DVS Digital Video Systems AG
Krepenstr. 8
30165 Hannover
GERMANY

VDCP-Command Extensions

Quick Command Structure Reminder:

serial line signal
- conforms to EIA RS-422A
- full duplex, asynchronous, bit serial, word serial
- 38.400 kb/s 1 start bit, 8 data bits, 1 parity bit (odd), 1 stop bit
- byte time .286 msec
- 116 bytes per frame at 29.97fps

a VDCP command consists of a fixed byte structure:

STX	BC	Type/UA	CMD-2	DATA-1	DATA-2	DATA-N	CHKSUM
0x02	byte count	CMD-1		data	data	data	(see below)

STX (Start of Text): 0x02 for commands and can be 0x04 or 0x05 for replies

BC: byte count between BC and CHKSUM

Type/UA: 2 nibbles, high nibble is type (see commands), low nibble is UA (unit address), 0 for regular VDCP commands and 14 is used for DVS extensions.

CMD-2: as stated in commands.

DATA: data bytes depending on command

CHKSUM: 2's complement of the least significant byte of the sum of all command and data bytes from the first command byte to immediately before the checksum

the response to a VDCP command is also dictated by the standard:
- command type 0, 1, 2, 8, 0xA: either ACK (0x04) or NAK (0x05)
- command type 3, 0xB: most significant bit of CMD-2 is set, so 0x29 becomes 0xa9 in reply.

For a full description see the VDCP standard.

Currently supported official VDCP commands

BC	CMD-1	CMD-2	Name		BC	CMD-1	CMD-2	Name
02	10	00	Stop				04	ACK
02	10	01	Play				04	ACK
02	10	02	Record				04	ACK
02	10	04	Still				04	ACK
02	10	05	Step				04	ACK
02	10	06	Continue				04	ACK
03	10	07	Jog				04	ACK
05	10	08	Vari.Play				04	ACK
03	20	21	ClosePort				04	ACK
03	20	22	SelectPort				04	ACK
0A/xx	20/A0	24	PlayCue				04	ACK
0E/xx	20/A0	23	RecordInit				04	ACK
12/xx	20/A0	25	CueWithData				04	ACK
12/xx	20/A0	2C	RecordInitWithData				04	ACK
04	30	01	OpenPort		03	30	81	Grant
02	30/B0	02	Next		xx	30/B0	82	NextID(s)
03	30	05	PortStatusRequest		xx	30	85	Status
03	30	06	PositionRequest		07	30	86	Position
02	30/B0	07	ActiveIDRequest		0B/xx	30/B0	87	ActiveID
02	30	08	DeviceTypeRequest		xx	30	88	DeviceType
03	30	10	SystemStatusRequest		xx	30	90	SystStatus
02	30/B0	11	ListofID(s)		xx	30/B0	91	FirstIDs
0A/xx	30/B0	14	IDSizeRequest		06	30/B0	94	IDSize
0A/xx	30/B0	16	IDRequest		05	30/B0	96	IDPresence
02	30/B0	18	IDsAddedListRequest		xx	30/B0	98	IDsAddedList
02	30/B0	19	IDsDeletedListRequest		xx	30/B0	99	IDsDeletedList

All data is in the hexadecimal system.

These commands behave according to the VDCP standard.

Note that we currently only support one channel per port so OpenPort/SelectPort/ClosePort can only be used with “1”.

Note also that RecordInit/RecordInitWithData do not support writing over an already existing clip, neither in full nor in part. Additionally, RecordInitWithData allows setting a timecode offset but this is currently not used.

DVS extensions to VDCP commands:

BC	CMD-1	CMD-2	Name		BC	CMD-1	CMD-2	Name
05	1E	08	VariPlay2				04	ACK
08	1E	70	PlayLoop				04	ACK
07	1E	71	VariPlayLoop				04	ACK
15/xx	2E/AE	25	CueWithData (Speed extension)				04	ACK
04	2E	75	AudioPreset				04	ACK
03	3E	06	PositionRequest (Speed/Loop Count extension)		0C	3E	86	Position

VariPlay2 (keeps loop mode)

D1	D2	D3
Speed MSB	Speed	Speed LSB

Alternative version of VariPlay that does not reset the loop state set by PlayLoop/VariPlayLoop. This way it is possible to pause and restart playback by using VariPlay2(0.0) and VariPlay2(1.0).

VariPlayLoop

D1	D2	D3	D4	D5
Speed MSB	Speed	Speed LSB	Loop Count MSB	Loop Count LSB

Similar to VariPlay but allows to loop the current clip. Note that using VariPlayLoop(1.0, 1) will have the same effect as using VariPlay(1.0). Issuing VariPlayLoop(1.0, 0) will start a non-stop loop.

PlayLoop

D1	D2	D3	D4	D5	D6
Speed MSB	Speed	Speed LSB	Flags	Loop Count MSB	Loop Count LSB

Similar to Play but additionally looping the cued clip and offering finer control for when playback starts.

Speed: a 24 bit signed (2's complement) value, speed is calculated exactly as for VariPlay.

Flags: set of values that can be ORd together, see table below for detailed meaning.

Loop Count: if the loop-flag (0x02) is set and this value is greater than 0 it states how many loops are played back before the clip stops.If set to 0 the clip is looped back indefinitely.

0x01	*playback delayed to end of clip/loop sequence* If set, the cued clip is not played back immediately but only when the last frame of the current clip is reached. This allows to cue a clip without having to poll for the end of the current one. For an indefinite loop (count was 0) this means the next clip is started after the end of the current loop, for a fixed count this means the next clip is started after the whole loop sequence finishes.
0x02	*loop* If set, the cued clip will be looped from start to end (for positive speeds) or from end to start (for negative speeds) either forever (Loop Count is 0) or for the given number. If not set, clip will be played back just once.
0x04	*reset current loop sequence* Only meaningful if both delayed playback is requested (0x01) and a counted loop is running. If set, the currently running loop is reset so the cued clip will already play back once the last frame of the current loop iteration is reached. If not set, the behaviour is exactly as described for the "playback delayed"-flag.
0x08	*negative zero speed (start with last frame)* Only meaningful if speed is set to 0. Since there exists an ambiguity in this case about where to start playback this flag will indicate which position is desired. If set, playback starts at the last frame. If not set, playback starts with the first frame.

CueWithData (ext.)

D1..N	DN+1	DN+2	DN+3
Clipname, Offset, Duration	Speed MSB	Speed	Speed LSB

Same as the regular CueWithData command but extended by 3 additional speed hint bytes that indicate the intended initial playback speed to the realtime server for improved buffering.

Speed: a 24 bit signed (2's complement) value, speed is calculated exactly as for VariPlay.

AudioPreset

D1	D2
Preset	Preset

Defines which audio channels will be played back. DATA-1 and DATA-2 are bit masks so 16 channels can be set at once.

This command can be issued at any time but is constrained by the same latency as transport control.

Position (ext.)

D1..5	D6	D7	D8	D9	D10
Type, Position	Speed MSB	Speed	Speed LSB	Loop Count MSB	Loop Count LSB

Same arguments and reply format as for the regular PositionRequest but adds 5 bytes to describe the current speed and loop count.

Speed: a 24 bit signed (2's complement) value, speed is calculated exactly as for VariPlay.

Loop Count: if Clip was started with PlayLoop for a fixed loop count then this counter will decrease after each loop beginning starting with the original loop count (so the final loop will return 1). For any other type of playback this will return 0.

Examples:

Regular VDCP playback

```
OpenPort(1,1)
SelectPort(1)

/* cue clip to play back next */
PlayCue("neko")

/* at the right time, start play,
   real playback will then start delayed by the (fixed) transport latency */
Play()

/* as early as possible cue next clip (current clip continues to play) */
PlayCue("inu")

/* when time has come, start playback of second clip */
Play()
[…]

Stop()

ClosePort(1)
```

Loop playback

```
OpenPort(1,1)
SelectPort(1)

/* select clip to play back in a loop */
PlayCue("nezumi")

/* start to loop clip, playback will start immediately (after the fixed transport latency)
   and will loop for 10 times */
PlayLoop(1.0, 0x02, 10)

/* cue next clip at some point,
   previous clip continues to loop */
PlayCue("shishi")

/* decide that after the loop sequence finishes the cued clip will
  replace it and will loop indefinitely since the count is not set */
PlayLoop(1.0, 0x03, 0)

[…]

Stop()

ClosePort(1)
```

Playback in reverse

```
OpenPort(1,1)
SelectPort(1)

/* cue the next clip which is to be played back in reverse */
CueWithData("inu", 00:05:00:00, 00:00:25:00, -1.0)
PlayLoop(-1.0, 0x00)

/* the "inu" clip will be played back immediately
   (after the fixed transport latency) */

Stop()
ClosePort(1)
```

Clearing a cued clip

```
/* sometimes it may be necessary to clear an already cued clip that
   was set with CueWithData or PlayCue. So this is shown here. */

OpenPort(1,1)
SelectPort(1)

/* cue a clip */
PlayCue("inu")

/* start playback of that clip */
Play()

/* cue another clip and use PlayLoop so it would get played back
   once the "inu" clip will finish */
PlayCue("neko")
PlayLoop(1.0, 0x01)

/* now some schedule changed and we no longer want the "neko" clip to
   be played back next so we simply cue an empty clip which will clear
   any cued clip so the "inu" clip will simply get shown completely and
   afterwards playback will stop */

PlayCue("")

ClosePort(1)
```

Document History

17Dec2009	Initial version
01Mar2010	adds command 311 (ListofIDs), 302 (Next), drop no longer relevant example
04Mar2010	corrects BC for ActiveID response
24Mar2010	add VariPlay2, VariPlayLoop and adds “reset loop sequence” flag bit to PlayLoop
21Apr2010	adds “negative zero” flag to PlayLoop
16Dec2010	document cleanup
14Oct2011	adds command 102 (Record), 223 (RecordInit), 22C (RecordInitWithData), 318 (IDsAddedListRequest), 319 (IDsDeletedListRequest)
21Nov2011	add descriptions