You are viewing documentation for version: **5.9** (latest) | 5.8 | Version History

GPIO

EMLIB

Detailed Description

General Purpose Input/Output (GPIO) API.

This module contains functions to control the GPIO peripheral of Silicon Labs 32-bit MCUs and SoCs. The GPIO peripheral is used for pin configuration and direct pin manipulation and sensing as well as routing for peripheral pin connections.

Enumerations

```
enum   GPIO_DriveMode_TypeDef {
         gpioDriveModeStandard = GPIO_P_CTRL_DRIVEMODE_STANDARD,
         gpioDriveModeLowest = GPIO_P_CTRL_DRIVEMODE_LOWEST,
         gpioDriveModeHigh = GPIO_P_CTRL_DRIVEMODE_HIGH,
         gpioDriveModeLow = GPIO_P_CTRL_DRIVEMODE_LOW
       }

enum   GPIO_Mode_TypeDef {
         gpioModeDisabled = _GPIO_P_MODEL_MODE0_DISABLED,
         gpioModeInput = _GPIO_P_MODEL_MODE0_INPUT,
         gpioModeInputPull = _GPIO_P_MODEL_MODE0_INPUTPULL,
         gpioModeInputPullFilter =
       _GPIO_P_MODEL_MODE0_INPUTPULLFILTER,
         gpioModePushPull = _GPIO_P_MODEL_MODE0_PUSHPULL,
         gpioModePushPullDrive = _GPIO_P_MODEL_MODE0_PUSHPULLDRIVE,
         gpioModeWiredOr = _GPIO_P_MODEL_MODE0_WIREDOR,
         gpioModeWiredOrPullDown =
       _GPIO_P_MODEL_MODE0_WIREDORPULLDOWN,
         gpioModeWiredAnd = _GPIO_P_MODEL_MODE0_WIREDAND,
         gpioModeWiredAndFilter =
       _GPIO_P_MODEL_MODE0_WIREDANDFILTER,
         gpioModeWiredAndPullUp =
       _GPIO_P_MODEL_MODE0_WIREDANDPULLUP,
         gpioModeWiredAndPullUpFilter =
_GPIO_P_MODEL_MODE0_WIREDANDPULLUPFILTER,
  gpioModeWiredAndDrive = _GPIO_P_MODEL_MODE0_WIREDANDDRIVE,
  gpioModeWiredAndDriveFilter =
_GPIO_P_MODEL_MODE0_WIREDANDDRIVEFILTER,
  gpioModeWiredAndDrivePullUp =
_GPIO_P_MODEL_MODE0_WIREDANDDRIVEPULLUP,
  gpioModeWiredAndDrivePullUpFilter =
_GPIO_P_MODEL_MODE0_WIREDANDDRIVEPULLUPFILTER
}
```

enum GPIO_Port_TypeDef

Functions

void	GPIO_DbgLocationSet (unsigned int location) Sets the pin location of the debug pins (Serial Wire interface).
__STATIC_INLINE void	GPIO_DbgSWDClkEnable (bool enable) Enable/disable serial wire clock pin.
__STATIC_INLINE void	GPIO_DbgSWDIOEnable (bool enable) Enable/disable serial wire data I/O pin.
__STATIC_INLINE void	GPIO_DbgSWOEnable (bool enable) Enable/Disable serial wire output pin.
void	GPIO_DriveModeSet (GPIO_Port_TypeDef port, GPIO_DriveMode_TypeDef mode) Sets drive mode for a GPIO port.
void	GPIO_ExtIntConfig (GPIO_Port_TypeDef port, unsigned int pin, unsigned int intNo, bool risingEdge, bool fallingEdge, bool enable) Configure the GPIO external pin interrupt.
__STATIC_INLINE void	GPIO_InputSenseSet (uint32_t val, uint32_t mask) Enable/disable input sensing.
__STATIC_INLINE void	GPIO_IntClear (uint32_t flags) Clear one or more pending GPIO interrupts.
__STATIC_INLINE void	GPIO_IntConfig (GPIO_Port_TypeDef port, unsigned int pin, bool

	`risingEdge, bool fallingEdge, bool enable)` Configure GPIO interrupt.
`__STATIC_INLINE void`	`GPIO_IntDisable (uint32_t flags)` Disable one or more GPIO interrupts.
`__STATIC_INLINE void`	`GPIO_IntEnable (uint32_t flags)` Enable one or more GPIO interrupts.
`__STATIC_INLINE uint32_t`	`GPIO_IntGet (void)` Get pending GPIO interrupts.
`__STATIC_INLINE uint32_t`	`GPIO_IntGetEnabled (void)` Get enabled and pending GPIO interrupt flags. Useful for handling more interrupt sources in the same interrupt handler.
`__STATIC_INLINE void`	`GPIO_IntSet (uint32_t flags)` Set one or more pending GPIO interrupts from SW.
`__STATIC_INLINE void`	`GPIO_Lock (void)` Locks the GPIO configuration.
`__STATIC_INLINE unsigned int`	`GPIO_PinInGet (GPIO_Port_TypeDef port, unsigned int pin)` Read the pad value for a single pin in a GPIO port.
`__STATIC_INLINE void`	`GPIO_PinLock (GPIO_Port_TypeDef port, unsigned int pin)` Lock all GPIO configuration settings for a given pin. The lock can only be cleared by a chip reset.
`GPIO_Mode_TypeDef`	`GPIO_PinModeGet (GPIO_Port_TypeDef port, unsigned int pin)` Get the mode for a GPIO pin.
`void`	`GPIO_PinModeSet (GPIO_Port_TypeDef port, unsigned int pin, GPIO_Mode_TypeDef mode, unsigned int out)` Set the mode for a GPIO pin.
`__STATIC_INLINE void`	`GPIO_PinOutClear (GPIO_Port_TypeDef port, unsigned int pin)` Set a single pin in GPIO data out port register to 0.
`__STATIC_INLINE unsigned int`	`GPIO_PinOutGet (GPIO_Port_TypeDef port, unsigned int pin)` Get current setting for a pin in a GPIO port data out register.

`__STATIC_INLINE void`	`GPIO_PinOutSet (GPIO_Port_TypeDef port, unsigned int pin)` Set a single pin in GPIO data out register to 1.
`__STATIC_INLINE void`	`GPIO_PinOutToggle (GPIO_Port_TypeDef port, unsigned int pin)` Toggle a single pin in GPIO port data out register.
`__STATIC_INLINE uint32_t`	`GPIO_PortInGet (GPIO_Port_TypeDef port)` Read the pad values for GPIO port.
`__STATIC_INLINE void`	`GPIO_PortOutClear (GPIO_Port_TypeDef port, uint32_t pins)` Set bits in DOUT register for a port to 0.
`__STATIC_INLINE uint32_t`	`GPIO_PortOutGet (GPIO_Port_TypeDef port)` Get current setting for a GPIO port data out register.
`__STATIC_INLINE void`	`GPIO_PortOutSet (GPIO_Port_TypeDef port, uint32_t pins)` Set bits GPIO data out register to 1.
`__STATIC_INLINE void`	`GPIO_PortOutSetVal (GPIO_Port_TypeDef port, uint32_t val, uint32_t mask)` Set GPIO port data out register.
`__STATIC_INLINE void`	`GPIO_PortOutToggle (GPIO_Port_TypeDef port, uint32_t pins)` Toggle pins in GPIO port data out register.
`__STATIC_INLINE void`	`GPIO_Unlock (void)` Unlocks the GPIO configuration.

Enumeration Type Documentation

```
enum GPIO_DriveMode_TypeDef
```

GPIO drive mode.

Enumerator	

gpioDriveModeStandard	Default 6mA.
gpioDriveModeLowest	0.5 mA.
gpioDriveModeHigh	20 mA.
gpioDriveModeLow	2 mA.

Definition at line `517` of file `em_gpio.h` .

```
enum GPIO_Mode_TypeDef
```

Pin mode. For more details on each mode, refer to the reference manual.

Enumerator	
gpioModeDisabled	Input disabled. Pull-up if DOUT is set.
gpioModeInput	Input enabled. Filter if DOUT is set.
gpioModeInputPull	Input enabled. DOUT determines pull direction.
gpioModeInputPullFilter	Input enabled with filter. DOUT determines pull direction.
gpioModePushPull	Push-pull output.
gpioModePushPullDrive	Push-pull output with drive-strength set by DRIVEMODE.
gpioModeWiredOr	

	Wired-or output.
gpioModeWiredOrPullDown	Wired-or output with pull-down.
gpioModeWiredAnd	Open-drain output.
gpioModeWiredAndFilter	Open-drain output with filter.
gpioModeWiredAndPullUp	Open-drain output with pull-up.
gpioModeWiredAndPullUpFilter	Open-drain output with filter and pull-up.
gpioModeWiredAndDrive	Open-drain output with drive-strength set by DRIVEMODE.
gpioModeWiredAndDriveFilter	Open-drain output with filter and drive-strength set by DRIVEMODE.
gpioModeWiredAndDrivePullUp	Open-drain output with pull-up and drive-strength set by DRIVEMODE.
gpioModeWiredAndDrivePullUpFilter	Open-drain output with filter, pull-up and drive-strength set by DRIVEMODE.

Definition at line `552` of file `em_gpio.h` .

`enum GPIO_Port_TypeDef`

GPIO ports IDs.

Definition at line `479` of file `em_gpio.h` .

Function Documentation

`void GPIO_DbgLocationSet ( unsigned int location )`

Sets the pin location of the debug pins (Serial Wire interface).

Note

Changing the pins used for debugging uncontrolled, may result in a lockout.

Parameters

[in]	`location`	The debug pin location to use (0-3).

Definition at line `79` of file `em_gpio.c` .

Referenced by `DBG_SWOEnable()` .

`__STATIC_INLINE void GPIO_DbgSWDClkEnable ( bool enable )`

Enable/disable serial wire clock pin.

Note

Disabling SWDClk will disable the debug interface, which may result in a lockout if done early in startup (before debugger is able to halt core).

Parameters

[in]	`enable`	• false - disable serial wire clock. • true - enable serial wire clock (default after reset).

Definition at line `623` of file `em_gpio.h` .

References `BUS_RegBitWrite()` .

`__STATIC_INLINE void GPIO_DbgSWDIOEnable ( bool enable )`

Enable/disable serial wire data I/O pin.

Note

Disabling SWDClk will disable the debug interface, which may result in a lockout if done early in startup (before debugger is able to halt core).

Parameters

[in]	`enable`	• false - disable serial wire data pin. • true - enable serial wire data pin (default after reset).

Definition at line `648` of file `em_gpio.h` .

References `BUS_RegBitWrite()` .

`__STATIC_INLINE void GPIO_DbgSWOEnable ( bool enable )`

Enable/Disable serial wire output pin.

Note

Enabling this pin is not sufficient to fully enable serial wire output, which is also dependent on issues outside the GPIO module. Refer to `DBG_SWOEnable()`.

Parameters

[in]	`enable`	• false - disable serial wire viewer pin (default after reset). • true - enable serial wire viewer pin.

Definition at line `676` of file `em_gpio.h` .

References `BUS_RegBitWrite()` .

Referenced by `DBG_SWOEnable()` .

```
void GPIO_DriveModeSet    ( GPIO_Port_TypeDef         port,
                            GPIO_DriveMode_TypeDef    mode
                          )
```

Sets drive mode for a GPIO port.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`mode`	Drive mode to use for the port.

Definition at line `107` of file `em_gpio.c` .

Referenced by `CAPLESENSE_setupGPIO()` .

```
void GPIO_ExtIntConfig    ( GPIO_Port_TypeDef    port,
                            unsigned int         pin,
                            unsigned int         intNo,
                            bool                 risingEdge,
                            bool                 fallingEdge,
                            bool                 enable
                          )
```

Configure the GPIO external pin interrupt.

It is recommended to disable interrupts before configuring the GPIO pin interrupt. See `GPIO_IntDisable()` for more information.

The GPIO interrupt handler must be in place before enabling the interrupt.

Notice that any pending interrupt for the selected interrupt is cleared by this function.

Note

On series 0 devices, the pin number parameter is not used. The pin number used on these devices is hardwired to the interrupt with the same number.
On series 1 devices, the pin number can be selected freely within a group. Interrupt numbers are divided into 4 groups (intNo / 4) and valid pin number within the interrupt groups are: 0: pins 0-3 (interrupt number 0-3) 1: pins 4-7 (interrupt number 4-7) 2: pins 8-11 (interrupt number 8-11) 3: pins 12-15 (interrupt number 12-15)

Parameters

[in]	`port`	The port to associate with the `pin`.
[in]	`pin`	The pin number on the port.
[in]	`intNo`	The interrupt number to trigger.
[in]	`risingEdge`	Set to true if the interrupt will be enabled on the rising edge. Otherwise, false.
[in]	`fallingEdge`	Set to true if the interrupt will be enabled on the falling edge. Otherwise, false.
[in]	`enable`	Set to true if the interrupt will be enabled after the configuration is complete. False to leave disabled. See `GPIO_IntDisable()` and `GPIO_IntEnable()`.

Definition at line `182` of file `em_gpio.c`.

References `BUS_RegBitWrite()`, `BUS_RegMaskedWrite()`, and `GPIO_IntClear()`.

Referenced by `BOARD_alsEnableIRQ()`, `BOARD_envSensEnableIRQ()`, `BOARD_gasSensorEnableIRQ()`, `BOARD_hallSensorEnableIRQ()`, `BOARD_imuEnableIRQ()`, `BOARD_pushButtonEnableIRQ()`, and `GPIO_IntConfig()`.

```
__STATIC_INLINE void GPIO_InputSenseSet      (uint32_t   val,
                                              uint32_t   mask
                                             )
```

Enable/disable input sensing.

Disabling input sensing if not used, can save some energy consumption.

Parameters

[in]	`val`	Bitwise logic OR of one or more of: • GPIO_INSENSE_INT - interrupt input sensing. • GPIO_INSENSE_PRS - peripheral reflex system input sensing.
[in]	`mask`	Mask containing bitwise logic OR of bits similar as for `val` used to indicate which input sense options to disable/enable.

Definition at line `806` of file `em_gpio.h` .

Referenced by `ezradio_hal_GpioInit()` .

```
__STATIC_INLINE void GPIO_IntClear        ( uint32_t    flags    )
```

Clear one or more pending GPIO interrupts.

Parameters

[in]	`flags`	Bitwise logic OR of GPIO interrupt sources to clear.

Definition at line `823` of file `em_gpio.h` .

Referenced by `BOARD_alsClearIRQ()` , `BOARD_gasSensorClearIRQ()` , `BOARD_hallSensorClearIRQ()` , `BOARD_imuClearIRQ()` , `BOARD_picGetFwRevision()` , `BOARD_pushButton0ClearIRQ()` , `BOARD_pushButton1ClearIRQ()` , `GPIO_ExtIntConfig()` , and `GPIOINT_CallbackRegister()` .

```
__STATIC_INLINE void          (GPIO_Port_TypeDef  port,
GPIO_IntConfig
                               unsigned int       pin,
                               bool               risingEdge,
                               bool               fallingEdge,
                               bool               enable
                              )
```

Configure GPIO interrupt.

If reconfiguring a GPIO interrupt that is already enabled, it is generally recommended to disable it first, see GPIO_Disable().

The actual GPIO interrupt handler must be in place before enabling the interrupt.

Notice that any pending interrupt for the selected pin is cleared by this function.

`Deprecated:`

Deprecated function. New code should use `GPIO_ExtIntConfig()`.

Note

A certain pin number can only be associated with one port; i.e., if GPIO interrupt 1 is assigned to port A/pin 1, then it is not possible to use pin 1 from any other ports for interrupts. Refer to the reference manual. On devices which implement GPIO_EXTIPINSEL registers a more flexible approach is possible, refer to `GPIO_ExtIntConfig()`.

Parameters

[in]	`port`	The port to associate with `pin` .
[in]	`pin`	The pin number on the port (== GPIO EXTI interrupt number).
[in]	`risingEdge`	Set to true if interrupts will be enabled on rising edge, otherwise false.
[in]	`fallingEdge`	Set to true if interrupts will be enabled on falling edge, otherwise false.
[in]	`enable`	Set to true if interrupt will be enabled after configuration completed, false to leave disabled. See `GPIO_IntDisable()` and `GPIO_IntEnable()`.

Definition at line `1296` of file `em_gpio.h` .

References `GPIO_ExtIntConfig()` .

Referenced by `BOARD_init()` , `BOARD_pushButtonEnableIRQ()` , and `ezradio_hal_GpioInit()` .

```
__STATIC_INLINE void GPIO_IntDisable        ( uint32_t    flags   )
```

Disable one or more GPIO interrupts.

Parameters

[in]	`flags`	GPIO interrupt sources to disable.

Definition at line `839` of file `em_gpio.h` .

Referenced by `BOARD_picGetFwRevision()` , and `DisableSi114xInterrupt()` .

```
__STATIC_INLINE void GPIO_IntEnable        ( uint32_t    flags  )
```

Enable one or more GPIO interrupts.

Note

Depending on the use, a pending interrupt may already be set prior to enabling the interrupt. To ignore a pending interrupt, consider using `GPIO_IntClear()` prior to enabling the interrupt.

Parameters

[in]	`flags`	GPIO interrupt sources to enable.

Definition at line `856` of file `em_gpio.h` .

Referenced by `BOARD_picGetFwRevision()` , and `EnableSi114xInterrupt()` .

```
__STATIC_INLINE uint32_t GPIO_IntGet                ( void      )
```

Get pending GPIO interrupts.

Returns

GPIO interrupt sources pending.

Definition at line `868` of file `em_gpio.h` .

`__STATIC_INLINE uint32_t GPIO_IntGetEnabled ( void )`

Get enabled and pending GPIO interrupt flags. Useful for handling more interrupt sources in the same interrupt handler.

Note

Interrupt flags are not cleared by the use of this function.

Returns

Pending and enabled GPIO interrupt sources. The return value is the bitwise AND combination of

- the OR combination of enabled interrupt sources in GPIO_IEN register and
- the OR combination of valid interrupt flags in GPIO_IF register.

Definition at line `888` of file `em_gpio.h` .

Referenced by `GPIOINT_CallbackRegister()` .

`__STATIC_INLINE void GPIO_IntSet ( uint32_t flags )`

Set one or more pending GPIO interrupts from SW.

Parameters

[in]	`flags`	GPIO interrupt sources to set to pending.

Definition at line `907` of file `em_gpio.h` .

Referenced by `EnableSi114xInterrupt()` .

```
__STATIC_INLINE unsigned int          (GPIO_Port_TypeDef  port,
GPIO_PinInGet
                                       unsigned int       pin
```

)

Read the pad value for a single pin in a GPIO port.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin number to read.

Returns

The pin value, 0 or 1.

Definition at line `938` of file `em_gpio.h` .

References `BUS_RegBitRead()` .

Referenced by `ADC0_IRQHandler()` , `BSP_McuBoard_UsbVbusOcFlagGet()` , `EnableSi114xInterrupt()` , `ezradio_hal_NirqLevel()` , and `TOUCH_IsBusy()` .

```
__STATIC_INLINE void GPIO_PinLock    ( GPIO_Port_TypeDef    port,
                                       unsigned int         pin
                                     )
```

Lock all GPIO configuration settings for a given pin. The lock can only be cleared by a chip reset.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin number to lock.

Definition at line `957` of file `em_gpio.h` .

References `BUS_RegBitWrite()` .

```
GPIO_Mode_TypeDef GPIO_PinModeGet   ( GPIO_Port_TypeDef   port,
                                      unsigned int        pin
                                    )
```

Get the mode for a GPIO pin.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin number in the port.

Returns

The pin mode.

Definition at line `327` of file `em_gpio.c`.

```
void GPIO_PinModeSet   ( GPIO_Port_TypeDef   port,
                         unsigned int        pin,
                         GPIO_Mode_TypeDef   mode,
                         unsigned int        out
                       )
```

Set the mode for a GPIO pin.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin number in the port.
[in]	`mode`	The desired pin mode.
[in]	`out`	A value to set for the pin in the DOUT register. The DOUT setting is important for some input mode configurations to determine the pull-up/down direction.

Definition at line `278` of file `em_gpio.c`.

References `GPIO_PinOutClear()`, `GPIO_PinOutSet()`, and `gpioModeDisabled`.

Referenced by `ADC0_IRQHandler()`, `BOARD_envSensEnable()`, `BOARD_flashDeepPowerDown()`, `BOARD_gasSensorEnable()`, `BOARD_hallSensorEnable()`, `BOARD_init()`, `BSP_BccPinsEnable()`, `BSP_BusControlModeSet()`, `BSP_EbiInit()`, `BSP_initBoard()`, `BSP_McuBoard_DeInit()`, `BSP_McuBoard_Init()`, `BSP_McuBoard_UsbVbusPowerEnable()`, `CAPLESENSE_setupGPIO()`, `DBG_SWOEnable()`, `ezradio_hal_GpioInit()`, `gpioInit()`, `I2CSPM_Init()`, `ICM20648_spiInit()`, `initGpio()`, `KSZ8851SNL_SPI_Init()`, `MIC_deInit()`, `MIC_init()`, `MICROSD_Deinit()`, `MICROSD_Init()`, `MSDD_Init()`, `RETARGET_SerialEnableFlowControl()`, `RETARGET_SerialInit()`, `sl_efp_init()`, `SPI_TFT_Init()`, `TFT_DirectGPIOConfig()`, and `UTIL_shutdown()`.

```
__STATIC_INLINE void GPIO_PinOutClear  (GPIO_Port_TypeDef   port,
                                        unsigned int        pin
                                       )
```

Set a single pin in GPIO data out port register to 0.

Note

In order for the setting to take effect on the output pad, the pin must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin to set.

Definition at line `986` of file `em_gpio.h`.

References `BUS_RegMaskedClear()`.

Referenced by `BOARD_envSensEnable()`, `BOARD_flashDeepPowerDown()`, `BOARD_gasSensorEnable()`, `BOARD_gasSensorWake()`, `BOARD_hallSensorEnable()`, `BOARD_imuEnable()`, `BOARD_ledSet()`, `BOARD_micEnable()`, `BOARD_picGetFwRevision()`, `BOARD_rgbledEnable()`, `BOARD_rgbledPowerEnable()`, `BSP_McuBoard_UsbStatusLedEnable()`, `CAPT_enable()`, `ezradio_hal_ClearNsel()`, `ezradio_hal_DeassertShutdown()`, `GPIO_PinModeSet()`, `I2CSPM_Init()`, `KSZ8851SNL_SPI_SetChipSelect()`, `sl_efp_enter_em2()`, and `SPI_TFT_WriteRegister()`.

```
__STATIC_INLINE unsigned int      (GPIO_Port_TypeDef  port,
GPIO_PinOutGet
                                   unsigned int       pin
                                  )
```

Get current setting for a pin in a GPIO port data out register.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin to get setting for.

Returns

The DOUT setting for the requested pin, 0 or 1.

Definition at line `1011` of file `em_gpio.h`.

References `BUS_RegBitRead()`.

```
__STATIC_INLINE void GPIO_PinOutSet   (GPIO_Port_TypeDef  port,
                                       unsigned int       pin
                                      )
```

Set a single pin in GPIO data out register to 1.

Note

In order for the setting to take effect on the output pad, the pin must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin to set.

Definition at line `1033` of file `em_gpio.h`.

References `BUS_RegMaskedSet()`.

Referenced by `BOARD_envSensEnable()`, `BOARD_flashDeepPowerDown()`, `BOARD_gasSensorEnable()`, `BOARD_gasSensorWake()`, `BOARD_hallSensorEnable()`, `BOARD_imuEnable()`, `BOARD_ledSet()`, `BOARD_micEnable()`, `BOARD_picGetFwRevision()`, `BOARD_rgbledEnable()`, `BOARD_rgbledPowerEnable()`, `BSP_McuBoard_UsbStatusLedEnable()`, `CAPT_enable()`, `ezradio_hal_AssertShutdown()`, `ezradio_hal_SetNsel()`, `GPIO_PinModeSet()`, `I2CSPM_Init()`, `initEbiCommon()`, `KSZ8851SNL_SPI_SetChipSelect()`, `sl_efp_enter_em0()`, and `SPI_TFT_WriteRegister()`.

```
__STATIC_INLINE void GPIO_PinOutToggle (GPIO_Port_TypeDef  port,
                                        unsigned int       pin
                                       )
```

Toggle a single pin in GPIO port data out register.

Note

In order for the setting to take effect on the output pad, the pin must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pin`	The pin to toggle.

Definition at line `1060` of file `em_gpio.h`.

```
__STATIC_INLINE uint32_t GPIO_PortInGet (GPIO_Port_TypeDef port )
```

Read the pad values for GPIO port.

Parameters

[in]	`port`	The GPIO port to access.

Definition at line `1080` of file `em_gpio.h`.

Referenced by `BOARD_pushButtonGetState()`.

```
__STATIC_INLINE void GPIO_PortOutClear (GPIO_Port_TypeDef port,
                                        uint32_t          pins
                                       )
```

Set bits in DOUT register for a port to 0.

Note

In order for the setting to take effect on the output pad, the pin must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pins`	Bit mask for bits to clear in DOUT register.

Definition at line `1102` of file `em_gpio.h`.

References `BUS_RegMaskedClear()`.

```
__STATIC_INLINE uint32_t GPIO_PortOutGet (GPIO_Port_TypeDef port )
```

Get current setting for a GPIO port data out register.

Parameters

[in]	`port`	The GPIO port to access.

Returns

The data out setting for the requested port.

Definition at line `1124` of file `em_gpio.h`.

```
__STATIC_INLINE void GPIO_PortOutSet (GPIO_Port_TypeDef port,
                                      uint32_t          pins
                                     )
```

Set bits GPIO data out register to 1.

Note

In order for the setting to take effect on the respective output pads, the pins must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pins`	Bit mask for bits to set to 1 in DOUT register.

Definition at line `1146` of file `em_gpio.h`.

References `BUS_RegMaskedSet()`.

```
__STATIC_INLINE void GPIO_PortOutSetVal (GPIO_Port_TypeDef port,
                                         uint32_t          val,
                                         uint32_t          mask
                                        )
```

Set GPIO port data out register.

Note

In order for the setting to take effect on the respective output pads, the pins must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`val`	Value to write to port data out register.
[in]	`mask`	Mask indicating which bits to modify.

Definition at line `1176` of file `em_gpio.h` .

```
__STATIC_INLINE void GPIO_PortOutToggle ( GPIO_Port_TypeDef port,
                                          uint32_t          pins
                                        )
```

Toggle pins in GPIO port data out register.

Note

In order for the setting to take effect on the output pad, the pin must have been configured properly. If not, it will take effect whenever the pin has been properly configured.

Parameters

[in]	`port`	The GPIO port to access.
[in]	`pins`	Bit mask with pins to toggle.

Definition at line `1200` of file `em_gpio.h` .
