UART Driver Sample Code Reference Guide

V1.00.001

Publication Release Date: Sep. 2011

Support Chips:

ISD9160

Support Platforms:

NuvotonPlatform_Keil

The information in this document is subject to change without notice.

The Nuvoton Technology Corp. shall not be liable for technical or editorial errors or omissions contained herein; nor for incidental or consequential damages resulting from the furnishing, performance, or use of this material.

Table of Contents

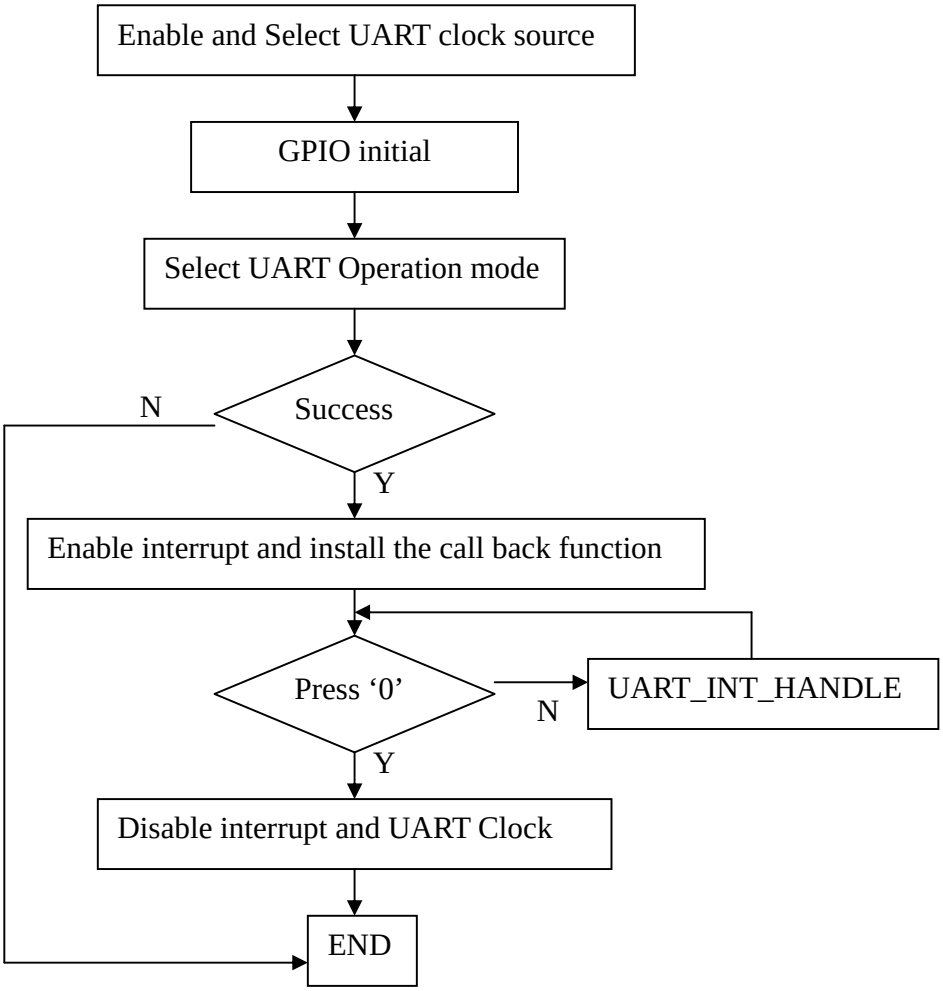
1 Introduction

This sample code will demo UART IP on ISD9160 chip.

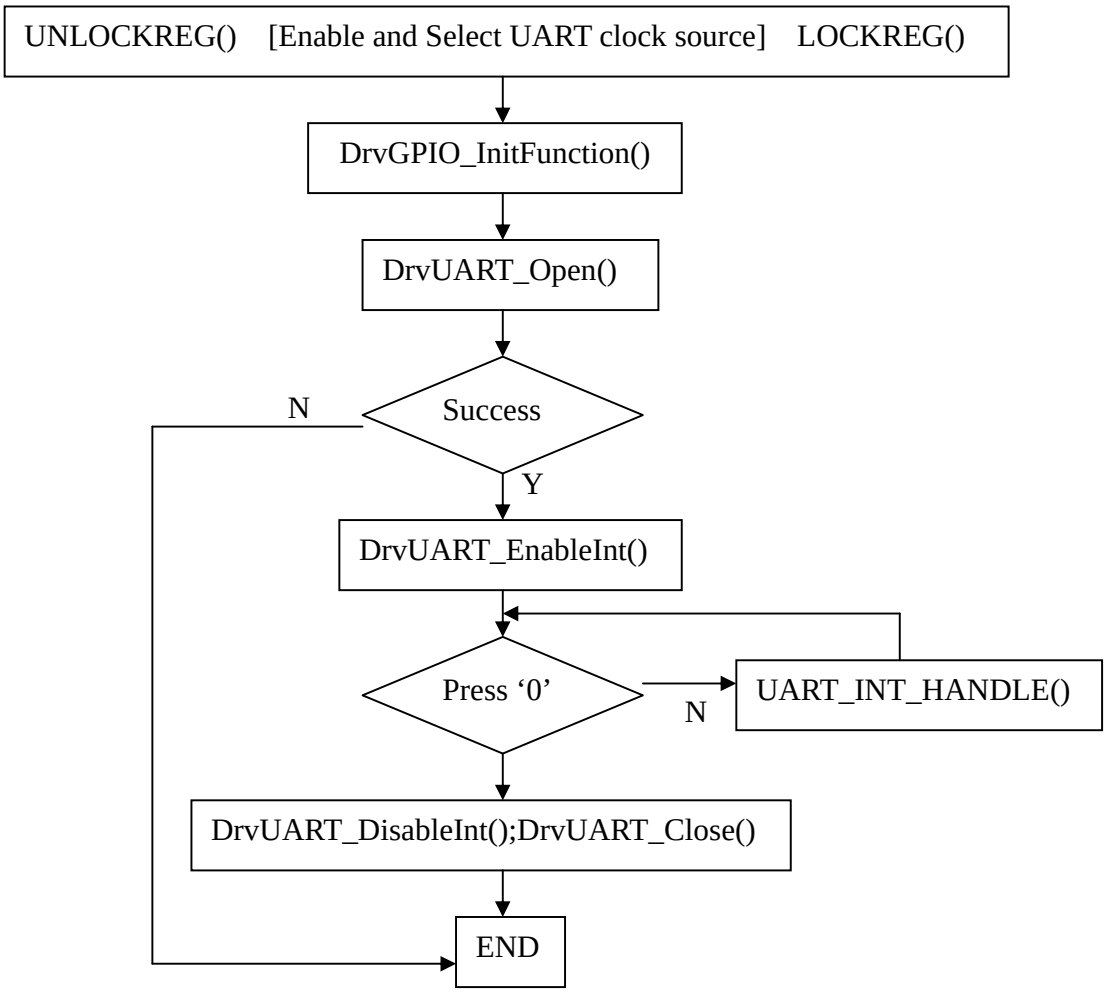
1.1 Feature

- 8 bytes entry FIFOs for received and transmitted data payloads.
- Auto flow control/flow control function (CTS, RTS) are supported.
- Fully programmable serial-interface characteristics:
  - 5-, 6-, 7-, or 8-bit character
  - Even, odd, or no-parity bit generation and detection
  - 1-, 1&1/2, or 2-stop bit generation
  - Baud rate generation
  - False start bit detection.

2 Block Diagram



3 Calling Sequence



4 Code Section –Smpl_DrvUART.c

```
/*---------------------------------------------------------------------------------------------------------*/
/*
*/
/* Copyright(c) 2011 Nuvoton Technology Corp. All rights reserved.
*/
/*
*/
/*---------------------------------------------------------------------------------------------------------*/
#include <stdio.h>
#include "Driver\DrvUART.h"
#include "Driver\DrvGPIO.h"
#include "ISD9xx.h"

#define RXBUFSIZE 64

/*---------------------------------------------------------------------------------------------------------*/
/* Global variables
*/
/*---------------------------------------------------------------------------------------------------------*/
volatile uint8_t comRbuf[RXBUFSIZE];
volatile uint16_t comRbytes = 0;          /* Available receiving bytes */
volatile uint16_t comRhead    = 0;
volatile uint16_t comRtail      = 0;
volatile int32_t g_bWait        = TRUE;

/*---------------------------------------------------------------------------------------------------------*/
/* Define functions prototype
*/
/*---------------------------------------------------------------------------------------------------------*/
void UART_INT_HANDLE(uint32_t u32IntStatus);

/*---------------------------------------------------------------------------------------------------------*/
/* UART Callback function
            */
/*---------------------------------------------------------------------------------------------------------*/
void UART_INT_HANDLE(uint32_t u32IntStatus)
{

        uint8_t bInChar[1]={0xFF};


        if(u32IntStatus & RDAIE)
        {
                printf("\nInput:");
```

```
            /* Get all the input characters */
            while(UART0->ISR.RDA_IF==1)
            {
                /* Get the character from UART Buffer */
                DrvUART_Read(UART_PORT0,bInChar,1);
                printf("%c ", bInChar[0]);
                if(bInChar[0] == '0')
                {
                    g_bWait = FALSE;
                }
                /* Check if buffer full */
                if(comRbytes < RXBUFSIZE)
                {
                    /* Enqueue the character */
                    comRbuf[comRtail] = bInChar[0];
                    comRtail = (comRtail == (RXBUFSIZE-1)) ? 0 : (comRtail+1);
                    comRbytes++;
                }
            }
            printf("\nTransmission Test:");
        }
        else if(u32IntStatus & THREIE)
        {
            uint16_t tmp;
            tmp = comRtail;
            if(comRhead != tmp)
            {
                bInChar[0] = comRbuf[comRhead];
                DrvUART_Write(UART_PORT0,bInChar,1);
                comRhead = (comRhead == (RXBUFSIZE-1)) ? 0 : (comRhead+1);
                comRbytes--;
            }
        }
}
/*---------------------------------------------------------------------------------------------------------*/
/* UART Test Sample                                                                                        */
/* Test Item                                                                                               */
/* It sends the received data to HyperTerminal.                                                            */
/*---------------------------------------------------------------------------------------------------------*/
int32_t main()
{
        STR_UART_T sParam;
        /* Step 1. Enable and Select UART clock source*/
        UNLOCKREG();
        SYSCLK->PWRCON.OSC49M_EN = 1;
        SYSCLK->PWRCON.OSC10K_EN = 1;
        SYSCLK->PWRCON.XTL32K_EN = 1;
        SYSCLK->CLKSEL0.STCLK_S = 3; //Use internal HCLK
//      SYSCLK->CLKSEL0.HCLK_S = 0; /* Select HCLK source as 48MHz */
        SYSCLK->CLKSEL0.HCLK_S = 1; /* Select HCLK source as 32KHz */
        SYSCLK->CLKDIV.HCLK_N    = 0;   /* Select no division                  */
```

```
    /* Step 1. Enable and Select UART clock source*/
    UNLOCKREG();
    SYSCLK->PWRCON.OSC49M_EN = 1;
    SYSCLK->PWRCON.OSC10K_EN = 1;
    SYSCLK->PWRCON.XTL32K_EN = 1;
    SYSCLK->CLKSEL0.STCLK_S = 3; //Use internal HCLK

//  SYSCLK->CLKSEL0.HCLK_S = 0; /* Select HCLK source as 48MHz */
    SYSCLK->CLKSEL0.HCLK_S = 1; /* Select HCLK source as 32KHz */
    SYSCLK->CLKDIV.HCLK_N    = 0;   /* Select no division                */
    SYSCLK->CLKSEL0.OSCFSel = 0;    /* 1= 32MHz, 0=48MHz */
    LOCKREG();

    /* Step 2. GPIO initial */
    DrvGPIO_InitFunction(FUNC_UART0);

    /* Step 3. Select UART Operation mode */
//    sParam.u32BaudRate        = 115200;
    sParam.u32BaudRate        = 2400;
    sParam.u8cDataBits        = DRVUART_DATABITS_8;
    sParam.u8cStopBits        = DRVUART_STOPBITS_1;
    sParam.u8cParity          = DRVUART_PARITY_NONE;
    sParam.u8cRxTriggerLevel= DRVUART_FIFO_1BYTES;

    if(DrvUART_Open(UART_PORT0,&sParam) == 0)
    {
        printf("\nUART Sample Demo. (Press '0' to exit)\n");

        /* Step 4. Enable Interrupt and install the call back function */
        DrvUART_EnableInt(UART_PORT0, (DRVUART_THREINT |
DRVUART_RDAINT),UART_INT_HANDLE);

        while(g_bWait);
        /* Disable Interrupt */
        DrvUART_DisableInt(UART_PORT0,DRVUART_RLSNT |
DRVUART_THREINT | DRVUART_RDAINT);

        /* Disable I & F bit */
        NVIC_DisableIRQ (UART0_IRQn);
        printf("\nUART Sample Demo End.\n");
        /* Disable UART Clock */
        DrvUART_Close(UART_PORT0);
        return TRUE;
    }
    else
        return FALSE;
}
```

5 Execution Environment Setup and Result

- Prepare a ISD9160 board.
- Compile the sample code.
- Connect the RS232 between ISD9160 and PC.
- What you press in the PC, ISD9160 will receive and transmit it again to the PC, except ‘0’. You can see what you press in the terminal of PC..

6 Revision History

Version	Date	Description
V1.00.01	Sep. 2011	Created