PMB (Net-Tech Developments)

PO Box 48-053
Wellington, New Zealand
Phone, Fax = (04) 527-7268

1A Beth Street
Trentham
Email = paul@pmb.co.nz

PRODUCT NOTE

DATE: 22-Dec-1998 FILE: D31-000105-NN03

CPU_1A1 BOOT-UP OPERATION

INTRODUCTION

Being an eight bit micro, the CPU_1A1 has a 64K address space. The recommended configuration puts the external SRAM in the bottom 32K and the FLASH in the top 32K.

Within the CPU there is also 1K of additional RAM and 512 bytes of EEPROM. The CPU also has a group of control registers that are memory mapped. These are internal to the CPU and by default, override external memory space. This means that they are always accessible.

The 68HC11 fetches reset and interrupt vectors from the top of the address space. This would normally be FLASH memory.

When the micro comes out of reset in expanded mode, the bank select lines are undefined. This means that any of the banks of FLASH could be selected, and until the bank select outputs are initialised, it can change.

Bank select outputs and chip selects, are configured by writing to internal CPU registers. This is one of the first things that should be done out of reset.

The solution to the reset problem is to map the CPU internal EEPROM to the top of the address space. This will override the top 512 bytes of every bank of FLASH memory. Reset and Interrupt vectors and a small boot program are saved into the top end EEPROM. This area can be protected from further writing if required.

Be careful of the EEPROM CONFIG register. It cannot be read directly, due to the working copy. It should be written with FFh to ensure that the EEPROM is mapped to the right place.

The small boot program in EEPROM will configure the memory map, bank select outputs and other CPU functions, then JUMP and run code in FLASH.

The boot program, vectors and EEPROM memory mapping should be configured using the special bootstrap mode (both links on).

The CPU can boot into one of four modes depending on the combination of links. Looking at the component side of the board with the CPU (large chip) on the left, there are two jumpers to the lower right. Left to right, they are Mode B and Mode A.

PO Box 48-053
Wellington, New Zealand
Phone, Fax = (04) 527-7268

1A Beth Street
Trentham
Email = paul@pmb.co.nz

MODE B	MODE A		
off	off	=	expanded mode, for running the application
off	on	=	single chip mode, not used
on	off	=	special test mode, not often used
on	on	=	bootstrap mode, for initalising/loading

EXAMPLE BOOT PROGRAM

Following is a code example that can be used to boot the CPU and run a program starting at the bottom of the first bank of FLASH.

The ORG directive locates the code when it's assembled. In this case 2000h is in SRAM. A loader program (not shown here) will load the code and then copy it to EEPROM. Starting at FF00h.

This code occupies the top 256 bytes of EEPROM, most of which is unused. Modifying the ORG directive could free up the unused space.

The interrupt vectors are all forced to the boot program in EEPROM. These would normally point to interrupt service routines somewhere.

```
;*******************************************************************
;*******************************************************************
; EEPROM RESIDENT BOOT CODE                              * SECTION 2 *

; This section of code is copied from RAM into the top of EEPROM
; Its purpose is to allow easy booting of the CPU without having to
;  pre-load the Flash banks with boot code. The Flash bank select
;  addressing is indeterminate (messed up) following reset because
;  port G defaults to input.
; The EEPROM is mapped to the top of the address space. It therefore
;  contains the reset and interrupt vectors. The top 288 bytes of
;  EEPROM are normally protected to minimise the possibiity of
;  accidentally making the module unbootable.

BOOT  ORG   2000H                   ; first 256 bytes of external RAM

; INITIAL CPU SETUP FOR LOADER

      LDS   #01FFH                  ; init stack
      LDX   #1000H                  ; register base address

      LDAA  #10010001B              ; adpu, irqe, dly, cop = 65mS
      STAA  OPTION,X

      LDAA  #00000101B
      STAA  CSCTL,X                 ; enable program CS for 32K
      LDAA  #00000000B
      STAA  CSGADR,X                ; RAM starts at address 0000H
      LDAA  #00000001B
```

PO Box 48-053 1A Beth Street
Wellington, New Zealand Trentham
Phone, Fax = (04) 527-7268 Email = paul@pmb.co.nz

```
      STAA  CSGSIZ,X                ; RAM block size is 32K
      LDAA  #00001111B
      STAA  DDRG,X                  ; bank select bits = outputs
      CLR   PORTG,X                 ; select 1ST bank

      JMP   8000H                   ; point to APPLICATION in FLASH

;************
; Copyright Notice

      DFB   "  LOADER V1.00 01 DEC 1998  "
      DFB   "  (c) 1998 PMB  "

      FILL  0FFH                    ; fill gap with "FF"

;************
; RESET & INTERRUPT VECTORS

; Make these point to the appropriate service routines within the application.

      ORG   20FFH-29H               ; Reset & Interrupt vectors for EEPROM

      DWM   0FF00H      ; SCI       ; D6   SCI (FFD6)
      DWM   0FF00H      ; SPI       ; D8   SPI
      DWM   0FF00H      ; PAIE      ; DA   PULSE ACCUMULATOR I/P EDGE
      DWM   0FF00H      ; PAO       ; DC   PULSE ACCUMULATOR OVERFLOW
      DWM   0FF00H      ; TOF       ; DE   TIMER OVERFLOW
      DWM   0FF00H      ; OC5       ; E0   TIMER O/P COMPARE 5
      DWM   0FF00H      ; OC4       ; E2   TIMER O/P COMPARE 4
      DWM   0FF00H      ; OC3       ; E4   TIMER O/P COMPARE 3
      DWM   0FF00H      ; OC2       ; E6   TIMER O/P COMPARE 2
      DWM   0FF00H      ; OC1       ; E8   TIMER O/P COMPARE 1
      DWM   0FF00H      ; IC3       ; EA   TIMER I/P COMPARE 3
      DWM   0FF00H      ; IC2       ; EC   TIMER I/P COMPARE 2
      DWM   0FF00H      ; IC1       ; EE   TIMER I/P COMPARE 1
      DWM   0FF00H      ; RTI       ; F0   REAL TIME INTERRUPT
      DWM   0FF00H      ; IRQ       ; F2   EXTERNAL IRQ
      DWM   0FF00H      ; XIRQ      ; F4   EXTERNAL XIRQ
      DWM   0FF00H      ; SCI       ; F6   SOFTWARE INTERRUPT (SWI)
      DWM   0FF00H      ; ILLOP     ; F8   ILLEGAL OPCODE
      DWM   0FF00H      ; COP       ; FA   COP OPERATED
      DWM   0FF00H      ; CLM       ; FC   CLOCK MONITOR OPERATED
      DWM   0FF00H      ; START     ; FE   RESET

BOOTEND                             ; end of boot code loaded into EEPROM

;*******************************************************************************
;*******************************************************************************
```

These code examples have been written for the Cross-32 Meta Assembler V4.0. Syntax will have to be changed for the Motorolla AS11 freeware assembler.

Code samples are provided as examples only. There is no guarantee that the examples will work in a particular environment when applied.