Video Streaming on the i.MX27 processor
Fabio Estevam
Applications Engineer – Multimedia Applications Division
Agenda

- i.MX27 Processor Overview
- i.MX27 Product Development Kit (PDK) features
- i.MX27 Processor Linux® BSP Setup
- Introduction to Gstreamer
- Basic Gstreamer Examples
- Streaming Video Examples

(Support material for this session is available under ‘Video_Streaming_on MX27’ link)
i.MX27 Processor Overview
i.MX27 Processor Overview

- Specifications:
  - CPU: ARM926EJ-S, 400MHz
  - Process: 90nm
  - Core Voltage: 1.2-1.5V
- Key i.MX27 Features and Advantages
  - 16 KB L1 I-Cache and D-Cache
  - 16-channel DMA
  - Multi-standard video codecs at D1 resolution (i.MX27 only)
    • Video pre- and post-processing, scaling
  - Security
    • Crypto Accelerator
    • Security controller with encrypted RAM storage
    • Electronically blown fuse box
    • High-assurance boot
  - Real-time OS/SW integrity checker
  - Dynamic Process temperature Compensation (DPTC)
  - Connectivity
    • Ethernet 802.3 MAC
    • USB 2.0 OTG 480Mbps
    • USB 2.0 Host 12Mbps, USB 2.0 Host 480Mbps
    • PCMCIA/CF, Audio MUX
    • MMC, SD, IrDA, 8x8 keypad, CMOS sensor interface
    • ATA-6, Memory Stick (i.MX27 only)
- Availability:
- Package types
  - 404-ball, 17x17, MAPBGA, 0.65mm pitch
  - 473 ball, 19 x 19, MAPBGA, 0.8mm pitch
i.MX27 Processor VPU Features

► Multi-standard video codec
  • MPEG-4 part-II simple profile encoding and decoding
  • H.264/AVC baseline profile encoding and decoding
  • H.263 P3 encoding and decoding
  • Multi-format/multi-instance operation is supported.
  • i.MX27 can decode an MPEG-4 bitstream and a H.264 bitstream simultaneously.
  • i.MX27 can encode an MPEG-4 bitstream and decode a H.264 bitstream simultaneously.
i.MX27 Processor VPU Features

► Bit-rate control (CBR and VBR)
► Pre/post rotation/mirroring
  • 8 rotation/mirroring modes for image to be encoded
  • 8 rotation/mirroring modes for image to be displayed
► Performance
  • All video standard decoders support up to 720x480 @ 30 fps and 720x576 @ 25 fps.
  • All video standard encoders support up to 720x480 @ 30 fps and 720x576 @ 25 fps.
  • i.MX27 supports full duplex operations up to VGA (640x480) @ 30 fps.
  • i.MX27 supports half duplex operations up to 720x480 @ 30 fps and 720x576 @ 25 fps.
i.MX27 PDK Features
i.MX27 PDK Overview

**Personality Module**
Provides the devices most commonly tailored to meet a specific target product or customer requirement:

- VGA Touch-screen Display
- Buttons
- Connectivity
- User I/O
- Communications
- Camera
- Storage
- External connectors

**i.MX27 Processor Module**
Can be attached to:
- De-bug module for software development
- Personality module for demonstration
- Both De-bug and Personality

**De-bug Module**
Provides Functions required for hardware and software development, but would not reside on a final product:

- Debug Ethernet
- Debug Serial Port
- JTAG
- Reset, Interrupt, boot switches
- Debug LEDs
- CodeTest interface
- Power Source
- Current/Power monitoring
i.MX27 Processor Linux BSP Setup
i.MX27 Processor Linux BSP Download

► Downloading the BSP

  • [http://www.freescale.com/imx27pdk](http://www.freescale.com/imx27pdk), Click on Downloads tab and then download LDK_IMX27_R1_1 file.

► Downloading the documentation

  • [http://www.freescale.com/imx27pdk](http://www.freescale.com/imx27pdk), Click on Documentation tab and then download PDK_IMX27_LINUXDOCS_BUNDLE file.

► The documentation package includes some useful documents, such as:
  • Release Notes, Linux Reference Manual, Linux Quick Start, etc.
Some BSP upgrades

► Apply OV2640 patch: This patch corrects camera clock allowing us to achieve the correct framerate

► Copy ov2640-performance.patch into /opt/freescale/pkgs

Edit ltib/dist/lfs-5.1/kernel/kernel-2.6.22-pdk27.spec.in, adding the following lines into the appropriate places:
Patch134: ov2640-performance.patch
And also
%patch 134 –p1

► Rebuild the kernel

./ltib –p kernel –m prep
./ltib –p kernel –m scbuild
./ltib –p kernel –m scdeploy
Some BSP upgrades

Upgrade the Gstreamer packages:

• In order to have better video stream quality, we recommend that the following Gstreamer packages are upgraded:

  - gst-plugins-base.spec (version 0.10.22)
  - gst-plugins-good.spec (version 0.10.14)
  - gst-plugins-core.spec (version 0.10.22)
  - liboil.spec (version 0.3.16)

• Simply replace the Gstreamer related spec files above in ltib/dist/lfs-5.1/gst*, copy the spec files into /opt/freescale/pkgs and rebuild them.
VPU Library Update

 ► Upgrade the VPU library to 2.2.4

 • Latest VPU code can be downloaded from: http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=IMX27_CODECS&fpsp=1&tab=Design_Tools_Tab

 • Click on MX27_VPU_FW_2.2.4_UPDATE_TO2.X_ONLY_LINUX, download and install the VPU update

 • There is a very useful document about the VPU API inside this package that explains the i.MX27 VPU programming model and also the programmable parameters.
Gstreamer Plugins Update

Upgraded the i.MX27 Gstreamer updates

- Latest i.MX27 Gstreamer package can be downloaded from:  

- Click on MX27_FULL_VPU_SW_LINUX_TO2.X_ONLY, download and install i.MX27 Gstreamer package.

- This package contains Gstreamer plugins that utilize the hardware accelerated block (VPU) on i.MX27

Available plugins: MPEG4 and H264 encoder, decoder, video source, video sink, MPEG4 and H264 parsers
Configuring the Bootloader

► Set debug board switches: SW9 UP – all others DOWN (This will make MX27 to boot from NAND)

► Power up the board and check the console output in minicom (115200 bps, no flow control)

RedBoot™ bootstrap and debug environment [ROMRAM]
Non-certified release, version FSL 200840 - built 15:01:21, Dec 11 2008

Platform: i.MX27 3STACK (Freescale i.MX27 based) PASS 2.1 [x32 SDR]

RAM: 0x00000000-0x07f00000, [0x00025220-0x07ed1000] available
FLASH: 0x00000000 - 0x10000000, 2048 blocks of 0x00020000 bytes each.
== Executing boot script in 1.000 seconds - enter ^C to abort
^C
RedBoot> ^C
RedBoot>
Configuring the Bootloader (continued)

► Make sure the Redboot version is 200840

► If Redboot is not flashed in the board, please use ATK tool to flash it. There is a pre-built image of the bootloader in the BSP package.

► ATK tool can be downloaded from:
  http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=i.MX27PDK&fpsp=1&tab=Design_Tools_Tab

(Click on IMX27_ATK_TOOLKIT_R160 )
Configuring the Bootloader (continued)

► We will boot the kernel via TFTP and the root file system via NFS

► Internal i.MX27 FEC will be used for streaming

RedBoot> fc -l
Run script at boot: true
Boot script:
.. load -r -b 0x100000 zImage
.. exec -b 0x100000 -l 0x200000 -c "noinitrd console=ttymxc0,115200 root=/dev/nfs
    nfsroot=10.29.240.185:/tftpboot/ltib init=/linuxrc ip=dhcp"

Boot script timeout (1000ms resolution): 1
Use BOOTP for network configuration: true
Default server IP address: 10.29.240.185
Board specifics: 0
Console baud rate: 115200
Set eth0 network hardware address [MAC]: false
Set FEC network hardware address [MAC]: false
GDB connection port: 9000
Force console for special debug messages: false
Network debug at boot time: false
Default network device: mxc_fec
RedBoot>
Booting Linux

► The kernel image (zImage) is copied into /tftpboot directory

► Root file system is exported through NFS

► We can boot Linux now! Just reboot the board and the boot process should start.

► After the whole boot process the Linux prompt shows up:

mx27#
Testing the BSP Installation

► Check if the required plugins were correctly installed

```
mx27# gst-inspect | grep mfw
mfw_vpudecoder:  mfw_vpudecoder: Freescale: Hardware (VPU) Decoder
mfw_vpuencoder:  mfw_vpuencoder: Freescale: Hardware (VPU) Encoder
mfw_v4lsrc:  mfw_v4lsrc: Freescale Video Source plug-in
mfw_v4lsink:  mfw_v4lsink: Freescale: v4l_sink
mfw_avidemuxer:  mfw_avidemuxer: FSL Avi Demuxer
mfw_mp4demuxer:  mfw_mp4demuxer: freescale-mp4 demuxer plugin
```
Introduction to Gstreamer
GStreamer Overview

GStreamer is a framework for creating streaming media applications.
Gstreamer Terminology

Elements

- An element is the most important class of objects in GStreamer. You will usually create a chain of elements linked together and let data flow through this chain of elements. An element has one specific function, which can be the reading of data from a file, decoding of this data or outputting this data to your sound card (or anything else).

- By chaining together several such elements, you create a pipeline that can do a specific task, for example media playback or capture.

- GStreamer ships with a large collection of elements by default, making the development of a large variety of media applications possible.
Gstreamer Terminology

► Pads
  • Pads are element's input and output, where you can connect other elements. They are used to negotiate links and data flow between elements in GStreamer.

► Bins and Pipelines
  • A bin is a container for a collection of elements. A pipeline is a special subtype of a bin that allows execution of all of its contained child elements. Since bins are subclasses of elements themselves, you can mostly control a bin as if it were an element, thereby abstracting away a lot of complexity for your application.
Example of a Gstreamer pipeline
GStreamer Overview (continued)

GStreamer is packaged into:

- gstreamer: the core package
- gst-plugins-base: an essential exemplary set of elements
- gst-plugins-good: a set of good-quality plug-ins under LGPL
- gst-plugins-ugly: a set of good-quality plug-ins that might pose distribution problems
- gst-plugins-bad: a set of plug-ins that need more quality
- gst-python: the python bindings
- a few others packages
Basic Gstreamer Examples
Gstreamer Examples (Lab1)

► Camera Loopback

- OV2640 is the camera on i.MX27 PDK
- mx27# gst-launch mfw_v4lsrc ! mfw_v4lsink

► This simple pipeline captures the camera image using the mfw_v4lsrc source element and displays it on the LCD screen through mfw_v4lsink sink plugin.
Gstreamer Examples (Lab2)

► Camera Loopback Full Screen

• We can have fullscreen image by adjusting the mfw_v4lsink properties:

```sh
mx27# gst-launch mfw_v4lsrc capture-width=640 capture-height=480 ! mfw_v4lsink fullscreen-width=480 fullscreen-height=640 fullscreen=true
```
Gstreamer Examples (Lab3)

► Video Playback

• Download the MX27_TEST_STREAMS package from the following URL:

• Extract the package and copy the Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 file from MPEG_VPU directory into the rootfs/home MX27 directory.

• To play this media file on your PC:

```
gst-launch filesrc location=Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! decodebin ! autovideosink
```
Gstreamer Examples (Lab4)

- Video Playback
  - To play this media file on the i.MX27 PDK:
    ```
gst-launch filesrc
location=/home/Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! decodebin ! autovideosink
(automatically detects the plugins via decodebin)
```
  - Same pipeline as used on the Linux PC!

- Or also
  ```
gst-launch filesrc
location=/home/Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! mfw_mp4demuxer ! mfw_vpudecoder codec-type=std_mpeg4 ! mfw_v4lsink
```
Gstreamer Examples (Lab5)

► MP3 Playback

- For MP3 playback the standard “mad” plugin can be used (not optimized):

  ```bash
  gst-launch filesrc location=/home/audio1.mp3 ! mad ! alsasink
  ```

- Connect a stereo headphone into J19 jack to hear the music track.
Gstreamer Examples (Lab6)

► Video and Audio Playback

gst-launch filesrc
location=/home/Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! mfw_mp4demuxer name=demux demux. ! queue ! mfw_vpudecoder codec-type=std_mpeg4 ! mfw_v4lsink demux. ! queue ! mad ! alsasink
Video Streaming Examples
Video Streaming Examples (Lab7)

► Streaming an H.264 camera stream from the MX27 to the Host PC running Gstreamer

• On i.MX27 PDK:
  ```bash
  export HOST=192.168.1.1
  mx27# gst-launch-0.10 -v mfw_v4lsrc capture-width=640 capture-height=480 ! mfw_vpuencoder width=640 height=480 codec-type=std_avc ! rtph264pay ! udpsink host=$HOST port=5000
  ```

• On Linux PC:
  ```bash
  gst-launch-0.10 -v udpsrc port=5000 caps ="application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, profile-level-id=(string)42001e, sprop-parameter-sets=(string)Z0IAHqaAoD2Q, payload=(int)96, ssrc=(guint)3296222373, clock-base=(guint)2921390826, seqnum-base=(guint)35161" ! rtph264depay ! ffdec_h264 ! autovideosink
  ```
Video Streaming Examples

► In the previous example (H.264 camera stream from the i.MX27 to the Host PC running Gstreamer), we can see that the CPU usage is only about 4% (as shown by the ‘top’ command)

► Thanks to the i.MX27 VPU that handles the video encoding, the CPU is almost entirely freed to perform other tasks.
Video Streaming Examples

► How to get the caps parameter?

► When launching the pipeline in the i.MX27PDK with –v option you get the following output

► ...

/GstPipeline:pipeline0/GstUDPSink:udpsink0.GstPad:sink: caps = application/x-rtp, media=(string)video, clock-rate=(int)90000, encoding-name=(string)H264, profile-level-id=(string)42001e, sprop-parameter-sets=(string)Z0lAHqaAoD2Q, payload=(int)96, ssrc=(guint)3296222373, clock-base=(guint)2921390826, seqnum-base=(guint)35161

► The caps should be used in the receiver (PC) side
Video Streaming Examples (Lab8)

Streaming an H.264 camera stream from the i.MX27 to the Host PC running VLC

gst-launch-0.10 -v mfw_v4lsrc capture-width=640 capture-height=480 !
mfw_vpuencoder width=640 height=480 codec-type=std_avc ! rtph264pay !
udpsink host=$HOST port=5000

• Save the following text into a file called
  • H264.sdp

• Right click on H264.sdp and select:
  • “Open with VLC”

• Host PC should receive the camera image in VLC application

v=0
o=- 1223457093460663 1 IN IP4 127.0.0.1
s=RTSP Server
i=Codec00
t=0 0
a=tool:LIVE555 Streaming Media v2008.07.24
a=type broadcast
b=AS:21
a=control:*
a=source-filter: incl IN IP4 127.0.0.1
a=rtcp-unicast: reflection
m=video 5000 RTP/AVP 96
a=rtpmap:96 H264/90000
a=framerate=30
a=fmtp:96 profile-level-id=42001e; sprop-parameter-sets=Z0IAHqaAoD2Q;
Video Streaming Examples (Lab9)

► Streaming an MPEG4 camera stream from the MX27 to the Host PC running VLC

```bash
gst-launch-0.10 -v mfw_v4lsrc blocksize=460800 capture-width=640 capture-height=480 ! mfw_vpuencoder width=640 height=480 codec-type=std_mpeg4 ! rtpmp4vpay send-config=true ! udpsink host=$HOST port=5000
```

- Save the following text into a file called `MPEG4.sdp`
- Right click on MPEG4.sdp and select: “Open with VLC”

► Host PC should receive the camera image in VLC application

```plaintext
v=0
o=- 1223457093460663 1 IN IP4 127.0.0.1
s=RTSP Streami=Codec00
t=0 0
a=tool:LIVE555 Streaming Media v2008.07.24 a=type broadcast
b=AS:21
a=control:*
a=source-filter: incl IN IP4 127.0.0.1
a=rtcp-unicast: reflection
m=video 5000 RTP/AVP 96
a=rtpmap:96 MP4V-ES/90000
a=framerate=30
a=fmtp:96 profile-level-id=4;
config=000001b004000001b59113000001000000012000c8880
0f514043c14103;
```
Video Streaming Examples (Lab11)

Streaming an MPEG4 file from the MX27 to the Host PC running Gstreamer

- On the i.MX27 PDK:
  ```
gst-launch-0.10 -v filesrc location=Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! qtdemux ! rtpmp4vpay send-config=true ! udpsink host=$HOST port=5000
  ```

- On the Linux PC:
  ```
gst-launch-0.10 -v --gst-debug=2 udpsrc port=5000 caps=<caps>! rtpmp4vdepay ! ffdec_mpeg4 ! autovideosink
  ```
Video Streaming Examples (Lab12)

Streaming an MPEG4 file from the MX27 to the Host PC running VLC

gst-launch-0.10 -v filesrc location=
Kaleidoscope_mp4v_mp3_320x240_30fps_1200kbps_a_48khz_128kbps.mp4 ! qtdemux ! rtpmp4vpay send-config=true ! udpsink host=$HOST port=5000

• Save the following text into a file called MPEG4_file.sdp
• Right click on MPEG4_file.sdp and select: “Open with VLC”
• Host PC should receive the camera image in VLC application

v=0
o=- 1223457093460663 1 IN IP4 10.29.240.185
s=RTSP Server
i=Codec00
t=0 0
a=tool:LIVE555 Streaming Media v2008.07.24
a=type: broadcast
b=AS:21
a=control:*
a=source-filter: incl IN IP4 10.29.240.190
a=rtcp-unicast: reflection
m=video 5000 RTP/AVP 96
a=rtpmap:96 MP4V-ES/90000
a=framerate=30
a=fmtp:96 profile-level-id=1;
config=000001b001000001b58913000001000000012000c48d880f50a041
e1463000001b24c61766335312e34392e30;
References

► Application note: Video Streaming on i.MX27 Processor:

► Gstreamer tutorial:

► Gstreamer mailing list:
   gstreamer-devel mailing list
   gstreamer-devel@lists.sourceforge.net
   https://lists.sourceforge.net/lists/listinfo/gstreamer-devel
Thank you for attending this presentation. We’ll now take a few moments to review the audience questions, and then we’ll begin the question and answer session.