Texas Instruments

Application Report
SPRA938A − July 2004

JPEG Network on the DM642 EVM

Video and Imaging Systems

ABSTRACT

This software demonstration combines real-time D1 Joint Photographic Experts Group (JPEG) encoding of images on the DM642 Evaluation Module (EVM) with networking functionality.

The JPEG standard pertains to compression of still images. Performing JPEG at the rate of 30 frames per second in isolation as individual images is known as motion JPEG (MJPEG). This demonstration uses:

- JPEG encoder library optimized for a DM642 EVM capable of real−time D1 encoding
- JPEG decoder library optimized for a DM642 EVM capable of real−time D1 decoding
- JPEG encoder and decoder library integrated with the IDMA layer specification
- JPEG encoder and decoder library implemented using XDAIS interfaces
- Sample integration of JPEG encoder and decoder under RF−5 framework, to demonstrate JPEG loopback (decode + encode) at programmable quality
- Sample integration of a peer-to-peer based networking system. Here, data is always terminated at the network for capture, and sourced at the network for display. This sample is thus the most applicable to real-world configurations
- A Windows "DOS box" example application to record and play video and implement an external loopback for testing

The addition of networking functionality allows an application to stream JPEG data in and over a network. Applications include security, web cameras, and point−to−point video distribution.

This application uses true independent streams for transmit and receive. Thus, the RF5 model in this example most closely models real-world networking applications.

Contents

List of Figures

1 Software Architecture/Data Flow

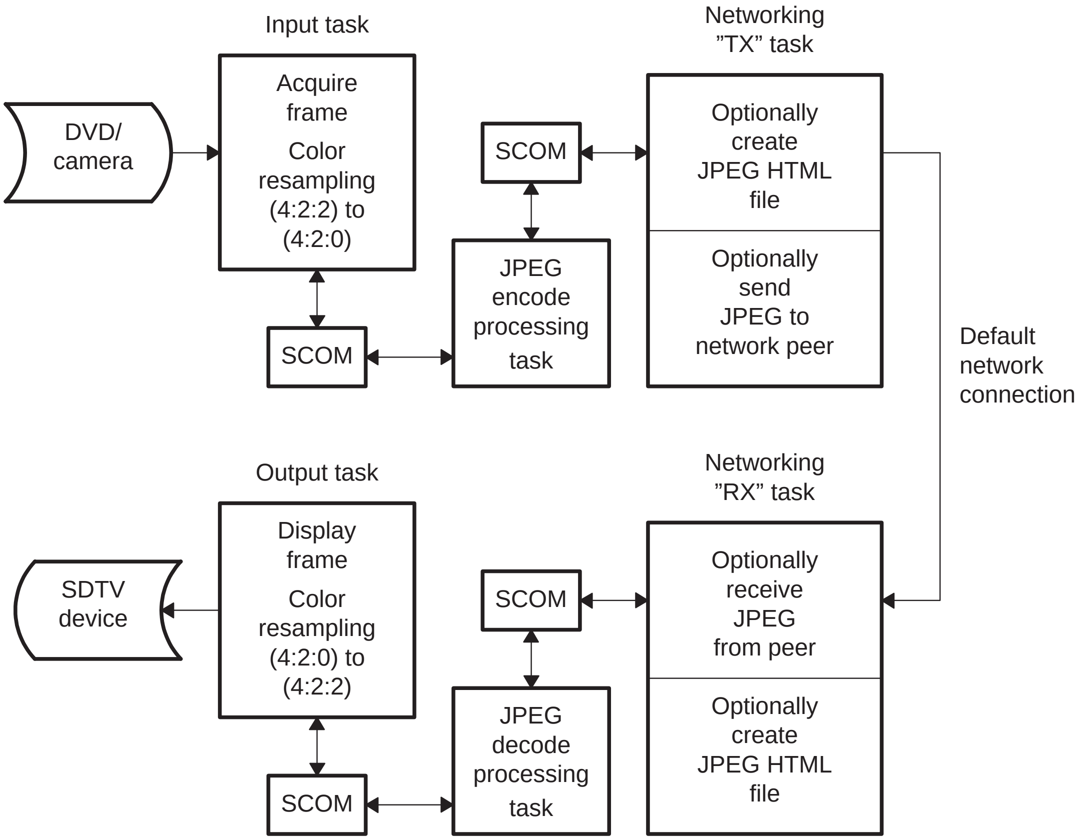

Input task
Acquire frame
Color resampling (4:2:2) to (4:2:0)
DVD/ camera
SCOM
SCOM
JPEG encode processing task
Networking ”TX” task
Optionally create JPEG HTML file
Optionally send JPEG to network peer
Default network connection
Output task
Display frame
Color resampling (4:2:0) to (4:2:2)
SDTV device
SCOM
SCOM
JPEG decode processing task
Networking ”RX” task
Optionally receive JPEG from peer
Optionally create JPEG HTML file

Figure 1. Data Flow Diagram

1.1 Data Flow Diagram for the Demonstration

The data flow in the demonstration follows this sequence for capture:

1. A frame is captured from the input source (DVD/camera), and the acquired frame data, in YUV 4:2:2 format, is resampled to YUV 4:2:0 format.
2. The frame is fed from the input task to the encode processing task via a SCOM queue.
3. The encoder processing task generates a JPEG encoded image at a desired quality set by the user. The JPEG file is sent to the networking transmit (TX) task via SCOM.
4. The networking TX task optionally creates a local copy of the JPEG for WEBCAM functionality. It then sends the JPEG image over the network to the connected peer. (By default, the example program “connects” to itself.)

The data flow in the demonstration (see Figure 1) follows this sequence for display:

1. The networking receive (RX) task receives a new JPEG image from the network peer. It optionally creates a local copy of the JPEG for WEBCAM functionality. It then sends the JPEG to the decode processing task via SCOM.
2. The decode processing taks sends the JPEG file to the JPEG decoder. The decoded JPEG image is fed to the output task via an SCOM queue.
3. The output task converts the decoded YUV 4:2:0 image to a YUV 4:2:2 image and sends it to the TV monitor.

1.2 Framework Flowchart

This demonstration uses RF-5 framework to integrate the JPEG encoder and decoder library as well as the networking task. The demonstration uses an eight task setup, six of which are shown in Figure 1.

The seventh task is a control task, which uses a mailbox to send messages to the processing task. The processing task receives messages from this mailbox and changes the application parameters based on the nature of the message. Currently, only the changing of the quality of the encoded image is implemented.

The eighth task is the network initialization task. It is defined in the CDB and manages initializing the networking environment. Once the network is ready, the networking task shown in the data flow diagram is created.

NOTE: This networking example is unique in that capture and display operate truly independently of each other. (Task priority is used to ensure that JPEG encode and JPEG decode do not run simultaneously, as they share a scratch buffer.)

Before coming to the DSP/BIOS™ task scheduler, the demonstration code initializes various modules used in the system. These include:

- Board and processor
  - The system performs DSP/BIOS™ initialization and CSL initialization.
  - The L2 cache mode is set to 128K cache.
  - EMIFA CE0 and EMIFA CE1 spaces are enabled for caching.
  - The DMA priority queue lengths are set to maximum.
  - Priority for L2 request priority set as high.
  - The DMA manager is initialized with allocated internal and external heap.
- RF-5 modules
  - The system initializes the channel module of RF-5.
  - The system initializes the ICC and SCOM modules of RF-5 required for intercell communication and messaging.
  - Channel setup is performed with the internal and external heap buffers.
- Capture and display channels
  - An instance of capture channel is created and started.
  - An instance of display channel is created and started.

After these initializations, the system enters the eight-task system managed by the DSP/BIOS™ scheduler. These eight tasks use SCOM module of RF-5 to communicate to each other:

- Input task

The input task is responsible for acquiring the frames from the NTSC/PAL input device. It uses FVID_exchange calls provided by the driver to acquire a frame. The acquired frame is in YUV 4:2:2 format and is resampled to YUV 4:2:0 format. It then sends the message to the encode processing task with the frame pointer embedded in the message. The task then waits for the message from the encode processing task to continue.

- Encode Processing task

The encode processing task has one cell in this demonstration. This cell is a JPEG encoder that accepts an input YUV 4:2:0 image and produces a compressed JPEG image at a desired quality.

The responsibility of the encode process task is to pass the input image to the JPEG encode cell. The JPEG image is passed to the networking TX task for transmission over the network.

- Decode processing task

The decode processing task has one cell in this demonstration. The cell is a JPEG decoder cell that accepts a JPEG encoded image and produces a decoded image in YUV:2:0 format.

When the networking RX task receives a JPEG image it passes it to the decode processing task for JPEG decode. After the completion of the JPEG decode, the output image is then passed to the output task via an SCOM message.

- Output task

The output task is responsible for displaying the frames on the NTSC/PAL output device. It uses FVID_exchange calls provided by the driver to display a frame. The acquired frame is in YUV 4:2:0 format and is resampled to YUV 4:2:2 format prior to display. The task then waits for a message from the processing task to continue.

- Control task

The control task is responsible for controlling the parameters that are variable within this demonstration. These include the quality factor of the demonstration. The control task checks for a change in value in the parameters defined in a global structure, ExternalControl, (visible to the user in global space). It then copies the values of the changed parameters into a local structure, externalControlPrev and posts messages in a mailbox to the process thread. The encode processing task periodically checks for messages and calls the corresponding cell's control function.

- Network initialization task

The network initialization task is charged with booting up the networking environment. When the network is ready, the initialization task creates the RX and TX networking tasks. The initialization source code for all networking tasks is remarkably similar. Refer to the TMS320C6000 TCP/IP Network Developer's Kit (NDK) User's Guide (SPRU523).

- TX networking task

The TX networking task is used to send JPEG images to a network peer. When the task is initialized, it creates a socket to listen on port 2000.

The networking task then waits for a SCOM message from the processing task indeicating that a new JPEG image is available. The networking task takes the JPEG image and optionally creates a RAM-based image file (IMAGE1.JPG) that the HTTP server recognizes and can serve to HTTP clients.

When a connection is available, the TX networking task sends the JPEG file to the networking peer. If a new network connection is detected, the IP address of the peer is recorded so that the EX networking task can connect to the same peer. This allows auto-connect of the second half of the two-way communication.

- RX networking task

The RX networking task is used to receive JPEG images from a network peer. When the task is first initialized, it attempts to connect to the default IP address (which is set to its own local address). During execution, the RX networking task can be told to reconnect to an alternate peer by either the user (through HTML interface) or the TX networking task (in response to a new connection).

When a conection is available, a JPEG image is received from the networking peer. The RX networking task then passes the JPEG image to the decode process task via an SCOM message.

2 System Requirements/Configuration

2.1 Software Requirements

- Microsoft Windows NT (SP6)/Microsoft Windows 2000 (SP1 and SP2)/Microsoft Windows XP
- Code Composer Studio™ Integrated Development Environment (IDE) version 2.21 or greater
- Reference Frameworks (RF 2.20)

2.2 Hardware Requirements

- Pentium machines with 450 MHz, 64MB RAM (minimum)
- DM642 EVM
- NTSC/PAL TV for display purposes
- Camera/DVD for NTSC/PAL capture purposes
- XDS 510/560 emulator

3 Hardware Setup

To run the demonstration, the hardware must be set up properly as shown in Figure 2:

- The DM642 EVM must be connected to the appropriate power source.
- The input video port (for composite video) must be connected to NTSC/PAL input.
- The source (DVD/camera) must be connected using RCA cable.
- The output video port (for composite video) must be connected to NTSC/PAL.

- The output device (SDTV) must be connected using RCA cable.
- The XDX510/560 emulator must be connected to the JTAG pins to download the demonstration code to the board and control it from Code Composer Studio™ IDE.

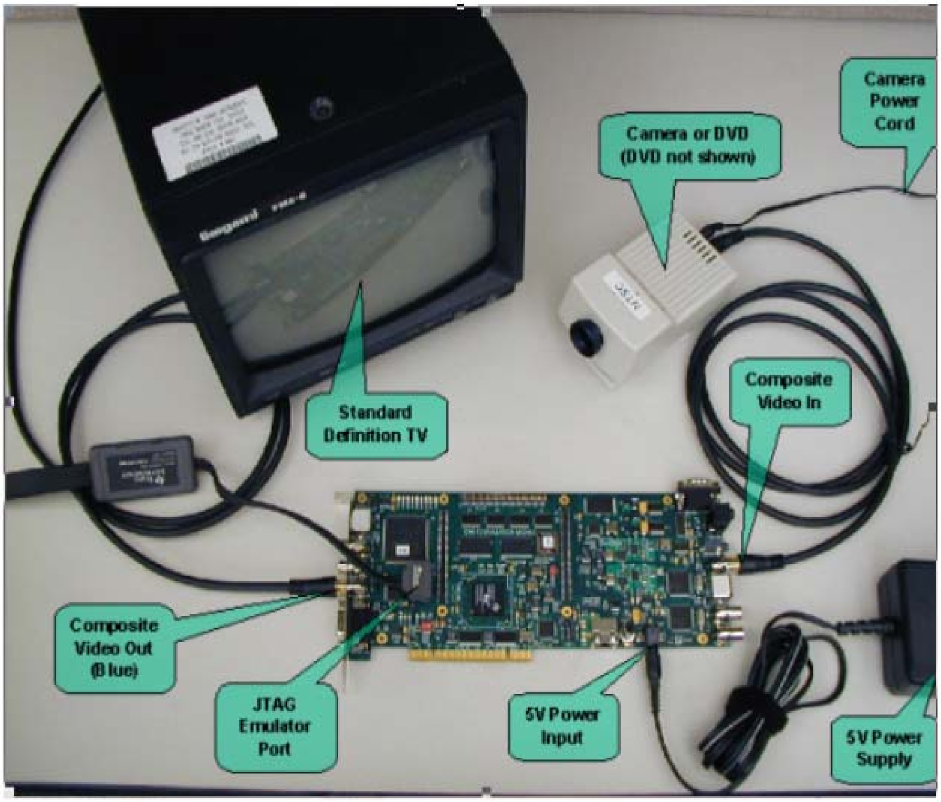


Figure 2. Hardware Setup

4 Demonstration Execution

NOTE: Some of the example applications described in this section require a network with support for DHCP. If DHCP is not available, the example must be rebuilt to use a fixed IP configuration. This can be done using Code Composer Studio™ IDE. Refer to the TMS320C6000 TCP/IP Network Developer's Kit (NDK) User's Guide (SPRU523) for details on network application initialization.

To run the demonstration, follow these steps:

1. Connect the NTSC/PAL input device (camera/DVD) using proper RCA cables.
2. Connect the NTSC/PAL output device (SDTV) using proper RCA cables.
3. Power up the DM642 EVM board.
4. Start Code Composer Studio™ IDE version 2.21 or greater.
5. Check the color bar on the output device.
6. Go to the bin folder under the jpeg_network directory and load jpeg_network_NTSC.out or jpeg_network_PAL.out.
7. Once the program is loaded, go to the Debug Menu in Code Composer Studio ™ IDE and press the Run option (F5).
8. On the output screen, watch the JPEG reconstructed image with the TI logo at the top-right corner of the screen.

In a successful execution, one of the status lines printed by the application displays the client IP address (either through DHCP or static configuration). Once this address displays, the DSP responds to requests made to its IP address. When using DHCP, it is possible that the application will be unable to obtain an IP address from a DHCP server. In this case, the application eventually prints a DHCP status message with the fault condition.

Once the networking task is ready and an IP address is obtained, you can connect an HTTP browser to that address to see the NETCAM images displayed on the browser window.

NOTE: The simple JAVA application used to implement the NETCAM is not compatible with all JVM's. Specifically, JVM's that cache display images will continuously display the same image. Caching can usually be disabled in the JVM configuration.

You can use this Web page to connect to a peer EVM running the same example program. The NETCAM can be set to display either the local or the peer-generated video stream. (The display monitor always shows the peer video stream.)

To record and playback video sequences, a client application is run from a Windows DOS box. From the command line, run the program vecho with the IP address as the argument. For example, if the IP address is 192.168.1.45, type:

```
vecho 192.168.1.45
```

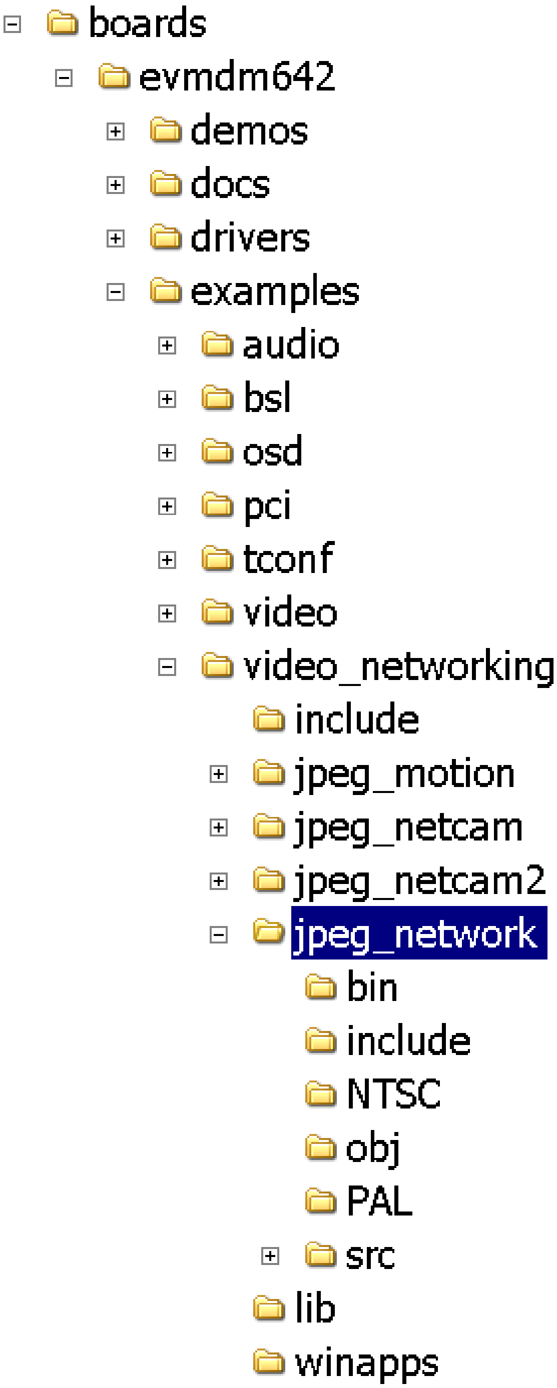
Once the echo program is fully connected, press the spacebar to get a list of vecho commands. Using the keyboard, you can record and play back video streams. By default, vecho simply sends back the same JPEG image it received from the peer.

NOTE: Do not run NETCAM and vecho on the same PC. The NETCAM java application uses up a lot of PC CPU cycles. If you want to run them simultaneously, use two PCs.

4.1 Controlling Parameters of the Demonstration

- On the Web page with the NETCAM display is a setting that allows the quality of the resulting JPEG decoded image to be varied. The allowed values for quality are of the form 1 <= quality <= 100 with both end values being legal.
- Another setting on the Web page allows the user to input the peer's IP address. Related to this, the 'NETCAM Update' setting controls the video stream you want to display.

5 Demonstration Code and Build Procedure



```
⊟ boards
    ⊟ evmdm642
        ⊞ demos
        ⊞ docs
        ⊞ drivers
        ⊟ examples
            ⊞ audio
            ⊞ bsl
            ⊞ osd
            ⊞ pci
            ⊞ tconf
            ⊞ video
            ⊟ video_networking
                  include
                ⊞ jpeg_motion
                ⊞ jpeg_netcam
                ⊞ jpeg_netcam2
                ⊟ jpeg_network
                      bin
                      include
                      NTSC
                      obj
                      PAL
                    ⊞ src
                  lib
                  winapps
```

Figure 3. Directory Structure for jpeg_network

5.1 Build Procedure

1. Start Code Composer Studio™ IDE version 2.21 or greater.
2. Open the jpeg_network project (jpeg_network.pjt) in the folder called boards\examples\video_networking\jpeg_network.
3. Go to Project−>Build and rebuild the project.
4. Compiler options used under the pre-processor are CHIP_DM642 = 1, C6000, and UTL_DBGLEVEL = 70.
5. Build the project and load the executable from the bin directory: jpeg_network_NTSC.out or jpeg_network_PAL.out.
6. Press F5 to watch the decoded JPEG image.

6 Known Bugs and Constraints

- The quality factor must be in the range 1 to 100.
- Decoder checks if decoded image is a valid JPEG and returns a negative error code on receiving an incorrect JPEG stream.

7 Performance of JPEG Encoder and JPEG Decoder

The performance of JPEG encoder and decoder images is content− and quality−dependent. The figures shown here are for a quality setting of 75 and reasonable complexity images, which are typical.

- D1 4:2:0 encode for typical images at a quality setting of 75% uses 23% of a 600−mHz C64x DSP.
- D1 4:2:0 decode for typical images at a quality setting of 75% uses 20% of a 600−mHz C64x DSP.

8 References

TMS320C6000 TCP/IP Network Developer’s Kit (NDK) User’s Guide (SPRU523)

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 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.

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.

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
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
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
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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