Nano130 Power Down Sample

Application Note for 32-bit NuMicro® Family

Rev.1.00 — Nov.1, 2012

Document Information

Abstract	The Nano130 provides a control to enter low power state called Power-down mode. The high frequency crystal and oscillator input will be disabled to reduce power consumption, while the low frequency crystal and oscillator can be enabled by program control to support peripheral functions (e.g. RTC, Watchdog Timer and LCD) to work in Power-down mode. In this application note, a few sample programs are created to demonstrate the power-down operations. This document describes the implementation method of these sample programs and gives an instruction to measure the power-down current.
Apply to	Nano130KE3BN/NuTiny-EVB-Nano130-LQFP128 Board

Table of Contents

1 Introduction

In NANO130, the CPU operating mode supports **Normal**, **Idle** and **Power-down** modes. From a hardware perspective, the difference among these operating modes is the state of chip clock sources. The chip clock sources include **HXT** (High Frequency Crystal), **LXT** (High Frequency Crystal), **HIRC** (High Frequency Oscillator) and **LIRC** (Low Frequency Oscillator). The **HIRC** and **LIRC** are automatically enabled after powered on. The program can enable or disable these chip clock sources individually.

1.1 Normal Mode

In Normal mode, all chip clock sources keep in the programmed state. The program in Flash memory is continuously executed by CPU.

1.2 Idle Mode

1.2.1 Entry

The CPU enters Idle mode as soon as a **WFI** instruction is executed. The CPU clock is gating so that program stops execution. In Idle mode, all clock sources keep in the programmed state. In other words, the clock source keeps working if it is enabled and remains stopping if it is disabled before CPU enters Idle mode. The peripheral functions, like DMA transfer, Timer Tick and others, may still be active if the acquired clock source is enabled. Usually, the peripheral function generates an interrupt after operation is completed or a selected event is received.

1.2.2 Exit

The CPU clock restarts as soon as an interrupt happens. Consequently the CPU returns to Normal mode and starts executing instructions. The interrupt service routine will be served first, and then jump to the code following the WFI instruction.

1.3 Power-down Mode

The way to enter Power-down mode is similar to entering Idle mode, except that the following control registers should be programmed before executing the WFI instruction.

Power-down Control Register (PWRCTL): 0x50000200

Bit 6

Enable Power-down mode. This bit should be set to 1.

Bit 5

Enable Power-down wake-up interrupt (PDWU). If this bit is set to 1, the PDWU interrupt will be active after CPU returns to Normal mode. If this bit is set to 0, the PDWU interrupt will be inactive after CPU returns to Normal mode.

System Control Register (SCR): 0xE000ED10

Bit 2

Enable Deep Sleep mode. This bit should be set as 1.

1.3.1 Entry

If both **PWRCTL[6]** and **SCR[2]** are set to 1, the CPU will enter Power-down mode as soon as a **WFI** instruction is executed. The chip clock sources **LXT** and **LIRC** keep in the previous state, while HXT and **HIRC** are disabled in this mode. The peripheral functions, RTC, Watchdog Timer and LCD, can keep operating in Power-down mode if it acquires clock from active **LXT** or **LIRC**. Therefore, the program has to enable and select **LXT** or **LIRC** as peripheral function's clock before entering Power-down mode.

1.3.2 Exit

In contrast with Idle mode, only the selected interrupt sources can wake up CPU from Power-down mode. The interrupt sources include **External interrupt 0** and **1**, **wake-up interrupt** generated by **UART**, **GPIO**, **RTC**, **USB**, **SPI**, **Timer**, **Watchdog Timer** and **BOD**. To successfully wake up CPU, the selected interrupt source should be enabled and the corresponding peripheral function should be properly programmed to generate a wake-up interrupt.

2 Sample Code

The power-down samples are created by using Keil MDK version 4.14 and IAR EWARM 6.21. The directory of the project files are listed below.

"\Samples\NUTINY-EVB_NANO130\PWRDWN_DEMO\KEIL";

"\Samples\NUTINY-EVB_NANO130\PWRDWN_RTC_DEMO\KEIL";

"Samples\NUTINY-EVB_NANO130\PWRDWN_LCD_RTC_DEMO\KEIL".

and

"\Samples\NUTINY-EVB_NANO130\PWRDWN_DEMO\IAR";

"\Samples\NUTINY-EVB_NANO130\PWRDWN_RTC_DEMO\IAR";

"Samples\NUTINY-EVB_NANO130\PWRDWN_LCD_RTC_DEMO\IAR".

The following figures show screen snapshots of opening the "***PWRDWN_DEMO***" project files.

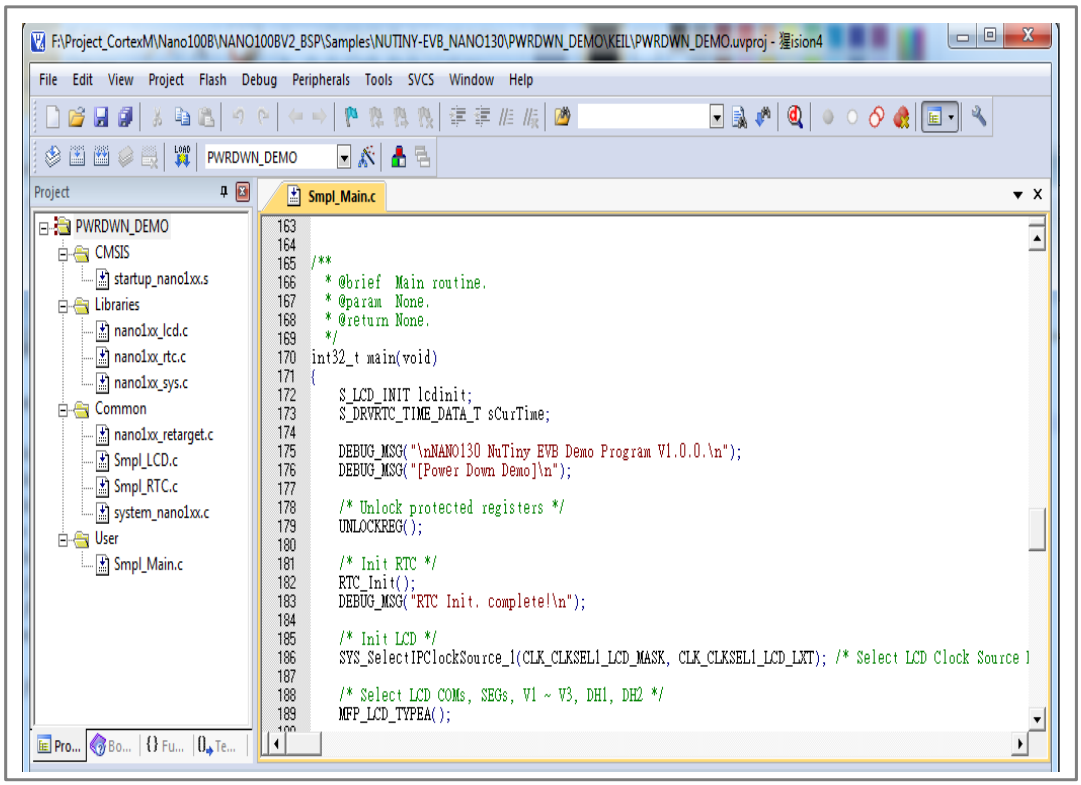

Figure 2-1 "PWRDWN_DEMO" Project Using Keil MDK 4.14

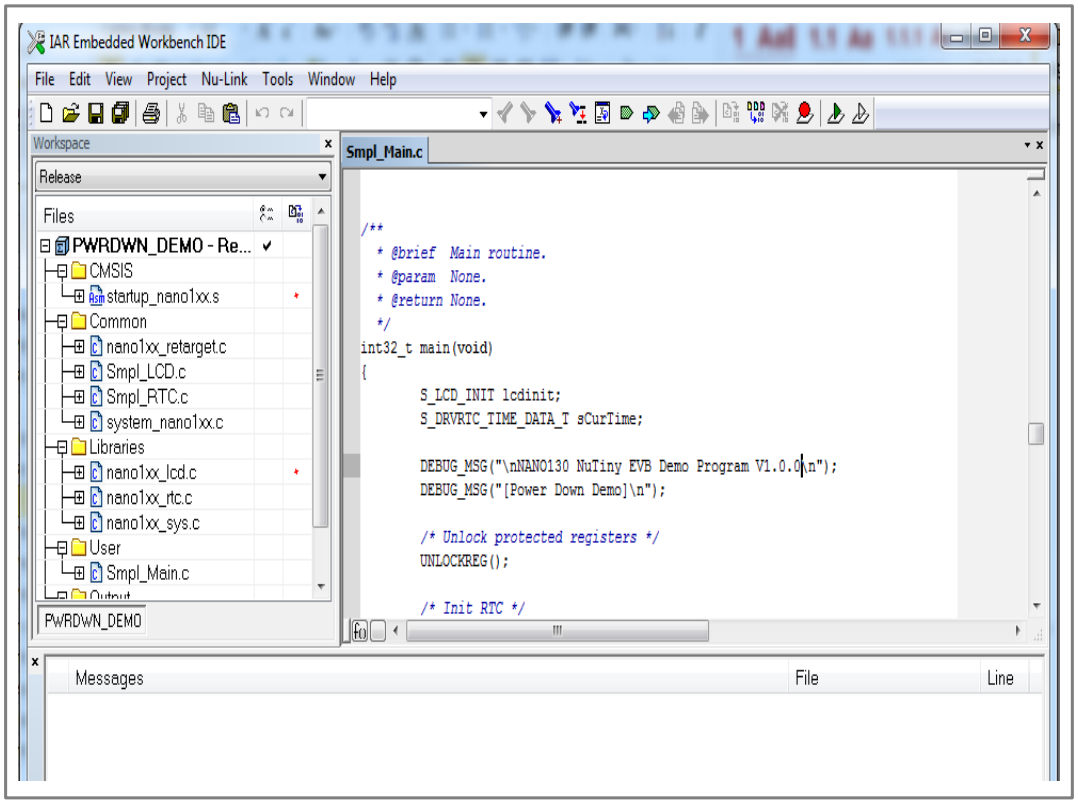

Figure 2-2 "PWRDWN_DEMO" Project Using IAR EWARM 6.21

The sample project to demonstrate power-down operation include "***PWRDWN_DEMO***", "***PWRDWN_RTC_DEMO***" and "***PWRDWN_LCD_RTC_DEMO***". All projects are based on individual sources (*Smpl_Main.c*, *nano1xx_isr.c*) and some common sources (stored in the directory "*\Samples\NUTINY-EVB_NANO130\COMMON*". For each project, the main routine and those subroutines of entering and exiting Power-down are defined in the "***Smpl_Main.c***".

When using Keil MDFK, please rebuild the execution image by selecting "**Project** -> **Rebuild all target files**" from the top menu. Alternatively, select "**Project** -> **Rebuild All**" if using IAR EWARM. The execution image file should be downloaded and run on the **NuTiny-EVB-Nano130-LQFP128** evaluation board.

2.1 PWRDWN_DEMO

The main routine initializes system clock, RTC and LCD functions. After initialization is completed, some messages are shown on the LCD panel and then the program goes to power down. No peripheral function is active during power down, and CPU will not wake up anymore.

Main routine: *main()*

- Set LCD and RTC functions; display messages on LCD panel.

- Call *Enter_PowerDown()*

Power-down subroutine: *Enter_PowerDown()*

- Set all I/O pins to GPIO mode, and enable pull-up for all I/O pins except for PF.2 and PF.3.
- Call ***SYS_SetUpPowerDown()*** to set **PWRCTL[6:5]** to 10b, and set **SCR[2]** to 1b.
- Execute the WFI instruction.

The following figure shows a flow chart of this sample program.

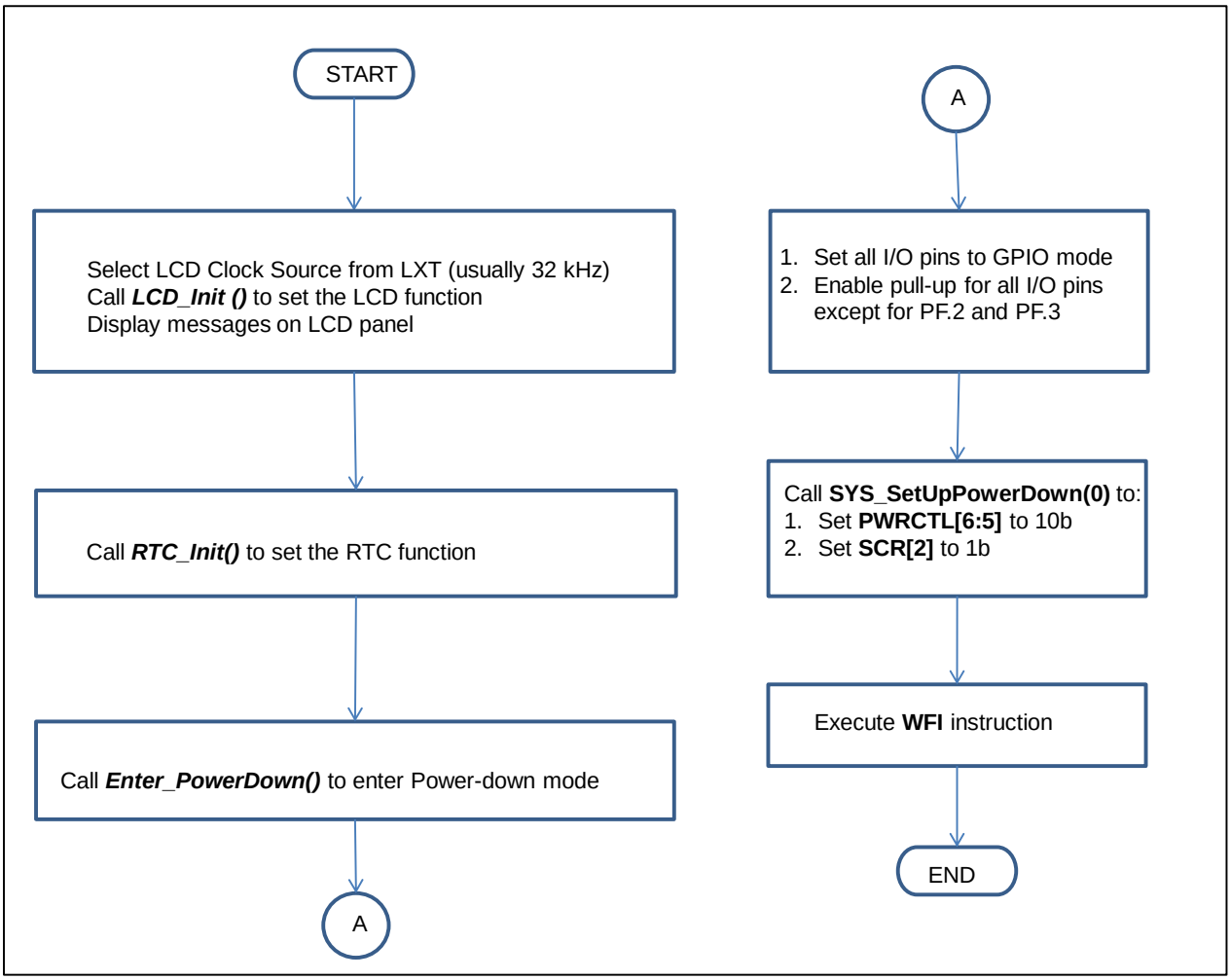

Figure 2-3 PWRDWN_DEMO Flow Chart

The code of entering power-down is defined at function void ***Enter_PowerDown()***. The following lists crucial statements of this function.

```
SYS_SetUpPowerDown(0);  /* Disable PDWU interrupt */
__WFI();     /* system really enter power down here ! */
```

2.2 PWRDWN_RTC_DEMO

In addition to the initialization as "***PWRDWN_DEMO***", the main routine enables PDWU interrupt and sets a RTC alarm timer. This RTC alarm timer will generate a RTC wake-up interrupt after 1 minute. The program goes to power-down after the above initialization is completed. The RTC timer keeps active while LCD display is disabled in Power-down mode.

As soon as RTC wake-up interrupt happens, the PDWU and RTC interrupt service routines are served first. A RTC alarm timer will be set in RTC interrupt service routine for the next wake-up event. Finally, the control returns to main routine. Some messages are displayed on LCD panel, and the program goes to power-down again. The power-down and wake-up procedure will be repeated eternally.

Main routine: *main()*

- Set LCD and RTC functions; display messages on LCD panel.
- Enable PDWU interrupt.
- Set a RTC alarm timer.
- Call ***Enter_PowerDown()***

Power-down subroutine: *Enter_PowerDown()*

- Set all I/O pins to GPIO mode, and enable pull-up for all I/O pins except for PF.2 and PF.3.
- Disable LCD clock.
- Call ***SYS_SetUpPowerDown()*** to set **PWRCTL[6:5]** to 11b, and set **SCR[2]** to 1b.
- Execute the WFI instruction.

Wake-up subroutine: *Leave_PowerDown()*

- Restore the setting of all I/O pins.
- Enable LCD clock.

The following figure shows a flow chart of this sample program.

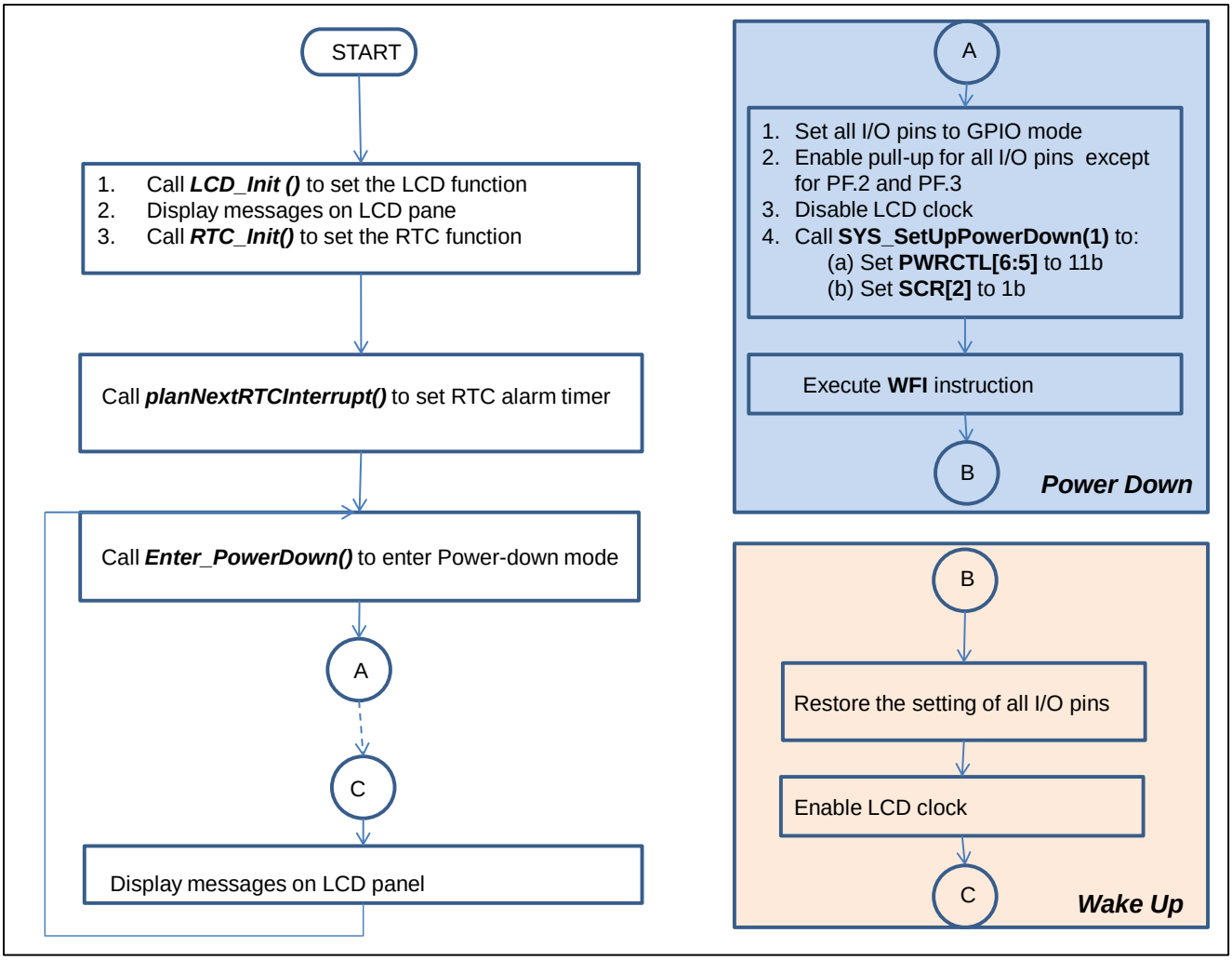

Figure 2-4 PWRDWN_RTC_DEMO Flow Chart

2.3 PWRDWN_LCD_RTC_DEMO

This sample supports almost the same features with "***PWRDWN_RTC_DEMO***". The only difference is that LCD display keeps active when CPU is in Power-down mode. To keep LCD display in Power-down, the LCD clock should not be disabled; meanwhile, the COM/SEG I/O pins should not be set to GPIO mode.

Main routine: *main()*

- Set LCD and RTC functions; display messages on LCD panel.
- Enable LCD control to display in Power-down mode.
- Enable PDWU interrupt; set a RTC alarm timer.
- Call ***Enter_PowerDown()***

Power-down subroutine: *Enter_PowerDown()*

- Set pull-up for all I/O pins except for PF.2 and PF.3.
- Call ***SYS_SetUpPowerDown()*** to set **PWRCTL[6:5]** to 11b, and set **SCR[2]** to 1b.
- Execute the WFI instruction.

Wake-up subroutine: *Leave_PowerDown()*

- Restore the setting of all I/O pins.

The following figure shows a flow chart of this sample program.

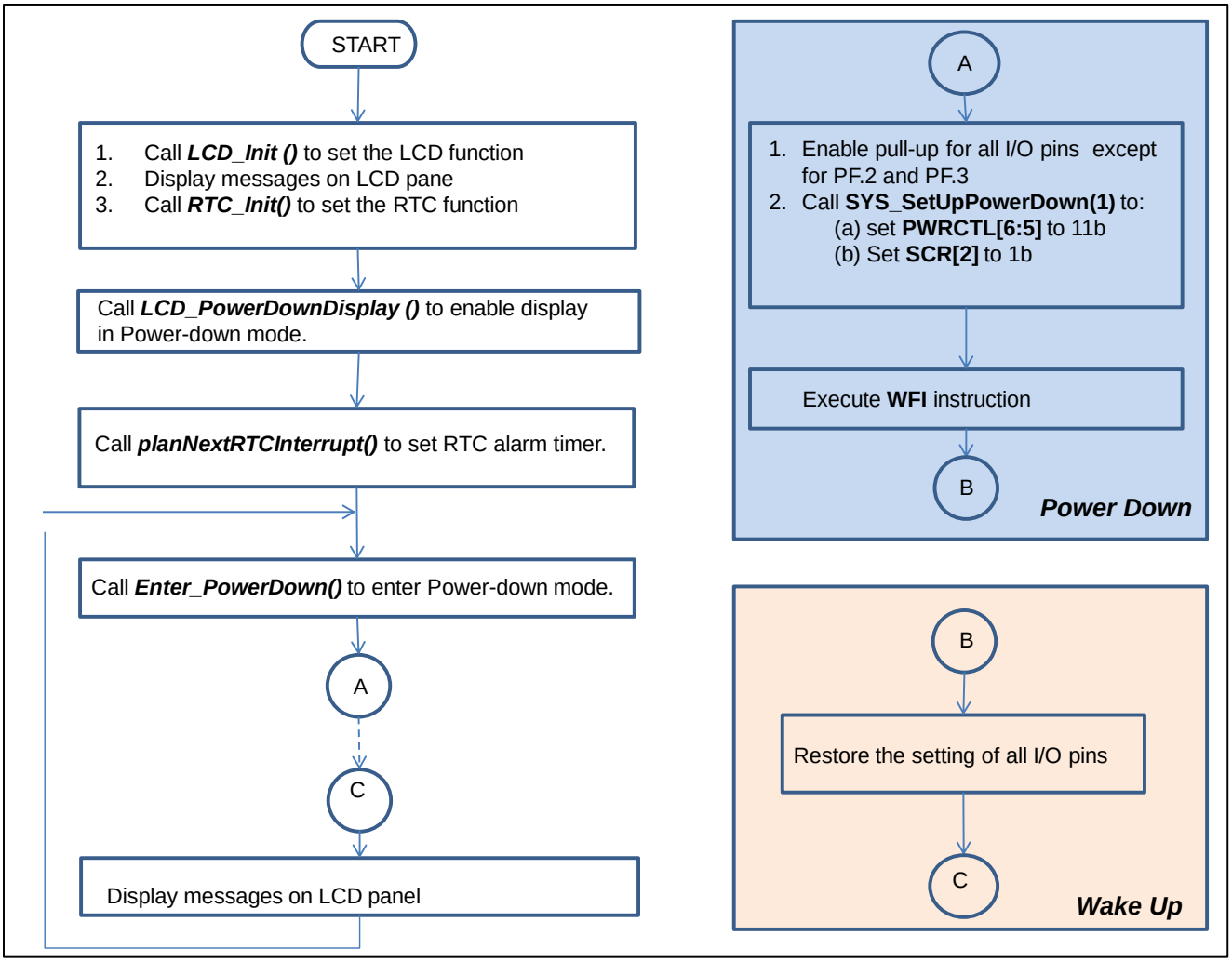

Figure 2-5 PWRDWN_LCD_RTC_DEMO Flow Chart

2.4 I/O Pin Control

To prevent current leakage, all I/O pins are usually set to GPIO mode with pull-up enabled before entering Power-down mode. It is better to save the mode setting of all I/O pins in advance.

```
/* Back up original setting */
SavePinSetting();
/* Set function pin to GPIO mode */
GCR->PA_L_MFP = 0;
GCR->PA_H_MFP = 0;
GCR->PB_L_MFP = 0;
GCR->PB_H_MFP = 0;
GCR->PC_L_MFP = 0;
GCR->PC_H_MFP = 0;
GCR->PD_L_MFP = 0;
GCR->PD_H_MFP = 0;
GCR->PE_L_MFP = 0;
GCR->PE_H_MFP = 0;
GCR->PF_L_MFP = 0x00007700; /* exclude PF.2 and PF.3 which are HXT OUT/IN */

/* Enable GPIO pull up */
GPIOA->PUEN = 0xFFFF;
GPIOB->PUEN = 0xFFFF;
GPIOC->PUEN = 0xFFFF;
GPIOD->PUEN = 0xFFFF;
GPIOE->PUEN = 0xFFFF;
GPIOF->PUEN = 0x0033;       /* exclude GPF2 and GPF3 which are HXT OUT/IN */
```

If the peripheral function has to work in Power-down mode, the corresponding I/O pins should not be set to GPIO mode. For example, the COM/SEG and other LCD control signal pins should not be changed for LCD to display in Power-down mode.

Generally, the program should restore the original mode setting of all I/O pins after wake-up.

3 Power-down Current Measurement

3.1 Current Measurement Method

3.1.1 Measuring Pure Current Consumption of NANO130

- Remove DC blocking diode **D1** **(SS24A)** of **NuTiny-EVB-Nano130-LQFP128 board** to avoid SS24A DC reverse current issue.
- Supply power by **JP2** (VDD33) and **JP3** (GND)
- Measure the power-down current value by a current meter.

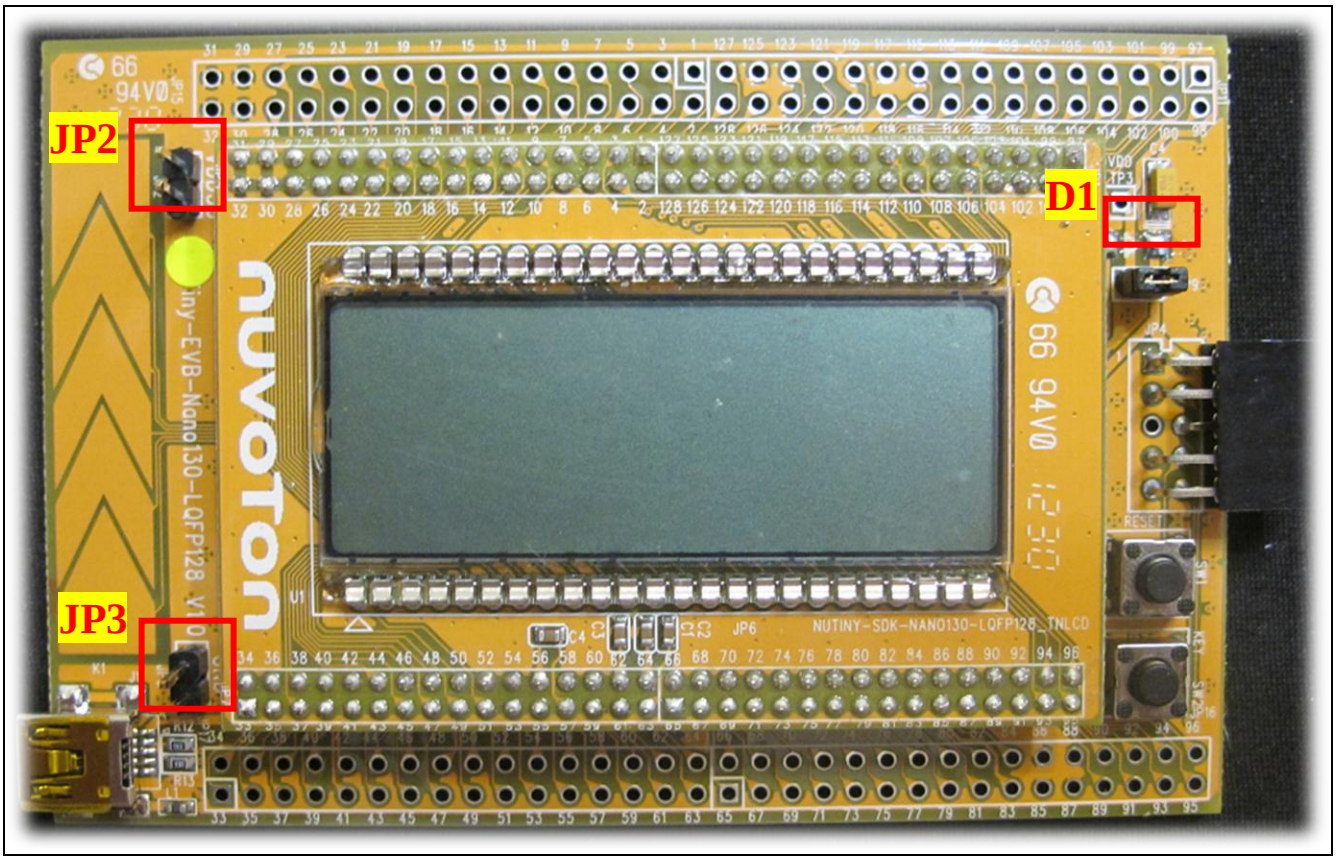

Figure 3-1 NuTiny-EVB-Nano130-LQFP128 Board

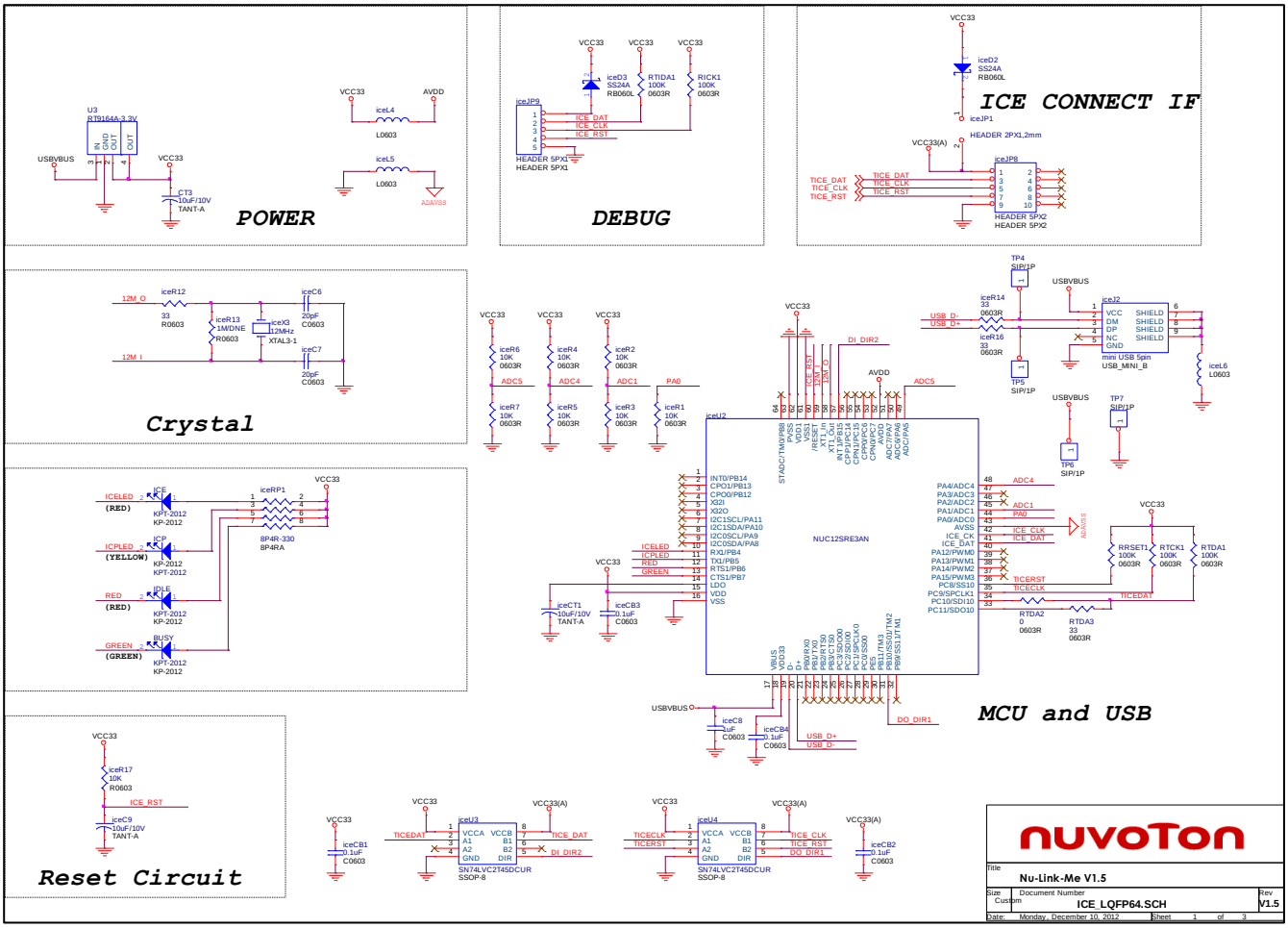

Figure 3-2 NuTiny-EVB-Nano130-LQFP128 Schematic

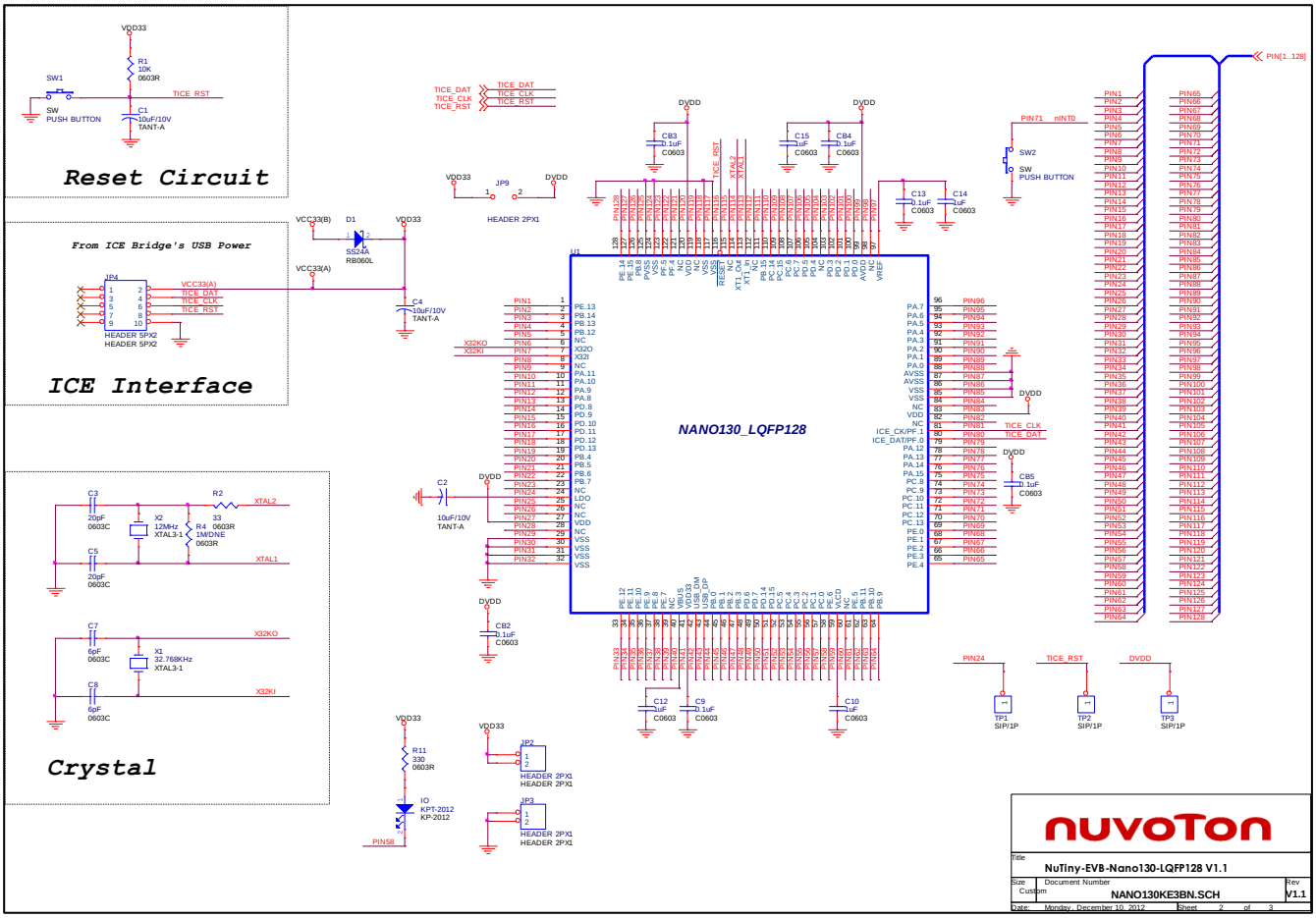

Figure 3-3 NuTiny-EVB-Nano130-LQFP128 Schematic

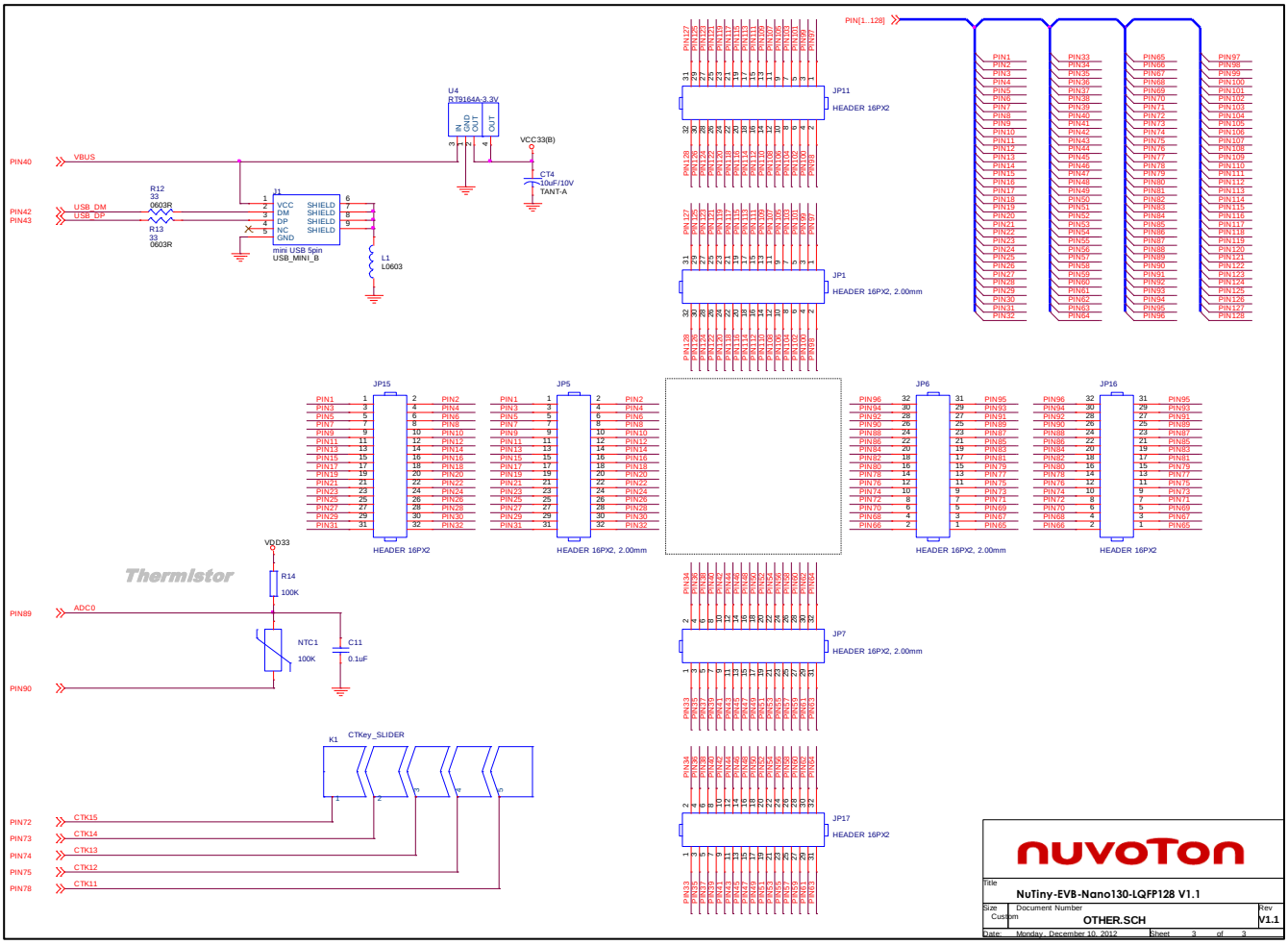

Figure 3-4 NuTiny-EVB-Nano130-LQFP128 Schematic

3.1.2 Reducing I/O Pins Leakage

It is very important to take care of the I/O pins for power consumption issues. The following shows some suggestions for reducing power consumption when designing the NANO100 series MCU system:

- Do not enable pull-up resister of input pins which will be forced 'Low' by external circuit in Power-down mode.
- Take care output status and consider the current flow of all output pins.
- Unused I/O pins need to be set to GPIO input mode with the pull-up resister enabled.

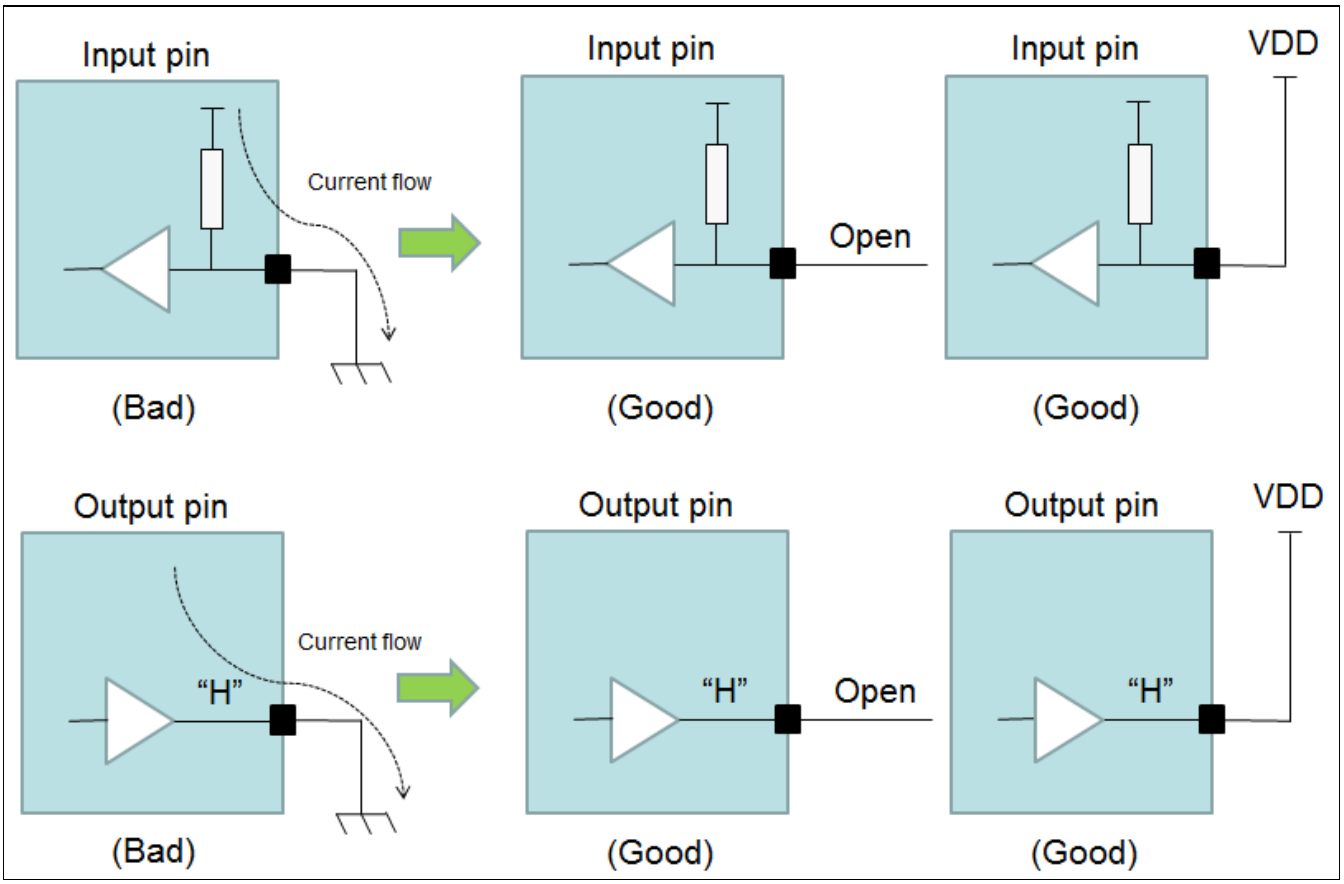

Figure 3-5 Suggestions for Reducing Power Consumption

3.2 Current Measurement Results

Part No	Test Conditions	VDD	Current
NuTiny-EVB Nano130-LQFP128 V1.0	RTC + LCD Mode: (RAM Retention) (Power-down with 32K and LCD enabled) CPU stopped Clock = 32.768 kHz Crystal Oscillator All peripheral disabled except RTC and LCD circuit Without panel loading	3.3V	10uA
	RTC Mode: (RAM Retention) (Power-down with 32K enabled) CPU stopped Clock = 32.768 kHz Crystal Oscillator All peripheral disabled except RTC circuit	3.3V	2.5uA
	Power-down Mode: (RAM Retention) CPU and all clocks stopped	3.3V	1.0uA

Revision History

Rev.	Date	Description
1.00	11-1-2012	Initially issued.

Important Notice

Nuvoton products are not designed, intended, authorized or warranted for use as components in systems or equipment intended for surgical implantation, atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for other applications intended to support or sustain life. Furthermore, Nuvoton products are not intended for applications wherein failure of Nuvoton products could result or lead to a situation wherein personal injury, death or severe property or environmental damage could occur.
Nuvoton customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from such improper use or sales.

Please note that all data and specifications are subject to change without notice. All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.