```
/*
 * (C) Copyright 2002
 * Sysgo Real-Time Solutions, GmbH <www.elinos.com>
 * Marius Groeger <mgroeger@sysgo.de>
 *
 * (C) Copyright 2002
 * Gary Jennejohn, DENX Software Engineering, <gj@denx.de>
 *
 * See file CREDITS for list of people who contributed to this
 * project.
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License as
 * published by the Free Software Foundation; either version 2 of
 * the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston,
 * MA 02111-1307 USA
 */

/*
 * CPU specific code
 */

#include <common.h>
#include <command.h>
#include <arm920t.h>

/* read co-processor 15, register #1 (control register) */
static unsigned long read_p15_c1 (void)
{
    unsigned long value;

    __asm__ __volatile__(
        "mrc     p15, 0, %0, c1, c0, 0   @ read control reg\n"
        : "=r" (value)
        :
        : "memory");

#ifdef MMU_DEBUG
    printf ("p15/c1 is = %08lx\n", value);
#endif
    return value;
}

/* write to co-processor 15, register #1 (control register) */
static void write_p15_c1 (unsigned long value)
{
#ifdef MMU_DEBUG
    printf ("write %08lx to p15/c1\n", value);
#endif
    __asm__ __volatile__(
        "mcr     p15, 0, %0, c1, c0, 0   @ write it back\n"
        :
        : "r" (value)
        : "memory");

    read_p15_c1 ();
}

static void cp_delay (void)
{
    volatile int i;

    /* copro seems to need some delay between reading and writing */
    for (i = 0; i < 100; i++);
}

// coprocessor register 세팅 부분을 볼 것. 특히 MMU쪽
/* See also ARM Ref. Man. */
#define C1_MMU      (1<<0)      /* mmu off/on */
#define C1_ALIGN    (1<<1)      /* alignment faults off/on */
#define C1_DC       (1<<2)      /* dcache off/on */
#define C1_BIG_ENDIAN   (1<<7)  /* big endian off/on */
#define C1_SYS_PROT (1<<8)      /* system protection */
```

```
 83 #define C1_ROM_PROT (1<<9)      /* ROM protection */
 84 #define C1_IC       (1<<12)     /* icache off/on */
 85 #define C1_HIGH_VECTORS (1<<13) /* location of vectors: low/high addresses */
 86 #define RESERVED_1  (0xf << 3)  /* must be 111b for R/W */
 87
 88 // lib_arm/board.c의 start_armboot의 초기화 함수 호출시 불림
 89 int cpu_init (void)
 90 {
 91     /*
 92      * setup up stack if necessary
 93      */
 94 #ifdef CONFIG_USE_IRQ // include/configs/smdk2410.h에 undefine되어 있음
 95     IRQ_STACK_START = _armboot_end +
 96             CONFIG_STACKSIZE + CONFIG_STACKSIZE_IRQ - 4;
 97     FIQ_STACK_START = IRQ_STACK_START + CONFIG_STACKSIZE_FIQ;
 98     _armboot_real_end = FIQ_STACK_START + 4;
 99 #else
100     // cpu/arm920t/start.S 에 선언만 되어 있고 실제로는 여기서 세팅한다.
101         // armboot의 코드와 스택크기를 더한값 다음번지이다.
102     _armboot_real_end = _armboot_end + CONFIG_STACKSIZE;
103 #endif  /* CONFIG_USE_IRQ */
104     return (0);
105 }
106
107 int cleanup_before_linux (void)
108 {
109     /*
110      * this function is called just before we call linux
111      * it prepares the processor for linux
112      *
113      * we turn off caches etc ...
114      */
115
116     unsigned long i;
117
118     disable_interrupts ();
119
120     /* turn off I/D-cache */
121     asm ("mrc p15, 0, %0, c1, c0, 0":"=r" (i));
122     i &= ~(C1_DC | C1_IC);
123     asm ("mcr p15, 0, %0, c1, c0, 0": :"r" (i));
124
125     /* flush I/D-cache */
126     i = 0;
127     asm ("mcr p15, 0, %0, c7, c7, 0": :"r" (i));
128     return (0);
129 }
130
131 /* ghcstop */
132 /* 인터럽트 disable시키고, 해당 cache들과 mmu의 TLB를 flush시키고
133 등등...을 한 다음에 pc에 0x00000000번지를 대입해서 다시 reset하게 한다. */
134 int do_reset (cmd_tbl_t *cmdtp, int flag, int argc, char *argv[])
135 {
136     extern void reset_cpu (ulong addr);
137
138     disable_interrupts ();
139     reset_cpu (0); // 여기서 0은 어드레스다, 이게 r0로 들어가서 start.S의 함수에서 pc로 들어간다.
140     /*NOTREACHED*/
141     return (0);
142 }
143
144 void icache_enable (void)
145 {
146     ulong reg;
147
148     reg = read_p15_c1 ();  // coprocessor 15, register 1을 읽고
149     cp_delay ();           // 약간의 delay를 주고
150     write_p15_c1 (reg | C1_IC); // 위에 정의된 값으로 세팅한다. (1<<12), 정확한 값 세팅은 메뉴얼을 볼 것
151 }
152
153 void icache_disable (void)
154 {
155     ulong reg;
156
157     reg = read_p15_c1 ();
158     cp_delay ();
159     write_p15_c1 (reg & ~C1_IC);
160 }
161
162 int icache_status (void)
163 {
164     return (read_p15_c1 () & C1_IC) != 0;
```

```
165 }
166
167 #ifdef USE_920T_MMU
168 /* It makes no sense to use the dcache if the MMU is not enabled */
169 void dcache_enable (void)
170 {
171     ulong reg;
172
173     reg = read_p15_c1 ();
174     cp_delay ();
175     write_p15_c1 (reg | C1_DC);
176 }
177
178 void dcache_disable (void)
179 {
180     ulong reg;
181
182     reg = read_p15_c1 ();
183     cp_delay ();
184     reg &= ~C1_DC;
185     write_p15_c1 (reg);
186 }
187
188 int dcache_status (void)
189 {
190     return (read_p15_c1 () & C1_DC) != 0;
191 }
192 #endif
193
```