# レジスタ

テーブル

自動的に生成された説明

テーブル

自動的に生成された説明

基本と拡張1の違 : RAMがなくなって，10msのカウンタが追加

テーブル

自動的に生成された説明

テーブル

中程度の精度で自動的に生成された説明

# 電源関係

テーブル

自動的に生成された説明

テーブル が含まれている画像

自動的に生成された説明

### int RTC\_RV8803\_U::checkLowPower(void)

* 0E番レジスタの0,1bitを読み出して返す

### int RTC\_RV8803\_U::clearPowerFlag(void)

* 0E番レジスタの0,1bitに0を書き込み

## テスト

#### 前提 : 電源OFFからスタート

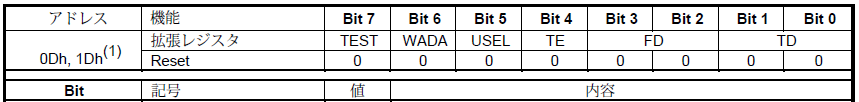
### checkLowPower()の出力とレジスタの内容を比較 (なにが起きるかは実機で要調査)

### clearPowerFlag()を実行し，レジスタの内容を確認

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xxxxxx00 | 00000011 | 0 |

# クロック信号出力





テーブル

自動的に生成された説明

### int RTC\_RV8803\_U::setClockOut(uint8\_t num, uint8\_t freq, int8\_t pin)

* num >= RTC\_RV8803\_NUM\_OF\_CLOCKOUTでエラー(RTC\_U\_ILLEGAL\_PARAM)
* clkoeにpinを代入して，clkoe端子を出力用に設定
* setClockOutMode(num, freq)を実行
* controlClockOut(num, 1)を実行

### int RTC\_RV8803\_U::setClockOutMode(uint8\_t num, uint8\_t freq)

* num >= RTC\_RV8803\_NUM\_OF\_CLOCKOUTでエラー(RTC\_U\_ILLEGAL\_PARAM)
* freqが3より大きい場合はエラー(RTC\_U\_ILLEGAL\_PARAM)
* freqの値を左2bitシフトして，0x0D番レジスタのFD(2,3bit)に上書き

### int RTC\_RV8803\_U::controlClockOut(uint8\_t num, uint8\_t mode)

* num >= RTC\_RV8803\_NUM\_OF\_CLOCKOUTでエラー(RTC\_U\_ILLEGAL\_PARAM)
* numが1より大きい場合はエラー(RTC\_U\_ILLEGAL\_PARAM)
* mode=1のとき，clkoeピンをHIGHに設定
* mode=0のとき，clkoeピンをLOWに設定

## **テスト**

### クロック出力設定を実施し，レジスタの内容を確認

* setClockOut(0,0b10,PIN\_NUM)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0D | xxxx10xx | 00001100 | 1000 |

### デフォルト値に戻して，レジスタの内容を確認

* controlClockOut(0, 0)
* setClockOutMode(0,0)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0D | xxxx00xx | 00001100 | 0 |

# 時計の進み具合の制御

## 発振周波数

グラフィカル ユーザー インターフェイス, アプリケーション, テーブル, Excel

自動的に生成された説明

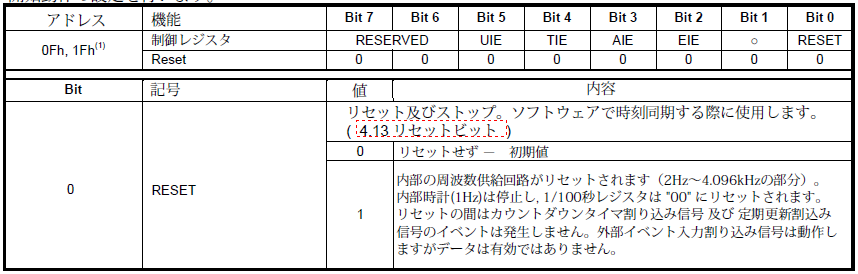
### int RTC\_RV8803\_U::setOscillator(uint8\_t mode)

* modeが63より大きい場合はエラー(RTC\_U\_ILLEGAL\_PARAM)
* 2C番レジスタの0から5bitにmodeの値を上書き

### int RTC\_RV8803\_U::getOscillator(void)

* 2C番レジスタの0から5bitまでを返す

## クロック調整



### int RTC\_RV8803\_U::controlClock(void)

* 0F番レジスタの最下位bitに1を書き込み

## テスト

### setOscillator(0b101010)を実行

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x2C | xx111111 | 00111111 | 101010 |

### getOscillator()の出力が0b101010であることを確認

### setOscillator(0)を実行

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x2C | xx000000 | 00111111 | 0 |

### getOscillator()の出力が0であることを確認

# 割り込み

グラフィカル ユーザー インターフェイス, テーブル

中程度の精度で自動的に生成された説明

テキスト

自動的に生成された説明

### int RTC\_RV8803\_U::checkInterupt(void)

* 0E番レジスタの2から5bitの値を返す(右シフトする)

### int RTC\_RV8803\_U::clearInterupt(uint16\_t type)

* typeが15より大きいとエラー(RTC\_U\_ILLEGAL\_PARAM)
* 左2ビットシフトして，0E番レジスタの2から5bitに上書き

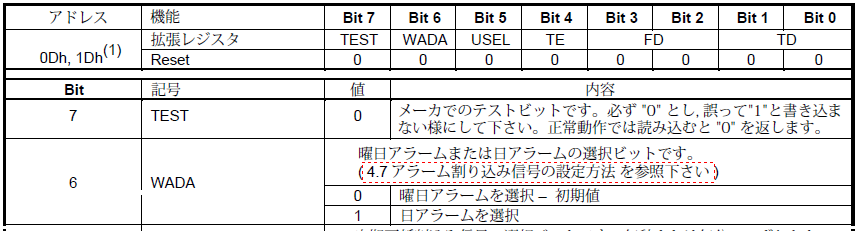
# アラーム

テーブル

中程度の精度で自動的に生成された説明

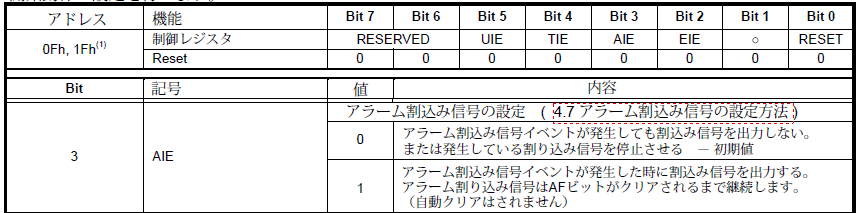
テーブル

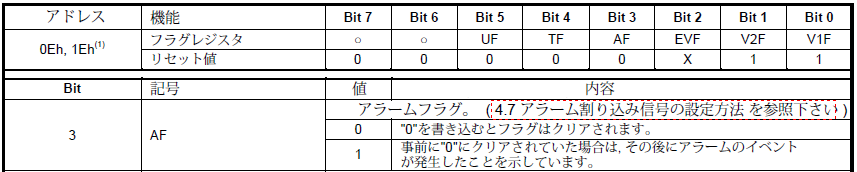
自動的に生成された説明



グラフィカル ユーザー インターフェイス, アプリケーション, テーブル

自動的に生成された説明





テキスト, 手紙

自動的に生成された説明

### int RTC\_RV8803\_U::setAlarm(uint8\_t num, alarm\_mode\_t \* mode, date\_t\* timing)

* num >= RTC\_RV8803\_NUM\_OF\_ALARMの場合はエラー(RTC\_U\_ILLEGAL\_PARAM)
* アラーム停止
* timingの処理
* setAlarmMode(num, mode)の実行
* controlAlarm(num, 1)の実行

#### アラーム停止

|  |  |  |  |
| --- | --- | --- | --- |
| データ名 | レジスタ番号 | bit | 値 |
| AIE | 0x0F | 3 | 0 |
| AF | 0x0E | 3 | 0 |

#### 週/日付アラームの判定方法

|  |  |  |  |
| --- | --- | --- | --- |
| メンバ変数 | データ名 | レジスタ | bit |
| mode-> type | WADA | 0x0D番 | 6 |
| 値 | 意味 |  |  |
| 0 | 曜日アラーム |  |  |
| 1 | 日付アラーム |  |  |

|  |  |
| --- | --- |
| timing->mday | WADA(0x0D番レジスタ, 6bit) |
| 31より大 | 週/日共に無効(0x0A番レジスタを0b10000000) |
| 31以下 | WADA(0x0D番レジスタ, 6bit)=1, timing->mdayを0x0A番レジスタ |

|  |  |
| --- | --- |
| timing->wday | WADA(0x0D番レジスタ, 6bit) |
| 6より大 | 週/日共に無効(0x0A番レジスタを0b10000000) |
| 6以下 | timing->wdayを0x0A番レジスタ |

|  |  |  |
| --- | --- | --- |
| timing->mday | timing->wday | WADA(0x0D番レジスタ, 6bit) |
| 31より大 | 6より大 | 週/日共に無効(0x0A番レジスタを0b10000000) |
| 31より大 | 6以下 | WADA(0x0D番レジスタ, 6bit)=0, timing->wdayを0x0A番レジスタ |
| 31以下 | 6より大 | WADA(0x0D番レジスタ, 6bit)=1, timing->mdayを0x0A番レジスタ |
| 31以下 | 6以下 | エラー(RTC\_U\_ILLEGAL\_PARAM) |

|  |  |  |
| --- | --- | --- |
| 変数 | 値 | 0x08番レジスタ |
| timing->minute | 60より大 | 0b10000000 |
| 60以下 | timing->minute(BCD) |
| timing->hour | 60より大 | 0b10000000 |
| 60以下 | timing->hour(BCD) |

#### 週アラームの場合の処理

|  |  |
| --- | --- |
| timing->wdayの値 | 0x0Aレジスタの値 |
| 0 | 0x00000001 |
| 1 | 0x00000010 |
| 2 | 0x00000100 |
| 3 | 0x00001000 |
| 4 | 0x00010000 |
| 5 | 0x00100000 |
| 6 | 0x01000000 |

#### アラーム再開

|  |  |  |  |
| --- | --- | --- | --- |
| データ名 | レジスタ番号 | bit | 値 |
| AIE | 0x0F | 3bit | 1 |

### int RTC\_RV8803\_U::setAlarmMode(uint8\_t num, alarm\_mode\_t \* mode)

|  |  |  |  |
| --- | --- | --- | --- |
| メンバ変数 | データ名 | レジスタ | bit |
| mode-> useInteruptPin | AIE | 0x0F番 | 3 |

### int RTC\_RV8803\_U::controlAlarm(uint8\_t num, uint8\_t action)

|  |  |
| --- | --- |
| actionの値 | 処理 |
| 0 | 0x08, 0x09, 0x0A番レジスタを0b10000000に書き換え |
| 1 | バックアップ用変数(alarm\_min, alarm\_hour, alarm\_day)の値を0x08, 0x09, 0x0A番レジスタに上書き |

## テスト

### 割り込みフラグをすべて消す : clearInterupt(0b1111)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xx0000xx | 00111100 | 0 |

### 現在時刻を取得して+2分でアラームを設定(日付け選択)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | レジスタ | 値 | マスク | 値 |
| 分 | 0x08 | 計算する | 11111111 | 計算する |
| 時 | 0x09 | 計算する | 11111111 | 計算する |
| 日/曜日 | 0x0A | 計算する | 11111111 | 計算する |
| 日/曜日選択 | 0x0D | x1xxxxxx | 01000000 | 1000000 |
| 割り込みピン有効 | 0x0F | xxxx1xxx | 00001000 | 1000 |

### 150秒待ち，割り込みが発生したかを確認

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xxxx1xxx | 00001000 | 1000 |

### 割り込みフラグをすべて消す : clearInterupt(0b1111)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xx0000xx | 00111100 | 0 |

### 現在時刻を取得して+2分でアラームを設定(曜日選択)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | レジスタ | 値 | マスク | 値 |
| 分 | 0x08 | 計算する | 11111111 | 計算する |
| 時 | 0x09 | 計算する | 11111111 | 計算する |
| 日/曜日 | 0x0A | 計算する | 11111111 | 計算する |
| 日/曜日選択 | 0x0D | x1xxxxxx | 01000000 | 0 |
| 割り込みピン有効 | 0x0F | xxxx1xxx | 00001000 | 1000 |

### アラームをストップcontrolAlarm(0,0)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | レジスタ | 値 | マスク | 値 |
| 分 | 0x08 | 1xxxxxxx | 10000000 | 10000000 |
| 時 | 0x09 | 1xxxxxxx | 10000000 | 10000000 |
| 日/曜日 | 0x0A | 1xxxxxxx | 10000000 | 10000000 |

# タイマ

### int RTC\_RV8803\_U::setTimer(uint8\_t num, timer\_mode\_t \* mode, uint16\_t multi)

|  |  |
| --- | --- |
| Numの値 | 処理 |
| 0 | setIntervalTimer(timer\_mode\_t \* mode) |
| 1 | setCountdownTimer(timer\_mode\_t \* mode, uint16\_t multi) |
| その他 | RTC\_U\_ILLEGAL\_PARAMを返す |

### int RTC\_RV8803\_U::setTimerMode(uint8\_t num, timer\_mode\_t \* mode)

|  |  |
| --- | --- |
| Numの値 | 処理 |
| 0 | setIntervalTimerMode(timer\_mode\_t \* mode) |
| 1 | setCountdownTimerMode(timer\_mode\_t \* mode) |
| その他 | RTC\_U\_ILLEGAL\_PARAMを返す |

### int RTC\_RV8803\_U::controlTimer(uint8\_t num, uint8\_t action)

|  |  |
| --- | --- |
| Numの値 | 処理 |
| 0 | controlIntervalTimer(uint8\_t action) |
| 1 | controlCountdownTimer(uint8\_t action) |
| その他 | RTC\_U\_ILLEGAL\_PARAMを返す |

## 定期周期タイマ

テーブル が含まれている画像

自動的に生成された説明

グラフィカル ユーザー インターフェイス, テーブル

自動的に生成された説明

グラフィカル ユーザー インターフェイス, テーブル

自動的に生成された説明

テキスト

自動的に生成された説明

### int RTC\_RV8803\_U::setIntervalTimer(timer\_mode\_t \* mode)

* 割り込み関係フラグを初期化

|  |  |  |  |
| --- | --- | --- | --- |
| 名前 | レジスタ | bit | 値 |
| UIE | 0x0F | 5 | 0 |
| UF | 0x0E | 5 | 0 |

* setIntervalTimerMode(mode)を呼び出し

### int RTC\_RV8803\_U::setIntervalTimerMode(timer\_mode\_t \* mode)

* modeの各メンバの値を該当するレジスタに書き込み

|  |  |  |  |
| --- | --- | --- | --- |
| modeのメンバ | 処理内容 | レジスタ | bit |
| pulse | 無視 |  |  |
| repeat | 無視 |  |  |
| useInteruptPin | UIEを上書き | 0x0F | 5 |
| interval | USELを上書き | 0x0D | 5 |

### int RTC\_RV8803\_U::controlIntervalTimer(uint8\_t action)

常時動作しており，外部ピンへ割り込み信号を出力すること以外は止めることができないため，RTC\_U\_UNSUPPORTEDを返す

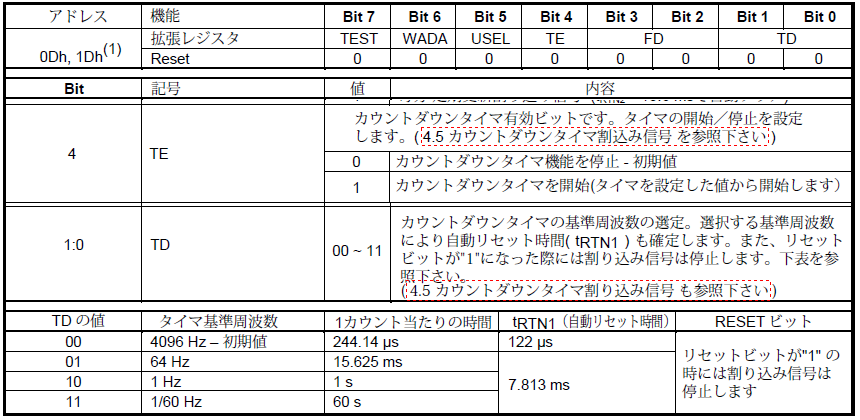
## カウントダウンタイマ

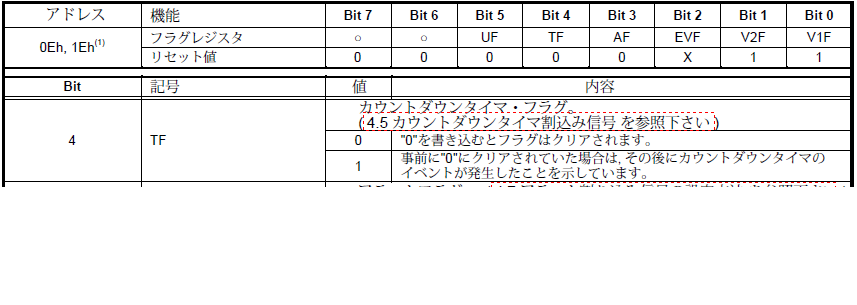
テーブル が含まれている画像

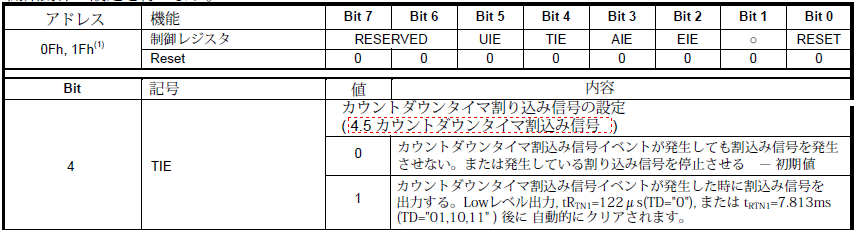
自動的に生成された説明

テーブル

自動的に生成された説明







テキスト, 手紙

自動的に生成された説明

### int RTC\_RV8803\_U::setCountdownTimer(timer\_mode\_t \* mode, uint16\_t multi)

* 割り込み関係フラグを初期化

|  |  |  |  |
| --- | --- | --- | --- |
| 名前 | レジスタ | bit | 値 |
| TE | 0x0D | 4 | 0 |
| TIE | 0x0F | 4 | 0 |
| TF | 0x0E | 4 | 0 |

* multiの値を書き込み

|  |  |  |
| --- | --- | --- |
| multiの部分 | レジスタ | bit |
| 下8bit | 0x0B | 0-7 |
| 上8bitのうち下4bit | 0x0C | 0-3 |

* setCountdownTimerMode(mode)を実行
* controlCountdownTimer(1)を実行

### int RTC\_RV8803\_U::setCountdownTimerMode(timer\_mode\_t \* mode)

* modeの各メンバの値を該当するレジスタに書き込み

|  |  |  |  |
| --- | --- | --- | --- |
| modeのメンバ | 処理内容 | レジスタ | bit |
| pulse | 無視 |  |  |
| repeat | 無視 |  |  |
| useInteruptPin | TIEを上書き | 0x0F | 4 |
| interval | TDを上書き | 0x0D | 1,0 |

### int RTC\_RV8803\_U::controlCountdownTimer(uint8\_t action)

|  |  |  |  |
| --- | --- | --- | --- |
| actionの値 | 処理 | レジスタ | bit |
| 0 | TEを0にセット | 0x0D | 4 |
| 1 | TEを1にセット | 0x0D | 4 |
| その他 | RTC\_U\_ILLEGAL\_PARAMを返す |  |  |

テーブル

自動的に生成された説明

## テスト

### 定期タイマ(分)の有効化 : setTimer(0,mode,0)

引数modeの値

|  |  |
| --- | --- |
| modeのメンバ | 値 |
| useInteruptPin | 1 |
| interval | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | レジスタ | 値 | マスク | 値 |
| 分選択 | 0x0D | xx1xxxxx | 00100000 | 00100000 |
| 定期タイマ有効化 | 0x0F | xx1xxxxx | 00100000 | 00100000 |

### 割り込みフラグをすべて消す : clearInterupt(0b1111)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xx0xxxxx | 00100000 | 0 |

### 90秒待って，割り込みフラグを確認 : checkInterupt()

* 返り値を取得し，マスクとのandをとったものが0b00100000であることを確認
* レジスタの内容が以下であることを確認

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xx1xxxxx | 00100000 | 00100000 |

### 割り込みフラグをすべて消す : clearInterupt(0b1111)

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xx0xxxxx | 00100000 | 0 |

### カウントダウンタイマを設定 : setTimer(1,mode, multi)

|  |  |  |
| --- | --- | --- |
| modeのメンバ | 値 | 意味 |
| useInteruptPin | 1 | 割り込み発生 |
| interval | 01 | 64Hz |

|  |  |  |
| --- | --- | --- |
| multi | 値 | 意味 |
| 下8bit | 0b01000000 | 64 |
| 上8bit | 0b00000010 | 512 |

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0B | 01000000 | 11111111 | 01000000 |
| 0x0C | xxxx0010 | 00001111 | 00000010 |
| 0x0D | xxx1xx01 | 00010011 | 10001 |
| 0x0F | xxx1xxxx | 00010000 | 00010000 |

### 10秒待って，カウントダウンタイマを止める : controlTimer(1,0)

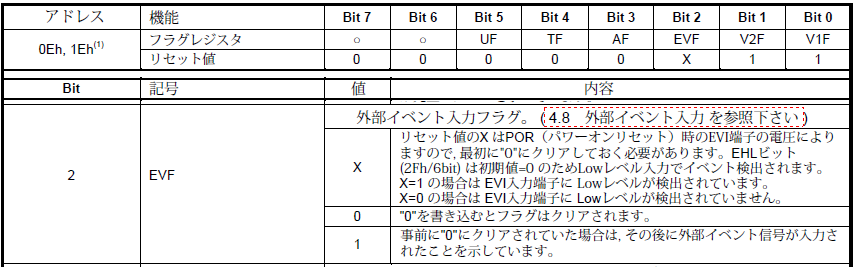
|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0D | xxx0xxxx | 00010000 | 0 |

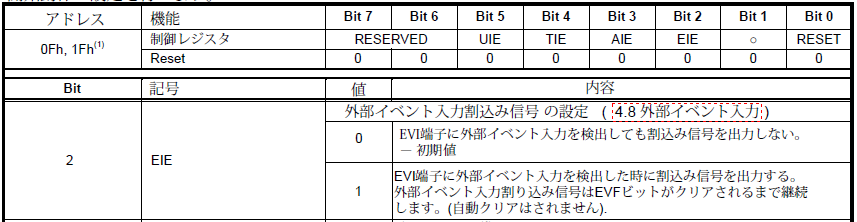
### 割り込みフラグを確認 : checkInterupt()

|  |  |  |
| --- | --- | --- |
| 返り値 | マスク | 計算値 |
| xxx1xxxx | 00010000 | 10000 |

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0E | xxx1xxxx | 00010000 | 10000 |

# 外部イベント





テーブル が含まれている画像

自動的に生成された説明

テーブル

中程度の精度で自動的に生成された説明

テーブル が含まれている画像

自動的に生成された説明

テキスト, 手紙

自動的に生成された説明

### int RTC\_RV8803\_U::setEvent(event\_mode\_t mode)

|  |  |  |  |
| --- | --- | --- | --- |
| 名前 | レジスタ | bit | 値 |
| EIE | 0x0F | 2 | 0 |
| EVF | 0x0E | 2 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| event\_mode\_tメンバ | | 名前 | レジスタ | bit |
| uint8\_t | useInteruptPin | EIE | 0x0F | 2 |
| bool | capture | ECP | 0x2F | 7 |
| uint8\_t | level | EHL | 0x2F | 6 |
| uint8\_t | filter | ET | 0x2F | 5,4 |
| bool | reset | ERST | 0x2F | 0 |

### int RTC\_RV8803\_U::getEvent(void)

|  |  |  |
| --- | --- | --- |
| 名前 | レジスタ | bit |
| EVF | 0x0E | 2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 返り値 | 内容 | レジスタ | bit | 形式 |
| 下8bit | 10m秒 | 0x20 | 0-7 | BCD |
| 上8bit | 秒 | 0x21 | 0-6 | BCD |

## テスト

### setEvent(event\_mode\_t mode)の実行

|  |  |  |
| --- | --- | --- |
| event\_mode\_tメンバ | | 値 |
| uint8\_t | useInteruptPin | 1 |
| bool | capture | true |
| uint8\_t | level | 1 |
| uint8\_t | filter | 11 |
| bool | reset | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| レジスタ | 値 | マスク | 値 |
| 0x0F | xxxxx1xx | 00000100 | 100 |
| 0x2F | 1111xxx1 | 11110001 | 11110001 |