Writing to Flash Memory on the ADSP-2106x

Contributed by John Tomarakos *Last Modified: 10/24/96*

Introduction:

This note explains how to write to ADSP-2106x (SHARC) BMS space using the boot select override (BSO mode). The explanation includes a general description of BSO operations and a source code example for writing to flash memory or EEPROM.

The BSO (Boot Select Override) mode bit in the SYSCON register allows the $\overline{\text{BMS}}$ pin to be asserted under software control. In PROM boot mode, an initial 256-word boot-kernel is automatically read from an 8-bit wide PROM. This kernel can set the BSO bit to let the DSP read the remaining application code from the PROM. In many systems, the boot data may need to be updated or modified. In these cases, the PROM may be substituted by a write-able EEPROM or FLASH memory. The code example demonstrates how to write to the same memory space, selected by the $\overline{\text{BMS}}$ pin, that the SHARC reads from in PROM boot mode.

Notes on BSO operation

When BSO is set, the $\overline{\text{BMS}}$ pin will assert only on an External Port (EP) DMA read or write. $\overline{\text{BMS}}$ does not assert on a DSP core access of the EP.

When BSO is set, an EP DMA does not assert the $\overline{\text{MS1}}$x pins. DSP core accesses are not affected by BSO.

To write to memory with the $\overline{\text{BMS}}$ asserted, use DMA channels 7, 8 or 9. DMA channel 6 should only be used for a read when BSO is set. This limitation of accesses appears because DMA channel 6 is hardwired for a special 8-bit boot read mode

When BSO is set, a write with DMA channel 6 with BSO set results in illegal chip operation.

When BSO is set, DMA channels 7-9 can be used with any of the modes available in the DMACx register (for read or write), any packing mode, and any data or instruction.

BSO Write Code Example

The following code example uses DMA channel 7 to write to an 8-bit wide memory connected to pins DATA23-16 of the EP bus. The connections between the SHARC and flash memory appear in Figure 1. Following the code example, an architecture file for a corresponding SHARC-flash-memory system appears.

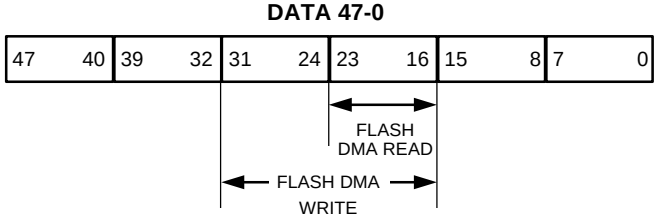

Figure 1 - SHARC-To-Flash-Memory Connections

```
/*_____________________________________________

  BSOWRITE.ASM
  ADSP-21060 Flash Write Transfer Example

_____________________________________________*/

/* define number of 48-bit words
   to write to FLASH */

#define N 8
#include "def21060.h"
.SEGMENT/DM  dm32_b1;
.VAR temp;
.ENDSEG;
.SEGMENT/DM  extdata;
             /* Location of FLASH memory */
```

```
.VAR flash[N*6];
.ENDSEG;
            /*  interrupt vector table  */
.SEGMENT/PM     pm48_1b0;
resoi:  nop;nop;nop;nop;
                              /* reserved */
reset:  nop;nop;nop;jump start;
res1i:  nop;nop;nop;rti;
                              /* reserved */
sovfi:  nop;nop;nop;rti;
   /* status stack ovflw, PC stack full */
tmzhi:  nop;nop;nop;rti;
                 /* timer high priority */
virpt:  nop;nop;nop;rti;
                     /* vector interrupt */
irq2i:  nop;nop;nop;rti;
                        /* irq2 asserted */
irq1i:  nop;nop;nop;rti;
                        /* irq1 asserted */
irq0i:  nop;nop;nop;rti;
                        /* irq0 asserted */
res2i:  nop;nop;nop;rti;
                              /* reserved */
spr0i:  nop;nop;nop;rti;
                   /* sport0 receive DMA */
spr1i:lp0i:  nop;nop;nop;rti;
   /* sport1 receive or link port 0 DMA */
spt0i:  nop;nop;nop;rti;
   /* sport0 transmit or link port 0 DMA*/
spt1i:lp1i:  nop;nop;nop;rti;
                 /* sport1 transmit DMA */
lp2i:  nop;nop;nop;rti;
                      /* link port 2 DMA */
lp3i:  nop;nop;nop;rti;
                      /* link port 3 DMA */
ep0i:lp4i:  nop;nop;nop;rti;
       /* ext port 0 or link port 4 DMA */
ep1i:lp5i:  nop;nop;nop;rti;
       /* ext port 1 or link port 5 DMA */
ep2i:  nop;nop;nop;rti;
                      /* ext port 2 DMA */
ep3i:  nop;nop;nop;rti;
                      /* ext port 3 DMA */

.VAR source[N];
      /* block of 48-bit wide test data */

/*_______ start of main routine ________*/

start:
  ustat1=0x00108421;
  dm(WAIT)=ustat1;
     /* No waistates enabled in example */
  px1=0x2211;
         /* Fill source with test data */
  px2=0x66554433;
  i8=source;
  l8=0;
  m8=1;
  lcntr=@source;
  do fill_tst until lce;
fill_tst:  pm(i8,m8)=px;

call write_block;   /* Write out source */

stop:   jump stop;

/* ---------------------------------------
Write block of 48-bit words to 8-bit wide
memory with BMS pin asserted
---------------------------------------
*/
write_block:
  i8=source;
  l8=0;
  m8=1;        /* Setup for DMA channel */
  dm(IM7)=m8;
  dm(EM7)=m8;
  r15=flash;
  dm(EI7)=r15;
  ustat1=dm(SYSCON);
  bit set ustat1 BSO;        /* Set bso */
  dm(SYSCON)=ustat1;
```

```
  lcntr=@source;
  do wb_loop until lce;
    call write_48;
    nop;
    nop;
wb_loop:  nop;
  ustat1=dm(SYSCON);
  bit clr ustat1 BSO;      /* Clear bso */
  dm(SYSCON)=ustat1;
  rts;


/*  ---  Write a 48-bit word to FLASH  ---
Extracts six 8-bit fields from a 48-bit
word, places them successively into
bits 0-7 and 16-23 of three 32-bit words.
This places the bytes into the proper
bit positions for three DMA writes to a
FLASH memory with 32/16 bit packing
enabled. The FLASH memory is connected to
EP pins DATA23-16. The bytes are
transferred LSB first. See comments for
the nibble fields operated
on.
---------------------------------------
*/
write_48:
  px=pm(i8,m8);          /* 665544332211 */
  r0=px1;                    /* 00002211 */
  r0=r0 or lshift r0 by 8;   /* xx22xx11 */
  call dma_16;
  r1=px2;                    /* 66554433 */
  r0=fext r1 by 0:16;        /* 00004433 */
  r0=r0 or lshift r0 by 8;   /* xx44xx33 */
  call dma_16;
  r0=fext r1 by 16:16;       /* 00006655 */
  r0=r0 or lshift r0 by 8;   /* xx66xx55 */
  call dma_16;
  rts;


/*  --- DMA two bytes with BSO set ---  */
dma_16:
  dm(temp)=r0;
  r15=0x2000;             /* Flush DMA 7 */
  dm(DMAC7)=r15;
  r15=temp;
  dm(II7)=r15;
  r15=1; /* Internal count 1 32-bit word */
  dm(C7)=r15;
  r15=2;
       /* External count 2 16-bit words */
  dm(EC7)=r15;
  r15=0x0245;
          /* DEN,16/32 pack,TRAN,MASTER */
  dm(DMAC7)=r15;
  nop;
/* 2 cycles to wait for DMASTAT to update */
  nop;
wait_dma7:
  ustat1=dm(DMASTAT);
  bit tst ustat1 0x80;
               /* Test Channel 7 status */
  if tf jump wait_dma7;
  rts;


.ENDSEG;
```

Architecture Description File
(For The BSO Write Code Example)

```
.SYSTEM         Example;
.PROCESSOR =    ADSP21062;

.SEGMENT
/RAM/BEGIN=0x020000/END=0x0200FF/PM/width=48
         isr_tabl;
.SEGMENT
/RAM/BEGIN=0x020100/END=0x021FFF/PM/width=48
         pm_code;
.SEGMENT
/RAM/BEGIN=0x023000/END=0x027FFF/PM/width=32
         pm_data;
.SEGMENT
/RAM/BEGIN=0x028000/END=0x02FFFF/DM/width=32
         dm_data;
.SEGMENT
/PORT/BEGIN=0x404000/END=0x404000/DM/width=32
         mafeadrs;
.SEGMENT
/PORT/BEGIN=0x404001/END=0x404001/DM/width=3232
         mafedata;

.ENDSYS;
```