AstroII ISC 用户指南

Version 1.0 January 2010

Capital Microelectronics, Inc.

Contents

AstroII ISC 用户指南

1 简介

AstroII 支持多配置映像 in system configuration (ISC)。用户可以根据系统的运行情况和外部条件不用关闭电源选择合适的配置映像配置 AstroII 并运行新的设计。AstroII 支持的配置映像数目取决于 SPI FLASH 的大小和配置文件的大小。配置文件包含 FPGA 的配置数据和 MCU 的代码。某些特定应用可以利用 AstroII 的 ISC 特性，用低成本的 SPI FLASH 空间换取高成本的 FPGA 和 SRAM 空间。这些特定的应用有相同的 I/O 要求， FPGA 设计和 MCU 设计功能相同，但是实现方法不同，且不同时运行。满足这些条件的设计都可用 ISC 特性来拓展用户系统的性价比。

AstroII 集成了高性能增强型 8051MCU， MCU 对 AstroII 的操作都简化为 MCU 对 SFR 的操作，MCU 对 AstroII 的操作就是软件对 SFR 的读写，非常灵活易用。

AstroII 的 8051MCU 可以很方便地读写配置 SPI FLASH 的多余空间。于 ISC 功能相结合，AstroII 就能方便的实现在线更新并运行。

2 ISC

MCU 对 AstroII 的 ISC 操作都是通过读写扩展的 SFR 来进行的，AstroII 的硬件配置逻辑实际控制对 SPI FLASH 的操作，不需要其他任何硬件设计。

Astro Ⅱ的配置 Image 由 FPGA 配置数据和 MSS 的 8051 程序代码组成，FPGA 配置数据大小基本上一样，MSS 的代码随程序的的大小变化。由于 SPI FLASH 的擦除特性，Astro Ⅱ的配置 Image 是按 Sector 存放在 SPI FLASH 里，一个 Image 可占用多个 Sector。图 3-10 描述了多个 Image 存放在 SPI FLASH 的映射关系，示意的是一个 Image 占用一个 Sector。在 Primace 软件也是依据这种关系生成并下载多个 Image 到 SPI FALSH 里。

2.1 原理示意图

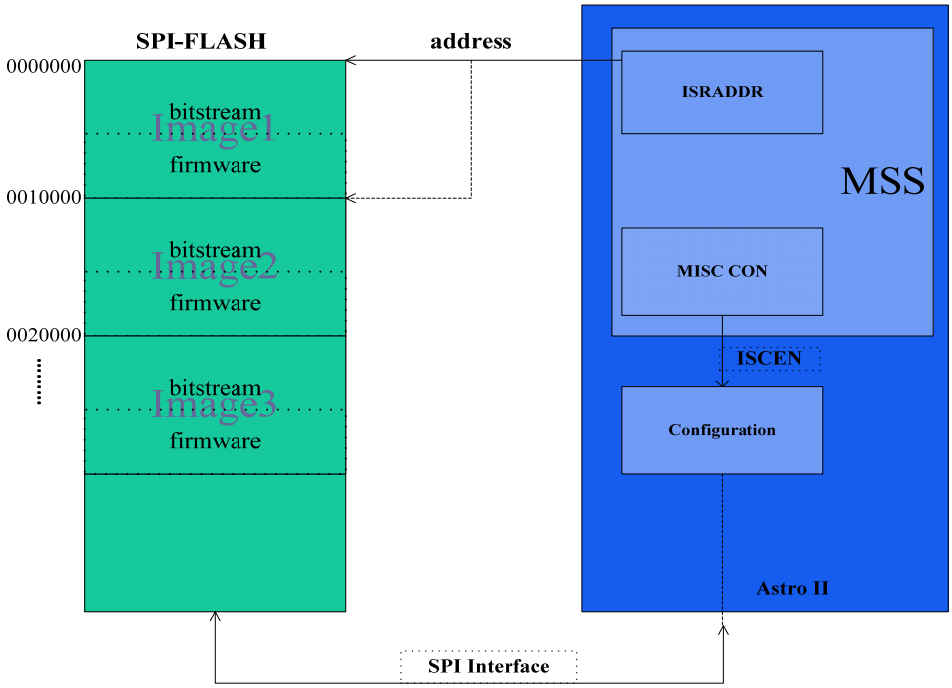


图 1 ISC 示意图

2.2 ISC SFR描述

Register Name	Address (10 bits)	R/W	Default value	Descriptions
register	**location**	**attribute**	**Reset value**	**description**
I2CSPISEL	ABh	R/W	8'H00	Bit0: SPI interface source. must be 1 Bit1：I²C IO select, contro the device IO attribute 0: the IO is used as general user IO 1: the IO is used as fixed I²C IO Bit[7:2] Reserved
ISCADDR0	Ach	R/W	8'H00	Reconfiguration start address [7:0]
ISCADDR1	ADh	R/W	8'H00	Reconfiguration start address [15:8]
ISCADDR2	AEh	R/W	8'H00	Reconfiguration start address [23:16]
ISCADDR3	AFh	R/W	8'H00	Reconfiguration start address [31:24]
MISCCON	F8h	R/W	8'H00	Bit0 ISCEN: reconfiguration enable, be used to trigger the reconfiguration sequence 0: disable 1:enable BIT1 DPRAMWP: DPRAM write protect from FP 0:disable write 1:enable write Bit2 PLLLOCK: read only pll_lock, the status of pll 0: not lock 1: lock Bit3 PLLPWD: pll_pwrdn, pll power down 0: open the pll power 1: close the pll power

				Bit4 CLKCPUTRIG: Triggers CLKCPUON and make it active when changes from 0 to 1. Bit5 CLKO0TRIG: Triggers CLKO0CON and make it active when changes from 0 to 1. Bit6 CLKO1TRIG: Triggers CLKO1CON and make it active when changes from 0 to 1. Bit7 CLKO2TRIG: Triggers CLKO2CON and make it active when changes from 0 to1.

Table 1 Description of registers

2.3 ISC 步骤

ISC 步骤如下：

- 向 ISCADDR3~ISCADDR0 4 个 SFR 写入想要配置的 Image 在 SPI FLASH 的起始地址
- 向 MISCCON（0xF8）的第 0 位 ISCEN 写入 1

这两步后，AstroII 从虚线指向的地址读取 Image 的数据，进行自我配置。在重配置过程中 MSS 是处在复位状态，ISCADDR3~ISCADDR0 以及 ISCEN 都自动清 0。

注意：AstroⅡ上电复位后，AstroⅡ默认读取 Image1 的数据进行配置。

3 在线更新

当用外部 SPI FLASH 作为 AstroII 的配置芯片时，AstroII 配置完成后，SPI 对应的 AstroII I/O 管脚可以用作普通 I/O。在 FPGA 设计中连接 8051 系统的 spi 接口与外部的 SPI FLASH 端口。

当 AstroII 内部集成了 SPI FLASH 时，在在 FPGA 设计中调用 spi_interface 模块，再连接 8051 系统的 spi 接口 spi_interface 端口。

下面的 8051 spi 用作 master，SPI FLASH 用作 slave 0。

AstroII 的在线更新就纯粹是 8051 的编程了。用户可以通过串口或别的接收新的配置文件数据并重新覆盖原来的配置文件，重上电或在线 ISC 就可使 AstroII 运行在新的设计下。

3.1 外部SPI FLASH

外部 SPI FLASH 的连接如下：

```
output flash_sclk ;
output flash_sdo ;
output flash_cson   ;
input flash_sdi ;

wire ssn;
wire misoi, misoo, misotri;
wire mosii, mosio, mositri;
wire scki, scko, scktri;
```

```
wire [7:0]spssn ;

assign      flash_sclk = scko;
assign      flash_sdi = mosio;
assign      flash_cson = spssn[0];
assign      misoi = flash_sdo;

mySystem    U_R8051XC(
            .clkcpu             (clk      ),
            .clkemif            (clk      ),
            .reset              (cpu_rst),
            .swd(),
            .ro(),
            .port0i             (port0i     ),
            .port1i             (port1i     ),
            .port2i             (port2i     ),
            .port3i             (port3i     ),
            .port0o             (port0o     ),
            .port1o             (port1o     ),
            .port2o             (port2o     ),
            .port3o             (port3o     ),

            .memaddr            (memaddr      ),
            .memdatao           (memdatao     ),
            .memdatai           (memdatai     ),
            .memack             (memack ),
            .memwr              (memwr        ),
            .memrd              (memrd        ),
            .scki               (       ),
            .scko               (scko),
            .scktri             (      ),
            .ssn                (1'b1 ),
            .misoi              (misoi ),
            .misoo              (      ),
            .misotri            (    ),
            .mosii              (      ),
            .mosio              (mosio ),
            .mositri            (    ),
            .spssn              (spssn )
      );
```

3.2 内部集成SPI FLASH

内部集成 SPI FLASH 的连接如下：

```
wire ssn;
wire misoi, misoo, misotri;
wire mosii, mosio, mositri;
wire scki, scko, scktri;
wire [7:0]spssn ;

spi_interface spi_interface_inst(
            .sclk(flash_sclk),
            .sdo(flash_sdo),
            .cson(flash_cson),
            .sdi(flash_sdi)
    );
mySystem    U_R8051XC(
            .clkcpu             (clk      ),
            .clkemif            (clk      ),
            .reset              (cpu_rst ),
            .swd(),
            .ro(),
            .port0i             (port0i     ),
            .port1i             (port1i     ),
            .port2i             (port2i     ),
            .port3i             (port3i     ),
            .port0o             (port0o     ),
            .port1o             (port1o     ),
            .port2o             (port2o     ),
            .port3o             (port3o     ),

            .memaddr            (memaddr    ),
            .memdatao           (memdatao   ),
            .memdatai           (memdatai   ),
            .memack             (memack ),
            .memwr              (memwr      ),
            .memrd              (memrd      ),
            .scki               (       ),
            .scko               (scko),
            .scktri             (     ),
            .ssn                (1'b1),
            .misoi              (misoi ),
            .misoo              (     ),
            .misotri            (    ),
            .mosii              (     ),
            .mosio              (mosio ),
            .mositri            (    ),
```

```
        .spssn                  (spssn )
    );
```

4 ISC实例

必须在设计里面嵌入 ISC 模块，才能在系统运行中进行 ISC 和在系统更新。

用户设计的 FPGA 设计可以相同或不同，8051firmware 设计也可以相同或不同，这取决于用户的设计。

下面以两个简单的设计来说明 ISC 的具体使用。实例是 FPGA 的设计不同，8051firmware 设计相同。板子加电后，系统运行，当按键按下触发 8051 的外部中断 0。中断 0 处理程序根据外部拨码开关的值选择设计 1 或 2 来运行。不同的设计用不同的 LED 显示来区分。

4.1 FPGA设计

4.1.1 设计 1

```
//dip
  input[1:0] sw_dip;
  //user button
  input[1:0] sw_user;
  //rs232
  input rs232_rx0;
  output rs232_tx0;
  //led
  output[3:0] led_user;

wire [7:0] port0i;
wire [7:0] port1i;
wire [7:0] port2i;
wire [7:0] port3i;
wire [7:0] port0o;
wire [7:0] port1o;
wire [7:0] port2o;
wire [7:0] port3o;

wire ssn;
wire misoi, misoo, misotri;
wire mosii, mosio, mositri;
wire scki, scko, scktri;
wire [7:0]spssn ;

    assign port3i[2] = ext_int0;
    assign port3i[3] = ext_int1;
```

```
        //dip
        assign port2i[1:0] = sw_dip;
        //user button
        assign port2i[5:4] = sw_user;
        //rs232 0
        assign port3i[0] = rs232_rx0;
        assign rs232_tx0 = port3o[1];
        //led
        assign led_user[3] = port2o[0] ;
        assign led_user[2] = port2o[1] ;
        assign led_user[1] = port2o[2] ;
        assign led_user[0] = cnt[23] ;
        //ext int
        assign ext_int0 = sw_user[1]; // int0 for isc

spi_interface spi_interface_inst(
                .sclk(flash_sclk),
                .sdo(flash_sdo),
                .cson(flash_cson),
                .sdi(flash_sdi)
    );
mySystem    U_R8051XC(
                .clkcpu             (clk      ),
                .clkemif            (clk      ),
                .reset              (cpu_rst ),
                .swd(),
                .ro(),
                .port0i             (port0i      ),
                .port1i             (port1i      ),
                .port2i             (port2i      ),
                .port3i             (port3i      ),
                .port0o             (port0o      ),
                .port1o             (port1o      ),
                .port2o             (port2o      ),
                .port3o             (port3o      ),

                .memaddr             (memaddr     ),
                .memdatao            (memdatao    ),
                .memdatai           (memdatai      ),
                .memack             (memack ),
                .memwr               (memwr        ),
                .memrd              (memrd         ),
                .scki                (        ),
                .scko                (scko),
```

```
        .scktri                 (     ),
        .ssn                    (1'b1),
        .misoi                  (misoi ),
        .misoo                  (     ),
        .misotri                (    ),
        .mosii                  (     ),
        .mosio                  (mosio ),
        .mositri                (    ),
        .spssn                  (spssn )
    );
外部按键连到中断 0
assign ext_int0 = sw_user[1];
中断 0 复用到 P3 口
assign port3i[2] = ext_int0;
```

4.1.2 设计 2

设计 2 与设计 1 的差别仅在 led 的显示不同：

```
 //led
   assign led_user[3] = cnt[23] ;
   assign led_user[2] = port2o[2] ;
   assign led_user[1] = port2o[1] ;
   assign led_user[0] = port2o[0] ;
```

4.2 Firmware设计

4.2.1 ISC函数

```
void    ISCReq (INT8U id)
{
      if(id <= 0xf)
   {
        ISCADDR0 = 0;
        ISCADDR1 = 0;
        ISCADDR2 = id;
        ISCADDR3 = 0;
   }
   else
   {
        ISCADDR0 = 0;
        ISCADDR1 = 0;
        ISCADDR2 = 0;
```

```
        ISCADDR3 = 0;
    }
    ISCEN = 1; // recfg
    _nop_();
}
```

函数 ISCReq 根据 id 设置新的配置映像地址，执行 ISCEN = 1 后，AstroII 就会从选定的地址里读取配置映像的数据配置 AstroII。8051 处于复位状态，直到新的配置完成后才重新开始运行。

4.2.2 Main程序

```
    EX0 = 1; //enable Ext Int 0
    EAL = 1;

    I2CSPISEL = I2CSPISEL | 0x1;// I2CSPISEL SFR 的第 0 位置 1，使能 SPI 接口
    while(1)
    {
        for(i = 0; i< 255 ; i++)
        {
            P2 = 0x00;
            延时 2 秒;
            P2 = 0xff;
            延时 3 秒;
        }
    }
```

主程序设置后无限循环，且 P2 输出波形。

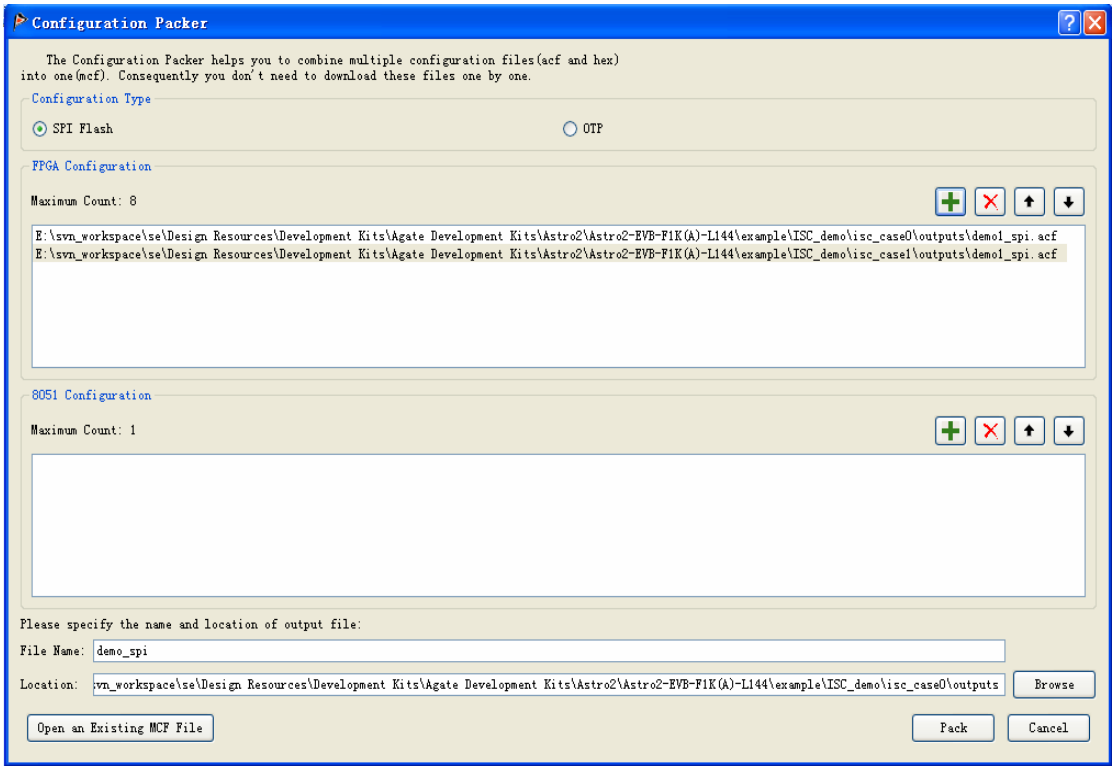
中断 0 中断处理程序如下：读取拨码开关的值，关中断，延时去抖动，调用 ISCReq 函数。

```
void isr_ExtInt0() interrupt 0
{

    id = (P2&0x3); // Get the external dip value to id,P2_0,P2_1
    EAL = 0;
    delay1ms(8000);      //delay for key jitter
    ISCReq(id);
    _nop_();
    EAL = 1;
}
```

4.3 操作步骤

1. 建立设计 1 和设计 2 工程，build 后生成 demo1_spi.acf 和 demo2_spi.acf。
2. 在 tools 底下用 configuration packer 工具生成 demo_spi.mcf。



3. download demo_spi.mcf。
4. 3 个 LED 同时闪（8051 控制），一个以不同的频率闪（FPGA 控制）。
5. 改变拨码开关的值，按下按键。LED 闪烁与前面的不一样，AstroII 切换到设计 2 运行。