```
/*
 * (C) Copyright 2002
 * Sysgo Real-Time Solutions, GmbH <www.elinos.com>
 * Marius Groeger <mgroeger@sysgo.de>
 *
 * (C) Copyright 2002
 * David Mueller, ELSOFT AG, <d.mueller@elsoft.ch>
 *
 * 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
 */

#include <common.h>
#include <s3c2410.h>

/* ------------------------------------------------------------------------- */

#define FCLK_SPEED 1

/* 메뉴얼 7-19, 7-20을 볼 것 */
#if FCLK_SPEED==0       /* Fout = 203MHz, Fin = 12MHz for Audio */
#define M_MDIV  0xC3
#define M_PDIV  0x4
#define M_SDIV  0x1
#elif FCLK_SPEED==1     /* Fout = 202.8MHz */
#define M_MDIV  0xA1
#define M_PDIV  0x3
#define M_SDIV  0x1
#endif

#define USB_CLOCK 1

#if USB_CLOCK==0
#define U_M_MDIV    0xA1
#define U_M_PDIV    0x3
#define U_M_SDIV    0x1
#elif USB_CLOCK==1 /* 이상한 값이다. 메뉴얼과는 틀리다. 이 값으로 계산하면 51.6M가 나온다 usb clock이 */
#define U_M_MDIV    0x48
#define U_M_PDIV    0x3
#define U_M_SDIV    0x2
#endif

static inline void delay (unsigned long loops)
{
    __asm__ volatile ("1:\n"
      "subs %0, %1, #1\n"
      "bne 1b":"=r" (loops):"0" (loops));
}

/*
 * Miscellaneous platform dependent initialisations
 */

int board_init (void)
{
    /*
     * include/asm-arm/global_data.h
     * #define DECLARE_GLOBAL_DATA_PTR     register gd_t *gd asm ("r8")
     * gd_t형 포인터인 gd를 r8에다 넣는 register변수 지정을 한다. 당근 stack변수니깐
     * 상관은 없겠다...함수 호출해도
     */
    DECLARE_GLOBAL_DATA_PTR;

        /*
        include/s3c24x0.h

        typedef volatile u8     S3C24X0_REG8;
```

```
 83      typedef volatile u16    S3C24X0_REG16;
 84      typedef volatile u32    S3C24X0_REG32;
 85
 86          CLOCK & POWER MANAGEMENT (see S3C2400 manual chapter 6)
 87                                (see S3C2410 manual chapter 7)
 88      typedef struct {
 89          S3C24X0_REG32   LOCKTIME;
 90          S3C24X0_REG32   MPLLCON;
 91          S3C24X0_REG32   UPLLCON;
 92          S3C24X0_REG32   CLKCON;
 93          S3C24X0_REG32   CLKSLOW;
 94          S3C24X0_REG32   CLKDIVN;
 95      }  S3C24X0_CLOCK_POWER;
 96
 97          include/s3c2410.h의 inline함수인 S3C24X0_GetBase_CLOCK_POWER()를 호출해서
 98          clock register들의 base address를 가지고 와서 해당 register 스트럭쳐로
 99          매핑한다.
100         */
101      S3C24X0_CLOCK_POWER * const clk_power = S3C24X0_GetBase_CLOCK_POWER();
102
103          /*
104          include/s3c24x0.h
105          I/O PORT (see manual chapter 9)
106      typedef struct {
107          #ifdef CONFIG_S3C2410
108          S3C24X0_REG32   GPACON;
109          S3C24X0_REG32   GPADAT;
110          S3C24X0_REG32   res1[2];
111          S3C24X0_REG32   GPBCON;
112          S3C24X0_REG32   GPBDAT;
113          S3C24X0_REG32   GPBUP;
114          S3C24X0_REG32   res2;
115          S3C24X0_REG32   GPCCON;
116          S3C24X0_REG32   GPCDAT;
117          S3C24X0_REG32   GPCUP;
118          S3C24X0_REG32   res3;
119          S3C24X0_REG32   GPDCON;
120          S3C24X0_REG32   GPDDAT;
121          S3C24X0_REG32   GPDUP;
122          S3C24X0_REG32   res4;
123          S3C24X0_REG32   GPECON;
124          S3C24X0_REG32   GPEDAT;
125          S3C24X0_REG32   GPEUP;
126          S3C24X0_REG32   res5;
127          S3C24X0_REG32   GPFCON;
128          S3C24X0_REG32   GPFDAT;
129          S3C24X0_REG32   GPFUP;
130          S3C24X0_REG32   res6;
131          S3C24X0_REG32   GPGCON;
132          S3C24X0_REG32   GPGDAT;
133          S3C24X0_REG32   GPGUP;
134          S3C24X0_REG32   res7;
135          S3C24X0_REG32   GPHCON;
136          S3C24X0_REG32   GPHDAT;
137          S3C24X0_REG32   GPHUP;
138          S3C24X0_REG32   res8;
139
140          S3C24X0_REG32   MISCCR;
141          S3C24X0_REG32   DCLKCON;
142          S3C24X0_REG32   EXTINT0;
143          S3C24X0_REG32   EXTINT1;
144          S3C24X0_REG32   EXTINT2;
145          S3C24X0_REG32   EINTFLT0;
146          S3C24X0_REG32   EINTFLT1;
147          S3C24X0_REG32   EINTFLT2;
148          S3C24X0_REG32   EINTFLT3;
149          S3C24X0_REG32   EINTMASK;
150          S3C24X0_REG32   EINTPEND;
151          S3C24X0_REG32   GSTATUS0;
152          S3C24X0_REG32   GSTATUS1;
153          S3C24X0_REG32   GSTATUS2;
154          S3C24X0_REG32   GSTATUS3;
155          S3C24X0_REG32   GSTATUS4;
156          #endif
157      }  S3C24X0_GPIO;
158
159          include/s3c2410.h의 inline함수인 S3C24X0_GetBase_GPIO()를 호출해서
160          gpio register들의 base address를 가지고 와서 해당 register 스트럭쳐로
161          매핑한다.
162      */
163      S3C24X0_GPIO * const gpio = S3C24X0_GetBase_GPIO();
164
```

```
165     /*
166         to reduce PLL lock time, adjust the LOCKTIME register:
167         pll lock time을 줄이기 위하여 LOCKTIME count register를 조절한다.
168         reset시 default값이랑 꼭같이 집어넣네!
169         */
170     clk_power->LOCKTIME = 0xFFFFFF;
171 
172     /*
173          * configure MPLL
174          위에  선언되어 있음
175          #define M_MDIV 0xA1
176      #define M_PDIV 0x3
177      #define M_SDIV 0x1
178          이것에 대한 설명은 manual의 주석을 볼 것
179          */
180     clk_power->MPLLCON = ((M_MDIV << 12) + (M_PDIV << 4) + M_SDIV);
181 
182     /* some delay between MPLL and UPLL */
183     delay (4000);
184 
185     /*
186          configure UPLL
187          #define U_M_MDIV   0x48
188          #define U_M_PDIV   0x3
189          #define U_M_SDIV   0x2
190 
191          굉장히 이상한 값으로 되어 있다. 원래는 48M가 나와야 되는게 아닌가라는 생각이 드는데
192           여기서는 계산했더니 51.6Mhz가 나오는 것 같다.  메뉴얼에 끄적여 놓은거 볼 것
193          */
194     clk_power->UPLLCON = ((U_M_MDIV << 12) + (U_M_PDIV << 4) + U_M_SDIV);
195 
196     /* some delay between MPLL and UPLL */
197     delay (8000);
198 
199     /* set up the I/O ports */
200     gpio->GPACON = 0x007FFFFF;   // GPA설정, reset value로 지정. manual 9-8
201     gpio->GPBCON = 0x00044555;   // GPB설정, manual 9-9
202     gpio->GPBUP  = 0x000007FF;   // 11개 포트 전부 pull-up disable, manual 9-9
203     gpio->GPCCON = 0xAAAAAAAA;   // 모든 port를 0b10으로 만든다...manual 9-10
204     gpio->GPCUP  = 0x0000FFFF;   // 16개 모든 포트 pull-up disable, manual 9-11
205     gpio->GPDCON = 0xAAAAAAAA;   // 모든 port를 0b10으로 만든다...manual 9-12
206     gpio->GPDUP  = 0x0000FFFF;   // 16개 모든 포트 pull-up disable, manual 9-13
207     gpio->GPECON = 0xAAAAAAAA;   // 모든 port를 0b10으로 만든다...manual 9-14
208     gpio->GPEUP  = 0x0000FFFF;   // 16개 모든 포트 pull-up disable, manual 9-15
209     gpio->GPFCON = 0x000055AA;   // GPF설정. manual 9-16
210     gpio->GPFUP  = 0x000000FF;   // 8개 모든 포트 pull-up disable, manual 9-16
211     gpio->GPGCON = 0xFF95FFBA;   // GPG설정. manual 9-17
212     gpio->GPGUP  = 0x0000FFFF;   // 16개 모든 포트 pull-up disable, manual 9-18
213     gpio->GPHCON = 0x002AFAAA;   // GPH설정. manual 9-19
214     gpio->GPHUP  = 0x000007FF;   // 10개 모든 포트 pull-up disable, manual 9-19
215 
216     /*
217      * arch number of SMDK2410-Board
218      * 2410용 커널의 arch/arm/tools/mach-types란 파일을 보면 된다.
219      * 참고로 s3c2410은 182이고, smdk2410보드는 193이다.
220      */
221     gd->bd->bi_arch_number = 193;
222 
223     /*
224     (ghcstop_caution)
225     adress of boot parameters: 커널로 각종 보드 설정에 대한 정보를 건네줄 번지 지정
226     이건 커널에서 받는 번지와 같게 설정해야 한다.
227     lib_arm/armlinux.c의 do_bootm_linux()의 setup_start_tag()호출하기 전의 설명을
228     볼 것
229     */
230     gd->bd->bi_boot_params = 0x30000100;
231 
232     icache_enable(); // => cpu/arm920t/cpu.c instruction cache enable
233     dcache_enable(); // => cpu/arm920t/cpu.c data cache enable
234 
235     return 0;
236 }
237 
238 /* ghcstop */
239 /*
240  * lib_arm/board.c의 start_armboot()의 초기화 함수 호출시 호출됨
241  *
242  * dram의 start번지와 size를 global data structure와 board description
243  * structure에 세팅해 넣는다.
244  */
245 int dram_init (void)
246 {
```

```
247     /*
248      * include/asm-arm/global_data.h
249      * #define DECLARE_GLOBAL_DATA_PTR     register gd_t *gd asm ("r8")
250      * gd_t형 포인터인 gd를 r8에다 넣는 register변수 지정을 한다. 당근 stack변수니깐
251      * 상관은 없겠다...함수 호출해도
252      */
253     DECLARE_GLOBAL_DATA_PTR;
254
255 /*
256  include/configs/smdk2410.h의 Physical Memory Map부분에 선언
257
258 #define CONFIG_NR_DRAM_BANKS    1          // we have 1 bank of DRAM
259 #define PHYS_SDRAM_1            0x30000000 // SDRAM Bank #1
260 #define PHYS_SDRAM_1_SIZE       0x04000000 // 64 MB
261 */
262     gd->bd->bi_dram[0].start = PHYS_SDRAM_1;
263     gd->bd->bi_dram[0].size = PHYS_SDRAM_1_SIZE;
264
265     return 0;
266 }
267
```