48 HAL QSPI Generic Driver

48.1 QSPI Firmware driver registers structures

48.1.1 QSPI_InitTypeDef

Data Fields

- ***uint32_t ClockPrescaler***
- ***uint32_t FifoThreshold***
- ***uint32_t SampleShifting***
- ***uint32_t FlashSize***
- ***uint32_t ChipSelectHighTime***
- ***uint32_t ClockMode***
- ***uint32_t FlashID***
- ***uint32_t DualFlash***

Field Documentation

- ***uint32_t QSPI_InitTypeDef::ClockPrescaler***
- ***uint32_t QSPI_InitTypeDef::FifoThreshold***
- ***uint32_t QSPI_InitTypeDef::SampleShifting***
- ***uint32_t QSPI_InitTypeDef::FlashSize***
- ***uint32_t QSPI_InitTypeDef::ChipSelectHighTime***
- ***uint32_t QSPI_InitTypeDef::ClockMode***
- ***uint32_t QSPI_InitTypeDef::FlashID***
- ***uint32_t QSPI_InitTypeDef::DualFlash***

48.1.2 QSPI_HandleTypeDef

Data Fields

- ***QUADSPI_TypeDef * Instance***
- ***QSPI_InitTypeDef Init***
- ***uint8_t * pTxBuffPtr***
- ***__IO uint32_t TxXferSize***
- ***__IO uint32_t TxXferCount***
- ***uint8_t * pRxBuffPtr***
- ***__IO uint32_t RxXferSize***
- ***__IO uint32_t RxXferCount***
- ***DMA_HandleTypeDef * hdma***
- ***__IO HAL_LockTypeDef Lock***
- ***__IO HAL_QSPI_StateTypeDef State***
- ***__IO uint32_t ErrorCode***
- ***uint32_t Timeout***

Field Documentation

- ***QUADSPI_TypeDef* QSPI_HandleTypeDef::Instance***
- ***QSPI_InitTypeDef QSPI_HandleTypeDef::Init***
- ***uint8_t* QSPI_HandleTypeDef::pTxBuffPtr***
- ***__IO uint32_t QSPI_HandleTypeDef::TxXferSize***
- ***__IO uint32_t QSPI_HandleTypeDef::TxXferCount***
- ***uint8_t* QSPI_HandleTypeDef::pRxBuffPtr***

- *__IO uint32_t QSPI_HandleTypeDef::RxXferSize*
- *__IO uint32_t QSPI_HandleTypeDef::RxXferCount*
- *DMA_HandleTypeDef* QSPI_HandleTypeDef::hdma*
- *__IO HAL_LockTypeDef QSPI_HandleTypeDef::Lock*
- *__IO HAL_QSPI_StateTypeDef QSPI_HandleTypeDef::State*
- *__IO uint32_t QSPI_HandleTypeDef::ErrorCode*
- *uint32_t QSPI_HandleTypeDef::Timeout*

48.1.3 QSPI_CommandTypeDef

Data Fields

- *uint32_t Instruction*
- *uint32_t Address*
- *uint32_t AlternateBytes*
- *uint32_t AddressSize*
- *uint32_t AlternateBytesSize*
- *uint32_t DummyCycles*
- *uint32_t InstructionMode*
- *uint32_t AddressMode*
- *uint32_t AlternateByteMode*
- *uint32_t DataMode*
- *uint32_t NbData*
- *uint32_t DdrMode*
- *uint32_t DdrHoldHalfCycle*
- *uint32_t SIOOMode*

Field Documentation

- *uint32_t QSPI_CommandTypeDef::Instruction*
- *uint32_t QSPI_CommandTypeDef::Address*
- *uint32_t QSPI_CommandTypeDef::AlternateBytes*
- *uint32_t QSPI_CommandTypeDef::AddressSize*
- *uint32_t QSPI_CommandTypeDef::AlternateBytesSize*
- *uint32_t QSPI_CommandTypeDef::DummyCycles*
- *uint32_t QSPI_CommandTypeDef::InstructionMode*
- *uint32_t QSPI_CommandTypeDef::AddressMode*
- *uint32_t QSPI_CommandTypeDef::AlternateByteMode*
- *uint32_t QSPI_CommandTypeDef::DataMode*
- *uint32_t QSPI_CommandTypeDef::NbData*
- *uint32_t QSPI_CommandTypeDef::DdrMode*
- *uint32_t QSPI_CommandTypeDef::DdrHoldHalfCycle*
- *uint32_t QSPI_CommandTypeDef::SIOOMode*

48.1.4 QSPI_AutoPollingTypeDef

Data Fields

- *uint32_t Match*
- *uint32_t Mask*
- *uint32_t Interval*
- *uint32_t StatusBytesSize*
- *uint32_t MatchMode*
- *uint32_t AutomaticStop*

Field Documentation

- ***uint32_t QSPI_AutoPollingTypeDef::Match***
- ***uint32_t QSPI_AutoPollingTypeDef::Mask***
- ***uint32_t QSPI_AutoPollingTypeDef::Interval***
- ***uint32_t QSPI_AutoPollingTypeDef::StatusBytesSize***
- ***uint32_t QSPI_AutoPollingTypeDef::MatchMode***
- ***uint32_t QSPI_AutoPollingTypeDef::AutomaticStop***

48.1.5 QSPI_MemoryMappedTypeDef

Data Fields

- ***uint32_t TimeOutPeriod***
- ***uint32_t TimeOutActivation***

Field Documentation

- ***uint32_t QSPI_MemoryMappedTypeDef::TimeOutPeriod***
- ***uint32_t QSPI_MemoryMappedTypeDef::TimeOutActivation***

48.2 QSPI Firmware driver API description

48.2.1 How to use this driver

Initialization

1. As prerequisite, fill in the HAL_QSPI_MspInit():
   – Enable QuadSPI clock interface with __HAL_RCC_QSPI_CLK_ENABLE().
   – Reset QuadSPI IP with __HAL_RCC_QSPI_FORCE_RESET() and __HAL_RCC_QSPI_RELEASE_RESET().
   – Enable the clocks for the QuadSPI GPIOS with __HAL_RCC_GPIOx_CLK_ENABLE().
   – Configure these QuadSPI pins in alternate mode using HAL_GPIO_Init().
   – If interrupt mode is used, enable and configure QuadSPI global interrupt with HAL_NVIC_SetPriority() and HAL_NVIC_EnableIRQ().
   – If DMA mode is used, enable the clocks for the QuadSPI DMA channel with __HAL_RCC_DMAx_CLK_ENABLE(), configure DMA with HAL_DMA_Init(), link it with QuadSPI handle using __HAL_LINKDMA(), enable and configure DMA channel global interrupt with HAL_NVIC_SetPriority() and HAL_NVIC_EnableIRQ().
2. Configure the flash size, the clock prescaler, the fifo threshold, the clock mode, the sample shifting and the CS high time using the HAL_QSPI_Init() function.

Indirect functional mode

1. Configure the command sequence using the HAL_QSPI_Command() or HAL_QSPI_Command_IT() functions:
   – Instruction phase: the mode used and if present the instruction opcode.
   – Address phase: the mode used and if present the size and the address value.
   – Alternate-bytes phase: the mode used and if present the size and the alternate bytes values.
   – Dummy-cycles phase: the number of dummy cycles (mode used is same as data phase).
   – Data phase: the mode used and if present the number of bytes.

- Double Data Rate (DDR) mode: the activation (or not) of this mode and the delay if activated.
- Sending Instruction Only Once (SIOO) mode: the activation (or not) of this mode.

2. If no data is required for the command, it is sent directly to the memory:
   - In polling mode, the output of the function is done when the transfer is complete.
   - In interrupt mode, HAL_QSPI_CmdCpltCallback() will be called when the transfer is complete.
3. For the indirect write mode, use HAL_QSPI_Transmit(), HAL_QSPI_Transmit_DMA() or HAL_QSPI_Transmit_IT() after the command configuration:
   - In polling mode, the output of the function is done when the transfer is complete.
   - In interrupt mode, HAL_QSPI_FifoThresholdCallback() will be called when the fifo threshold is reached and HAL_QSPI_TxCpltCallback() will be called when the transfer is complete.
   - In DMA mode, HAL_QSPI_TxHalfCpltCallback() will be called at the half transfer and HAL_QSPI_TxCpltCallback() will be called when the transfer is complete.
4. For the indirect read mode, use HAL_QSPI_Receive(), HAL_QSPI_Receive_DMA() or HAL_QSPI_Receive_IT() after the command configuration:
   - In polling mode, the output of the function is done when the transfer is complete.
   - In interrupt mode, HAL_QSPI_FifoThresholdCallback() will be called when the fifo threshold is reached and HAL_QSPI_RxCpltCallback() will be called when the transfer is complete.
   - In DMA mode, HAL_QSPI_RxHalfCpltCallback() will be called at the half transfer and HAL_QSPI_RxCpltCallback() will be called when the transfer is complete.

Auto-polling functional mode

1. Configure the command sequence and the auto-polling functional mode using the HAL_QSPI_AutoPolling() or HAL_QSPI_AutoPolling_IT() functions:
   - Instruction phase: the mode used and if present the instruction opcode.
   - Address phase: the mode used and if present the size and the address value.
   - Alternate-bytes phase: the mode used and if present the size and the alternate bytes values.
   - Dummy-cycles phase: the number of dummy cycles (mode used is same as data phase).
   - Data phase: the mode used.
   - Double Data Rate (DDR) mode: the activation (or not) of this mode and the delay if activated.
   - Sending Instruction Only Once (SIOO) mode: the activation (or not) of this mode.
   - The size of the status bytes, the match value, the mask used, the match mode (OR/AND), the polling interval and the automatic stop activation.
2. After the configuration:
   - In polling mode, the output of the function is done when the status match is reached. The automatic stop is activated to avoid an infinite loop.
   - In interrupt mode, HAL_QSPI_StatusMatchCallback() will be called each time the status match is reached.

Memory-mapped functional mode

1. Configure the command sequence and the memory-mapped functional mode using the HAL_QSPI_MemoryMapped() functions:
   - Instruction phase: the mode used and if present the instruction opcode.
   - Address phase: the mode used and the size.
   - Alternate-bytes phase: the mode used and if present the size and the alternate bytes values.

  - Dummy-cycles phase: the number of dummy cycles (mode used is same as data phase).
  - Data phase: the mode used.
  - Double Data Rate (DDR) mode: the activation (or not) of this mode and the delay if activated.
  - Sending Instruction Only Once (SIOO) mode: the activation (or not) of this mode.
  - The timeout activation and the timeout period.
2. After the configuration, the QuadSPI will be used as soon as an access on the AHB is done on the address range. HAL_QSPI_TimeOutCallback() will be called when the timeout expires.

Errors management and abort functionality

1. HAL_QSPI_GetError() function gives the error raised during the last operation.
2. HAL_QSPI_Abort() and HAL_QSPI_AbortIT() functions aborts any on-going operation and flushes the fifo:
  - In polling mode, the output of the function is done when the transfer complete bit is set and the busy bit cleared.
  - In interrupt mode, HAL_QSPI_AbortCpltCallback() will be called when the transfer complete bi is set.

Control functions

1. HAL_QSPI_GetState() function gives the current state of the HAL QuadSPI driver.
2. HAL_QSPI_SetTimeout() function configures the timeout value used in the driver.
3. HAL_QSPI_SetFifoThreshold() function configures the threshold on the Fifo of the QSPI IP.
4. HAL_QSPI_GetFifoThreshold() function gives the current of the Fifo's threshold

Workarounds linked to Silicon Limitation

1. Workarounds Implemented inside HAL Driver
  - Extra data written in the FIFO at the end of a read transfer

48.2.2 Initialization and Configuration functions

This subsection provides a set of functions allowing to:

- Initialize the QuadSPI.
- De-initialize the QuadSPI.

This section contains the following APIs:

- ***HAL_QSPI_Init()***
- ***HAL_QSPI_DeInit()***
- ***HAL_QSPI_MspInit()***
- ***HAL_QSPI_MspDeInit()***

48.2.3 IO operation functions

This subsection provides a set of functions allowing to:

- Handle the interrupts.
- Handle the command sequence.
- Transmit data in blocking, interrupt or DMA mode.
- Receive data in blocking, interrupt or DMA mode.
- Manage the auto-polling functional mode.
- Manage the memory-mapped functional mode.

This section contains the following APIs:

- ***HAL_QSPI_IRQHandler()***
- ***HAL_QSPI_Command()***
- ***HAL_QSPI_Command_IT()***
- ***HAL_QSPI_Transmit()***
- ***HAL_QSPI_Receive()***
- ***HAL_QSPI_Transmit_IT()***
- ***HAL_QSPI_Receive_IT()***
- ***HAL_QSPI_Transmit_DMA()***
- ***HAL_QSPI_Receive_DMA()***
- ***HAL_QSPI_AutoPolling()***
- ***HAL_QSPI_AutoPolling_IT()***
- ***HAL_QSPI_MemoryMapped()***
- ***HAL_QSPI_ErrorCallback()***
- ***HAL_QSPI_AbortCpltCallback()***
- ***HAL_QSPI_CmdCpltCallback()***
- ***HAL_QSPI_RxCpltCallback()***
- ***HAL_QSPI_TxCpltCallback()***
- ***HAL_QSPI_RxHalfCpltCallback()***
- ***HAL_QSPI_TxHalfCpltCallback()***
- ***HAL_QSPI_FifoThresholdCallback()***
- ***HAL_QSPI_StatusMatchCallback()***
- ***HAL_QSPI_TimeOutCallback()***

48.2.4 Peripheral Control and State functions

This subsection provides a set of functions allowing to:

- Check in run-time the state of the driver.
- Check the error code set during last operation.
- Abort any operation.

This section contains the following APIs:

- ***HAL_QSPI_GetState()***
- ***HAL_QSPI_GetError()***
- ***HAL_QSPI_Abort()***
- ***HAL_QSPI_Abort_IT()***
- ***HAL_QSPI_SetTimeout()***
- ***HAL_QSPI_SetFifoThreshold()***
- ***HAL_QSPI_GetFifoThreshold()***

48.2.5 Detailed description of functions

HAL_QSPI_Init

Function name	**HAL_StatusTypeDef HAL_QSPI_Init (QSPI_HandleTypeDef * hqspi)**
Function description	Initialize the QSPI mode according to the specified parameters in the QSPI_InitTypeDef and initialize the associated handle.
Parameters	• **hqspi:** QSPI handle
Return values	• **HAL:** status

HAL_QSPI_DeInit

Function name	**HAL_StatusTypeDef HAL_QSPI_DeInit (QSPI_HandleTypeDef * hqspi)**
Function description	De-Initialize the QSPI peripheral.
Parameters	• **hqspi:** QSPI handle
Return values	• **HAL:** status

HAL_QSPI_MspInit

Function name	**void HAL_QSPI_MspInit (QSPI_HandleTypeDef * hqspi)**
Function description	Initialize the QSPI MSP.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_MspDeInit

Function name	**void HAL_QSPI_MspDeInit (QSPI_HandleTypeDef * hqspi)**
Function description	DeInitialize the QSPI MSP.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_IRQHandler

Function name	**void HAL_QSPI_IRQHandler (QSPI_HandleTypeDef * hqspi)**
Function description	Handle QSPI interrupt request.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_Command

Function name	**HAL_StatusTypeDef HAL_QSPI_Command (QSPI_HandleTypeDef * hqspi, QSPI_CommandTypeDef * cmd, uint32_t Timeout)**
Function description	Set the command configuration.
Parameters	• **hqspi:** QSPI handle • **cmd:** : structure that contains the command configuration information • **Timeout:** : Timeout duration
Return values	• **HAL:** status
Notes	• This function is used only in Indirect Read or Write Modes

HAL_QSPI_Transmit

Function name	**HAL_StatusTypeDef HAL_QSPI_Transmit (QSPI_HandleTypeDef * hqspi, uint8_t * pData, uint32_t**

Timeout)

Function description	Transmit an amount of data in blocking mode.
Parameters	• **hqspi:** QSPI handle • **pData:** pointer to data buffer • **Timeout:** : Timeout duration
Return values	• **HAL:** status
Notes	• This function is used only in Indirect Write Mode

HAL_QSPI_Receive

Function name	**HAL_StatusTypeDef HAL_QSPI_Receive (QSPI_HandleTypeDef * hqspi, uint8_t * pData, uint32_t Timeout)**
Function description	Receive an amount of data in blocking mode.
Parameters	• **hqspi:** QSPI handle • **pData:** pointer to data buffer • **Timeout:** : Timeout duration
Return values	• **HAL:** status
Notes	• This function is used only in Indirect Read Mode

HAL_QSPI_Command_IT

Function name	**HAL_StatusTypeDef HAL_QSPI_Command_IT (QSPI_HandleTypeDef * hqspi, QSPI_CommandTypeDef * cmd)**
Function description	Set the command configuration in interrupt mode.
Parameters	• **hqspi:** QSPI handle • **cmd:** : structure that contains the command configuration information
Return values	• **HAL:** status
Notes	• This function is used only in Indirect Read or Write Modes

HAL_QSPI_Transmit_IT

Function name	**HAL_StatusTypeDef HAL_QSPI_Transmit_IT (QSPI_HandleTypeDef * hqspi, uint8_t * pData)**
Function description	Send an amount of data in non-blocking mode with interrupt.
Parameters	• **hqspi:** QSPI handle • **pData:** pointer to data buffer
Return values	• **HAL:** status
Notes	• This function is used only in Indirect Write Mode

HAL_QSPI_Receive_IT

Function name	**HAL_StatusTypeDef HAL_QSPI_Receive_IT**

(QSPI_HandleTypeDef * hqspi, uint8_t * pData)

Function description: Receive an amount of data in non-blocking mode with interrupt.

Parameters:
- **hqspi:** QSPI handle
- **pData:** pointer to data buffer

Return values:
- **HAL:** status

Notes:
- This function is used only in Indirect Read Mode

HAL_QSPI_Transmit_DMA

Function name: **HAL_StatusTypeDef HAL_QSPI_Transmit_DMA (QSPI_HandleTypeDef * hqspi, uint8_t * pData)**

Function description: Send an amount of data in non-blocking mode with DMA.

Parameters:
- **hqspi:** QSPI handle
- **pData:** pointer to data buffer

Return values:
- **HAL:** status

Notes:
- This function is used only in Indirect Write Mode
- If DMA peripheral access is configured as halfword, the number of data and the fifo threshold should be aligned on halfword
- If DMA peripheral access is configured as word, the number of data and the fifo threshold should be aligned on word

HAL_QSPI_Receive_DMA

Function name: **HAL_StatusTypeDef HAL_QSPI_Receive_DMA (QSPI_HandleTypeDef * hqspi, uint8_t * pData)**

Function description: Receive an amount of data in non-blocking mode with DMA.

Parameters:
- **hqspi:** QSPI handle
- **pData:** pointer to data buffer.

Return values:
- **HAL:** status

Notes:
- This function is used only in Indirect Read Mode
- If DMA peripheral access is configured as halfword, the number of data and the fifo threshold should be aligned on halfword
- If DMA peripheral access is configured as word, the number of data and the fifo threshold should be aligned on word

HAL_QSPI_AutoPolling

Function name: **HAL_StatusTypeDef HAL_QSPI_AutoPolling (QSPI_HandleTypeDef * hqspi, QSPI_CommandTypeDef * cmd, QSPI_AutoPollingTypeDef * cfg, uint32_t Timeout)**

Function description: Configure the QSPI Automatic Polling Mode in blocking mode.

Parameters:
- **hqspi:** QSPI handle
- **cmd:** structure that contains the command configuration information.

- **cfg:** structure that contains the polling configuration information.
- **Timeout:** : Timeout duration

Return values
- **HAL:** status

Notes
- This function is used only in Automatic Polling Mode

HAL_QSPI_AutoPolling_IT

Function name **HAL_StatusTypeDef HAL_QSPI_AutoPolling_IT (QSPI_HandleTypeDef * hqspi, QSPI_CommandTypeDef * cmd, QSPI_AutoPollingTypeDef * cfg)**

Function description Configure the QSPI Automatic Polling Mode in non-blocking mode.

Parameters
- **hqspi:** QSPI handle
- **cmd:** structure that contains the command configuration information.
- **cfg:** structure that contains the polling configuration information.

Return values
- **HAL:** status

Notes
- This function is used only in Automatic Polling Mode

HAL_QSPI_MemoryMapped

Function name **HAL_StatusTypeDef HAL_QSPI_MemoryMapped (QSPI_HandleTypeDef * hqspi, QSPI_CommandTypeDef * cmd, QSPI_MemoryMappedTypeDef * cfg)**

Function description Configure the Memory Mapped mode.

Parameters
- **hqspi:** QSPI handle
- **cmd:** structure that contains the command configuration information.
- **cfg:** structure that contains the memory mapped configuration information.

Return values
- **HAL:** status

Notes
- This function is used only in Memory mapped Mode

HAL_QSPI_ErrorCallback

Function name **void HAL_QSPI_ErrorCallback (QSPI_HandleTypeDef * hqspi)**

Function description Transfer Error callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_AbortCpltCallback

Function name **void HAL_QSPI_AbortCpltCallback (QSPI_HandleTypeDef * hqspi)**

Function description Abort completed callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_FifoThresholdCallback

Function name **void HAL_QSPI_FifoThresholdCallback (QSPI_HandleTypeDef * hqspi)**

Function description FIFO Threshold callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_CmdCpltCallback

Function name **void HAL_QSPI_CmdCpltCallback (QSPI_HandleTypeDef * hqspi)**

Function description Command completed callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_RxCpltCallback

Function name **void HAL_QSPI_RxCpltCallback (QSPI_HandleTypeDef * hqspi)**

Function description Rx Transfer completed callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_TxCpltCallback

Function name **void HAL_QSPI_TxCpltCallback (QSPI_HandleTypeDef * hqspi)**

Function description Tx Transfer completed callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_RxHalfCpltCallback

Function name **void HAL_QSPI_RxHalfCpltCallback (QSPI_HandleTypeDef * hqspi)**

Function description Rx Half Transfer completed callback.

Parameters
- **hqspi:** QSPI handle

Return values
- **None:**

HAL_QSPI_TxHalfCpltCallback

Function name	**void HAL_QSPI_TxHalfCpltCallback (QSPI_HandleTypeDef * hqspi)**
Function description	Tx Half Transfer completed callback.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_StatusMatchCallback

Function name	**void HAL_QSPI_StatusMatchCallback (QSPI_HandleTypeDef * hqspi)**
Function description	Status Match callback.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_TimeOutCallback

Function name	**void HAL_QSPI_TimeOutCallback (QSPI_HandleTypeDef * hqspi)**
Function description	Timeout callback.
Parameters	• **hqspi:** QSPI handle
Return values	• **None:**

HAL_QSPI_GetState

Function name	**HAL_QSPI_StateTypeDef HAL_QSPI_GetState (QSPI_HandleTypeDef * hqspi)**
Function description	Return the QSPI handle state.
Parameters	• **hqspi:** QSPI handle
Return values	• **HAL:** state

HAL_QSPI_GetError

Function name	**uint32_t HAL_QSPI_GetError (QSPI_HandleTypeDef * hqspi)**
Function description	Return the QSPI error code.
Parameters	• **hqspi:** QSPI handle
Return values	• **QSPI:** Error Code

HAL_QSPI_Abort

Function name	**HAL_StatusTypeDef HAL_QSPI_Abort (QSPI_HandleTypeDef * hqspi)**
Function description	Abort the current transmission.
Parameters	• **hqspi:** QSPI handle

Return values
- **HAL:** status

HAL_QSPI_Abort_IT

Function name **HAL_StatusTypeDef HAL_QSPI_Abort_IT (QSPI_HandleTypeDef * hqspi)**

Function description Abort the current transmission (non-blocking function)

Parameters
- **hqspi:** QSPI handle

Return values
- **HAL:** status

HAL_QSPI_SetTimeout

Function name **void HAL_QSPI_SetTimeout (QSPI_HandleTypeDef * hqspi, uint32_t Timeout)**

Function description Set QSPI timeout.

Parameters
- **hqspi:** QSPI handle.
- **Timeout:** Timeout for the QSPI memory access.

Return values
- **None:**

HAL_QSPI_SetFifoThreshold

Function name **HAL_StatusTypeDef HAL_QSPI_SetFifoThreshold (QSPI_HandleTypeDef * hqspi, uint32_t Threshold)**

Function description Set QSPI Fifo threshold.

Parameters
- **hqspi:** QSPI handle.
- **Threshold:** Threshold of the Fifo (value between 1 and 16).

Return values
- **HAL:** status

HAL_QSPI_GetFifoThreshold

Function name **uint32_t HAL_QSPI_GetFifoThreshold (QSPI_HandleTypeDef * hqspi)**

Function description Get QSPI Fifo threshold.

Parameters
- **hqspi:** QSPI handle.

Return values
- **Fifo:** threshold (value between 1 and 16)

48.3 QSPI Firmware driver defines

48.3.1 QSPI

QSPI Address Mode

QSPI_ADDRESS_NONE	No address
QSPI_ADDRESS_1_LINE	Address on a single line
QSPI_ADDRESS_2_LINES	Address on two lines
QSPI_ADDRESS_4_LINES	Address on four lines

QSPI Address Size

QSPI_ADDRESS_8_BITS	8-bit address
QSPI_ADDRESS_16_BITS	16-bit address
QSPI_ADDRESS_24_BITS	24-bit address
QSPI_ADDRESS_32_BITS	32-bit address

QSPI Alternate Bytes Mode

QSPI_ALTERNATE_BYTES_NONE	No alternate bytes
QSPI_ALTERNATE_BYTES_1_LINE	Alternate bytes on a single line
QSPI_ALTERNATE_BYTES_2_LINES	Alternate bytes on two lines
QSPI_ALTERNATE_BYTES_4_LINES	Alternate bytes on four lines

QSPI Alternate Bytes Size

QSPI_ALTERNATE_BYTES_8_BITS	8-bit alternate bytes
QSPI_ALTERNATE_BYTES_16_BITS	16-bit alternate bytes
QSPI_ALTERNATE_BYTES_24_BITS	24-bit alternate bytes
QSPI_ALTERNATE_BYTES_32_BITS	32-bit alternate bytes

QSPI Automatic Stop

QSPI_AUTOMATIC_STOP_DISABLE	AutoPolling stops only with abort or QSPI disabling
QSPI_AUTOMATIC_STOP_ENABLE	AutoPolling stops as soon as there is a match

QSPI ChipSelect High Time

QSPI_CS_HIGH_TIME_1_CYCLE	nCS stay high for at least 1 clock cycle between commands
QSPI_CS_HIGH_TIME_2_CYCLE	nCS stay high for at least 2 clock cycles between commands
QSPI_CS_HIGH_TIME_3_CYCLE	nCS stay high for at least 3 clock cycles between commands
QSPI_CS_HIGH_TIME_4_CYCLE	nCS stay high for at least 4 clock cycles between commands
QSPI_CS_HIGH_TIME_5_CYCLE	nCS stay high for at least 5 clock cycles between commands
QSPI_CS_HIGH_TIME_6_CYCLE	nCS stay high for at least 6 clock cycles between commands
QSPI_CS_HIGH_TIME_7_CYCLE	nCS stay high for at least 7 clock cycles between commands
QSPI_CS_HIGH_TIME_8_CYCLE	nCS stay high for at least 8 clock cycles between commands

QSPI Clock Mode

QSPI_CLOCK_MODE_0	Clk stays low while nCS is released
QSPI_CLOCK_MODE_3	Clk goes high while nCS is released

QSPI Data Mode

QSPI_DATA_NONE	No data
QSPI_DATA_1_LINE	Data on a single line
QSPI_DATA_2_LINES	Data on two lines
QSPI_DATA_4_LINES	Data on four lines

QSPI DDR Data Output Delay

QSPI_DDR_HHC_ANALOG_DELAY	Delay the data output using analog delay in DDR mode
QSPI_DDR_HHC_HALF_CLK_DELAY	Delay the data output by 1/2 clock cycle in DDR mode

QSPI DDR Mode

QSPI_DDR_MODE_DISABLE	Double data rate mode disabled
QSPI_DDR_MODE_ENABLE	Double data rate mode enabled

QSPI Dual Flash Mode

QSPI_DUALFLASH_ENABLE	Dual-flash mode enabled
QSPI_DUALFLASH_DISABLE	Dual-flash mode disabled

QSPI Error Code

HAL_QSPI_ERROR_NONE	No error
HAL_QSPI_ERROR_TIMEOUT	Timeout error
HAL_QSPI_ERROR_TRANSFER	Transfer error
HAL_QSPI_ERROR_DMA	DMA transfer error
HAL_QSPI_ERROR_INVALID_PARAM	Invalid parameters error

QSPI Exported Macros

__HAL_QSPI_RESET_HANDLE_STATE

Description:

- Reset QSPI handle state.

Parameters:

- __HANDLE__: QSPI handle.

Return value:

- None

__HAL_QSPI_ENABLE

Description:

- Enable the QSPI peripheral.

Parameters:

- __HANDLE__: specifies the QSPI Handle.

Return value:

- None

__HAL_QSPI_DISABLE

Description:

- Disable the QSPI peripheral.

Parameters:

- __HANDLE__: specifies the QSPI Handle.

Return value:

- None

__HAL_QSPI_ENABLE_IT

Description:

- Enable the specified QSPI interrupt.

Parameters:

- __HANDLE__: specifies the QSPI Handle.
- __INTERRUPT__: specifies the QSPI interrupt source to enable. This parameter can be one of the following values:
  - QSPI_IT_TO: QSPI Timeout interrupt
  - QSPI_IT_SM: QSPI Status match interrupt
  - QSPI_IT_FT: QSPI FIFO threshold interrupt
  - QSPI_IT_TC: QSPI Transfer complete interrupt
  - QSPI_IT_TE: QSPI Transfer error interrupt

Return value:

- None

__HAL_QSPI_DISABLE_IT

Description:

- Disable the specified QSPI interrupt.

Parameters:

- __HANDLE__: specifies the QSPI Handle.
- __INTERRUPT__: specifies the QSPI interrupt source to disable. This parameter can be one of the following values:
  - QSPI_IT_TO: QSPI Timeout interrupt
  - QSPI_IT_SM: QSPI Status match interrupt
  - QSPI_IT_FT: QSPI FIFO threshold interrupt
  - QSPI_IT_TC: QSPI Transfer complete interrupt
  - QSPI_IT_TE: QSPI Transfer error interrupt

Return value:

- None

__HAL_QSPI_GET_IT_SOURCE

Description:

- Check whether the specified QSPI interrupt source is enabled or not.

Parameters:

- __HANDLE__: specifies the QSPI Handle.
- __INTERRUPT__: specifies the QSPI

interrupt source to check. This parameter can be one of the following values:
- QSPI_IT_TO: QSPI Timeout interrupt
- QSPI_IT_SM: QSPI Status match interrupt
- QSPI_IT_FT: QSPI FIFO threshold interrupt
- QSPI_IT_TC: QSPI Transfer complete interrupt
- QSPI_IT_TE: QSPI Transfer error interrupt

Return value:

- The: new state of __INTERRUPT__ (TRUE or FALSE).

__HAL_QSPI_GET_FLAG

Description:

- Check whether the selected QSPI flag is set or not.

Parameters:

- __HANDLE__: specifies the QSPI Handle.
- __FLAG__: specifies the QSPI flag to check. This parameter can be one of the following values:
  - QSPI_FLAG_BUSY: QSPI Busy flag
  - QSPI_FLAG_TO: QSPI Timeout flag
  - QSPI_FLAG_SM: QSPI Status match flag
  - QSPI_FLAG_FT: QSPI FIFO threshold flag
  - QSPI_FLAG_TC: QSPI Transfer complete flag
  - QSPI_FLAG_TE: QSPI Transfer error flag

Return value:

- None

__HAL_QSPI_CLEAR_FLAG

Description:

- Clears the specified QSPI's flag status.

Parameters:

- __HANDLE__: specifies the QSPI Handle.
- __FLAG__: specifies the QSPI clear register flag that needs to be set This parameter can be one of the following values:
  - QSPI_FLAG_TO: QSPI Timeout flag
  - QSPI_FLAG_SM: QSPI Status match flag
  - QSPI_FLAG_TC: QSPI Transfer complete flag
  - QSPI_FLAG_TE: QSPI Transfer error

flag

Return value:

- None

QSPI Flags

QSPI_FLAG_BUSY	Busy flag: operation is ongoing
QSPI_FLAG_TO	Timeout flag: timeout occurs in memory-mapped mode
QSPI_FLAG_SM	Status match flag: received data matches in autopolling mode
QSPI_FLAG_FT	Fifo threshold flag: Fifo threshold reached or data left after read from memory is complete
QSPI_FLAG_TC	Transfer complete flag: programmed number of data have been transferred or the transfer has been aborted
QSPI_FLAG_TE	Transfer error flag: invalid address is being accessed

QSPI Flash Select

QSPI_FLASH_ID_1	FLASH 1 selected
QSPI_FLASH_ID_2	FLASH 2 selected

QSPI Instruction Mode

QSPI_INSTRUCTION_NONE	No instruction
QSPI_INSTRUCTION_1_LINE	Instruction on a single line
QSPI_INSTRUCTION_2_LINES	Instruction on two lines
QSPI_INSTRUCTION_4_LINES	Instruction on four lines

QSPI Interrupts

QSPI_IT_TO	Interrupt on the timeout flag
QSPI_IT_SM	Interrupt on the status match flag
QSPI_IT_FT	Interrupt on the fifo threshold flag
QSPI_IT_TC	Interrupt on the transfer complete flag
QSPI_IT_TE	Interrupt on the transfer error flag

QSPI Match Mode

QSPI_MATCH_MODE_AND	AND match mode between unmasked bits
QSPI_MATCH_MODE_OR	OR match mode between unmasked bits

QSPI Sample Shifting

QSPI_SAMPLE_SHIFTING_NONE	No clock cycle shift to sample data
QSPI_SAMPLE_SHIFTING_HALFCYCLE	1/2 clock cycle shift to sample data

QSPI Send Instruction Mode

QSPI_SIOO_INST_EVERY_CMD	Send instruction on every transaction
QSPI_SIOO_INST_ONLY_FIRST_CMD	Send instruction only for the first command

QSPI Timeout Activation

QSPI_TIMEOUT_COUNTER_DISABLE	Timeout counter disabled, nCS remains active

QSPI_TIMEOUT_COUNTER_ENABLE	Timeout counter enabled, nCS released when timeout expires

QSPI Timeout definition

HAL_QPSI_TIMEOUT_DEFAULT_VALUE

49 HAL PWR Generic Driver

49.1 PWR Firmware driver registers structures

49.1.1 PWR_PVDTypeDef

Data Fields

- ***uint32_t PVDLevel***
- ***uint32_t Mode***

Field Documentation

- ***uint32_t PWR_PVDTypeDef::PVDLevel***
  PVDLevel: Specifies the PVD detection level. This parameter can be a value of ***PWR_PVD_detection_level***.
- ***uint32_t PWR_PVDTypeDef::Mode***
  Mode: Specifies the operating mode for the selected pins. This parameter can be a value of ***PWR_PVD_Mode***.

49.2 PWR Firmware driver API description

49.2.1 Initialization and de-initialization functions

This section contains the following APIs:

- ***HAL_PWR_DeInit()***
- ***HAL_PWR_EnableBkUpAccess()***
- ***HAL_PWR_DisableBkUpAccess()***

49.2.2 Peripheral Control functions

PVD configuration

- The PVD is used to monitor the VDD power supply by comparing it to a threshold selected by the PVD Level (PLS[2:0] bits in PWR_CR2 register).
- PVDO flag is available to indicate if VDD/VDDA is higher or lower than the PVD threshold. This event is internally connected to the EXTI line16 and can generate an interrupt if enabled. This is done through __HAL_PVD_EXTI_ENABLE_IT() macro.
- The PVD is stopped in Standby mode.

WakeUp pin configuration

- WakeUp pins are used to wakeup the system from Standby mode or Shutdown mode. The polarity of these pins can be set to configure event detection on high level (rising edge) or low level (falling edge).

Low Power modes configuration

The devices feature 8 low-power modes: