THIS SPEC IS OBSOLETE

Spec No: 002-05746

Spec Title: AN205326 - F2MC-8FX FAMILY MB95200H/210H SERIES WILD REGISTER FUNCTION(ZH)

Replaced by: None

AN205326

F²MC-8FX 家族 MB95200H/210H 系列 Wild 寄存器功能

该应用笔记介绍了如何使用 Wild 寄存器功能并描述如何通过样本代码使用该功能。

目录

1 概要 1
2 Wild 寄存器功能 1
2.1 Wild 寄存器功能 1
2.2 Wild 寄存器功能的框图 1
2.3 Wild 寄存器功能的寄存器 3
3 Wild 寄存器的软件配置 6
3.1 Wild 寄存器功能的设置步骤 6
3.2 软件设置流程 7
3.3 样本代码 8
3.4 软件调试环境 8
4 典型的硬件连接示例 9
4.1 典型的硬件连接 10
4.2 操作流程 10
4.3 固件 10
5 Wild 寄存器的使用注意事项 11
5.1 Wild 寄存器可使用的地址 11
5.2 软件的注意事项 11
6 更多信息 11
A 附录 12
A.1 样本代码 12
文档修改记录 14

OBSOLETE

1 概要

该应用笔记介绍了如何使用 Wild 寄存器功能。

该应用笔记介绍了 Wild 寄存器功能并描述如何通过样本代码使用该功能。

2 Wild 寄存器功能

本节介绍了 Wild 寄存器功能。

2.1 Wild 寄存器功能

Wild 寄存器包括 3 个 Wild 寄存器数据设置寄存器、3 个 Wild 寄存器地址设置寄存器、1 字节地址比较使能寄存器和 1 字节 Wild 寄存器数据测试设置寄存器。若将待修改地址和数据设置在这些寄存器内，ROM 中的数据就可被寄存器中的修改数据置换。最多可修改三个不同地址的数据。

使用 Wild 寄存器功能，可在掩模后调试程序并为程序中的缺陷打补丁。

2.2 Wild 寄存器功能的框图

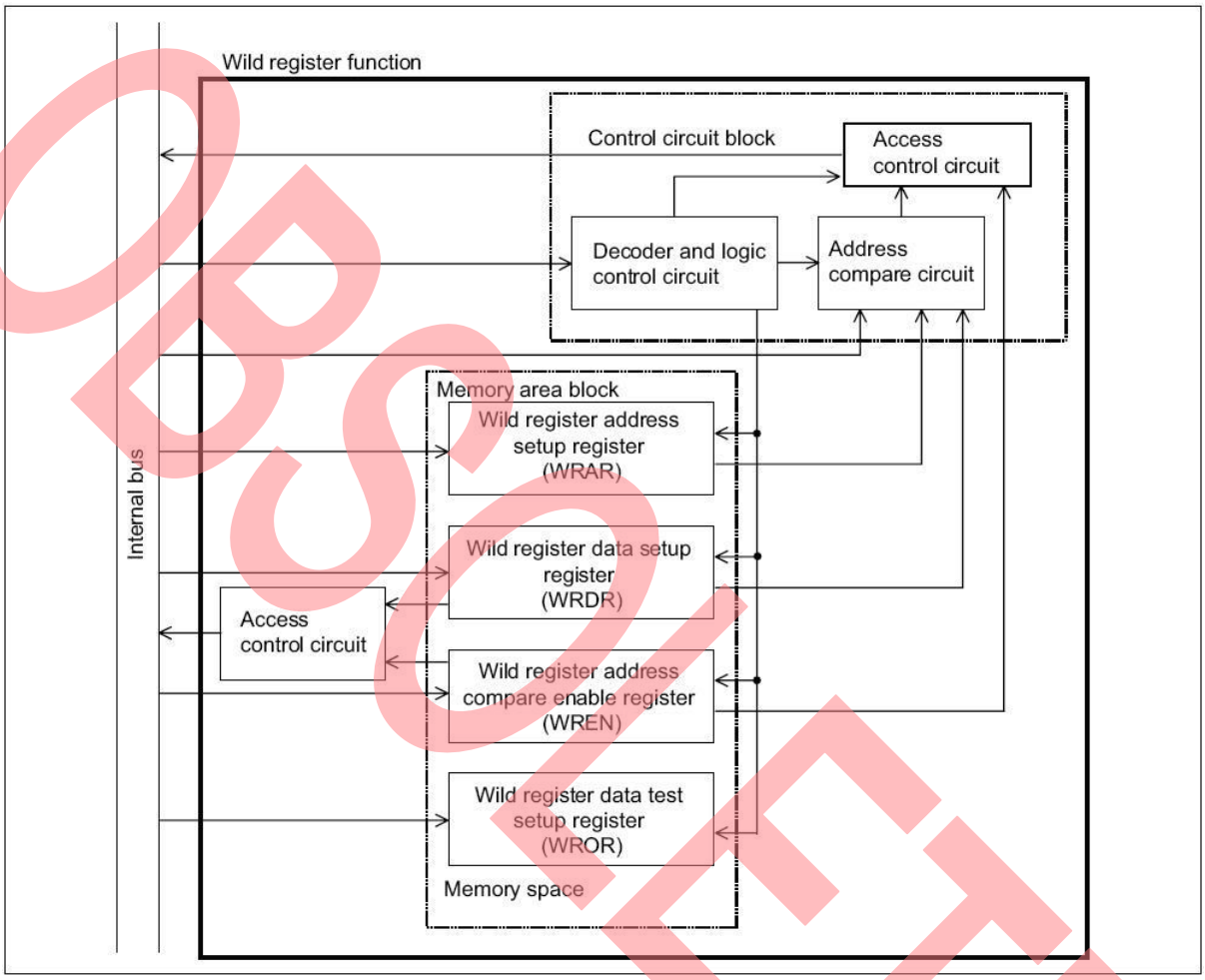
Wild 寄存器的框图如下所示。Wild 寄存器包括以下几个部分:

- 存储器区域部分
  Wild 寄存器数据设置寄存器(WRDR0 ~ WRDR2)
  Wild 寄存器地址设置寄存器(WRAR0 ~ WRAR2)
  Wild 寄存器地址比较使能寄存器(WREN)
  Wild 寄存器数据测试设置寄存器(WROR)

- 控制电路部分

图 1 是 Wild 寄存器功能的组成部分框图。

图 1. Wild 寄存器功能的框图



OBSOLETE

2.2.1 存储器区域部分

储存器区域部分由 Wild 寄存器数据设置寄存器(WRDR)、Wild 寄存器地址设置寄存器(WRAR)、Wild 寄存器地址比较使能寄存器(WREN)和 Wild 寄存器数据测试设置寄存器(WROR)构成。使用 Wild 寄存器功能，可指定需要置换的地址和数据。Wild 寄存器地址比较使能寄存器(WREN)启用各 Wild 寄存器数据设置寄存器(WRDR)的 Wild 寄存器功能。另外，Wild 寄存器数据测试设置寄存器(WROR)启用各 Wild 寄存器数据设置寄存器(WRDR)的正常读取功能。

2.2.2 控制电路部分

该电路将在 Wild 寄存器地址设置寄存器(WRAR)中设置的地址与实际地址比较。如果两个地址值匹配，该电路将数据从 Wild 寄存器数据设置寄存器(WRDR)输出到数据总线。该控制电路部分的操作由 Wild 寄存器地址比较使能寄存器(WREN)控制。

2.3 Wild 寄存器功能的寄存器

Wild 寄存器功能的寄存器包括 Wild 寄存器数据设置寄存器(WRDR)、Wild 寄存器地址设置寄存器(WRAR)、Wild 寄存器地址比较使能寄存器(WREN)和 Wild 寄存器数据测试设置寄存器(WROR)。

Wild 寄存器功能的寄存器一览

图 2. Wild 寄存器功能的寄存器一览

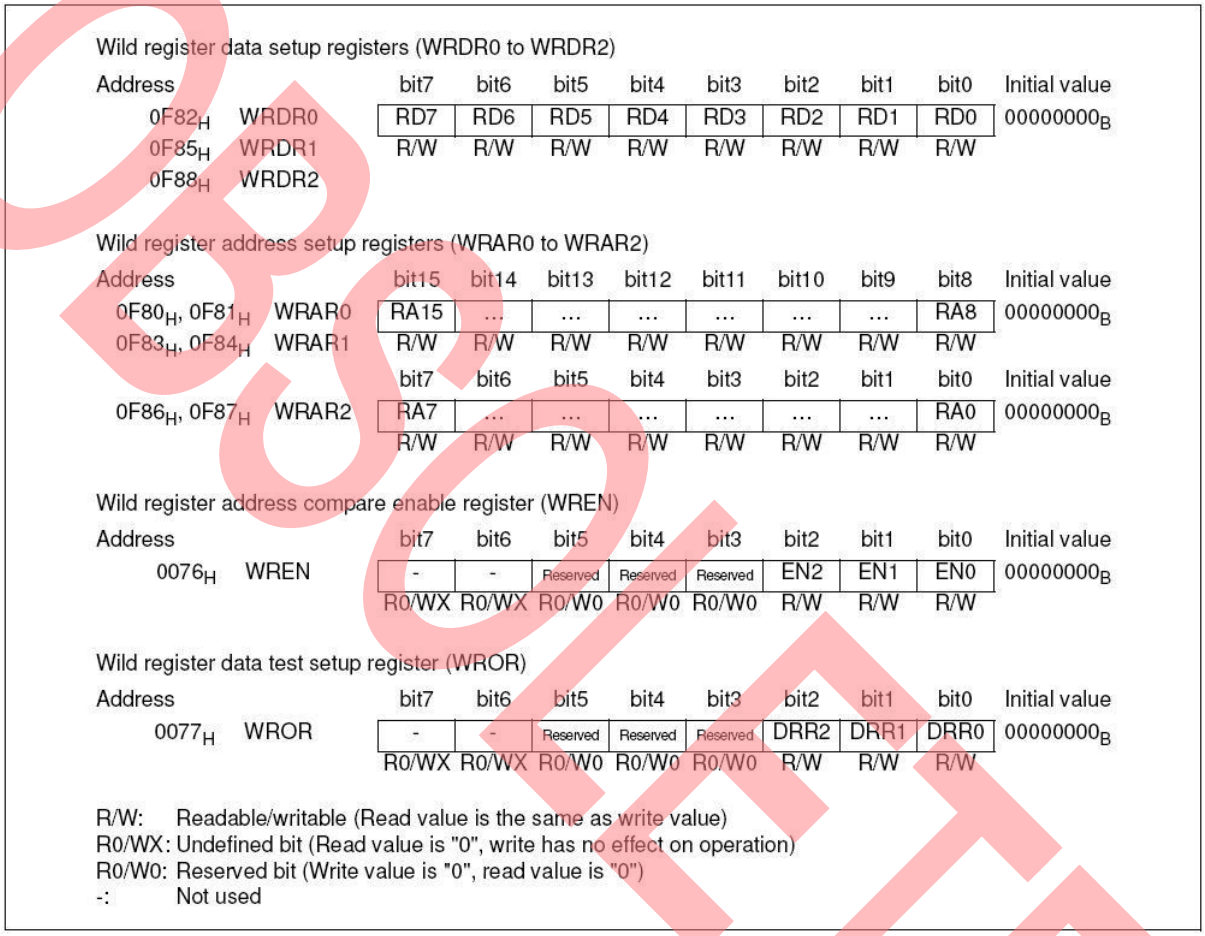


图 2 汇总了 Wild 寄存器功能的全部的寄存器。关于详细信息，参照 MB95200 系列的硬件手册第 13 章。

各 Wild 寄存器地址设置寄存器(WRAR)和 Wild 寄存器数据设置寄存器(WRDR)均有其对应的 Wild 寄存器号。

表 1. Wild 寄存器的对应号

Wild 寄存器号	Wild 寄存器地址设置寄存器(WRAR)	Wild 寄存器数据设置寄存器(WRDR)
0	WRAR0	WRDR0
1	WRAR1	WRDR1
2	WRAR2	WRDR2

OBSOLETE

2.3.1 Wild 寄存器数据设置寄存器(WRDR0 ~ WRDR2)

Wild 寄存器数据设置寄存器(WRDR0 ~ WRDR2)使用 Wild 寄存器功能指定待修改数据。

表 2. WRDR 的位功能

位名称		功能
WRDR0 WRDR1 WRDR2	bit7 ~ bit0	这些位使用 Wild 寄存器功能指定待修改的数据。 这些位用于将修改数据设置在 Wild 寄存器地址设置寄存器(WRAR)指定的地址上。在 Wild 寄存器号的对应地址上数据有效。 只有 Wild 寄存器数据测试设置寄存器(WROR)中对应数据测试设置的位为"1"时，读取这些位才会有效。

2.3.2 Wild 寄存器地址设置寄存器(WRAR0 ~ WRAR2)

Wild 寄存器地址设置寄存器(WRAR0 ~ WRAR2)设置需要 Wild 寄存器功能修改的地址。

表 3. WRAR 的位功能

位名称		功能
WRAR0 WRAR1 WRAR2	bit15 ~ bit0	这些位设置需要 Wild 寄存器功能修改的地址。 修改数据的指定地址设置为这些位。根据对应 Wild 寄存器号地址设置寄存器的 Wild 寄存器号指定地址。

OBSOLETE

2.3.3 Wild 寄存器地址比较使能寄存器(WREN)

Wild 寄存器地址比较使能寄存器(WREN)通过各自的 Wild 寄存器号允许/禁止 Wild 寄存器功能的操作。

表 4. WREN 的功能描述

位名称		功能
bit7, bit6	未定义位	这些位未定义。 读值为"0"。 对这些位写值无效。
bit5 ~ bit3	保留位	这些位是保留位。 读值为"0"。 始终将这些位清"0"。
bit2 ~ bit0	EN2, EN1, EN0： Wild 寄存器地址比较使能位	这些位允许/禁止 Wild 寄存器工作。 EN0 对应 Wild 寄存器号 0。 EN1 对应 Wild 寄存器号 1。 EN2 对应 Wild 寄存器号 2。 写"0": 禁止 Wild 寄存器功能工作。 写"1": 允许 Wild 寄存器功能工作。

OBSOLETE

2.3.4 Wild 寄存器数据测试设置寄存器(WROR)

Wild 寄存器数据测试设置寄存器(WROR)允许/禁止对应的 Wild 寄存器数据设置寄存器 (WRDR0 ~ WRDR2)进行读取。

表 5. WROR 的功能描述

位名称		功能
bit7, bit6	未定义位	这些位未定义。 读值为"0"。 对这些位写值无效。
bit5 ~ bit3	保留位	这些位是保留位。 读值为"0"。 始终将这些位清"0"。
bit2 ~ bit0	DRR2, DRR1,DRR0： Wild 寄存器数据测试设置位	这些位允许/禁止对应的 Wild 寄存器数据设置寄存器进行正常读取。 DRR0 允许/禁止 Wild 寄存器数据设置寄存器(WRDR0)读取。 DRR1 允许/禁止 Wild 寄存器数据设置寄存器(WRDR1)读取。 DRR2 允许/禁止 Wild 寄存器数据设置寄存器(WRDR2)读取。 写"0": 禁止读取。 写"1": 允许读取。

OBSOLETE

3 Wild 寄存器的软件配置

本节介绍了 Wild 寄存器功能的软件设置步骤。

3.1 Wild 寄存器功能的设置步骤

使用 Wild 寄存器功能前，要先在用户程序中准备可从外部存储器（例：EEPROM 或 FRAM）读取 Wild 寄存器值的程序。Wild 寄存器的设置方法如下。

本节不涉及外部存储器和器件间的通信方法。

- 对 Wild 寄存器地址设置寄存器(WRAR0 ~ WRAR2)写入待修改的内置 ROM 代码的地址。
- 对写入地址的 Wild 寄存器地址设置寄存器对应的 Wild 寄存器数据设置寄存器(WRDR0 ~ WRDR2)写入新的代码。
- 在对应 Wild 寄存器号的 Wild 寄存器地址比较使能寄存器(WREN)的 EN 位写"1"，以启用 Wild 寄存器号所代表的 Wild 寄存器功能。

表 6 列出了 Wild 寄存器的寄存器设置步骤。

表 6. Wild 寄存器的寄存器设置步骤

步骤	操作	操作例
1	从外部的外设功能通过某一通信方法读取置换数据。	例如待修改的内置 ROM 码在地址 F011H，待修改数据在"B5H"，有三个内置 ROM 码待修改。
2	对 Wild 寄存器地址设置寄存器(WRAR0 ~ WRAR2)写入置换地址。	将 Wild 寄存器设置寄存器设定为(WRAR0 = F011H, WRAR1 = ..., WRAR2 = ...)。
3	对 Wild 寄存器数据设置寄存器(WRDR0 ~ WRDR2)写入新的 ROM 码(置换内置 ROM 代码)。	将 Wild 寄存器数据设置寄存器设定为(WRDR0 = B5H, WRDR1 =..., WRDR2 =...)。
4	启用所用 Wild 寄存器功能的 Wild 寄存器号对应的 Wild 寄存器地址比较使能寄存器(WREN)的 EN 位。	将地址比较使能寄存器(WREN) 的 bit0 置"1"，可启用 Wild 寄存器 0 号对应的 Wild 寄存器功能。如果地址与地址 设置寄存器(WRAR)的设定值匹配，数据设置寄存器 (WRDR)的值置换内置 ROM 码。置换一个以上的内置 ROM 码时，需启用对应内置 ROM 码的 Wild 寄存器地址比较使能寄存器(WREN)的 EN 位。

3.2 软件设置流程

启用 Wild 寄存器功能，需做以下设置。

- 选择 Wild 寄存器号(0, 1, 2)
- 对 WRARx 写入地址
- 对 WRDRx 写入新代码
- 设置 WREN 以启用 Wild 寄存器功能
- 若地址匹配设置地址，则 Wild 寄存器功能修正指定代码。

OBSOLETE

3.3 样本代码

样本代码显示如何使用 Wild 寄存器置换指定地址的 ROM 数据。

```
/*****************************************************************************
*the example code set modified address and modified data
*WRAR2 = c081H, WRDR2 = 0x66, choose Wild register number 2
*****************************************************************************/

include "mb95200.h"

/*****************************************************************************
AME     :  vSysInit ()
UNCTION: Initialize Wild register. Choose WRAR2 work
*****************************************************************************/void
 vSysInit (void)

 WRARH2 = 0xC0;                 //ROM ADDH
RARL2 = 0x81;                   //ROM ADDL address = 0xC081
 WRDR2 = 0x66;                  //be modified data = 0x66
 WREN_EN2 = 1;                  //enable Wild 2
 WROR_DRR2 = 1;                 //enable reading


/*****************************************************************************
AME     :  main ()
UNCTION: Main loop, modified PDR0 = 0x66
*****************************************************************************/
oid main(void)

 vSysInit();                            //initialize function

 #pragma asm
        jmp    0xC080                   //jump to 0xC080
 #pragma endasm

 while(1)                               //main loop
 {

        #pragma asm
               .SECTION     Test, CODE, Locate = 0xC080
               MOV    A, #44H
               MOV __pdr0, A
/setting PDR0=0x44, but it will be modified 0x66
        #pragma endasm

        asm("\tNOP");
 }
```

注意：全体样本代码叫做工程。工程名为 Wild_Register。

3.4 软件调试环境

图 3 显示的是如何使用 Wild 寄存器设置指定地址和修正数据。

OBSOLETE

软件设置和调试流程：

初始化 Wild 寄存器：WRAR2 = 0xC081（修正地址），WRDR2 = 0x66（修正数据）。

在主循环中，地址 0xC081 设置为 Prot0 = 0x44。

进入运行，地址为 0xC081 时，Wild 寄存器设置 Prot0 = 0x66。

图 3. 调试环境

```
       23:   WRARH2 = 0xC0;          //ROM ADDH
C0FA: 04C0            MOV     A,#C0
C0FC: 610F86          MOV     >\__wrarh2,A
       24:   WRARL2 = 0x81;          //ROM ADDL address = C081H
C0FF: 0481            MOV     A,#81
C101: 610F87          MOV     >\__wrarl2,A
       25:   WRDR2 =  0x66;          //modified data = 66
C104: 0466            MOV     A,#66
C106: 610F88          MOV     >\__wrdr2,A
       26:   WREN_EN2 = 1;           //enable wild 2
C109: AA76            SETB    \__wren:02
       27:   WROR_DRR2 = 1;          //enable reading
C10B: AA77            SETB    \__wror:02
       28:
       29: }
C10D: 20              RET
       30:
       31: /*-------------------------------------------------------------------------
       32:    Name: main ();
       33:    Function: main loop, modified PDR0 = 0x66;
       34: -------------------------------------------------------------------------*/
       35:
       36: void main(void)
       37: {
       38:
       39:         vSysInit();                   //initial funciton
C10E: 31C0F4          CALL    \vSysInit
       40:
       41:         #pragma asm
       42:                 jmp     0xC080        //jump to 0xC080
C111: 21C080          JMP     C080
       43:         #pragma endasm
       44:
       45:
       46:         while(1)                      //main loop
       47:         {
       48:                 #pragma asm
       49:                 .SECTION        Test, CODE, Locate = 0xC080
       50:                 MOV     A,#44H
C080: 0444            MOV     A,#44
       51:                 MOV     __pdr0,A //setting port0 = 0x44,but modified 0x66
C082: 4500            MOV     \__pdr0,A
       52:                 #pragma endasm
       53:
       54:                 asm("\tNOP");         //delay timer
C084: 00              NOP
       55:
       56:         }
```

OBSOLETE

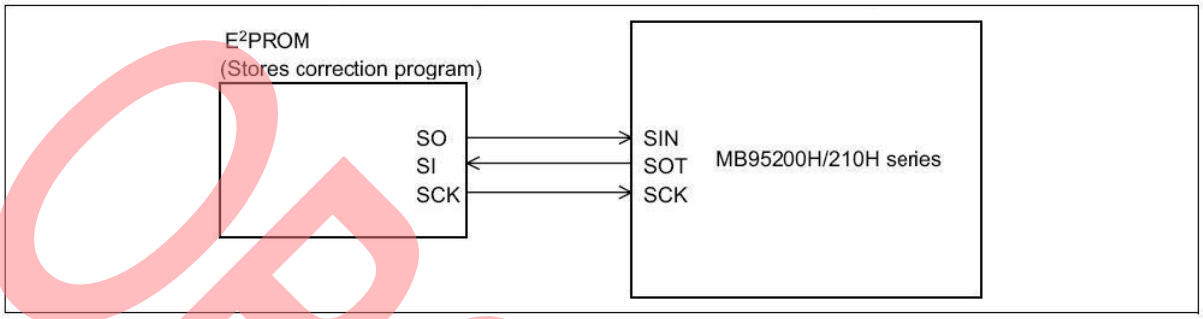
4 典型的硬件连接示例

本节介绍了 Wild 寄存器的硬件接口。

4.1 典型的硬件连接

下图是使用 Wild 寄存器功能时采用的典型硬件连接示例。

图 4. 典型硬件连接示例



注意：当 MB95200 连接 EEPROM 时，推荐使用该电路。

4.2 操作流程

MCU 连接 EEPROM 时的电路操作流程。

MCU 读取 EEPROM 数据
将 EEPROM 数据复制到 Wild 寄存器
启用 Wild 寄存器
修正数据

4.3 固件

需要包含以下固件。

EEPROM ：Wild 寄存器的修正地址
Wild 寄存器的修正数据
MB95200 ：与 EEPROM 通信的接口
Wild 寄存器模块初始化功能

OBSOLETE

5 Wild 寄存器的使用注意事项

本节介绍了 Wild 寄存器的使用注意事项。

5.1 Wild 寄存器可使用的地址

Wild 寄存器可用于地址空间除"0078H"以外的任何地址。因为地址"0078H"用作寄存器组指针的和直接存储区组指针的镜像地址，不可在该地址上打补丁。

5.2 软件的注意事项

使用 Wild 寄存器功能调试已掩膜处理过的程序并为程序打补丁时，要先预备好 MB95200 原始固件中的外部存储器通信接口。这样 MB95200 才可读取外部存储器（例如 EEPROM 或 FRAM）中的指定地址和待修正数据，然后设置 Wild 寄存器。

6 更多信息

如欲了解有关 Cypress 微控制器的更多详情，敬请访问以下网址：

www.cypress.com/documentation/application-notes/mb95200-wild-register

OBSOLETE

A 附录

A.1 样本代码

A.1.1 工程名: Wild_Register

名称： Wild_Register

功能： 样本代码将地址 0xC081 上待修改的内置 ROM 代码和待修改数据设置为"0x66"。

main.c

```
#include "mb95200.h"

/*****************************************************************************
NAME    :           vSysInit ()
FUNCTION:           Initialize Wild register. Choose WRAR2 work
*****************************************************************************/
void vSysInit(void)
{
        DDR0 = 0xff;                            //P0 output
        PDR0 = 0xff;                                    //port0 setting 0

        WRARH2 = 0xC0;                          //ROM ADDH
        WRARL2 = 0x81;                          //ROM ADDL address = 0xC081
        WRDR2 =  0x66;                          //modified data = 0x66
        WREN_EN2 = 1;                           //enable Wild register 2
        WROR_DRR2 = 1;                          //enable reading
}

/*****************************************************************************
NAME    :           main ()
FUNCTION:           Main loop, modified PDR0 = 0x66
*****************************************************************************/
void main(void)
{
        vSysInit();                                             //initialize function

        #pragma asm
                jmp     0xC080                          //jump to 0xC080
        #pragma endasm
```

OBSOLETE

```
        while(1)                                        //main loop
        {
                #pragma asm
                        .SECTION        Test, CODE, Locate = 0xC080
                        MOV     A, #44H
                        MOV     __pdr0, A
//setting port0 = 0x44, but will be modified to 0x66 by Wild register
                        #pragma endasm

                        asm("\tNOP");                   //delay timer
        }
}
```

OBSOLETE

文档修改记录

文档标题：AN205326 - F²MC-8FX 家族 MB95200H/210H 系列 Wild 寄存器功能

文档编号：002-05746

修订版	ECN	变更者	提交日期	变更说明
**	—	CBZH	03/20/2008	初稿
			07/22/2008	修改后的图像
*A	5346977	CBZH	07/28/2016	已将 Spansion 应用手册《MCU-AN-500008-Z-11》转换成 Cypress 格式。
*B	5731004	CBZH	05/11/2017	Obsolete the document.

OBSOLETE

全球销售和设计支持

赛普拉斯公司拥有一个由办事处、解决方案中心、厂商代表和经销商组成的全球性网络。如果想要查找离您最近的办事处，请访问赛普拉斯所在地。

产品

ARM® Cortex® 微控制器	cypress.com/arm
汽车级	cypress.com/automotive
时钟与缓冲器	cypress.com/clocks
接口	cypress.com/interface
照明和电源控制	cypress.com/powerpsoc
存储器	cypress.com/memory
PSoC	cypress.com/psoc
触摸感应	cypress.com/touch
USB 控制器	cypress.com/usb
无线/射频	cypress.com/wireless

PSoC® 解决方案

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

赛普拉斯开发者社区

论坛 | 项目 | 视频 | 博客 | 培训 | 组件

技术支持

cypress.com/support

PSoC 是赛普拉斯半导体公司的注册商标。PSoC Creator 是赛普拉斯半导体公司的商标。 此处引用的所有其他商标或注册商标都归其各自所有者所有。

赛普拉斯半导体
198 Champion Court
San Jose, CA 95134-1709
电话 : 408-943-2600
传真 : 408-943-4730
网站地址 : www.cypress.com

©赛普拉斯半导体公司，2008-2017 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属个人性质的、非独家且不可转让的如下许可（无再许可权）：（1）在赛普拉斯特软件著作权项下的下列许可权（一）对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和（二）仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码的形式向外部最终用户提供（无论直接提供或通过经销商和分销商间接提供），和（2）在被软件（由赛普拉斯公司提供，且未经修改）侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。
在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。在适用法律允许的限度内，赛普拉斯保留更改本文件的权利，届时将不另行通知。赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失的其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何索赔、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的索赔，并使之免受损失。
赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。敬请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。

OBSOLETE