MS MCU AUTO Emulation Tool Development Plan

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comment |
| V0.1 | 2018/03/19 | Dan.Wu | Initial version. |
| V0.2 | 2018/03/21 | Dan.Wu | Add测试项详细pattern： |
| V0.3 | 2018/3/30 | Dan.wu | 修改部分语句描述。  根据实现将详细测试pattern的描述，由plan该为实际实现。 |

## 概述：

MS MCU AUTO Emulation Tool（简称MCU tool）用于初步测试公司的800x系列MCU的 功能项，整个过程自动话完成，提高Emulation效率，缩短周期。

## 实现方案：

MCU tool 采用逐项测试的方法，每项测试框架采用相同的结构，方便修改、新增和维护。

## 系统框图：



## 测试项框架。

### 测试步骤

* 1. Update MCU firmware.
  2. Sent Pattern 1.
  3. Check result.
  4. Repeat Step2、Step3 until finish.

### Test pattern：

Test pattern是指MCU tool 通过串口发送给待测800x mcu的指令和判断标准，MCU收到指令后会输出pattern相应的结果，MCU tool再根据判断标准判定是PASS还是FAIl。

### 串口通讯：

收发有反馈，用于Check result Fail时区分未收到指令还是相应功能异常。具体如下：



### Source code 管理：

测试项相关代码放到项目文件夹下的“../User/ Test\_item”文件夹中。其中每个测试项可单独拥有原文件，共同享有同一个头文件Item\_Interface.h。

## 目前计划添加的测试项：

1. [GPIO & External interruption](#_GPIO_&_External)
2. Beep
3. PWM
4. Timer
5. UART
6. WDT
7. Power down mode
8. Information block
9. Reset
10. Read protection

## 测试项详细pattern：

### GPIO & External interruption

**MCU 程序（800x）：**

1. 可通过UART读写SFR和RAM。具体协议参考测试框架的 [串口通讯](#_串口通讯：)。
2. 外部中断服务函数，分别对每个Port口的中断标志进行判断，当存在有中断标志置起时记下中断标志值，并将此Port口对应的变量增加1。



关键程序：

1. RAM 定义

UINT8 idata g\_RAMP0IRQ \_at\_ 0x80**;**

UINT8 idata g\_RAMP1IRQ \_at\_ 0x81**;**

UINT8 idata g\_RAMP2IRQ \_at\_ 0x82**;**

UINT8 idata g\_RAMP0ExtCount \_at\_ 0x90**;**

UINT8 idata g\_RAMP1ExtCount \_at\_ 0x91**;**

UINT8 idata g\_RAMP2ExtCount \_at\_ 0x92**;**

1. 外部中断服务函数

void Ext\_interruptserver**()** interrupt EEXT0\_VECTOR

**{**

**if(**P0IRQ **!=** 0x00**)**

**{**

g\_RAMP0IRQ **=** P0IRQ**;**

g\_RAMP0ExtCount **++;**

P0IRQ **&=** 0x00**;**

**}**

**if(**P1IRQ **!=** 0x00**)**

**{**

g\_RAMP1IRQ **=** P1IRQ**;**

g\_RAMP1ExtCount **++;**

P1IRQ **&=** 0x00**;**

**}**

**if(**P2IRQ **!=** 0x00**)**

**{**

g\_RAMP2IRQ **=** P2IRQ**;**

g\_RAMP2ExtCount **++;**

P2IRQ **&=** 0x00**;**

**}**

**}**

**测试程序（STM32）：**

1. **GPIO输出功能：**
   1. 通过SFR配置MCU GPIO为输出功能，IO口依次输出0x00 -> 0xFF。
   2. STM32 实时检测IO口状态，判断是否符合预期。
2. **GPIO输入功能：**
   1. 通过SFR配置MCU GPIO为输入功能。
   2. STM32 测试用IO口依次输出0x00 -> 0xFF。
   3. STM32 实时检测SFR状态，判断是否符合预期。
3. **外部中断：**
4. 通过SFR使能MCU外部中断功能。
5. 测试前先清除MCU 外部中断用的全局变量。
6. 触发相应IO，检测全局变量是否符合预期（仅有触发IO对应的变量增加指定数量，且中断标志位正确）。

**不便测试项：**

1. 用于通讯的GPIO口。

### BEEP

**MCU 程序：**

1. 可通过UART读写SFR。

**测试程序：**

1. BEEP功能：
   1. 通过SFR配置使能MCU BEEP功能。
   2. STM32捕获BEEP输出频率，并判断是否符合预期。
   3. 通过SFR调节 BEEP频率，并重复步骤b。

**不便测试项：**

1. 无

### PWM

**MCU 程序：**

1. 可通过UART读写SFR。
2. PWM中断服务函数，完成IO口反转。

**测试程序：**

1. PWM功能：
   1. 通过SFR配置MCU PWM的频率、占空比、极性，并使能。
   2. STM32 捕获相应输出的频率、占空比、极性，判断是否符合预期。
2. PWM中断：
   1. 通过SFR使能PWM中断功能。
   2. STM32 捕获相应IO口反转频率是否符合预期。

**不便测试项：**

1. 无。

### Timer

**MCU 程序：**

1. 可通过UART读写SFR。
2. Timer中断服务函数，完成IO口反转。

**测试程序：**

1. 查询模式：
   1. 通过SFR配置MCU Timer功能，和检测标志位是否被置起、是否可通过SFR清除。
2. 中断模式：
   1. 通过SFR使能MCU Timer中断功能。
   2. STM32捕获相应IO输出频率，并判断是否符合预期。
   3. 通过SFR调节 Timer频率，并重复步骤b。
   4. 通过SFR更改Timer时钟源，并重复步骤b。

**不便测试项：**

1. 无。

### UART

**MCU 程序：**

1. 可通过UART读写SFR。
2. 中断服务函数完成将收到的数据原样发回。

**测试程序：**

1. 收发功能：
   1. 通过SFR分别配置UART到不同模式、几种常见波特率。
   2. STM32 配置到相同模式，发送接收数据，对比发送接收的数据是否相同。

**不便测试项：**

1. 用于SFR通讯的UART只能测试到一种模式，其他模式不方便测试。

### WDT

**MCU 程序：**

1. 程序初始化代码中利用串口发送诸如“Init Done”之类的字符到STM32；
2. 串口接收到进入WDT测试项指令后，配置SFR使其进入WDT工作状态；
3. 程序上不进行任何喂狗操作，等待MCU产生复位信号；

**测试程序：**

1. STM32通过串口配置MCU的SFR寄存器使其WDT进入工作状态；
2. 等待MCU串口是否发送诸如“Init Done”之类的字符以判断其有没有被WDT 复位

### PD Mode

**MCU 程序：**

* **Idle Mode**

1. Time 唤醒：

**测试程序：**

### Information Block

**MCU 程序：**

1. 程序初始化即将所有的信息区域对应的SFR打印出来判断；

**测试程序：**

1. 进入该测试项后需要给对应的信息区域写入一个值；
2. 随后给MCU的RST引脚一个高低脉冲，串口接收数据判断相应的SFR是否被更新正确的值；

### Reset

**MCU 程序：**

1. 初始化程序设置让其打印RSTSTAT的值；
2. 紧接着加入判断，如果检测到PORF标志位为1（代表是上电复位），就将RSTSTAT寄存器清零；
3. 打印RSTSTAT的值（消除POR复位对其他复位标志位的不确定控制）；

**测试程序：**

1. 判断上传上来的RSTSTAT的值是否符合预期即可；

### Read Protection

**MCU 程序：**

1. 无；

**测试程序：**

1. 所有测试项结束后进行该项测试，将信息区域对应的位置写1使能读保护；
2. 给RST一个高低脉冲；
3. 使用ISP方式读取OTP，判断读到的数据是否为全0；
4. 使用密钥往指定地址写数据，看能否解除保护；