Building GStreamer

Prateek Bansal, Brian Jeff, Isara Indra

ABSTRACT

GStreamer is a pipeline-based multimedia framework that allows you to create a variety of media-handling components, including simple audio playback, audio and video playback, recording, streaming, and editing. The pipeline design allows you to easily write any type of streaming multimedia application.

This software release helps you leverage the DV-EVM software infrastructure (Codec Engine, MV Linux LSP, DSP Codecs, etc.) in developing open source GStreamer multimedia framework to DaVinci class devices. These open-source frameworks increase productivity as they can capably handle multiple tasks such as AV synchronization, demuxing/muxing, network streaming, etc.

This application report demonstrates how to build a simple media playback application using GStreamer and the DV-EVM software foundation. The GStreamer multimedia framework is first cross-compiled and adapted to run on the ARM926EJ-S core of the DaVinci DM6446 device. The codec engine includes various audio/video decoder modules that are then adapted into pluggable decoder components that can be mixed and matched into arbitrary pipelines. Finally, you will use these components to create various types of audio/video playback applications.

Contents

List of Figures

1 System Requirements

First, you must install the GStreamer components and plug-ins on the existing DV-EVM setup. The software package associated with this application report includes all the components needed for this process. You must have the following setup to use this software:

- Host PC running Red Hat Enterprise Linux 4
- Host PC has DV-EVM 1.20.00

This setup also allows you to access to the MontaVista cross compiler tools set. The build procedures described below have been tested with the DV-EVM v1.20 software tools. The DV-EVM v1.20 software can be downloaded following registration of your DM6446 DV-EVM at http://www.ti.com/dvevmupdates. The compiler tools from the DV-EVM installation are necessary to cross-compile the GStreamer libraries and the plug-ins for the DaVinci platform.

The DV-EVM software foundation and GStreamer software package are designed to enable the GStreamer multimedia framework on TI's DaVinci platform (DV-EVM). The GStreamer framework runs from the application level that utilizes a VISA API/codec engine as a low-level multimedia engine (see Figure 1).

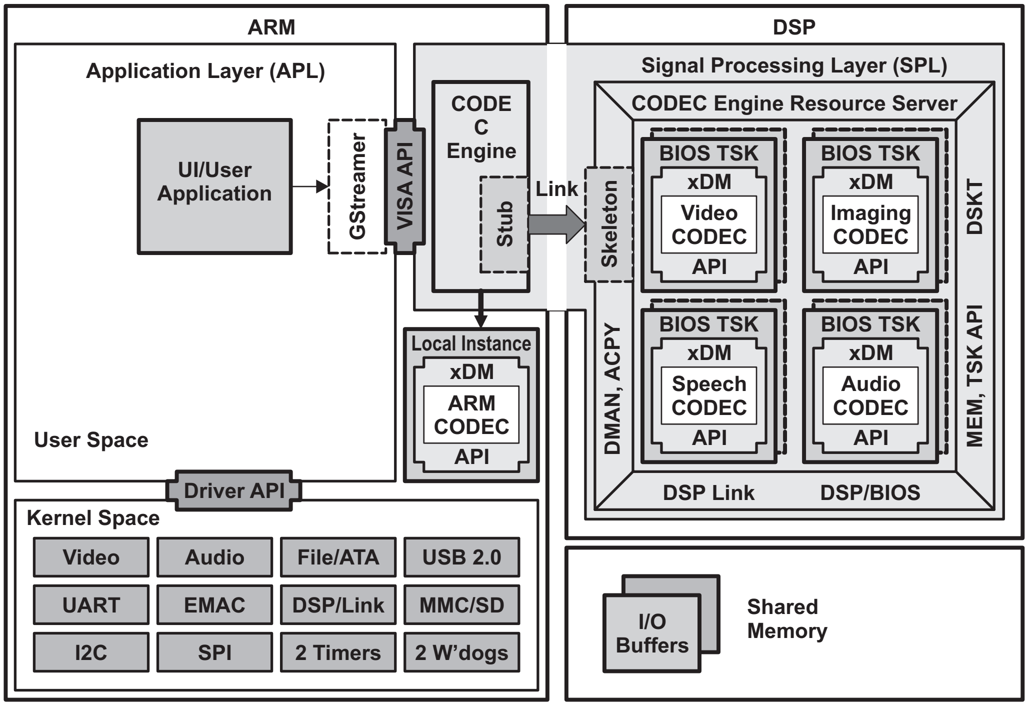

Figure 1. GStreamer/DaVinci DV-EVM Software Architecture

2 GStreamer Components

GStreamer is a media-processing library that provides an abstract model of a transformation that is based on a pipeline. Media flows in a defined direction from input to output. GStreamer has gained wide popularity in the digital video programming community through its ability to abstract the manipulation of different media; thus, simplifying the programming process. GStreamer makes it possible to write a general video or music player that can support many different formats and networks. Most operations are performed by plug-ins, rather than by the GStreamer core. Figure 2 displays this media pipeline concept:

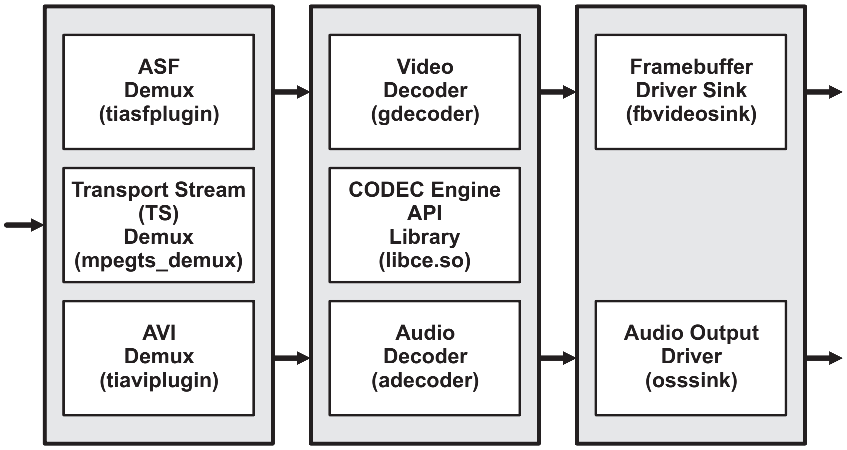
ASF Demux (tiasfplugin)
Transport Stream (TS) Demux (mpegts_demux)
AVI Demux (tiaviplugin)
Video Decoder (gdecoder)
CODEC Engine API Library (libce.so)
Audio Decoder (adecoder)
Framebuffer Driver Sink (fbvideosink)
Audio Output Driver (osssink)

Figure 2. GStreamer Multimedia Pipeline on DaVinci Platform

The following sections list the libraries included in the GStreamer software package.

2.1 GStreamer Core Libraries

The source files for these packages can be found on the corresponding open source community site.

- OS: glib-2.12.4 (Source package: http://ftp.gnome.org/pub/GNOME/sources/glib/2.12/)

  GLib is a general-purpose utility library, providing many data types, macros, type conversions, string utilities, file utilities, a main loop abstraction, etc. It works on many UNIX-like platforms, Windows, OS/2 and BeOS. GLib is released under the GNU Library General Public License (GNU LGPL).
- OS: check-0.9.3 (Source package: http://sourceforge.net/project/showfiles.php?group_id=28255&package_id=20116)

  Check is a unit test framework for C. It features a simple interface for defining unit tests. Tests are run in a separate address space, so Check can catch both assertion failures and code errors that cause segmentation faults or other signals. The output from unit tests can be used within source code editors and IDEs.
- OS: liboil-0.3.9 (Source package: http://liboil.freedesktop.org/download/)

  Liboil is a library of simple functions that are optimized for various CPUs. These functions are generally loops implementing simple algorithms, such as converting an array of N integers to floating-point numbers or multiplying and summing an array of N numbers. Such functions are candidates for significant optimization using various techniques, especially using the extended instructions provided by modern CPUs (Altivec, MMX, SSE, etc.).
- OS: libxml2-2.6.16 (Source package: http://ftp.gnome.org/pub/GNOME/sources/libxml2/2.6/

  XML parser library from the GNOME project.
- OS: gstreamer-0.10.1 (Source package: http://gstreamer.freedesktop.org/src/)

  GStreamer is a library that allows the construction of graphs of media-handling components, ranging from simple Ogg/Vorbis playback to complex audio (mixing) and video (non-linear editing) processing. It features basic functionality and libraries, some essential elements, documentation, and testing.

2.2 GStreamer Plug-In Libraries

GStreamer plug-in libraries in this package include both Open Source plug-ins and TI-developed plug-ins. While the Open Source GStreamer plug-ins provide the basic framework for a multimedia system (sound driver, file parser), the TI-developed GStreamer plug-ins leverage the DSP for video decoding and run on an ARM device under the Linux operating system. TI also provides Linux peripheral drivers, which conform to standard open-source mechanisms for the driver interface, as well as a codec engine API that abstracts many of the complexities of programming DSPs. Open Source packages can also be found at http://gstreamer.freedesktop.org/src/. The following lists all the plug-ins included in the GStreamer software package:

- OS: gst-plugins-base-0.10.11

  This package is part of the base plug-ins for GStreamer. It provides a set of required plug-ins and elements.
- OS: mpegts_demux

  This plug-in is for Transport Stream demultiplexing into individual audio and video streams. This code is based on Fluendo MPEG TS demux code.
- OS: OSS (open sound system) audio sink driver

  This is an open-source plug-in to send audio or sound to a sound card based on the OSS (Open Sound System) digital audio architecture for Linux.
- TI: fbvideosink (Framebuffer video sink)

  This is a TI-developed plug-in available as source code to display video using the Video Processing Back End subsystem on DaVinci DM644x devices. This is based on the framebuffer digital video audio architecture for Linux.
- TI: gdecoder (Video Decoder)

  This is a TI-developed plug-in available as source code to decode elementary video streams using the Codec Engine software foundation available on DaVinci devices. This software foundation allows for remote procedure calls to DSP to decode the video streams. The video decoder plug-in can support elementary video stream of VC1/WMV9, H.264 MP, MPEG4 SP, and MPEG2 MP@ML decoding.

- TI: adecoder (Audio Decoder)

  This is a TI-developed plug-in available as source code to decode elementary audio streams using the Codec Engine software foundation available on DaVinci devices. This software foundation allows for remote procedure calls to DSP to decode the audio streams. The audio decoder plug-in can support elementary audio streams of AAC, MP3, and WMA decoding.
- TI: tiavidemux (AVI Demuxer)

  This is a TI-developed plug-in available as source code to de-multiplex the AVI container files.
- TI: tiasfdemux (ASF Demuxer)

  This is a TI-developed plug-in available as source code to de-multiplex the ASF container files into individual RCV video and RCA audio streams.

3 Build Instructions

The software package released with this application note has grouped the components described in the previous sections into an Open source package (gstreamer_opensource.tar.gz) and a TI specific package (gstreamer_tibuild.tar.gz). Build-scripts have also been created for both packages, using the following assumptions:

1. You have installed DVEVM/DVSDK v1.20 on the Linux host machine per the EVM startup guide, and the kernel image from DV-EVM v1.20 is flashed on the hardware EVM board.
2. The username for the machine is assumed to be *user*, so the default home folder is /home/user. If you are using a different folder name, you must change the directory names in the Shell Script files (make_opensource.sh and make_tiplugins.sh). The two main paths that need to be changed are MVISTA_INSTALL_DIR and FILESYS_DIR in each shell script. The default paths are specified below.
   # MVISTA_INSTALL_DIR should point to your MontaVista directory installation: export
   MVISTA_INSTALL_DIR=/opt/dvevm/mv_pro_4.0/MontaVista.
   #FILESYS_DIR should point to the NFS shared target root directory: export
   FILESYS_DIR=/home/user/workdir/filesys.
   The above paths need to be changed based on your MontaVista installation directory path and the target root directory that is NFS shared. The above settings assume that the target (DV-EVM) file system is mounted under /home/user/workdir/filesys directory. For make_tiplugins.sh, you also need to set the DVEVM_PATH to your DV-EVM installation. An example of a DV-EVM directory path setting is
   #Location of DVEVM_PATH to point to DV-EVM directory: export
   DVEVM_PATH=/home/user/dvevm_1_20
3. This release has been tested with the MontaVista Pro 4.01 target file system.
4. You have audio and video files to run the demo application in the following container formats: AVI, Transport Stream, MP3, AAC, and WMA.
5. AVI file support has been tested with the video stream as MPEG4 SP compliant and audio stream as MP3 compliant.
6. TS (transport stream) file support has been tested with video stream as H.264 MP compliant and audio stream as MP3 compliant.
7. The ASF file is not currently supported on this release, but support for those files is planned for future releases.

3.1 Build Steps

1. Unzip the Open source package (gstreamer_opensource.tar.gz) and TI specific package (gstreamer_tibuild.tar.gz) under the /home/<user>/workdir/gstreamer directory. This creates ti_build/ and opensource_build/ directories.
2. Add MontaVista cross compiler tools in the PATH environment variable. For example:
   export PATH=$PATH:/opt/MontaVista/pro/devkit/arm/v5t_le/bin/
3. Create the following directories on the target file system:
   - /home/user/workdir/filesys/opt/gstreamer
   - /home/user/workdir/filesys/opt/system_files_gstreamer
4. Create the following soft links if the directories do not already exist:

- ln -s <MONTAVISTA INSTALL PATH> /opt/montavista
  Example: If MontaVista installation directory is in /opt/dvevm/mv_pro_4.0/montavista, the soft link will be ln -s /opt/dvevm/mv_pro_4.0/montavista /opt/montavista
- ln -s /bin/sed /opt/montavista/common/bin/sed

3.2 Building Open Source Plug-Ins

1. Open the opensource_build directory.
2. Run the make_opensource script under the gstreamer/opensource_build directory using the following command:
   ./make_opensource.sh
3. The script will ask if you want to build different packages or not. If you are building the first time, say YES to building all the packages. You can do incremental builds following the first full build.

3.2.1 Building TI Plug-Ins

1. Go to the ti_build directory.
2. Run the make_tiplugins shell script using the following command:
   ./make_tiplugins.sh
3. The script will ask if you want to build different packages or not. If you are building the first time, say YES to building all the packages. You can do incremental builds following the first full build.

4 How to Run the Demo on the Target

The previous build steps were done on the host file system. Now you can run the demo on the target (DV-EVM) file system.

1. Boot the hardware DV-EVM with the Linux kernel image *uImage* from DV-EVM v1.20. A copy of this uImage is also located in following host directory:
   /home/user/workdir/filesys/opt/system_files_gstreamer. Also, it is assumed that you have the file system mounted over NFS in uboot settings.
2. Once you are at the login prompt, type *root* and enter.
3. Go to the /opt/system_files_gstreamer directory. The system_files_gstreamer directory is copied from the ti_build folder. This directory contains uImage (DV-EVM v1.20 Kernel Image), cmemk.ko (CMEMK kernel module), dsplinkk.ko (DSPLINK kernel module), DSP Server Executables (*.x64P files), and Test Scripts to run demos. If you are running a custom kernel, you will need to:
   a. Build the cmemk and dsplinkk modules for your custom kernel if you have not already done so.
   b. Copy the cmemk and dsplinkk modules to the system_fileS_gstreamer directory.
   c. Set u-boot to use the custom uImage, instead of the one shipped by default under the /opt/system_files_gstreamer directory.

   If you have your own DSP Server executables that you will to run, you will need to copy your *.x64P files in the system_files_gstreamer directory. You might also have to change the TI PLUGINS, specifically the gdecoder and adecoder. One of the changes would be to use the appropriate Engine Names, as per your *.x64P files under Engine_open call.
4. Run the ./start_demo.sh file.
5. Copy your media files (TS, AVI, AAC, MP3, WMA) under the /opt/media directory to the DV-EVM file system.
6. There are 6 different GStreamer pipelines that have been currently tested using this package. These pipelines are stored under test_***.sh scripts. Other pipelines can be created using the same procedure.
7. Run the following shell to play an AAC file (audio only)
   ./test_AAC.sh <AAC file_name>
   You will hear a beeping sound after every couple of seconds, as this is evaluation audio codec.
8. Run the following shell to play an MP3 file (audio only)
   ./test_MP3.sh <MP3 file_name>
   You will hear a beeping sound after every couple of seconds, as this is evaluation audio codec.

9. Run the following shell to play a TS file (both audio and video)
   ./test_TS.sh <TS file_name>
   You will hear a beeping sound after every couple of seconds, as this is evaluation audio codec and there will be TI logo displayed on right hand corner, as this is evaluation video codec.
10. Run the following shell to play an AVI file (both audio and video)
   ./test_AVI.sh <TS file_name>
   You will hear a beeping sound after every couple of seconds, as this is evaluation audio codec and there will be TI logo displayed on right hand corner, as this is evaluation video codec.
11. Run the following shell to play a WMA file (audio only)
   ./test_WMA.sh <WMA file_name>
   You will hear a beeping sound after every couple of seconds, as this is evaluation audio codec.
12. To stop the demo, click on Ctrl-C.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright 2008, Texas Instruments Incorporated