Introduction

This migration guide is designed to help you use the enhanced features supported by the MH2103A device

Supported model list:

Supported model	MH2103Axxxx

Catalogue

The MH2103A series microcontrollers are basically compatible with the SXX32F103 series, while enhancing many features, which are detailed in this document.

1. Quickly replace the SXX32F103 chip

●Step 1: Compare peripheral specifications, Flash capacity, SRAM capacity, etc., unsolder SXX32F103 and replace it with the corresponding model MH2103A

●Step 2: Use ISP or KEIL to download the SXX32F103 HEX file or BIN file.

●Step 3: Download data other than SXX32F103 HEX file or BIN file or perform system corrections if necessary.

●Step 4: Check whether the program works properly.

●Step 5: For quick troubleshooting of other problems, refer to the migration manual.

●Step 6: If the program does not work properly after the above steps, please refer to the other sections of this document, or contact the distributor and MH support staff for assistance.

2. MH2103A features are enhanced

2.1 PLL high frequency configuration

●Description: The MH2103A's built-in PLL can output a 216MHz clock

●Example usage:

Refer to the MDK\ModuleDemo\RCC\RCC_ClockConfig project

```
void RCC_ClkConfiguration(void)
{
    RCC_DeInit();

    RCC_HSEConfig(RCC_HSE_ON);
    while(RCC_GetFlagStatus(RCC_FLAG_HSERDY) == RESET);

    RCC_PLLCmd(DISABLE);
    MH_RCC_PLLConfig(RCC_PLLSource_HSE_Div1,RCC_PLLMul_27,1);

    RCC_PLLCmd(ENABLE);
    while(RCC_GetFlagStatus(RCC_FLAG_PLLRDY) == RESET);

    RCC_SYSCLKConfig(RCC_SYSCLKSource_PLLCLK);

    RCC_HCLKConfig(RCC_SYSCLK_Div1);
    RCC_PCLK1Config(RCC_HCLK_Div2);
    RCC_PCLK2Config(RCC_HCLK_Div1);

    RCC_LSICmd(ENABLE);
    while(RCC_GetFlagStatus(RCC_FLAG_LSIRDY) == RESET);
    RCC_HSICmd(ENABLE);
    while(RCC_GetFlagStatus(RCC_FLAG_HSIRDY) == RESET);
}
```

2.2 GPIO Supports independent drop-down control

●Description: The MH2103A supports independent pull-down control (40K), which can replace external circuit resistors when IO is multiplexed

Eg:

(1) When using the SDIO module, D0-D3 and CMD can use internal pull-up resistors

(2) The use of IIC is, when the IIC rate is less than or equal to 100K, the internal pull-up resistor can be used

Pulling resistance

●Example usage:

Reference MH2103A\MDK\ModuleDemo\IIC\IIC_IntTransmit

```
// Enable the internal pull-up function
GPIO_ForcePuPdCmd(GPIOB, ENABLE);
GPIO_ForcePullUpConfig(GPIOB,GPIO_Pin_6);
GPIO_ForcePullUpConfig(GPIOB,GPIO_Pin_7);
```

2.3 1.5K pull-up resistor is optional inside USB

●Description: USB internal DP optional 1.5K pull-up resistance, can replace the external circuit pull resistance; And can achieve software re-enumeration (no need to PCB external triode control)

●Example usage:

Reference MH2103A\MDK\ModuleDemo\USB\Virtual_COM_Port

```
/*******************************************************************************/
/*                            General registers                                */
/*******************************************************************************/

/* Control register */
#define CNTR    ((__IO unsigned *)(RegBase + 0x40))
/* Interrupt status register */
#define ISTR    ((__IO unsigned *)(RegBase + 0x44))
/* Frame number register */
#define FNR     ((__IO unsigned *)(RegBase + 0x48))
/* Device address register */
#define DADDR   ((__IO unsigned *)(RegBase + 0x4C))
/* Buffer Table address register */
#define BTABLE  ((__IO unsigned *)(RegBase + 0x50))

#define DP_PUUP  *((__IO unsigned *)(RegBase + 0x54))
```

Open the pull-up:

```
DP_PUUP = 1;
```

2.4 USB supports 1/1.5/2/2.5/3/3.5/4/4.5x of the PLL clock as USB clock

●Description: Support 1/1.5/2/2.5/3/3.5/4/4.5 times of PLL clock as USB clock

●Example usage:

Reference MH2103A\MDK\ModuleDemo\USB\Virtual_COM_Port

```
void Set_USBClock(void)
{
    RCC_USBCLKConfig(RCC_USBCLKSource_PLLCLK_4Div5);
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_USB, ENABLE);
}
```

```
/** @defgroup USB_Device_clock_source
  * @{
  */
#define RCC_USBCLKSource_PLLCLK_Div4     ((uint32_t)0x80C00000)
#define RCC_USBCLKSource_PLLCLK_4Div5    ((uint32_t)0x80800000)
#define RCC_USBCLKSource_PLLCLK_Div3     ((uint32_t)0x80400000)
#define RCC_USBCLKSource_PLLCLK_3Div5    ((uint32_t)0x80000000)
#define RCC_USBCLKSource_PLLCLK_Div2     ((uint32_t)0x00C00000)
#define RCC_USBCLKSource_PLLCLK_2Div5    ((uint32_t)0x00800000)
#define RCC_USBCLKSource_PLLCLK_Div1     ((uint32_t)0x00400000)
#define RCC_USBCLKSource_PLLCLK_1Div5    ((uint32_t)0x00000000)
```

2.5 Supports multiple CRC modes

●Description: Supports multiple CRC modes

You can select whether the Byte byte is flipped, whether the calculation result is flipped,

whether the calculation result is 0xFFFFFFFF XOR, CRC16/32, or CRC16 polynomial

●Example usage:

Reference MH2103A\MDK\ModuleDemo\CRC\CRC_DifferentCrcMode

```
CRC_ResultInfo CRCResult[10] =
{
    {"CRC_16_IBM Calc", CRC_16_IBM_RESULT},
    {"CRC_16_MAXIM Calc", CRC_16_MAXIM_RESULT},
    {"CRC_16_USB Calc", CRC_16_USB_RESULT},
    {"CRC_16_MODBUS Calc", CRC_16_MODBUS_RESULT},
    {"CRC_16_CCITT Calc", CRC_16_CCITT_RESULT},
    {"CRC_16_CCITT_FALSE Calc", CRC_16_CCITT_FALSE_RESULT},
    {"CRC_16_X25 Calc", CRC_16_X25_RESULT},
    {"CRC_16_XMODEM Calc", CRC_16_XMODEM_RESULT},
    {"CRC_32 Calc", CRC_32_RESULT},
    {"CRC_32_MPEG_2 Calc", CRC_32_MPEG_2_RESULT},
};
```

2.6 MCO supports output PLL2-16 frequency division output

●Description: MCO supports output PLL 2-16 frequency division output

●Example usage:

Reference MH2103A\MDK\ModuleDemo\MCO\MCO_PllDiv

```
enum
{
    RCC_MCO_NoClock = 0x00,
    RCC_MCO_SYSCLK = 0x04,
    RCC_MCO_HSI,
    RCC_MCO_HSE,
    RCC_MCO_PLLCLK_Div2,
    RCC_MCO_PLLCLK_Div3,
    RCC_MCO_PLLCLK_Div4,
    RCC_MCO_PLLCLK_Div5,
    RCC_MCO_PLLCLK_Div6,
    RCC_MCO_PLLCLK_Div7,
    RCC_MCO_PLLCLK_Div8,
    RCC_MCO_PLLCLK_Div9,
    RCC_MCO_PLLCLK_Div10,
    RCC_MCO_PLLCLK_Div11,
    RCC_MCO_PLLCLK_Div12,
    RCC_MCO_PLLCLK_Div13,
    RCC_MCO_PLLCLK_Div14,
    RCC_MCO_PLLCLK_Div15,
    RCC_MCO_PLLCLK_Div16,
};
```

Historical version

Date	Edition	Alter
2021.10.13	1.00	Initial version