Application Note

RBTX4964 TX4964 Reference Board Video Capture Application Note

Highlights

- Learn about the technical details of video capture, rotation, performance, image artifacts, non-blocking rotation and how to improve display rate.
- Learn how to write benchmark graphics programs using Toshiba's graphics library for the TX4964 driver information display controller.
- Learn how to use the Toshiba TX4964 single-chip solution that integrates the CPU, hardware-graphics engine, digital-camera interface and frame-buffer memory in a compact 176-pin QFP package.

Introduction

The purpose of this application note is to describe the key technical considerations of a TX4964 video capture demo program. This application note attempts to cover the technical details on video capture, rotation performance, image artifacts, nonblocking rotation and improvement of display rate.

Block Diagram

The following picture is a video capture block diagram using the TX4964. The input format of the video stream is NTSC, which has a resolution of 720x480. The Frame Grabber (FG) captures the video frames, scales them down by 1/4 and saves them into the system memory. The pictures are then rotated by 90 degrees by the Graphics Accelerator (GA), and displayed on the monitor by the Graphics Display Controller (GDC).

Development Environment

Hardware

- RBTX4964 reference board
- RBTX4964EX-CVBS daughter board with ADV7180 video decoder
- NTSC video camera
- Green Hills Probe
- Monitor
- Windows host machine

Software

- Green Hill MULTI software development environment
- TX4964 video capture program

Video Capture

The analog video source (NTSC Format) is converted to digital video format (ITU656) using the RBTX4964 daughter board's video decoder chip, the ADV7180.

ITU656 uses the YCbCr 4:2:2 color mode and the HSYNC and VSYNC signals are provided inside the data stream. The chip is set to 16-bit output mode. All of the configurations to the ADV7180 are made via the I^{2}C bus connection to the TX4964.

ITU656 video data is then input into the TX4964FG. The input data stream is captured pixel-by-pixel and converted to 16-bit RGB format. The picture can be processed during the capture, for example, scaling or cropping can be performed. Afterwards, the data is written into capture buffers in system memory. The whole process has no MPU intervention.

Key FG configuration settings for the video

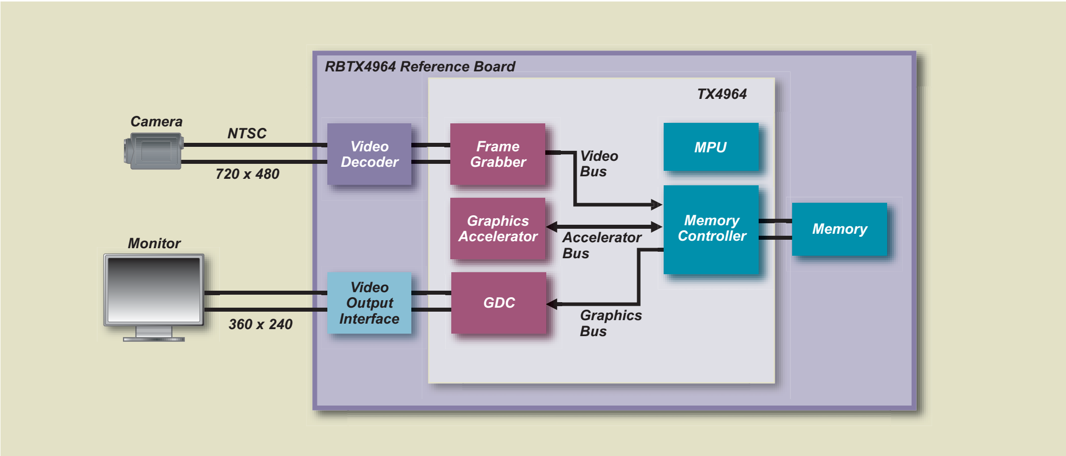

Figure 1. Video Capture Overview

capture demo:

- RGB parallel input mode
- Input data is interlaced
- ITU-R 656 input mode
- No de-interlace output. The interlaced picture is output in each video capture buffer
- 16-bit RGB output mode

Rotation

Basic Operation

The rotation engine is embedded in the GA. It is triggered by the Command Scheduler. In order to execute rotation operation, it is necessary to establish a command list that includes rotation related parameters. Figure 2 gives an explanation of the rotation operation.

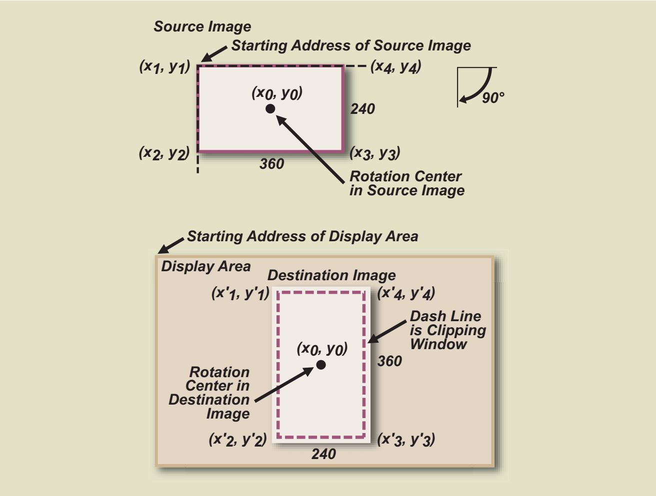

Figure 2. Rotation Operation

Performance

Figure 3. shows the execution time of the rotation for a 360x240 sized image.

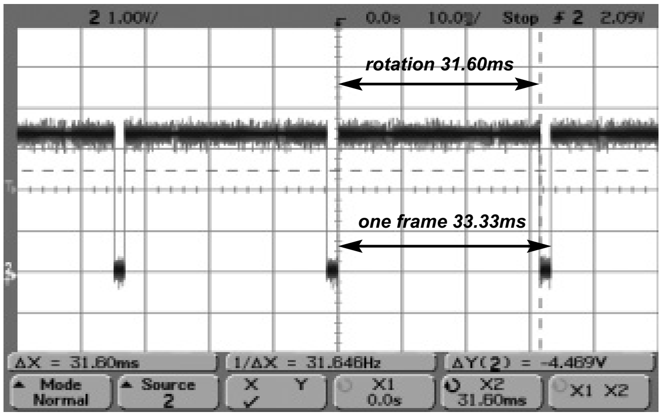

Figure 3. Execution Time of Rotation

The rotation engine takes around 31.60 msec, which is too long and the system does not have enough time to perform other graphics functions within one frame time. To solve the issue, we can utilize the GA read Cache Buffer.

The idea of the Cache Buffer is to provide a very high-speed working storage area for the graphics accelerator unit. The Cache Buffer is organized in 64 cache lines. Each cache line holds 32 bytes. The Cache Buffer only performs burst accesses to memory. On a cache miss, a complete cache line is fetched via an aligned 32 bytes burst read from memory and stored in the cache buffer. If the source image can be divided into small pieces that can fit inside the Cache Buffer, the rotation operation will complete faster because the Rotation Engine loads these pixel's data from the Cache Buffer directly instead of from the memory. By breaking a rotation operation up into a series of piecewise rotations, the overall execution time can be reduced. Figure 4 shows this optimal rotation operation. The source image is divided into 40x40 pixel blocks. Figure 5 shows the execution time of the optimal rotation.

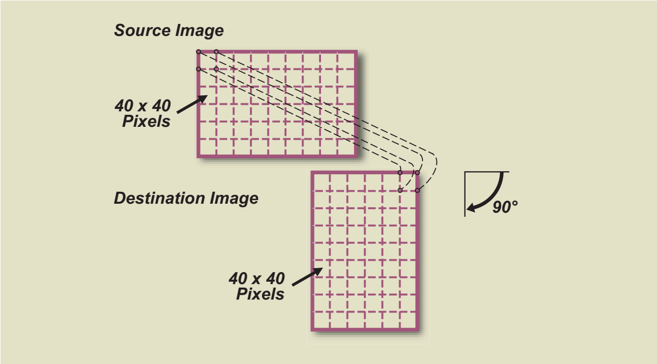

Figure 4. Optimized Rotation

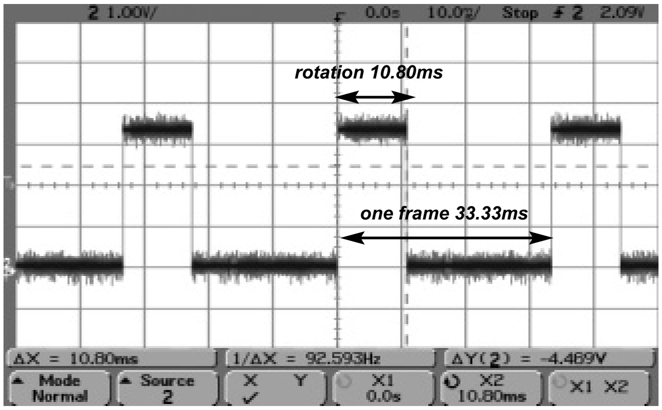

Figure 5. Execution Time of Optimized Rotation

Artifacts

Artifacts are distortions in the captured video frames that are a side-effect of the frame-grabbing process. The TX4946FG offers several ways to deal with these artifacts, allowing the programmer to make tradeoffs between memory size and artifact suppression.

Single Capture Buffer Mode

Single Capture Buffer mode has lower memory requirements. The video data is simply transferred directly into the memory location of the Capture Buffer area and is then rotated by the Rotation Engine. This scenario might show artifacts since the data in the Capture Buffer may contain a portion of the new video frame and a portion of the old video frame at the moment when the Rotation Engine fetches the data from the Capture Buffer, see Figure 6 for the image artifact.

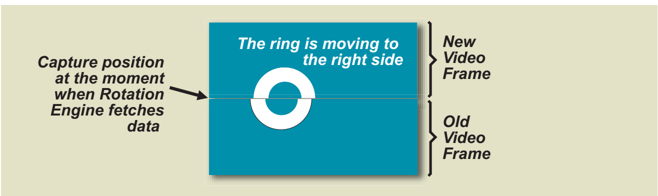

Figure 6. One Capture Buffer – Image Artifact

Single Frame Buffer

Similarly, the Single Frame Buffer might show some artifacts too, since the Rotation Engine keeps sending rotated data into the Frame Buffer. On the other hand, GDC keeps scanning the Frame Buffer starting at every V-sync. So at the moment V-sync occurs, the frame buffer might contain some portion of the new rotated video frame and some portion of an old frame. See Figure 7 for the artifact.

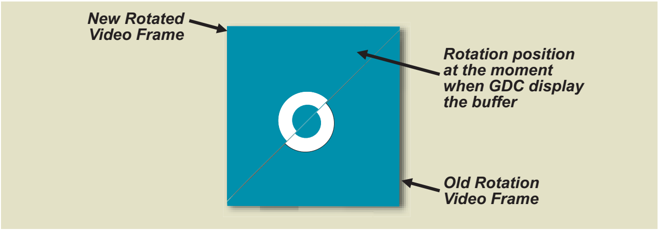

Figure 7. One Frame Buffer – image Artifacts

Double Capture Buffers and Frame Buffers

To solve the above artifacts, double capture buffers and double frame buffers can be used.

With two capture buffers, FG gets two capture buffer start addresses, which are automatically switched after the reception of each complete frame. The Rotation Engine waits for the completion of a frame, then fetches data from the capture buffer, and starts the rotation operation. After the rotation operation completes, the data is output into a write frame buffer, the input capture buffer can be released for the future capture, and the Rotation Engine waits for the next capture buffer available.

At the next GDC v-sync, the frame buffers are switched, the write frame buffer becomes the display buffer, and the old display frame buffer becomes the write frame buffer for filling in a new rotated video frame. Figure 8 illustrates this principle.

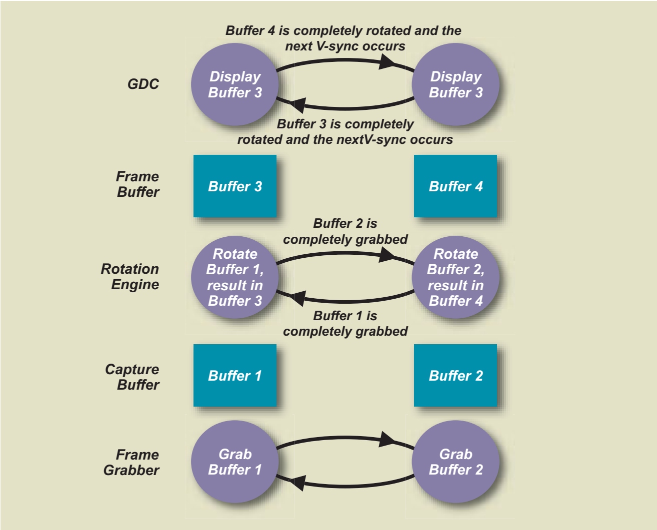

Figure 8. Double Capture Buffers and Double Frame Buffers

Frame Rate

30Hz

To show 30 frames per second, we can only choose one field (odd field, for example) of input image and get rid of the other field.

The input video is NTSC format. It has a refresh rate of 60 interlaced fields per second. This means there are 30 odd fields and 30 even fields per second. Since the output image is required to be downscaled by 1/4 of the input image, and for each field, the height is 1/2 of the original size, so we can only choose one field, down-scale its horizontal side by 1/2 and the output image will be the required size, and output frame rate is 30 Hz.

To implement this, the de-interlace mode has to be turned off in FG, odd and even fields are captured into two separate buffers and only one buffer of data (odd or even) is used for rotation and display.

60Hz

In the above section, only one field is used, this leads to a loss of information. To keep all information, we used two display layers, one for each field, two sets of frame

TAEC Sales Offices

Toshiba America Electronic Components, Inc.

Headquarters

19900 MacArthur Boulevard
Suite 400
Irvine, CA 92612
TEL: 949-623-2900
FAX: 949-474-1330

Toshiba America Electronic Components, Inc.

2590 Orchard Parkway
San Jose, CA 95131
TEL: 408-526-2400
FAX: 408-526-2410

5959 Gateway West,
Suite 405
El Paso, TX 79925
TEL: 915-771-8156

2150 E. Lake Cook Road,
Suite 310
Buffalo Grove, IL 60089
TEL: 847-484-2400
FAX: 847-541-7287

48679 Alpha Drive #120
Wixom, MI 48393
TEL: 248-347-2607
FAX: 248-347-2602

959 Route 46 East
Parsippany, NJ 07054
TEL: 973-541-4715
FAX: 973-541-4716

290 Donald Lynch Blvd.,
Suite 201
Marlborough, MA 01752
TEL: 508-481-0034
FAX: 508-481-8828

3700 Crestwood Parkway,
Suite 1070
Duluth, GA 30096
TEL: 770-931-3363
FAX: 770-931-7602

www.Toshiba.com/taec

buffers and two sets of capture buffers, seen in Figure 9. The alpha-blending algorithm will calculate the final output image.

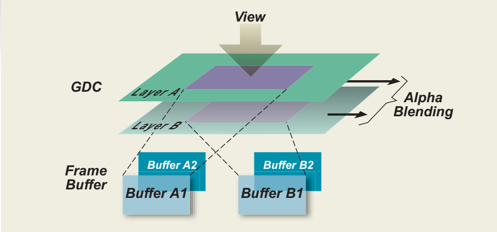

Figure 9. Two Display Layers with Alpha-blending

Non-blocking Rotation

For piecewise rotation (see Figure 4), if one command list of rotation is used for each piece, the CPU needs to keep checking the status of the rotation engine and make sure the execution of the previous list is finished before starting the execution of the next command list. This increases the CPU overhead. Also the command scheduler has to be initialized for every command list.

Actually the command list can be as long as possible. The command scheduler could synchronize all activities within one command list. So in order to decrease the CPU overhead and save CPU time, the most optimal scenario is to have a long command list for one frame, which means all of the piecewise rotations within one frame are in only one list. Figure 10 shows rotation execution time for the long command list.

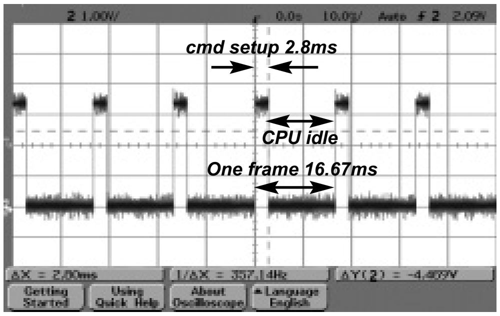

Figure 10. Rotation Execution Time for a Long Command List