Extend Format and Streaming uAPI

Hans Verkuil
Cisco Systems Norway

Problems in Current uAPI

- Single vs Multi planar makes the uAPI overly complicated.
- Lack of support for DRM modifiers.
- Lack of support for gaps between planes in a single buffer due to different alignment requirements.
- An RFC patch series introducing new format and streaming APIs was initiated in 2019, and is currently at v7 (with several people working on it), but it’s still in RFC.

Proposed new VIDIOC_G/S_EXT_PIX_FMT

```
struct v4l2_plane_pix_format {
        __u32           sizeimage;
        __u32           bytesperline;
        __u16           reserved[6];
}
struct v4l2_ext_pix_format {
        __u32                           type;
        __u32                           width;
        __u32                           height;
        __u32                           pixelformat;
        __u64                           modifier;
        __u32                           field;
        __u32                           colorspace;

        struct v4l2_plane_pix_format    plane_fmt[VIDEO_MAX_PLANES];
        __u8                            flags;
        union {
                __u8                            ycbcr_enc;
                __u8                            hsv_enc;
        };
        __u8                            quantization;
        __u8                            xfer_func;
        __u32                           reserved[10];
}
```

Proposed new VIDIOC_G/S_EXT_PIX_FMT

- Struct v4l2_fmtdesc takes two reserved fields from the reserved array and uses them to expose the modifier.
- The pixelformat field will only support single-planar variants (really: single-buffer). The multi-planar variants (i.e. where each plane has its own buffer) will be signaled using the modifier field.
- Colorspace properties can be set by userspace, so the V4L2_PIX_FMT_FLAG_SET_CSC flag is no longer needed. Question: do we want this? Or just keep the SET_CSC behavior?

Proposed new v4l2_ext_buffer

```
struct v4l2_ext_plane {
	__u32 offset;
	__u32 bytesused;
	union {
		__u32 mmap_offset;
		__u64 userptr;
		__s32 dmabuf_fd;
	} m;
	__u32 reserved[6];
};

struct v4l2_ext_buffer {
	__u32 index;
	__u32 type;
	__u32 field;
	__u32 sequence;
	__u64 flags;
	__u64 timestamp;
	__u32 memory;
	__s32 request_fd;
	struct v4l2_ext_plane planes[VIDEO_MAX_PLANES];
	__u32 reserved[10];
};
```

Proposed new v4l2_ext_buffer

- Do we still want to support USERPTR in this new API?
- The API is much easier to use since there is no difference anymore between single and multi planar.

Open Questions

- There is a translation layer so the new uAPI can be used with old drivers that do not support it.
- There is no translation layer if a driver just supports the new API, but not the old one. So the driver has to support both old and new APIs at the moment (if I understand this correctly). Is this what we want?
- The API is much easier to use since there is no difference anymore between single and multi planar.
- Are there other important features that we want to support? Or can we start to turn this RFC series into something real?

CREATE_BUFS/DELETE_BUF

- VIDIOC_CREATE_BUFS is poorly designed: it uses v4l2_format to indicate the size of the buffers that should be allocated. Originally the idea was that this is the most flexible method, but in the end only the sizeimage field was used.
- Suggestion: create a new VIDIOC_EXT_CREATE_BUFS that replaces the format field with: __u32 sizeimage[VIDEO_MAX_PLANES];
- Benjamin Gaignard is working on the DELETE_BUF ioctl: https://patchwork.linuxtv.org/project/linux-media/list/?series=10708
- Are there objections to continuing this work to add this ioctl?