Using a PCMCIA Card as a Boot ROM on an Élan™SC300 Microcontroller Design

AMD

Application Note

The Élan™SC300 microcontroller system can provide jumpers that allow BIOS/system firmware to load and run from a linear memory card in a PCMCIA socket instead of from the on-board boot ROM. On systems with soldered-down boot ROMS, these jumper provisions are useful for boot ROM development or any situation where the boot ROM may be corrupted. This application note describes the issues involved in using this boot-from-PCMCIA feature on an ÉlanSC300 microcontroller design.

BACKGROUND

On reset, the Élan™SC300 microcontroller begins fetching code by asserting $\overline{\text{ROMCS}}$ and $\overline{\text{MEMR}}$ and putting the 24-bit address FFFFF0 on the system address lines SA23–SA0. The fetch address will remain in the top 64-Kbyte of the 24-bit address space (FFxxxx) until a far jump loads the code segment register. At that time, if the microcontroller is in protected mode, it could jump anywhere in the 24-bit address space, but if the microcontroller is in real mode at the time of the far jump, the target will be in the lowest 1 Mbyte (0xxxxx). On reset, addresses in the range FF0000–FFFFFF (FFxxxx) and 0F0000–0FFFFF (0Fxxxx) always assert $\overline{\text{ROMCS}}$ and are default 8-bit accesses.

Note that on most ÉlanSC300 microcontroller systems based on an AT-compatible BIOS, the boot code does a far jump in real mode to an address in the 0Fxxxx range, and the boot ROM is a 64- or 128-Kbyte device that only looks at the low 16 or 17 address bits. On such systems, FFxxxx and 0Fxxxx in $\overline{\text{ROMCS}}$ space will map to the same ROM device.

A PCMCIA memory card can have two 64-Kbyte address spaces addressed (attribute and common memory) using 26 address lines and a $\overline{\text{REG}}$ line, which selects one of the two address spaces. Data widths of either 16 or 8 bits are supported using low and high byte select ($\overline{\text{CEL}}$ and $\overline{\text{CEH}}$). Either an $\overline{\text{OE}}$ line is asserted to indicate a read, or a $\overline{\text{WE}}$ line is asserted to indicate a write.

On the ÉlanSC300 microcontroller, the PCMCIA address lines A23–A0 are the same as the system address lines SA23–SA0, while PCMCIA address lines A25 and A24, which are controlled by CA24–CA25 control registers 1–3 (indexes B5h–B7h), are unique to PCMCIA. The PCMCIA data lines, D15–D0, are the same as system data lines SD15–SD0. For PCMCIA $\overline{\text{OE}}$ and $\overline{\text{WE}}$, the ÉlanSC300 microcontroller can be programmed using bit 4 in the Miscellaneous 3 Register (Index BAh) to assert either $\overline{\text{MEMR}}$ and $\overline{\text{MEMW}}$ on PCMCIA accesses or to assert $\overline{\text{PCMCOE}}$ and $\overline{\text{PCMCWE}}$. ($\overline{\text{PCMCOE}}$ and $\overline{\text{PCMCWE}}$ take over pins 84 and 89 that are normally part of the parallel port interface. They are useful in systems that need to distinguish between PCMCIA accesses and ISA accesses and do not use the parallel port.)

On ÉlanSC300 microcontroller designs that use buffered PCMCIA slots, the ICDIR pin can be used to tell the buffer whether it should accept data or drive data. Unbuffered PCMCIA designs do not use the ICDIR signal. At reset, the ICDIR pin from the ÉlanSC300 microcontroller is driven High, indicating that the buffers should accept data rather than drive data. ICDIR then remains High at all times unless the microcontroller is doing a PCMCIA read.

JUMPERS TO ENABLE BOOT FROM PCMCIA

Chip Select

Because $\overline{\text{ROMCS}}$ will be asserted at boot time, there must be a jumper that breaks the path of $\overline{\text{ROMCS}}$ from the microcontroller to the $\overline{\text{CS}}$ of the boot ROM, and the path of $\overline{\text{MCEL\_A}}$ from the microcontroller to $\overline{\text{CEL}}$ of the PCMCIA socket. Then, the jumper must route $\overline{\text{ROMCS}}$ from the microcontroller to $\overline{\text{CEL}}$ of the PCMCIA socket. This will tell the PCMCIA card to place the addressed byte, whether from an even or odd address, on D7–D0, just as in an 8-bit ROM access. An example is shown in Figure 1. Jumper Configurations. For normal operation, there are jumpers from 1 to 2 and from 3 to 4. For boot from PCMCIA operation, there is a jumper from 2 to 3.

Publication# **21824** Rev: **A** Amendment/**0**
Issue Date: **July 1997**

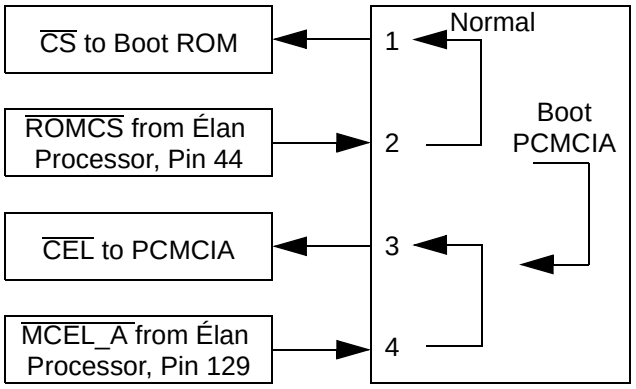

Figure 1. Jumper Configurations

While jumpered for PCMCIA boot, the PCMCIA socket cannot be used for PCMCIA accesses. The jumper could be moved back after boot, assuming any required ROMCS memory has been shadowed to DRAM. In a design with two PCMCIA sockets, the second socket can still be used for PCMCIA accesses.

PCMCIA Output Enable

Because the default for index register BAh, bit 4, is 0 at reset, only MEMR will be asserted at boot time, and so therefore either the PCMCIA socket must be designed to use MEMR (rather than PCMCOE) or a jumper must route MEMR to OE.

Buffered PCMCIA designs must also address the ICDIR issue. ICDIR is High during the ROMCS cycles that occur at boot time, High indicating that the buffer should accept data rather than drive data. For PCMCIA boot, the buffer must drive data whenever ROMCS and MEMR are both asserted. Thus, there must be a jumper that determines whether buffer ICDIR comes from the ÉlanSC300 microcontroller ICDIR (normal mode), or whether buffer ICDIR comes from a logical OR of ROMCS and MEMR (boot from PCMCIA mode).

PCMCIA Address Lines

At boot time, CA25 and CA24 are 0, REG is deasserted (indicating common memory), and SA23–SA0 are generally either FFxxxx or 0Fxxxx as described in the Background section. The PCMCIA card determines whether any address lines need to be jumpered to support boot-from-PCMCIA mode. It is optional whether a PCMCIA card fully decodes the 26-bit address it receives, or only decodes the address lines that correspond to memory actually on the card. For example, a 2-Mbyte memory card could ignore A21–A25 without affecting performance.

Because addresses in the 16-Mbyte range and the 1-Mbyte range will be driven, a memory card of a size less than 16 Mbyte that fully decodes the address must ensure that all PCMCIA address lines up to A23, greater than the size of the memory card, are jumpered to ground. If the memory card is a type that ignores High address lines, then no such jumpering is required.

Note that the extra PCMCIA address lines must be grounded and not merely disconnected. Disconnected address lines may work with some memory cards and not with others.

PUTTING THE ROM IMAGE ON THE MEMORY CARD

This section assumes that you have a binary image of the boot ROM contents.

Generally, the PCMCIA card has a larger memory size than the boot ROM device it is replacing. This, and the fact that accesses to addresses between FFxxxx and 0Fxxxx will assert ROMCS, means that care must be taken in placing the ROM image on the memory card.

For example, assume a 64-Kbyte boot ROM image and a 4-Mbyte memory card. The 64-Kbyte boot ROM only uses the low 16 address bits, and so maps FFxxxx and 0Fxxxx to the same 64-Kbyte address block. However, the 4-Mbyte memory card uses the low 22 address bits (addresses up to 3FFFFF), and so the 64-Kbyte binary image must be placed at both 3F0000 and 0F0000. The only code that must reside in 3Fxxxx is any code that occurs before the first far jump occurs. After the first far jump, the high-order address lines are not asserted by the CPU and access will be below 1 Mbyte. The 3F0000 block will satisfy accesses in the FFxxxx range, and the 0F0000 block will satisfy accesses in the 0Fxxxx range. For additional information on this subject refer to the application note entitled *Using 16-Bit ROMCS Designs in Élan™SC300 and ÉlanSC310 Microcontrollers*, order #21825.

Finally, you must use a utility program to write the binary ROM image to the necessary blocks on the memory card. AMD has a utility that works for the simple case of an SRAM memory card. Contact AMD for a copy of the utility (see the back page of this application note for information on how to contact AMD). This utility could be extended to support Flash memory cards with the addition of a Flash memory programming algorithm.

ÉlanSC300 DEMONSTRATION BOARD IMPLEMENTATION

The ÉlanSC300 microcontroller demonstration board uses a ROMCS to CEL jumper as described in the Chip Select section. The demonstration board uses MEMR for PCMCIA reads. The demonstration board uses unbuffered PCMCIA sockets, and so ICDIR is not an issue. On the ÉlanSC300 microcontroller demonstration board, address lines A23–A20 are disconnected for PCMCIA boot. This works with any memory card of 1 Mbyte or greater, whether it fully decodes the address or not. This also allows a single copy of the ROM image to be placed on the memory card at 0F0000.

REFERENCE MATERIAL

- *Élan™SC300 Microcontroller Data Sheet*, order #18514
- *Élan™SC300 Programmer's Reference Manual,* order #18470
- *Using 16-Bit $\overline{\text{ROMCS}}$ Designs in Élan™SC300 and Élan™SC310 Microcontrollers Application Note*, order #21825