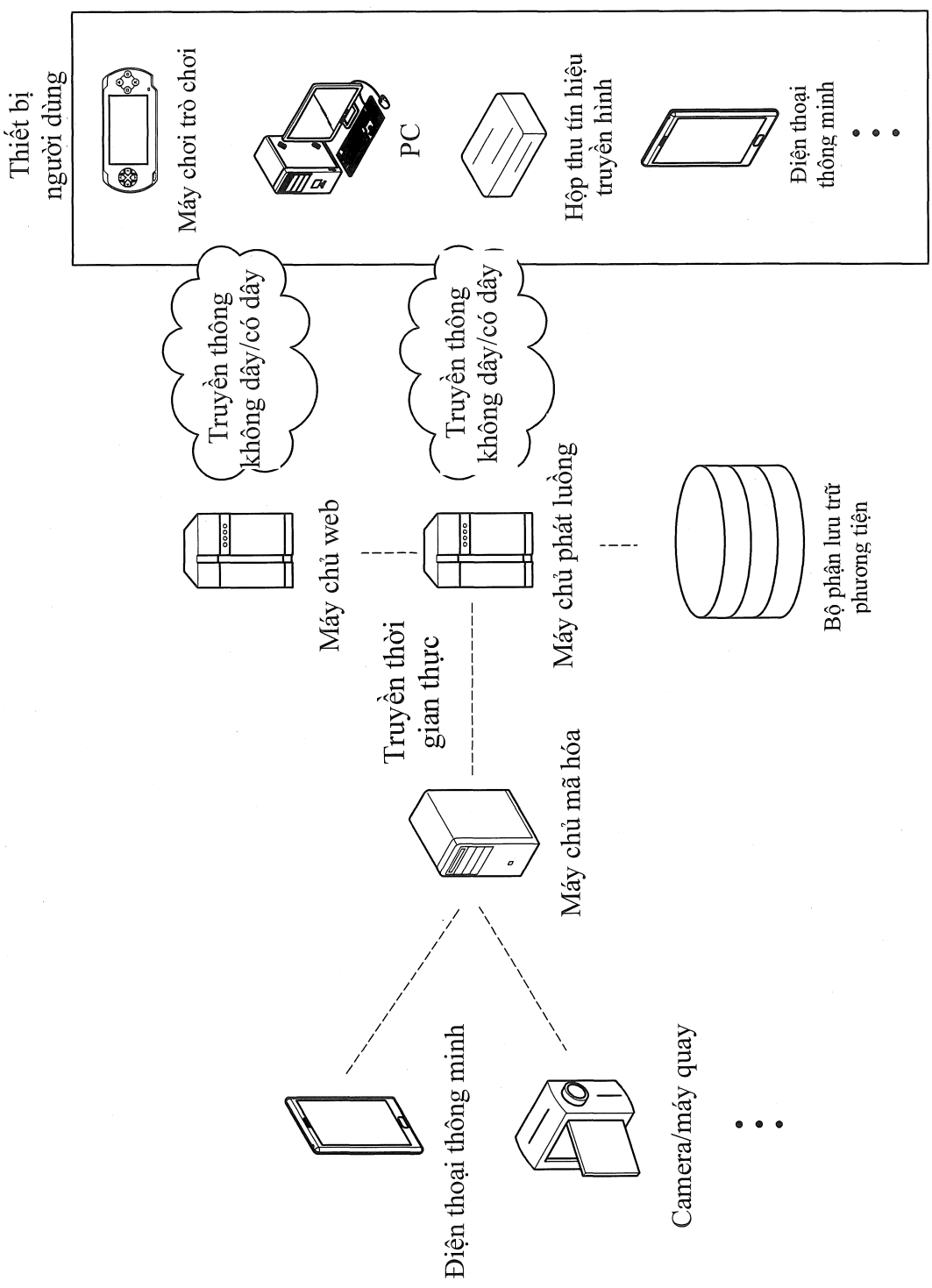

310
330
Thiết bị giải mã
Luồng bit
Bộ giải mã entropi
Bộ dự đoán
Ảnh được xây dựng lại
Bộ xử lý phần dư
Bộ cộng
320
340

FIG. 12


Thiết bị người dùng
Máy chơi trò chơi
PC
Hộp thu tín hiệu truyền hình
Điện thoại thông minh
...
Truyền thông không dây/có dây
Truyền thông không dây/có dây
Máy chủ web
Máy chủ phát luồng
Bộ phận lưu trữ phương tiện
Truyền thời gian thực
Máy chủ mã hóa
Điện thoại thông minh
Camera/máy quay
...