第8章

10ms以下の制御が求められる用途では厳しいかも…？

ちょこっと実験！ Linuxアプリの応答速度

久保田 英晃，末永 正治

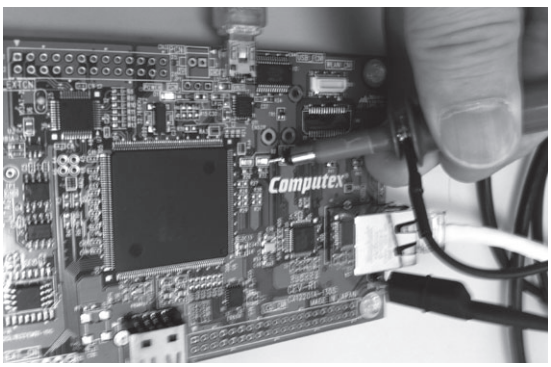

写真1　LEDのON/OFF繰り返し波形をチェックしてLinuxアプリの高速応答性を試す

実験方法

本稿では，Linuxアプリケーションの応答速度について簡単な実験を行いました．

具体的には，RZボードで動作するLinuxアプリケーションでsleep()関数がどのくらい正確に時間を計れるかを実験してみました．

RZボードにはGPIOでつながったLEDが二つあるので，このLEDをON/OFFする時間を計ってみます．

時間の計測にはオシロスコープを使います．

写真1は時間を計測するためにオシロスコープのプローブをLEDにあてているところです．

実験ソフトウェア

リスト1に今回の実験に使ったソースを示します．

LEDは，LED1がGPIOのP7_8，LED2がP7_9に接続されています．ここではLED1を使います．このGPIOはLEDクラスに定義しているので，アプリケーションからアクセスするにはsysfsのLEDクラスを使って簡単にアクセスすることができます．

"/sys/class/leds/led1/brightness"がデバイス(LED)のファイルになります．

LEDの基本的なアクセスは以下の方法で行います．

①LED1アクセス用のsysfsファイルをオープンする

```
fd = open("/sys/class/leds/led1/
     brightness", O_RDWR);
```

②LED1を点灯する

```
write(fd,"1",1);
```

③LED1を消灯する

```
write(fd,"0",1);
```

④LED1の状態を確認する（0なら消灯，1なら点灯）

```
read(fd,&c,1);
printf("%d",c);
```

⑤LED1アクセス用のsysfsファイルをクローズする

```
close(fd)
```

時間を決めてLEDを点灯/消灯するためにusleep()とnanosleep()関数にいろいろな時間を引き数に与えてみました．

リスト1　Linuxの応答速度を測るためのシンプルLED ON/OFFプログラム(test1.c)

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
int main(int argc,char **argv)
{
    int i,fd;
    // nanosleep用引き数
    struct timespec treq, trem;
    // sysfsのLEDクラスをオープンする
    fd = open("/sys/class/leds/led1/brightness",
                                        O_RDWR);
//  treq.tv_sec = (time_t)0;
//  treq.tv_nsec = 1;
    for (i = 0 ; i < 10000000 ;i++ ){
        write(fd,"0",1);
//      nanosleep(&treq, &trem);
//      usleep(100*1000);
        write(fd,"1",1);
//      nanosleep(&treq, &trem);
//      usleep(100*1000);
    }
    close(fd);
    return 0;
}
```