```
/******************************************************************************
*
* Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reserved.
*
* Permission is hereby granted, free of charge, to any person obtaining a copy
* of this software and associated documentation files (the "Software"), to deal
* in the Software without restriction, including without limitation the rights
* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
* copies of the Software, and to permit persons to whom the Software is
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be included in
* all copies or substantial portions of the Software.
*
* Use of the Software is limited solely to applications:
* (a) running on a Xilinx device, or
* (b) that interact with a Xilinx device through a bus or interconnect.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
* XILINX  BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,
* WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF
* OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
* SOFTWARE.
*
* Except as contained in this notice, the name of the Xilinx shall not be used
* in advertising or otherwise to promote the sale, use or other dealings in
* this Software without prior written authorization from Xilinx.
*
******************************************************************************/

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

//------------------------------------------------------------------------
#include <stdio.h>
#include "platform.h"
#include "xsysmon.h"

#define SYSMON_DEVICE_ID XPAR_SYSMON_0_DEVICE_ID //ID of xadc_wiz_0
#define XSysMon_RawToExtVoltage(AdcData) \
((((float)(AdcData))*(1.0f))/65536.0f)      //(ADC 16bit result)/16/4096 = (ADC 16bit
result)/65536
                                            // voltage value = (ADC 16bit result)/65536 * 1Volt
```

```
static XSysMon SysMonInst; //a sysmon instance
static int SysMonFractionToInt(float FloatNum);

int main()
{

    u32 TempRawData,VccIntRawData,ExtVolRawData;
    float TempData,VccIntData,ExtVolData;
    int xStatus;
    XSysMon_Config *SysMonConfigPtr;
    XSysMon *SysMonInstPtr = &SysMonInst;
    init_platform();
    printf("Hello World\n\r");
    printf("Test1\n\r");
    //---------------------------------------------------------------------- SysMon Initialize
    SysMonConfigPtr = XSysMon_LookupConfig(SYSMON_DEVICE_ID);
    if(SysMonConfigPtr == NULL)     printf("LookupConfig FAILURE\n\r");
    printf("Test2\n\r");
    xStatus = XSysMon_CfgInitialize(SysMonInstPtr, SysMonConfigPtr,SysMonConfigPtr->BaseAddress);
    if(XST_SUCCESS != xStatus)      printf("CfgInitialize FAILED\n\r");
    printf("Test3\n\r");
    //-----------------------------------------------------------------------------------------


    XSysMon_GetStatus(SysMonInstPtr); // Clear the old status
    while(1)
    { //wait until EOS activated
        while ((XSysMon_GetStatus(SysMonInstPtr) & XSM_SR_EOS_MASK) != XSM_SR_EOS_MASK);

        TempRawData = XSysMon_GetAdcData(SysMonInstPtr, XSM_CH_TEMP);//Read the on-chip
Temperature Data
            TempData = XSysMon_RawToTemperature(TempRawData);
            printf("\r\nThe Current Temperature is %0d.%03d Centigrades.\n\r",
            (int)(TempData), SysMonFractionToInt(TempData));

        VccIntRawData = XSysMon_GetAdcData(SysMonInstPtr, XSM_CH_VCCINT); //Read the on-chip
Vccint Data
            VccIntData = XSysMon_RawToVoltage(VccIntRawData);
            printf("The Current VCCINT is %0d.%03d Volts. \n\r",
            (int)(VccIntData),   SysMonFractionToInt(VccIntData));

        ExtVolRawData = XSysMon_GetAdcData(SysMonInstPtr,XSM_CH_AUX_MIN+14);//Read the external
Vaux8 Data
            ExtVolData = XSysMon_RawToExtVoltage(ExtVolRawData);
            printf("The Current Vaux14 is %0d.%03d Volts. \n\r",
            (int)(ExtVolData),  SysMonFractionToInt(ExtVolData));

    }
    return 0;
}
//---------------------------------------------------------------------------------------------
int SysMonFractionToInt(float FloatNum)
{


float Temp;
Temp = FloatNum;
if  (FloatNum  <  0)  {
Temp  =  -(FloatNum);


}
return( ((int)((Temp -(float)((int)Temp)) * (1000.0f))));
}

//-------------------------------------------------------------------------------------------
```