

راهنمای استفاده از کتابخانه های سیستم های نهفته

(کتابخانه i2c_memory)

آخرین تغییرات :

Wednesday, July 10, 2019

نگارنده کد و پشتیبان : مجید درهم بفش (Majid Derhambakhsh)



[GitHub : Majid-Derhambakhsh](#)

[E-mail](#)

فهرست

3 ----- راهنمای استفاده از راه انداز i2c_memory

تنظیمات اولیه

توابع

۷ ----- نیازمندی ها

۷ ----- نکات مهم

۸ ----- خطا و اختار ها

۹ ----- حافظه های پشتیبانی شده

۱۰ ----- میکروکنترلر های پشتیبانی شده

۱۱ ----- تغییرات نسخه

راهنمای استفاده

برای استفاده از این کتابخانه ابتدا فایل c آنرا به برنامه اضافه نمایید و هدر زیر را وارد نمایید.

```
#include "i2c_memory.h"
```

تنظیمات حافظه

جهت تنظیم کتابخانه مراحل زیر را دنبال کنید :

۱ . ابتدا فایل **i2c_memory_conf.h** را باز نمایید :

جهت تنظیم کد آی سی مورد استفاده و پین های آدرس آن مقادیر زیر را ویرایش کنید:

```
#define _AT24C04

#define _MEMORY_A0_PIN_STATE 0
#define _MEMORY_A1_PIN_STATE 0
#define _MEMORY_A2_PIN_STATE 0
```

-- شماره آی سی مورد استفاده (موجود در بخش حافظه های پشتیبانی شده).

-- منطق پین های آدرس حافظه (۰ یا ۱).

جهت تنظیم حافظت در برابر نوشتن مقادیر زیر را ویرایش کنید:

```
#define _WP_DISABLE

#define _WRITE_PROTECT_PORT GPIOx
#define _WRITE_PROTECT_PIN 0
```

-- غیر فعال ساز حفاظت در برابر نوشتن (جهت فعال سازی، این مورد را کامنت کنید).

-- پورت و پین متصل به پین حفاظت در برابر نوشتن حافظه :

مقدار قابل قبول پورت : **GPIOx** در **ARM** یا **PORTx** در **AVR** .

مقدار قابل قبول پین : ۰ ~ تعداد پین های پشتیبانی شده هر پورت.

تنظیمات i2c

جهت تنظیم i2c مراحل زیر را دنبال کنید :

۱. i2c در AVR :

ابتدا فایل **i2c_unit_conf.h** درون فولدر **I2C_UNIT** را باز نموده و مقادیر زیر را ویرایش نمایید :

```
#define _F_SCL 100000UL  
  
#define _PRESCALER _PRE1
```

-- فرکانس کارکرد i2c .

-- تقسیم کننده وامد i2c (در صورت عدم عملکرد صمیع با مقادیر جدول ۱-۲ ویرایش شود) :

مقدار	شناسه
1	_PRE1
4	_PRE4
16	_PRE16
64	_PRE64

جدول ۱-۲

نکته : در کامپایلر های AVR فایل **i2c_unit.c** و **gpio_unit.c** را به برنامه اضافه کنید.

۲. i2c در ARM :

سری STM32 :

ابتدا فایل **stm32_i2c_conf.h** درون فولدر **STM32_I2C** را باز نموده و مقادیر زیر را ویرایش نمایید :

```
#define STM32F1  
  
#define _CONNECTED_I2C hi2c1  
  
#define _MEM_DEF_VAL_BUFF_LENGTH 50
```

-- سری میکروکنترلر مورد استفاده .

-- استراکت مورد استفاده جهت دسترسی به i2c .

-- سائز بافر جهت پاک کردن حافظه (طول بیشتر سرعت بیشتر).

نکته : در کامپایلر های ARM فایل **stm32_i2c.c** را به برنامه اضافه کنید.

توابع

I2C_Memory_Init

نام تابع `void I2C_Memory_Init(void)`

شرح تابع این تابع جهت راه اندازی حافظه مورد استفاده قرار می گیرد.

پارامتر ها -

مقادیر بازگشتی -

مثال :

```
I2C_Memory_Init();
```

I2C_MemoryIsReady

نام تابع `uint8_t I2C_MemoryIsReady(uint16_t time_out)`

شرح تابع این تابع جهت بررسی در دسترس بودن حافظه مورد استفاده قرار می گیرد.

پارامتر ها

• `time_out` : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

• `_STAT_OK / _STAT_ERROR : AVR`

• `HAL_OK / HAL_ERROR : ARM`

مثال :

```
device_status = I2C_MemoryIsReady(100);
```

I2C_Memory_SingleWrite

نام تابع `uint8_t I2C_Memory_SingleWrite(uint32_t address,uint8_t udata,uint16_t time_out)`

شرح تابع این تابع جهت نوشتن یک بایت دیتا بر روی حافظه مورد استفاده قرار می گیرد.

پارامتر ها

• `address` : آدرس حافظه جهت نوشتن دیتا.

• `udata` : دیتای مورد نظر برای نوشتن بر روی حافظه.

• `time_out` : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

• `_MEM_SIZE_ERROR`

• `_STAT_OK / _STAT_ERROR : AVR`

• `HAL_OK / HAL_ERROR : ARM`

مثال :

```
com_status = I2C_Memory_SingleWrite(0,'A',50);
```

I2C_Memory_BurstWrite

نام تابع `uint8_t I2C_Memory_BurstWrite(uint32_t address, uint8_t *udata, uint32_t size, uint16_t time_out)`

شرح تابع این تابع جهت نوشتن یک رشته از دیتا بر روی حافظه مورد استفاده قرار می گیرد.

پارامتر ها

- **address** : آدرس حافظه جهت نوشتن دیتا.
- **udata** : دیتا های مورد نظر برای نوشتن بر روی حافظه.
- **size** : مقدار دیتا ها جهت ارسال به حافظه.
- **time_out** : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

- `_MEM_SIZE_ERROR`
- `_STAT_OK / _STAT_ERROR : AVR`
- `HAL_OK / HAL_ERROR : ARM`

مثال :

```
com_status = I2C_Memory_BurstWrite(0, "Hello", 5, 50);
```

I2C_Memory_SingleRead

نام تابع `uint8_t I2C_Memory_SingleRead(uint32_t address, uint8_t *udata, uint16_t time_out)`

شرح تابع این تابع جهت خواندن یک بایت دیتا از حافظه مورد استفاده قرار می گیرد.

پارامتر ها

- **address** : آدرس حافظه جهت خواندن دیتا.
- **udata** : اشاره گر به متغیر مورد نظر جهت ذخیره دیتای دریافت شده از حافظه.
- **time_out** : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

- `_MEM_SIZE_ERROR`
- `_STAT_OK / _STAT_ERROR : AVR`
- `HAL_OK / HAL_ERROR : ARM`

مثال :

```
com_status = I2C_Memory_SingleRead(0, &my_data, 50);
```

I2C_Memory_BurstRead

نام تابع `uint8_t I2C_Memory_BurstRead(uint32_t address, uint8_t *udata, uint32_t size, uint16_t time_out)`

شرح تابع این تابع جهت خواندن یک رشته دیتا از حافظه مورد استفاده قرار می گیرد.

پارامتر ها

- **address** : آدرس حافظه جهت خواندن دیتا.
- **udata** : اشاره گر به رشته مورد نظر جهت ذخیره دیتا های دریافت شده از حافظه.
- **size** : مقدار دیتا ها جهت خواندن از حافظه.

- `time_out` : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

- `_MEM_SIZE_ERROR`
- `_STAT_OK / _STAT_ERROR : AVR`
- `HAL_OK / HAL_ERROR : ARM`

مثال :

```
com_status = I2C_Memory_BurstRead(12,received_data_array,10,50);
```

I2C_Memory_Erase

نام تابع `uint8_t I2C_Memory_Erase(uint32_t address,uint32_t quantity,uint16_t time_out)`

شرح تابع این تابع جهت پاک کردن دیتا از حافظه مورد استفاده قرار می گیرد.

پارامتر ها

- `address` : آدرس شروع حافظه جهت پاک کردن دیتا.
- `quantity` : تعداد پاکسازی دیتا.
- `time_out` : مدت زمان فروغ در صورت عدم پاسخ.

مقادیر بازگشتی

- `_MEM_SIZE_ERROR`
- `_STAT_OK / _STAT_ERROR : AVR`
- `HAL_OK / HAL_ERROR : ARM`

مثال :

```
com_status = I2C_Memory_Erase(0,65000,50);
```

نیازمندی ها

- درایور `i2c_unit` و `gpio_unit` برای میکروکنترلر های AVR .
- درایور های `HAL` و `stm32_i2c` برای میکروکنترلر های ARM سری STM32 .

نکات مهم

- کلیه دستورات و تنظیمات با `_` شروع می شوند.
- کلیه توابع به صورت کامل کیس نوشته شده اند. (شروع هر کلمه با حرف بزرگ).
- کدهای مورد استفاده جهت راه اندازی و استفاده در میکروکنترلر های مختلف یکسان می باشد.

خطا و افتارها

خطاها :

- Chip or I2C Library not supported : این خطا زمانی نمایان می شود که میکروکنترلر یا کتابخانه آن پشتیبانی نشود.

- Memory is not selected Or not supported : این خطا زمانی نمایان می شود که حافظه به درستی تعیین نشده باشد و یا توسط کتابخانه پشتیبانی نشود.

افتارها :

- Your Ax Pin state in not correct : این افتار زمانی نمایان می شود که پین آدرس اعلام شده حافظه درست تنظیم نشده باشد.

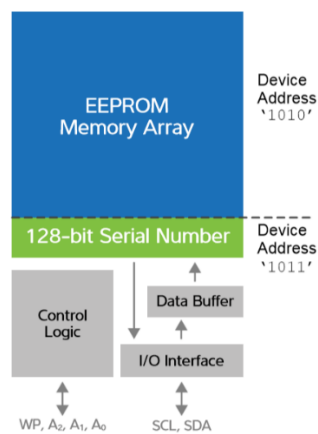
حافظه های پشتیبانی شده



کلیه حافظه های خانواده 24Cxx شامل :

AT24C01 – AT24C02 – AT24C04 – AT24C08 – AT24C16 – AT24C32 – AT24C64 – AT24C128 – AT24C256 – AT24C512

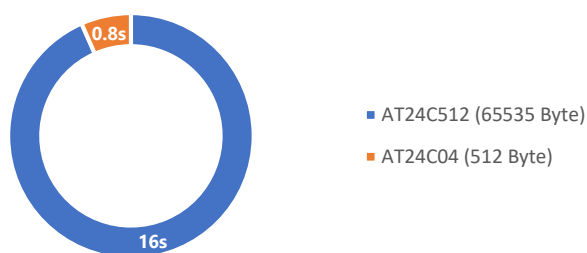
AT24C1024



بلوک حافظه

سرعت نوشتن داده ها

سرعت نوشتن بر روی حافظه با استفاده از میکروکنترلر ARM با کلاک ۷۲ مگاهرتز و فرکانس ۴۰۰ کیلوهرتز در I2C



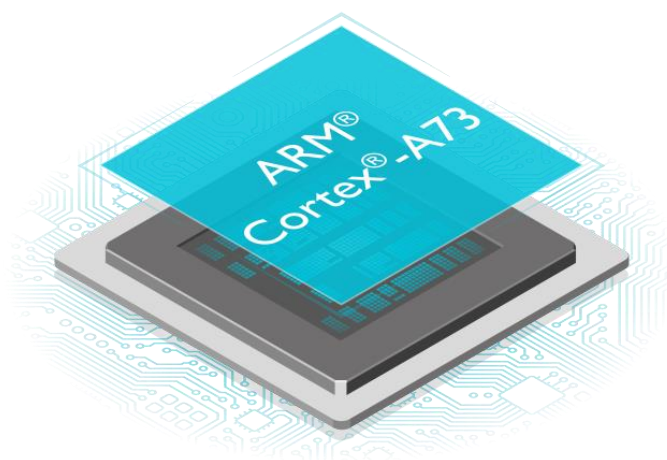
میکروکنترلر های پشتیبانی شده



میکروکنترلر های AVR سری tiny – mega توسط توابع i2c_unit و gpio_unit

کامپایلر های Codevision و GNUC نظیر AtmelStudio

codevision[®]



کلیه میکروکنترلر های ARM سری STM توسط توابع HAL و stm32_i2c

کلیه کامپایلر های ARM



IAR Embedded Workbench

تغییرات نسخه

نسخه ۰,۰,۰

نسخه پایدار و تست شده