Parts Not Suitable for New Designs

For Additional Information

End-Of-Life Product Change Notice

Order this document as
EB417/D

EB417/D

Swapping ROM and RAM mapping on the MC68307

By James Gilbert, J. D. Round,
Applications Group, High-performance Products,
Motorola Ltd.,
East Kilbride, Scotland.

Introduction

It is often essential for embedded systems to locate the exception and interrupt vectors in read/write memory, to allow vector table changes after system boot-up. Indeed, before being able to port some debug monitors to an application, this is a requirement.

For CPU32-based 683xx integrated processors, the Vector Base Register (VBR) resolves this issue, permitting vectors to be located anywhere in the address space. For 683xx processors with a 68000 processor core (MC68302, MC68306 and MC68307, etc.) this feature is not available, and a ROM/RAM swap technique is necessary to exchange the usual ROM at address $0 with RAM.

This engineering bulletin demonstrates a RAM/ROM swap mechanism specifically for the MC68307. However, the method can be applied equally to the other 68000 core-based members of the 683xx family.

The swap mechanism relies on a short section of position-independent code and careful reprogramming of the chip selects registers during boot up. Software Listing 1 provides full details of the code used. Initially, the ROM and RAM chip selects reserve areas in the memory map at base address $0 and address $200000 respectively. Thereafter, a swap code routine is copied from ROM into RAM, before jumping to RAM to execute it. On completion, the execution returns back to continue in ROM, which is now relocated at base address $080000. Finally, the RAM is then relocated at base address $0 as required. The resulting memory map is shown in Figure 1 below.

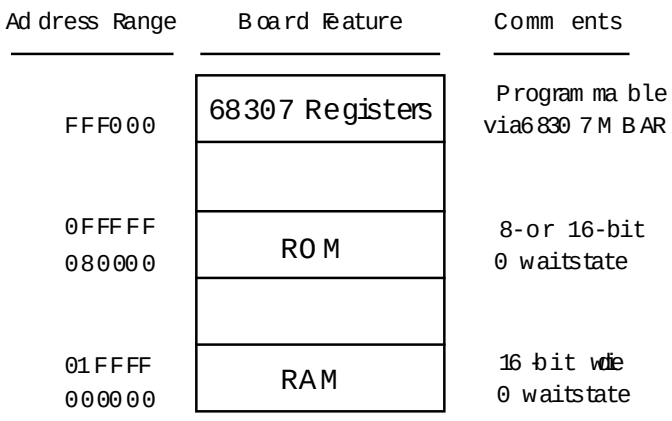

Figure 1 Memory map after swap complete

The method described uses an application with 512K of ROM and 128K of SRAM. If further details of the hardware are required, a full description is available in AN490/D “Multiple Bus Interfaces using the MC68307”.

The swap relies on the code being loaded at address $080000 during the link process, although any other multiple of the ROM size could be used (e.g. $080000, $100000 and so on....), so long as the ROM chip select is programmed accordingly.

Software Listing 1

```
*=====================================================================
* 68307 PROCESSOR REGISTERS.
REG_BASE    EQU    $FFF000           ; start of internal registers

MBAR        EQU    $0000F2           ;Base Address Register
SCR         EQU    $0000F4           ;System Control Register(32 bits)

BR0         EQU    REG_BASE+$040     ;CS0 Base Register
OR0         EQU    REG_BASE+$042     ;CS0 Option register
BR1         EQU    REG_BASE+$044     ;CS1 Base register
OR1         EQU    REG_BASE+$046     ;CS1 Option Register
WRR         EQU    REG_BASE+$12A     ;Software Watchdog Reference Reg

*=====================================================================
TMP_BASE    EQU    $00200000         ;TEMP RAM BASE FOR ROM/RAM SWAP

*=====================================================================
* Startup code section
        SECT 9,,c
        DC.L      STACK        ; Initial SP
        DC.L      $08          ; Initial PC

START   AND.W     #$FFFF,MBAR    ; MC68307 register base at $FFF000
        MOVE.W    #$0000,WRR     ; Disable Software Watchdog

        MOVE.W    #$1F02,OR0     ; 0 waits, 512kB block, r/w not masked, FC's masked
        MOVE.W    #$C001,BR0     ; CS0 Supervisor program space, base $0, read, EN

        MOVE.W    #$1FC0,OR1     ; 0 waits, 128kB block, r/w bit masked, FC's masked
        MOVE.W    #$A401,BR1     ; CS1 Supervisor data space, base $200000, read, EN

*---------------------------------------------------------------------------------
* Copy swap code to RAM
*---------------------------------------------------------------------------------
        LEA       SWAPC(PC),A0   ; Get start address of SWAP code
        LEA       SWAPCE(PC),A1  ; Get end address of SWAP code
        MOVEA.L   #TMP_BASE,A2   ; Get address of temp RAM base
SWAPLP  CMPA.L    A0,A1          ; Has all code been copied?
        BEQ       JUMPSC         ; If so, jump to code RAM
        MOVE.W    (A0)+,(A2)+    ; Otherwise,copy ROM code to RAM
        BRA       SWAPLP         ; Continue in swapping loop
JUMPSC  JMP       (TMP_BASE).L   ; Execute SWAP code in RAM

*---------------------------------------------------------------------------------
* Code for RAM to $0 and ROM to $80000 swap
* A section of code is set up to relocate ROM to $80000
* This is copied to RAM, jumped to in RAM and executed,
* before jumping back to ROM and relocating RAM at $0.
*---------------------------------------------------------------------------------
SWAPC    MOVE.W    #$C101,BR0    ; Place ROM at $80000 by writing to BR0
         JMP       (SWAPCE).L    ; Return back to new ROM location
SWAPCE
         MOVE.W    #$A001,BR1    ; Place RAM at $0 by writing to BR1
```

Literature Distribution Centres:

ASIA PACIFIC: Motorola Semiconductors (H.K.) Ltd., Silicon Harbour Center,
No. 2, Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong.

JAPAN: Nippon Motorola Ltd., 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141, Japan.

USA: Motorola Literature Distribution, P.O. Box 20912, Phoenix, Arizona 85036.