Technology Consulting Company
Research, Development &
Global Standard

SoCビデオ処理支援機能を使った Androidマルチメディア処理改善の試み

松原　克弥
株式会社イーゲル

2009.12.18

背景:現状の確認

igel

CE Linux Forum

SHプロセッサでAndroid 1.5が動いた！でも…

SoCアクセラレーションH/W

SH-MobileR2（SH7723）の仕様 （プレスリリース資料より抜粋）

製品名	SH7723（R8A77230C400BG）	SH7723（R8A77230D400BG）
動作温度範囲	−20～70℃	−40～85℃
CPUコア	SH-4A（MMU搭載）	
電源電圧	内部：1.15～1.3V 外部：3.0～3.6V DDR1 SDRAM：2.3～2.7V	
最大動作周波数	400MHz	
最大処理性能	720MIPS、2.8GFLOPS（400MHz動作時）	

⋮

主な内蔵周辺機能	● ビデオI/O（500万画素カメラモジュール直結インタフェース） ● ビデオ画像処理機能（色変換、画像拡大・縮小、フィルタ処理） ● 画像ブレンド機能 ● VPU5F（H.264、MPEG-4、VC-1） ● ビデオ出力ユニット ● 24ビットTFTカラー液晶パネル対応LCDコントローラ ● 2Dグラフィックスアクセラレータ ● USB2.0 ホスト、ファンクション（ハイスピード対応）

多種多様な
on-chip
デバイス！

Android
マルチメディア処理の最適化

- マルチメディア処理をアクセラレーションH/Wへオフロード
  - ビデオデコード
  - 色変換
  - 回転、拡大縮小、クリッピング
- メインライン機能 Linux UIOを使って、ユーザ空間からデバイス利用（制御）

Linux UIO（User-space I/O）

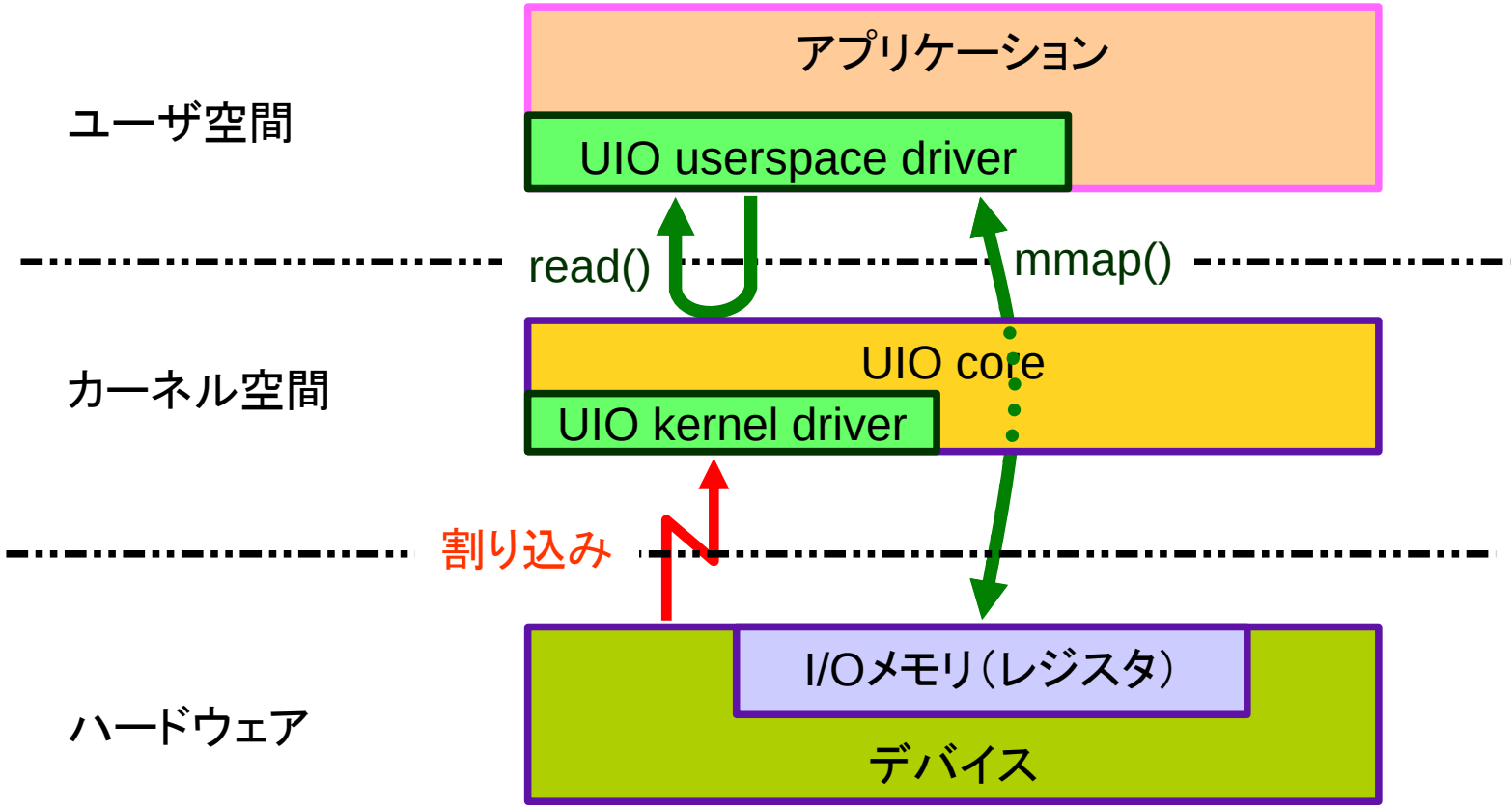
ユーザ空間からデバイスを直接制御するための汎用フレームワーク

- mmap()によるI/Oメモリへのアクセス
- 連続物理メモリの確保とアクセス
- read()による割り込み通知の受信
- ユーザ空間での他カーネル機能との連携不可
- デバイス共有のための仕組みなし

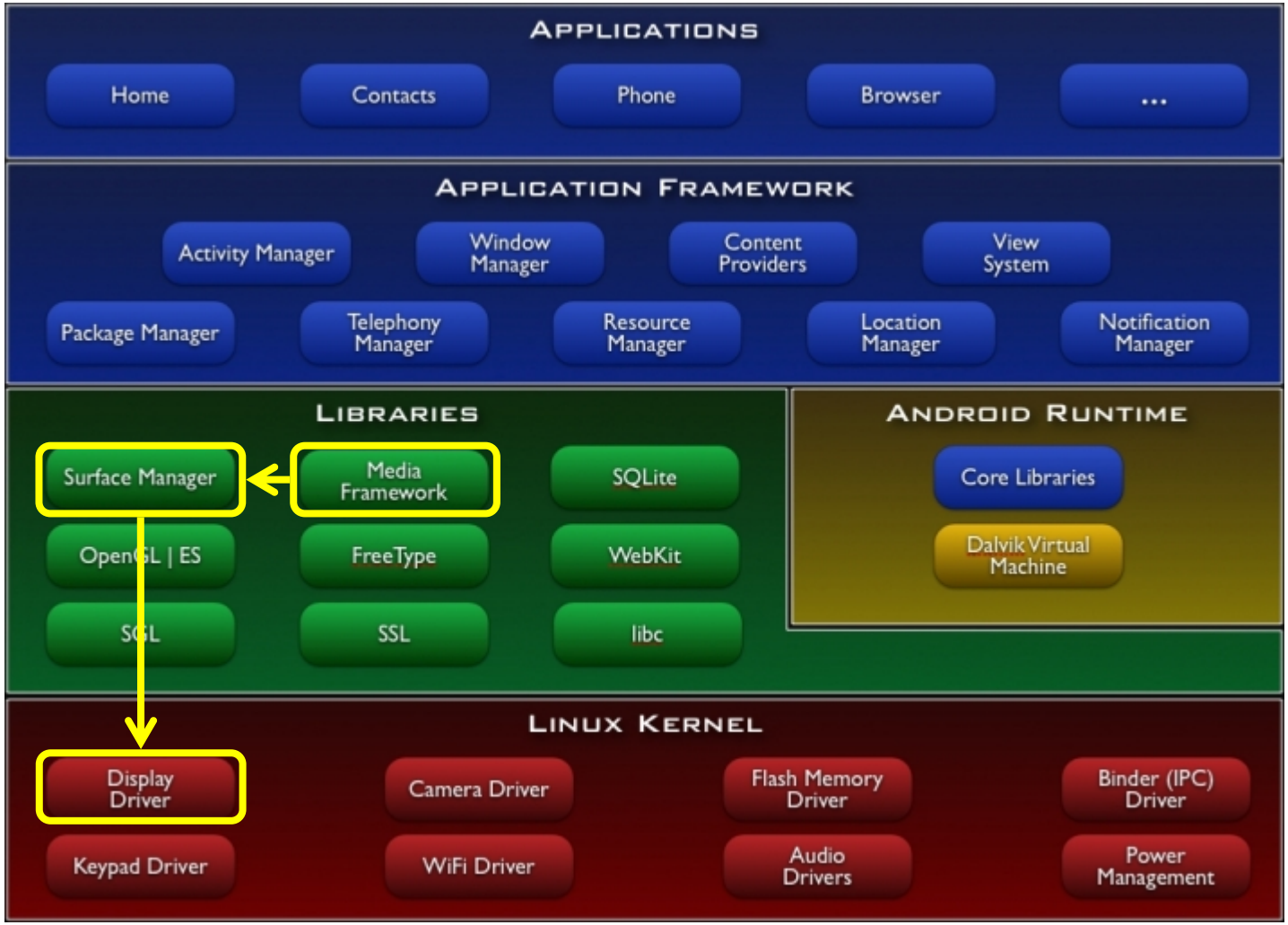
メモリコピーの削減

コンテキストスイッチの削減

UIOアーキテクチャ


ユーザ空間
アプリケーション
UIO userspace driver
read()
mmap()
カーネル空間
UIO core
UIO kernel driver
割り込み
ハードウェア
I/Oメモリ（レジスタ）
デバイス

Android
マルチメディアフレームワーク

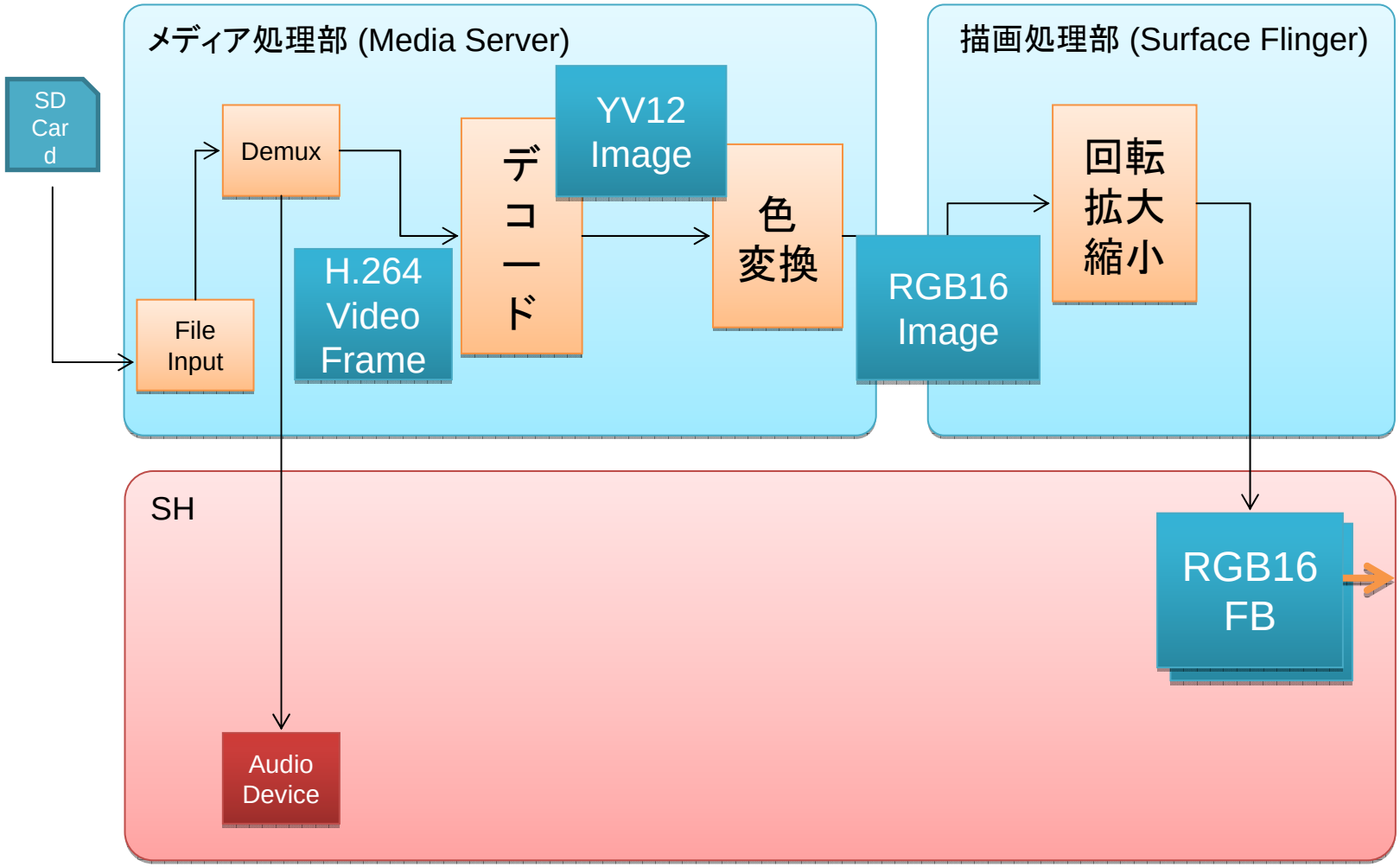


出典:
http://developer.android.com/guide/basics/what-is-android.html

Android
マルチメディアフレームワーク

- Media Framework
  - Packet Video社が開発したマルチメディアエンジン OpenCOREを採用
  - メディアプレーヤ、AVデコード／エンコードエンジン
  - Androidでは、mediaserverプロセスとして動作
  - コードツリーでは、/external/opencoreに配置
- Surface Flinger
  - フレームバッファ（描画面）の管理
  - コードツリーでは、frameworks/base/libs/{surfaceflinger、ui}に配置
- Display Driver
  - Linux fbdevを使用
  - フレームバッファの提供（mmap）とフリップの制御

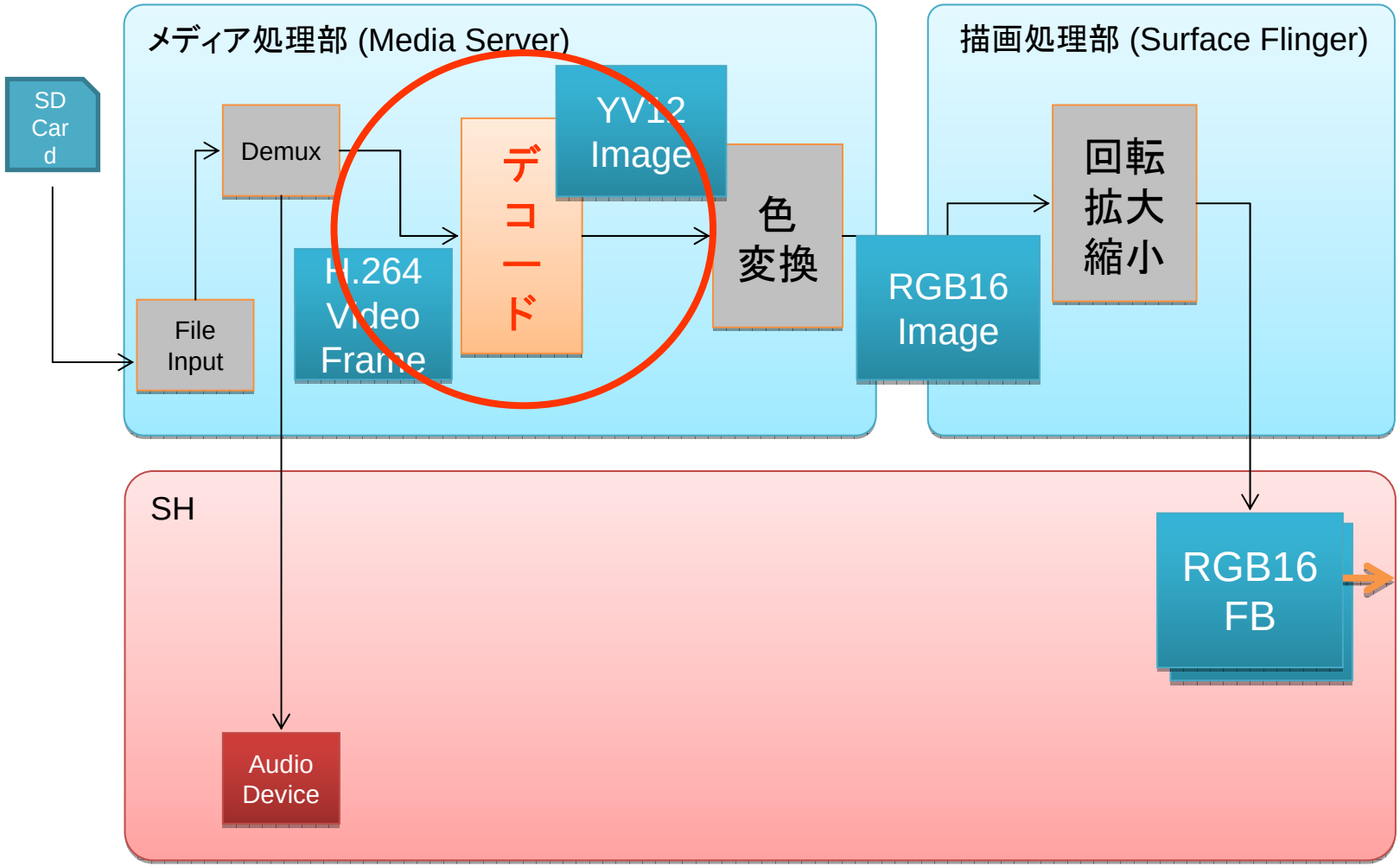
ビデオ再生の内部処理フロー



H/Wへオフロードする処理

1. ビデオデコード
2. 色変換
3. 回転・拡大縮小・クリッピング

1. ビデオデコードのオフロード



OpenCOREコーデックプラグイン仕様：OpenMAX IL

OpenCOREでは、コーデック処理モジュールの
インタフェースにOpenMAX ILコンポーネント仕様を採用

- OpenMAX
  - Khronosグループによって、マルチメディア処理向けのAPI規格として策定
  - ILが統合ライブラリレイヤ、DLがハードウェアレイヤのインタフェース規格
  - 最新仕様は、1.1.2
  - http://www.khronos.org/openmax/

OpenCOREが求めるOpenMAX ILコンポーネント仕様

- PV OpenMAX ILコンポーネントラッパーが必要
  - コンストラクタ
  - デストラクタ
  - OMX_GetHandle()等のAPI
- コードツリー（/external/opencore/doc）にドキュメントあり
  - OMX Core Integration Guide
  - OpenMAX Call Sequences

VPU UIOカーネルドライバ

CONFIG_UIO_PDRV_GENIRQ

- UIO共通の割り込みドライバ
- 自動割り込み停止
- writeシステムコールによる割り込み制御（禁止／許可）
- 割り込み共有には非対応
- 2.6.27でマージ済み

```
static struct platform_device vpu_device = {
        .name                   = "uio_pdrv_genirq",
        .id                     = 0,
        .dev                    = {
                .platform_data  = &vpu_platform_data,
        },
        .resource               = vpu_resources,
        .num_resources          = ARRAY_SIZE(vpu_resources),
};
```

VPU UIOカーネルドライバ


```
static struct uio_info vpu_platform_data = {
        .name = "VPU5",
        .version = "0",
        .irq = 60,
};

static struct resource vpu_resources[] = {
        [0] = {
                .name   = "VPU",
                .start  = 0xfe900000,
                .end    = 0xfe902807,
                .flags  = IORESOURCE_MEM,
        },
        [1] = {
                /* place holder for contiguous memory */
        },
};
```

VPUミドルウェア

- libshcodecs：VPUを使ったエンコード／デコードを支援するミドルウェア
- omxil-sh：SH アクセラレーションH/Wを使ったコーデックエンジンを実現するOpenMAX ILコンポーネント
  - Bellagio OpenMAX ILがベース

BellagioとOpenCOREとの接続

- 準拠しているOpenMAX ILバージョンの差異
  - Android 1.5のOpenCOREはOpenMAX IL 1.0
  - Bellagio OpenMAX ILは1.1

  ⇒OpenCOREが使っている部分に差異がないので、定義を変更
- OpenMAX ILコンポーネントの追加
  - PV OMXILラッパを実装する必要あり
  - BellagioとOpenCORE OMXILの差異

  ⇒dlopen()によるコンポーネントコードの差し替え

  ⇒デストラクタは省略（プロセスまるごとkill）
    - Zygoteが自動的に再起動

注）正しい実装ではありません。
真似しないでください。

OpenMAX ILコンポーネントの変更

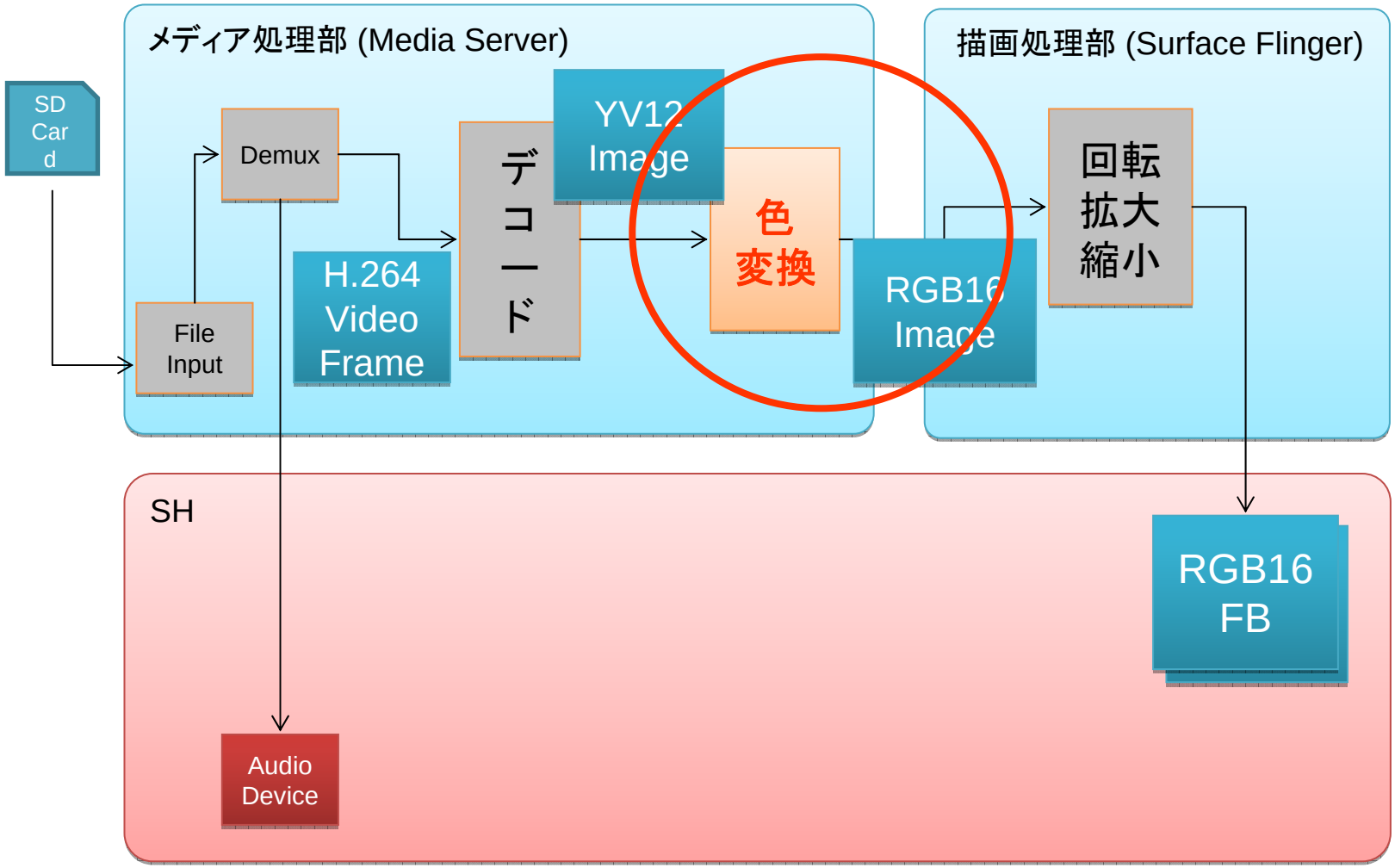

```
@@ -42,7 +86,80 @@ OSCL_EXPORT_REF OMX_ERRORTYPE
AvcOmxComponentFactory(OMX_OUT
OMX_HANDLETYPE* pHa
    OpenmaxAvcAO* pOpenmaxAOType;
    OMX_ERRORTYPE Status;

+       int (*q)(stLoaderComponentType **);
+       p = dlopen ("/system/lib/bellagio/libomxshvpudec.so", RTLD_NOW);
+       q = (int(*)(stLoaderComponentType **))
+           dlsym (p, "omx_component_library_Setup");
+       n = q (NULL);
+           stLoaderComponentType d[n], *dd[n];
+       int i;
+       unsigned int j;
+       OMX_ERRORTYPE e;
+       OMX_COMPONENTTYPE *c;
```

OpenMAX ILコンポーネントの変更


```
+       for (i = 0; i < n; i++)
+           dd[i] = &d[i];
+       q (dd);
+       c = new OMX_COMPONENTTYPE;

…

+       e = d[i].constructor (c, (OMX_STRING)
+                           "OMX.st.video_decoder.avc.shvpu");
+      *pHandle = (OMX_HANDLETYPE *)c;

         pOpenmaxAOType = (OpenmaxAvcAO*) OSCL_NEW(OpenmaxAvcAO, ());
```

注）正しい実装ではありません。真似しないでください。

2. 色変換のオフロード


メディア処理部 (Media Server)
描画処理部 (Surface Flinger)
SD Card
Demux
File Input
H.264 Video Frame
デコード
YV12 Image
色変換
RGB16 Image
回転拡大縮小
SH
RGB16 FB
Audio Device

OpenCOREがサポートする カラーフォーマット

OpenCOREでは…

- デコード出力はYUV（YV12）のみを想定
  - This is the format of choice for many software MPEG codecs. It comprises an NxM Y plane followed by (N/2)x(M/2) V and U planes.
  - OpenMAX IL仕様では、コンポーネントが出力する色フォーマットを通知できる仕組みがあるが、OpenCOREがシカト
- 画面出力はRGB565を想定
- アクセラレーションH/Wの出力はNV12がメジャー
  - YUV 4:2:0 image with a plane of 8 bit Y samples followed by an interleaved U/V plane containing 8 bit 2x2 subsampled colour difference samples.

OpenCORE内の色変換処理

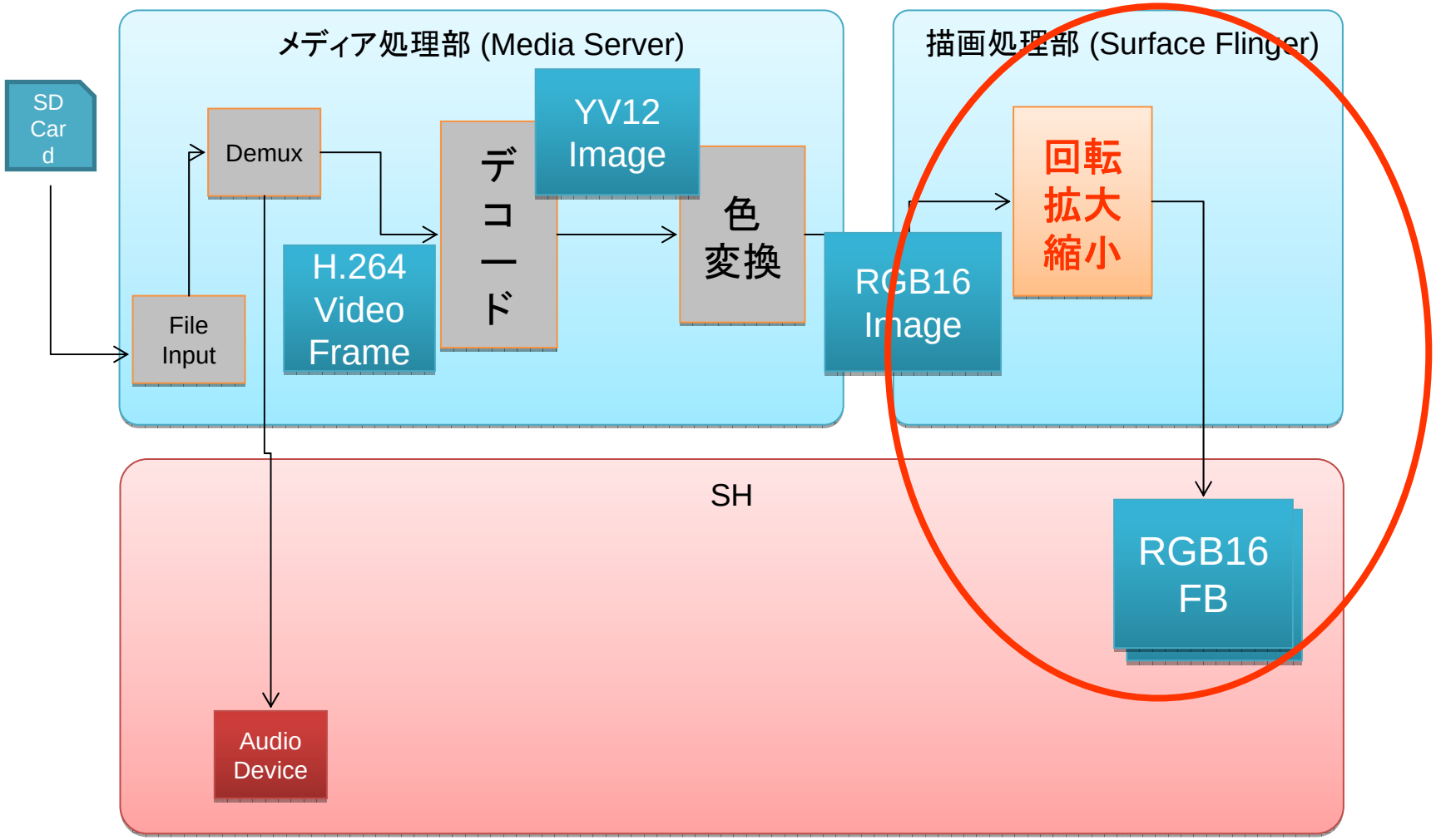
- ビデオ画像の色変換処理はOpenCORE（mediaserver）内で実施
  - external/opencore/codecs_v2/utilities/colorconvert
  - external/opencore/android/android_surface_output.cpp

```
iColorConverter = ColorConvert16::NewL();
    iColorConverter->Init(displayWidth, displayHeight, frameWidth, displayWidth,
displayHeight, displayWidth, CCROTATE_NONE);
    iColorConverter->SetMemHeight(frameHeight);
    iColorConverter->SetMode(1);

…

iColorConverter->Convert(aData, static_cast<uint8*>(mFrameHeap->base()) +
                                mFrameBuffers[mFrameBufferIndex]);
```

色変換処理の差し替え


```
-    iColorConverter->Convert(aData, static_cast<uint8*>(mFrameHeap->base()) +
-                                    mFrameBuffers[mFrameBufferIndex]);
+     shveu_operation(veuIndex,
+                     pSrcY, pSrcC,
+                     displayWidth, displayHeight, frameWidth,
+                     SHVEU_YCbCr420,
+                     pDstRGB, pDstDummy,
+                     displayWidth, displayHeight, displayWidth,
+                     SHVEU_RGB565,
+                     SHVEU_NO_ROT);
+
+     memcpy(static_cast<uint8*>(mFrameHeap->base()) +
+            mFrameBuffers[mFrameBufferIndex],
+            vDstRGB, frameSize);
```

3. 回転・拡大縮小・クリッピングのオフロード



SurfaceFlingerの処理

画面サイズ＆向きにあわせるための回転・拡大縮小・クリッピングを実施

- SurfaceFlingerに渡るまでは一定サイズ（ビデオは、元ビデオサイズ、カメラは320x240のような固定）で描画

⇒色変換と同様にビデオ画像処理機能（VEU）へオフロード

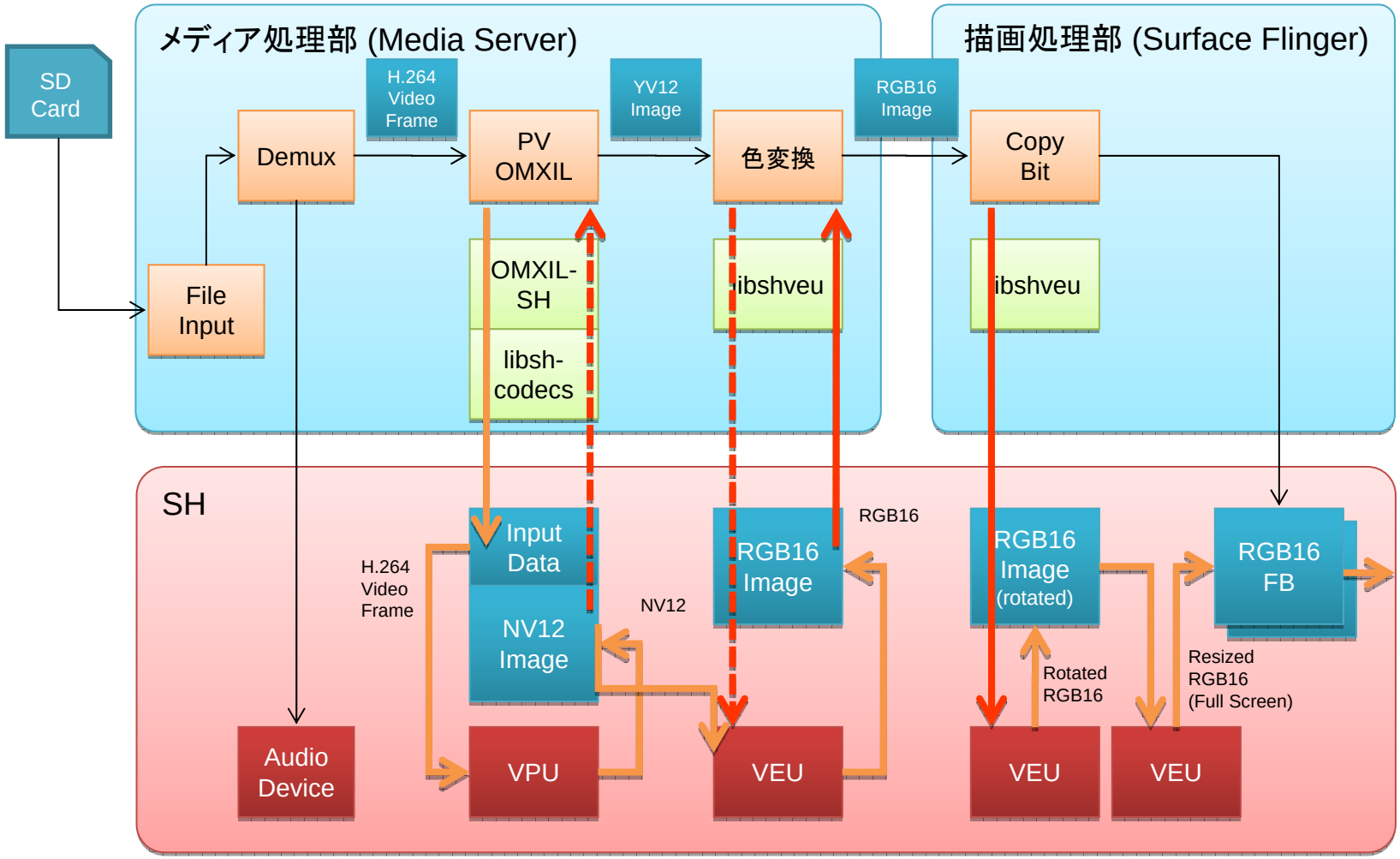
SurfaceFlinger描画プラグイン 仕様：copybit

アクセラレーションH/Wへオフロードするためのプラグイン

- 起動時に配置されていると自動ロード＆リンク
  - /system/lib/hw/copybit.*boardname*.so
- hardware/libhardware/include/hardware/copybit.h

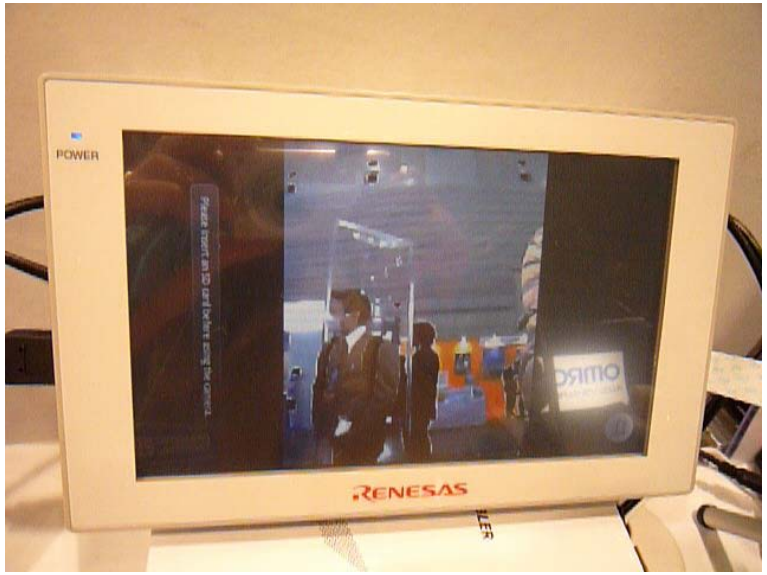
```
struct copybit_device_t {
    struct hw_device_t common;
int (*set_parameter)(struct copybit_device_t *dev, int name, int value);
int (*get)(struct copybit_device_t *dev, int name);
int (*blit)(struct copybit_device_t *dev, struct copybit_image_t const *dst,
                struct copybit_image_t const *src, struct copybit_region_t const *region);
int (*stretch)(struct copybit_device_t *dev, struct copybit_image_t const *dst,
                    struct copybit_image_t const *src, struct copybit_rect_t const *dst_rect,
                    struct copybit_rect_t const *src_rect,
                    struct copybit_region_t const *region);
};
```

改善した処理フロー


メディア処理部 (Media Server)
描画処理部 (Surface Flinger)
SD Card
File Input
Demux
H.264 Video Frame
PV OMXIL
YV12 Image
色変換
RGB16 Image
Copy Bit
OMXIL-SH
libsh-codecs
libshveu
libshveu
SH
H.264 Video Frame
Input Data
NV12 Image
NV12
RGB16 Image
RGB16
RGB16 Image (rotated)
RGB16 FB
Rotated RGB16
Resized RGB16 (Full Screen)
Audio Device
VPU
VEU
VEU
VEU

最適化の効果

- より大きなサイズのビデオでも再生がなめらかに
- copybitによりカメラプレビューも改善



適用前

copybit 適用後

Android on SHに必要なコードは ここからダウンロードできます。

- Android patches for Renesas SH
  https://review.source.android.com/#dashboard,1001893
- libshcodecs
  http://github.com/kfish/libshcodecs
- omxil-sh
  http://github.com/kfish/omxil-sh
- libshveu
  http://github.com/kfish/libshveu