LUT

Implementing the Gamma Correction Algorithm Using the TMS320C2xx DSP

APPLICATION REPORT: SPRA361

Vivian Shao
Field Application Engineer
Semiconductor Sales & Marketing
Texas Instruments Taiwan Limited Technical Staff

Digital Signal Processing Solutions
September 1997

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

TRADEMARKS

Contents

Examples

Table

.

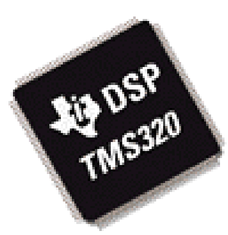
Implementing the Gamma Correction Algorithm Using the TMS320C2xx DSP

Abstract

A nonlinear effect of signal transfer exists between an electrical and an optical device. For example, the transfer function of a cathode-ray picture tube (CRT) produces an output intensity proportional to some power (usually about 2.2 and referred to as the *gamma factor*) of the signal voltage. The nonlinear effect distorts the color displayed by the CRT. To compensate for the nonlinear effect, a *gamma correction* is applied to the video signal before the CRT displays it to make the intensity output of the CRT linear.

The gamma correction is an image-processing algorithm that compensates for the nonlinear effect of signal transfer between electrical and optical devices. The image processing performed by video applications, such as CRTs, digital cameras, color printers, and scanners, includes a gamma correction for the output. Although a PC may implement image-processing algorithms for its peripheral equipment, digital signal processors (DSPs), such as the Texas Instruments (TI™) TMS320C2xx (’C2xx) DSP, are essential in implementing the image-processing algorithms for stand-alone systems.

This application note describes how to implement the gamma correction algorithm included with the ’C2xx DSP software. The document includes two main parts: the basic gamma correction theory and formula, and the ’C2xx gamma correction software description. Appendixes A through D present the assembly source code.



Product Support on the World Wide Web

TI's World Wide Web site at www.ti.com contains the most up-to-date product information, revisions, and additions. New users must register with TI&ME before they can access the data sheet archive. TI&ME allows users to build custom information pages and receive new product updates automatically via email.

Gamma Correction Theory and Formula

The following transformation equation uses a gamma factor of 2.2, which is typical in the consumer video environment.

$$
\begin{aligned}
R_{display} &= R_{received}{}^{2.2} \\
G_{display} &= G_{received}{}^{2.2} \qquad \text{(equation 1)} \\
B_{display} &= B_{received}{}^{2.2}
\end{aligned}
$$

where R, G, B values are normalized to the range of [0,1].

To compensate for the nonlinear processing of the display, the linear RGB data is gamma-corrected as follows:

$$
\begin{aligned}
R_{transmit} &= R_{received}{}^{0.45} \\
G_{transmit} &= G_{received}{}^{0.45} \qquad \text{(equation 2)} \\
B_{transmit} &= B_{received}{}^{0.45}
\end{aligned}
$$

where the R, G, B values are normalized to the range of [0,1].

Therefore, the displayed signals become linear to the receive signals.

$$
\begin{aligned}
R_{display} &= R_{transmit}{}^{2.2} = (R_{received}{}^{0.45})^{2.2} = R_{received} \\
G_{display} &= G_{transmit}{}^{2.2} = (G_{received}{}^{0.45})^{2.2} = G_{received} \\
B_{display} &= B_{transmit}{}^{2.2} = (B_{received}{}^{0.45})^{2.2} = B_{received}
\end{aligned}
$$

This application report assumes that the input digital signal is 8 bits wide; that is, the range of the input digital signal is within range [0, 255]. The compensate gamma factor is 0.45. Therefore, the gamma correction formula is

$$
\begin{aligned}
R_{transmit} &= 255 \times \left(\frac{R_{received}}{256} \right)^{0.45} \\
G_{transmit} &= 255 \times \left(\frac{G_{received}}{256} \right)^{0.45} \qquad \text{(equation 3)} \\
B_{transmit} &= 255 \times \left(\frac{B_{received}}{256} \right)^{0.45}
\end{aligned}
$$

The TMS320C2xx assembly language code described in this application report implements the gamma correction formula shown in equation (3).

Software Description

The gamma correction software contains three parts:

- The first part creates a look-up table, which is the most efficient way to obtain the corrected output data.
- The second part declares the variables and initializes the coefficients and tables used in the main program.
- The third part is the main program, which derives the corrected data from the original signal.

Creating the Gamma Correction Look-Up Table

The gamma correction look-up table avoids the complicated calculation of power. Data is related to the original input as generated by equation 3 (see Table 1).

Table 1. Creating the Gamma Correction Look-Up Table

(R,G,B)received (hex)	00	01	02	03	04	05	....
(R,G,B)transmit (hex)	00	15	1D	22	27	2B	....

TABLES.ASM defines the look-up table (see Appendix B).

Variable Declaration and Initial Values

Two blocks of memory space are declared in the program:

GAMMA_TABLE Defines the gamma correction look-up table as shown in Table 1.

GAMMA_BLOCK Image signal data array

Both blocks are declared in C2xx assembly language code as shown in Example 1.

Example 1. Memory Blocks Declared in 'C2xx Assembly Language Code

```
;-------------------------------------------------------------------------
;      Define variables & data blocks
;-------------------------------------------------------------------------
GAMMA_TABLE        .usect  ".gma_tbl", 256
GAMMA_BLOCK        .usect  ".gma_blk", IMAGE_SIZE
```

Example 2 shows the GAMMA_TABLE initialization based on constants defined in TABLES.ASM.

Example 2. Gamma Correction Table Initialization

```
;--------------------------------------------------------------------------
;       GAMMA CORRECTION TABLE INITIALIZATION
;--------------------------------------------------------------------------
GAMMA_INIT:
        LAR             AR2, #GAMMA_TABLE
        MAR             *, AR2
        RPT             #255
        BLPD    #GTBL0, *+

        SPM             0
        SETC    SXM
        CLRC    OVM
        RET
```

Gamma Correction Using the Look-Up Table

The program shown in Example 3 assigns four auxiliary registers.

Example 3. Auxiliary Register Assignment

```
;------------------------------------------------------------------------------------------
;       AR assignment:
;       AR0 =  #GAMMA_TABLE             ; address of GAMMA_TABLE
;       AR2 -> GAMMA_BLOCK              ; data array
;       AR3 -> pointer for look-up table        ; look-up table
;       AR7 -> counter
;------------------------------------------------------------------------------------------
        LAR             AR0, #GAMMA_TABLE
        LAR             AR2, #GAMMA_BLOCK
        LAR             AR7, #IMAGE_SIZE-1
        MAR             *, AR2
```

The program shown in Example 4 shows how to obtain the corrected results from the table.

Example 4. Look-Up Table Results

```
GAMMA_LOOP:
        LAR             AR3, *, AR3     ; AR3 = content of AR2
        MAR             *0+             ; AR3 = AR3 + #GAMMA_TABLE
        LACC    *, 0, AR2
        SACL    *+, 0, AR7
        BANZ    GAMMA_LOOP, *-, AR2
```

The original data is replaced by the corrected data. The complete program is named GAMMA.ASM. Appendixes A through D contain all of the source code and linker command files.

Summary

The gamma correction algorithm included in the TMS320C2xx software compensates for the nonlinear effect of signal transfer that exists between electrical and optical devices.

The gamma correction software uses the look-up table to obtain the corrected output data and avoid the complicated and time-consuming calculation of power. The look-up table is very effective, although it requires extra memory, must be calculated in advance, and is fixed. In the example code used in this application report, data needs only five cycles to complete the gamma correction operation. The program control, BANZ, needs four extra cycles but is optional.

Reference

Video Demystified, A Handbook for the Digital Engineer, Second Edition, pp58-61, Keith Jack, HighText Interactive, Inc., San Diego, 1996.

Appendix A. gamma.asm

```
;-------------------------------------------------------------------
;     Filename:              GAMMA.ASM
;     Description:           GAMMA CORRECTION
;     Author:                Vivian Shao
;     Date:        04/02/1997
;-------------------------------------------------------------------
            .def   start
            .def   GAMMA_INIT, GAMMA_CORRECTION
            .ref   GTBL0

;-----------------------------------------------------------------
;     Define variables & data blocks
;-----------------------------------------------------------------
IMAGE_SIZE             .set    16*16
GAMMA_TABLE            .usect  ".gma_tbl", 256
GAMMA_BLOCK            .usect  ".gma_blk", IMAGE_SIZE

            .text
start:
      CALL       GAMMA_INIT
      CALL       GAMMA_CORRECTION
      B          start

;---------------------------------------------------------------------------
;     GAMMA CORRECTION TABLE INITIALIZATION
;---------------------------------------------------------------------------
GAMMA_INIT:
      LAR        AR2, #GAMMA_TABLE
      MAR        *, AR2
      RPT        #255
      BLPD       #GTBL0, *+

      SPM        0
      SETC       SXM
      CLRC       OVM
      RET

;---------------------------------------------------------------------------
;     GAMMA CORRECTION MAIN
;---------------------------------------------------------------------------
GAMMA_CORRECTION:
;-------------------------------------------------------
;     AR assignment:
;     AR0 =  #GAMMA_TABLE
;     AR2 -> GAMMA_BLOCK
;     AR3 -> pointer for look-up table
;     AR7 -> counter
;-------------------------------------------------------
      LAR        AR0, #GAMMA_TABLE
      LAR        AR2, #GAMMA_BLOCK
      LAR        AR7, #IMAGE_SIZE-1
      MAR        *, AR2

;-------------------------------------------------------
```

```
;       GAMMA CORRECTION FORMULA:
;                                  original data       0.45
;       corrected data = 255 * ( -------------------- )
;                                        256
;------------------------------------------------------------
GAMMA_LOOP:
        LAR     AR3, *, AR3             ; AR3 = content of AR2
        MAR     *0+                     ; AR3 = AR3 + #GAMMA_TABLE
        LACC    *, 0, AR2
        SACL    *+, 0, AR7
        BANZ    GAMMA_LOOP, *-, AR2

        RET
```

Appendix B. tables.asm

```
;--------------------------------------------------------------------------------------------------
;    Filename:        TABLES.ASM
;    Description:     INITIAL VALUES OF GAMMA CORRECTION TABLE
;    Author:          Vivian Shao
;    Date:            04/02/1997
;--------------------------------------------------------------------------------------------------
     .def   GTBL0
     .data
GTBL0:
     .word    0h,     15h,    1dh,    22h,    27h,    2bh,    2fh,    32h
     .word    36h,    39h,    3bh,    3eh,    40h,    43h,    45h,    47h
     .word    49h,    4bh,    4dh,    4fh,    51h,    53h,    55h,    56h
     .word    58h,    5ah,    5bh,    5dh,    5eh,    60h,    61h,    63h
     .word    64h,    65h,    67h,    68h,    69h,    6bh,    6ch,    6dh
     .word    6fh,    70h,    71h,    72h,    73h,    75h,    76h,    77h
     .word    78h,    79h,    7ah,    7bh,    7ch,    7eh,    7fh,    80h
     .word    81h,    82h,    83h,    84h,    85h,    86h,    87h,    88h
     .word    89h,    8ah,    8bh,    8bh,    8ch,    8dh,    8eh,    8fh
     .word    90h,    91h,    92h,    93h,    94h,    95h,    95h,    96h
     .word    97h,    98h,    99h,    9ah,    9ah,    9bh,    9ch,    9dh
     .word    9eh,    9fh,    9fh,    0a0h,   0a1h,   0a2h,   0a2h,   0a3h
     .word    0a4h,   0a5h,   0a6h,   0a6h,   0a7h,   0a8h,   0a9h,   0a9h
     .word    0aah,   0abh,   0abh,   0ach,   0adh,   0aeh,   0aeh,   0afh
     .word    0b0h,   0b0h,   0b1h,   0b2h,   0b3h,   0b3h,   0b4h,   0b5h
     .word    0b5h,   0b6h,   0b7h,   0b7h,   0b8h,   0b9h,   0b9h,   0bah
     .word    0bbh,   0bbh,   0bch,   0bdh,   0bdh,   0beh,   0bfh,   0bfh
     .word    0c0h,   0c0h,   0c1h,   0c2h,   0c2h,   0c3h,   0c4h,   0c4h
     .word    0c5h,   0c5h,   0c6h,   0c7h,   0c7h,   0c8h,   0c8h,   0c9h
     .word    0cah,   0cah,   0cbh,   0cbh,   0cch,   0cdh,   0cdh,   0ceh
     .word    0ceh,   0cfh,   0d0h,   0d0h,   0d1h,   0d1h,   0d2h,   0d2h
     .word    0d3h,   0d4h,   0d4h,   0d5h,   0d5h,   0d6h,   0d6h,   0d7h
     .word    0d7h,   0d8h,   0d9h,   0d9h,   0dah,   0dah,   0dbh,   0dbh
     .word    0dch,   0dch,   0ddh,   0ddh,   0deh,   0deh,   0dfh,   0e0h
     .word    0e0h,   0e1h,   0e1h,   0e2h,   0e2h,   0e3h,   0e3h,   0e4h
     .word    0e4h,   0e5h,   0e5h,   0e6h,   0e6h,   0e7h,   0e7h,   0e8h
     .word    0e8h,   0e9h,   0e9h,   0eah,   0eah,   0ebh,   0ebh,   0ech
     .word    0ech,   0edh,   0edh,   0eeh,   0eeh,   0efh,   0efh,   0f0h
     .word    0f0h,   0f1h,   0f1h,   0f2h,   0f2h,   0f3h,   0f3h,   0f3h
     .word    0f4h,   0f4h,   0f5h,   0f5h,   0f6h,   0f6h,   0f7h,   0f7h
     .word    0f8h,   0f8h,   0f9h,   0f9h,   0fah,   0fah,   0fah,   0fbh
     .word    0fbh,   0fch,   0fch,   0fdh,   0fdh,   0feh,   0feh,   0ffh
```

Appendix C. vectors.asm

```
;----------------------------------------------------------------------
;       Filename:           VECTORS.ASM
;       Description:        Define Vector Table
;       Author:             Vivian Shao
;       Date:               11/15/1996
;----------------------------------------------------------------------
        .def        reset
        .ref        start

        .sect       "vectors"
reset:  B           start
```

Appendix D. gamma.cmd

```
/*********************************************************************************************/
/*                                                                                         */
/*    Usage:      dsplnk <obj files...> -o <out file> -m <map file> lab.cmd                 */
/*                                                                                  */
/*********************************************************************************************/
vectors.obj
tables.obj
gamma.obj
-v0
-m gamma.map
-o gamma.out

MEMORY
{
            /* Program Space */
   PAGE 0 :      VECS     : origin =    0h , length =   040h             /* Vectors */
                 PROG     : origin =  040h , length =  0FC0h         /* 4K ROM */

            /* Data Space */
   PAGE 1 :      MMREGS   : origin =    0h , length =    60h          /* MMRS    */
                 B2       : origin =  060h , length =   020h   /* On-chip DARAM B2 */
                 B0       : origin = 0200h , length =  0100h             /* B0  */
                 B1       : origin = 0300h , length =  0100h             /* B1 */
                 SARAM    : origin = 1000h , length =  1000h         /* Internal RAM */
}

/*--------------------------------------------------------------------------------*/
/* SECTIONS ALLOCATION                                                            */
/*--------------------------------------------------------------------------------*/
SECTIONS
{
   vectors :       { } > VECS      PAGE 0             /* INTERRUPT VECTOR TABLE       */
   .text   :       { } > PROG      PAGE 0             /* CODE                         */
   .data   :       { } > PROG      PAGE 0             /* INITIALIZATION DATA TABLES   */
   .bss    :       { } > B2        PAGE 1             /* UNINITIALIZED DATA           */
   .gma_blk:       { } > SARAM     PAGE 1             /* GAMMA correction data buffer */
   .gma_tbl:       { } > B0        PAGE 1             /* GAMMA correction look-up table */
}
```