Subject: [PATCH] watchdog: spin_lock_init() fixes
Posted by Alexey Dobriyan on Mon, 29 Oct 2007 14:40:02 GMT
View Forum Message <> Reply to Message

Some watchdog drivers initialize global spinlocks in module's init function
which is tolerable, but some do it in PCI probe function. So, switch to
static initialization to fix theoretical bugs and, more importantly, stop
giving people bad examples.

Signed-off-by: Alexey Dobriyan <adobriyan@sw.ru>

```
 drivers/watchdog/alim1535_wdt.c   |    4 +---
 drivers/watchdog/davinci_wdt.c    |    4 +---
 drivers/watchdog/i6300esb.c       |    4 +---
 drivers/watchdog/ib700wdt.c       |    4 +---
 drivers/watchdog/machzwd.c        |    7 ++-----
 drivers/watchdog/mpc83xx_wdt.c    |    5 +----
 drivers/watchdog/pc87413_wdt.c    |    4 +---
 drivers/watchdog/pnx4008_wdt.c    |    4 +---
 drivers/watchdog/sbc8360.c        |    3 +--
 drivers/watchdog/sc1200wdt.c      |    3 +--
 drivers/watchdog/sc520_wdt.c      |    4 +---
 drivers/watchdog/smsc37b787_wdt.c |    4 +---
 drivers/watchdog/w83627hf_wdt.c   |    4 +---
 drivers/watchdog/w83697hf_wdt.c   |    4 +---
 drivers/watchdog/w83877f_wdt.c    |    4 +---
 drivers/watchdog/w83977f_wdt.c    |    4 +---
 drivers/watchdog/wafer5823wdt.c   |    4 +---
 drivers/watchdog/wdt977.c         |    4 +---
 drivers/watchdog/wdt_pci.c        |    3 +--
 19 files changed, 20 insertions(+), 57 deletions(-)

--- a/drivers/watchdog/alim1535_wdt.c
+++ b/drivers/watchdog/alim1535_wdt.c
@@ -31,7 +31,7 @@ static unsigned long ali_is_open;
 static char ali_expect_release;
 static struct pci_dev *ali_pci;
 static u32 ali_timeout_bits; /* stores the computed timeout */
-static spinlock_t ali_lock; /* Guards the hardware */
+static DEFINE_SPINLOCK(ali_lock); /* Guards the hardware */

 /* module parameters */
 static int timeout = WATCHDOG_TIMEOUT;
@@ -398,8 +398,6 @@ static int __init watchdog_init(void)
 {
  int ret;
```

```
- spin_lock_init(&ali_lock);
-
  /* Check whether or not the hardware watchdog is there */
  if (ali_find_watchdog() != 0) {
   return -ENODEV;
--- a/drivers/watchdog/davinci_wdt.c
+++ b/drivers/watchdog/davinci_wdt.c
@@ -61,7 +61,7 @@

 static int heartbeat = DEFAULT_HEARTBEAT;

-static spinlock_t io_lock;
+static DEFINE_SPINLOCK(io_lock);
 static unsigned long wdt_status;
 #define WDT_IN_USE        0
 #define WDT_OK_TO_CLOSE   1
@@ -200,8 +200,6 @@ static int davinci_wdt_probe(struct platform_device *pdev)
  int ret = 0, size;
  struct resource *res;

- spin_lock_init(&io_lock);
-
  if (heartbeat < 1 || heartbeat > MAX_HEARTBEAT)
   heartbeat = DEFAULT_HEARTBEAT;

--- a/drivers/watchdog/i6300esb.c
+++ b/drivers/watchdog/i6300esb.c
@@ -77,7 +77,7 @@

 /* internal variables */
 static void __iomem *BASEADDR;
-static spinlock_t esb_lock; /* Guards the hardware */
+static DEFINE_SPINLOCK(esb_lock); /* Guards the hardware */
 static unsigned long timer_alive;
 static struct pci_dev *esb_pci;
 static unsigned short triggered; /* The status of the watchdog upon boot */
@@ -456,8 +456,6 @@ static int __init watchdog_init (void)
 {
        int ret;

-       spin_lock_init(&esb_lock);
-
        /* Check whether or not the hardware watchdog is there */
        if (!esb_getdevice () || esb_pci == NULL)
                return -ENODEV;
--- a/drivers/watchdog/ib700wdt.c
+++ b/drivers/watchdog/ib700wdt.c
@@ -48,7 +48,7 @@
```

```
 static struct platform_device *ibwdt_platform_device;
 static unsigned long ibwdt_is_open;
-static spinlock_t ibwdt_lock;
+static DEFINE_SPINLOCK(ibwdt_lock);
 static char expect_close;

 /* Module information */
@@ -308,8 +308,6 @@ static int __devinit ibwdt_probe(struct platform_device *dev)
 {
  int res;

- spin_lock_init(&ibwdt_lock);
-
 #if WDT_START != WDT_STOP
  if (!request_region(WDT_STOP, 1, "IB700 WDT")) {
   printk (KERN_ERR PFX "STOP method I/O %X is not available.\n", WDT_STOP);
--- a/drivers/watchdog/machzwd.c
+++ b/drivers/watchdog/machzwd.c
@@ -123,8 +123,8 @@ static void zf_ping(unsigned long data);
 static int zf_action = GEN_RESET;
 static unsigned long zf_is_open;
 static char zf_expect_close;
-static spinlock_t zf_lock;
-static spinlock_t zf_port_lock;
+static DEFINE_SPINLOCK(zf_lock);
+static DEFINE_SPINLOCK(zf_port_lock);
 static DEFINE_TIMER(zf_timer, zf_ping, 0, 0);
 static unsigned long next_heartbeat = 0;

@@ -438,9 +438,6 @@ static int __init zf_init(void)

  zf_show_action(action);

- spin_lock_init(&zf_lock);
- spin_lock_init(&zf_port_lock);
-
  if(!request_region(ZF_IOBASE, 3, "MachZ ZFL WDT")){
   printk(KERN_ERR "cannot reserve I/O ports at %d\n",
        ZF_IOBASE);
--- a/drivers/watchdog/mpc83xx_wdt.c
+++ b/drivers/watchdog/mpc83xx_wdt.c
@@ -56,7 +56,7 @@ static int prescale = 1;
 static unsigned int timeout_sec;

 static unsigned long wdt_is_open;
-static spinlock_t wdt_spinlock;
+static DEFINE_SPINLOCK(wdt_spinlock);
```

```
 static void mpc83xx_wdt_keepalive(void)
 {
@@ -185,9 +185,6 @@ static int __devinit mpc83xx_wdt_probe(struct platform_device *dev)
  printk(KERN_INFO "WDT driver for MPC83xx initialized. "
   "mode:%s timeout=%d (%d seconds)\n",
   reset ? "reset":"interrupt", timeout, timeout_sec);
-
- spin_lock_init(&wdt_spinlock);
-
  return 0;

 err_unmap:
--- a/drivers/watchdog/pc87413_wdt.c
+++ b/drivers/watchdog/pc87413_wdt.c
@@ -61,7 +61,7 @@ static unsigned long timer_enabled = 0;  /* is the timer enabled? */

 static char expect_close;               /* is the close expected? */

-static spinlock_t io_lock;              /* to guard the watchdog from io races */
+static DEFINE_SPINLOCK(io_lock);/* to guard the watchdog from io races */

 static int nowayout = WATCHDOG_NOWAYOUT;

@@ -561,8 +561,6 @@ static int __init pc87413_init(void)
 {
  int ret;

- spin_lock_init(&io_lock);
-
  printk(KERN_INFO PFX "Version " VERSION " at io 0x%X\n", WDT_INDEX_IO_PORT);

  /* request_region(io, 2, "pc87413"); */
--- a/drivers/watchdog/pnx4008_wdt.c
+++ b/drivers/watchdog/pnx4008_wdt.c
@@ -80,7 +80,7 @@
 static int nowayout = WATCHDOG_NOWAYOUT;
 static int heartbeat = DEFAULT_HEARTBEAT;

-static spinlock_t io_lock;
+static DEFINE_SPINLOCK(io_lock);
 static unsigned long wdt_status;
 #define WDT_IN_USE        0
 #define WDT_OK_TO_CLOSE   1
@@ -254,8 +254,6 @@ static int pnx4008_wdt_probe(struct platform_device *pdev)
  int ret = 0, size;
  struct resource *res;
```

```
- spin_lock_init(&io_lock);
-
  if (heartbeat < 1 || heartbeat > MAX_HEARTBEAT)
   heartbeat = DEFAULT_HEARTBEAT;

--- a/drivers/watchdog/sbc8360.c
+++ b/drivers/watchdog/sbc8360.c
@@ -54,7 +54,7 @@
 #include <asm/system.h>

 static unsigned long sbc8360_is_open;
-static spinlock_t sbc8360_lock;
+static DEFINE_SPINLOCK(sbc8360_lock);
 static char expect_close;

 #define PFX "sbc8360: "
@@ -359,7 +359,6 @@ static int __init sbc8360_init(void)
   goto out_noreboot;
  }

- spin_lock_init(&sbc8360_lock);
  res = misc_register(&sbc8360_miscdev);
  if (res) {
   printk(KERN_ERR PFX "failed to register misc device\n");
--- a/drivers/watchdog/sc1200wdt.c
+++ b/drivers/watchdog/sc1200wdt.c
@@ -74,7 +74,7 @@ static int io = -1;
 static int io_len = 2;  /* for non plug and play */
 static struct semaphore open_sem;
 static char expect_close;
-static spinlock_t sc1200wdt_lock; /* io port access serialisation */
+static DEFINE_SPINLOCK(sc1200wdt_lock); /* io port access serialisation */

 #if defined CONFIG_PNP
 static int isapnp = 1;
@@ -375,7 +375,6 @@ static int __init sc1200wdt_init(void)

  printk("%s\n", banner);

- spin_lock_init(&sc1200wdt_lock);
  sema_init(&open_sem, 1);

 #if defined CONFIG_PNP
--- a/drivers/watchdog/sc520_wdt.c
+++ b/drivers/watchdog/sc520_wdt.c
@@ -125,7 +125,7 @@ static DEFINE_TIMER(timer, wdt_timer_ping, 0, 0);
 static unsigned long next_heartbeat;
 static unsigned long wdt_is_open;
```

```
 static char wdt_expect_close;
-static spinlock_t wdt_spinlock;
+static DEFINE_SPINLOCK(wdt_spinlock);

 /*
  * Whack the dog
@@ -383,8 +383,6 @@ static int __init sc520_wdt_init(void)
 {
  int rc = -EBUSY;

- spin_lock_init(&wdt_spinlock);
-
  /* Check that the timeout value is within it's range ; if not reset to the default */
  if (wdt_set_heartbeat(timeout)) {
   wdt_set_heartbeat(WATCHDOG_TIMEOUT);
--- a/drivers/watchdog/smsc37b787_wdt.c
+++ b/drivers/watchdog/smsc37b787_wdt.c
@@ -83,7 +83,7 @@ static unsigned long timer_enabled = 0;   /* is the timer enabled? */

 static char expect_close;       /* is the close expected? */

-static spinlock_t io_lock; /* to guard the watchdog from io races */
+static DEFINE_SPINLOCK(io_lock);/* to guard the watchdog from io races */

 static int nowayout = WATCHDOG_NOWAYOUT;

@@ -540,8 +540,6 @@ static int __init wb_smsc_wdt_init(void)
 {
  int ret;

- spin_lock_init(&io_lock);
-
  printk("SMsC 37B787 watchdog component driver " VERSION " initialising...\n");

  if (!request_region(IOPORT, IOPORT_SIZE, "SMsC 37B787 watchdog")) {
--- a/drivers/watchdog/w83627hf_wdt.c
+++ b/drivers/watchdog/w83627hf_wdt.c
@@ -48,7 +48,7 @@

 static unsigned long wdt_is_open;
 static char expect_close;
-static spinlock_t io_lock;
+static DEFINE_SPINLOCK(io_lock);

 /* You must set this - there is no sane way to probe for this board. */
 static int wdt_io = 0x2E;
@@ -328,8 +328,6 @@ wdt_init(void)
 {
```

```
  int ret;

- spin_lock_init(&io_lock);
-
  printk(KERN_INFO "WDT driver for the Winbond(TM) W83627HF/THF/HG Super I/O chip
initialising.\n");

  if (wdt_set_heartbeat(timeout)) {
--- a/drivers/watchdog/w83697hf_wdt.c
+++ b/drivers/watchdog/w83697hf_wdt.c
@@ -47,7 +47,7 @@

 static unsigned long wdt_is_open;
 static char expect_close;
-static spinlock_t io_lock;
+static DEFINE_SPINLOCK(io_lock);

 /* You must set this - there is no sane way to probe for this board. */
 static int wdt_io = 0x2e;
@@ -376,8 +376,6 @@ wdt_init(void)
 {
  int ret, i, found = 0;

- spin_lock_init(&io_lock);
-
  printk (KERN_INFO PFX "WDT driver for W83697HF/HG initializing\n");

  if (wdt_io == 0) {
--- a/drivers/watchdog/w83877f_wdt.c
+++ b/drivers/watchdog/w83877f_wdt.c
@@ -94,7 +94,7 @@ static DEFINE_TIMER(timer, wdt_timer_ping, 0, 0);
 static unsigned long next_heartbeat;
 static unsigned long wdt_is_open;
 static char wdt_expect_close;
-static spinlock_t wdt_spinlock;
+static DEFINE_SPINLOCK(wdt_spinlock);

 /*
  * Whack the dog
@@ -350,8 +350,6 @@ static int __init w83877f_wdt_init(void)
 {
  int rc = -EBUSY;

- spin_lock_init(&wdt_spinlock);
-
  if(timeout < 1 || timeout > 3600) /* arbitrary upper limit */
  {
   timeout = WATCHDOG_TIMEOUT;
```

```
--- a/drivers/watchdog/w83977f_wdt.c
+++ b/drivers/watchdog/w83977f_wdt.c
@@ -50,7 +50,7 @@ static int timeoutW;   /* timeout in watchdog counter units */
 static unsigned long timer_alive;
 static int testmode;
 static char expect_close;
-static spinlock_t spinlock;
+static DEFINE_SPINLOCK(spinlock);

 module_param(timeout, int, 0);
 MODULE_PARM_DESC(timeout,"Watchdog timeout in seconds (15..7635), default="
__MODULE_STRING(DEFAULT_TIMEOUT) ")");
@@ -476,8 +476,6 @@ static int __init w83977f_wdt_init(void)

        printk(KERN_INFO PFX DRIVER_VERSION);

- spin_lock_init(&spinlock);
-
  /*
   * Check that the timeout value is within it's range ;
   * if not reset to the default
--- a/drivers/watchdog/wafer5823wdt.c
+++ b/drivers/watchdog/wafer5823wdt.c
@@ -45,7 +45,7 @@

 static unsigned long wafwdt_is_open;
 static char expect_close;
-static spinlock_t wafwdt_lock;
+static DEFINE_SPINLOCK(wafwdt_lock);

 /*
  * You must set these - there is no sane way to probe for this board.
@@ -252,8 +252,6 @@ static int __init wafwdt_init(void)

  printk(KERN_INFO "WDT driver for Wafer 5823 single board computer initialising.\n");

- spin_lock_init(&wafwdt_lock);
-
  if (timeout < 1 || timeout > 255) {
   timeout = WD_TIMO;
   printk (KERN_INFO PFX "timeout value must be 1<=x<=255, using %d\n",
--- a/drivers/watchdog/wdt977.c
+++ b/drivers/watchdog/wdt977.c
@@ -59,7 +59,7 @@ static int timeoutM;    /* timeout in minutes */
 static unsigned long timer_alive;
 static int testmode;
 static char expect_close;
-static spinlock_t spinlock;
```

```
+static DEFINE_SPINLOCK(spinlock);

 module_param(timeout, int, 0);
 MODULE_PARM_DESC(timeout,"Watchdog timeout in seconds (60..15300), default="
__MODULE_STRING(DEFAULT_TIMEOUT) ")");
@@ -448,8 +448,6 @@ static int __init wd977_init(void)

  printk(KERN_INFO PFX DRIVER_VERSION);

- spin_lock_init(&spinlock);
-
  /* Check that the timeout value is within it's range ; if not reset to the default */
  if (wdt977_set_timeout(timeout))
  {
--- a/drivers/watchdog/wdt_pci.c
+++ b/drivers/watchdog/wdt_pci.c
@@ -74,7 +74,7 @@
 static int dev_count;

 static struct semaphore open_sem;
-static spinlock_t wdtpci_lock;
+static DEFINE_SPINLOCK(wdtpci_lock);
 static char expect_close;

 static int io;
@@ -606,7 +606,6 @@ static int __devinit wdtpci_init_one (struct pci_dev *dev,
  }

  sema_init(&open_sem, 1);
- spin_lock_init(&wdtpci_lock);

  irq = dev->irq;
  io = pci_resource_start (dev, 2);
```