Subject: [PATCH 1/3] Introduce cpuid_on_cpu() and cpuid_eax_on_cpu()
Posted by Alexey Dobriyan on Mon, 02 Apr 2007 11:32:05 GMT
View Forum Message <> Reply to Message

They will be used by cpuid driver and powernow-k8 cpufreq driver.

With these changes powernow-k8 driver could run correctly on OpenVZ kernels
with virtual cpus enabled (SCHED_VCPU).

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

```
 arch/i386/lib/Makefile         |    2 -
 arch/i386/lib/cpuid-on-cpu.c   |   67 ++++++++++++++++++++++++++++++++++++++++
 arch/x86_64/lib/Makefile       |    2 -
 arch/x86_64/lib/cpuid-on-cpu.c |    1
 include/asm-i386/processor.h   |   15 +++++++++
 include/asm-x86_64/msr.h       |   13 +++++++
 6 files changed, 98 insertions(+), 2 deletions(-)

--- a/arch/i386/lib/Makefile
+++ b/arch/i386/lib/Makefile
@@ -8,4 +8,4 @@ lib-y = checksum.o delay.o usercopy.o getuser.o putuser.o memcpy.o strstr.o
\

 lib-$(CONFIG_X86_USE_3DNOW) += mmx.o

-obj-$(CONFIG_SMP) += msr-on-cpu.o
+obj-$(CONFIG_SMP) += cpuid-on-cpu.o msr-on-cpu.o
--- a/arch/x86_64/lib/Makefile
+++ b/arch/x86_64/lib/Makefile
@@ -5,7 +5,7 @@
 CFLAGS_csum-partial.o := -funroll-loops

 obj-y := io.o iomap_copy.o
-obj-$(CONFIG_SMP) += msr-on-cpu.o
+obj-$(CONFIG_SMP) += cpuid-on-cpu.o msr-on-cpu.o

 lib-y := csum-partial.o csum-copy.o csum-wrappers.o delay.o \
  usercopy.o getuser.o putuser.o  \
--- a/include/asm-i386/processor.h
+++ b/include/asm-i386/processor.h
@@ -643,6 +643,21 @@ static inline unsigned int cpuid_edx(unsigned int op)
  return edx;
 }

+#ifdef CONFIG_SMP
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx);
```

```
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op);
+#else
+static inline void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32
*edx)
+{
+ cpuid(op, eax, ebx, ecx, edx);
+}
+
+static inline u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ return cpuid_eax(op);
+}
+#endif
+
 /* generic versions from gas */
 #define GENERIC_NOP1 ".byte 0x90\n"
 #define GENERIC_NOP2     ".byte 0x89,0xf6\n"
--- a/include/asm-x86_64/msr.h
+++ b/include/asm-x86_64/msr.h
@@ -163,6 +163,9 @@ static inline unsigned int cpuid_edx(unsigned int op)
 #ifdef CONFIG_SMP
 void rdmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 *l, u32 *h);
 void wrmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 l, u32 h);
+
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx);
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op);
 #else  /*  CONFIG_SMP  */
 static inline void rdmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 *l, u32 *h)
 {
@@ -172,6 +175,16 @@ static inline void wrmsr_on_cpu(unsigned int cpu, u32 msr_no, u32 l,
u32 h)
 {
  wrmsr(msr_no, l, h);
 }
+
+static inline void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32
*edx)
+{
+ cpuid(op, eax, ebx, ecx, edx);
+}
+
+static inline u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ return cpuid_eax(op);
+}
 #endif  /*  CONFIG_SMP  */

 #endif
```

```
--- /dev/null
+++ b/arch/i386/lib/cpuid-on-cpu.c
@@ -0,0 +1,67 @@
+#include <linux/module.h>
+#include <linux/preempt.h>
+#include <linux/smp.h>
+#include <linux/types.h>
+
+struct cpuid_info {
+ u32 op;
+ u32 eax, ebx, ecx, edx;
+};
+
+static void __cpuid_on_cpu(void *info)
+{
+ struct cpuid_info *rv = info;
+
+ cpuid(rv->op, &rv->eax, &rv->ebx, &rv->ecx, &rv->edx);
+}
+
+void cpuid_on_cpu(unsigned int cpu, u32 op, u32 *eax, u32 *ebx, u32 *ecx, u32 *edx)
+{
+ preempt_disable();
+ if (smp_processor_id() == cpu)
+  cpuid(op, eax, ebx, ecx, edx);
+ else {
+  struct cpuid_info rv;
+
+  rv.op = op;
+  smp_call_function_single(cpu, __cpuid_on_cpu, &rv, 0, 1);
+  *eax = rv.eax;
+  *ebx = rv.ebx;
+  *ecx = rv.ecx;
+  *edx = rv.edx;
+ }
+ preempt_enable();
+}
+
+struct cpuid_eax_info {
+ u32 op;
+ u32 eax;
+};
+
+static void __cpuid_eax_on_cpu(void *info)
+{
+ struct cpuid_info *rv = info;
+
+ rv->eax = cpuid_eax(rv->op);
```

```
+}
+
+u32 cpuid_eax_on_cpu(unsigned int cpu, u32 op)
+{
+ u32 ret;
+
+ preempt_disable();
+ if (smp_processor_id() == cpu)
+  ret = cpuid_eax(op);
+ else {
+  struct cpuid_eax_info rv;
+
+  rv.op = op;
+  smp_call_function_single(cpu, __cpuid_eax_on_cpu, &rv, 0, 1);
+  ret = rv.eax;
+ }
+ preempt_enable();
+ return ret;
+}
+
+EXPORT_SYMBOL(cpuid_on_cpu);
+EXPORT_SYMBOL(cpuid_eax_on_cpu);
--- /dev/null
+++ b/arch/x86_64/lib/cpuid-on-cpu.c
@@ -0,0 +1 @@
+#include "../../i386/lib/cpuid-on-cpu.c"
```