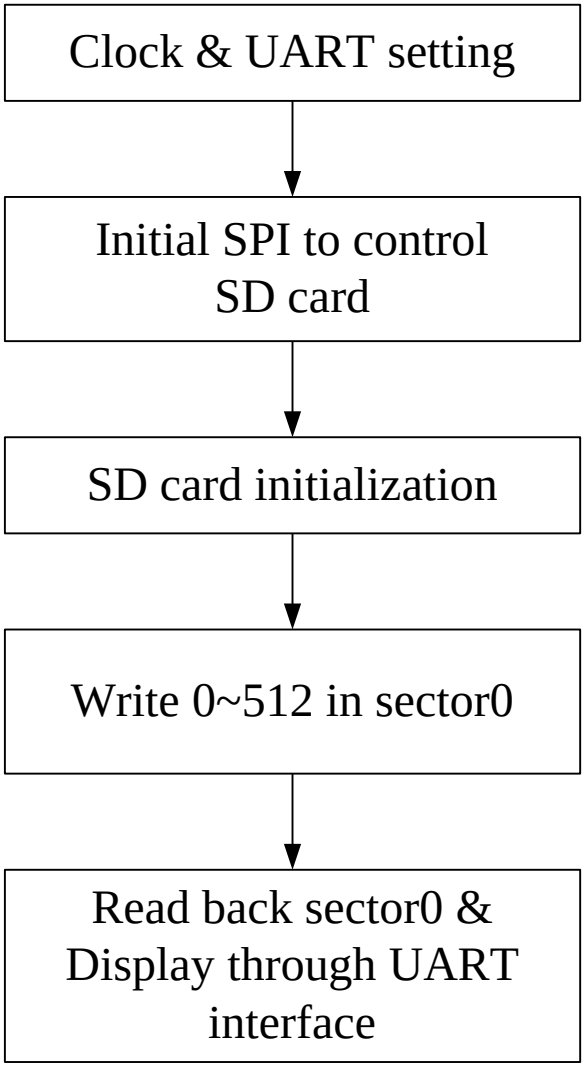
1. SDCard Driver Introduction

This sample code will demo read/write SD card on ISD9160 chip.

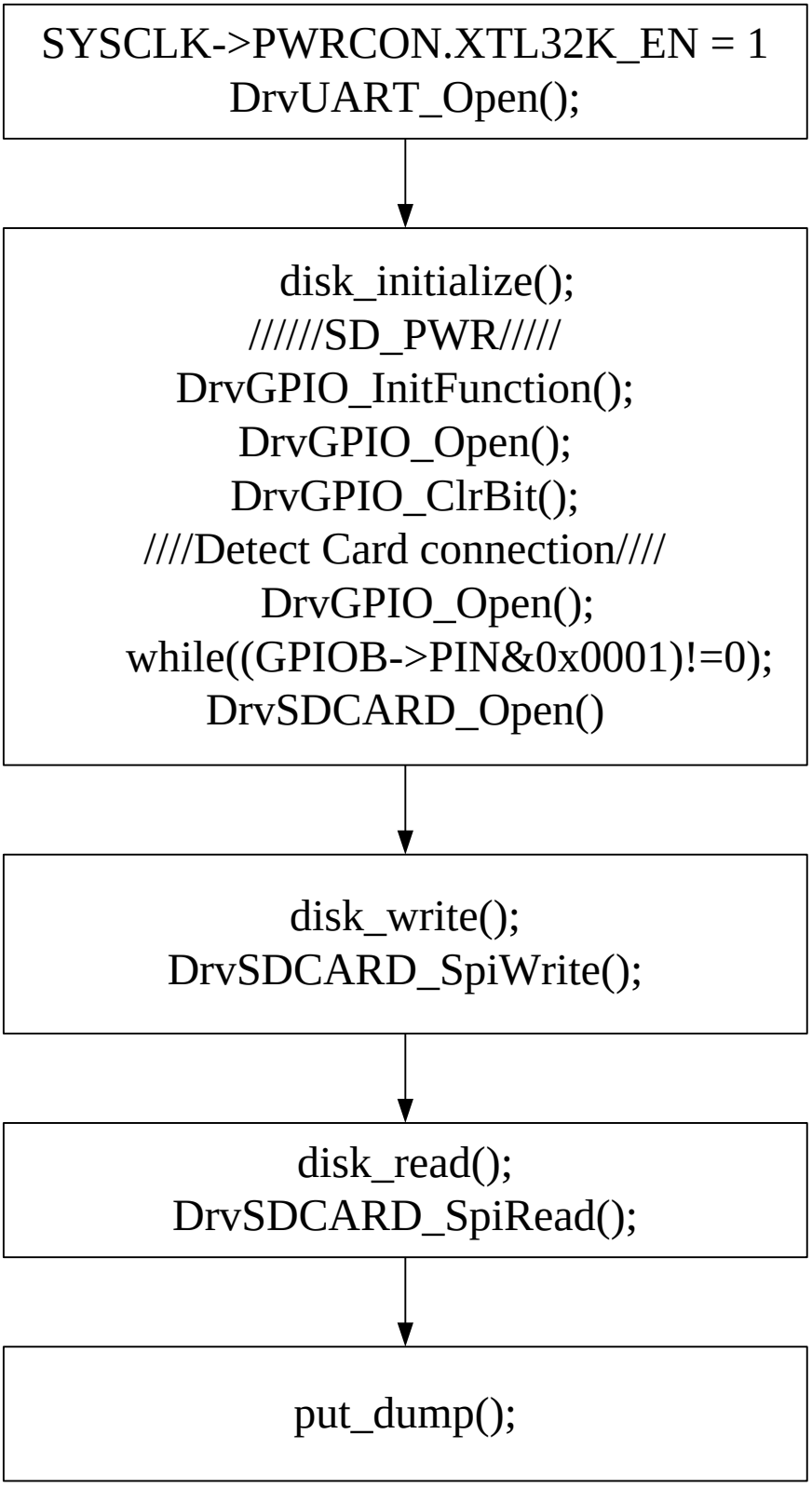
1.1 Feature

1. Using SPI mode write to SD card.
2. Using SPI mode read back from SD card.

2. Block Diagram



3. Calling Sequence



4. Code Section –Smpl_DrvSDCard.c

```
/*---------------------------------------------------------------------------*/
/*                                                                           */
/* Copyright(c) 2011 Nuvoton Technology Corp. All rights reserved. */
/*                                                                           */
/*---------------------------------------------------------------------------*/
#include <stdio.h>
#include "DrvSPI.h"
#include "DrvGPIO.h"
#include "DrvSYS.h"
#include "DrvUART.h"
#include "DrvSDCARD.h"
#include "diskio.h"

/*---------------------------------------------------------------------------*/
/* Global variables                                                          */
/*---------------------------------------------------------------------------*/
unsigned char ucWrBuff[512],ucRdBuff[512];

/*---------------------------------------------------------------------------*/
/* Define functions prototype                                                */
/*---------------------------------------------------------------------------*/
void Delay(uint32_t delayCnt)
{
    while(delayCnt--)
    {
        __NOP();
        __NOP();
    }
}
```

```
/*---------------------------------------------*/
/* Dump a block of byte array                  */
/*---------------------------------------------*/
/* buff: Pointer to the byte array to be dumped */
/* addr: Heading address value                 */
/*    cnt: Number of bytes to be dumped        */
void put_dump (const unsigned char* buff, unsigned long addr, int cnt)
{
	int i;

	printf("%08lX ", addr);

	for (i = 0; i < cnt; i++)
		printf(" %02X", buff[i]);

	putchar(' ');
	for (i = 0; i < cnt; i++)
		putchar((char)((buff[i] >= ' ' && buff[i] <= '~') ? buff[i] : '.'));

	putchar('\n');
}

/*-----------------------------------------------------------------------------------*/
/*    MAIN function                                                                  */
/*-----------------------------------------------------------------------------------*/
int32_t main(void)
{
	int iCnt;
	unsigned char *ucBuff;
	unsigned long ulOfs;

	STR_UART_T sParam;

	UNLOCKREG();
	SYSCLK->PWRCON.OSC49M_EN = 1;
	SYSCLK->CLKSEL0.HCLK_S = 0; /* Select HCLK source as 48MHz */
	SYSCLK->CLKDIV.HCLK_N    = 0;    /* Select no division                 */
	SYSCLK->CLKSEL0.OSCFSel = 0;     /* 1= 32MHz, 0=48MHz */
	LOCKREG();
```

```
    /* Set UART Pin */
    DrvGPIO_InitFunction(FUNC_UART0);

    /* UART Setting */
    sParam.u32BaudRate      = 115200;
    sParam.u8cDataBits      = DRVUART_DATABITS_8;
    sParam.u8cStopBits      = DRVUART_STOPBITS_1;
    sParam.u8cParity        = DRVUART_PARITY_NONE;
    sParam.u8cRxTriggerLevel= DRVUART_FIFO_1BYTES;

    /* Set UART Configuration */
    DrvUART_Open(UART_PORT0,&sParam);

    Delay(1000);

    printf("+-----------------------------------------------------------------+\n");
    printf("|                        SD Card Sample Code                      |\n");
    printf("+-----------------------------------------------------------------+\n");
    printf("| Write integer 0~511 to sector 0                                 |\n");
    printf("| Read back and check the data                                    |\n");
    printf("+-----------------------------------------------------------------+\n");
    printf("\n");
    printf("rc=%d\n", (DSTATUS)disk_initialize(0));

    for(iCnt=0; iCnt<512; iCnt++)
        ucWrBuff[iCnt] = iCnt;
    printf("rc=%u\n", disk_write(0, ucWrBuff, 0, 1));

////Read back
    printf("rc=%u\n", disk_read(0, ucRdBuff, 0, 1));

//      for(iCnt=0; iCnt<512; iCnt++)
//      {
//          if(ucRdBuff[iCnt] != ucWrBuff[iCnt])
//              printf("Error in Addr[%d]:%02X\n",iCnt,ucRdBuff[iCnt]);
//      }

    for (ucBuff=(unsigned char*)ucRdBuff, ulOfs = 0; ulOfs < 0x200; ucBuff+=16, ulOfs+=16)
        put_dump(ucBuff, ulOfs, 16);

}
```

```
        ////////////////////////////////////////////////////DiskIO.C/////////////////////////////////////////////////////////
/*-----------------------------------------------------------------------------------------------*/
/* Low level disk control module for Win32                    (C)ChaN, 2007        */
/*-----------------------------------------------------------------------------------------------*/
#include <stdio.h>
#include "diskio.h"
#include "DrvSDCARD.h"
#include "DrvGPIO.h"

extern void DrvSDCARD_SpiRead(uint32_t addr, uint32_t size, uint8_t *buffer);
extern void DrvSDCARD_SpiWrite(uint32_t addr, uint32_t size,    uint8_t *buffer);

void RoughDelay(uint32_t t)
{
      volatile int32_t delay;

      delay = t;

      while(delay-- >= 0);
}

/*-----------------------------------------------------------------------*/
/* Initialize Disk Drive                                                 */
/*-----------------------------------------------------------------------*/
/* ucDrv : Physical drive nmuber */
DSTATUS disk_initialize (unsigned char ucDrv)
{
      DSTATUS sta;

//////SD_PWR
      DrvGPIO_InitFunction(FUNC_GPIO);
      DrvGPIO_Open(GPB,1,IO_OUTPUT);
      DrvGPIO_ClrBit(GPB,1);

////Detect Card connection
      DrvGPIO_Open(GPB,0,IO_INPUT);
      while((GPIOB->PIN&0x0001)!=0);
      RoughDelay(100000);
```

```
	if(DrvSDCARD_Open() == E_SUCCESS)
	{
		sta =	RES_OK;
		printf("SDCard Open success\n");
	}
	else
	{
		sta = STA_NOINIT;
		printf("SDCard Open failed\n");
	}
	return sta;
}

/*-----------------------------------------------------------------------*/
/* Get Disk Status                                                       */
/*-----------------------------------------------------------------------*/
/* ucDrv : Physical drive nmuber */
DSTATUS disk_status (unsigned char ucDrv)
{
	DSTATUS sta1=STA_OK;
	if (ucDrv)
		sta1 =	STA_NOINIT;
	return sta1;
}



/*-----------------------------------------------------------------------*/
/* Read Sector(s)                                                        */
/*-----------------------------------------------------------------------*/
/* ucDrv :	Physical drive nmuber (0) */
/* ucBuff:	Pointer to the data buffer to store read data */
/* ulSector: Start sector number (LBA) */
/* ucCount : Sector count (1..255) */
DRESULT disk_read (unsigned char ucDrv, unsigned char *ucBuff,
				unsigned long ulSector, unsigned char ucCount)
{
	DRESULT res;
	uint32_t size;

	if (ucDrv)
	{
		res = (DRESULT)STA_NOINIT;
		return res;
	}
```

```
                if(ucCount==0||ucCount>=2)
        {
                res =      (DRESULT)STA_NOINIT;
                return res;
        }

        size = ucCount*512;
        DrvSDCARD_SpiRead(ulSector, size, ucBuff);
        res =RES_OK;        /* Clear STA_NOINIT */;

        return res;
}



/*-----------------------------------------------------------------------*/
/* Write Sector(s)                                                       */
/*-----------------------------------------------------------------------*/
/* Physical drive nmuber (0) */
/* Pointer to the data to be written */
/* Start sector number (LBA) */
/* Sector count (1..255) */
DRESULT disk_write (unsigned char ucDrv, const unsigned char *ucBuff,
                            unsigned long ulSector,   unsigned char ucCount)
{
        DRESULT    res;
        uint32_t size;

        if (ucDrv) {
                res = (DRESULT)STA_NOINIT;
                return res;
        }

        if(ucCount==0||ucCount>=2)
        {
                res = (DRESULT)    STA_NOINIT;
                return res;
        }

        size = ucCount*512;
        DrvSDCARD_SpiWrite(ulSector, size,(uint8_t *)ucBuff);
        res = RES_OK;

        return res;
}
```

5. Execution Environment Setup and Result

- Prepare a ISD9160 board.
- Compile the sample code.
- Console window show result of SD card.