Chapter 1

在 SDK 中启用 Gstreamer

欲使用本公司之 media framework，请于 buildroot 当中开启 Gstreamer 相关包：

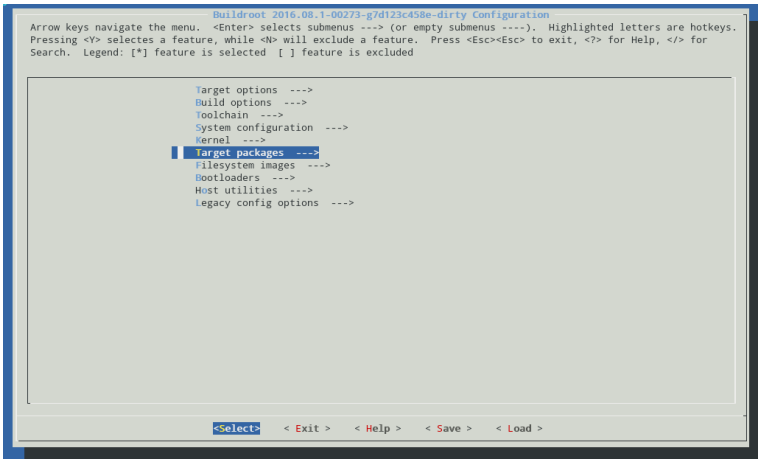

Figure 1.1: User space 包选单

Gstreamer 属于 Audio and Video application，所以请进入下面的选单:

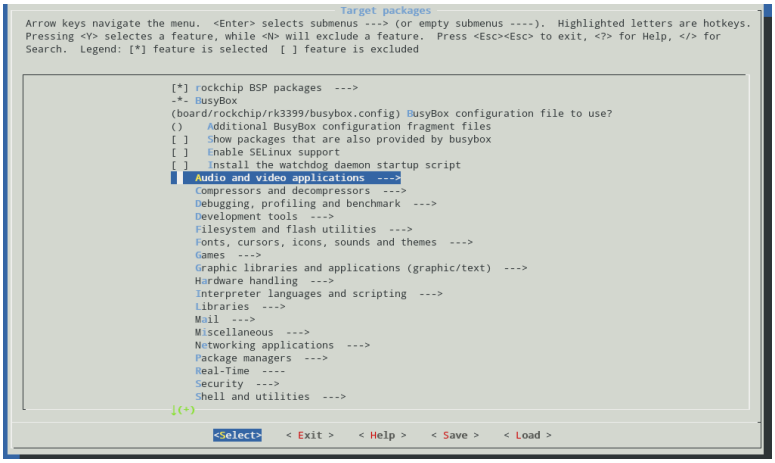

Figure 1.2: 大部分 audio 与 video 的函式库所在的选单

按照默认的配置，需要的包已经大数被选中，如果不需要音讯的解码，可以不要选 gst1-libav 这个包，这个包是依赖 ffmpeg 来进行解码的。我们是使用 Gstreamer 1.x 世代的，可以不用去管 Gstreamer 0.10 的包，Gstreamer 官方目前已经基本上没有在维护。

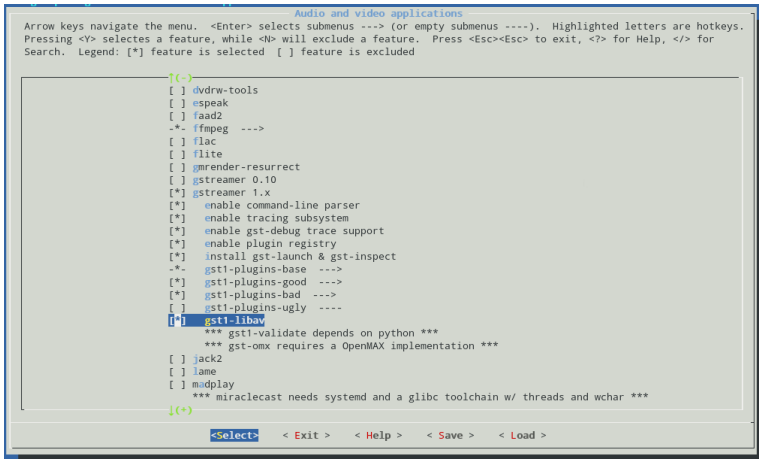

Figure 1.3: 官方的 Gstreamer 包

您需要何种 Gstreamer plugins，或者任何的输入输出组建，可以上到下面的位置查询目前已经被支援的项目：

1. Gstreamer plugins base
2. Gstreamer plugins good
3. Gstreamer plugins bad

Rockchip 平台的上的 Video encoder 与 Video decoder 的支援，全部放置于 BSP packages 之中，返回图1.2所示状态，移动选单，参考图1.4的位置进入

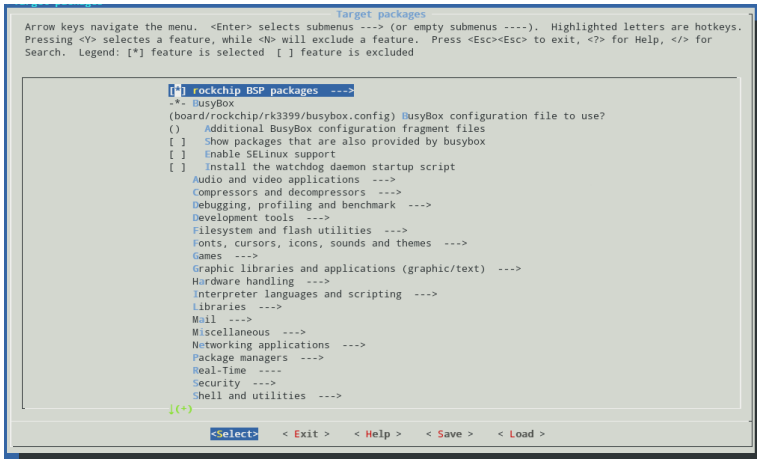

Figure 1.4: Rockchip BSP 包

按照下图选中相关的包，默认配置下应该是已经选中了。

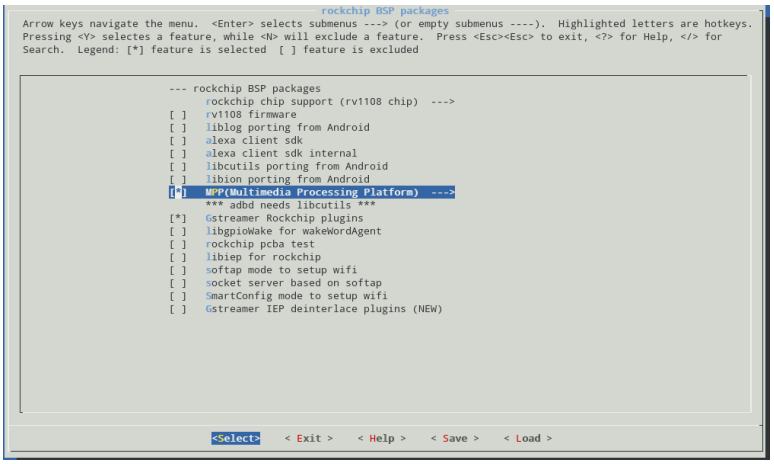

Figure 1.5: Gstreamer rockchip 和 Rockchip MPP

Chapter 2

Gstreamer 应用程式介面

Gstreamer 一般推荐自我包裹的方式工作，因为在 Gstreamer 中比较好处理影音同步，QoS 和缓冲等问题，但是如果您期望直接访问 Gstreamer 中的资料，可以参考下面两个文档：

1. Gstreamer 用例手册
2. Gstreamer Appsink 手册

在Gstreamer rockchip的包中，Rockchip 亦提供了相关示例给您参考。

2.1 示例解说

在示例中，创建了一个 pipeline 模板，并使用了 decodebin，根据在 Gstreamer rockchip 中的权重设定，会优先使用 Rockchip 上的解码器。

```
226   pipeline =
          "filesrc name=\"src\" ! decodebin name=\"decode\" ! video/x-raw ! appsink sync=
        false name=\"sink\"";
228   dec->pipeline = gst_parse_launch (pipeline, NULL);
```

gst-decoder-app.c

这边指定上面 pipeline 中的来源文件位置，如果使用 rtsp 等来源，可以更换上面所使用的 plugins，并指定 uri。

```
242   src = gst_bin_get_by_name (GST_BIN (dec->pipeline), "src");
      g_object_set (G_OBJECT (src), "location", filename, NULL);
244   gst_object_unref (src);
```

gst-decoder-app.c

这边是在示例中，用来处理输出资料的 thread，并且从 pipeline 中的 appsink 这个 plugin 中获得影片资料。

```
346 video_frame_loop (void *arg)
    {
      struct decoder *dec = arg;
348
      do {
350     GstSample *samp;
        GstBuffer *buf;
352
```

```
    samp = gst_app_sink_pull_sample (GST_APP_SINK (dec->sink));
```

gst-decoder-app.c

下面的部分示范了获得输出影片的解析度等资料

```
    pixfmt = GST_VIDEO_INFO_FORMAT (&(dec->info));
324 pixfmt_str = gst_video_format_to_string (pixfmt);

326 printf ("====================================\n");
    printf ("GStreamer video stream information:\n");
328 printf ("  size: %u x %u pixel\n", width, height);
    printf ("  pixel format: %s  number of planes: %u\n", pixfmt_str, nplanes);
330 printf ("  can use zero-copy: %s\n", yesno (is_dmabuf_mem));
    printf ("  video meta found: %s\n", yesno (meta != NULL));
332 printf ("====================================\n");
```

gst-decoder-app.c

在示例当中我们是把当前的画面资料直接写入到储存器当中，受限到储存器速度的限制，可能速度会比较慢，可以考虑采用不同的方法在处理。

```
  g_snprintf (filename, sizeof (filename), "img%05d.%s", dec->frame,
336     pixfmt_str);
  g_file_set_contents (filename, map_info.data, map_info.size, NULL);
```

gst-decoder-app.c