# RTMP FMT TYPE

/\*\*

\* 6.1.2. Chunk Message Header

\* There are four different formats for the chunk message header,

\* selected by the "fmt" field in the chunk basic header.

\*/

// 6.1.2.1. Type 0

// Chunks of Type 0 are 11 bytes long. This type MUST be used at the

// start of a chunk stream, and whenever the stream timestamp goes

// backward (e.g., because of a backward seek).

#define RTMP\_FMT\_TYPE0 0

// 6.1.2.2. Type 1

// Chunks of Type 1 are 7 bytes long. The message stream ID is not

// included; this chunk takes the same stream ID as the preceding chunk.

// Streams with variable-sized messages (for example, many video

// formats) SHOULD use this format for the first chunk of each new

// message after the first.

#define RTMP\_FMT\_TYPE1 1

// 6.1.2.3. Type 2

// Chunks of Type 2 are 3 bytes long. Neither the stream ID nor the

// message length is included; this chunk has the same stream ID and

// message length as the preceding chunk. Streams with constant-sized

// messages (for example, some audio and data formats) SHOULD use this

// format for the first chunk of each message after the first.

#define RTMP\_FMT\_TYPE2 2

// 6.1.2.4. Type 3

// Chunks of Type 3 have no header. Stream ID, message length and

// timestamp delta are not present; chunks of this type take values from

// the preceding chunk. When a single message is split into chunks, all

// chunks of a message except the first one, SHOULD use this type. Refer

// to example 2 in section 6.2.2. Stream consisting of messages of

// exactly the same size, stream ID and spacing in time SHOULD use this

// type for all chunks after chunk of Type 2. Refer to example 1 in

// section 6.2.1. If the delta between the first message and the second

// message is same as the time stamp of first message, then chunk of

// type 3 would immediately follow the chunk of type 0 as there is no

// need for a chunk of type 2 to register the delta. If Type 3 chunk

// follows a Type 0 chunk, then timestamp delta for this Type 3 chunk is

// the same as the timestamp of Type 0 chunk.

#define RTMP\_FMT\_TYPE3 3

# RTMP CID

/\*\*

\* the chunk stream id used for some under-layer message,

\* for example, the PC(protocol control) message.

\*/

#define RTMP\_CID\_ProtocolControl 0x02

/\*\*

\* the AMF0/AMF3 command message, invoke method and return the result, over NetConnection.

\* generally use 0x03.

\*/

#define RTMP\_CID\_OverConnection 0x03

/\*\*

\* the AMF0/AMF3 command message, invoke method and return the result, over NetConnection,

\* the midst state(we guess).

\* rarely used, e.g. onStatus(NetStream.Play.Reset).

\*/

#define RTMP\_CID\_OverConnection2 0x04

/\*\*

\* the stream message(amf0/amf3), over NetStream.

\* generally use 0x05.

\*/

#define RTMP\_CID\_OverStream 0x05

/\*\*

\* the stream message(amf0/amf3), over NetStream, the midst state(we guess).

\* rarely used, e.g. play("mp4:mystram.f4v")

\*/

#define RTMP\_CID\_OverStream2 0x08

/\*\*

\* the stream message(video), over NetStream

\* generally use 0x06.

\*/

#define RTMP\_CID\_Video 0x06

/\*\*

\* the stream message(audio), over NetStream.

\* generally use 0x07.

\*/

#define RTMP\_CID\_Audio 0x07

# RTMP HEAD TYPE

#define RTMP\_MSG\_SetChunkSize 0x01

#define RTMP\_MSG\_AbortMessage 0x02

#define RTMP\_MSG\_Acknowledgement 0x03

#define RTMP\_MSG\_UserControlMessage 0x04

#define RTMP\_MSG\_WindowAcknowledgementSize 0x05

#define RTMP\_MSG\_SetPeerBandwidth 0x06

#define RTMP\_MSG\_EdgeAndOriginServerCommand 0x07

#define RTMP\_MSG\_AMF3CommandMessage 17 // 0x11

#define RTMP\_MSG\_AMF0CommandMessage 20 // 0x14

#define RTMP\_MSG\_AMF0DataMessage 18 // 0x12

#define RTMP\_MSG\_AMF3DataMessage 15 // 0x0F

#define RTMP\_MSG\_AMF3SharedObject 16 // 0x10

#define RTMP\_MSG\_AMF0SharedObject 19 // 0x13

#define RTMP\_MSG\_AudioMessage 8 // 0x08

#define RTMP\_MSG\_VideoMessage 9 // 0x09

#define RTMP\_MSG\_AggregateMessage 22 // 0x16

# RTMP AMF0

#define RTMP\_AMF0\_Number 0x00

#define RTMP\_AMF0\_Boolean 0x01

#define RTMP\_AMF0\_String 0x02

#define RTMP\_AMF0\_Object 0x03

#define RTMP\_AMF0\_MovieClip 0x04 // reserved, not supported

#define RTMP\_AMF0\_Null 0x05

#define RTMP\_AMF0\_Undefined 0x06

#define RTMP\_AMF0\_Reference 0x07

#define RTMP\_AMF0\_EcmaArray 0x08

#define RTMP\_AMF0\_ObjectEnd 0x09

#define RTMP\_AMF0\_StrictArray 0x0A

#define RTMP\_AMF0\_Date 0x0B

#define RTMP\_AMF0\_LongString 0x0C

#define RTMP\_AMF0\_UnSupported 0x0D

#define RTMP\_AMF0\_RecordSet 0x0E // reserved, not supported

#define RTMP\_AMF0\_XmlDocument 0x0F

#define RTMP\_AMF0\_TypedObject 0x10

// AVM+ object is the AMF3 object.

#define RTMP\_AMF0\_AVMplusObject 0x11

// origin array whos data takes the same form as LengthValueBytes

#define RTMP\_AMF0\_OriginStrictArray 0x20

// User defined

#define RTMP\_AMF0\_Invalid 0x3F