Hammer LED Userspace Application

This is an example program that uses the /dev/mem device node to access the physical memory registers directly.

Usage

To compile the source use your toolchain command line:

```
arm-linux-uclibc-gcc -o ledblink ledblink.c
```

Then transfer the ledblink binary to the hammer file system. Make sure that the permissions on the file are set to executable.

```
chmod 755 ledblink
./ledblink

Usage:  ./ledblink { seconds }
        seconds : number of seconds to blink

./ledblink 20
```

This will cause the onboard LED to blink for 20 seconds.

```
/*
 * ledblink.c: Simple program to blink a led on GPIO F0
 *
 *    Copyright (c) 2008 TinCanTools
 *    David Anders <danders@amltd.com>
 *
 * 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 <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/mman.h>

#define MAP_SIZE 4096UL       /* the size of one page of virtual memory */
#define MAP_MASK (MAP_SIZE - 1)     /* mask used for the one page of virtual memory */

#define GPIOF_CON 0x56000050 /* physical address of the GPIO F control register */
#define GPIOF_DAT 0x56000054 /* physical address of the GPIO F data register */

int main(int argc, char **argv) {
    int fd,count;
    void *map_base, *virt_addr;
```

```
unsigned long read_result, writeval;

if(argc < 2) {
  fprintf(stderr, "\nUsage:\t%s { seconds }\n"
      "\tseconds : number of seconds to blink\n\n",
      argv[0]);
  exit(1);
}

count = strtoul(argv[1], 0, 0);

/* open the memory device file descriptor */
if((fd = open("/dev/mem", O_RDWR | O_SYNC)) == -1)
  return -1;

/* Map one page */
map_base = mmap(0, MAP_SIZE, PROT_READ | PROT_WRITE, MAP_SHARED, fd, GPIOF_CON & ~MAP_MASK);
if(map_base == (void *) -1)
  return -1;

/* set the virtual memory address to the GPIO F control register */
virt_addr = map_base + (GPIOF_CON & MAP_MASK);

/* read the current value for GPIO F control register */
read_result = *((unsigned long *) virt_addr);

writeval = read_result;
writeval &= ~0x03;        /* mask off the first two bits */
writeval |= 0x01;         /* set gpio f0 to output with value of 0x01 */

/* write the modified value to GPIO F0 control register to set GPIO F0 to output */
*((unsigned long *) virt_addr) = writeval;


/* change virtual memory address to GPIO F data register */
virt_addr = map_base + (GPIOF_DAT & MAP_MASK);

for ( count ; count > 0 ; count--) {
```

```
      /* read the current value for GPIO F data register */
      read_result = *((unsigned long *) virt_addr);
      writeval = read_result;

      if ( (read_result&(1<<0)) )
          writeval &= ~(1<<0);      /* set GPIO F0 to a value of 0(low-on) */
      else
          writeval |= (1<<0);       /* set GPIO F0 to a value of 1(high-off) */

      /* write the modified value to the GPIO F0 data register */
      *((unsigned long *) virt_addr) = writeval;
      sleep(1);

    }


    /* read the current value for GPIO F control register */
    read_result = *((unsigned long *) virt_addr);

    /* make sure the LED is off when leaving the program */
    writeval = read_result;
    writeval |= (1<<0);       /* set GPIO F0 to a value of 1(high-off) */

    /* write the modified value to the GPIO F0 data register */
    *((unsigned long *) virt_addr) = writeval;

    /* unmap the virtual memory before exiting the program */
    if(munmap(map_base, MAP_SIZE) == -1)
      return -1;

    close(fd);
    return 0;

}
```
