토큰 시퀀스 정보를 이용한 MPEG RVC 프레임워크 기능부 구현

박민수 김현규 이승욱 장의선
한양대학교
esjang@hanyang.ac.kr

Token Sequence-based FU Implementation for MPEG RVC Framework

Minsoo Park Hyungyu Kim Seungwook Lee Euee. S. Jang
Hanyang University

요약

본 논문에서는 MPEG RVC(Reconfigurable Video Coding) 프레임워크에서의 효율적인 복호화기 생성을 위한 토큰 정렬 방법을 제안한다. MPEG RVC 프레임워크는 현존하는 비디오 표준을 모듈 단위로 나누어 그에 대한 입력 및 출력의 동작을 명시하고 있다. 현재의 RVC 프레임워크 표준은 각 기능부(functional unit)들의 입력 및 출력의 행동만을 서술할 뿐 특정 비디오 코덱에서 고유적으로 정의하는 각 기능부 사이에 소비되는 토큰의 계산 모델(model of computation)을 제공하지 않는다. 이러한 점은 계산 모델이 다른 환경에서 RVC 프레임워크 솔루션을 개발하는데 큰 어려움으로 작용한다. 따라서 효율적인 RVC 복호화기의 구성을 위해 복호화 기술 정보 상의 명백한 토큰 정렬 정보를 이용하여 RVC 프레임워크의 기능부들 사이의 행동을 결정지어 주는 방법을 제안한다. 제안하는 방법은 토큰 정렬에 의해 계산 모델을 명확하게 해주고 개발자로 하여금 코덱 개발 단계에서의 디버깅 및 테스팅에 따른 부담을 줄여줄 것이다.

1. 서론

비디오 코덱 및 플랫폼들의 다양화되면서 서로 다른 시스템에서의 코덱의 구현에 따른 부담은 계속해서 증가하고 있다. 이러한 코덱의 구현에서의 부담을 줄이기 위해 MPEG에서는 RVC 프레임워크(Reconfigurable Video Codec Framework) 표준을 제정하였다 [1][2]. MPEG RVC 프레임워크는 비디오 복호화기를 기능부(Functional Unit)로 나누어 모듈화하고, 비트스트림 정보를 포함하는 복호화 기술 정보(decoder description)와 각 기능부의 연결 정보를 제공한다. RVC 프레임워크는 이러한 모듈화된 디자인을 통해 다수의 코덱에서의 특정 기능부의 재사용을 가능하게 하며 이는 코덱 개발자의 설계, 디버깅 및 테스트에 도움을 주고자 하는 목적을 가진다.

현재 RVC Framework 표준에서는 비디오 코덱을 구성하는 각 기능부를 정의하면서 그 입력 및 출력에 따른 행동을 명시하고 있다 [1][2]. 하지만 서로 연결된 기능부들 사이에서의 계산 모델(model of computation) [3] 정보를 정의하고 있지 않기에 서로 다른 플랫폼 상에서 RVC 프레임워크를 이용하여 코덱을 구현하려 할 때 다음과 같은 문제들이 발생한다.

1) 기능부의 재사용(reuse) 불가: 데이터 입력 및 출력 규약이 다른 경우 같은 기능을 가진 기능부 간에 호환성을 보장할 수 없다.
2) 다른 플랫폼에서의 기능부 상호 운용(interoperability) 불가: 계산 모델이 다른 플랫폼에서 특정 기능부의 정상 동작 여부를 보장 할 수 있는 방법이 없다.
3) 모듈 단위 디버깅 및 동일성 테스트의 어려움: 위의 두 가지 요소로 인해 표준화 된 방법으로 기능부의 정상 동작 여부를 검증하고 디버깅 및 테스팅하는 것이 불가능하다.

이러한 문제들은 RVC 프레임워크의 본래 목적인 비디오 코덱의 쉬운 구성 및 검증을 근본적으로 힘들게 한다. 따라서 본 논문에서는 위에 서술한 문제들을 해소하기 위해 RVC 프레임워크에서의 토큰 정렬 정보의 필요성을 주장한다. 토큰 정렬 방법의 추가는 RVC 프레임워크에서의 각 기능부들의 계산 모델을 명확하게 해줄 것이다.

본 논문의 구성은 다음과 같다. 제 2장에서는 RVC 프레임워크의 개요를 설명하고, 제 3장에서는 본 논문에서 제안하는 토큰 정렬 정보를 소개한다. 제 4장에서는 토큰 정렬에 따른 실험 결과를 비교하고, 제 5장에서 결론을 맺는다.

2. 토큰 정렬 문제 분석

현재 MPEG RVC는 CCR(Codec Configuration Representation) [1] 과 VTL(Video Tool Library) [2] 의 두 개의 표준으로 이루어져있다. CCR은 비트스트림 구문 정보(Bitstream syntax description) 및 기능부 연결 정보(Functional unit network description)를 포함하는 복호화 기술정보를 정의하고 있으며, VTL에서는 비디오 코덱의 기능부를 정의한다. 즉, MPEG RVC 프레임워크는 그림 1과 같이 CCR에서 정의된 복호화 기술 정보에 따라 VTL에서 제공하는 기능부를 조합하여 복호화기를 구성한다.

This work was supported by the strategic technology development program of MCST/MKE/KEIT. [KI001798, Development of Full 3D Reconstruction Technology for Broadcasting Communication Fusion]

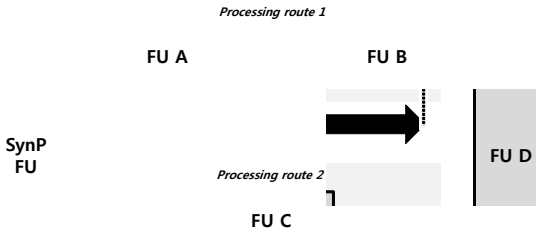
[그림 1] MPEG RVC 프레임워크의 복호화기 구성

연결된 파서 및 기능부들은 그림 2와 같이 서로 독립적으로 행동하는 일종의 데이터 플로우 모델 (dataflow model) [4] 을 형성한다. 네트워크를 이룬 기능부들은 토큰이라 불리우는 데이터를 서로의 정해진 행동에 따라 주고 받는다. 각 기능부들은 입력된 토큰에 따라서 행동이 결정되고 다른 기능부들의 연결 관계에 상관없이 토큰을 출력하게 된다. 입력된 토큰에 따라서 기능부의 행동이 결정되므로 네트워크 상에서의 토큰의 입력 및 출력을 조절하는 것이 중요하다. 하지만 현재의 MPEG RVC 표준에서는 기능부의 입력 및 출력에 따른 행동 및 단순 연결 관계만을 정의하는 반면 토큰의 타이밍 정보를 서술하고 있지 않다. 이는 RVC 프레임워크에서 다음과 같은 문제를 발생시킨다 [5].

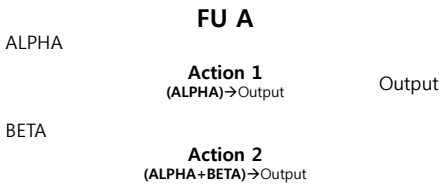
[그림 2] 기능부 네트워크 예제

그림 3에서는 기능부 D가 기다리고 있는 두 개의 입력 중 무엇이 먼저 들어올지 모르는 토큰 추월 문제가 발생할 수 있다. 두 개의 Processing Route 중에 기능부 처리 속도에 따라서 기능부 D에 도달하는 입력 토큰의 타이밍이 다를 가능성이 있기에 발생하는 문제이다. 이 경우 기능부 D에서 입력 토큰에 따른 제어를 정확히 하지 않는다면 예정과 다른 행동을 하게 되어 오류가 발생하게 된다.

또한 그림 4에서는 토큰 페어링 문제를 설명한다. 이는 ALPHA와 BETA의 입력이 동시에 들어왔을 경우 ALPHA만을 처리하는 ACTION1과 ALPHA와 BETA 전부를 처리하는 ACTION2 중 어느 행동을 할지 확정하지 못하는 문제를 말한다. 이것은 기능부 A가 여러 가지 입력 토큰에 따른 행동이 미리 결정되어 있지 않기에 벌어지는 문제이다. 따라서 이러한 비결정적(Non-deterministic) [6][7] 기능부들을 제어하기 위한 토큰 입력 정보가 반드시 필요하게 된다.



[그림 3] 토큰 추월 문제



[그림 4] 토큰 페어링 문제

3. 토큰 정렬 방법

제 2장에서 제시한 토큰 정렬에 따른 문제를 해결하기 위해 **그림 5**처럼 각 기능부를 감싸는 토큰 스케줄러를 고안하였다. 그림에서는 기능부가 토큰의 입력이 발생하기 전에 토큰 스케줄러를 한 단계 거치면서 해당 기능부가 필요로 하는 토큰을 스케줄러가 상황에 맞게 기능부로 보내주는 역할을 하게 된다.

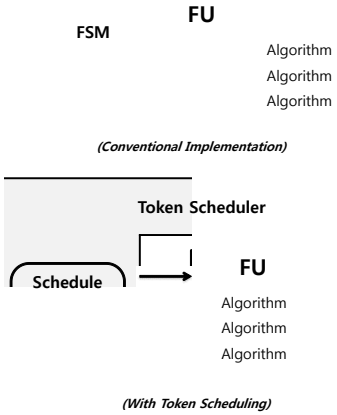
[그림 5] 토큰 정렬을 위한 스케줄러

그림 6은 스케줄러가 없는 기능부와 스케줄러가 포함된 기능부를 보여준다. 기존의 기능부는 내부에 유한 상태 기계(Finite State Machine, FSM)를 이용하여 토큰 입출력 과정을 제어하고 있다. 이렇게 기능부에 내포된 토큰 입출력 과정은 기능부를 구현하는 방법에 따라 달라질 수 있다. 따라서 상기 언급하였던 기능부 재사용 문제를 유발한다.

우리는 이렇게 기능부 내부에서 제어되던 토큰 입출력 과정을 외부에서 부가 정보로 제공하는 방식을 제안한다. 보다 구체적으로는, 기능부 내부 토큰의 입력 및 출력 제어 과정을 외부의 스케줄러가 제어하도록 하는 방법이다. 이 스케줄러의 동작은 플랫폼 독립적인 형태의 토큰 정렬 정보 (Token ordering information)에 의해 제어된다.

이렇게 토큰의 입력 및 출력 과정을 외부에서 제어함으로써 기능부의 동작 과정에서 존재하던 비결정적 요소를 제거할 수 있으며, 결과적으로 기능부를 보다 표준화된 방법으로 구현할 수 있게 한다. 이를

통하여 상술하였던 기능부 재사용 문제를 해결한다. 또한, 스케줄러로 하여금 플랫폼 독립적인 외부 정보를 읽어 토큰 정렬을 수행하도록 함으로써 플랫폼의 계산 모델에 따라 토큰 입출력 방법이 바뀌는 것을 예방한다. 이는 기능부의 플랫폼 간 상호 운용 또한 가능하도록 한다.



[그림 6] 기존 기능부와 스케줄러가 포함된 기능부

또한, 독립된 스케줄러를 이용하여 같은 정보를 나타내는 일련의 토큰들의 전달 순서를 각 기능부 별로 제어할 수 있게 된다. 동일한 정보라 할지라도 적절한 순서에 의해 토큰이 전달되는 경우 기능부의 처리 속도를 향상시킬 수 있다. 기존에는 이 측면에서의 최적화를 수행하기 위해서는 기능부 내부의 처리 과정 전반에 개입할 필요가 있었으나, 토큰 정렬 과정을 외부에 둠으로서 그 부담을 경감할 수 있다.

4. 실험

토큰 정렬 기술의 실제 구현 가능성을 탐색하고자, 우리는 상기에 언급하였던 토큰 정렬 기술의 장점들 가운데 토큰 정렬 순서가 실제 기능부 동작 시간에 끼치는 영향을 평가하였다.

표 1은 MPEG-4 Simple Profile 코덱의 기능부 중 하나인 IS(Inverse Scan) 기능부의 토큰 스케줄링 정보를 나타낸다. 표에 따르면 AC_PRED_DIR 토큰이 들어온 후 QFS_AC 토큰 64개를 입력받아 PQF_AC 토큰 64개를 출력한다. 이는 AC_PRED_DIR 토큰이 입력되지 않으면 IS 기능부는 동작을 하지 않고 해당 토큰이 입력될 때까지 대기를 한다는 것을 나타낸다. 이러한 이유에서 해당 기능부는 적합한 방식으로 토큰이 전달되지 않을 경우 전체적인 복호화 과정에서 상당한 오버로드를 줄 것으로 예상되었다.

[표 1] IS 기능부의 토큰 스케줄링 정보

Algo_IS_ZigzagOrAlternateHorizontalVertical_8x8	
Input	AC_PRED_DIR QFS_AC
Output	PQF_AC
Order	1. AC_PRED_DIR→QFS_AC*64→PQF_AC*64 2. AC_PRED_DIR→QFS_AC*64, PQF_AC*64

우리는 동일한 IS 기능부를 이용하여 두 개의 서로 다른 토큰 정렬 방식을 사용하여 그 처리 속도를 측정, 비교하였다. 토큰 정렬 방식을 적용하는 과정에서, 우리가 제안하는 토큰 정렬 정보 및 스케줄러에 기반한 설계가 적용되었다. 비교에 사용된 비트스트림은 다음과 같다:

1) 표준 (standard) 표본: 표준 MPEG RVC Simulation Model에서 토큰을 정렬하는 순서를 재현한 정렬 방법
2) 결함 (defected) 표본: 기능부의 동작 절차에 관여하는 AC_PRED_DIR 토큰의 전달 순서를 고의로 지연시킨 정렬 방법

표 2는 위에서 설명한 두 가지 표본을 이용해 실제 IS 기능부를 동작시킨 뒤 그 수행 시간을 측정한 결과이다. 표 2에서 보여지듯, 결함 표본의 토큰 정렬 방식을 따를 경우 표준 표본의 경우보다 처리 시간이 두 배 이상 느리다는 것을 볼 수 있다. 이 결과로부터, 토큰을 적합한 방식으로 정렬, 제공하는 것이 특정 기능부의 수행 시간에 큰 영향을 줄 수 있음이 확인된다. 따라서 토큰 정렬 정보를 최적화된 방법으로 제공하는 것을 통해 코덱의 복호화 시간을 단축할 수 있는 가능성 또한 시사된다.

[표 2] 토큰의 정렬 여부에 따른 비트스트림의 복호화 처리 속도

기능부	비트스트림	처리 시간(s)
Algo_IS_ZigzagOrAlternateHorizontalVertical_8x8	원본	204.87
	최악	476.03

5. 결론

본 논문에서는 RVC 프레임워크에서의 토큰 정렬에 따른 문제와 그 문제를 해결할 수 있는 토큰 정렬 방법에 대해서 제안하였다. 토큰 정렬 방법을 이용하여 RVC 프레임워크에서 발생하는 재사용, 상호 운용, 디버깅 및 테스팅 문제가 해결 가능함을 보여주었다. 또한, 실험을 통해 토큰 정렬에 따라 기능부의 처리 속도가 영향을 받는다는 것을 확인 할 수 있었다.

토큰 정렬 정보가 없는 현재의 RVC 프레임워크 표준은 본 논문에서 지적한 문제들로 인해 RVC 프레임워크가 가진 본래 목적인 비디오 코덱의 효율적인 구성이 힘들다는 위험성이 내재되어 있다. 따라서 이러한 위험을 없애고 보다 효율적인 코덱 구성을 위해 토큰 정렬 정보가 제공되어야 할 것이다.

6. 참고 문헌

[1] “Information technology – MPEG systems technologies – Part 4: Codec configuration representation”, ISO/IEC FDIS 23001-4:2009
[2] “Information technology – MPEG video technologies – Part 4: Video tool library”, ISO/IEC FDIS 23002-4:2009
[3] Edward A. Lee and Alberto Sangiovanni-Vincentelli, “A Framework for Comparing Models of Computation”, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 17, NO. 12, DECEMBER 1998
[4] EDWARD ASHFORD LEE AND DAVID G. MESSERSCHMI′TT,

"Static Scheduling of Synchronous Data Flow Programs for Digital Signal Processing", IEEE TRANSACTIONS ON COMPUTERS, VOL. '2-36, NO. 1 . JANUARY 1987

[5] Hyungyu Kim, Sowon Kim, Seungwook Lee, Bonki Koo, Minsoo Park, Taehee Lim, and Euee S. Jang., "Proposal of Token Scheduling Description in Reconfigurable Codecs (RVC/RGC)", ISO/IEC JTC1/SC29/WG11 MPEG2010, M17930, July 2010, Geneva, Switzerland

[6] SHUVRA S. BHATTACHARYYA AND EDWARD A. LEE , "Scheduling Synchronous Dataflow Graphs for Efficient Looping", Journal of VLSI Signal Processing, 6, 271-288 (1993)

[7] Edward A. Lee and Alberto Sangiovanni-Vincentelli, "THE TAGGED SIGNAL MODEL - A PRELIMINARY VERSION OF A DENOTATIONAL FRAMEWORK FOR COMPARING MODELS OF COMPUTATION", Memorandum UCB/ERL M96/33 June 4, 1996