Suspend.txt
S3C24XX Suspend Support
=======================

Introduction

   The S3C24XX supports a low-power suspend mode, where the SDRAM is kept
   in Self-Refresh mode, and all but the essential peripheral blocks are
   powered down. For more information on how this works, please look
   at the relevant CPU datasheet from Samsung.

Requirements
----

   1) A bootloader that can support the necessary resume operation

   2) Support for at least 1 source for resume

   3) CONFIG_PM enabled in the kernel

   4) Any peripherals that are going to be powered down at the same
      time require suspend/resume support.

Resuming

   The S3C2410 user manual defines the process of sending the CPU to
   sleep and how it resumes. The default behaviour of the Linux code
   is to set the GSTATUS3 register to the physical address of the
   code to resume Linux operation.

   GSTATUS4 is currently left alone by the sleep code, and is free to
   use for any other purposes (for example, the EB2410ITX uses this to
   save memory configuration in).

Machine Support

   The machine specific functions must call the s3c_pm_init() function
   to say that its bootloader is capable of resuming. This can be as
   simple as adding the following to the machine's definition:

   INITMACHINE(s3c_pm_init)

   A board can do its own setup before calling s3c_pm_init, if it
   needs to setup anything else for power management support.

   There is currently no support for over-riding the default method of
   saving the resume address, if your board requires it, then contact
   the maintainer and discuss what is required.

   Note, the original method of adding an late_initcall() is wrong,

  and will end up initialising all compiled machines' pm init!

  The following is an example of code used for testing wakeup from
  an falling edge on IRQ_EINT0:


```
static irqreturn_t button_irq(int irq, void *pw)
{
        return IRQ_HANDLED;
}

statuc void __init machine_init(void)
{
        ...

        request_irq(IRQ_EINT0, button_irq, IRQF_TRIGGER_FALLING,
                    "button-irq-eint0", NULL);

        enable_irq_wake(IRQ_EINT0);

        s3c_pm_init();
}
```


Debugging

  There are several important things to remember when using PM suspend:

  1) The uart drivers will disable the clocks to the UART blocks when
     suspending, which means that use of printascii() or similar direct
     access to the UARTs will cause the debug to stop.

  2) Whilst the pm code itself will attempt to re-enable the UART clocks,
     care should be taken that any external clock sources that the UARTs
     rely on are still enabled at that point.

  3) If any debugging is placed in the resume path, then it must have the
     relevant clocks and peripherals setup before use (ie, bootloader).

     For example, if you transmit a character from the UART, the baud
     rate and uart controls must be setup beforehand.

Configuration

  The S3C2410 specific configuration in `System Type` defines various
  aspects of how the S3C2410 suspend and resume support is configured

  `S3C2410 PM Suspend debug`

    This option prints messages to the serial console before and after
    the actual suspend, giving detailed information on what is
    happening

`S3C2410 PM Suspend Memory CRC`

Allows the entire memory to be checksummed before and after the
suspend to see if there has been any corruption of the contents.

Note, the time to calculate the CRC is dependant on the CPU speed
and the size of memory. For an 64Mbyte RAM area on an 200MHz
S3C2410, this can take approximately 4 seconds to complete.

This support requires the CRC32 function to be enabled.

`S3C2410 PM Suspend CRC Chunksize (KiB)`

Defines the size of memory each CRC chunk covers. A smaller value
will mean that the CRC data block will take more memory, but will
identify any faults with better precision

Document Author

Ben Dooks, Copyright 2004 Simtec Electronics