VS1000B/C Battery Monitor

VS1000 "VLSI Solution Ogg Vorbis Player"

Project Code:
Project Name: VS1000

Revision History			
Rev.	**Date**	**Author**	**Description**
1.00	2008-09-19	PO	Initial version.

1 VS1000 Battery Monitor

VS1000 does not have a analog-to-digital converter that could be used for monitoring battery voltage. However, in player mode the voltage of the AVDD regulator can be changed quite freely without it affecting the audio quality (20 steps from 12 to 31). By using a simple comparator circuit the AVDD can act as a changable reference to find out what the battery voltage level is.

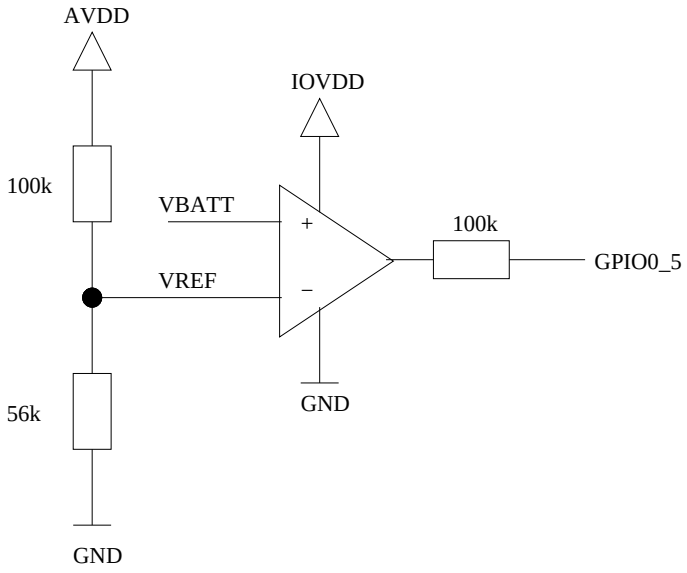

Figure 1.1: Voltage Monitor

The resistor divider in AVDD determines the comparison range.

GPIO0[5] is used in the example, but any other suitable GPIO pin can be used.

The following code performs the voltage measurement. First it puts GPIO0[5] into GPIO mode and assumes it is in input mode. Then the code configures the user voltage set, and waits a bit to allow AVDD to stabilize. After that GPIO0[5] is sampled and AVDD is either raised or lowered to reflect the current battery level.

At the end the old operating mode for GPIO0 is restored, as is the regulator configuration.

The measured battery level is returned.

```
u_int16 CheckBatteryLevel(void) {
    register u_int16 i;
    register u_int16 mode = PERIP(GPIO0_MODE);

    PERIP(GPIO0_MODE) = mode & ~0x20;          /* GPIO0[5] to GPIO mode.*/
    PowerSetVoltages(&voltages[voltCoreUser]); /* Set user voltages. */

    /* Wait a bit... about 200us */
    for (i=0;i<30000;i++)
        PERIP(GPIO0_IDATA);

    if ((PERIP(GPIO0_IDATA) & 0x20)) { /* Check GPIO0[5]. */
        /* VBATT > VREF(=0.36*AVDD) -- Raise VREF and try again. */
        i = voltages[voltAnaUser];
        if (voltages[voltAnaUser] < 31)
            voltages[voltAnaUser]++;
    } else {
        /* VBATT < VREF(=0.36*AVDD) -- Lower VREF. */
        if (voltages[voltAnaUser] > 12)
            voltages[voltAnaUser]--;
        i = voltages[voltAnaUser];
    }
    PERIP(GPIO0_MODE) = mode;
    /* Restore player voltages - this is not strictly needed. */
    PowerSetVoltages(&voltages[voltCorePlayer]);
    return i;
}
```

Note that the return value changes at most by one for each call, so at startup call CheckBatteryLevel() 32 times to initialize your battery level variable.

```
{
    register int i;
    for (i=0;i<32;i++)
        batteryLevel = CheckBatteryLevel();
}
```

Because of the small wait in the battery level detection routine, you should not call the function too often. You can use something like the following routine to call it only once per second.

```
u_int32 batteryCheckTime = 0;
..
    if (batteryCheckTime < ReadTimeCount()) {
        batteryCheckTime = ReadTimeCount() + 1*TIMER_TICKS; /*one second*/
        batteryLevel = CheckBatteryLevel();
    }
```